

**Extension**

**d'ADOBE® FLASH® PROFESSIONAL CS5 & CS5.5**

## **Informations juridiques**

Vous trouverez des informations juridiques à l'adresse [http://help.adobe.com/fr\\_FR/legalnotices/index.html](http://help.adobe.com/fr_FR/legalnotices/index.html).

# Sommaire

## Chapitre 1 : Présentation

Utilisation de l'API JavaScript .....	1
Nouveautés de l'API JavaScript .....	5
Objets de l'API JavaScript .....	7
Exemples d'implémentation .....	13

## Chapitre 2 : Fonctions et méthodes de niveau supérieur

activate() .....	15
alert() .....	16
configureTool() .....	16
confirm() .....	17
deactivate() .....	18
keyDown() .....	18
keyUp() .....	19
mouseDoubleClick() .....	20
mouseDown() .....	20
mouseMove() .....	21
mouseUp() .....	22
notifySettingsChanged() .....	22
prompt() .....	23
setCursor() .....	24

## Chapitre 3 : Objet actionsPanel

actionsPanel.getClassForObject() .....	25
actionsPanel.getScriptAssistMode() .....	26
actionsPanel.getSelectedText() .....	27
actionsPanel.getText() .....	27
actionsPanel.hasSelection() .....	28
actionsPanel.replaceSelectedText() .....	28
actionsPanel.setScriptAssistMode() .....	29
actionsPanel.setSelection() .....	30
actionsPanel.setText() .....	31

## Chapitre 4 : Objet BitmapInstance

bitmapInstance.getBits() .....	32
bitmapInstance.hPixels .....	33
bitmapInstance.setBits() .....	34
bitmapInstance.vPixels .....	34

## Chapitre 5 : Objet BitmapItem

bitmapItem.allowSmoothing .....	36
bitmapItem.compressionType .....	37
bitmapItem.exportToFile() .....	37
bitmapItem.fileLastModifiedDate .....	38

bitmapItem.originalCompressionType .....	38
bitmapItem.quality .....	39
bitmapItem.sourceFileExists .....	39
bitmapItem.sourceFilesCurrent .....	40
bitmapItem.sourceFilePath .....	40
bitmapItem.useDeblocking .....	41
bitmapItem.useImportedJPEGQuality .....	41

#### **Chapitre 6 : Objet CompiledClipInstance**

compiledClipInstance.accName .....	43
compiledClipInstance.actionScript .....	44
compiledClipInstance.description .....	44
compiledClipInstance.forceSimple .....	45
compiledClipInstance.shortcut .....	45
compiledClipInstance.silent .....	45
compiledClipInstance.tabIndex .....	46

#### **Chapitre 7 : Objet compilerErrors**

compilerErrors.clear() .....	47
compilerErrors.save() .....	48

#### **Chapitre 8 : Objet ComponentInstance**

componentInstance.parameters .....	49
------------------------------------	----

#### **Chapitre 9 : Objet componentsPanel**

componentsPanel.addItemToDocument() .....	50
componentsPanel.reload() .....	51

#### **Chapitre 10 : Objet Contour**

contour.fill .....	52
contour.getHalfEdge() .....	53
contour.interior .....	54
contour.orientation .....	54

#### **Chapitre 11 : Objet Document**

document.accName .....	63
document.addDataToDocument() .....	64
document.addDataToSelection() .....	64
document.addFilter() .....	65
document.addItem() .....	66
document.addNewLine() .....	67
document.addNewOval() .....	67
document.addNewPrimitiveOval() .....	68
document.addNewPrimitiveRectangle() .....	69
document.addNewPublishProfile() .....	70
document.addNewRectangle() .....	71
document.addNewScene() .....	72
document.addNewText() .....	72

document.align()	73
document.allowScreens()	74
document.arrange()	74
document.as3AutoDeclare	75
document.as3Dialect	75
document.as3ExportFrame	76
document.as3StrictMode	76
document.as3WarningsMode	77
document.asVersion	77
document.autoLabel	78
document.backgroundColor	79
document.breakApart()	79
document.canEditSymbol()	80
document.canRevert()	80
document.canTestMovie()	81
document.canTestScene()	81
document.changeFilterOrder()	82
document.clipCopy()	83
document.clipCut()	83
document.clipPaste()	84
document.close()	85
document.convertLinesToFills()	85
document.convertToSymbol()	86
document.crop()	86
document.currentPublishProfile	87
document.currentTimeline	87
document.debugMovie()	88
document.deleteEnvelope()	89
document.deletePublishProfile()	89
document.deleteScene()	90
document.deleteSelection()	90
document.description	91
document.disableAllFilters()	91
document.disableFilter()	92
document.disableOtherFilters()	92
document.distribute()	93
document.distributeToLayers()	94
document.docClass	94
document.documentHasData()	95
document.duplicatePublishProfile()	95
document.duplicateScene()	96
document.duplicateSelection()	96
document.editScene()	97
document.enableAllFilters()	97
document.enableFilter()	98
document.enterEditMode()	99

document.exitEditMode()	99
document.exportPNG()	100
document.exportPublishProfile()	101
document.exportPublishProfileString()	101
document.exportSWF()	102
document.externalLibraryPath	103
document.forceSimple	103
document.frameRate	104
document.getAlignToDocument()	104
document.getBlendMode()	105
document.getCustomFill()	105
document.getCustomStroke()	106
document.getDataFromDocument()	107
document.getElementProperty()	107
document.getElementTextAttr()	108
document.getFilters()	109
document.getMetadata()	110
document.getMobileSettings()	110
document.getPlayerVersion()	111
document.getSelectionRect()	112
document.getTextString()	112
document.getTimeline()	113
document.getTransformationPoint()	114
document.group()	115
document.height	115
document.id	116
document.importFile()	116
document.importPublishProfile()	117
document.importPublishProfileString()	117
document.importSWF()	118
document.intersect()	119
document.library	119
document.libraryPath	120
document.livePreview	120
document.loadCuepointXML()	121
document.match()	121
document.mouseClick()	122
document.mouseDbIClk()	123
document.moveSelectedBezierPointsBy()	123
document.moveSelectionBy()	124
document.name	125
document.optimizeCurves()	125
document.path	126
document.pathURI	126
document.publish()	127
document.publishProfiles	127

document.punch()	128
document.removeAllFilters()	128
document.removeDataFromDocument()	129
document.removeDataFromSelection()	129
document.removeFilter()	130
document.renamePublishProfile()	130
document.renameScene()	131
document.reorderScene()	132
document.resetOvalObject()	132
document.resetRectangleObject()	133
document.resetTransformation()	133
document.revert()	134
document.rotate3DSelection()	134
document.rotateSelection()	135
document.save()	136
document.saveAndCompact()	136
document.scaleSelection()	137
document.screenOutline	138
document.selectAll()	138
document.selection	139
document.selectNone()	141
document.setAlignToDocument()	141
document.setBlendMode()	142
document.setCustomFill()	142
document.setCustomStroke()	143
document.setElementProperty()	144
document.setElementTextAttr()	144
document.setFillColor()	145
document.setFilterProperty()	146
document.setFilters()	146
document.setInstanceAlpha()	147
document.setInstanceBrightness()	148
document.setInstanceTint()	148
document.setMetadata()	149
document.setMobileSettings()	150
document.setOvalObjectProperty()	151
document.setPlayerVersion()	152
document.setRectangleObjectProperty()	152
document.setSelectionBounds()	153
document.setSelectionRect()	154
document.setStageVanishingPoint()	155
document.setStageViewAngle()	155
document.setStroke()	156
document.setStrokeColor()	156
document.setStrokeSize()	157
document.setStrokeStyle()	157

**Sommaire**

document.setTextRectangle()	158
document.setTextSelection()	159
document.setTextString()	159
document.setTransformationPoint()	160
document.silent	161
document.skewSelection()	161
document.smoothSelection()	162
document.sourcePath	163
document.space()	163
document.straightenSelection()	164
document.swapElement()	164
document.swapStrokeAndFill()	165
document.testMovie()	165
document.testScene()	166
document.timelines	167
document.traceBitmap()	167
document.translate3DCenter()	168
document.translate3DSelection()	168
document.transformSelection()	169
document.unGroup()	170
document.union()	170
document.unlockAllElements()	171
document.viewMatrix	171
document.width	172
document.xmlPanel()	172
document.zoomFactor	173

**Chapitre 12 : Objet drawingLayer**

drawingLayer.beginDraw()	174
drawingLayer.beginFrame()	175
drawingLayer.cubicCurveTo()	175
drawingLayer.curveTo()	176
drawingLayer.drawPath()	177
drawingLayer.endDraw()	177
drawingLayer.endFrame()	178
drawingLayer.lineTo()	178
drawingLayer.moveTo()	179
drawingLayer.newPath()	179
drawingLayer.setColor()	180
drawingLayer.setFill()	181
drawingLayer.setStroke()	181

**Chapitre 13 : Objet Edge**

edge.cubicSegmentIndex	182
edge.getControl()	183
edge.getHalfEdge()	183
edge.id	184



edge.isLine .....	184
edge.setControl() .....	185
edge.splitEdge() .....	185
edge.stroke .....	186

#### Chapitre 14 : Objet Element

element.depth .....	188
element.elementType .....	188
element.getPersistentData() .....	189
element.getTransformationPoint() .....	190
element.hasPersistentData() .....	190
element.height .....	191
element.layer .....	191
element.left .....	192
element.locked .....	192
element.matrix .....	193
element.name .....	193
element.removePersistentData() .....	194
element.rotation .....	194
element.scaleX .....	194
element.scaleY .....	195
element.selected .....	195
element.setPersistentData() .....	196
element.setTransformationPoint() .....	196
element.skewX .....	197
element.skewY .....	198
element.top .....	198
element.transformX .....	199
element.transformY .....	199
element.width .....	200
element.x .....	200
element.y .....	201

#### Chapitre 15 : Objet Fill

fill.bitmapsClipped .....	202
fill.bitmapPath .....	203
fill.color .....	203
fill.colorArray .....	204
fill.focalPoint .....	204
fill.linearRGB .....	205
fill.matrix .....	205
fill.overflow .....	206
fill.posArray .....	206
fill.style .....	207

## Chapitre 16 : Objet Filter

filter.angle	209
filter.blurX	209
filter.blurY	210
filter.brightness	210
filter.color	211
filter.contrast	211
filter.distance	212
filter.enabled	213
filter.hideObject	213
filter.highlightColor	214
filter.hue	214
filter.inner	215
filter.knockout	215
filter.name	216
filter.quality	216
filter.saturation	217
filter.shadowColor	218
filter.strength	218
filter.type	219

## Chapitre 17 : Objet flash (fl)

fl.actionsPanel	223
fl.addEventListener()	223
fl.as3PackagePaths	224
fl.browseForFileURL()	225
fl.browseForFolderURL()	226
fl.clearPublishCache()	226
fl.clipCopyString()	227
fl.closeAll()	227
fl.closeAllPlayerDocuments()	228
fl.closeDocument()	229
fl.compilerErrors	229
fl.componentsPanel	230
fl.configDirectory	230
fl.configURI	230
fl.contactSensitiveSelection	231
fl.createDocument()	231
fl.createNewDocList	232
fl.createNewDocListType	232
fl.createNewTemplateList	233
fl.documents	233
fl.drawingLayer	234
fl.exportPublishProfileString()	234
fl.externalLibraryPath	235
fl.fileExists()	235
fl.findDocumentDOM()	236

fl.findDocumentIndex()	236
fl.findObjectInDocByName()	237
fl.findObjectInDocByType()	238
fl.flexSDKPath	240
fl.getAppMemoryInfo()	240
fl.getDocumentDOM()	241
fl.getSwfPanel()	242
fl.installedPlayers	242
fl.isFontInstalled()	243
fl.languageCode	244
fl.libraryPath	244
fl.mapPlayerURL()	245
fl.Math	245
fl.mruRecentFileList	246
fl.mruRecentFileListType	246
fl.objectDrawingMode	247
fl.openDocument()	247
fl.openScript()	248
fl.outputPanel	249
fl.packagePaths	249
fl.presetPanel	249
fl.publishCacheDiskSizeMax	250
fl.publishCacheEnabled	250
fl.publishCacheMemoryEntrySizeLimit	251
fl.publishCacheMemorySizeMax	251
fl.publishDocument()	252
fl.quit()	252
fl.reloadEffects()	253
fl.reloadTools()	253
fl.removeEventListener()	254
fl.resetAS3PackagePaths()	255
fl.resetPackagePaths()	255
fl.revertDocument()	256
fl.runScript()	256
fl.saveAll()	257
fl.saveDocument()	258
fl.saveDocumentAs()	259
fl.scriptURI	259
fl.selectElement()	260
fl.selectTool()	261
fl.setActiveWindow()	262
fl.showIdleMessage()	262
fl.sourcePath	263
fl.swfPanels	263
fl.toggleBreakpoint()	264
fl.tools	264

**Sommaire**

fl.trace()	265
fl.version	265
fl.xmlui	266

**Chapitre 18 : Objet FLfile**

FLfile.copy()	268
FLfile.createFolder()	269
FLfile.exists()	270
FLfile.getAttributes()	271
FLfile.getCreationDate()	272
FLfile.getCreationDateObj()	272
FLfile.getModificationDate()	273
FLfile.getModificationDateObj()	274
FLfile.getSize()	275
FLfile.listFolder()	275
FLfile.platformPathToURI()	276
FLfile.read()	277
FLfile.remove()	278
FLfile.setAttributes()	279
FLfile.uriToPlatformPath()	280
FLfile.write()	280

**Chapitre 19 : Objet folderItem****Chapitre 20 : Objet fontItem**

fontItem.bitmap	283
fontItem.bold	284
fontItem.embeddedCharacters	284
fontItem.embedRanges	284
fontItem.embedVariantGlyphs	285
fontItem.font	287
fontItem.isDefineFont4Symbol	287
fontItem.italic	288
fontItem.size	288

**Chapitre 21 : Objet Frame**

frame.convertMotionObjectTo2D()	290
frame.convertMotionObjectTo3D()	291
frame.actionScript	292
frame.duration	292
frame.elements	292
frame.getCustomEase()	293
frame.getMotionObjectXML()	294
frame.hasCustomEase	294
frame.hasMotionPath()	295
frame.is3DMotionObject()	295
frame.isMotionObject()	296
frame.labelType	296

**Sommaire**

frame.motionTweenOrientToPath .....	297
frame.motionTweenRotate .....	297
frame.motionTweenRotateTimes .....	298
frame.motionTweenScale .....	298
frame.motionTweenSnap .....	298
frame.motionTweenSync .....	299
frame.name .....	299
frame.selectMotionPath() .....	299
frame.setCustomEase() .....	300
frame.setMotionObjectDuration() .....	301
frame.setMotionObjectXML() .....	302
frame.shapeTweenBlend .....	302
frame.soundEffect .....	303
frame.soundLibraryItem .....	303
frame.soundLoop .....	303
frame.soundLoopMode .....	304
frame.soundName .....	304
frame.soundSync .....	305
frame.startFrame .....	305
frame.tweenEasing .....	305
frame.tweenInstanceName .....	306
frame.tweenType .....	306
frame.useSingleEaseCurve .....	307
 <b>Chapitre 22 : Objet halfEdge</b>	
halfEdge.getEdge() .....	308
halfEdge.getNext() .....	309
halfEdge.getOppositeHalfEdge() .....	309
halfEdge.getPrev() .....	310
halfEdge.getVertex() .....	311
halfEdge.id .....	311
halfEdge.index .....	312
 <b>Chapitre 23 : Objet Instance</b>	
instance.instanceType .....	313
instance.libraryItem .....	314
 <b>Chapitre 24 : Objet Item</b>	
item.addData() .....	316
item.getData() .....	316
item.hasData() .....	317
item.itemType .....	317
item.linkageBaseClass .....	318
item.linkageClassName .....	318
item.linkageExportForAS .....	319
item.linkageExportForRS .....	319
item.linkageExportInFirstFrame .....	320

item.linkageIdentifier .....	320
item.linkageImportForRS .....	321
item.linkageURL .....	321
item.name .....	322
item.removeData() .....	322

### Chapitre 25 : Objet Layer

layer.color .....	323
layer.frameCount .....	324
layer.frames .....	324
layer.height .....	325
layer.layerType .....	325
layer.locked .....	326
layer.name .....	326
layer.outline .....	326
layer.parentLayer .....	327
layer.visible .....	327

### Chapitre 26 : Objet library

library.addItemToDocument() .....	330
library.addNewItem() .....	331
library.deleteItem() .....	331
library.duplicateItem() .....	332
library.editItem() .....	332
library.expandFolder() .....	333
library.findItemIndex() .....	334
library.getItemProperty() .....	334
library.getItemType() .....	335
library.getSelectedItems() .....	335
library.importEmbeddedSWF() .....	336
library.itemExists() .....	336
library.items .....	337
library.moveToFolder() .....	337
library.newFolder() .....	338
library.renameItem() .....	339
library.selectAll() .....	339
library.selectItem() .....	340
library.selectNone() .....	340
library.setItemProperty() .....	341
library.updateItem() .....	342

### Chapitre 27 : Objet Math

Math.concatMatrix() .....	343
Math.invertMatrix() .....	344
Math.pointDistance() .....	344

## Chapitre 28 : Objet Matrix

matrix.a	346
matrix.b	347
matrix.c	347
matrix.d	348
matrix.tx	348
matrix.ty	349

## Chapitre 29 : Objet outputPanel

outputPanel.clear()	350
outputPanel.save()	351
outputPanel.trace()	351

## Chapitre 30 : Objet Oval

OvalObject.closePath	353
OvalObject.endAngle	354
OvalObject.innerRadius	354
OvalObject.startAngle	355

## Chapitre 31 : Objet Parameter

parameter.category	356
parameter.insertItem()	357
parameter.listIndex	357
parameter.name	358
parameter.removeItem()	358
parameter.value	359
parameter.valueType	360
parameter.verbose	360

## Chapitre 32 : Objet Path

path.addCubicCurve()	361
path.addCurve()	362
path.addPoint()	363
path.clear()	363
path.close()	364
path.makeShape()	364
path.newContour()	365
path.nPts	366

## Chapitre 33 : Objet presetItem

presetItem.isDefault	367
presetItem.isFolder	368
presetItem.level	368
presetItem.name	369
presetItem.open	369
presetItem.path	370

### Chapitre 34 : Objet presetPanel

presetPanel.addNewItem()	372
presetPanel.applyPreset()	372
presetPanel.deleteFolder()	373
presetPanel.deleteItem()	374
presetPanel.expandFolder()	374
presetPanel.exportItem()	375
presetPanel.findItemIndex()	376
presetPanel.getSelectedItems()	377
presetPanel.importItem()	378
presetPanel.items	378
presetPanel.moveToFolder()	379
presetPanel.newFolder()	380
presetPanel.renameItem()	380
presetPanel.selectItem()	381

### Chapitre 35 : Objet Rectangle

RectangleObject.bottomLeftRadius	383
RectangleObject.bottomRightRadius	384
RectangleObject.lockFlag	384
RectangleObject.topLeftRadius	385
RectangleObject.topRightRadius	385

### Chapitre 36 : Objet Shape

shape.beginEdit()	387
shape.contours	387
shape.deleteEdge()	388
shape.edges	388
shape.endEdit()	388
shape.getCubicSegmentPoints()	389
shape.isDrawingObject	390
shape.isGroup	390
shape.isOvalObject	391
shape.isRectangleObject	391
shape.members	392
shape.numCubicSegments	392
shape.vertices	393

### Chapitre 37 : Objet SoundItem

soundItem.bitRate	395
soundItem.bits	395
soundItem.compressionType	396
soundItem.convertStereoToMono	396
soundItem.exportToFile()	397
soundItem.fileLastModifiedDate	398
soundItem.originalCompressionType	398
soundItem.quality	399



soundItem.sampleRate .....	399
soundItem.sourceFileExists .....	400
soundItem.sourceFileIsCurrent .....	400
soundItem.sourceFilePath .....	401
soundItem.useImportedMP3Quality .....	401

### Chapitre 38 : Objet Stroke

stroke.breakAtCorners .....	403
stroke.capType .....	403
stroke.color .....	404
stroke.curve .....	404
stroke.dash1 .....	405
stroke.dash2 .....	405
stroke.density .....	406
stroke.dotSize .....	406
stroke.dotSpace .....	407
stroke.hatchThickness .....	407
stroke.jiggle .....	408
stroke.joinType .....	408
stroke.length .....	408
stroke.miterLimit .....	409
stroke.pattern .....	409
stroke.rotate .....	410
stroke.scaleType .....	410
stroke.shapeFill .....	411
stroke.space .....	411
stroke.strokeHinting .....	412
stroke.style .....	412
stroke.thickness .....	413
stroke.variation .....	413
stroke.waveHeight .....	414
stroke.waveLength .....	414

### Chapitre 39 : Objet swfPanel

swfPanel.call() .....	416
swfPanel.name .....	418
swfPanel.path .....	419
swfPanel.setFocus() .....	419

### Chapitre 40 : Objet SymbolInstance

symbolInstance.accName .....	422
symbolInstance.actionScript .....	423
symbolInstance.backgroundColor .....	423
symbolInstance.bitmapRenderMode .....	424
symbolInstance.blendMode .....	424
symbolInstance.buttonTracking .....	425
symbolInstance.cacheAsBitmap .....	425

symbolInstance.colorAlphaAmount .....	426
symbolInstance.colorAlphaPercent .....	426
symbolInstance.colorBlueAmount .....	427
symbolInstance.colorBluePercent .....	427
symbolInstance.colorGreenAmount .....	427
symbolInstance.colorGreenPercent .....	428
symbolInstance.colorMode .....	428
symbolInstance.colorRedAmount .....	428
symbolInstance.colorRedPercent .....	429
symbolInstance.description .....	429
symbolInstance.filters .....	430
symbolInstance.firstFrame .....	430
symbolInstance.forceSimple .....	431
symbolInstance.loop .....	431
symbolInstance.shortcut .....	432
symbolInstance.silent .....	432
symbolInstance.symbolType .....	433
symbolInstance.tabIndex .....	433
symbolInstance.usesBackgroundColor .....	434
symbolInstance.visible .....	434

#### Chapitre 41 : Objet SymbolItem

symbolItem.convertToCompiledClip() .....	435
symbolItem.exportSWC() .....	436
symbolItem.exportSWF() .....	436
symbolItem.scalingGrid .....	437
symbolItem.scalingGridRect .....	437
symbolItem.sourceAutoUpdate .....	438
symbolItem.sourceFilePath .....	438
symbolItem.sourceLibraryName .....	439
symbolItem.symbolType .....	439
symbolItem.timeline .....	439

#### Chapitre 42 : Objet Text

text.accName .....	442
text.antiAliasSharpness .....	443
text.antiAliasThickness .....	443
text.autoExpand .....	444
text.border .....	444
text.description .....	444
text.embeddedCharacters .....	445
text.embedRanges .....	445
text.embedVariantGlyphs .....	446
text.fontRenderingMode .....	446
text.getTextAttr() .....	447
text.getTextString() .....	448
text.length .....	449

text.lineType .....	449
text.maxCharacters .....	450
text.orientation .....	450
text.renderAsHTML .....	450
text.scrollable .....	451
text.selectable .....	451
text.selectionEnd .....	452
text.selectionStart .....	452
text.setTextAttr() .....	453
text.setTextString() .....	453
text.shortcut .....	454
text.silent .....	455
text.tabIndex .....	455
text.textRuns .....	456
text.textType .....	456
text.useDeviceFonts .....	456
text.variableName .....	457

#### Chapitre 43 : Objet TextAttrs

textAttrs.aliasText .....	459
textAttrs.alignment .....	459
textAttrs.autoKern .....	459
textAttrs.bold .....	460
textAttrs.characterPosition .....	460
textAttrs.characterSpacing .....	461
textAttrs.face .....	461
textAttrs.fillColor .....	461
textAttrs.indent .....	462
textAttrs.italic .....	462
textAttrs.leftMargin .....	463
textAttrs.letterSpacing .....	463
textAttrs.lineSpacing .....	463
textAttrs.rightMargin .....	464
textAttrs.rotation .....	464
textAttrs.size .....	465
textAttrs.target .....	465
textAttrs.url .....	465

#### Chapitre 44 : Objet TextRun

textRun.textAttrs .....	467
textRun.characters .....	467

#### Chapitre 45 : Objet Timeline

timeline.addMotionGuide() .....	471
timeline.addNewLayer() .....	472
timeline.clearFrames() .....	472
timeline.clearKeyframes() .....	473

timeline.convertToBlankKeyframes()	474
timeline.convertToKeyframes()	474
timeline.copyFrames()	475
timeline.copyLayers()	476
timeline.copyMotion()	476
timeline.copyMotionAsAS3()	477
timeline.createMotionObject()	478
timeline.createMotionTween()	479
timeline.currentFrame	479
timeline.currentLayer	480
timeline.cutFrames()	480
timeline.cutLayers()	481
timeline.deleteLayer()	481
timeline.duplicateLayers()	482
timeline.expandFolder()	483
timeline.findLayerIndex()	484
timeline.frameCount	484
timeline.getFrameProperty()	485
timeline.getGuidelines()	485
timeline.getLayerProperty()	486
timeline.getSelectedFrames()	486
timeline.getSelectedLayers()	487
timeline.insertBlankKeyframe()	488
timeline.insertFrames()	488
timeline.insertKeyframe()	490
timeline.layerCount	491
timeline.layers	491
timeline.libraryItem	491
timeline.name	492
timeline.pasteFrames()	492
timeline.pasteLayers()	493
timeline.pasteMotion()	494
timeline.removeFrames()	494
timeline.removeMotionObject()	495
timeline.reorderLayer()	496
timeline.reverseFrames()	497
timeline.selectAllFrames()	497
timeline setFrameProperty()	498
timeline.setGuidelines()	499
timeline.setLayerProperty()	499
timeline.setSelectedFrames()	500
timeline.setSelectedLayers()	501
timeline.showLayerMasking()	501
timeline.startPlayback()	502
timeline.stopPlayback()	502

## Chapitre 46 : Objet ToolObj

toolObj.depth	505
toolObj.enablePIControl()	505
toolObj.iconID	506
toolObj.position	507
toolObj.setIcon()	507
toolObj.setMenuString()	508
toolObj.setOptionsFile()	508
toolObj.setPI()	509
toolObj.setToolName()	510
toolObj.setToolTip()	510
toolObj.showPIControl()	511
toolObj.showTransformHandles()	512

## Chapitre 47 : Objet Tools

tools.activeTool	514
tools.altIsDown	514
tools.constrainPoint()	514
tools.ctrlIsDown	515
tools.getKeyDown()	515
tools.mouselsDown	516
tools.penDownLoc	516
tools.penLoc	517
tools.setCreatingBbox()	517
tools.setCursor()	518
tools.shiftIsDown	518
tools.snapPoint()	519
tools.toolObjs	519

## Chapitre 48 : Objet Vertex

vertex.getHalfEdge()	520
vertex.setLocation()	521
vertex.x	521
vertex.y	522

## Chapitre 49 : Objet VideoItem

videoItem.exportToFLV()	523
videoItem.fileLastModifiedDate	524
videoItem.sourceFileExists	524
videoItem.sourceFilesCurrent	525
videoItem.sourceFilePath	525
videoItem.videoType	526

## Chapitre 50 : Objet XMLUI

xmlui.accept()	527
xmlui.cancel()	528
xmlui.get()	528
xmlui.getControlItemElement()	529

xmlui.setEnabled()	530
xmlui.getVisible()	530
xmlui.set()	531
xmlui.setControlItemElement()	531
xmlui.setControlItemElements()	532
xmlui.setEnabled()	533
xmlui.setVisible()	533

## **Chapitre 51 : Extensibilité de niveau C**

A propos de l'extensibilité	535
Intégration de fonctions en C	535
Types de données	541
API de niveau C	541

# Chapitre 1 : Présentation

En tant qu'utilisateur de l'application Adobe® Flash® Professional CS5 ou CS5.5, vous connaissez peut-être le langage Adobe® ActionScript®, qui permet de créer des scripts s'exécutant dans Adobe® Flash® Player au moment de son exécution. L'interface de programmation d'applications JavaScript (ou API JavaScript) de Flash décrite dans ce document est un outil de programmation complémentaire qui permet de créer des scripts exploitables dans l'environnement de création.

Ce document décrit les objets, méthodes et propriétés disponibles dans l'API JavaScript. Il suppose la maîtrise des commandes de l'environnement de programmation. Pour toute question au sujet du rôle d'une commande spécifique, consultez d'autres documents dans l'aide de Flash, comme *Utilisation de Flash*.

Ce document suppose également que vous connaissez la syntaxe JavaScript ou ActionScript et les concepts de base de la programmation (fonctions, paramètres et types de données).

## Utilisation de l'API JavaScript

L'API Flash JavaScript permet d'écrire des scripts destinés à effectuer plusieurs actions dans l'environnement de développement Flash (pendant que l'utilisateur exécute le programme Flash). Cette fonctionnalité est différente du langage ActionScript qui permet d'écrire des scripts destinés à effectuer des actions dans l'environnement Flash Player (pendant la lecture d'un fichier SWF). Elle diffère également des commandes JavaScript utilisables sur les pages qui s'affichent dans un navigateur Web.

Grâce à l'utilisation de l'API JavaScript, vous écrivez des scripts d'application Flash qui permettent de simplifier le processus de programmation. Par exemple, vous pouvez écrire des scripts pour automatiser des tâches répétitives ou ajouter des outils personnalisés au panneau Outils.

L'API JavaScript de Flash a été conçue sur le même modèle que les API JavaScript d'Adobe® Dreamweaver® et d'Adobe® Fireworks® (elles-mêmes basées sur l'API JavaScript de Netscape). L'API JavaScript de Flash repose sur un modèle d'objet document (DOM, Document Object Model) qui permet d'accéder aux documents Flash à l'aide d'objets créés en JavaScript. Elle comporte tous les éléments de l'API JavaScript de Netscape, ainsi que le DOM de Flash. Les objets qui sont ajoutés, ainsi que leurs méthodes et propriétés, sont décrits dans le présent document. Vous pouvez utiliser tous les éléments du langage JavaScript natif dans un script Flash, mais seuls les éléments dont l'utilisation est logique dans le contexte d'un document Flash prendront effet.

L'API JavaScript contient également un certain nombre de méthodes qui permettent d'implémenter des extensions par une combinaison de code JavaScript et C personnalisé. Pour plus d'informations, voir la section « [Extensibilité de niveau C](#) » à la page 535.

L'interpréteur JavaScript de Flash est la version 1.6 du moteur SpiderMonkey de Mozilla, disponible sur Internet à l'adresse suivante : [www.mozilla.org/js/spidermonkey/](http://www.mozilla.org/js/spidermonkey/). SpiderMonkey, l'une des deux implémentations de référence du langage JavaScript développé par Mozilla.org, est intégré au navigateur Mozilla.

SpiderMonkey implémente le langage JavaScript de base tel qu'il est défini par la spécification ECMAScript (ECMA-262) édition 3 et est conforme à tous les niveaux à celle-ci. Seuls les objets hôte propres au navigateur (qui ne figurent pas dans la spécification ECMA-262) ne sont pas pris en charge. De même, de nombreux guides de référence JavaScript établissent une distinction entre le langage JavaScript de base et le langage JavaScript côté client (associé au navigateur). Seul le langage JavaScript de base s'applique à l'interpréteur Flash JavaScript.

## Création de fichiers JSFL

Pour créer et modifier des fichiers JavaScript Flash, ou fichiers JSFL, utilisez Adobe Flash Professional ou l'éditeur de texte de votre choix. Si vous utilisez Flash, ces fichiers auront par défaut l'extension .jsfl. Pour rédiger un script, sélectionnez Fichier > Nouveau > Fichier Flash JavaScript.

Il est également possible de créer un fichier JSFL en sélectionnant des commandes dans le panneau Historique. Puis, cliquez sur le bouton Enregistrer du panneau Historique ou choisissez Enregistrer comme commande dans le menu. Le fichier de commandes est alors enregistré dans le dossier Commands avec l'extension JSFL (voir « [Enregistrement des fichiers JSFL](#) » à la page 2). Vous pouvez ensuite l'ouvrir et le modifier comme tout autre fichier script.

Le panneau Historique fournit lui aussi d'autres options utiles. Vous pouvez copier les commandes sélectionnées dans le Presse-papiers et visualiser les commandes JavaScript qui sont générées lors de l'utilisation de Flash.

### Pour copier des commandes dans le Presse-papiers à partir du panneau Historique :

- 1 Sélectionnez une ou plusieurs commandes dans le panneau Historique.
- 2 Effectuez l'une des opérations suivantes :
  - Cliquez sur le bouton Copier.
  - Dans le menu du panneau, sélectionnez Copier ces étapes.

### Pour visualiser les commandes JavaScript dans le panneau Historique :

- Dans le menu du panneau, choisissez Affichage > JavaScript dans le panneau.

## Enregistrement des fichiers JSFL

Vous pouvez rendre les scripts JSFL disponibles dans l'environnement de programmation Flash en les stockant dans un ou plusieurs sous-dossiers du dossier Configuration. Par défaut, le dossier Configuration se trouve à l'emplacement suivant :

- Windows® 7™ :  
*lecteur de démarrage\Utilisateurs\nom d'utilisateur\AppData\Local\Adobe\Flash CS5 ou CS5.5\langue\Configuration\*
- Windows® Vista™ :  
*lecteur de démarrage\Utilisateurs\nom d'utilisateur\Local Settings\Application Data\Adobe\Flash CS5 ou CS5.5\langue\Configuration\*
- Windows XP :  
*lecteur de démarrage\Documents and Settings\nom d'utilisateur\Local Settings\Application Data\Adobe\Flash CS5 ou CS5.5\langue\Configuration\*
- Mac OS® X :  
*Macintosh HD/Utilisateurs/nom d'utilisateur/Bibliothèque/Application Support/Adobe/Flash CS5 ou CS5.5/langue/Configuration/*

Pour déterminer l'emplacement du dossier Configuration, utilisez `fl.configDirectory` ou `fl.configURI`, comme dans l'exemple suivant :

```
// store directory to a variable
var configDir = fl.configDirectory;
// display directory in the Output panel
fl.trace(fl.configDirectory);
```



Dans le dossier Configuration, les dossiers suivants peuvent contenir des scripts auxquels vous pouvez accéder dans l'environnement de programmation : Behaviors (pour prendre en charge l'interface utilisateur des comportements), Commands (pour les scripts qui figurent dans le menu Commandes), JavaScript (pour les scripts utilisés par l'Assistant de script), Tools (pour les outils extensibles du panneau Outils) et WindowSWF (pour les panneaux qui s'affichent dans le menu Fenêtres). Ce document aborde les scripts utilisés pour les commandes et les outils.

Si vous modifiez un script dans le dossier Commands, il est immédiatement disponible dans Flash. Si vous modifiez un script pour un outil extensible, fermez et redémarrez Flash, ou utilisez la commande `fl.reloadTools()`.

Toutefois, si vous avez utilisé un script pour ajouter un outil extensible au panneau Outils puis que vous modifiez ce script, supprimez l'outil du panneau Outils puis ajoutez-le de nouveau, ou fermez Flash et redémarrez-le, pour que l'outil modifié soit disponible.

Pour être accessibles à partir de l'environnement de programmation, les fichiers de commandes et d'outils peuvent être stockés en deux endroits différents.

- Pour les scripts qui prennent la forme d'éléments du menu Commandes, enregistrez les fichiers JSFL dans le dossier Commands qui se trouve à l'emplacement suivant :

Système d'exploitation	Emplacement
Windows 7	<i>lecteur de démarrage</i> \Utilisateurs\ <i>nom d'utilisateur</i> \AppData\Local\Adobe\Flash CS5 ou CS5.5\langue\Configuration\Commands
Windows Vista	<i>lecteur d'amorçage</i> \Utilisateurs\ <i>nom d'utilisateur</i> \Local Settings\Application Data\Adobe\Flash CS5\langue\Configuration\Commands
Windows XP	<i>lecteur d'amorçage</i> \Documents and Settings\ <i>utilisateur</i> \Local Settings\Application Data\Adobe\FlashCS5\langue\Configuration\Commands
Mac OS X	Macintosh HD/Utilisateurs/ <i>utilisateur</i> /Bibliothèque/Application Support/Adobe/FlashCS5/ <i>langue</i> /Configuration/Commands

- Pour les scripts qui prennent la forme d'outils extensibles dans le panneau Outils, enregistrez les fichiers JSFL dans le dossier Tools qui se trouve à l'emplacement suivant :

Système d'exploitation	Emplacement
Windows 7	<i>lecteur de démarrage</i> \Utilisateurs\ <i>nom d'utilisateur</i> \AppData\Local\Adobe\Flash CS5 ou CS5.5\langue\Configuration\Tools
Windows Vista	<i>lecteur d'amorçage</i> \Utilisateurs\ <i>nom d'utilisateur</i> \Local Settings\Application Data\Adobe\Flash CS5\langue\Configuration\Tools
Windows XP	<i>lecteur d'amorçage</i> \Documents and Settings\ <i>utilisateur</i> \Local Settings\Application Data\Adobe\FlashCS5\langue\Configuration\Tools
Mac OS X	Macintosh HD/Utilisateurs/ <i>utilisateur</i> /Bibliothèque/Application Support/Adobe/FlashCS5/ <i>langue</i> /Configuration/Tools

Si d'autres fichiers doivent accompagner un fichier JSFL, par exemple des fichiers XML, enregistrez-les dans le même répertoire que ce fichier JSFL.

## Exécution des scripts

Il existe plusieurs manières d'exécuter des scripts. Les plus courantes sont développées dans la section ci-dessous.

### Pour exécuter un script affiché ou en cours de modification :

- Cliquez du bouton droit (touche Commande sous Macintosh) et choisissez Exécuter le script.

- Cliquez sur l'icône Exécuter le script de la barre d'outils de la fenêtre du script.

Cette option vous permet d'exécuter un script avant de l'avoir enregistré. Elle vous permet également d'exécuter un script même si aucun fichier FLA n'est ouvert.

**Pour exécuter un script stocké dans le dossier Commands, effectuez l'une des actions suivantes :**

- Dans l'environnement de programmation, choisissez *Commands > Nom du script*.
- Utilisez un raccourci clavier que vous aurez attribué au script. Pour attribuer un raccourci clavier, utilisez *Edition > Raccourcis clavier*, puis choisissez *Commands* du menu dessin dans la liste déroulante *Commands*. Développez le noeud *Commands* de l'arborescence afin d'afficher la liste des scripts disponibles.

**Pour exécuter un script de commandes qui ne réside pas dans le dossier Commands, effectuez l'une des actions suivantes :**

- Dans l'environnement de programmation, choisissez *Commands > Exécuter une commande*, puis sélectionnez le script à exécuter.
- Depuis un script, utilisez la commande `fl.runScript()`.
- Dans le système de fichiers, double-cliquez sur le fichier de script.

**Pour ajouter dans le panneau Outils un élément implémenté dans un fichier JSFL :**

- 1 Copiez le fichier JSFL de cet outil (ainsi que les éventuels autres fichiers associés) dans le dossier Tools (voir « [Enregistrement des fichiers JSFL](#) » à la page 2).
- 2 Choisissez *Edition > Personnaliser le panneau Outils (Windows)* ou *Flash > Personnaliser le panneau Outils (Macintosh)*.
- 3 Ajoutez le nouvel outil à la liste des outils disponibles.
- 4 Cliquez sur OK.

Vous pouvez ajouter des commandes individuelles de l'API JavaScript dans un fichier ActionScript à l'aide de la fonction `MMExecute()` décrite dans le *Guide de référence du langage ActionScript 3.0*. Toutefois, la fonction `MMExecute()` n'est exécutée que dans le contexte d'un élément d'interface utilisateur personnalisé, par exemple comme composant de l'inspecteur Propriétés, ou dans un panneau SWF au sein de l'environnement de programmation. Même si elles sont appelées par ActionScript, les commandes API JavaScript sont inopérantes dans Flash Player ou hors de l'environnement de programmation.

**Pour lancer une commande à partir d'un script ActionScript :**

- Employez la syntaxe suivante (vous pouvez concaténer plusieurs commandes en une seule chaîne) :

```
MMExecute(Javascript command string);
```

Vous pouvez également exécuter un script à partir de la ligne de commande.

**Pour exécuter un script à partir de la ligne de commande de Windows :**

- Utilisez la syntaxe suivante (en ajoutant les informations requises concernant le chemin d'accès) :

```
"flash.exe" myTestFile.jsfl
```

**Pour exécuter un script à partir de l'application « Terminal » sur Macintosh :**

- Utilisez la syntaxe suivante (en ajoutant les informations requises concernant le chemin d'accès) :

```
osascript -e 'tell application "flash" to open alias "Mac OS X:Users:user:myTestFile.jsfl" '
```

La commande `osascript` peut elle aussi exécuter du code AppleScript dans un fichier. Par exemple, vous pouvez placer le texte suivant dans un fichier nommé `myScript` :

```
tell application "flash"
open alias "Mac OS X:Users:user:myTestFile.jsfl"
end tell
```

Ensuite, pour exécuter le script, utilisez la commande suivante :

```
osascript myScript
```

## Nouveautés de l'API JavaScript

Dans Flash CS5.5 et CS5.5, certains objets, méthodes et propriétés ont été ajoutés et d'autres supprimés. Ces modifications sont résumées ci-dessous.

Si vous n'avez encore jamais utilisé l'API JavaScript, vous pouvez passer directement à la section « [Objets de l'API JavaScript](#) » à la page 7.

### Nouvelles méthodes et propriétés

Les méthodes et propriétés suivantes des objets existants constituent une nouveauté dans Flash Pro CS5 :

- [Objet Document](#)

- `document.debugMovie()`
- `document.loadCuepointXML()`

- [Objet flash \(fl\)](#)

- `fl.languageCode`
- `fl.toggleBreakpoint`

- [Objet Frame](#)

- `frame.convertMotionObjectTo2D()`
- `frame.convertMotionObjectTo3D()`
- `frame.getMotionObjectXML()`
- `frame.hasMotionPath()`
- `frame.isMotionObject()`
- `frame.is3DMotionObject()`
- `frame.selectMotionPath()`
- `frame.setMotionObjectDuration()`
- `frame.setMotionObjectXML()`
- `frame.tweenInstanceName`

- [Objet Timeline](#)

- `timeline.createMotionObject()`
- `timeline.libraryItem`

- `timeline.removeMotionObject()`
- `timeline.startPlayback`
- `timeline.stopPlayback`

Les méthodes et propriétés suivantes des objets existants constituent une nouveauté dans Flash Pro CS5.5 :

- **Objet `SymbolInstance`**
  - `symbolInstance.bitmapRenderMode`
  - `symbolInstance.backgroundColor`
  - `symbolInstance.usesBackgroundColor`
  - `symbolInstance.visible`
- **Objet `Timeline`**
  - `timeline.copyLayers()`
  - `timeline.cutLayers()`
  - `timeline.duplicateLayers()`
  - `timeline.pasteLayers()`
- **Objet `flash (fl)`**
  - `fl.getSwfPanel()`
  - `fl.installedPlayers()`
  - `fl.publishCacheEnabled`
  - `fl.publishCacheDiskSizeMax`
  - `fl.publishCacheMemorySizeMax`
  - `fl.publishCacheMemoryEntrySizeLimit`
  - `fl.clearPublishCache()`
- **Objet `swfPanel`**
  - `swfPanel.setFocus()`

## Autres modifications

Les méthodes et propriétés suivantes sont mises à jour dans Flash CS5 :

- `fl.openScript()`
- `fl.publishDocument()`
- `fontItem.embedRanges`
- `fontItem.embeddedCharacters`
- `fontItem.embedVariantGlyphs`

Les objets et les méthodes suivants ne sont plus disponibles dans Flash CS5 :

- Objet `Screen`
- Objet `ScreenOutline`
- `document.canSaveAVersion()`
- `document.revertToLastVersion()`

- `document.saveAVersion()`
- `document.synchronizeWithHeadVersion()`
- `fl.downloadLatestVersion()`
- `fl.revertDocumentToLastVersion()`
- `fl.saveAVersionOfDocument()`
- `fl.synchronizeDocumentWithHeadVersion()`

## Objets de l'API JavaScript

Cette section résume les objets disponibles dans l'API JavaScript de Flash et décrit leur utilisation. Toutes les commandes JavaScript habituelles sont également disponibles avec l'API JavaScript.

Le tableau suivant décrit brièvement chacun des objets de l'API JavaScript. Les objets apparaissent par ordre alphabétique.

Objet	Description
Objet actionsPanel	L'objet actionsPanel représente le panneau Actions actuellement affiché.
Objet BitmapInstance	L'objet BitmapInstance est une sous-classe de l'objet Instance et représente un bitmap dans une image.
Objet BitmapItem	L'objet BitmapItem fait référence à une image bitmap dans la bibliothèque d'un document. L'objet BitmapItem est une sous-classe de l'objet Item.
Objet CompiledClipInstance	L'objet CompiledClipInstance est une sous-classe de l'objet Instance.
Objet compilerErrors	L'objet compilerErrors représente le panneau Erreurs de compilation. Il s'agit d'une propriété de l'objet flash ( <code>fl.compilerErrors</code> ).
Objet ComponentInstance	L'objet ComponentInstance est une sous-classe de l'objet SymbolInstance et représente un composant dans une image.
Objet componentsPanel	L'objet componentsPanel, qui représente le panneau Composants, est une propriété de l'objet flash ( <code>fl.componentsPanel</code> ).
Objet Contour	L'objet Contour représente un tracé fermé en tirets marquant les limites d'une forme.
Objet Document	L'objet Document représente la scène.
Objet drawingLayer	L'objet drawingLayer est accessible en JavaScript sous forme d'un enfant de l'objet flash.
Objet Edge	L'objet Edge représente les limites d'une forme sur la scène.
Objet Element	Tout ce qui apparaît sur la scène est du type Element.
Objet Fill	L'objet Fill contient toutes les propriétés du paramètre de couleur de remplissage du panneau Outils ou de la forme sélectionnée.
Objet Filter	L'objet Filter contient toutes les propriétés de tous les filtres.
Objet flash (fl)	L'objet flash représente l'application Flash.
Objet FLfile	L'objet FLfile permet de créer des extensions Flash capables d'accéder aux dossiers et fichiers locaux, de les modifier et de les supprimer.
Objet folderItem	L'objet folderItem est une sous-classe de l'objet Item.
Objet fontItem	L'objet fontItem est une sous-classe de l'objet Item.

Objet	Description
Objet Frame	L'objet Frame représente les images du calque.
Objet halfEdge	Côté dirigé du bord d'un objet Shape.
Objet Instance	L'objet Instance est une sous-classe de l'objet Element.
Objet Item	L'objet Item est une classe de base abstraite.
Objet Layer	L'objet Layer représente un calque dans le scénario.
Objet library	L'objet library représente le panneau Bibliothèque.
Objet Math	L'objet Math est disponible sous forme de propriété en lecture seule de l'objet flash ( <code>fl.Math</code> ).
Objet Matrix	L'objet Matrix représente une matrice de transformation.
Objet outputPanel	L'objet outputPanel représente le panneau Sortie, qui affiche des informations facilitant le débogage, par exemple les erreurs de syntaxe. Il s'agit d'une propriété de l'objet flash ( <code>fl.outputPanel</code> ).
Objet Oval	L'objet Oval est une forme tracée à l'aide de l'outil Ovale. Pour savoir si un élément est un objet Oval, utilisez <code>shape.isOvalObject</code> .
Objet Parameter	Le type d'objet Parameter figure dans le tableau <code>screen.parameters</code> (qui correspond à la fenêtre de l'inspecteur Propriétés dans l'environnement de programmation Flash) ou dans le tableau <code>componentInstance.parameters</code> (qui correspond au composant inspecteur Propriétés dans l'environnement de programmation).
Objet Path	L'objet Path définit une séquence de segments de ligne (droite, courbe ou les deux), qui est particulièrement destinée à la création d'outils extensibles.
Objet presetItem	L'objet presetItem représente un élément (présélection ou dossier) du panneau Présélections de mouvement.
Objet presetPanel	L'objet presetPanel représente le panneau Présélections de mouvement (Fenêtre > Présélections de mouvement). Il s'agit d'une propriété de l'objet flash ( <code>fl.presetPanel</code> ).
Objet Rectangle	L'objet Rectangle est une forme tracée à l'aide de l'outil Rectangle. Pour savoir si un élément est un objet Rectangle, utilisez <code>shape.isRectangleObject</code> .
Objet Screen	L'objet Screen représente un seul écran dans une diapositive ou un formulaire.
Objet ScreenOutline	L'objet ScreenOutline représente le groupe d'écrans d'une diapositive ou d'un formulaire.
Objet Shape	L'objet Shape est une sous-classe de l'objet Element. L'objet Shape autorise un contrôle plus précis que les API de dessin pour la manipulation ou la création de formes géométriques sur la scène.
Objet SoundItem	L'objet SoundItem est une sous-classe de l'objet Item. Il représente un élément de bibliothèque qui est utilisé pour créer un son.
Objet Stroke	L'objet Stroke contient tous les paramètres d'un trait, y compris les paramètres personnalisés.
Objet swfPanel	L'objet swfPanel représente un panneau Windows SWF. Les panneaux Windows SWF sont des fichiers SWF qui implémentent des applications exécutables dans l'environnement de programmation Flash. Le tableau d'objets swfPanel est une propriété de l'objet flash ( <code>fl.swfPanels</code> ).
Objet SymbolInstance	L'objet SymbolInstance est une sous-classe de l'objet Instance et représente un symbole dans une image.
Objet SymbolItem	L'objet SymbolItem est une sous-classe de l'objet Item.
Objet Text	L'objet Text représente un élément de texte unique dans un document.

Objet	Description
Objet TextAttrs	L'objet TextAttrs contient toutes les propriétés de texte pouvant être appliquées à une sous-sélection. Cet objet est une sous-classe de l'objet Texte.
Objet TextRun	L'objet TextRun représente un flux de caractères dont les attributs correspondent à toutes les propriétés de l'objet TextAttrs.
Objet Timeline	L'objet Timeline représente le scénario Flash, qui est accessible, pour le document actuel, à l'aide de <code>fl.getDocumentDOM().getTimeline()</code> .
Objet ToolObj	L'objet toolObj représente un outil individuel dans le panneau Outils.
Objet Tools	L'objet Tools est accessible à partir de l'objet flash ( <code>fl.tools</code> ).
Objet Vertex	L'objet Vertex est la partie de la structure des données d'une forme qui contient les coordonnées.
Objet VideoItem	L'objet VideoItem est une sous-classe de l'objet Item.
Objet XMLUI	L'objet XMLUI permet de lire et définir les propriétés d'une boîte de dialogue XMLUI, et d'en accepter ou d'en annuler une.

## Modèle d'objet document (DOM) de Flash

Le DOM de l'API JavaScript de Flash se compose d'un ensemble de fonctions de niveau supérieur (voir « [Fonctions et méthodes de niveau supérieur](#) » à la page 15) et de deux objets de haut niveau : FLfile et flash (fl). La disponibilité de chaque objet dans les scripts est assurée, puisqu'il existe dès l'ouverture de l'environnement de programmation de Flash. Pour plus d'informations, voir les sections [Objet FLfile](#) et [Objet flash \(fl\)](#).

La référence à l'objet flash peut être `flash` ou `fl`. Par exemple, pour fermer tous les fichiers FLA ouverts, vous pouvez utiliser l'une des instructions suivantes :

```
flash.closeAll();  
fl.closeAll();
```

L'objet flash contient les *objets enfant* suivants :

Objet	Accès
Objet actionsPanel	Utilisez <code>fl.actionsPanel</code> pour accéder à l'objet actionsPanel. Cet objet correspond au panneau Actions de l'environnement de programmation Flash.
Objet compilerErrors	Utilisez <code>fl.compilerErrors</code> pour accéder à l'objet compilerErrors. Cet objet correspond au panneau Erreurs de compilation dans l'environnement de programmation Flash.
Objet componentsPanel	Utilisez <code>fl.componentsPanel</code> pour accéder à l'objet componentsPanel. Cet objet correspond au panneau Composants de l'environnement de programmation Flash.
Objet Document	Utilisez <code>fl.documents</code> pour récupérer dans un tableau la liste de tous les documents ouverts ; utilisez <code>fl.documents[index]</code> pour accéder à un document particulier ; et utilisez <code>fl.getDocumentDOM()</code> pour accéder au document actif (celui qui a le focus).
Objet drawingLayer	Utilisez <code>fl.drawingLayer</code> pour accéder à l'objet drawingLayer.
Objet Math	Utilisez <code>fl.Math</code> pour accéder à l'objet Math.
Objet outputPanel	Utilisez <code>fl.outputPanel</code> pour accéder à l'objet outputPanel. Cet objet correspond au panneau Sortie de l'environnement de programmation Flash.
Objet presetPanel	Utilisez <code>fl.presetPanel</code> pour accéder à l'objet presetPanel. Cet objet correspond au panneau Présélections de mouvement (Fenêtre > Présélections de mouvement).

Objet	Accès
Objet swfPanel	Utilisez <code>fl.swfPanels</code> pour accéder à un tableau d'objets swfPanel. Ces objets correspondent aux panneaux Window SWF.
Objet Tools	Utilisez <code>fl.tools</code> pour accéder à un tableau d'objets Tools.
Objet XMLUI	Utilisez <code>fl.xmlui</code> pour accéder à un objet de l'interface utilisateur XML (XMLUI). L'objet XMLUI permet de lire et définir les propriétés d'une boîte de dialogue de XMLUI.

## Objet Document

La propriété `fl.documents` est l'une des plus importantes de l'objet flash de niveau supérieur. Cette propriété contient un tableau d'objets Document, représentant chacun l'un des fichiers FLA actuellement ouverts dans l'environnement de programmation. Les propriétés de chaque objet Document représentent la plupart des éléments que peut contenir un fichier FLA. Une grande partie du DOM est donc composée d'objets et de propriétés enfant de l'objet Document. Pour plus d'informations, voir la section [Objet Document](#).

Ainsi, pour vous référer au premier document ouvert, vous pouvez utiliser l'instruction `flash.documents[0]` ou `fl.documents[0]`. Il s'agit du premier document Flash qui a été ouvert dans l'environnement de programmation au cours de la session actuelle. Lors de la fermeture du premier document, les index des autres documents ouverts sont décrémentés.

Pour connaître l'index d'un document particulier, utilisez `flash.findDocumentIndex(nomDuDocument)` ou `fl.findDocumentIndex(nomDuDocument)`. Voir [fl.findDocumentIndex\(\)](#).

Pour accéder au document qui a le focus, utilisez l'instruction `flash.getDocumentDOM()` ou `fl.getDocumentDOM()`. Voir [fl.getDocumentDOM\(\)](#). Cette dernière est la syntaxe utilisée dans la plupart des exemples de ce guide.

Pour trouver un document particulier dans le tableau `fl.documents`, créez une boucle qui testera la propriété `document.name` de chaque document. Voir [fl.documents](#) et [document.name](#).

Vous pouvez accéder à tous les objets du DOM qui ne figurent pas dans la liste ci-dessus (voir « [Modèle d'objet document \(DOM\) de Flash](#) » à la page 9) à l'aide de l'objet Document. Par exemple, pour accéder à la bibliothèque d'un document, utilisez la propriété `document.library`, qui renvoie un objet library :

```
fl.getDocumentDOM().library
```

Pour accéder au tableau des éléments présents dans la bibliothèque, utilisez la propriété `library.items` ; chaque élément de ce tableau est un objet Item :

```
fl.getDocumentDOM().library.items
```

Pour accéder à un élément spécifique de la bibliothèque, indiquez un membre du tableau `library.items` :

```
fl.getDocumentDOM().library.items[0]
```

Autrement dit, l'objet Item est un enfant de l'objet library, qui est lui-même un enfant de l'objet Document. Pour plus d'informations, voir [document.library](#), [Objet library](#), [library.items](#) et [Objet Item](#).

## Désignation de la cible d'une action

Sauf indication contraire, les méthodes affectent la sélection ou l'élément qui a le focus. Par exemple, le script ci-dessous double la taille de la sélection actuelle, car aucun objet précis n'est spécifié :

```
fl.getDocumentDOM().scaleSelection(2, 2);
```



Il peut être nécessaire de définir comme cible d'une action l'élément actuellement sélectionné dans le document Flash. Pour ce faire, utilisez le tableau renvoyé par la propriété `document.selection` (voir [document.selection](#)). Le premier élément de ce tableau représente l'élément actuellement sélectionné, comme le montre l'exemple ci-dessous :

```
var accDescription = fl.getDocumentDOM().selection[0].description;
```

Le script ci-dessous double la taille du premier élément de la scène contenu dans le tableau d'éléments, et non pas l'élément sélectionné :

```
var element = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
if (element) {
    element.width = element.width*2;
    element.height = element.height*2;
}
```

Il est également possible de parcourir en boucle la liste de tous les éléments de la scène pour augmenter leur largeur et leur hauteur d'un pourcentage spécifique, comme dans l'exemple suivant :

```
var elementArray =
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements;
for (var i=0; i < elementArray.length; i++) {
    var offset = 10;
    elementArray[i].width += offset;
    elementArray[i].height += offset;
}
```

## Résumé de la structure du DOM

La liste suivante présente la structure arborescente du DOM. Les chiffres qui apparaissent au début de chaque ligne représentent le niveau de chaque objet. Par exemple, un objet précédé de « 03 » est un enfant de l'objet de niveau « 02 » qui le précède immédiatement, et celui-ci est lui-même un enfant de l'objet de niveau « 01 » qui le précède.

Dans certains cas, il est possible de référencer un objet en spécifiant une propriété de son objet parent. Par exemple, la propriété `document.timelines` contient un tableau d'objets Timeline. Ces propriétés sont indiquées dans l'arborescence ci-dessous.

Certains objets sont des sous-classes (et non pas des enfants) d'autres objets. Un objet qui est une sous-classe d'un autre possède ses propres méthodes et/ou propriétés, outre les méthodes et propriétés de l'objet parent dont il est dérivé (superclasse). Dans la hiérarchie des objets, les sous-classes sont au même niveau que leur superclasse. Par exemple, l'objet `Item` est une superclasse de l'objet `BitmapItem`. Ces relations sont indiquées dans l'arborescence ci-dessous :

- 01 Top-Level Functions and Methods
- 01 FLfile object
- 01 flash object (fl)
  - 02 compilerErrors object
  - 02 componentsPanel object
  - 02 Document object (fl.documents array)
    - 03 Filter object
    - 03 Matrix object
    - 03 Fill object
    - 03 Stroke object
    - 03 library object
      - 04 Item object (library.items array)
      - 04 BitmapItem object (subclass of Item object)
      - 04 folderItem object (subclass of Item object)
      - 04 fontItem object (subclass of Item object)
      - 04 SoundItem object (subclass of Item object)
      - 04 SymbolItem object (subclass of Item object)
      - 04 VideoItem object (subclass of Item object)
  - 03 Timeline object (document.timelines array)
    - 04 Layer object (timeline.layers array)
      - 05 Frame object (layer.frames array)
        - 06 Element object (frame.elements array)
          - 07 Matrix object (element.matrix)
        - 06 Instance object (abstract class, subclass of Element object)
        - 06 BitmapInstance object (subclass of Instance object)
        - 06 CompiledClipInstance object (subclass of Instance object)
        - 06 ComponentInstance object (subclass of SymbolInstance object)
          - 07 Parameter object (componentInstance.parameters array)
        - 06 SymbolInstance object (subclass of Instance object)
        - 06 Text object (subclass of Element object)
          - 07 TextRun object (text.textRuns array)
            - 08 TextAttrs object (textRun.textAttrs array)
      - 06 Shape object (subclass of Element object)
        - 07 Oval object
        - 07 Rectangle object
        - 07 Contour object (shape.contours array)
          - 08 HalfEdge object
          - 09 Vertex object
          - 09 Edge object

```
07 Edge object (shape.edges array)
08 HalfEdge object
09 Vertex object
09 Edge object
07 Vertex object (shape.vertices array)
08 HalfEdge object
09 Vertex object
09 Edge object
05 Parameter object (screen.parameters array)
02 drawingLayer object
03 Path object
04 Contour object
02 Math object
02 outputPanel object
02 presetPanel object
03 presetItem object (presetPanel.items array)
02 swfPanel object
02 Tools object (fl.tools array)
03 ToolObj object (tools.toolObjs array)
02 XMLUI object
```

## Exemples d'implémentation

Plusieurs exemples d'implémentations JSFL sont disponibles pour Adobe Flash Professional CS5 et CS5.5. Vous pouvez consulter et installer ces fichiers pour vous familiariser avec l'API JavaScript. Les exemples sont stockés dans un dossier Samples/ExtendingFlash dans le fichier Samples.zip à l'adresse [www.adobe.com/go/learn\\_fl\\_samples\\_fr](http://www.adobe.com/go/learn_fl_samples_fr).

### Exemple de commande Shape

Un exemple de script d'API JavaScript appelé Shape.jsfl est disponible dans le dossier ExtendingFlash/Shape (voir « Exemples d'implémentation » ci-dessus). Ce script affiche des informations sur les contours de la forme dans le panneau Sortie.

#### Pour installer et exécuter le script Shape :

- 1 Copiez le fichier Shape.jsfl dans le dossier Configuration/Commands (voir « [Enregistrement des fichiers JSFL](#) » à la page 2).
- 2 Dans un document Flash (fichier FLA), sélectionnez un objet shape.
- 3 Sélectionnez Commandes > Forme pour exécuter le script.

### Exemple de commande de filtres get et set

Un exemple de script d'API JavaScript nommé filtersGetSet.jsfl est disponible dans le dossier ExtendingFlash/filtersGetSet (voir " Exemples d'implémentation " ci-dessus). Ce script ajoute des filtres à un objet sélectionné et affiche des informations sur les filtres en cours d'ajout dans le panneau Sortie.

#### Pour installer et exécuter le script filtersGetSet :

- 1 Copiez le fichier filtersGetSet.jsfl dans le dossier Configuration/Commands (voir « [Enregistrement des fichiers JSFL](#) » à la page 2).
- 2 Dans un document Flash (fichier FLA), sélectionnez un objet de type texte, clip ou bouton.

- 3 Sélectionnez Commandes > filtersGetSet pour exécuter le script.

## Exemple d'outil PolyStar

Un exemple de script d'API JavaScript nommé PolyStar.jsfl est disponible dans le dossier ExtendingFlash/PolyStar (voir " Exemples d'implémentation " ci-dessus).

Le fichier PolyStar.jsfl duplique l'outil Polygone intégré au panneau Outils de Flash. Le script illustre la construction de l'outil Polygone à l'aide de l'API JavaScript et comprend des commentaires détaillés sur l'action du code. La lecture attentive de ce fichier permet de mieux comprendre le fonctionnement de l'API JavaScript. Consultez également le fichier PolyStar.xml dans le répertoire Tools pour vous familiariser avec la compilation de vos propres outils.

## Exemple du panneau de tracé de bitmap

Les fichiers nommés TraceBitmap.fla et TraceBitmap.swf se trouvent dans le dossier ExtendingFlash/TraceBitmapPanel (voir « Exemples d'implémentation » ci-dessus). Ces fichiers illustrent la manière de concevoir et créer un panneau pour contrôler les fonctions de Flash. Ils montrent également l'emploi de la fonction `MMExecute()` pour appeler des commandes JavaScript depuis un script ActionScript.

### Pour exécuter l'exemple TraceBitmap :

- 1 Si Flash est en cours d'exécution, fermez-le.
- 2 Copiez le fichier TraceBitmap.swf dans le dossier WindowSWF, un sous-dossier du dossier Configuration (voir « [Enregistrement des fichiers JSFL](#) » à la page 2). Par exemple, sous Windows XP, le chemin du dossier est *lecteur d'amorçage*\Documents and Settings\utilisateur\Local Settings\Application Data\Adobe\Flash CS5\langue\Configuration\WindowSWF.
- 3 Démarrez Flash.
- 4 Créez ou ouvrez un document Flash (fichier FLA), puis importez-y un fichier bitmap ou une image JPEG.  
Vous pouvez utiliser le fichier flower.jpg qui se trouve dans le dossier TraceBitmapPanel, ou toute autre image de votre choix.
- 5 Lorsque l'image importée est sélectionnée, choisissez Fenêtre > Autres panneaux > TraceBitmap.
- 6 Cliquez sur Envoyer.  
L'image est convertie en un groupe de formes.

## Exemple de DLL

Un exemple de déploiement d'une DLL se trouve dans le dossier ExtendingFlash/dllSampleComputeSum (voir « Exemples d'implémentation » ci-dessus). Pour plus d'informations sur le développement des DLL, voir la section « [Extensibilité de niveau C](#) » à la page 535.

# Chapitre 2 : Fonctions et méthodes de niveau supérieur

## A propos de cette section

Cette section décrit les fonctions et les méthodes de niveau supérieur disponibles lorsque vous utilisez l'API JavaScript d'Adobe Flash. Pour plus d'informations sur l'emplacement de stockage des fichiers de l'API JavaScript, voir la section « [Enregistrement des fichiers JSFL](#) » à la page 2.

## Méthodes globales

Vous pouvez appeler les méthodes suivantes à partir de tout script de l'API JavaScript :

```
alert()
confirm()
prompt()
```

## Outils extensibles

Les fonctions suivantes sont disponibles dans les scripts qui créent des outils extensibles :

```
activate()
configureTool()
deactivate()
keyDown()
keyUp()
mouseDoubleClick()
mouseDown()
mouseMove()
mouseUp()
notifySettingsChanged()
setCursor()
```

## activate()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function activate() {
    // statements
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Fonction appelée lorsque l'outil extensible est activé (c'est-à-dire lorsqu'il est sélectionné dans le panneau Outils). Cette fonction vous permet d'exécuter toute tâche d'initialisation requise par l'outil.

### Exemple

L'exemple suivant définit la valeur de `tools.activeTool` lorsque l'outil extensible est sélectionné dans le panneau Outils :

```
function activate() {  
    var theTool = fl.tools.activeTool  
}
```

### Voir aussi

`tools.activeTool`

## alert()

### Disponibilité

Flash MX 2004.

### Utilisation

```
alert ( alertText )
```

### Paramètres

**alertText** Chaîne qui spécifie le message devant apparaître dans la boîte de dialogue d'alerte.

### Valeur renvoyée

Aucune.

### Description

Méthode qui affiche une chaîne et un bouton OK dans une boîte de dialogue modale.

### Exemple

L'exemple suivant affiche le message Process Complete (Traitement terminé) dans une boîte de dialogue d'alerte.

```
alert("Process Complete");
```

### Voir aussi

`confirm()`, `prompt()`

## configureTool()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function configureTool() {  
    // statements  
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Fonction appelée à l'ouverture de Flash et lorsque l'outil extensible est chargé dans le panneau Outils. Elle permet de définir toutes les informations dont Flash a besoin à propos de cet outil.

### Exemple

Les exemples suivants montrent deux implémentations possibles de cette fonction :

```
function configureTool() {  
    theTool = fl.tools.activeTool;  
    theTool.setToolName("myTool");  
    theTool.setIcon("myTool.png");  
    theTool.setMenuString("My Tool's menu string");  
    theTool.setToolTip("my tool's tool tip");  
    theTool.setOptionsFile("mtTool.xml");  
}
```

```
function configureTool() {  
    theTool = fl.tools.activeTool;  
    theTool.setToolName("ellipse");  
    theTool.setIcon("Ellipse.png");  
    theTool.setMenuString("Ellipse");  
    theTool.setToolTip("Ellipse");  
    theTool.showTransformHandles( true );  
}
```

## confirm()

### Disponibilité

Flash 8.

### Utilisation

```
confirm ( strAlert )
```

### Paramètres

**strAlert** Chaîne qui spécifie le message devant apparaître dans la boîte de dialogue d'alerte.

### Valeur renvoyée

Valeur booléenne : `true` si l'utilisateur clique sur OK, `false` si l'utilisateur clique sur Annuler.

### Description

Cette méthode affiche dans une boîte de dialogue modale une chaîne accompagnée des boutons OK et Annuler.

*Remarque : si aucun document (fichier FLA) n'est ouvert, cette méthode échoue en produisant une condition d'erreur.*

### Exemple

L'exemple suivant affiche le message «Sort data?» (Trier les données ?) dans une boîte de dialogue d'alerte.

```
confirm("Sort data?");
```

### Voir aussi

`alert()`, `prompt()`

## deactivate()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function deactivate() {  
    // statements  
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Cette fonction est appelée lorsque l'outil extensible est désactivé (c'est-à-dire lorsque l'utilisateur sélectionne un autre outil). Elle permet d'effectuer toute opération de nettoyage nécessaire avant la désactivation de cet outil.

### Exemple

L'exemple suivant affiche un message dans le panneau Sortie lorsque l'outil devient inactif.

```
function deactivate() {  
    fl.trace( "Tool is no longer active" );  
}
```

## keyDown()

### Disponibilité

Flash MX 2004.



### Utilisation

```
function keyDown() {  
    // statements  
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Cette fonction est appelée si l'outil extensible est actif lorsque l'utilisateur appuie sur une touche. Le script doit alors appeler `tools.getKeyDown()` pour identifier la touche en question.

### Exemple

L'exemple suivant affiche des informations sur la touche pressée par l'utilisateur lorsque l'outil extensible est actif.

```
function keyDown() {  
    fl.trace("key " + fl.tools.getKeyDown() + " was pressed");  
}
```

### Voir aussi

[keyUp\(\)](#), [tools.getKeyDown\(\)](#)

## keyUp()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function keyUp() {  
    // statements  
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Cette fonction est appelée si l'outil extensible est actif lorsque l'utilisateur relâche une touche.

### Exemple

L'exemple suivant affiche un message dans le panneau Sortie lorsque l'outil extensible est actif et que l'utilisateur relâche une touche.

```
function keyUp() {  
    fl.trace("Key is released");  
}
```

#### Voir aussi

[keyDown\(\)](#)

## mouseDoubleClick()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
function mouseDoubleClick() {  
    // statements  
}
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Aucune.

#### Description

Fonction appelée lorsque l'outil extensible est actif et que l'utilisateur double-clique sur la scène.

#### Exemple

L'exemple suivant affiche un message dans le panneau Sortie lorsque l'outil extensible est actif et que l'utilisateur double-clique sur le bouton de sa souris.

```
function mouseDoubleClick() {  
    fl.trace("Mouse was double-clicked");  
}
```

## mouseDown()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
function mouseDown( [ pt ] ) {  
    // statements  
}
```

### Paramètres

**pt** Point indiquant l'emplacement de la souris lors du clic. Il est transmis à la fonction lors de l'appui sur le bouton de la souris. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Cette fonction est appelée si l'outil extensible est actif lors d'un clic de souris alors que le pointeur se trouve sur la scène.

### Exemple

Les exemples suivants illustrent l'utilisation de cette fonction lorsque l'outil extensible est actif. Le premier exemple affiche un message dans le panneau Sortie pour signaler un clic de souris. Le second exemple affiche les coordonnées *x* et *y* de la souris lors du clic.

```
function mouseDown() {  
    fl.trace("Mouse button has been pressed");  
}  
function mouseDown(pt) {  
    fl.trace("x = "+ pt.x+" :: y = "+pt.y);  
}
```

## mouseMove()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function mouseMove( [ pt ] ) {  
    // statements  
}
```

### Paramètres

**pt** Point indiquant les coordonnées actuelles de la souris. Il est transmis à la fonction lors de tout mouvement de la souris, ce qui permet de toujours connaître l'emplacement de celle-ci. Si la scène est en mode Modification ou Modification sur place, les coordonnées du point sont relatives à l'objet en cours de modification. Sinon, les coordonnées du point sont relatives à la scène. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Fonction appelée lorsque l'outil extensible est actif et que l'utilisateur survole un point précis de la scène avec sa souris. Le ou les boutons de la souris peuvent être enfoncés ou non.

### Exemple

Les exemples suivants illustrent les utilisations possibles de cette fonction. Le premier exemple affiche un message dans le panneau Sortie pour signaler un déplacement de la souris. Le second exemple affiche les coordonnées  $x$  et  $y$  de la souris lors de son déplacement.

```
function mouseMove() {  
    fl.trace("moving");  
}  
  
function mouseMove(pt) {  
    fl.trace("x = " + pt.x + " :: y = " + pt.y);  
}
```

## mouseUp()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function mouseUp() {  
    // statements  
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Fonction appelée lorsque l'outil extensible est actif et que l'utilisateur relâche le bouton de sa souris après avoir cliqué sur la scène.

### Exemple

L'exemple suivant affiche un message dans le panneau Sortie lorsque l'outil extensible est actif et que l'utilisateur relâche le bouton de sa souris.

```
function mouseUp() {  
    fl.trace("mouse is up");  
}
```

## notifySettingsChanged()

### Disponibilité

Flash MX 2004.

### Utilisation

```
function notifySettingsChanged() {  
    // statements  
}
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Cette fonction est appelée lorsqu'un outil est actif et que l'utilisateur en modifie les options dans l'inspecteur Propriétés. Vous pouvez utiliser la propriété `tools.activeTool` pour demander les valeurs actuelles des options (voir [tools.activeTool](#)).

### Exemple

L'exemple suivant affiche un message dans le panneau Sortie lorsque l'outil extensible est actif et que l'utilisateur modifie ses options dans l'inspecteur Propriétés.

```
function notifySettingsChanged() {  
    var theTool = fl.tools.activeTool;  
    var newValue = theTool.myProp;  
}
```

## prompt()

### Disponibilité

Flash MX 2004.

### Utilisation

```
prompt(promptMsg [,text])
```

### Paramètres

**promptMsg** Chaîne à afficher dans la boîte de dialogue Invite (limitée à 256 caractères dans Mac OS X).

**text** Chaîne facultative à afficher comme valeur par défaut pour le champ de texte.

### Valeur renvoyée

La chaîne saisie par l'utilisateur, si celui-ci a ensuite cliqué sur OK, ou `null` s'il a cliqué sur Annuler.

### Description

Cette méthode affiche dans une boîte de dialogue modale une invite et un texte facultatif, accompagnés des boutons OK et Annuler.

### Exemple

L'exemple suivant invite l'utilisateur à taper son nom d'utilisateur. Si l'utilisateur saisit un nom et clique sur OK, ce nom s'affiche dans le panneau Sortie.

```
var userName = prompt("Enter user name", "Type user name here");  
fl.trace(userName);
```

#### Voir aussi

[alert\(\)](#), [confirm\(\)](#)

## setCursor()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
function setCursor() {  
    // statements  
}
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Aucune.

#### Description

Cette fonction est appelée lorsque l'outil extensible est actif et que l'utilisateur déplace sa souris, ce qui permet au script de définir des pointeurs personnalisés. Le script doit appeler `tools.setCursor()` pour désigner le pointeur à utiliser. La liste des pointeurs et des valeurs entières correspondantes figure dans la section [tools.setCursor\(\)](#).

#### Exemple

```
function setCursor() {  
    fl.tools.setCursor( 1 );  
}
```

# Chapitre 3 : Objet actionsPanel

## Disponibilité

Flash CS3 Professional.

## Description

L'objet actionsPanel, qui représente le panneau Actions affiché actuellement, est une propriété de l'objet flash (voir [fl.actionsPanel](#)).

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet actionsPanel :

Méthode	Description
<code>actionsPanel.getClassForObject()</code>	Renvoie la classe d'une variable spécifiée.
<code>actionsPanel.getScriptAssistMode()</code>	Précise si le mode Assistant de script est activé ou non.
<code>actionsPanel.getSelectedText()</code>	Renvoie le texte actuellement sélectionné dans le panneau Actions.
<code>actionsPanel.getText()</code>	Renvoie le texte du panneau Actions.
<code>actionsPanel.hasSelection()</code>	Indique si du texte est actuellement sélectionné dans le panneau Actions.
<code>actionsPanel.replaceSelectedText()</code>	Remplace le texte actuellement sélectionné par celui qui est spécifié.
<code>actionsPanel.setScriptAssistMode()</code>	Active ou désactive le mode Assistant de script.
<code>actionsPanel.setSelection()</code>	Sélectionne un jeu de caractères particulier dans le panneau Actions.
<code>actionsPanel.setText()</code>	Efface le texte affiché dans le panneau Actions pour ensuite y ajouter celui qui est spécifié.

## actionsPanel.getClassForObject()

### Disponibilité

Flash CS3 Professional.

### Utilisation

`actionsPanel.getClassForObject(ASvariableName)`

### Paramètres

**ASvariableName** Chaîne représentant le nom d'une variable ActionScript.

### Valeur renvoyée

Chaîne représentant la classe dont *ASvariableName* est membre.

### Description

Méthode qui renvoie la classe de la variable spécifiée devant être définie dans le panneau Actions actuellement affiché. En outre, le curseur ou le texte sélectionné dans le panneau Actions doit être positionné après la définition de la variable.

### Exemple

L'exemple suivant affiche la classe attribuée à la variable myVar si le curseur est placé après l'instruction var myVar:ActivityEvent; dans le panneau Actions.

```
// Place the following code in the Actions panel,  
// and position the cursor somewhere after the end of the line  
var myVar:ActivityEvent;  
// Place the following code in the JSFL file  
var theClass = fl.actionsPanel.getClassForObject("myVar");  
fl.trace(theClass); // traces: "ActivityEvent"
```

## actionsPanel.getScriptAssistMode()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
actionsPanel.getScriptAssistMode()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne précisant si le mode Assistant de script est activé (true) ou pas (false).

### Description

Méthode qui précise si le mode Assistant de script est activé ou non.

### Exemple

L'exemple suivant affiche un message si le mode Assistant de script n'est pas activé.

```
mAssist = fl.actionsPanel.getScriptAssistMode();  
if (!mAssist) {  
    alert("For more guidance when writing ActionScript code, try Script Assist mode");  
}
```

### Voir aussi

[actionsPanel.setScriptAssistMode\(\)](#)



## actionsPanel.getSelectedText()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
actionsPanel.getSelectedText()
```

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne contenant le texte actuellement sélectionné dans le panneau Actions.

### Description

Méthode qui renvoie le texte actuellement sélectionné dans le panneau Actions.

### Exemple

L'exemple suivant affiche le texte actuellement sélectionné dans le panneau Actions.

```
var apText = fl.actionsPanel.getSelectedText();  
fl.trace(apText);
```

### Voir aussi

[actionsPanel.getText\(\)](#), [actionsPanel.hasSelection\(\)](#), [actionsPanel.replaceSelectedText\(\)](#),  
[actionsPanel.setSelection\(\)](#)

## actionsPanel.getText()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
actionsPanel.getText()
```

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne contenant tout le texte affiché dans le panneau Actions.

### Description

Méthode qui renvoie le texte présent dans le panneau Actions.

### Exemple

L'exemple suivant affiche le texte présent dans le panneau Actions.

```
var apText = fl.actionsPanel.getText();  
fl.trace(apText);
```

#### Voir aussi

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.setText\(\)](#)

## actionsPanel.hasSelection()

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

```
actionsPanel.hasSelection()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Valeur booléenne spécifiant si du texte est sélectionné (`true`) ou non (`false`) dans le panneau Actions.

#### Description

Méthode qui indique si du texte est actuellement sélectionné dans le panneau Actions.

#### Exemple

L'exemple suivant affiche le texte actuellement sélectionné dans le panneau Actions. Si aucun texte n'est sélectionné, l'intégralité du texte contenu dans le panneau Actions est affiché.

```
if (fl.actionsPanel.hasSelection()) {  
    var apText = fl.actionsPanel.getSelectedText();  
}  
else {  
    var apText = fl.actionsPanel.getText();  
}  
fl.trace(apText);
```

#### Voir aussi

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.getText\(\)](#), [actionsPanel.replaceSelectedText\(\)](#),  
[actionsPanel.setSelection\(\)](#)

## actionsPanel.replaceSelectedText()

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

```
actionsPanel.replaceSelectedText(replacementText)
```

### Paramètres

**`replacementText`** Chaîne représentant le texte qui doit remplacer le texte sélectionné dans le panneau Actions.

### Valeur renvoyée

Valeur booléenne, `true` si le panneau Actions est trouvé, `false` dans le cas contraire.

### Description

Méthode qui remplace le texte actuellement sélectionné par celui qui est spécifié dans *replacementText*. Si *replacementText* contient plus de caractères que le texte sélectionné, tout caractère situé après le texte sélectionné suivra désormais *replacementText* et ne sera donc pas écrasé.

### Exemple

L'exemple suivant remplace le texte actuellement sélectionné dans le panneau Actions.

```
if (fl.actionsPanel.hasSelection()) {  
    fl.actionsPanel.replaceSelectedText("// © 2006 Adobe Inc.");  
}
```

### Voir aussi

`actionsPanel.getSelectedText()`, `actionsPanel.hasSelection()`, `actionsPanel.setSelection()`,  
`actionsPanel.setText()`

## actionsPanel.setScriptAssistMode()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
actionsPanel.setScriptAssistMode(bScriptAssist)
```

### Paramètres

**`bScriptAssist`** Valeur booléenne spécifiant si le mode Assistant de script doit être activé ou désactivé.

### Valeur renvoyée

Valeur booléenne précisant si le mode Assistant de script a bien été activé ou désactivé.

### Description

Méthode qui active ou désactive le mode Assistant de script.

### Exemple

L'exemple suivant active/désactive le mode Assistant de script.

```
fl.trace(fl.actionsPanel.getScriptAssistMode());  
if (fl.actionsPanel.getScriptAssistMode()) {  
    fl.actionsPanel.setScriptAssistMode(false);  
}  
else {  
    fl.actionsPanel.setScriptAssistMode(true);  
}  
fl.trace(fl.actionsPanel.getScriptAssistMode());
```

#### Voir aussi

[actionsPanel.getScriptAssistMode\(\)](#)

## actionsPanel.setSelection()

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

```
actionsPanel.setSelection(startIndex, numberOfChars)
```

#### Paramètres

**startIndex** Nombre entier (commençant à zéro) qui indique le premier caractère à sélectionner.

**numberOfChars** Nombre entier spécifiant le nombre de caractères à sélectionner.

#### Valeur renvoyée

Valeur booléenne qui indique si les caractères demandés peuvent être sélectionnés (`true`) ou non (`false`).

#### Description

Méthode qui sélectionne un jeu de caractères spécifié dans le panneau Actions.

#### Exemple

L'exemple suivant remplace les caractères « 2006 », affichés dans le panneau Actions, par le texte spécifié.

```
// Type the following as the first line in the Actions panel  
// 2006 - Addresses user request 40196  
// Type the following in the JSFL file  
fl.actionsPanel.setSelection(3,4);  
fl.actionsPanel.replaceSelectedText("// Last updated: 2007");
```

#### Voir aussi

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.hasSelection\(\)](#),  
[actionsPanel.replaceSelectedText\(\)](#)

## **actionsPanel.setText()**

### **Disponibilité**

Flash CS3 Professional.

### **Utilisation**

```
actionsPanel.setText(replacementText)
```

### **Paramètres**

**replacementText** Chaîne représentant le texte à introduire dans le panneau Actions.

### **Valeur renvoyée**

Valeur booléenne, `true` si le texte spécifié a été placé dans le panneau Actions, `false` dans le cas contraire.

### **Description**

Méthode qui efface le texte affiché dans le panneau Actions pour ensuite y ajouter celui qui est spécifié dans *replacementText*.

### **Exemple**

L'exemple suivant remplace le texte actuellement affiché dans le panneau Actions par le texte spécifié.

```
fl.actionsPanel.setText("// Deleted this code - no longer needed");
```

### **Voir aussi**

[actionsPanel.getText\(\)](#), [actionsPanel.replaceSelectedText\(\)](#)

# Chapitre 4 : Objet BitmapInstance

**Héritage**    [Objet Element](#) > [Objet Instance](#) > Objet BitmapInstance

## Disponibilité

Flash MX 2004.

## Description

L'objet BitmapInstance est une sous-classe de l'objet Instance et représente un bitmap dans une image (voir [Objet Instance](#)).

## Résumé des méthodes

Outre les méthodes de l'objet [Objet Instance](#), l'objet BitmapInstance gère les méthodes suivantes :

Méthode	Description
<code>bitmapInstance.getBits()</code>	Permet de créer des effets de bitmap en extrayant les bits du bitmap, en les manipulant et en les renvoyant à Flash.
<code>bitmapInstance.setBits()</code>	Définit les bits d'un élément bitmap existant.

## Résumé des propriétés

Outre les propriétés de l'objet [Objet Instance](#), l'objet BitmapInstance gère les propriétés suivantes :

Propriété	Description
<code>bitmapInstance.hPixels</code>	En lecture seule ; nombre entier indiquant la largeur du bitmap, en pixels.
<code>bitmapInstance.vPixels</code>	En lecture seule ; nombre entier indiquant la hauteur du bitmap, en pixels.

## bitmapInstance.getBits()

### Disponibilité

Flash MX 2004.

### Utilisation

`bitmapInstance.getBits()`

### Paramètres

Aucun.

### Valeur renvoyée

Objet qui contient les propriétés `width`, `height`, `depth`, `bits` et, si le bitmap comporte une table de couleurs, `cTab`. L'élément `bits` est un tableau d'octets. L'élément `cTab` est un tableau de valeurs colorimétriques sous la forme "#RRGGBB". La longueur du tableau est celle de la table de couleurs.

le tableau d'octets n'est pertinent que s'il est référencé par une DLL ou une bibliothèque partagée. En règle générale, il est uniquement destiné à la création d'un outil extensible ou d'un effet. Pour plus d'informations sur la création de DLL à utiliser avec Flash JavaScript, voir la section « [Extensibilité de niveau C](#) » à la page 535.

### Description

Méthode permettant de créer des effets de bitmap en extrayant les bits du bitmap, en les manipulant et en les renvoyant à Flash.

### Exemple

Le code suivant crée une référence à l'objet sélectionné, teste s'il est de type bitmap, et lit la hauteur, la largeur et la profondeur de couleurs du bitmap :

```
var isBitmap = fl.getDocumentDOM().selection[0].instanceType;
if (isBitmap == "bitmap") {
    var bits = fl.getDocumentDOM().selection[0].getBits();
    fl.trace("height = " + bits.height);
    fl.trace("width = " + bits.width);
    fl.trace("depth = " + bits.depth);
}
```

### Voir aussi

[bitmapInstance.setBits\(\)](#)

## bitmapInstance.hPixels

### Disponibilité

Flash MX 2004.

### Utilisation

`bitmapInstance.hPixels`

### Description

Propriété en lecture seule ; nombre entier indiquant la largeur du bitmap, c'est-à-dire le nombre de pixels dans sa dimension horizontale.

### Exemple

Le code suivant récupère la largeur du bitmap, en pixels :

```
// Get the number of pixels in the horizontal dimension.
var bmObj = fl.getDocumentDOM().selection[0];
var isBitmap = bmObj.instanceType;
if (isBitmap == "bitmap") {
    var numHorizontalPixels = bmObj.hPixels;
}
```

### Voir aussi

[bitmapInstance.vPixels](#)

## `bitmapInstance.setBits()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
bitmapInstance.setBits(bitmap)
```

### Paramètres

**bitmap** Objet qui contient les propriétés `height`, `width`, `depth`, `bits` et `cTab`. Les propriétés `height`, `width` et `depth` sont des entiers. La propriété `bits` est un tableau d'octets. La propriété `cTab` n'est nécessaire que pour les bitmaps ayant une profondeur de couleurs égale ou inférieure à 8 bits. Il s'agit d'une chaîne qui représente une valeur colorimétrique sous la forme "#RRGGBB".

*Remarque : le tableau d'octets n'est pertinent que s'il est référencé par une bibliothèque externe. En règle générale, il est uniquement destiné à la création d'un outil extensible ou d'un effet.*

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit les bits d'un élément bitmap existant. Elle permet de créer des effets de bitmap en extrayant les bits du bitmap, en les manipulant et en renvoyant le bitmap à Flash.

### Exemple

Le code suivant vérifie si la sélection est de type bitmap, puis définit la hauteur de ce bitmap sur 150 pixels :

```
var isBitmap = fl.getDocumentDOM().selection[0].instanceType;
if(isBitmap == "bitmap") {
    var bits = fl.getDocumentDOM().selection[0].getBits();
    bits.height = 150;
    fl.getDocumentDOM().selection[0].setBits(bits);
}
```

### Voir aussi

[bitmapInstance.getBits\(\)](#)

## `bitmapInstance.vPixels`

### Disponibilité

Flash MX 2004.

### Utilisation

```
bitmapInstance.vPixels
```



### Description

Propriété en lecture seule ; nombre entier indiquant la hauteur du bitmap, c'est-à-dire le nombre de pixels dans sa dimension verticale.

### Exemple

Le code suivant récupère la hauteur du bitmap, en pixels :

```
// Get the number of pixels in the vertical dimension.  
var bmObj = fl.getDocumentDOM().selection[0];  
var isBitmap = bmObj.instanceType;  
if(isBitmap == "bitmap"){  
    var numVerticalPixels = bmObj.vPixels;  
}
```

### Voir aussi

[bitmapInstance.hPixels](#)

# Chapitre 5 : Objet BitmapItem

**Héritage**    [Objet Item](#) > Objet BitmapItem

## Disponibilité

Flash MX 2004.

## Description

L'objet BitmapItem fait référence à une image bitmap dans la bibliothèque d'un document. L'objet BitmapItem est une sous-classe de l'objet Item (voir [Objet Item](#)).

## Résumé des propriétés

Outre les propriétés de l'[Objet Item](#), l'objet BitmapItem a les propriétés suivantes :

Propriété	Description
<code>bitmapItem.allowSmoothing</code>	Valeur booléenne qui spécifie si le lissage d'un bitmap est autorisé ou non.
<code>bitmapItem.compressionType</code>	Chaîne qui détermine le type de la compression d'image appliquée au bitmap.
<code>bitmapItem.fileLastModifiedDate</code>	Nombre de secondes écoulées entre le 1er janvier 1970 et la date de modification du fichier original.
<code>bitmapItem.originalCompressionType</code>	Spécifie si l'élément a été importé sous forme de fichier jpeg.
<code>bitmapItem.sourceFileExists</code>	Spécifie si le fichier importé dans la bibliothèque existe toujours à l'emplacement d'où il provient.
<code>bitmapItem.sourceFileIsCurrent</code>	Spécifie si la date de modification du fichier de l'élément de la bibliothèque correspond à celle trouvée sur le disque du fichier importé.
<code>bitmapItem.sourceFilePath</code>	Chemin et nom du fichier importé dans la bibliothèque.
<code>bitmapItem.useDeblocking</code>	Spécifie si le dégroupage est activé.
<code>bitmapItem.useImportedJPEGQuality</code>	Valeur booléenne qui spécifie si la qualité JPEG importée par défaut doit être utilisée ou non.

## Résumé des méthodes

Outre les propriétés de l'[Objet Item](#), l'objet BitmapItem a les méthodes suivantes :

Méthode	Description
<code>bitmapItem.exportToFile()</code>	Exporte l'élément spécifié dans un fichier PNG ou JPG.

## bitmapItem.allowSmoothing

## Disponibilité

Flash MX 2004.

### Utilisation

`bitmapItem.allowSmoothing`

### Description

Propriété : valeur booléenne indiquant si le lissage d'un bitmap est autorisé (`true`) ou non (`false`).

### Exemple

Le code suivant définit la propriété `allowSmoothing` du premier élément de la bibliothèque du document actif sur `true` :

```
fl.getDocumentDOM().library.items[0].allowSmoothing = true;  
alert(fl.getDocumentDOM().library.items[0].allowSmoothing);
```

## bitmapItem.compressionType

### Disponibilité

Flash MX 2004.

### Utilisation

`bitmapItem.compressionType`

### Description

Propriété : chaîne déterminant le type de la compression d'image appliquée au bitmap. Les valeurs acceptables sont `photo` ou `lossless`. Si la valeur de `bitmapItem.useImportedJPEGQuality` est `false`, `photo` correspond à une qualité JPEG comprise entre 0 et 100 ; si la valeur de `bitmapItem.useImportedJPEGQuality` est `true`, `photo` correspond à la qualité JPEG avec la valeur de qualité par défaut du document. La valeur `lossless` correspond aux formats GIF et PNG (voir [bitmapItem.useImportedJPEGQuality](#)).

### Exemple

Le code suivant définit la propriété `compressionType` du premier élément de la bibliothèque du document actuel sur `photo` :

```
fl.getDocumentDOM().library.items[0].compressionType = "photo";  
alert(fl.getDocumentDOM().library.items[0].compressionType);
```

## bitmapItem.exportToFile()

### Disponibilité

Flash CS4 Professional.

### Utilisation

`bitmapItem.exportToFile(fileURI)`

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et spécifiant le chemin et le nom du fichier exporté.

### Valeur renvoyée

Valeur booléenne `true` si le fichier a bien été exporté, `false` dans le cas contraire.

### Description

Méthode qui exporte l'élément spécifié dans un fichier PNG ou JPG.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément bitmap, le code suivant l'exporte sous forme de fichier JPG :

```
var imageFileURL = "file:///C:/exportTest/out.jpg";  
var libItem = fl.getDocumentDOM().library.items[0];  
libItem.exportToFile(imageFileURL);
```

## bitmapItem.fileLastModifiedDate

### Disponibilité

Flash CS4 Professional.

### Utilisation

`bitmapItem.fileLastModifiedDate`

### Description

Propriété en lecture seule ; chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1er janvier 1970 et la date de modification du fichier original au moment où il a été importé dans la bibliothèque. Si le fichier n'existe plus, cette valeur est « 00000000 ».

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément bitmap, le code suivant affiche un nombre hexadécimal tel que décrit ci-dessus :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("Mod date when imported = " + libItem.fileLastModifiedDate);
```

### Voir aussi

[bitmapItem.sourceFileExists](#), [bitmapItem.sourceFileIsCurrent](#), [bitmapItem.sourceFilePath](#), [FLfile.getModificationDate\(\)](#)

## bitmapItem.originalCompressionType

### Disponibilité

Flash CS4 Professional.

### Utilisation

`bitmapItem.originalCompressionType`

### Description

Propriété en lecture seule ; chaîne indiquant si l'élément spécifié est importé sous forme de fichier jpeg. Les valeurs possibles de cette propriété sont « photo » (pour les fichiers jpeg) et « lossless » (pour les types de fichiers non compressés tels que GIF et PNG).

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément bitmap, le code suivant affiche « photo » si le fichier a été importé dans la bibliothèque sous forme de fichier jpeg, ou « lossless » si ce n'est pas le cas :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("Imported compression type = "+ libItem.originalCompressionType);
```

### Voir aussi

[bitmapItem.compressionType](#)

## bitmapItem.quality

### Disponibilité

Flash MX 2004.

### Utilisation

bitmapItem.quality

### Description

Propriété ; entier définissant la qualité du bitmap. Pour utiliser la qualité par défaut du document, spécifiez -1 ; sinon, spécifiez un entier entre 0 et 100. Uniquement disponible pour la compression JPEG.

### Exemple

Le code suivant définit la propriété `quality` du premier élément de la bibliothèque du document actuel sur 65 :

```
fl.getDocumentDOM().library.items[0].quality = 65;  
alert(fl.getDocumentDOM().library.items[0].quality);
```

## bitmapItem.sourceFileExists

### Disponibilité

Flash CS4 Professional.

### Utilisation

bitmapItem.sourceFileExists

### Description

Propriété en lecture seule ; valeur booléenne, `true` si le fichier importé dans la bibliothèque existe toujours à l'emplacement d'où il provient ; `false` dans le cas contraire.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément bitmap, le code suivant affiche « true » si le fichier importé dans la bibliothèque existe toujours :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("sourceFileExists = " + libItem.sourceFileExists);
```

### Voir aussi

[bitmapItem.sourceFileIsCurrent](#),  
[bitmapItem.sourceFilePath](#)

## bitmapItem.sourceFileIsCurrent

### Disponibilité

Flash CS4 Professional.

### Utilisation

`bitmapItem.sourceFileIsCurrent`

### Description

Propriété en lecture seule ; valeur booléenne, `true` si la date de modification du fichier de l'élément de la bibliothèque correspond à la date de modification sur disque du fichier importé, `false` dans le cas contraire.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément bitmap, le code suivant affiche « true » si le fichier importé n'a pas été modifié sur le disque depuis son importation :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("fileIsCurrent = " + libItem.sourceFileIsCurrent);
```

### Voir aussi

[bitmapItem.fileLastModifiedDate](#), [bitmapItem.sourceFilePath](#)

## bitmapItem.sourceFilePath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`bitmapItem.sourceFilePath`

### Description

Propriété en lecture seule ; chaîne de type « fichier:///URI » qui représente le chemin et le nom du fichier importé dans la bibliothèque.

### Exemple

L'exemple suivant affiche le nom et le chemin du fichier source de tout élément de la bibliothèque de type

« bitmap » :

```
for (idx in fl.getDocumentDOM().library.items) {  
  if (fl.getDocumentDOM().library.items[idx].itemType == "bitmap") {  
    var myItem = fl.getDocumentDOM().library.items[idx];  
    fl.trace(myItem.name + " source is " + myItem.sourceFilePath);  
  }  
}
```

### Voir aussi

[bitmapItem.sourceFileExists](#)

## bitmapItem.useDeblocking

### Disponibilité

Flash CS4 Professional.

### Utilisation

`bitmapItem.useDeblocking`

### Description

Propriété ; valeur booléenne indiquant si le dégroupage est activé (`true`) ou non (`false`).

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément bitmap, le code suivant active le dégroupage pour l'élément :

```
var libItem = fl.getDocumentDOM().library.items[0];  
libItem.useDeblocking = true;
```

## bitmapItem.useImportedJPEGQuality

### Disponibilité

Flash MX 2004.

### Utilisation

`bitmapItem.useImportedJPEGQuality`

### Description

Propriété ; valeur booléenne indiquant si la qualité JPEG importée par défaut doit être utilisée (`true`) ou non (`false`). Uniquement disponible pour la compression JPEG.

### **Exemple**

Le code suivant définit la propriété `useImportedJPEGQuality` du premier élément de la bibliothèque du document actuel sur `true` :

```
fl.getDocumentDOM().library.items[0].useImportedJPEGQuality = true;  
alert(fl.getDocumentDOM().library.items[0].useImportedJPEGQuality);
```



# Chapitre 6 : Objet CompiledClipInstance

**Héritage**    [Objet Element](#) > [Objet Instance](#) > Objet CompiledClipInstance

## Disponibilité

Flash MX 2004.

## Description

L'objet CompiledClipInstance est une sous-classe de l'objet Instance. Il s'agit généralement d'une occurrence d'un clip qui a été converti en élément de bibliothèque de clips compilé (voir [Objet Instance](#)).

## Résumé des propriétés

Outre les propriétés de l'[Objet Instance](#), l'objet CompiledClipInstance possède les propriétés suivantes :

Propriété	Description
<code>compiledClipInstance.accName</code>	Chaîne équivalente au champ Nom du panneau Accessibilité.
<code>compiledClipInstance.actionScript</code>	Chaîne qui représente le code ActionScript de cette occurrence, équivalente à <code>symbolInstance.actionScript</code> .
<code>compiledClipInstance.description</code>	Chaîne équivalente au champ Description du panneau Accessibilité.
<code>compiledClipInstance.forceSimple</code>	Valeur booléenne qui active et désactive l'accessibilité aux enfants de l'objet.
<code>compiledClipInstance.shortcut</code>	Chaîne équivalente au champ Raccourci du panneau Accessibilité.
<code>compiledClipInstance.silent</code>	Valeur booléenne qui active et désactive l'accessibilité à l'objet ; équivalente à l'inverse de la fonction Rendre l'objet accessible dans le panneau Accessibilité.
<code>compiledClipInstance.tabIndex</code>	Nombre entier équivalent au champ Index de tabulation du panneau Accessibilité.

## compiledClipInstance.accName

## Disponibilité

Flash MX 2004.

## Utilisation

`compiledClipInstance.accName`

## Description

Propriété ; chaîne équivalente au champ Nom du panneau Accessibilité. Les logiciels de lecture vocale identifient l'objet en prononçant ce nom.

## Exemple

L'exemple suivant récupère et définit le nom d'accessibilité du premier objet sélectionné :

```
// Get the name of the object.  
var theName = fl.getDocumentDOM().selection[0].accName;  
// Set the name of the object.  
fl.getDocumentDOM().selection[0].accName = 'Home Button';
```

## compiledClipInstance.actionScript

### Disponibilité

Flash MX 2004.

### Utilisation

`compiledClipInstance.actionScript`

### Description

Propriété ; chaîne qui représente le code ActionScript de cette occurrence ; équivalent à [symbolInstance.actionScript](#).

### Exemple

Le code suivant assigne du code ActionScript aux éléments spécifiés :

```
// Assign some ActionScript to a specified Button compiled clip instance.  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0]  
    .actionScript = "on(click) {trace('button is clicked');}";  
// Assign some ActionScript to the currently selected Button compiled clip instance.  
fl.getDocumentDOM().selection[0].actionScript =  
    "on(click) {trace('button is clicked');}";
```

## compiledClipInstance.description

### Disponibilité

Flash MX 2004.

### Utilisation

`compiledClipInstance.description`

### Description

Propriété ; chaîne équivalente au champ Description du panneau Accessibilité. Cette description est lue par les logiciels de lecture vocale.

### Exemple

L'exemple suivant montre comment récupérer et définir la propriété `description` :

```
// Get the description of the current selection.  
var theDescription = fl.getDocumentDOM().selection[0].description;  
// Set the description of the current selection.  
fl.getDocumentDOM().selection[0].description =  
    "This is compiled clip number 1";
```

## compiledClipInstance.forceSimple

### Disponibilité

Flash MX 2004.

### Utilisation

`compiledClipInstance.forceSimple`

### Description

Propriété ; valeur booléenne qui active et désactive l'accessibilité aux enfants de l'objet. Cette propriété est équivalente à l'inverse du paramètre Rendre les objets enfant accessibles du panneau Accessibilité. Si `forceSimple` est `true`, son effet est similaire à la désactivation de l'option Rendre les objets enfant accessibles. Si `forceSimple` est `false`, son effet est similaire à l'activation de l'option Rendre les objets enfant accessibles.

### Exemple

L'exemple suivant montre comment récupérer et définir la propriété `forceSimple` :

```
// Query if the children of the object are accessible.  
var areChildrenAccessible = fl.getDocumentDOM().selection[0].forceSimple;  
// Allow the children of the object to be accessible.  
fl.getDocumentDOM().selection[0].forceSimple = false;
```

## compiledClipInstance.shortcut

### Disponibilité

Flash MX 2004.

### Utilisation

`compiledClipInstance.shortcut`

### Description

Propriété ; chaîne équivalente au champ Raccourci du panneau Accessibilité. Ce raccourci est lu par les logiciels de lecture vocale. Cette propriété n'est pas disponible pour les champs de texte dynamiques.

### Exemple

L'exemple suivant montre comment récupérer et définir la propriété `shortcut` :

```
// Get the shortcut key of the object.  
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;  
// Set the shortcut key of the object.  
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+I";
```

## compiledClipInstance.silent

### Disponibilité

Flash MX 2004.

### Utilisation

`compiledClipInstance.silent`

### Description

Propriété ; valeur booléenne qui active et désactive l'accessibilité à l'objet ; équivalente à l'inverse de la fonction Rendre l'objet accessible dans le panneau Accessibilité. Ainsi, si `silent` est `true`, l'option Rendre l'objet accessible est désactivée. Si `silent` est `false`, l'option Rendre l'objet accessible est activée.

### Exemple

L'exemple suivant montre comment récupérer et définir la propriété `silent` :

```
// Query if the object is accessible.  
var isSilent = fl.getDocumentDOM().selection[0].silent;  
// Set the object to be accessible.  
fl.getDocumentDOM().selection[0].silent = false;
```

## compiledClipInstance.tabIndex

### Disponibilité

Flash MX 2004.

### Utilisation

`compiledClipInstance.tabIndex`

### Description

Propriété ; nombre entier équivalent au champ Index de tabulation du panneau Accessibilité. Cette propriété crée l'ordre de tabulation dans lequel les objets sont sélectionnés lorsque l'utilisateur appuie sur la touche de tabulation.

### Exemple

L'exemple suivant montre comment récupérer et définir la propriété `tabIndex` :

```
// Get the tabIndex of the object.  
var theTabIndex = fl.getDocumentDOM().selection[0].tabIndex;  
// Set the tabIndex of the object.  
fl.getDocumentDOM().selection[0].tabIndex = 1;
```

# Chapitre 7 : Objet `compilerErrors`

## Disponibilité

Flash CS3 Professional.

## Description

L'objet `compilerErrors`, qui représente le panneau Erreurs de compilation, est une propriété de l'objet flash (`fl`) et est accessible via `fl.compilerErrors` (voir [Objet flash \(fl\)](#)).

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet `compilerErrors` :

Méthode	Description
<code>compilerErrors.clear()</code>	Efface le contenu du panneau Erreurs de compilation.
<code>compilerErrors.save()</code>	Enregistre le contenu du panneau Erreurs de compilation dans un fichier texte local.

## `compilerErrors.clear()`

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
compilerErrors.clear()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui efface le contenu du panneau Erreurs de compilation.

### Exemple

L'exemple suivant efface le contenu du panneau Erreurs de compilation :

```
fl.compilerErrors.clear();
```

### Voir aussi

[compilerErrors.save\(\)](#)

## `compilerErrors.save()`

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
compilerErrors.save(fileURI [, bAppendToFile [, bUseSystemEncoding]])
```

### Paramètres

**fileURI** <Check Alignment of PHs>Chaîne exprimée sous la forme « fichier:///URI » et spécifiant le nom du fichier enregistré. Si *fileURI* existe déjà alors que vous n'avez pas spécifié de valeur `true` pour *bAppendToFile*, le remplacement de *fileURI* s'effectue sans message d'avertissement.

**bAppendToFile** Valeur booléenne facultative spécifiant si le contenu du panneau Erreurs de compilation doit être ajouté à *fileURI* (`true`) ou non (`false`). La valeur par défaut est `false`.

**bUseSystemEncoding** Valeur booléenne facultative spécifiant si le texte du panneau Erreur de compilation doit être enregistré ou non en utilisant le codage système. Si cette valeur est `false` (valeur par défaut), le texte du panneau Erreurs de compilation est enregistré en codage UTF-8 et intègre des caractères marqueurs d'ordre d'octets au début du texte. La valeur par défaut est `false`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui enregistre le contenu du panneau Erreurs de compilation dans un fichier texte local.

### Exemple

L'exemple suivant enregistre le contenu du panneau Erreurs de compilation dans le fichier nommé `errors.log` qui est stocké dans le dossier `C:\tests` :

```
fl.compilerErrors.save("file:///c:/tests/errors.log");
```

### Voir aussi

[`compilerErrors.clear\(\)`](#)

# Chapitre 8 : Objet ComponentInstance

**Héritage**    [Objet Element](#) > [Objet Instance](#) > [Objet SymbolInstance](#) > [Objet ComponentInstance](#)

## Disponibilité

Flash MX 2004.

## Description

L'objet ComponentInstance est une sous-classe de l'objet SymbolInstance et représente un composant dans une image (voir [Objet SymbolInstance](#))

## Résumé des propriétés

Outre toutes les propriétés de l'[Objet SymbolInstance](#), l'objet ComponentInstance possède les propriétés suivantes :

Propriété	Description
<code>componentInstance.parameters</code>	Lecture seule ; tableau contenant les propriétés ActionScript 2.0, accessibles à l'aide de l'inspecteur des propriétés du composant.

## componentInstance.parameters

## Disponibilité

Flash MX 2004.

## Utilisation

`componentInstance.parameters`

## Description

Propriété en lecture seule ; tableau contenant les propriétés ActionScript 2.0 accessibles à l'aide de l'Inspecteur de propriétés du composant. Voir l'[Objet Parameter](#).

## Exemple

L'exemple suivant montre comment récupérer et définir la propriété `parameters` :

```
var parms = fl.getDocumentDOM().selection[0].parameters;
parms[0].value = "some value";
```

## Voir aussi

[Objet Parameter](#)

# Chapitre 9 : Objet componentsPanel

## Disponibilité

Flash MX 2004.

## Description

L'objet componentsPanel, qui représente le panneau Composants, est une propriété de l'objet flash (fl) et est accessible via `fl.componentsPanel` (voir [Objet flash \(fl\)](#)).

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet componentsPanel :

Méthode	Description
<code>componentsPanel.addItemToDocument()</code>	Ajoute le composant spécifié au document à l'emplacement indiqué.
<code>componentsPanel.reload()</code>	Actualise la liste de composants affichée dans le panneau Composants.

## componentsPanel.addItemToDocument()

## Disponibilité

Flash MX 2004.

## Utilisation

```
componentsPanel.addItemToDocument(position, categoryName, componentName)
```

## Paramètres

**position** Point (par exemple, `{x:0,y:100}`) indiquant l'emplacement où le composant doit être ajouté. Spécifiez le paramètre *position* par rapport au point central du composant, et non par rapport à son point d'alignement (également *point d'origine* ou *point zéro*).

**categoryName** Chaîne spécifiant le nom de la catégorie du composant (par exemple, "Data"). Les noms de catégories valides sont énumérés dans le panneau Composants.

**componentName** Chaîne spécifiant le nom du composant dans la catégorie spécifiée (par exemple, "WebServiceConnector"). Les noms de composants valides sont énumérés dans le panneau Composants.

## Valeur renvoyée

Aucune.

## Description

Ajoute le composant spécifié au document à l'emplacement indiqué.

## Exemple

Les exemples ci-dessous illustrent diverses façons d'utiliser cette méthode :



```
fl.componentsPanel.addItemToDocument({x:0, y:0}, "User Interface", "CheckBox");  
fl.componentsPanel.addItemToDocument({x:0, y:100}, "Data", "WebServiceConnector");  
fl.componentsPanel.addItemToDocument({x:0, y:200}, "User Interface", "Button");
```

## componentsPanel.reload()

### Disponibilité

Flash 8.

### Utilisation

```
componentsPanel.reload()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne `true` si la liste du panneau Composants est actualisée, et `false` dans le cas contraire.

### Description

Méthode qui actualise la liste de composants affichée dans le panneau Composants.

### Exemple

L'exemple suivant actualise le panneau Composants :

```
fl.componentsPanel.reload();
```

# Chapitre 10 : Objet Contour

## Disponibilité

Flash MX 2004.

## Description

Un objet Contour représente une trajectoire en tirets fermée marquant les limites d'une forme.

## Résumé des méthodes

Vous pouvez utiliser la méthode suivante avec l'objet Contour :

Méthode	Description
<code>contour.getHalfEdge()</code>	Renvoie un objet <a href="#">Objet halfEdge</a> sur le contour de la sélection.

## Résumé des propriétés

Vous pouvez utiliser les propriétés suivantes avec l'objet Contour :

Propriété	Description
<code>contour.fill</code>	Objet <a href="#">Objet Fill</a> .
<code>contour.interior</code>	Lecture seule ; sa valeur est <code>true</code> si le contour enferme une zone, <code>false</code> dans le cas contraire.
<code>contour.orientation</code>	Lecture seule ; nombre entier indiquant l'orientation du contour.

## contour.fill

## Disponibilité

Flash CS4 Professional.

## Utilisation

`contour.fill`

## Description

Propriété ; [Objet Fill](#).

## Exemple

En supposant que vous avez un contour avec un remplissage sélectionné, l'exemple suivant affiche la couleur de remplissage du contour dans le panneau Sortie :

```
var insideContour = fl.getDocumentDOM().selection[0].contours[1];
var insideFill = insideContour.fill;
fl.trace(insideFill.color);
```

## contour.getHalfEdge()

### Disponibilité

Flash MX 2004.

### Utilisation

```
contour.getHalfEdge()
```

### Paramètres

Aucun.

### Valeur renvoyée

Objet [Objet halfEdge](#).

### Description

Méthode qui renvoie un objet [Objet halfEdge](#) sur le contour de la sélection.

### Exemple

Cet exemple analyse tous les contours de la forme sélectionnée et affiche les coordonnées des sommets dans le panneau Sortie :

```
// with a shape selected

var elt = fl.getDocumentDOM().selection[0];
elt.beginEdit();

var contourArray = elt.contours;
var contourCount = 0;
for (i=0;i<contourArray.length;i++)
{
    var contour = contourArray[i];
    contourCount++;
    var he = contour.getHalfEdge();

    var iStart = he.id;
    var id = 0;
    while (id != iStart)
    {
        // Get the next vertex.
        var vrt = he.getVertex();

        var x = vrt.x;
        var y = vrt.y;
        fl.trace("vrt: " + x + ", " + y);

        he = he.getNext();
        id = he.id;
    }
}
elt.endEdit();
```

## contour.interior

### Disponibilité

Flash MX 2004.

### Utilisation

`contour.interior`

### Description

Propriété en lecture seule ; la valeur est `true` si le contour enferme une zone ou `false` dans le cas contraire.

### Exemple

Cet exemple analyse tous les contours de la forme sélectionnée et affiche la valeur de la propriété `interior` pour chaque contour dans le panneau Sortie :

```
var elt = fl.getDocumentDOM().selection[0];
elt.beginEdit();

var contourArray = elt.contours;

var contourCount = 0;
for (i=0;i<contourArray.length;i++) {
    var contour = contourArray[i];
    fl.trace("Next Contour, interior:" + contour.interior );
    contourCount++;
}
elt.endEdit();
```

## contour.orientation

### Disponibilité

Flash MX 2004.

### Utilisation

`contour.orientation`

### Description

Propriété en lecture seule ; nombre entier indiquant l'orientation du contour. La valeur de l'entier est -1 si l'orientation est anti-horaire, 1 si l'objet est orienté dans le sens horaire, et 0 si ce contour ne renferme pas de surface.

### Exemple

Cet exemple analyse tous les contours de la forme sélectionnée et affiche la valeur de la propriété `orientation` pour chaque contour dans le panneau Sortie :

```
var elt = fl.getDocumentDOM().selection[0];
elt.beginEdit();

var contourArray = elt.contours;

var contourCount = 0;
for (i=0;i<contourArray.length;i++) {
    var contour = contourArray[i];
    fl.trace("Next Contour, orientation:" + contour.orientation);
    contourCount++;
}
elt.endEdit();
```

# Chapitre 11 : Objet Document

## Disponibilité

Flash MX 2004.

## Description

L'objet Document représente la scène. En d'autres termes, seuls les fichiers FLA sont considérés comme des documents. Pour renvoyer l'objet Document du document actif, utilisez `fl.getDocumentDOM()`.

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet Document :

Méthode	Description
<code>document.addDataToDocument()</code>	Enregistre les données spécifiées avec un document.
<code>document.addDataToSelection()</code>	Enregistre les données spécifiées avec le ou les objets sélectionnés.
<code>document.addFilter()</code>	Applique un filtre aux objets sélectionnés.
<code>document.addItem()</code>	Ajoute à l'objet Document spécifié un élément provenant d'un document ouvert ou d'une bibliothèque.
<code>document.addNewLine()</code>	Ajoute un nouveau tracé entre deux points.
<code>document.addNewOval()</code>	Ajoute un nouvel objet Oval dans le rectangle de délimitation spécifié.
<code>document.addNewPrimitiveOval()</code>	Ajoute une nouvelle primitive ovale adaptée aux limites spécifiées.
<code>document.addNewPrimitiveRectangle()</code>	Ajoute une nouvelle primitive rectangle adaptée aux limites spécifiées.
<code>document.addNewPublishProfile()</code>	Ajoute un nouveau profil de publication et l'active.
<code>document.addNewRectangle()</code>	Ajoute un nouveau rectangle ou rectangle arrondi dans la délimitation spécifiée.
<code>document.addNewScene()</code>	Ajoute une nouvelle séquence ( <a href="#">Objet Timeline</a> ) après la séquence actuellement sélectionnée, puis fait de cette nouvelle séquence la séquence sélectionnée.
<code>document.addNewText()</code>	Insère un nouveau champ de texte vide.
<code>document.align()</code>	Aligne la sélection.
<code>document.allowScreens()</code>	Cette méthode doit être utilisée avant la propriété « <code>document.screenOutline</code> » à la page 138.
<code>document.arrange()</code>	Organise la sélection sur la scène.
<code>document.breakApart()</code>	Effectue une opération de séparation sur la sélection en cours.
<code>document.canEditSymbol()</code>	Indique si le menu Modifier les symboles et ses fonctionnalités sont activés.
<code>document.canRevert()</code>	Détermine s'il est possible d'utiliser la méthode <code>document.revert()</code> ou <code>fl.revertDocument()</code> .

Méthode	Description
<code>document.canTestMovie()</code>	Détermine s'il est possible d'utiliser la méthode <code>document.testMovie()</code> .
<code>document.canTestScene()</code>	Détermine s'il est possible d'utiliser la méthode <code>document.testScene()</code> .
<code>document.changeFilterOrder()</code>	Change l'index du filtre dans la liste Filtres.
<code>document.clipCopy()</code>	Copie la sélection actuelle du document dans le Presse-papiers.
<code>document.clipCut()</code>	Coupe la sélection actuelle du document et la place dans le Presse-papiers.
<code>document.clipPaste()</code>	Colle le contenu du Presse-papiers dans le document.
<code>document.close()</code>	Ferme le document spécifié.
<code>document.convertLinesToFills()</code>	Convertit des lignes en remplissages pour les objets sélectionnés.
<code>document.convertToSymbol()</code>	Convertit le ou les éléments sélectionnés sur la scène en un nouveau symbole.
<code>document.crop()</code>	Sur la base de l'objet de dessin sélectionné qui se trouve au-dessus, recadre tous les objets de dessin de niveau inférieur.
<code>document.debugMovie()</code>	Démarre une session de débogage avec le document.
<code>document.deleteEnvelope()</code>	Supprime l'enveloppe (cadre de sélection contenant un ou plusieurs objets) de l'objet sélectionné.
<code>document.deletePublishProfile()</code>	Supprime le profil actif, s'il existe plusieurs profils.
<code>document.deleteScene()</code>	Supprime la séquence actuelle (Objet Timeline) et, si cette séquence n'était pas la dernière, définit la prochaine comme l'objet Timeline actif.
<code>document.deleteSelection()</code>	Supprime la sélection actuelle sur la scène.
<code>document.disableAllFilters()</code>	Désactive tous les filtres des objets sélectionnés.
<code>document.disableFilter()</code>	Dans la liste Filtres, désactive le filtre spécifié.
<code>document.disableOtherFilters()</code>	Dans la liste Filtres, désactive tous les filtres à l'exception de celui qui se trouve à la position spécifiée.
<code>document.distribute()</code>	Répartit la sélection.
<code>document.distributeToLayers()</code>	Effectue sur la sélection actuelle une opération de répartition vers les calques ; cette méthode est l'équivalent de l'option de menu Répartir vers les calques.
<code>document.documentHasData()</code>	Recherche dans le document des données persistantes portant le nom spécifié.
<code>document.duplicatePublishProfile()</code>	Duplique le profil actuellement actif et donne le focus à ce duplicata.
<code>document.duplicateScene()</code>	Crée une copie de la séquence actuellement sélectionnée, en donnant à cette nouvelle séquence un nom unique et en en faisant la séquence active.
<code>document.duplicateSelection()</code>	Duplique la sélection sur la scène.
<code>document.editScene()</code>	Sélectionne la séquence spécifiée pour modification.

Méthode	Description
<code>document.enableAllFilters()</code>	Active tous les filtres de la liste Filtres pour le ou les objets sélectionnés.
<code>document.enableFilter()</code>	Active le filtre spécifié pour le ou les objets sélectionnés.
<code>document.enterEditMode()</code>	Bascule l'environnement de programmation dans le mode de modification spécifié par le paramètre.
<code>document.exitEditMode()</code>	Quitte le mode Modification de symbole et renvoie le focus au niveau supérieur suivant.
<code>document.exportPNG()</code>	Exporte le document dans un ou plusieurs fichiers PNG.
<code>document.exportPublishProfile()</code>	Exporte le profil actif dans un fichier XML.
<code>document.exportPublishProfileString()</code>	Renvoie une chaîne représentant le profil spécifié, au format XML.
<code>document.exportSWF()</code>	Exporte le document au format Flash SWF.
<code>document.getAlignToDocument()</code>	Récupère la valeur du bouton Sur la scène dans le panneau Aligner.
<code>document.getBlendMode()</code>	Renvoie une chaîne indiquant le mode de fusion pour le ou les objets sélectionnés.
<code>document.getCustomFill()</code>	Récupère l'objet de remplissage (Fill) de la forme sélectionnée ou, selon le paramètre, du panneau Outils et de l'inspecteur Propriétés.
<code>document.getCustomStroke()</code>	Renvoie l'objet trait (Stroke) de la forme sélectionnée ou, selon le paramètre, du panneau Outils et de l'inspecteur Propriétés.
<code>document.getDataFromDocument()</code>	Récupère la valeur des données spécifiées.
<code>document.getElementProperty()</code>	Récupère la propriété <code>Element</code> spécifiée pour la sélection.
<code>document.getElementTextAttr()</code>	Récupère la propriété <code>TextAttrs</code> spécifiée pour les objets Text sélectionnés.
<code>document.getFilters()</code>	Renvoie un tableau contenant la liste des filtres appliqués aux objets actuellement sélectionnés.
<code>document.getMetadata()</code>	Renvoie une chaîne contenant les métadonnées XML associées au document.
<code>document.getMobileSettings()</code>	Renvoie la chaîne transmise à <code>document.setMobileSettings()</code> .
<code>document.getPlayerVersion()</code>	Renvoie une chaîne représentant la version de Flash Player ciblée pour le document spécifié.
<code>document.getSelectionRect()</code>	Récupère les coordonnées du rectangle de délimitation de la sélection.
<code>document.getTextString()</code>	Récupère le texte sélectionné.
<code>document.getTimeline()</code>	Récupère l' <a href="#">Objet Timeline</a> dans le document.
<code>document.getTransformationPoint()</code>	Récupère l'emplacement du point de transformation de la sélection.
<code>document.group()</code>	Convertit la sélection en un groupe.
<code>document.importFile()</code>	Importe un fichier dans le document.



Méthode	Description
<code>document.importPublishProfile()</code>	Importe un profil à partir d'un fichier.
<code>document.importPublishProfileString()</code>	Importe une chaîne XML représentant un profil de publication et la définit en tant que profil actuel.
<code>document.importSWF()</code>	Importe un fichier SWF dans le document.
<code>document.intersect()</code>	Crée un objet de dessin Intersection à partir de tous les objets de dessin sélectionnés.
<code>document.loadCuepointXML()</code>	Charge un fichier XML de points de repère.
<code>document.match()</code>	Donne la même taille à tous les objets sélectionnés.
<code>document.mouseClick()</code>	Effectue un clic de souris à partir de l'outil Sélection.
<code>document.mouseDbClick()</code>	Effectue un double clic de souris à partir de l'outil Sélection.
<code>document.moveSelectedBezierPointsBy()</code>	Si la sélection contient au moins un tracé comptant au moins un point de courbe de Bézier, cette méthode déplace de la valeur spécifiée tous les points de courbe de Bézier de tous les tracés sélectionnés.
<code>document.moveSelectionBy()</code>	Déplace les objets sélectionnés de la distance spécifiée.
<code>document.optimizeCurves()</code>	Optimise le lissage de la sélection, le cas échéant en autorisant plusieurs passages pour obtenir un lissage optimal équivalent au choix de l'option Modifier > Forme > Optimiser.
<code>document.publish()</code>	Publie le document en fonction des paramètres de publication actuels (Fichier > Paramètres de publication) ; cette méthode est l'équivalent de l'option Fichier > Publier.
<code>document.punch()</code>	Sur la base de l'objet de dessin sélectionné le plus élevé, poinçonne tous les objets de dessin de niveau inférieur sélectionnés.
<code>document.removeAllFilters()</code>	Supprime tous les filtres du ou des objets sélectionnés.
<code>document.removeDataFromDocument()</code>	Supprime les données persistantes associées au document et correspondant au nom spécifié.
<code>document.removeDataFromSelection()</code>	Supprime les données persistantes associées à la sélection et correspondant au nom spécifié.
<code>document.removeFilter()</code>	Supprime le filtre spécifié de la liste Filtres pour le ou les objets sélectionnés.
<code>document.renamePublishProfile()</code>	Renomme le profil actuel.
<code>document.renameScene()</code>	Renomme la séquence sélectionnée dans le panneau Séquences.
<code>document.reorderScene()</code>	Place la séquence spécifiée devant une autre séquence spécifiée.
<code>document.resetOvalObject()</code>	Dans l'inspecteur Propriétés, définit toutes les valeurs sur les paramètres par défaut de l'objet Oval.
<code>document.resetRectangleObject()</code>	Dans l'inspecteur Propriétés, définit toutes les valeurs sur les paramètres par défaut de l'objet Rectangle.
<code>document.resetTransformation()</code>	Réinitialise la matrice de transformation ; cette méthode est l'équivalent de l'option de menu Modification > Transformer > Supprimer la transformation.

Méthode	Description
<code>document.revert()</code>	Restaure la dernière version enregistrée du document spécifié ; cette méthode est l'équivalent de l'option de menu Fichier > Rétablir.
<code>document.rotate3DSelection()</code>	Applique une rotation 3D à la sélection.
<code>document.rotateSelection()</code>	Fait pivoter la sélection du nombre de degrés spécifié.
<code>document.save()</code>	Enregistre le document à son emplacement par défaut ; cette méthode est l'équivalent de l'option de menu Fichier > Enregistrer.
<code>document.saveAndCompact()</code>	Enregistre et compresse le fichier ; cette méthode est l'équivalent de l'option de menu Fichier > Enregistrer et compresser.
<code>document.scaleSelection()</code>	Effectue une mise à l'échelle de la sélection de la valeur spécifiée ; cette méthode est l'équivalent de l'outil Transformation libre utilisé pour modifier l'échelle de l'objet.
<code>document.selectAll()</code>	Sélectionne tous les éléments figurant sur la scène ; cette méthode est l'équivalent des raccourcis clavier Ctrl+A (Windows) ou Commande+A (Macintosh), ou de l'option de menu Edition > Sélectionner tout.
<code>document.selectNone()</code>	Désélectionne les éléments sélectionnés.
<code>document.setAlignToDocument()</code>	Définit les préférences afin d'activer <code>document.align()</code> , <code>document.distribute()</code> , <code>document.match()</code> et <code>document.space()</code> pour le document. Cette commande est équivalente à l'activation de l'option Sur la scène du panneau Aligner.
<code>document.setBlendMode()</code>	Définit le mode de fusion pour les objets sélectionnés.
<code>document.setCustomFill()</code>	Définit les paramètres de remplissage pour le panneau Outils, l'inspecteur Propriétés et les formes sélectionnées, le cas échéant.
<code>document.setCustomStroke()</code>	Définit les paramètres de trait pour le panneau Outils, l'inspecteur Propriétés et les formes sélectionnées, le cas échéant.
<code>document.setElementProperty()</code>	Définit la propriété <code>Element</code> spécifiée pour le ou les objets sélectionnés du document.
<code>document.setElementTextAttr()</code>	Définit selon la valeur spécifiée la propriété <code>TextAttrs</code> spécifiée pour les éléments de texte sélectionnés.
<code>document.setFillColor()</code>	Modifie la couleur de remplissage de la sélection avec la couleur spécifiée.
<code>document.setFilterProperty()</code>	Définit la propriété de filtre spécifiée pour le ou les objets sélectionnés.
<code>document.setFilters()</code>	Applique des filtres aux objets sélectionnés.
<code>document.setInstanceAlpha()</code>	Définit l'opacité de l'occurrence.
<code>document.setInstanceBrightness()</code>	Définit la luminosité de l'occurrence.
<code>document.setInstanceTint()</code>	Définit la teinte de l'occurrence.
<code>document.setMetadata()</code>	Définit les métadonnées XML du document spécifié, en remplacement des métadonnées existantes.

Méthode	Description
<code>document.setMobileSettings()</code>	Définit la valeur d'une chaîne de paramètres XML dans un fichier FLA mobile.
<code>document.setOvalObjectProperty()</code>	Spécifie la valeur d'une propriété spécifiée des objets Oval de type primitif.
<code>document.setPlayerVersion()</code>	Définit la version de Flash Player ciblée par le document spécifié.
<code>document.setRectangleObjectProperty()</code>	Spécifie la valeur d'une propriété spécifiée des objets Rectangle de type primitif.
<code>document.setSelectionBounds()</code>	Déplace et redimensionne la sélection en une seule opération.
<code>document.setSelectionRect()</code>	Trace un cadre de sélection rectangulaire sur la scène, à partir des coordonnées spécifiées.
<code>document.setStageVanishingPoint()</code>	Spécifie le point de fuite pour l'affichage des objets 3D.
<code>document.setStageViewAngle()</code>	Spécifie l'angle de perspective pour l'affichage des objets 3D.
<code>document.setStroke()</code>	Définit la couleur, la largeur et le style des traits sélectionnés.
<code>document.setStrokeColor()</code>	Modifie la couleur de trait de la sélection avec la couleur spécifiée.
<code>document.setStrokeSize()</code>	Modifie la taille de trait de la sélection avec la taille spécifiée.
<code>document.setStrokeStyle()</code>	Modifie le style de trait de la sélection avec le style spécifié.
<code>document.setTextRectangle()</code>	Modifie le rectangle de délimitation de l'élément de texte sélectionné en lui donnant la taille spécifiée.
<code>document.setTextSelection()</code>	Définit la sélection de texte du champ de texte sélectionné avec les valeurs spécifiées dans <i>startIndex</i> et <i>endIndex</i> .
<code>document.setTextString()</code>	Insère une chaîne de texte.
<code>document.setTransformationPoint()</code>	Déplace le point de transformation de la sélection.
<code>document.skewSelection()</code>	Incline la sélection de la valeur spécifiée.
<code>document.smoothSelection()</code>	Lisse la courbe de chaque contour de remplissage ou segment incurvé sélectionné.
<code>document.space()</code>	Espace régulièrement les objets sélectionnés.
<code>document.straightenSelection()</code>	Redresse les traits sélectionnés ; cette méthode est l'équivalent du bouton Redresser dans le panneau Outils.
<code>document.swapElement()</code>	Intervertit la sélection actuelle avec la sélection spécifiée.
<code>document.swapStrokeAndFill()</code>	Permute les couleurs de trait et de remplissage.
<code>document.testMovie()</code>	Exécute une opération Tester l'animation sur le document.
<code>document.testScene()</code>	Exécute une opération Tester la séquence sur la séquence active dans le document.
<code>document.traceBitmap()</code>	Convertit le bitmap sélectionné en graphique vectoriel ; cette méthode est l'équivalent de l'option de menu Modification > Bitmap > Tracer le bitmap.
<code>document.transformSelection()</code>	Effectue une transformation générale de la sélection, en appliquant la matrice transmise en arguments.

Méthode	Description
<code>document.translate3DCenter()</code>	Définit la position XYZ autour de laquelle est effectuée la translation ou la rotation de la sélection.
<code>document.translate3DSelection()</code>	Applique une translation 3D à la sélection.
<code>document.unGroup()</code>	Dissocie la sélection.
<code>document.union()</code>	Combine toutes les formes sélectionnées sous forme d'un objet de dessin.
<code>document.unlockAllElements()</code>	Déverrouille tous les éléments verrouillés dans l'image sélectionnée.
<code>document.xmlPanel()</code>	Envoie une boîte de dialogue XMLUI.

### Résumé des propriétés

Vous pouvez utiliser les propriétés suivantes avec l'objet Document :

Propriété	Description
<code>document.accName</code>	Chaîne équivalente au champ Nom du panneau Accessibilité.
<code>document.as3AutoDeclare</code>	Valeur booléenne indiquant si les occurrences placées sur la scène sont automatiquement ajoutées aux classes Timeline (scénario) définies par l'utilisateur.
<code>document.as3Dialect</code>	Chaîne décrivant le « dialecte » d'ActionScript 3.0 utilisé dans le document spécifié.
<code>document.as3ExportFrame</code>	Entier spécifiant dans quelle image exporter les classes ActionScript 3.0.
<code>document.as3StrictMode</code>	Valeur booléenne spécifiant si le compilateur ActionScript 3.0 doit compiler avec l'option du mode strict activée ou non.
<code>document.as3WarningsMode</code>	Valeur booléenne spécifiant si le compilateur ActionScript 3.0 doit compiler avec l'option Mode avertissements activée ou non.
<code>document.asVersion</code>	Entier indiquant la version ActionScript utilisée dans le fichier spécifié.
<code>document.autoLabel</code>	Valeur booléenne équivalente à la case à cocher Etiquetage auto du panneau Accessibilité.
<code>document.backgroundColor</code>	Chaîne, valeur hexadécimale ou entier représentant la couleur d'arrière-plan.
<code>document.currentPublishProfile</code>	Chaîne spécifiant le nom du profil de publication actif pour le document spécifié.
<code>document.currentTimeline</code>	Nombre entier spécifiant l'index du scénario actif.
<code>document.description</code>	Chaîne équivalente au champ Description du panneau Accessibilité.
<code>document.docClass</code>	Spécifie la classe ActionScript 3.0 de niveau supérieur qui est associée au document.
<code>document.externalLibraryPath</code>	Chaîne qui contient la liste des éléments du chemin de bibliothèque externe ActionScript 3.0 du document, spécifiant l'emplacement des fichiers SWC utilisés en tant que bibliothèques partagées par le moteur d'exécution.
<code>document.forceSimple</code>	Valeur booléenne indiquant si les enfants de l'objet spécifié sont accessibles.

Propriété	Description
<code>document.frameRate</code>	Nombre en virgule flottante indiquant le nombre d'images affichées par seconde pendant la lecture du fichier SWF ; cette valeur est 12 par défaut.
<code>document.height</code>	Nombre entier indiquant la hauteur en pixels du document (scène).
<code>document.id</code>	Nombre entier unique (attribué automatiquement) qui identifie un document pendant une session Flash.
<code>document.library</code>	Lecture seule ; <a href="#">Objet library</a> d'un document.
<code>document.libraryPath</code>	Chaîne qui contient la liste des éléments de chemin de la bibliothèque ActionScript 3.0 du document, spécifiant l'emplacement des fichiers SWC ou des dossiers les contenant.
<code>document.livePreview</code>	Valeur booléenne indiquant si la fonction Aperçu en direct est activée.
<code>document.name</code>	Lecture seule ; chaîne représentant le nom d'un document (fichier FLA).
<code>document.path</code>	Lecture seule ; chaîne représentant le chemin d'accès du document dans un format spécifique à la plate-forme.
<code>document.pathURI</code>	Lecture seule ; chaîne représentant le chemin d'accès du document, exprimée sous la forme « fichier:/// URI ».
<code>document.publishProfiles</code>	Lecture seule ; tableau des noms des profils de publication du document.
<code>document.screenOutline</code>	Lecture seule ; objet ScreenOutline actuel du document. A partir de Flash Professional CS5, l'objet ScreenOutline est déconseillé.
<code>document.selection</code>	Tableau des objets sélectionnés dans le document.
<code>document.silent</code>	Valeur booléenne indiquant si l'objet est accessible.
<code>document.sourcePath</code>	Propriété ; chaîne qui contient la liste des éléments du chemin source ActionScript 3.0 du document, spécifiant l'emplacement des fichiers de classe ActionScript.
<code>document.timelines</code>	Lecture seule ; tableau des objets Timeline (voir <a href="#">Objet Timeline</a> ).
<code>document.viewMatrix</code>	Lecture seule ; <a href="#">Objet Matrix</a> .
<code>document.width</code>	Nombre entier indiquant la largeur en pixels du document (scène).
<code>document.zoomFactor</code>	Spécifie le pourcentage de zoom de la scène lors de la programmation.

## document.accName

### Disponibilité

Flash MX 2004.

### Utilisation

`document.accName`

### Description

Propriété ; chaîne équivalente au champ Nom du panneau Accessibilité. Les logiciels de lecture vocale identifient l'objet en prononçant ce nom.

### Exemple

L'exemple suivant définit le nom d'accessibilité du document sur "Main Movie" :

```
fl.getDocumentDOM().accName = "Main Movie";
```

L'exemple suivant récupère le nom d'accessibilité du document :

```
fl.trace(fl.getDocumentDOM().accName);
```

## document.addDataToDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.addDataToDocument(name, type, data)
```

### Paramètres

**name** Chaîne spécifiant le nom des données à ajouter.

**type** Chaîne définissant le type de données à ajouter. Les valeurs possibles sont "integer", "integerArray", "double", "doubleArray", "string" et "byteArray".

**data** Valeur à ajouter. Les types valides dépendent du paramètre *type*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui enregistre les données spécifiées avec un document. Les données sont écrites dans le fichier FLA et sont accessibles en JavaScript à la réouverture du fichier.

### Exemple

L'exemple suivant ajoute un entier ayant la valeur 12 au document actif :

```
fl.getDocumentDOM().addDataToDocument("myData", "integer", 12);
```

L'exemple suivant renvoie la valeur des données nommées "myData" et affiche le résultat dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().getDataFromDocument("myData"));
```

### Voir aussi

[document.getDataFromDocument\(\)](#), [document.removeDataFromDocument\(\)](#)

## document.addDataToSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.addDataToSelection(name, type, data)
```

### Paramètres

**name** Chaîne spécifiant le nom des données persistantes.

**type** Définit le type de données. Les valeurs possibles sont "integer", "integerArray", "double", "doubleArray", "string" et "byteArray".

**data** Valeur à ajouter. Les types valides dépendent du paramètre *type*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui enregistre les données spécifiées avec le ou les objets sélectionnés. Les données sont écrites dans le fichier FLA et sont accessibles en JavaScript à la réouverture du fichier. Seuls les symboles et les bitmaps acceptent des données persistantes.

### Exemple

L'exemple suivant ajoute un entier ayant la valeur 12 à l'objet sélectionné :

```
fl.getDocumentDOM().addDataToSelection("myData", "integer", 12);
```

### Voir aussi

[document.removeDataFromSelection\(\)](#)

## document.addFilter()

### Disponibilité

Flash 8.

### Utilisation

```
document.addFilter(filterName)
```

### Paramètres

**filterName** Chaîne indiquant le filtre à ajouter à la liste Filtres et à activer pour le ou les objets sélectionnés. Les valeurs possibles sont "adjustColorFilter", "bevelFilter", "blurFilter", "dropShadowFilter", "glowFilter", "gradientBevelFilter" et "gradientGlowFilter".

### Valeur renvoyée

Aucune.

### Description

Méthode qui applique un filtre aux objets sélectionnés et le place à la fin de la liste Filtres.

### Exemple

L'exemple suivant applique un effet néon aux objets sélectionnés :

```
fl.getDocumentDOM().addFilter("glowFilter");
```

#### Voir aussi

```
document.changeFilterOrder(), document.disableFilter(), document.enableFilter(),  
document.getFilters(), document.removeFilter(), document.setBlendMode(),  
document.setFilterProperty()
```

## document.addItem()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.addItem(position, item)
```

#### Paramètres

**position** Point indiquant les coordonnées *x* et *y* de l'emplacement auquel le composant doit être ajouté. Ce point est le centre pour un symbole ou le coin supérieur gauche pour une image bitmap ou une vidéo.

**item** Objet Item spécifiant l'élément à ajouter et la bibliothèque dont il est extrait (voir [Objet Item](#)).

#### Valeur renvoyée

Valeur booléenne : *true* en cas de réussite, *false* dans le cas contraire.

#### Description

Méthode qui ajoute à l'objet Document spécifié un élément provenant d'un document ouvert ou d'une bibliothèque.

#### Exemple

L'exemple suivant ajoute le premier élément de la bibliothèque au premier document, à l'emplacement indiqué pour le symbole, le bitmap ou la vidéo sélectionné(e) :

```
var item = fl.documents[0].library.items[0];  
fl.documents[0].addItem({x:0,y:0}, item);
```

L'exemple suivant ajoute le symbole *myMovieClip* au document actif à partir de sa propre bibliothèque :

```
var itemIndex = fl.getDocumentDOM().library.findItemIndex("myMovieClip");  
var theItem = fl.getDocumentDOM().library.items[itemIndex];  
fl.getDocumentDOM().addItem({x:0,y:0}, theItem);
```

L'exemple suivant ajoute le symbole *myMovieClip* du deuxième document du tableau des documents au troisième document de ce tableau :

```
var itemIndex = fl.documents[1].library.findItemIndex("myMovieClip");  
var theItem = fl.documents[1].library.items[itemIndex];  
fl.documents[2].addItem({x:0,y:0}, theItem);
```



## document.addNewLine()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.addNewLine(startPoint, endpoint)
```

### Paramètres

**startPoint** Paire de nombres en virgule flottante indiquant les coordonnées *x* et *y* de départ de la ligne.

**endpoint** Paire de nombres en virgule flottante indiquant les coordonnées *x* et *y* de fin de la ligne.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute un nouveau trajet entre deux points. La méthode utilise les attributs de trait actuels du document et ajoute le trajet dans l'image et le calque actifs. Cette méthode est l'équivalent d'un clic sur l'outil Ligne et du traçage d'une ligne.

### Exemple

L'exemple suivant ajoute une ligne entre le point de début et le point de fin spécifiés :

```
fl.getDocumentDOM().addNewLine({x:216.7, y:122.3}, {x:366.8, y:165.8});
```

## document.addNewOval()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.addNewOval(boundingRectangle [, bSuppressFill [, bSuppressStroke ]])
```

### Paramètres

**boundingRectangle** Rectangle délimitant l'ovale à ajouter. Pour plus d'informations sur le format de *boundingRectangle*, voir la méthode [document.addNewRectangle\(\)](#).

**bSuppressFill** Valeur booléenne. Si elle est définie sur *true*, la méthode crée la forme sans remplissage. La valeur par défaut est *false*. Ce paramètre est facultatif.

**bSuppressStroke** Valeur booléenne. Si elle est *true*, la méthode crée la forme sans trait. La valeur par défaut est *false*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute un nouvel objet Oval dans le rectangle de délimitation spécifié. Cette méthode effectue la même opération que l'outil Oval. La méthode utilise les attributs de trait et de remplissage par défaut actuels du document et ajoute l'ovale dans l'image et le calque actifs. Si les valeurs de *bSuppressFill* et *bSuppressStroke* sont toutes deux définies sur *true*, la méthode est sans effet.

### Exemple

L'exemple suivant ajoute un ovale de 164 pixels de large sur 178 pixels de haut dans la zone définie par les coordonnées spécifiées :

```
fl.getDocumentDOM().addNewOval({left:72,top:50,right:236,bottom:228});
```

L'exemple suivant trace l'ovale sans remplissage :

```
fl.getDocumentDOM().addNewOval({left:72,top:50,right:236,bottom:228}, true);
```

L'exemple suivant trace l'ovale sans trait :

```
fl.getDocumentDOM().addNewOval({left:72,top:50,right:236,bottom:228}, false, true);
```

### Voir aussi

[document.addNewPrimitiveOval\(\)](#)

## document.addNewPrimitiveOval()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.addNewPrimitiveOval( boundingRectangle [, bSpupressFill [, bSuppressStroke ]] )
```

### Paramètres

**boundingRectangle** Rectangle qui délimite la zone de création dans laquelle la nouvelle primitive ovale est ajoutée. Pour plus d'informations sur le format de *boundingRectangle*, voir [document.addNewRectangle\(\)](#).

**bSuppressFill** Valeur booléenne qui, lorsqu'elle est définie sur *true*, oblige la méthode à créer l'ovale sans remplissage. La valeur par défaut est *false*. Ce paramètre est facultatif.

**bSuppressStroke** Valeur booléenne qui, lorsqu'elle est définie sur *true*, oblige la méthode à créer l'ovale sans trait. La valeur par défaut est *false*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute une nouvelle primitive ovale adaptée aux limites spécifiées. Cette méthode effectue la même opération que l'outil Primitive ovale. La primitive ovale utilise les attributs de trait et de remplissage par défaut du document et est ajoutée dans l'image et le calque actifs. Si les valeurs de *bSuppressFill* et *bSuppressStroke* sont toutes deux définies sur *true*, la méthode est sans effet.

### Exemple

L'exemple suivant ajoute des primitives ovales aux coordonnées spécifiées, avec ou sans remplissage et trait :

```
// Add an oval primitive with fill and stroke
fl.getDocumentDOM().addNewPrimitiveOval({left:0,top:0,right:100,bottom:100});
// Add an oval primitive without a fill
fl.getDocumentDOM().addNewPrimitiveOval({left:100,top:100,right:200,bottom:200}, true);
// Add an oval primitive without a stroke
fl.getDocumentDOM().addNewPrimitiveOval({left:200,top:200,right:300,bottom:300}, false, true);
```

### Voir aussi

[document.addNewOval\(\)](#)

## document.addNewPrimitiveRectangle()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.addNewPrimitiveRectangle( boundingRectangle, roundness, [, bSuppressFill [,
bSuppressStroke ]] )
```

### Paramètres

**rect** Rectangle qui délimite la zone de création du nouveau rectangle. Pour plus d'informations sur le format de *boundingRectangle*, voir [document.addNewRectangle\(\)](#).

**roundness** Nombre entier compris entre 0 et 999 représentant le nombre de points utilisés pour spécifier l'arrondi des angles.

**bSuppressFill** Valeur booléenne qui, lorsqu'elle est définie sur `true`, oblige la méthode à créer le rectangle sans remplissage. La valeur par défaut est `false`. Ce paramètre est facultatif.

**bSuppressStroke** Valeur booléenne. Si elle est `true`, la méthode crée le rectangle sans trait. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute une nouvelle primitive rectangle adaptée aux limites spécifiées. Cette méthode effectue la même opération que l'outil Primitive rectangle. La primitive rectangle utilise les attributs de trait et de remplissage par défaut du document et est ajoutée dans l'image et le calque actifs. Si les valeurs de *bSuppressFill* et *bSuppressStroke* sont toutes deux définies sur `true`, la méthode est sans effet.

### Exemple

L'exemple suivant ajoute des primitives rectangle aux coordonnées spécifiées, avec et sans remplissage et trait, et avec des niveaux d'arrondis différents :

```
// Add a rectangle primitive with fill and stroke
fl.getDocumentDOM().addNewPrimitiveRectangle({left:0,top:0,right:100,bottom:100}, 0);
// Add a rectangle primitive without a fill
fl.getDocumentDOM().addNewPrimitiveRectangle({left:100,top:100,right:200,bottom:200}, 20,
true);
// Add a rectangle primitive without a stroke
fl.getDocumentDOM().addNewPrimitiveRectangle({left:200,top:200,right:300,bottom:300},
50,false,true);
```

#### Voir aussi

[document.addNewRectangle\(\)](#)

## document.addNewPublishProfile()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.addNewPublishProfile([profileName])
```

#### Paramètres

**profileName** Nom unique du nouveau profil. Si vous ne spécifiez pas de nom, un nom par défaut est utilisé. Ce paramètre est facultatif.

#### Valeur renvoyée

Nombre entier correspondant à l'index du nouveau profil dans la liste des profils. Renvoie -1 s'il est impossible de créer un nouveau profil.

#### Description

Méthode qui ajoute un nouveau profil de publication et l'active.

#### Exemple

L'exemple suivant ajoute un nouveau profil de publication avec un nom par défaut, puis affiche le nom du profil dans le panneau Sortie :

```
fl.getDocumentDOM().addNewPublishProfile();
fl.outputPanel.trace(fl.getDocumentDOM().currentPublishProfile);
```

L'exemple suivant ajoute un nouveau profil de publication avec le nom "my profile" :

```
fl.getDocumentDOM().addNewPublishProfile("my profile");
```

#### Voir aussi

[document.deletePublishProfile\(\)](#)

## document.addNewRectangle()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.addNewRectangle(boundingRectangle, roundness  
    [, bSuppressFill [, bSuppressStroke]])
```

### Paramètres

**boundingRectangle** Rectangle qui délimite la zone de création du nouveau rectangle, exprimé au format `{left:value1,top:value2,right:value3,bottom:value4}`. Les valeurs `left` et `top` spécifient l'emplacement du coin supérieur gauche (exemple : `left:0,top:0` représente le coin supérieur gauche de la scène), et les valeurs `right` et `bottom` indiquent l'emplacement du coin inférieur droit. De ce fait, la largeur du rectangle correspond à l'écart entre `left` et `right`, et sa hauteur à l'écart entre `top` et `bottom`.

En d'autres termes, les limites du rectangle ne correspondent pas toutes aux valeurs indiquées dans l'inspecteur Propriétés. Les valeurs `left` et `top` correspondent respectivement aux valeurs X et Y de l'inspecteur Propriétés. Toutefois, les valeurs `right` et `bottom` ne correspondent pas aux valeurs L et H de l'inspecteur Propriétés. Par exemple, supposons un rectangle possédant les limites suivantes :

```
{left:10,top:10,right:50,bottom:100}
```

Ce rectangle posséderait les valeurs suivantes dans l'inspecteur Propriétés :

```
X = 10, Y = 10, W = 40, H = 90
```

**roundness** Valeur entière comprise entre 0 et 999 qui représente le niveau d'arrondi des angles. La valeur est exprimée sous la forme d'un nombre de points. Plus elle est élevée, plus l'arrondi est important.

**bSuppressFill** Valeur booléenne. Si elle est définie sur `true`, la méthode crée la forme sans remplissage. La valeur par défaut est `false`. Ce paramètre est facultatif.

**bSuppressStroke** Valeur booléenne. Si elle est `true`, la méthode crée le rectangle sans trait. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute un nouveau rectangle ou un rectangle arrondi dans la délimitation spécifiée. Cette méthode effectue la même opération que l'outil Rectangle. La méthode utilise les attributs de trait et de remplissage par défaut actuels du document et ajoute le rectangle dans l'image et le calque actifs. Si les valeurs de `bSuppressFill` et `bSuppressStroke` sont toutes deux définies sur `true`, la méthode est sans effet.

### Exemple

L'exemple suivant ajoute un rectangle sans angles arrondis aux coordonnées indiquées, de 100 pixels de large sur 100 pixels de haut :

```
fl.getDocumentDOM().addNewRectangle({left:0,top:0,right:100,bottom:100},0);
```

L'exemple suivant ajoute un rectangle, sans angles arrondis ni remplissage, de 100 pixels de large sur 200 pixels de haut :

```
fl.getDocumentDOM().addNewRectangle({left:10,top:10,right:110,bottom:210},0, true);
```

L'exemple suivant ajoute un rectangle, sans angles arrondis ni contour, de 200 pixels de large sur 100 pixels de haut :

```
fl.getDocumentDOM().addNewRectangle({left:20,top:20,right:220,bottom:120},0, false, true);
```

#### Voir aussi

[document.addNewPrimitiveRectangle\(\)](#)

## document.addNewScene()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.addNewScene ( [name] )
```

#### Paramètres

**name** Spécifie le nom de la séquence. Si vous ne spécifiez pas de nom, un nouveau nom de séquence par défaut est créé.

#### Valeur renvoyée

Valeur booléenne, `true` si la séquence a bien été ajoutée, `false` dans le cas contraire.

#### Description

Méthode qui ajoute une nouvelle séquence ([Objet Timeline](#)) après la séquence actuellement sélectionnée, puis fait de cette nouvelle séquence la séquence sélectionnée. Si le nom de séquence indiqué existe déjà, la séquence n'est pas ajoutée et la méthode renvoie une erreur.

#### Exemple

L'exemple suivant ajoute une nouvelle séquence nommée `myScene` après la séquence actuelle dans le document actif. La variable `success` est définie sur `true` après la création de la séquence, sur `false` dans le cas contraire.

```
var success = fl.getDocumentDOM().addNewScene("myScene");
```

L'exemple suivant ajoute une nouvelle séquence en utilisant la convention de nom par défaut. S'il n'existe encore qu'une seule séquence, la nouvelle séquence sera nommée "Scene 2".

```
fl.getDocumentDOM().addNewScene();
```

## document.addNewText()

#### Disponibilité

Flash MX 2004 ; paramètre *text* facultatif ajouté dans Flash CS3 Professional.

#### Utilisation

```
document.addNewText (boundingRectangle [, text ])
```

### Paramètres

**boundingRectangle** Spécifie la taille et l'emplacement du champ de texte. Pour plus d'informations sur le format de *boundingRectangle*, voir la méthode `document.addNewRectangle()`.

**text** Chaîne facultative qui spécifie le texte à placer dans le champ. Si vous omettez ce paramètre, la sélection dans le panneau Outils bascule sur l'outil Texte. Par conséquent, si vous ne voulez pas changer d'outil, transmettez une valeur pour *text*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui insère un nouveau champ de texte et, le cas échéant, y place du texte. Si vous omettez le paramètre *text*, vous pouvez appeler `document.setTextString()` pour remplir le champ de texte.

### Exemple

L'exemple suivant crée un nouveau champ de texte dans le coin supérieur gauche de la scène et y insère la chaîne de texte "Hello World" :

```
fl.getDocumentDOM().addNewText({left:0, top:0, right:100, bottom:100} , "Hello World!" );  
fl.getDocumentDOM().setTextString('Hello World!');
```

### Voir aussi

`document.setTextString()`

## document.align()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.align(alignmode [, bUseDocumentBounds])
```

### Paramètres

**alignmode** Chaîne indiquant comment aligner la sélection. Les valeurs possibles sont "left", "right", "top", "bottom", "vertical center" et "horizontal center".

**bUseDocumentBounds** Valeur booléenne. Si elle est `true`, la méthode aligne les objets sélectionnés sur les limites du document. Sinon, la méthode utilise les limites des objets sélectionnés. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui aligne la sélection.

### Exemple

L'exemple suivant aligne les objets à gauche et sur les limites de la scène. Cette commande est équivalente à l'activation de l'option Sur la scène du panneau Aligner, puis à un clic sur le bouton Aligner à gauche :

```
fl.getDocumentDOM().align("left", true);
```

### Voir aussi

[document.distribute\(\)](#), [document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

## document.allowScreens()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.allowScreens()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si `document.screenOutline` peut être utilisé en toute sécurité, `false` dans le cas contraire.

### Description

Méthode, utilisée avant la propriété `document.screenOutline`. Si la méthode renvoie la valeur `true`, il est possible d'accéder sans risque à la propriété `document.screenOutline`. Flash affiche une erreur si la propriété `document.screenOutline` est appelée dans un document ne comportant pas d'écrans.

### Exemple

L'exemple suivant détermine s'il est possible d'utiliser les méthodes `screens` dans le document actif :

```
if(fl.getDocumentDOM().allowScreens()) {  
    fl.trace("screen outline is available.");  
}  
else {  
    fl.trace("whoops, no screens.");  
}
```

### Voir aussi

[document.screenOutline](#)

## document.arrange()

### Disponibilité

Flash MX 2004.



### Utilisation

`document.arrange(arrangeMode)`

### Paramètres

**arrangeMode** Spécifie la direction dans laquelle la sélection doit être déplacée. Les valeurs gérées sont "back", "backward", "forward" et "front". Offre les mêmes possibilités que les options équivalentes du menu Modification > Disposer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui organise la sélection sur la scène. Cette méthode s'applique uniquement aux objets qui ne sont pas des formes.

### Exemple

L'exemple suivant place la sélection au premier plan :

```
fl.getDocumentDOM().arrange("front");
```

## document.as3AutoDeclare

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.as3AutoDeclare`

### Description

Propriété ; valeur booléenne indiquant si les occurrences placées sur la scène sont automatiquement ajoutées aux classes Timeline (scénario) définies par l'utilisateur. La valeur par défaut est `true`.

### Exemple

L'exemple suivant précise que les occurrences placées sur la scène dans le document actif doivent être ajoutées manuellement aux classes Timeline (scénario) définies par l'utilisateur.

```
fl.getDocumentDOM().as3AutoDeclare=false;
```

## document.as3Dialect

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.as3Dialect`

### Description

Propriété ; chaîne décrivant le « dialecte » d'ActionScript 3.0 utilisé dans le document spécifié. La valeur par défaut est "AS3". Si vous souhaitez autoriser des classes de prototype, telles qu'elles ont été admises dans les précédentes spécifications ECMAScript, définissez cette valeur sur "ES".

### Exemple

L'exemple suivant précise que le dialecte utilisé dans le document actif est ECMAScript :

```
fl.getDocumentDOM().as3Dialect="ES";
```

### Voir aussi

[document.asVersion](#)

## document.as3ExportFrame

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.as3ExportFrame`

### Description

Propriété ; nombre entier spécifiant dans quelle image exporter les classes ActionScript 3.0. Par défaut, les classes sont exportées dans l'image 1.

### Exemple

L'exemple suivant change l'image dans laquelle les classes sont exportées, l'image 5 remplace l'image par défaut 1.

```
var myDocument = fl.getDocumentDOM();  
fl.outputPanel.trace("'Export classes in frame:' value before modification is " +  
myDocument.as3ExportFrame);  
myDocument.as3ExportFrame = 5;  
fl.outputPanel.trace("'Export classes in frame:' value after modification is " +  
myDocument.as3ExportFrame);
```

## document.as3StrictMode

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.as3StrictMode`

### Description

Propriété ; valeur booléenne indiquant si le compilateur ActionScript 3.0 doit compiler avec l'option du mode strict activée (`true`) ou non (`false`). Le Mode strict signale les avertissements comme des erreurs, ce qui signifie que la compilation échouera tant que ces erreurs persisteront. La valeur par défaut est `true`.

### Exemple

L'exemple suivant désactive l'option Mode strict du compilateur.

```
var myDocument = fl.getDocumentDOM();
fl.outputPanel.trace("Strict Mode value before modification is " + myDocument.as3StrictMode);
myDocument.as3StrictMode = false;
fl.outputPanel.trace("Strict Mode value after modification is " + myDocument.as3StrictMode);
```

### Voir aussi

[document.as3WarningsMode](#)

## document.as3WarningsMode

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.as3WarningsMode`

### Description

Propriété ; valeur booléenne spécifiant si le compilateur ActionScript 3.0 doit compiler avec l'option Mode avertissements activée (`true`) ou non (`false`). Le mode avertissements déclenche des alarmes supplémentaires devant être signalées, utiles lors de la mise à jour du code ActionScript 2.0 vers ActionScript 3.0 pour repérer d'éventuelles incompatibilités. La valeur par défaut est `true`.

### Exemple

L'exemple suivant désactive l'option Mode avertissements du compilateur.

```
var myDocument = fl.getDocumentDOM();
fl.outputPanel.trace("Warnings Mode value before modification is " +
myDocument.as3WarningsMode);
myDocument.as3WarningsMode = false;
fl.outputPanel.trace("Warnings Mode value after modification is " +
myDocument.as3WarningsMode);
```

### Voir aussi

[document.as3StrictMode](#)

## document.asVersion

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.asVersion`

### Description

Propriété ; nombre entier spécifiant quelle version d'ActionScript est utilisée dans le document spécifié. Les valeurs possibles sont 1, 2 et 3.

Pour déterminer la version de Flash Player ciblée pour le document spécifié, utilisez `document.getPlayerVersion()` ; cette méthode renvoie une chaîne, elle est donc utilisable par les lecteurs Flash® Lite™.

### Exemple

L'exemple suivant définit la version d'ActionScript dans le document actif sur ActionScript 2.0 si elle est actuellement définie sur ActionScript 1.0.

```
if (fl.getDocumentDOM().asVersion == 1) {  
    fl.getDocumentDOM().asVersion = 2;  
}
```

### Voir aussi

`document.as3Dialect`, `document.getPlayerVersion()`

## document.autoLabel

### Disponibilité

Flash MX 2004.

### Utilisation

`document.autoLabel`

### Description

Propriété ; valeur booléenne équivalente à la case à cocher Etiquetage auto du panneau Accessibilité. Cette propriété permet d'indiquer à Flash d'étiqueter automatiquement les objets sur la scène avec le texte qui leur est associé.

### Exemple

L'exemple suivant récupère la valeur de la propriété `autoLabel` et affiche le résultat dans le panneau Sortie :

```
var isAutoLabel = fl.getDocumentDOM().autoLabel;  
fl.trace(isAutoLabel);
```

L'exemple suivant définit la propriété `autoLabel` sur `true`, indiquant ainsi à Flash que les objets présents sur la scène doivent être étiquetés automatiquement :

```
fl.getDocumentDOM().autoLabel = true;
```

# document.backgroundColor

## Disponibilité

Flash MX 2004.

## Utilisation

`document.backgroundColor`

## Description

Propriété ; couleur de l'arrière-plan, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

## Exemple

L'exemple suivant définit la couleur d'arrière-plan sur Noir :

```
fl.getDocumentDOM().backgroundColor = '#000000';
```

# document.breakApart()

## Disponibilité

Flash MX 2004.

## Utilisation

`document.breakApart()`

## Paramètres

Aucun.

## Valeur renvoyée

Aucune.

## Description

Méthode qui effectue une opération de séparation sur la sélection en cours.

## Exemple

L'exemple suivant sépare la sélection :

```
fl.getDocumentDOM().breakApart();
```

## document.canEditSymbol()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.canEditSymbol()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si le menu et la fonctionnalité Modifier les symboles sont disponibles, `false` dans le cas contraire.

### Description

Méthode qui indique si le menu Modifier les symboles et ses fonctionnalités sont activés. Cette méthode n'indique pas s'il est possible de modifier la sélection. Elle ne doit pas être utilisée pour tester si l'appel de `fl.getDocumentDOM().enterEditMode()` est autorisé.

### Exemple

L'exemple suivant affiche dans le panneau Sortie l'état du menu Modifier les symboles et ses fonctionnalités :

```
fl.trace("fl.getDocumentDOM().canEditSymbol() returns: " +  
fl.getDocumentDOM().canEditSymbol());
```

## document.canRevert()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.canRevert()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si vous pouvez utiliser les méthodes `document.revert()` ou `fl.revertDocument()`, `false` dans le cas contraire.

### Description

Méthode qui détermine s'il est possible d'utiliser la méthode `document.revert()` ou `fl.revertDocument()`.

### Exemple

L'exemple suivant vérifie s'il est possible de restaurer le document actif avec sa dernière version enregistrée. Dans l'affirmative, `fl.getDocumentDOM().revert()` rétablit la dernière version enregistrée.

```
if (fl.getDocumentDOM().canRevert()) {  
    fl.getDocumentDOM().revert();  
}
```

## document.canTestMovie()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.canTestMovie()`

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si vous pouvez utiliser la méthode `document.testMovie()`, `false` dans le cas contraire.

### Description

Méthode qui détermine s'il est possible d'utiliser la méthode `document.testMovie()`.

### Exemple

L'exemple suivant vérifie si `fl.getDocumentDOM().testMovie()` peut être utilisée. Si c'est le cas, la méthode est ensuite appelée.

```
if (fl.getDocumentDOM().canTestMovie()) {  
    fl.getDocumentDOM().testMovie();  
}
```

### Voir aussi

`document.canTestScene()`, `document.testScene()`

## document.canTestScene()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.canTestScene()`

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si vous pouvez utiliser la méthode `document.testScene()`, `false` dans le cas contraire.

### Description

Méthode qui détermine s'il est possible d'utiliser la méthode `document.testScene()`.

### Exemple

L'exemple suivant vérifie d'abord si `fl.getDocumentDOM().testScene()` est utilisable. Si c'est le cas, la méthode est ensuite appelée.

```
if (fl.getDocumentDOM().canTestScene()) {  
    fl.getDocumentDOM().testScene();  
}
```

### Voir aussi

`document.canTestMovie()`, `document.testMovie()`

## document.changeFilterOrder()

### Disponibilité

Flash 8.

### Utilisation

`document.changeFilterOrder(oldIndex, newIndex)`

### Paramètres

**oldIndex** Nombre entier représentant la position d'index (en base 0) du filtre à déplacer au sein de la liste Filtres.

**newIndex** Nombre entier représentant la nouvelle position d'index du filtre dans la liste.

### Valeur renvoyée

Aucune.

### Description

Méthode qui modifie la position d'index du filtre dans la liste Filtres. Tout filtre qui suit ou précède *newIndex* est décalé en conséquence vers le haut ou vers le bas. Par exemple, avec les filtres indiqués ci-dessous, après la commande `fl.getDocumentDOM().changeFilterOrder(3, 0)`, les filtres sont réorganisés comme suit :

Avant	Après
blurFilterdropShadowFilterglowFiltergradientBevelFilter	gradientBevelFilterblurFilterdropShadowFilterglowFilter

Si vous appelez ensuite la commande `fl.getDocumentDOM().changeFilterOrder(0, 2)`, les filtres sont réorganisés comme suit :

Avant	Après
gradientBevelFilterblurFilterdropShadowFilterglowFilter	blurFilterdropShadowFiltergradientBevelFilterglowFilter



### Exemple

L'exemple suivant déplace le Filtre actuellement en seconde position dans la liste Filtres et le fait passer en première position :

```
fl.getDocumentDOM().changeFilterOrder(1,0);
```

### Voir aussi

[document.addFilter\(\)](#), [document.disableFilter\(\)](#), [document.enableFilter\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#), [Objet Filter](#)

## document.clipCopy()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.clipCopy()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui copie la sélection actuelle du document dans le Presse-papiers.

Pour copier une chaîne dans le Presse-papiers, utilisez [fl.clipCopyString\(\)](#).

### Exemple

L'exemple suivant copie la sélection actuelle du document dans le Presse-papiers :

```
fl.getDocumentDOM().clipCopy();
```

### Voir aussi

[document.clipCut\(\)](#), [document.clipPaste\(\)](#)

## document.clipCut()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.clipCut()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui coupe la sélection actuelle du document et la place dans le Presse-papiers.

### Exemple

L'exemple suivant coupe la sélection actuelle du document et la place dans le Presse-papiers :

```
fl.getDocumentDOM().clipCut();
```

### Voir aussi

[document.clipCopy\(\)](#), [document.clipPaste\(\)](#), [fl.clipCopyString\(\)](#)

## document.clipPaste()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.clipPaste([bInPlace])
```

### Paramètres

**bInPlace** Valeur booléenne. Si elle est `true`, la méthode effectue une opération de Coller en place. La valeur par défaut est `false`. La méthode effectue dans ce cas une opération Coller dans le centre du document. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui colle le contenu du Presse-papiers dans le document.

### Exemple

L'exemple suivant colle le contenu du Presse-papiers au centre du document :

```
fl.getDocumentDOM().clipPaste();
```

L'exemple suivant colle le contenu du Presse-papiers à sa place dans le document actif :

```
fl.getDocumentDOM().clipPaste(true);
```

### Voir aussi

[document.clipCopy\(\)](#), [document.clipCut\(\)](#), [fl.clipCopyString\(\)](#)

## document.close()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.close ([bPromptToSaveChanges] )
```

### Paramètres

**bPromptToSaveChanges** Valeur booléenne. Si elle est `true`, la méthode affiche une boîte de dialogue pour inviter l'utilisateur à enregistrer les modifications du document. Si *bPromptToSaveChanges* est défini sur `false`, l'utilisateur n'est pas invité à enregistrer les éventuelles modifications. La valeur par défaut est `true`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ferme le document spécifié.

### Exemple

L'exemple suivant ferme le document actif et affiche une boîte de dialogue pour inviter l'utilisateur à enregistrer les modifications :

```
fl.getDocumentDOM().close();
```

L'exemple suivant ferme le document sans enregistrer les modifications :

```
fl.getDocumentDOM().close(false);
```

## document.convertLinesToFills()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.convertLinesToFills()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui convertit des lignes en remplissages pour les objets sélectionnés.

### Exemple

L'exemple suivant convertit en remplissages les lignes sélectionnées :

```
fl.getDocumentDOM().convertLinesToFills();
```

## document.convertToSymbol()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.convertToSymbol(type, name, registrationPoint)
```

### Paramètres

**type** Chaîne spécifiant le type de symbole à créer. Les valeurs possibles sont "movie clip", "button" et "graphic".

**name** Chaîne spécifiant le nom du nouveau symbole. Ce nom doit être unique. Il est possible de transmettre une chaîne vide. La méthode crée alors un nom de symbole unique.

**registration point** Spécifie le point d'alignement, qui représente les coordonnées 0,0 du symbole. Les valeurs possibles sont : "top left", "top center", "top right", "center left", "center", "center right", "bottom left", "bottom center" et "bottom right".

### Valeur renvoyée

Un objet pour le nouveau symbole, ou null s'il est impossible de créer le symbole.

### Description

Méthode qui convertit le ou les éléments sélectionnés dans la scène en un nouveau symbole. Pour plus d'informations sur la définition des liaisons et des propriétés des actifs partagés pour un symbole, voir la section [Objet Item](#).

### Exemple

Les exemples suivants créent un symbole de clip avec le nom spécifié, un symbole de bouton avec le nom spécifié, et un symbole de clip avec un nom par défaut :

```
newMc = fl.getDocumentDOM().convertToSymbol("movie clip", "mcSymbolName", "top left");  
newButton = fl.getDocumentDOM().convertToSymbol("button", "btnSymbolName", "bottom right");  
newClipWithDefaultName = fl.getDocumentDOM().convertToSymbol("movie clip", "", "top left");
```

## document.crop()

### Disponibilité

Flash 8.

### Utilisation

```
document.crop()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne : `true` en cas de réussite, `false` dans le cas contraire.

### Description

Méthode qui, sur la base de l'objet de dessin sélectionné le plus élevé, recadre tous les objets de dessin de niveau inférieur. La méthode renvoie `false` s'il n'existe pas d'objets de dessin sélectionnés, ou si l'un des éléments sélectionnés n'est pas un objet de dessin.

### Exemple

L'exemple suivant recadre les objets actuellement sélectionnés :

```
fl.getDocumentDOM().crop();
```

### Voir aussi

`document.deleteEnvelope()`, `document.intersect()`, `document.punch()`, `document.union()`,  
`shape.isDrawingObject`

## document.currentPublishProfile

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.currentPublishProfile
```

### Description

Propriété ; chaîne spécifiant le nom du profil de publication actif pour le document spécifié.

### Exemple

L'exemple suivant ajoute un nouveau profil de publication avec le nom par défaut, puis affiche le nom du profil dans le panneau Sortie :

```
fl.getDocumentDOM().addNewPublishProfile();  
fl.outputPanel.trace(fl.getDocumentDOM().currentPublishProfile);
```

L'exemple suivant remplace le profil sélectionné par le profil par défaut ("Default") :

```
fl.getDocumentDOM().currentPublishProfile = "Default";
```

## document.currentTimeline

### Disponibilité

Flash MX 2004.

### Utilisation

`document.currentTimeline`

### Description

Propriété ; nombre entier spécifiant l'index du scénario actif. Il est possible de définir le scénario actif en changeant la valeur de cette propriété ; l'effet est pratiquement identique à un appel de `document.editScene()`. La seule différence réside dans le fait qu'aucun message d'erreur n'est affiché si l'index du scénario n'est pas valide. La propriété n'est pas modifiée, ce qui provoque un échec silencieux.

### Exemple

L'exemple suivant affiche l'index du scénario actif :

```
var myCurrentTL = fl.getDocumentDOM().currentTimeline;
fl.trace("The index of the current timeline is: " + myCurrentTL);
```

L'exemple suivant change le scénario actif en remplaçant le scénario principal par une séquence nommée "myScene".

```
var i = 0;
var curTimelines = fl.getDocumentDOM().timelines;
while(i < fl.getDocumentDOM().timelines.length) {
    if(curTimelines[i].name == "myScene") {
        fl.getDocumentDOM().currentTimeline = i;
    }
    ++i;
}
```

### Voir aussi

`document.getTimeline()`

## document.debugMovie()

### Disponibilité

Flash Professional CS5

### Utilisation

`document.DebugMovie([Boolean abortIfErrorsExist])`

### Description

Méthode ; invoque la commande Déboguer l'animation sur le document.

### Paramètres

**abortIfErrorsExist** Valeur booléenne ; la valeur par défaut est false. Si cette valeur est définie sur true, la session de débogage ne démarre pas et la fenêtre .swf ne s'ouvre pas en cas d'erreurs du compilateur. Les erreurs du compilateur n'annulent pas la commande.

### Exemple

L'exemple suivant ouvre le document en cours en mode de débogage, mais abandonne l'opération en cas d'erreurs du compilateur :

```
fl.getDocumentDOM().debugMovie(1);
```

## document.deleteEnvelope()

### Disponibilité

Flash 8.

### Utilisation

```
document.deleteEnvelope()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne : `true` en cas de réussite, `false` dans le cas contraire.

### Description

Méthode qui supprime l'enveloppe (cadre contenant un ou plusieurs objets) de l'objet sélectionné.

### Exemple

L'exemple suivant supprime l'enveloppe des objets sélectionnés :

```
fl.getDocumentDOM().deleteEnvelope();
```

### Voir aussi

[document.crop\(\)](#), [document.intersect\(\)](#), [document.punch\(\)](#), [document.union\(\)](#), [shape.isDrawingObject](#)

## document.deletePublishProfile()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.deletePublishProfile()
```

### Paramètres

Aucun.

### Valeur renvoyée

Nombre entier qui représente l'index du nouveau profil actif. Si aucun nouveau profil n'est disponible, la méthode conserve le profil actuel et renvoie son index.

### Description

Méthode qui supprime le profil actif, s'il existe plusieurs profils. Il doit rester au moins un profil.

### Exemple

L'exemple suivant supprime le profil actif s'il existe plusieurs profils et affiche l'index du nouveau profil actif :

```
alert(fl.getDocumentDOM().deletePublishProfile());
```

#### Voir aussi

```
document.addNewPublishProfile()
```

## document.deleteScene()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.deleteScene()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Valeur booléenne, `true` si la séquence a bien été supprimée, `false` dans le cas contraire.

#### Description

Méthode qui supprime la séquence actuelle ([Objet Timeline](#)) et, si cette séquence n'était pas la dernière, définit la prochaine comme objet Timeline actif. Si la séquence supprimée était la dernière, le premier objet est défini comme objet Timeline actif. S'il n'existe qu'un seul objet Timeline (séquence), la méthode renvoie la valeur `false`.

#### Exemple

En supposant qu'il existe trois séquences (`Scene0`, `Scene1` et `Scene2`) dans le document actif, l'exemple suivant active la séquence `Scene2`, puis la supprime :

```
fl.getDocumentDOM().editScene(2);  
var success = fl.getDocumentDOM().deleteScene();
```

## document.deleteSelection()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.deleteSelection()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Aucune.



### Description

Méthode qui supprime la sélection actuelle sur la scène. Un message d'erreur est affiché s'il n'existe pas de sélection.

### Exemple

L'exemple suivant supprime la sélection du document :

```
fl.getDocumentDOM().deleteSelection();
```

## document.description

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.description
```

### Description

Propriété ; chaîne équivalente au champ Description du panneau Accessibilité. Cette description est lue par les logiciels de lecture vocale.

### Exemple

L'exemple suivant définit la description du document :

```
fl.getDocumentDOM().description= "This is the main movie";
```

L'exemple suivant récupère la description du document et l'affiche dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().description);
```

## document.disableAllFilters()

### Disponibilité

Flash 8.

### Utilisation

```
document.disableAllFilters()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui désactive tous les filtres des objets sélectionnés.

### Exemple

L'exemple suivant désactive tous les filtres des objets sélectionnés :

```
fl.getDocumentDOM().disableAllFilters();
```

### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#),  
[document.disableOtherFilters\(\)](#), [document.enableAllFilters\(\)](#), [document.getFilters\(\)](#),  
[document.removeAllFilters\(\)](#), [Objet Filter](#)

## document.disableFilter()

### Disponibilité

Flash 8.

### Utilisation

```
document.disableFilter(filterIndex)
```

### Paramètres

**filterIndex** Nombre entier représentant l'index (en base 0) du filtre dans la liste Filtres.

### Valeur renvoyée

Aucune.

### Description

Méthode qui dans la liste Filtres, désactive le filtre spécifié.

### Exemple

L'exemple suivant désactive des objets sélectionnés les premier et troisième filtres (valeur d'index 0 et 2) de la liste Filtres.

```
fl.getDocumentDOM().disableFilter(0);  
fl.getDocumentDOM().disableFilter(2);
```

### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableAllFilters\(\)](#),  
[document.disableOtherFilters\(\)](#), [document.enableFilter\(\)](#), [document.getFilters\(\)](#),  
[document.removeFilter\(\)](#), [Objet Filter](#)

## document.disableOtherFilters()

### Disponibilité

Flash 8.

### Utilisation

```
document.disableOtherFilters(enabledFilterIndex)
```

### Paramètres

**enabledFilterIndex** Nombre entier représentant l'index (en base 0) du filtre qui doit rester activé après que tous les autres filtres ont été désactivés.

### Valeur renvoyée

Aucune.

### Description

Méthode qui, dans la liste Filtres, désactive tous les filtres à l'exception de celui qui se trouve à la position spécifiée.

### Exemple

L'exemple suivant désactive tous les filtres, à l'exception du deuxième filtre de la liste (index 1).

```
fl.getDocumentDom().disableOtherFilters(1);
```

### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableAllFilters\(\)](#),  
[document.disableFilter\(\)](#), [document.enableFilter\(\)](#), [document.getFilters\(\)](#),  
[document.removeFilter\(\)](#), [Objet Filter](#)

## document.distribute()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.distribute(distributemode [, bUseDocumentBounds])
```

### Paramètres

**distributemode** Chaîne spécifiant où répartir les objets sélectionnés. Les valeurs possibles sont "left edge", "horizontal center", "right edge", "top edge", "vertical center" et "bottom edge".

**bUseDocumentBounds** Valeur booléenne. Si elle est `true`, la méthode répartit les objets sélectionnés dans les limites du document. Sinon, la méthode utilise les limites des objets sélectionnés. La valeur par défaut est `false`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui répartit la sélection.

### Exemple

L'exemple suivant répartit les objets sélectionnés sur leurs bords supérieurs :

```
fl.getDocumentDOM().distribute("top edge");
```

L'exemple suivant répartit les objets sélectionnés sur leurs bords supérieurs et définit explicitement le paramètre *bUseDocumentBounds* :

```
fl.getDocumentDOM().distribute("top edge", false);
```

L'exemple suivant répartit les objets sélectionnés sur leurs bords supérieurs, en les alignant sur les limites du document :

```
fl.getDocumentDOM().distribute("top edge", true);
```

#### **Voir aussi**

[document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

## **document.distributeToLayers()**

#### **Disponibilité**

Flash MX 2004.

#### **Utilisation**

```
document.distributeToLayers()
```

#### **Paramètres**

Aucun.

#### **Valeur renvoyée**

Aucune.

#### **Description**

Méthode qui effectue sur la sélection actuelle une opération de répartition vers les calques ; cette méthode est l'équivalent de l'option de menu Répartir vers les calques. Un message d'erreur est affiché s'il n'existe pas de sélection.

#### **Exemple**

L'exemple suivant répartit la sélection dans les calques :

```
fl.getDocumentDOM().distributeToLayers();
```

## **document.docClass**

#### **Disponibilité**

Flash CS3 Professional.

#### **Utilisation**

```
document.docClass
```

#### **Description**

Propriété ; chaîne spécifiant la classe ActionScript 3.0 de niveau supérieur qui sera associée au document. Si le document n'est pas configuré pour utiliser ActionScript 3.0, cette propriété est ignorée.

### Exemple

L'exemple suivant spécifie que la classe ActionScript 3.0 associée au document est `com.mycompany.ManagerClass` et qu'elle est définie dans `com/mycompany/ManagerClass.as` :

```
var myDocument = fl.getDocumentDOM();  
// set the property  
myDocument.docClass = "com.mycompany.ManagerClass";  
// get the property  
fl.outputPanel.trace("document.docClass has been set to " + myDocument.docClass);
```

### Voir aussi

[item.linkageBaseClass](#)

## document.documentHasData()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.documentHasData (name)
```

### Paramètres

**name** Chaîne spécifiant le nom des données à vérifier.

### Valeur renvoyée

Valeur booléenne, `true` si le document comporte des données persistantes, `false` dans le cas contraire.

### Description

Méthode qui recherche dans le document des données persistantes ayant le nom spécifié.

### Exemple

L'exemple suivant recherche dans le document des données persistantes ayant le nom `"myData"` :

```
var hasData = fl.getDocumentDOM().documentHasData ("myData");
```

### Voir aussi

[document.addDataToDocument\(\)](#), [document.getDataFromDocument\(\)](#),  
[document.removeDataFromDocument\(\)](#)

## document.duplicatePublishProfile()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.duplicatePublishProfile ([profileName])
```

### Paramètres

**profileName** Chaîne spécifiant le nom unique du profil dupliqué. Si vous ne spécifiez pas de nom, le nom par défaut est utilisé. Ce paramètre est facultatif.

### Valeur renvoyée

Nombre entier qui est l'index du nouveau profil dans la liste des profils. Renvoie -1 s'il est impossible de dupliquer le profil.

### Description

Méthode qui duplique le profil actuellement actif et donne le focus à ce duplicata.

### Exemple

L'exemple suivant duplique le profil actuellement actif et affiche l'index du nouveau profil dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().duplicatePublishProfile("dup profile"));
```

## document duplicateScene()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.duplicateScene()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si la séquence a bien été dupliquée, `false` dans le cas contraire.

### Description

Méthode qui crée une copie de la séquence actuellement sélectionnée, en donnant à cette nouvelle séquence un nom unique et en en faisant la séquence active.

### Exemple

L'exemple suivant duplique la seconde séquence du document actif :

```
fl.getDocumentDOM().editScene(1); //Set the middle scene to current scene.  
var success = fl.getDocumentDOM().duplicateScene();
```

## document duplicateSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.duplicateSelection()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui duplique la sélection sur la scène.

### Exemple

L'exemple suivant duplique la sélection. Cette opération est semblable à un appui sur la touche Alt suivi d'un clic et d'un glisser-déposer de l'objet :

```
fl.getDocumentDOM().duplicateSelection();
```

## document.editScene()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.editScene(index)
```

### Paramètres

**index** Nombre entier de base zéro indiquant la séquence à modifier.

### Valeur renvoyée

Aucune.

### Description

Méthode qui sélectionne la séquence spécifiée pour modification.

### Exemple

En supposant qu'il existe trois séquences (Scene0, Scene1 et Scene2) dans le document actif, l'exemple suivant active la séquence Scene2, puis la supprime :

```
fl.getDocumentDOM().editScene(2);  
fl.getDocumentDOM().deleteScene();
```

## document.enableAllFilters()

### Disponibilité

Flash 8.

### Utilisation

```
document.enableAllFilters()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui active tous les filtres de la liste Filtres pour les objets sélectionnés.

### Exemple

L'exemple suivant active tous les filtres de la liste Filtres pour les objets sélectionnés :

```
fl.getDocumentDOM().enableAllFilters();
```

### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableAllFilters\(\)](#),  
[document.enableFilter\(\)](#), [document.getFilters\(\)](#), [document.removeAllFilters\(\)](#), [Objet Filter](#)

## document.enableFilter()

### Disponibilité

Flash 8.

### Utilisation

```
document.enableFilter(filterIndex)
```

### Paramètres

**filterIndex** Nombre entier indiquant l'index (en base 0) dans la liste Filtres du filtre à activer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui active le filtre spécifié pour le ou les objets sélectionnés.

### Exemple

L'exemple suivant active le second filtre des objets sélectionnés :

```
fl.getDocumentDOM().enableFilter(1);
```

### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#),  
[document.enableAllFilters\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#), [Objet Filter](#)



## document.enterEditMode()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.enterEditMode([editMode])
```

### Paramètres

**editMode** Chaîne qui spécifie le mode de modification. Les valeurs gérées sont "inPlace" ou "newWindow". Si aucun paramètre n'est spécifié, le mode par défaut est celui de modification de symbole. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui bascule l'environnement de programmation dans le mode de modification spécifié par le paramètre. Si aucun paramètre n'est spécifié, le mode par défaut est Modification de symbole. Le résultat est semblable à un clic droit sur le symbole pour afficher le menu contextuel, puis au choix de l'option Modifier.

### Exemple

L'exemple suivant met Flash en mode Modification en place pour le symbole sélectionné :

```
fl.getDocumentDOM().enterEditMode('inPlace');
```

L'exemple suivant met Flash en mode Modifier dans une nouvelle fenêtre pour le symbole sélectionné :

```
fl.getDocumentDOM().enterEditMode('newWindow');
```

### Voir aussi

[document.exitEditMode\(\)](#)

## document.exitEditMode()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.exitEditMode()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui quitte le mode Modification de symbole et renvoie le focus au niveau supérieur suivant. Par exemple, si vous modifiez un symbole inséré dans un autre symbole, cette méthode remonte d'un niveau à partir du symbole en cours de modification, soit au niveau du symbole parent.

### Exemple

L'exemple suivant quitte le mode Modification de symbole :

```
fl.getDocumentDOM().exitEditMode();
```

### Voir aussi

[document.enterEditMode\(\)](#)

## document.exportPNG()

### Disponibilité

Flash 8.

### Utilisation

```
document.exportPNG([fileURI [, bCurrentPNGSettings [, bCurrentFrame]])
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le nom du fichier à exporter. Si *fileURI* est une chaîne vide ou non spécifiée, Flash affiche la boîte de dialogue Exporter l'animation.

**bCurrentPNGSettings** Valeur booléenne indiquant s'il faut utiliser les paramètres de publication PNG actuels (*true*) ou bien afficher la boîte de dialogue Exporter (*false*). Ce paramètre est facultatif. La valeur par défaut est *false*.

**bCurrentFrame** Valeur booléenne indiquant s'il faut exporter uniquement l'image actuelle (*true*) ou bien exporter toutes les images, chacune dans un fichier PNG distinct (*false*). Ce paramètre est facultatif. La valeur par défaut est *false*.

### Valeur renvoyée

Valeur booléenne, *true* si le fichier a bien été exporté sous forme de fichier PNG, *false* dans le cas contraire.

### Description

Méthode qui exporte le document sous forme d'un ou plusieurs fichiers PNG. Si *fileURI* est spécifié et que le fichier existe déjà, il est écrasé sans avertissement.

### Exemple

L'exemple suivant exporte l'image actuelle du document actif vers myFile.png, en utilisant les paramètres de publication PNG actuels :

```
fl.getDocumentDOM().exportPNG("file:///C:/myProject/myFile.png", true, true);
```

## document.exportPublishProfile()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.exportPublishProfile(fileURI)
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le chemin d'accès du fichier XML dans lequel le profil est exporté.

### Valeur renvoyée

Aucune.

### Description

Méthode qui exporte le profil actif dans un fichier XML.

### Exemple

L'exemple suivant exporte le profil actif dans le fichier profile.xml, qui réside dans le dossier /Documents and Settings/utilisateur/Desktop sur le lecteur C :

```
fl.getDocumentDOM().exportPublishProfile('file:///C:/Documents and  
Settings/username/Desktop/profile.xml');
```

### Voir aussi

```
document.exportPublishProfileString(), document.importPublishProfile()
```

## document.exportPublishProfileString()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.exportPublishProfileString( [profileName] )
```

### Paramètres

**profileName** Chaîne qui spécifie le nom du profil à exporter dans une chaîne XML. Ce paramètre est facultatif.

### Valeur renvoyée

Chaîne XML.

### Description

Méthode qui renvoie une chaîne spécifiant, au format XML, le profil spécifié. Si vous ne transmettez pas de valeur pour *profileName*, le profil actuel est exporté.

### Exemple

L'exemple suivant stocke une chaîne XML représentant le profil actuel dans une variable nommée `profileXML`, puis l'affiche dans le panneau Sortie :

```
var profileXML=fl.getDocumentDOM().exportPublishProfileString();  
fl.trace(profileXML);
```

### Voir aussi

`document.exportPublishProfile()`, `document.importPublishProfileString()`

## document.exportSWF()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.exportSWF([fileURI [, bCurrentSettings]])
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le nom du fichier exporté. Si *fileURI* est vide ou non spécifié, Flash affiche la boîte de dialogue Exporter l'animation. Ce paramètre est facultatif.

**bCurrentSettings** Valeur booléenne. Si elle est `true`, Flash utilise les paramètres de publication en SWF actuels. Sinon, Flash affiche la boîte de dialogue Exporter au format Flash Player. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui exporte le document au format Flash SWF.

### Exemple

L'exemple suivant exporte le document à l'emplacement spécifié avec les paramètres de publication actuels :

```
fl.getDocumentDOM().exportSWF("file:///C:/Documents and  
Settings/joe_user/Desktop/qwerty.swf");
```

L'exemple suivant affiche les boîtes de dialogue Exporter l'animation, puis Exporter au format Flash Player, et exporte le document selon les paramètres spécifiés :

```
fl.getDocumentDOM().exportSWF("", true);
```

L'exemple suivant affiche la boîte de dialogue Exporter l'animation, puis exporte le document selon les paramètres spécifiés :

```
fl.getDocumentDOM().exportSWF();
```

## document.externalLibraryPath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`document.externalLibraryPath`

### Description

Propriété ; chaîne qui contient la liste des éléments du chemin de bibliothèque externe ActionScript 3.0 du document, spécifiant l'emplacement des fichiers SWC utilisés au moment de l'exécution en tant que bibliothèques partagées. Les éléments de la chaîne sont délimités par des points-virgules. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Fichier > Paramètres de publication, puis en cliquant sur le bouton Paramètres de Script ActionScript 3.0 dans l'onglet Flash.

### Exemple

L'exemple suivant définit le chemin de bibliothèque externe du document sur "." et "../mySWCLibrary" :

```
var myDocument = fl.getDocumentDOM();
myDocument.externalLibraryPath = ".;../mySWCLibrary";
fl.trace(myDocument.externalLibraryPath);
```

### Voir aussi

[document.libraryPath](#), [document.sourcePath](#), [fl.externalLibraryPath](#)

## document.forceSimple

### Disponibilité

Flash MX 2004.

### Utilisation

`document.forceSimple`

### Description

Propriété ; valeur booléenne indiquant si les enfants de l'objet spécifié sont accessibles. Cette propriété est équivalente à l'inverse du paramètre Rendre les objets enfant accessibles du panneau Accessibilité. Ainsi, si `forceSimple` est `true`, son effet est similaire à la désactivation de l'option Rendre les objets enfant accessibles. Si `forceSimple` est `false`, son effet est similaire à l'activation de l'option Rendre les objets enfant accessibles.

### Exemple

L'exemple suivant donne à la variable `areChildrenAccessible` la valeur de la propriété `forceSimple` ; une valeur `false` signifie que les enfants sont accessibles :

```
var areChildrenAccessible = fl.getDocumentDOM().forceSimple;
```

L'exemple suivant modifie la propriété `forceSimple` pour rendre les enfants du document accessibles :

```
fl.getDocumentDOM().forceSimple = false;
```

## document.frameRate

### Disponibilité

Flash MX 2004.

### Utilisation

`document.frameRate`

### Description

Propriété ; nombre en virgule flottante indiquant le nombre d'images affichées par seconde pendant la lecture du fichier SWF ; la valeur par défaut est 12. Cette propriété équivaut à définir la cadence dans la boîte de dialogue Propriétés du document (Modification > Document) dans le fichier FLA.

### Exemple

L'exemple suivant définit la cadence de 25,5 images par seconde :

```
fl.getDocumentDOM().frameRate = 25.5;
```

## document.getAlignToDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.getAlignToDocument()`

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si l'alignement des objets par rapport à la scène est défini dans les préférences, `false` dans le cas contraire.

### Description

Méthode qui a le même effet que la récupération de la valeur du bouton Sur la scène dans le panneau Aligner. La préférence ainsi récupérée peut être utilisée sur ce document pour les méthodes `document.align()`, `document.distribute()`, `document.match()` et `document.space()`.

### Exemple

L'exemple suivant récupère la valeur du bouton Sur la scène dans le panneau Aligner. Si la valeur renvoyée est `true`, le bouton Sur la scène est activé, sinon, il est désactivé.

```
var isAlignToDoc = fl.getDocumentDOM().getAlignToDocument();  
fl.getDocumentDOM().align("left", isAlignToDoc);
```

### Voir aussi

`document.setAlignToDocument()`

## document.getBlendMode()

### Disponibilité

Flash 8.

### Utilisation

```
document.getBlendMode()
```

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne indiquant le mode de fusion pour les objets sélectionnés. Si plusieurs objets aux modes de fondu distincts sont sélectionnés, la chaîne reflète le mode de fusion de l'objet dont la profondeur est la plus élevée.

***Remarque :** la valeur renvoyée est imprévisible si certains objets sélectionnés ne gèrent pas les modes de fondu ou si leur mode de fusion est défini sur la valeur "normal".*

### Description

Méthode qui renvoie une chaîne indiquant le mode de fusion pour les objets sélectionnés.

### Exemple

L'exemple suivant affiche le nom du mode de fusion dans le panneau Sortie :

```
fl.trace(fl.getDocumentDom().getBlendMode());
```

## document.getCustomFill()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getCustomFill([objectToFill])
```

### Paramètres

**objectToFill** Chaîne indiquant l'emplacement de l'objet Fill (remplissage). Les valeurs suivantes sont possibles :

- "toolbar" renvoie l'objet Fill (remplissage) du panneau Outils et de l'inspecteur Propriétés.
- "selection" renvoie l'objet Fill (remplissage) de la sélection.

Si vous omettez ce paramètre, la valeur par défaut est "selection". S'il n'y a aucune sélection, la méthode renvoie `undefined`. Ce paramètre est facultatif.

### Valeur renvoyée

**Objet Fill** spécifié par le paramètre *objectToFill*, en cas de réussite, `undefined` dans le cas contraire.

### Description

Méthode qui récupère l'objet Fill (remplissage) de la forme sélectionnée ou, selon le paramètre, du panneau Outils et de l'inspecteur Propriétés.

### Exemple

L'exemple suivant récupère l'objet Fill (remplissage) de la sélection, puis change la couleur de la sélection en blanc :

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = '#FFFFFF';
fill.style = "solid";
fl.getDocumentDOM().setCustomFill(fill);
```

L'exemple suivant renvoie l'objet de remplissage du panneau Outils et de l'inspecteur Propriétés, puis active un dégradé linéaire dans le nuancier :

```
var fill = fl.getDocumentDOM().getCustomFill("toolbar");
fill.style = "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray = [0, 100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

### Voir aussi

[document.setCustomFill\(\)](#)

## document.getCustomStroke()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getCustomStroke([locationOfStroke])
```

### Paramètres

**locationOfStroke** Chaîne indiquant l'emplacement de l'objet trait. Les valeurs suivantes sont possibles :

- "toolbar" renvoie l'objet Stroke du panneau Outils et de l'inspecteur Propriétés, si elle est définie.
- "selection" renvoie l'objet Stroke de la sélection, si elle est définie.

Si vous omettez ce paramètre, la valeur par défaut est "selection". S'il n'y a aucune sélection, la méthode renvoie `undefined`. Ce paramètre est facultatif.

### Valeur renvoyée

[Objet Stroke](#) spécifié par le paramètre *locationOfStroke* en cas de réussite, `undefined` dans le cas contraire.

### Description

Renvoie l'objet Stroke de la forme sélectionnée ou, selon le paramètre, du panneau Outils et de l'inspecteur Propriétés.

### Exemple

L'exemple suivant renvoie les paramètres de trait actuels de la sélection et définit l'épaisseur de trait sur 2 :



```
var stroke = fl.getDocumentDOM().getCustomStroke("selection");
stroke.thickness = 2;
fl.getDocumentDOM().setCustomStroke(stroke);
```

L'exemple suivant renvoie les paramètres de trait actuels du panneau Outils et de l'inspecteur Propriétés, puis change la couleur de trait en rouge :

```
var stroke = fl.getDocumentDOM().getCustomStroke("toolbar");
stroke.color = "#FF0000";
fl.getDocumentDOM().setCustomStroke(stroke);
```

#### Voir aussi

[document.setCustomStroke\(\)](#)

## document.getDataFromDocument()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.getDataFromDocument (name)
```

#### Paramètres

**name** Chaîne spécifiant le nom des données à renvoyer.

#### Valeur renvoyée

Les données spécifiées.

#### Description

Méthode qui récupère la valeur des données spécifiées. Le type renvoyé dépend du type des données stockées.

#### Exemple

L'exemple suivant ajoute une valeur entière de 12 au document actif et appelle cette méthode pour afficher la valeur dans le panneau Sortie :

```
fl.getDocumentDOM().addDataToDocument("myData", "integer", 12);
fl.trace(fl.getDocumentDOM().getDataFromDocument("myData"));
```

#### Voir aussi

[document.addDataToDocument\(\)](#), [document.documentHasData\(\)](#), [document.removeDataFromDocument\(\)](#)

## document.getElementProperty()

#### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getElementProperty(propertyName)
```

### Paramètres

**propertyName** Chaîne spécifiant le nom de la propriété Element dont la valeur est recherchée.

### Valeur renvoyée

La valeur de la propriété spécifiée. Renvoie `null` si la propriété est dans un état indéterminé, par exemple lorsque plusieurs éléments possédant des valeurs différentes pour cette propriété sont sélectionnés. Renvoie `undefined` si la propriété n'est pas une propriété valide de l'élément sélectionné.

### Description

Méthode qui récupère la propriété Element spécifiée pour la sélection. La liste des valeurs possibles est indiquée dans le tableau Résumé des propriétés de l'[Objet Element](#).

### Exemple

L'exemple suivant récupère le nom (name) de la propriété Element de la sélection :

```
// elementName = the instance name of the selected object.  
var elementName = fl.getDocumentDOM().getElementProperty("name");
```

### Voir aussi

[document.setElementProperty\(\)](#)

## document.getElementTextAttr()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getElementTextAttr(attrName [, startIndex [, endIndex]])
```

### Paramètres

**attrName** Chaîne spécifiant le nom de la propriété TextAttrs à renvoyer. La liste des noms de propriété et des valeurs possibles est indiquée dans le tableau Résumé des propriétés de l'[Objet TextAttrs](#).

**startIndex** Nombre entier spécifiant l'index du premier caractère, 0 (zéro) étant l'index de la première position. Ce paramètre est facultatif.

**endIndex** Nombre entier qui spécifie l'index du dernier caractère. Ce paramètre est facultatif.

### Valeur renvoyée

Si un champ de texte est sélectionné, la propriété est renvoyée si une seule valeur est utilisée dans tout le texte. Renvoie `undefined` si le champ de texte comporte plusieurs valeurs pour cette propriété. Si plusieurs champs de texte sont sélectionnés, et si toutes les valeurs d'alignement de texte sont égales, la méthode renvoie cette valeur. Si plusieurs champs de texte sont sélectionnés, mais que toutes les valeurs d'alignement de texte ne sont pas égales, la méthode renvoie `undefined`. Si les arguments facultatifs ne sont pas transmis, ces règles s'appliquent à la plage de texte sélectionnée, ou à tout le champ de texte si le texte n'est pas en cours de modification. Si seul le paramètre *startIndex* est transmis, la propriété du caractère situé à droite de l'index est renvoyée, sous réserve que tous les objets Text sélectionnés aient une valeur identique. Si *startIndex* et *endIndex* sont transmis, la valeur renvoyée correspond à toute la plage de caractères comprise entre *startIndex* et le caractère précédant *endIndex*.

### Description

Méthode qui récupère la propriété `TextAttrs` spécifiée pour les objets Text sélectionnés. Les objets sélectionnés qui ne sont pas des champs de texte sont ignorés. La liste des noms de propriété et des valeurs possibles est indiquée dans le tableau Résumé des propriétés de l'Objet `TextAttrs`. Voir aussi `document.setElementTextAttr()`.

### Exemple

L'exemple suivant récupère la taille des champs de texte sélectionnés :

```
fl.getDocumentDOM().getElementTextAttr("size");
```

L'exemple suivant récupère la couleur du caractère d'indice 3 des champs de texte sélectionnés :

```
fl.getDocumentDOM().getElementTextAttr("fillColor", 3);
```

L'exemple suivant récupère le nom de la police de caractères du texte à partir de l'index 2 et jusqu'à l'index 10 (non inclus) des champs de texte sélectionnés :

```
fl.getDocumentDOM().getElementTextAttr("face", 2, 10);
```

## document.getFilters()

### Disponibilité

Flash 8.

### Utilisation

```
document.getFilters()
```

### Paramètres

Aucun.

### Valeur renvoyée

Tableau contenant la liste des filtres appliqués aux objets actuellement sélectionnés.

### Description

Méthode qui renvoie un tableau contenant la liste des filtres appliqués aux objets actuellement sélectionnés. Si plusieurs objets sont sélectionnés et que leurs filtres ne sont pas identiques, cette méthode renvoie la liste des filtres appliqués au premier objet sélectionné.

### Exemple

Voir `document.setFilters()`.

### Voir aussi

`document.addFilter()`, `document.changeFilterOrder()`, `document.setFilters()`, [Objet Filter](#)

## document.getMetadata()

### Disponibilité

Flash 8.

### Utilisation

`document.getMetadata()`

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne contenant les métadonnées XML associées au document, ou une chaîne vide s'il n'existe pas de métadonnées.

### Description

Méthode qui renvoie une chaîne contenant les métadonnées XML associées au document, ou une chaîne vide s'il n'existe pas de métadonnées.

### Exemple

L'exemple suivant affiche dans le panneau Sortie les métadonnées XML du document actif :

```
fl.trace("XML Metadata is :" + fl.getDocumentDOM().getMetadata());
```

### Voir aussi

`document.setMetadata()`

## document.getMobileSettings()

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.getMobileSettings()`

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne représentant les paramètres XML pour le document. Si aucune valeur n'est spécifiée, une chaîne vide est renvoyée.

### Description

Méthode qui renvoie les paramètres XML mobiles pour le document.

### Exemple

L'exemple suivant affiche la chaîne des paramètres XML pour le document actif :

```
fl.trace(fl.getDocumentDOM().getMobileSettings());  
//traces a string like the following"<? xml version="1.0" encoding="UTF-16" standalone="no"  
?><mobileSettings> <contentType id="standalonePlayer" name="Standalone Player"/>  
<testDevices> <testDevice id="1170" name="Generic Phone" selected="yes"/> </testDevices>  
<outputMsgFiltering info="no" trace="yes" warning="yes"/> <testWindowState height="496"  
splitterClosed="No" splitterXPos="400" width="907"/> </mobileSettings>"
```

### Voir aussi

[document.setMobileSettings\(\)](#)

## document.getPlayerVersion()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
document.getPlayerVersion()
```

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne représentant la version de Flash Player spécifiée par l'intermédiaire de `document.setPlayerVersion()`. Si aucune valeur n'est définie, renvoie la valeur spécifiée dans la boîte de dialogue Paramètres de publication.

### Description

Méthode qui renvoie une chaîne représentant la version de Flash Player ciblée pour le document spécifié. Pour obtenir la liste des valeurs pouvant être renvoyées par cette méthode, voir la méthode [document.setPlayerVersion\(\)](#).

Pour déterminer la version d'ActionScript visée dans le fichier spécifié, utilisez [document.asVersion](#).

### Exemple

L'exemple suivant illustre le ciblage des versions du lecteur spécifiées pour le document actif, puis la récupération de ces valeurs :

```
fl.getDocumentDOM().setPlayerVersion("6");  
var version = fl.getDocumentDOM().getPlayerVersion();  
fl.trace(version) // displays "6"  
fl.getDocumentDOM().setPlayerVersion("FlashPlayer10");  
var version = fl.getDocumentDOM().getPlayerVersion();  
fl.trace(version) // displays "FlashPlayer10"
```

#### Voir aussi

[document.setPlayerVersion\(\)](#)

## document.getSelectionRect()

#### Disponibilité

Flash MX 2004.

#### Utilisation

`document.getSelectionRect()`

#### Paramètres

Aucun.

#### Valeur renvoyée

Le rectangle de délimitation de la sélection actuelle, ou 0 s'il n'existe pas de sélection. Pour plus d'informations sur le format de la valeur renvoyée, voir la méthode [document.addNewRectangle\(\)](#).

#### Description

Méthode qui récupère les coordonnées du rectangle de délimitation de la sélection. Pour une sélection non rectangulaire, c'est le plus petit rectangle encadrant la sélection entière qui est renvoyé. Le rectangle est basé sur l'espace du document ou, en mode de modification, sur le point d'alignement (également *point d'origine* ou *point zéro*) du symbole en cours de modification.

#### Exemple

L'exemple suivant récupère le rectangle de délimitation de la sélection actuelle, puis affiche ses propriétés :

```
var newRect = fl.getDocumentDOM().getSelectionRect();  
var outputStr = "left: " + newRect.left + " top: " + newRect.top + " right: " + newRect.right  
+ " bottom: " + newRect.bottom;  
alert(outputStr);
```

#### Voir aussi

[document.selection](#), [document.setSelectionRect\(\)](#)

## document.getTextString()

#### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getTextString([startIndex [, endIndex]])
```

### Paramètres

**startIndex** Nombre entier qui spécifie l'index du premier caractère à récupérer. Ce paramètre est facultatif.

**endIndex** Nombre entier qui spécifie l'index du dernier caractère à récupérer. Ce paramètre est facultatif.

### Valeur renvoyée

Chaîne qui contient le texte sélectionné.

### Description

Méthode qui récupère le texte sélectionné. Si les paramètres facultatifs ne sont pas transmis, la sélection de texte actuelle est utilisée. Si le texte n'est pas en cours de modification, toute la chaîne de texte est renvoyée. Si seul le paramètre *startIndex* est transmis, la méthode renvoie la chaîne qui commence à cet index et se termine à la fin du champ. Si *startIndex* et *endIndex* sont transmis, la valeur renvoyée correspond à toute la plage de caractères comprise entre *startIndex* et le caractère précédant *endIndex*.

Si plusieurs champs de texte sont sélectionnés, une concaténation de toutes les chaînes est renvoyée.

### Exemple

L'exemple suivant récupère le contenu des champs de texte sélectionnés :

```
fl.getDocumentDOM().getTextString();
```

L'exemple suivant récupère le contenu des champs de texte sélectionnés à partir du caractère d'indice 5 :

```
fl.getDocumentDOM().getTextString(5);
```

L'exemple suivant récupère la chaîne à partir du caractère d'index 2 et jusqu'au caractère d'index 10 non compris :

```
fl.getDocumentDOM().getTextString(2, 10);
```

### Voir aussi

[document.setTextString\(\)](#)

## document.getTimeline()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getTimeline()
```

### Paramètres

Aucun.

### Valeur renvoyée

L'objet Timeline actuel.

### Description

Méthode qui récupère l'[Objet Timeline](#) dans le document. Le scénario actuel peut être la séquence actuelle, le symbole en cours de modification, ou l'écran actuel.

### Exemple

L'exemple suivant récupère l'objet Timeline et renvoie le nombre d'images du calque le plus long :

```
var longestLayer = fl.getDocumentDOM().getTimeline().frameCount;  
fl.trace("The longest layer has" + longestLayer + "frames");
```

L'exemple suivant passe au mode de modification en place pour le symbole sélectionné sur la scène et insère une image dans le scénario du symbole.

```
fl.getDocumentDOM().enterEditMode("inPlace");  
fl.getDocumentDOM().getTimeline().insertFrames();
```

L'exemple suivant récupère l'objet Timeline et affiche son nom :

```
var timeline = fl.getDocumentDOM().getTimeline();  
alert(timeline.name);
```

### Voir aussi

[document.currentTimeline](#), [document.timelines](#), [symbolItem.timeline](#)

## document.getTransformationPoint()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.getTransformationPoint()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un point (par exemple, {x:10,y:20}, sachant que x et y sont des nombres à virgule flottante) qui précise la position du point de transformation (également *point d'origine* ou *point zéro*) dans le système de coordonnées de l'élément sélectionné.

### Description

Méthode qui récupère l'emplacement du point de transformation de la sélection. Le point de transformation peut être utilisé pour des permutations, par exemple rotation et inclinaison.

**Remarque :** les points de transformation sont définis par rapport à différents emplacements, en fonction du type d'élément sélectionné. Pour plus d'informations, voir la méthode [document.setTransformationPoint\(\)](#).



### Exemple

L'exemple suivant récupère l'emplacement du point de transformation de la sélection. La propriété `transPoint.x` donne la coordonnée *x* du point de transformation. La propriété `transPoint.y` donne la coordonnée *y* du point de transformation.

```
var transPoint = fl.getDocumentDOM().getTransformationPoint();
```

### Voir aussi

[document.setTransformationPoint\(\)](#), [element.getTransformationPoint\(\)](#)

## document.group()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.group()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui convertit la sélection en un groupe.

### Exemple

L'exemple suivant convertit en groupe les objets de la sélection actuelle :

```
fl.getDocumentDOM().group();
```

### Voir aussi

[document.unGroup\(\)](#)

## document.height

### Disponibilité

Flash MX 2004.

### Utilisation

`document.height`

### Description

Propriété ; nombre entier indiquant la hauteur en pixels du document (scène).

### Exemple

L'exemple suivant définit la hauteur de la scène sur 400 pixels :

```
fl.getDocumentDOM().height = 400;
```

### Voir aussi

[document.width](#)

## document.id

### Disponibilité

Flash CS3 Professional.

### Utilisation

`document.id`

### Description

Propriété en lecture seule ; nombre entier unique (attribué automatiquement) qui identifie un document lors d'une session Flash. Utilisez cette propriété conjointement avec [fl.findDocumentDOM\(\)](#) afin de spécifier un document particulier pour une action.

### Exemple

L'exemple suivant affiche l'ID de document du document actif :

```
fl.trace("Current doc's internal ID is: " + fl.getDocumentDOM().id);
```

### Voir aussi

[fl.findDocumentDOM\(\)](#)

## document.importFile()

### Disponibilité

Flash 8.

### Utilisation

`document.importFile(fileURI [, importToLibrary])`

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le nom du fichier à importer.

**importToLibrary** Valeur booléenne indiquant s'il faut importer le fichier uniquement dans la bibliothèque du document (`true`) ou bien en placer également une copie sur la scène (`false`). La valeur par défaut est `false`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui importe un fichier dans un document. Cette méthode a le même effet que la commande Importer dans la bibliothèque ou Importer vers la scène. Pour importer un profil de publication, utilisez `document.importPublishProfile()`.

### Exemple

L'exemple suivant permet à l'utilisateur de rechercher un fichier à importer vers la scène :

```
var dom = fl.getDocumentDOM();  
var URI = fl.browseForFileURL("select", "Import File");  
dom.importFile(URI);
```

### Voir aussi

`document.importSWF()`, `fl.browseForFileURL()`

## document.importPublishProfile()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.importPublishProfile( fileURI )
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le chemin d'accès du fichier XML définissant le profil à importer.

### Valeur renvoyée

Nombre entier qui est l'index du profil importé dans la liste des profils. Renvoie -1 s'il est impossible d'importer le profil.

### Description

Méthode qui importe un profil à partir d'un fichier.

### Exemple

L'exemple suivant importe le profil contenu dans le fichier profile.xml et affiche son index dans la liste des profils :

```
alert(fl.getDocumentDOM().importPublishProfile('file:///C:/Documents and  
Settings/janeUser/Desktop/profile.xml'));
```

## document.importPublishProfileString()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.importPublishProfileString(xmlString)
```

### Paramètres

**xmlString** Chaîne contenant les données XML à importer en tant que profil actuel.

### Valeur renvoyée

Valeur booléenne, `true` si la chaîne a bien été importée, `false` dans le cas contraire.

### Description

Méthode qui importe une chaîne XML représentant un profil de publication et la définit en tant que profil actif. Pour générer une chaîne XML à importer, utilisez `document.exportPublishProfileString()` avant cette méthode.

### Exemple

Dans l'exemple suivant, le profil par défaut est exporté sous forme de chaîne XML. La commande JavaScript standard `replace` est utilisée pour modifier la chaîne XML. La chaîne est alors importée et le paramètre de sortie `ActionScript 3` par défaut est défini sur `ActionScript 1`.

```
var profileXML=fl.getDocumentDOM().exportPublishProfileString('Default');  
fl.trace(profileXML);  
var newProfileXML = profileXML.replace("<ActionScriptVersion>3</ActionScriptVersion>",  
"<ActionScriptVersion>1</ActionScriptVersion>");  
fl.getDocumentDOM().importPublishProfileString(newProfileXML);
```

## document.importSWF()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.importSWF(fileURI)
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le nom du fichier SWF à exporter.

### Valeur renvoyée

Aucune.

### Description

Méthode qui importe un fichier SWF dans le document. Cette méthode effectue une opération identique à l'utilisation de la commande Importer pour spécifier un fichier SWF. Dans Flash8 et les versions ultérieures, vous pouvez également utiliser `document.importFile()` pour importer un fichier SWF (ainsi que d'autres types de fichiers).

### Exemple

L'exemple suivant importe le fichier "mySwf.swf" depuis le dossier de configuration de Flash :

```
fl.getDocumentDOM().importSWF(fl.configURI+"mySwf.swf");
```

**Voir aussi**

`document.importFile()`

## document.intersect()

**Disponibilité**

Flash 8.

**Utilisation**

`document.intersect()`

**Paramètres**

Aucun.

**Valeur renvoyée**

Valeur booléenne : `true` en cas de réussite, `false` dans le cas contraire.

**Description**

Méthode qui crée un objet de dessin Intersection à partir de tous les objets de dessin sélectionnés. La méthode renvoie `false` s'il n'existe pas d'objets de dessin sélectionnés, ou si l'un des éléments sélectionnés n'est pas un objet de dessin.

**Exemple**

L'exemple suivant crée un objet de dessin Intersection à partir de tous les objets de dessin sélectionnés :

```
fl.getDocumentDOM().intersect();
```

**Voir aussi**

`document.crop()`, `document.deleteEnvelope()`, `document.punch()`, `document.union()`,  
`shape.isDrawingObject`

## document.library

**Disponibilité**

Flash MX 2004.

**Utilisation**

`document.library`

**Description**

Propriété en lecture seule ; [Objet library](#) d'un document.

**Exemple**

L'exemple suivant récupère la bibliothèque du document actif.

```
var myCurrentLib = fl.getDocumentDOM().library;
```

En supposant que le document actuellement actif n'est pas `fl.documents[1]`, l'exemple suivant récupère la bibliothèque non active ou une bibliothèque ouverte à l'aide de la commande Fichier > Ouvrir comme bibliothèque externe :

```
var externalLib = fl.documents[1].library;
```

## document.libraryPath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`document.libraryPath`

### Description

Propriété ; chaîne qui contient la liste des éléments de chemin de la bibliothèque ActionScript 3.0 du document, spécifiant l'emplacement des fichiers SWC ou des dossiers contenant les fichiers SWC. Les éléments de la chaîne sont délimités par des points-virgules. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Fichier > Paramètres de publication, puis en cliquant sur le bouton Paramètres de Script ActionScript 3.0 dans l'onglet Flash.

### Exemple

L'exemple suivant ajoute le dossier ../Files au chemin de bibliothèque du document, puis affiche ce chemin dans le panneau Sortie :

```
var myDoc = fl.getDocumentDOM();
fl.trace(myDoc.libraryPath);
myDoc.libraryPath = "../Files;" + myDoc.libraryPath;
fl.trace(myDoc.libraryPath);
```

### Voir aussi

[document.externalLibraryPath](#), [document.sourcePath](#), [fl.libraryPath](#)

## document.livePreview

### Disponibilité

Flash MX 2004.

### Utilisation

`document.livePreview`

### Description

Propriété ; valeur booléenne indiquant si la fonction Aperçu en direct est activée. Si cette propriété est `true`, les composants apparaissent sur la scène tels qu'ils apparaîtront dans le contenu Flash publié, avec leur taille approximative. Si elle est `false`, seuls les contours des composants apparaissent. La valeur par défaut est `true`.

### Exemple

L'exemple suivant définit la fonction Aperçu en direct sur `false` :

```
fl.getDocumentDOM().livePreview = false;
```

## document.loadCuepointXML()

### Disponibilité

Flash Professional CS5

### Utilisation

```
document.loadCuepointXML(String URI)
```

### Paramètres

**URI** Chaîne ; le chemin absolu au fichier XML de points de repère.

### Description

Méthode ; charge un fichier XML de points de repère. Le format et la DTD du fichier XML sont les mêmes que ceux importés et exportés par l'inspecteur des propriétés des points de repère. La valeur renvoyée est la même que la chaîne sérialisée dans la propriété Point de repère de l'objet contenant l'occurrence d'un composant FLVPlayback.

### Exemple

Dans l'exemple suivant, le fichier XML de points de repère est situé à l'emplacement `C:\\testCuePoints.xml` :

```
var cuePoints = fl.getDocumentDOM().LoadCuepointXML("c:\\testCuePoints.xml");
```

## document.match()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.match(bWidth, bHeight [, bUseDocumentBounds])
```

### Paramètres

**bWidth** Valeur booléenne. Si elle est `true`, la méthode donne une largeur identique à tous les éléments sélectionnés.

**bHeight** Valeur booléenne. Si elle est `true`, la méthode donne une hauteur identique à tous les éléments sélectionnés.

**bUseDocumentBounds** Valeur booléenne. Si elle est `true`, la méthode aligne la taille des objets sur les limites du document. Sinon, la méthode utilise les limites de l'objet ayant la plus grande taille. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui donne la même taille à tous les objets sélectionnés.

### Exemple

L'exemple suivant donne une largeur identique à tous les éléments sélectionnés :

```
fl.getDocumentDOM().match(true, false);
```

L'exemple suivant donne une hauteur identique à tous les éléments sélectionnés :

```
fl.getDocumentDOM().match(false, true);
```

L'exemple suivant donne une largeur correspondant aux limites du document :

```
fl.getDocumentDOM().match(true, false, true);
```

### Voir aussi

[document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

## document.mouseClick()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.mouseClick(position, bToggleSel, bShiftSel)
```

### Paramètres

**position** Paire de valeurs numériques à virgule flottante indiquant les coordonnées *x* et *y* du clic, en pixels.

**bToggleSel** Valeur booléenne indiquant l'état de la touche Maj : *true* si la touche est enfoncée, *false* si elle ne l'est pas.

**bShiftSel** Valeur booléenne indiquant l'état de la préférence Sélection avec la touche Maj : *true* si la préférence est activée, *false* si elle ne l'est pas.

### Valeur renvoyée

Aucune.

### Description

Méthode qui effectue un clic de souris à l'aide de l'outil Sélection.

### Exemple

L'exemple suivant simule un clic de souris à l'emplacement indiqué :

```
fl.getDocumentDOM().mouseClick({x:300, y:200}, false, false);
```

### Voir aussi

[document.mouseDblClk\(\)](#)



## document.mouseDblClick()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.mouseDblClick(position, bAltDown, bShiftDown, bShiftSelect)
```

### Paramètres

**position** Paire de valeurs numériques à virgule flottante indiquant les coordonnées *x* et *y* du clic, en pixels.

**bAltDown** Valeur booléenne qui indique si la touche Alt est enfoncée au moment de l'événement : *true* si la touche est enfoncée, *false* si elle ne l'est pas.

**bShiftDown** Valeur booléenne qui indique si la touche Maj est enfoncée au moment de l'événement : *true* si la touche est enfoncée, *false* si elle ne l'est pas.

**bShiftSelect** Valeur booléenne indiquant l'état de la préférence Sélection avec la touche Maj : *true* si la préférence est activée, *false* si elle ne l'est pas.

### Valeur renvoyée

Aucune.

### Description

Méthode qui effectue un double-clic de souris à l'aide de l'outil Sélection.

### Exemple

L'exemple suivant simule un double-clic de souris à l'emplacement indiqué :

```
fl.getDocumentDOM().mouseDblClick({x:392.9, y:73}, false, false, true);
```

### Voir aussi

[document.mouseClick\(\)](#)

## document.moveSelectedBezierPointsBy()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.moveSelectedBezierPointsBy(delta)
```

### Paramètres

**delta** Paire de nombres en virgule flottante indiquant les coordonnées *x* et *y* de déplacement, en pixels, des points de courbe de Bézier sélectionnés. Par exemple, les paramètres (*{x:1, y:2}*) spécifient un emplacement situé à droite (de un pixel) et en dessous (de deux pixels) de l'emplacement actuel.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui, si la sélection contient au moins un trajet comptant au moins un point de courbe de Bézier, déplace de la valeur spécifiée tous les points de courbe de Bézier de tous les trajets sélectionnés.

#### Exemple

L'exemple suivant déplace les points de courbe de Bézier sélectionnés de 10 pixels vers la droite et de 5 pixels vers le bas :

```
fl.getDocumentDOM().moveSelectedBezierPointsBy({x:10, y:5});
```

## document.moveSelectionBy()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.moveSelectionBy(distanceToMove)
```

#### Paramètres

**distanceToMove** Paire de nombres en virgule flottante indiquant les valeurs de déplacement *x* et *y* de la sélection. Par exemple, les paramètres (*{x:1, y:2}*) spécifient un emplacement situé à droite (de un pixel) et en dessous (de deux pixels) de l'emplacement actuel.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui déplace les objets sélectionnés de la distance spécifiée.

**Remarque :** lorsque l'utilisateur se sert des touches de direction (flèches) pour déplacer un élément, le panneau Historique combine tous les appuis de touches fléchées en un seul déplacement. Lorsque l'utilisateur appuie à répétition sur les touches fléchées, plutôt que d'enregistrer plusieurs étapes dans le panneau Historique, la méthode effectue tous les déplacements en une seule étape, et les arguments sont actualisés pour reproduire les actions répétées sur les touches fléchées.

Pour plus d'informations sur les sélections, voir les sections [document.setSelectionRect\(\)](#), [document.mouseClick\(\)](#), [document.mouseDblClick\(\)](#) et [Objet Element](#).

#### Exemple

L'exemple suivant déplace l'élément sélectionné de 62 pixels vers la droite et 84 pixels vers le bas :

```
fl.getDocumentDOM().moveSelectionBy({x:62, y:84});
```

## document.name

### Disponibilité

Flash MX 2004.

### Utilisation

`document.name`

### Description

Propriété en lecture seule ; chaîne représentant le nom d'un document (fichier FLA).

### Exemple

L'exemple suivant donne à la variable `fileName` le nom de fichier du premier document du tableau des documents :

```
var fileName = flash.documents[0].name;
```

L'exemple suivant affiche dans le panneau Sortie les noms de tous les documents ouverts :

```
var openDocs = fl.documents;  
for(var i=0;i < openDocs.length; i++){  
    fl.trace(i + " " + openDocs[i].name + "\n");  
}
```

## document.optimizeCurves()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.optimizeCurves(smoothing, bUseMultiplePasses)`

### Paramètres

**smoothing** Nombre entier compris entre 0 et 100, avec 0 signifiant pas de lissage, et 100 signifiant un lissage maximum.

**bUseMultiplePasses** Valeur booléenne. Si elle est `true`, la méthode doit utiliser plusieurs passages. Cette option est plus lente mais produit de meilleurs résultats. Ce paramètre a le même effet qu'un clic sur le bouton Appliquer plusieurs passages dans la boîte de dialogue Optimiser les courbes.

### Valeur renvoyée

Aucune.

### Description

Méthode qui optimise le lissage de la sélection, le cas échéant en autorisant plusieurs passages pour obtenir un lissage optimal. Cette méthode est l'équivalent de l'option de menu Modification > Forme > Optimiser.

### Exemple

L'exemple suivant optimise la courbe de la sélection à 50° de lissage en plusieurs passages :

```
fl.getDocumentDOM().optimizeCurves(50, true);
```

## document.path

### Disponibilité

Flash MX 2004.

### Utilisation

`document.path`

### Description

Propriété en lecture seule ; chaîne représentant le chemin d'accès du document dans un format spécifique à la plateforme. Si le document n'a pas encore été enregistré, cette propriété renvoie `undefined`.

### Exemple

L'exemple suivant affiche dans le panneau Sortie le chemin d'accès du premier document du tableau des documents. Enregistrez le document avant d'exécuter ce script. Dans l'exemple, le fichier est nommé `test fla` et est enregistré dans le dossier Mes documents sur un ordinateur Windows.

```
var filePath = flash.documents[0].path;
fl.trace(filePath);
// displays C:\Documents and Settings\<user name>\My Documents\test fla
```

### Voir aussi

[document.pathURI](#)

## document.pathURI

### Disponibilité

Flash CS4 Professional.

### Utilisation

`document.pathURI`

### Description

Propriété en lecture seule ; chaîne représentant le chemin du document, exprimée sous la forme « fichier:///URI ». Si le document n'a pas encore été enregistré, cette propriété renvoie `undefined`.

### Exemple

L'exemple suivant affiche le chemin du premier document du tableau des documents sous forme de chaîne « fichier:///URI » dans le panneau Sortie. Enregistrez le document avant d'exécuter ce script. Dans l'exemple, le fichier est nommé `test fla` et est enregistré dans le dossier Mes documents sur un ordinateur Windows.

```
var filePathURI = flash.documents[0].pathURI;
fl.trace(filePathURI);
// displays file:///C:/Documents%20and%20Settings/<userName>/My%20Documents/test fla
```

**Voir aussi**

[document.path](#)

## document.publish()

**Disponibilité**

Flash MX 2004.

**Utilisation**

`document.publish()`

**Paramètres**

Aucun.

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui publie le document en fonction des paramètres de publication actuels (Fichier > Paramètres de publication). Cette méthode est l'équivalent de l'option de menu Fichier > Publier.

**Exemple**

L'exemple suivant publie le document actif :

```
fl.getDocumentDOM().publish();
```

## document.publishProfiles

**Disponibilité**

Flash MX 2004.

**Utilisation**

`document.publishProfiles`

**Description**

Propriété en lecture seule ; tableau des noms des profils de publication du document.

**Exemple**

L'exemple suivant affiche les noms des profils de publication du document :

```
var myPubProfiles = fl.getDocumentDOM().publishProfiles;
for (var i=0; i < myPubProfiles.length; i++){
    fl.trace(myPubProfiles[i]);
}
```

## document.punch()

### Disponibilité

Flash 8.

### Utilisation

```
document.punch()
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne : `true` en cas de réussite, `false` dans le cas contraire.

### Description

Méthode qui, sur la base de l'objet de dessin sélectionné le plus élevé, poinçonne tous les objets de dessin de niveau inférieur. La méthode renvoie `false` s'il n'existe pas d'objets de dessin sélectionnés, ou si l'un des éléments sélectionnés n'est pas un objet de dessin.

### Exemple

L'exemple suivant poinçonne tous les objets de dessin de niveau inférieur à l'objet de dessin sélectionné :

```
fl.getDocumentDOM().punch();
```

### Voir aussi

[document.crop\(\)](#), [document.deleteEnvelope\(\)](#), [document.intersect\(\)](#), [document.union\(\)](#),  
[shape.isDrawingObject](#)

## document.removeAllFilters()

### Disponibilité

Flash 8.

### Utilisation

```
document.removeAllFilters()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime tous les filtres des objets sélectionnés.

### Exemple

L'exemple suivant supprime tous les filtres des objets sélectionnés :

```
fl.getDocumentDOM().removeAllFilters();
```

### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableAllFilters\(\)](#),  
[document.getFilters\(\)](#), [document.removeFilter\(\)](#), [Objet Filter](#)

## document.removeDataFromDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.removeDataFromDocument (name)
```

### Paramètres

**name** Chaîne spécifiant le nom des données à supprimer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime les données persistantes associées au document et correspondant au nom spécifié.

### Exemple

L'exemple suivant supprime du document les données persistantes ayant le nom "myData" :

```
fl.getDocumentDOM().removeDataFromDocument ("myData");
```

### Voir aussi

[document.addDataToDocument\(\)](#), [document.documentHasData\(\)](#), [document.getDataFromDocument\(\)](#)

## document.removeDataFromSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.removeDataFromSelection (name)
```

### Paramètres

**name** Chaîne spécifiant le nom des données persistantes à supprimer.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui supprime les données persistantes associées à la sélection et correspondant au nom spécifié.

#### Exemple

L'exemple suivant supprime de la sélection les données persistantes ayant le nom "myData" :

```
fl.getDocumentDOM().removeDataFromSelection("myData");
```

#### Voir aussi

[document.addDataToSelection\(\)](#)

## document.removeFilter()

#### Disponibilité

Flash 8.

#### Utilisation

```
document.removeFilter(filterIndex)
```

#### Paramètres

**filterIndex** Nombre entier indiquant dans la liste Filtres l'index (en base 0) du filtre à supprimer des objets sélectionnés.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui supprime le filtre spécifié de la liste Filtres des objets sélectionnés.

#### Exemple

L'exemple suivant supprime le premier filtre (index 0) de la liste Filtres des objets sélectionnés :

```
fl.getDocumentDOM().removeFilter(0);
```

#### Voir aussi

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#),  
[document.getFilters\(\)](#), [document.removeAllFilters\(\)](#), [Objet Filter](#)

## document.renamePublishProfile()

#### Disponibilité

Flash MX 2004.



### Utilisation

```
document.renamePublishProfile([profileNewName])
```

### Paramètres

**profileNewName** Paramètre facultatif qui indique le nouveau nom du profil. Ce nouveau nom doit être unique. Si vous ne spécifiez pas de nom, un nom par défaut est utilisé.

### Valeur renvoyée

Valeur booléenne, `true` si le nom a bien été modifié, `false` dans le cas contraire.

### Description

Méthode qui renomme le profil actuel.

### Exemple

L'exemple suivant renomme le profil actuel avec un nom par défaut qui est ensuite affiché :

```
alert(fl.getDocumentDOM().renamePublishProfile());
```

## document.renameScene()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.renameScene(name)
```

### Paramètres

**name** Chaîne spécifiant le nouveau nom de la séquence.

### Valeur renvoyée

Valeur booléenne, `true` si le nom a bien été modifié, `false` dans le cas contraire. Si le nouveau nom n'est pas unique, par exemple, la méthode renvoie `false`.

### Description

Méthode qui renomme la séquence sélectionnée dans le panneau Séquences. Le nouveau nom de la séquence doit être unique.

### Exemple

L'exemple suivant renomme la séquence active avec le nom "new name" :

```
var success = fl.getDocumentDOM().renameScene("new name");
```

## document.reorderScene()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.reorderScene(sceneToMove, sceneToPutItBefore)
```

### Paramètres

**sceneToMove** Nombre entier spécifiant l'index de la séquence à déplacer, 0 (zéro) étant la première séquence.

**sceneToPutItBefore** Nombre entier spécifiant la séquence devant laquelle la séquence définie par *sceneToMove* doit être placée. Indiquez 0 (zéro) pour la première séquence. Par exemple, si vous spécifiez 1 pour *sceneToMove* et 0 pour *sceneToPutItBefore*, la seconde séquence est placée devant la première. Spécifiez -1 pour placer la séquence à la fin.

### Valeur renvoyée

Aucune.

### Description

Méthode qui place la séquence spécifiée devant une autre séquence spécifiée.

### Exemple

L'exemple suivant place la seconde séquence devant la première :

```
fl.getDocumentDOM().reorderScene(1, 0);
```

## document.resetOvalObject()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
document.resetOvalObject()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui, dans l'inspecteur Propriétés, définit toutes les valeurs sur les paramètres par défaut de l'objet Oval. Si des objets Oval sont sélectionnés, leurs propriétés sont également redéfinies sur les valeurs par défaut.

### Exemple

L'exemple suivant redéfinit les propriétés de l'objet Oval du document actif sur les valeurs par défaut :

```
fl.getDocumentDOM().resetOvalObject();
```

**Voir aussi**

[document.resetRectangleObject\(\)](#)

## document.resetRectangleObject()

**Disponibilité**

Flash CS3 Professional.

**Utilisation**

```
document.resetRectangleObject()
```

**Paramètres**

Aucun.

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui, dans l'inspecteur Propriétés, définit toutes les valeurs sur les paramètres par défaut de l'objet Rectangle. Si des objets Rectangle sont sélectionnés, leurs propriétés sont également redéfinies sur les valeurs par défaut.

**Exemple**

L'exemple suivant redéfinit les propriétés de l'objet Rectangle du document actif sur les valeurs par défaut :

```
fl.getDocumentDOM().resetRectangleObject();
```

**Voir aussi**

[document.resetOvalObject\(\)](#)

## document.resetTransformation()

**Disponibilité**

Flash MX 2004.

**Utilisation**

```
document.resetTransformation()
```

**Paramètres**

Aucun.

**Valeur renvoyée**

Aucune.

### Description

Méthode qui réinitialise la matrice de transformation. Cette méthode est l'équivalent de l'option de menu Modification > Transformer > Supprimer la transformation.

### Exemple

L'exemple suivant réinitialise la matrice de transformation de la sélection.

```
fl.getDocumentDOM().resetTransformation();
```

## document.revert()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.revert();
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui restaure la dernière version enregistrée du document spécifié. Cette méthode est l'équivalent de l'option de menu Fichier > Rétablir.

### Exemple

L'exemple suivant restaure la dernière version enregistrée du document actif.

```
fl.getDocumentDOM().revert();
```

### Voir aussi

[document.canRevert\(\)](#), [fl.revertDocument\(\)](#)

## document.rotate3DSelection()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.rotate3DSelection(xyzCoordinate, bGlobalTransform)
```

### Paramètres

**xyzCoordinate** Point de coordonnées XYZ spécifiant les axes d'une rotation 3D.

**bGlobalTransform** Valeur booléenne qui spécifie si le mode de transformation doit être global (`true`) ou local (`false`).

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui applique une rotation 3D à la sélection. Cette méthode n'est disponible que pour les clips.

#### Exemple

Dans l'exemple suivant, la sélection pivote d'abord par rapport à la scène (globalement), puis sur elle-même (localement).

```
var myDocument = fl.getDocumentDOM();  
myDocument.rotate3DSelection({x:52.0, y:0, z:0}, true);  
myDocument.rotate3DSelection({x:52.0, y:0, z:-55.2}, false);
```

## document.rotateSelection()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.rotateSelection(angle [, rotationPoint])
```

#### Paramètres

**angle** Valeur en virgule flottante spécifiant l'angle de rotation.

**rotationPoint** Chaîne spécifiant le côté du rectangle de délimitation sur lequel doit s'effectuer la rotation. Les valeurs gérées sont "top right" (haut droite), "top left" (haut gauche), "bottom right" (bas droite), "bottom left" (bas gauche), "top center" (centre haut), "right center" (centre droite), "bottom center" (centre bas) et "left center" (centre gauche). Si ce paramètre n'est pas spécifié, la méthode utilise le point de transformation. Ce paramètre est facultatif.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui effectue sur la sélection une rotation du nombre de degrés spécifié. L'effet est identique à celui de l'outil Transformation libre utilisé pour une rotation de l'objet.

#### Exemple

L'exemple suivant effectue une rotation de 45° de la sélection autour de son point de transformation :

```
fl.getDocumentDOM().rotateSelection(45);
```

L'exemple suivant effectue une rotation de 45 degrés de la sélection autour de son coin inférieur gauche :

```
fl.getDocumentDOM().rotateSelection(45, "bottom left");
```

## document.save()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.save ([bOkToSaveAs])
```

### Paramètres

**bOkToSaveAs** Paramètre facultatif. S'il est défini sur `true` ou omis, et si le fichier n'a jamais été enregistré, la boîte de dialogue Enregistrer sous apparaît. Si sa valeur est `false`, le fichier n'est pas enregistré, même s'il ne l'a jamais été.

### Valeur renvoyée

Valeur booléenne, `true` en cas de réussite de l'enregistrement, `false` dans le cas contraire.

### Description

Méthode qui enregistre le document à son emplacement par défaut. Cette méthode est l'équivalent de l'option de menu Fichier > Enregistrer.

Pour donner un autre nom au fichier (au lieu de l'enregistrer avec un nom identique), utilisez `fl.saveDocument()`.

***Remarque :** si le fichier est nouveau et qu'il n'a été ni enregistré ni modifié, ou s'il existe et n'a pas été modifié depuis son dernier enregistrement, cette méthode n'a aucun effet et la valeur `false` est renvoyée. Pour permettre l'enregistrement d'un fichier qui n'a pas été enregistré ni modifié, utilisez `document.saveAndCompact()` ou `fl.saveDocumentAs()`.*

### Exemple

L'exemple suivant enregistre le document actif à son emplacement par défaut :

```
fl.getDocumentDOM().save();
```

### Voir aussi

`document.saveAndCompact()`, `fl.saveAll()`, `fl.saveDocument()`, `fl.saveDocumentAs()`

## document.saveAndCompact()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.saveAndCompact ([bOkToSaveAs])
```

### Paramètres

**bOkToSaveAs** Paramètre facultatif. S'il est `true` ou s'il est omis, et si le fichier n'a jamais été enregistré, la boîte de dialogue Enregistrer sous apparaît. Si sa valeur est `false`, le fichier n'est pas enregistré, même s'il ne l'a jamais été. La valeur par défaut est `true`.

### Valeur renvoyée

Valeur booléenne, `true` en cas de réussite de l'enregistrement avec compression, `false` dans le cas contraire.

### Description

Méthode qui enregistre et compresse le fichier. Cette méthode est l'équivalent de l'option de menu Fichier > Enregistrer et compresser.

*Remarque : si le fichier n'a jamais été enregistré, cette méthode renvoie `true` même si l'utilisateur annule la boîte de dialogue Enregistrer sous. Pour déterminer précisément si le fichier a été enregistré ou non, utilisez `fl.saveDocumentAs()`.*

### Exemple

L'exemple suivant enregistre et compresse le document actif :

```
fl.getDocumentDOM().saveAndCompact();
```

### Voir aussi

`document.save()`, `fl.saveDocumentAs()`, `fl.saveDocument()`, `fl.saveAll()`

## document.scaleSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.scaleSelection(xScale, yScale [, whichCorner])
```

### Paramètres

**xScale** Valeur en virgule flottante spécifiant la valeur *x* de mise à l'échelle horizontale.

**yScale** Valeur en virgule flottante spécifiant la valeur *y* de mise à l'échelle verticale.

**whichCorner** Chaîne indiquant le coin à partir duquel est appliquée la transformation. Si ce paramètre n'est pas spécifié, la méthode utilise le point de transformation. Les valeurs possibles sont : "bottom left" (bas gauche), "bottom right" (bas droite), "top right" (haut droite), "top left" (haut gauche), "top center" (centre haut), "right center" (centre droite), "bottom center" (centre bas) et "left center" (centre gauche). Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui effectue sur la sélection une mise à l'échelle de la valeur spécifiée. L'effet est identique à celui de l'outil Transformation libre utilisé pour une mise à l'échelle de l'objet.

### Exemple

L'exemple suivant double la largeur originale de la sélection et réduit sa hauteur de moitié :

```
fl.getDocumentDOM().scaleSelection(2.0, 0.5);
```

L'exemple suivant fait pivoter la sélection sur l'axe vertical :

```
fl.getDocumentDOM().scaleSelection(1, -1);
```

L'exemple suivant fait pivoter la sélection sur l'axe horizontal :

```
fl.getDocumentDOM().scaleSelection(-1, 1);
```

L'exemple suivant effectue une mise à l'échelle verticale de la sélection, avec un facteur de 1,9 à partir du centre haut :

```
fl.getDocumentDOM().scaleSelection(1, 1.90, 'top center');
```

## document.screenOutline

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.screenOutline
```

### Description

Propriété en lecture seule ; objet ScreenOutline actuel du document. Avant d'accéder à cet objet pour la première fois, il est conseillé d'utiliser `document.allowScreens()` pour vérifier l'existence de la propriété.

### Exemple

L'exemple suivant affiche un tableau des valeurs de la propriété `screenOutline` :

```
var myArray = new Array();
for(var i in fl.getDocumentDOM().screenOutline) {
    myArray.push(" "+i+" : "+fl.getDocumentDOM().screenOutline[i]) ;
}
fl.trace("Here is the property dump for screenOutline: "+myArray);
```

### Voir aussi

[document.allowScreens\(\)](#)

## document.selectAll()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.selectAll()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.



### Description

Méthode qui sélectionne tous les éléments sur la scène. Cette méthode est l'équivalent des raccourcis clavier Ctrl+A (Windows) ou Commande+A (Macintosh), ou de l'option de menu Edition > Sélectionner tout.

### Exemple

L'exemple suivant sélectionne tous les éléments actuellement visibles pour l'utilisateur :

```
fl.getDocumentDOM().selectAll();
```

### Voir aussi

[document.selection](#), [document.selectNone\(\)](#)

## document.selection

### Disponibilité

Flash MX 2004.

### Utilisation

`document.selection`

### Description

Propriété ; tableau des objets sélectionnés dans le document. Si rien n'est sélectionné, la propriété renvoie un tableau de longueur zéro. Si aucun document n'est ouvert, elle renvoie `null`.

Pour ajouter des objets au tableau, vous devez d'abord les sélectionner à l'aide de l'une des techniques suivantes :

- Sélectionnez manuellement les objets sur la scène.
- Utilisez l'une des méthodes de sélection, telles que [document.setSelectionRect\(\)](#), [document.setSelectionBounds\(\)](#), [document.mouseClick\(\)](#), [document.mouseDbClick\(\)](#) ou [document.selectAll\(\)](#).
- Sélectionnez manuellement une ou plusieurs images.
- Utilisez l'une des méthodes de l'[Objet Timeline](#) pour sélectionner une ou plusieurs images, telles que [timeline.getSelectedFrames\(\)](#), [timeline.setSelectedFrames\(\)](#) ou [timeline.selectAllFrames\(\)](#).
- Spécifiez tous les éléments dans une image particulière (voir [Objet Element](#)). Reportez-vous au premier exemple ci-dessous.
- Créez un tableau d'un ou plusieurs éléments, puis attribuez-le au tableau `document.selection`. Reportez-vous au troisième exemple ci-dessous.

### Exemple

L'exemple suivant assigne tous les éléments de l'image 11 à la sélection actuelle (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().currentFrame = 10;  
fl.getDocumentDOM().selection =  
fl.getDocumentDOM().getTimeline().layers[0].frames[10].elements;
```

L'exemple suivant crée un rectangle dans le coin supérieur gauche de la scène, puis une chaîne de texte sous ce rectangle. Ces deux objets sont ensuite sélectionnés à l'aide de `document.setSelectionRect()` et ajoutés au tableau `document.selection`. Enfin, le contenu de `document.selection` est affiché dans le panneau Sortie.

```
fl.getDocumentDOM().addNewRectangle({left:0, top:0, right:99, bottom:99}, 0);
fl.getDocumentDOM().addNewText({left:-1, top:117.3, right:9.2, bottom:134.6});
fl.getDocumentDOM().setTextString('Hello World');
fl.getDocumentDOM().setSelectionRect({left:-28, top:-22, right:156.0, bottom:163});

var theSelectionArray = fl.getDocumentDOM().selection;

for(var i=0;i<theSelectionArray.length;i++){
fl.trace("fl.getDocumentDOM().selection["+i+"] = " + theSelectionArray[i]);
}
```

L'exemple suivant est plus complexe. Il montre comment parcourir en boucle le tableau des calques et le tableau des éléments pour retrouver les occurrences d'un symbole particulier et les sélectionner. Cet exemple pourrait être étendu en y ajoutant des boucles analysant les images ou les séquences. Cet exemple affecte à la sélection active toutes les occurrences du clip `myMovieClip` figurant dans la première image :

```
// Assigns the layers array to the variable "theLayers".
var theLayers = fl.getDocumentDOM().getTimeline().layers;
// Creates an array to hold all the elements
// that are instances of "myMovieClip".
var myArray = new Array();
// Counter variable
var x = 0;
// Begin loop through all the layers.
for (var i = 0; i < theLayers.length; i++) {
    // Gets the array of elements in Frame 1
    // and assigns it to the array "theElems".
    var theElems = theLayers[i].frames[0].elements;
    // Begin loop through the elements on a layer.
    for (var c = 0; c < theElems.length; c++) {
        // Checks to see if the element is of type "instance".
        if (theElems[c].elementType == "instance") {
            // If the element is an instance, it checks
            // if it is an instance of "myMovieClip".
            if (theElems[c].libraryItem.name == "myMovieClip") {
                // Assigns elements that are instances of "myMovieClip" to "myArray".
                myArray[x] = theElems[c];
                // Increments counter variable.
                x++;
            }
        }
    }
}

// Now that you have assigned all the instances of "myMovieClip"
// to "myArray", you then set the document.selection array
// equal to myArray. This selects the objects on the Stage.
fl.getDocumentDOM().selection = myArray;
```

## document.selectNone()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.selectNone()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui désélectionne les éléments sélectionnés.

### Exemple

L'exemple suivant désélectionne tous les éléments sélectionnés :

```
fl.getDocumentDOM().selectNone();
```

### Voir aussi

[document.selectAll\(\)](#), [document.selection](#)

## document.setAlignToDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setAlignToDocument(bToStage)
```

### Paramètres

**bToStage** Valeur booléenne. Si elle est définie sur `true`, la méthode aligne les objets sur la scène. Si elle est `false`, elle ne les aligne pas.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit les préférences afin d'activer [document.align\(\)](#), [document.distribute\(\)](#), [document.match\(\)](#) et [document.space\(\)](#) pour le document. Cette commande est équivalente à l'activation de l'option Sur la scène dans le panneau Aligner.

### Exemple

L'exemple suivant active le bouton Sur la scène dans le panneau Aligner, afin d'aligner les objets sur la scène :

```
fl.getDocumentDOM().setAlignToDocument(true);
```

### Voir aussi

[document.getAlignToDocument\(\)](#)

## document.setBlendMode()

### Disponibilité

Flash 8.

### Utilisation

```
document.setBlendMode(mode)
```

### Paramètres

**mode** Chaîne représentant le mode de fusion désiré pour les objets sélectionnés. Les valeurs gérées sont "normal", "layer", "multiply", "screen", "overlay", "hardlight", "lighten", "darken", "difference", "add", "subtract", "invert", "alpha" et "erase".

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit le mode de fusion pour les objets sélectionnés.

### Exemple

L'exemple suivant définit sur "add" le mode de fusion pour l'objet sélectionné.

```
fl.getDocumentDOM().setBlendMode("add");
```

### Voir aussi

[document.addFilter\(\)](#), [document.setFilterProperty\(\)](#), [symbolInstance.blendMode](#)

## document.setCustomFill()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setCustomFill(fill)
```

### Paramètres

**fill** Objet Fill qui spécifie les paramètres de remplissage requis. Voir [Objet Fill](#).

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit les paramètres de remplissage pour le panneau Outils, l'inspecteur Propriétés et les formes sélectionnées, le cas échéant. Cette opération permet au script de définir les paramètres de remplissage avant de tracer un objet, ce qui est plus élégant que dessiner l'objet, le sélectionner et modifier ses paramètres de remplissage. Elle permet aussi de modifier par script les paramètres de remplissage du panneau Outils et de l'inspecteur Propriétés.

### Exemple

L'exemple suivant définit le blanc comme couleur active dans la palette des couleurs de remplissage du panneau Outils, de l'inspecteur Propriétés et des éventuelles formes sélectionnées :

```
var fill = fl.getDocumentDOM().getCustomFill();  
fill.color = 'FFFFFF';  
fill.style = "solid";  
fl.getDocumentDOM().setCustomFill(fill);
```

### Voir aussi

[document.getCustomFill\(\)](#)

## document.setCustomStroke()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setCustomStroke(stroke)
```

### Paramètres

**stroke** Objet [Objet Stroke](#).

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit les paramètres de trait pour le panneau Outils, l'inspecteur Propriétés et les formes sélectionnées, le cas échéant. Cette opération permet au script de définir les paramètres de trait avant de tracer un objet, ce qui est plus élégant que dessiner l'objet, le sélectionner et modifier ses paramètres de trait. Elle permet aussi de modifier par script les paramètres de trait du panneau Outils et de l'inspecteur Propriétés.

### Exemple

L'exemple suivant modifie les paramètres d'épaisseur de trait du panneau Outils, de l'inspecteur Propriétés et des formes sélectionnées, le cas échéant :

```
var stroke = fl.getDocumentDOM().getCustomStroke();  
stroke.thickness += 2;  
fl.getDocumentDOM().setCustomStroke(stroke);
```

#### Voir aussi

`document.getCustomStroke()`

## document.setElementProperty()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.setElementProperty(property, value)
```

#### Paramètres

**property** Chaîne spécifiant le nom de la propriété `Element` à définir. La liste complète des propriétés et des valeurs est indiquée dans le tableau Résumé des propriétés de l'[Objet Element](#).

Cette méthode ne permet pas de modifier les propriétés en lecture seule, telles que `element.elementType`, `element.top` ou `element.left`.

**value** Nombre entier spécifiant la valeur à attribuer à la propriété `Element` spécifiée.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui définit la propriété `Element` spécifiée pour les objets sélectionnés du document. La méthode est inopérante s'il n'existe pas de sélection.

#### Exemple

L'exemple suivant définit la largeur de tous les éléments sélectionnés sur 100 et leur hauteur sur 50 :

```
fl.getDocumentDOM().setElementProperty("width", 100);  
fl.getDocumentDOM().setElementProperty("height", 50);
```

## document.setElementTextAttr()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.setElementTextAttr(attrName, attrValue [, startIndex [, endIndex]])
```

#### Paramètres

**attrName** Chaîne spécifiant le nom de la propriété `TextAttrs` à modifier.

**attrValue** Valeur à affecter à la propriété `TextAttrs`. La liste des noms de propriété et des valeurs possibles est indiquée dans le tableau Résumé des propriétés de l'[Objet TextAttrs](#).

**startIndex** Nombre entier spécifiant l'index du premier caractère concerné. Ce paramètre est facultatif.

**endIndex** Nombre entier spécifiant l'index du dernier caractère concerné. Ce paramètre est facultatif.

#### Valeur renvoyée

Valeur booléenne, *true* si au moins une propriété d'attribut de texte est modifiée, *false* dans le cas contraire.

#### Description

Méthode qui définit sur la valeur spécifiée la propriété *TextAttrs* des éléments de texte sélectionnés. La liste des noms de propriété et des valeurs possibles est indiquée dans le tableau Résumé des propriétés de l'Objet *TextAttrs*. Si les paramètres facultatifs ne sont pas transmis, la méthode définit le style de la plage de texte sélectionnée, ou de tout le champ de texte si aucun texte n'est sélectionné. Si seul le paramètre *startIndex* est transmis, la méthode définit les attributs de ce caractère. Si *startIndex* et *endIndex* sont transmis, la méthode définit les attributs de toute la plage de caractères comprise entre *startIndex* et le caractère précédant *endIndex*. Si des styles de paragraphe sont spécifiés, tous les paragraphes de la plage sélectionnée sont modifiés.

#### Exemple

Les exemples suivants affectent les attributs de texte *fillColor*, *italic* et *bold* aux éléments de texte sélectionnés :

```
var success = fl.getDocumentDOM().setElementTextAttr("fillColor", "#00ff00");  
var pass = fl.getDocumentDOM().setElementTextAttr("italic", true, 10);  
var ok = fl.getDocumentDOM().setElementTextAttr("bold", true, 5, 15);
```

## document.setFillColor()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.setFillColor(color)
```

#### Paramètres

**color** Couleur du remplissage, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBBA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

Si la valeur du paramètre est *null*, aucune couleur de remplissage n'est définie, ce qui est l'équivalent du choix de l'option Pas de remplissage dans la palette des couleurs de remplissage.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui modifie la couleur de remplissage de la sélection avec la couleur spécifiée. Pour plus d'informations sur la modification de la couleur de remplissage dans le panneau Outils et l'inspecteur Propriétés, voir la méthode

[document.setCustomFill\(\)](#).

### Exemple

Dans l'exemple suivant, les trois premières instructions définissent la couleur de remplissage à l'aide des trois différents formats de définition de la couleur. La quatrième instruction annule le remplissage.

```
fl.getDocumentDOM().setFillColor("#cc00cc");  
fl.getDocumentDOM().setFillColor(0xcc00cc);  
fl.getDocumentDOM().setFillColor(120000);  
fl.getDocumentDOM().setFillColor(null);
```

## document.setFilterProperty()

### Disponibilité

Flash 8.

### Utilisation

```
document.setFilterProperty(property, filterIndex, value)
```

### Paramètres

**property** Chaîne spécifiant la propriété à définir. Les valeurs gérées sont "blurX", "blurY", "quality", "angle", "distance", "strength", "knockout", "inner", "bevelType", "color", "shadowColor" et "highlightColor".

**filterIndex** Nombre entier spécifiant l'index (en base 0) du filtre dans la liste Filtres.

**value** Nombre ou chaîne spécifiant la valeur à attribuer à la propriété de filtre spécifiée. Les valeurs acceptables dépendent de la propriété et du filtre à définir.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la propriété filter spécifiée pour les objets sélectionnés (en supposant que l'objet prenne en charge le filtre spécifié).

### Exemple

L'exemple suivant affecte à la propriété `quality` la valeur 2 pour le second filtre (index 1) de la liste Filtres des objets sélectionnés, puis définit la propriété `shadowColor` du premier filtre de la liste Filtres sur les objets sélectionnés :

```
fl.getDocumentDOM().setFilterProperty("quality", 1, 2);  
fl.getDocumentDOM().setFilterProperty("shadowColor", 0, "#FF00FF");
```

### Voir aussi

[document.addFilter\(\)](#), [document.getFilters\(\)](#), [document.setBlendMode\(\)](#), [document.setFilters\(\)](#),  
[Objet Filter](#)

## document.setFilters()

### Disponibilité

Flash 8.



### Utilisation

```
document.setFilters(filterArray)
```

### Paramètres

**filterArray** Tableau des filtres spécifiés.

### Valeur renvoyée

Aucune.

### Description

Méthode qui applique des filtres aux objets sélectionnés. Utilisez cette méthode après avoir appelé `document.getFilters()` et effectué toute modification requise dans les filtres.

### Exemple

L'exemple suivant récupère la liste des filtres appliqués à l'objet sélectionné et définit la propriété `blurX` de tous les filtres Blur sur 50 :

```
var myFilters = fl.getDocumentDOM().getFilters();
for (i=0; i < myFilters.length; i++) {
    if (myFilters[i].name == "blurFilter") {
        myFilters[i].blurX = 50;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

[document.addFilter\(\)](#), [document.getFilters\(\)](#), [document.setFilterProperty\(\)](#), [Objet Filter](#)

## document.setInstanceAlpha()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setInstanceAlpha(opacity)
```

### Paramètres

**opacity** Nombre entier compris entre 0 (transparent) et 100 (opacité complète) qui définit la transparence de l'occurrence.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit l'opacité de l'occurrence.

### Exemple

L'exemple suivant définit l'opacité de la teinte sur 50 :

```
fl.getDocumentDOM().setInstanceAlpha(50);
```

## document.setInstanceBrightness()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setInstanceBrightness(brightness)
```

### Paramètres

**brightness** Nombre entier définissant la luminosité, avec une valeur comprise entre -100 (noir) et 100 (blanc).

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la luminosité de l'occurrence.

### Exemple

L'exemple suivant définit la luminosité de l'occurrence sur 50 :

```
fl.getDocumentDOM().setInstanceBrightness(50);
```

## document.setInstanceTint()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setInstanceTint( color, strength )
```

### Paramètres

**color** Couleur de la teinte, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

**strength** Entier compris entre 0 et 100 qui définit l'opacité de la teinte.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la teinte de l'occurrence.

### Exemple

L'exemple suivant définit sur rouge la teinte de l'occurrence sélectionnée, avec une opacité de 50 :

```
fl.getDocumentDOM().setInstanceTint(0xff0000, 50);
```

## document.setMetadata()

### Disponibilité

Flash 8.

### Utilisation

```
document.setMetadata(strMetadata)
```

### Paramètres

**strMetadata** Chaîne contenant les métadonnées XML associées au document. Pour plus d'informations, voir la description ci-dessous.

### Valeur renvoyée

Valeur booléenne : `true` en cas de réussite, `false` dans le cas contraire.

### Description

Méthode qui définit les métadonnées XML du document spécifié, en remplacement des métadonnées existantes. Avant d'être enregistré, le code XML transmis dans *strMetadata* fait l'objet d'une validation et est susceptible d'être réécrit. S'il est impossible de les valider comme XML standard ou si elles enfreignent des règles spécifiques, les métadonnées XML ne sont pas appliquées et la méthode renvoie `false`. (Si la valeur `false` est renvoyée, il n'est pas possible d'obtenir plus d'informations sur l'erreur.)

**Remarque :** même si la méthode renvoie `true`, il est possible que le code XML appliqué ne soit pas exactement identique à la chaîne transmise. Pour extraire la valeur exacte du code XML, appelez `document.getMetadata()`.

Les métadonnées sont au format RDF qui est conforme à la spécification XMP. Pour plus d'informations sur RDF et XMP, voir les ressources suivantes :

- Le guide RDF à l'adresse [www.w3.org/TR/rdf-primer/](http://www.w3.org/TR/rdf-primer/)
- La spécification RDF à l'adresse [www.w3.org/TR/1999/REC-rdf-syntax-19990222/](http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/)
- La page d'accueil de XMP à l'adresse [www.adobe.com/products/xmp/](http://www.adobe.com/products/xmp/)

### Exemple

Les exemples suivants illustrent diverses méthodes autorisées de représentation de données identiques. Dans tous les cas de figure, à l'exception du deuxième, si les données étaient transmises à `Document.setMetadata()`, elles ne seraient pas réécrites (mis à part la suppression des sauts de ligne).

Dans le premier exemple, les métadonnées sont imbriquées dans les balises, les différents schémas étant placés dans des balises `rdf:Description` distinctes :

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
<rdf:Description rdf:about='' xmlns:dc='http://purl.org/dc/1.1/'>
<dc:title>Simple title</dc:title>
<dc:description>Simple description</dc:description>
</rdf:Description>
<rdf:Description rdf:about='' xmlns:xmp='http://ns.adobe.com/xap/1.0/'>
<xmp:CreateDate>2004-10-12T10:29-07:00</xmp:CreateDate>
<xmp:CreatorTool>Flash Authoring WIN 8,0,0,215</xmp:CreatorTool>
</rdf:Description>
</rdf:RDF>
```

Dans le deuxième exemple, les métadonnées sont imbriquées dans les balises, mais tous les schémas sont regroupés dans la balise `rdf:Description`. Cet exemple comprend également des commentaires, qui ne seront pas pris en compte et seront ignorés par `Document.setMetadata()` :

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!-- This is before the first rdf:Description tag -->
<rdf:Description rdf:about='' xmlns:dc='http://purl.org/dc/1.1/'>
<dc:title>Simple title</dc:title>
<dc:description>Simple description</dc:description>
</rdf:Description>
  <!-- This is between the two rdf:Description tags -->
<rdf:Description rdf:about='' xmlns:xmp='http://ns.adobe.com/xap/1.0/'>
<xmp:CreateDate>2004-10-12T10:29-07:00</xmp:CreateDate>
<xmp:CreatorTool>Flash Authoring WIN 8,0,0,215</xmp:CreatorTool>
</rdf:Description>
  <!-- This is after the second rdf:Description tag -->
</rdf:RDF>
```

Dans le troisième exemple, les métadonnées sont imbriquées dans des attributs et tous les schémas sont regroupés dans une balise `rdf:Description` :

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
<rdf:Description rdf:about='' xmlns:dc='http://purl.org/dc/1.1/' dc:title='Simple title'
dc:description='Simple description' />
<rdf:Description rdf:about='' xmlns:xmp='http://ns.adobe.com/xap/1.0/'
xmp:CreateDate='2004-10-12T10:29-07:00' xmp:CreatorTool='Flash Authoring WIN 8,0,0,215' />
</rdf:RDF>
```

#### Voir aussi

[document.getMetadata\(\)](#)

## document.setMobileSettings()

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

`document.setMobileSettings(xmlString)`

#### Paramètres

**xmlString** Chaîne décrivant les paramètres XML dans un fichier FLA mobile.

### Valeur renvoyée

Valeur `true` si les paramètres ont été correctement définis, `false` dans le cas contraire.

### Description

Méthode qui définit la valeur d'une chaîne de paramètres XML dans un fichier FLA mobile. (La plupart des fichiers FLA mobiles sont pourvus d'une chaîne XML servant à décrire les paramètres dans le document.)

### Exemple

L'exemple suivant définit la chaîne de paramètres XML pour un fichier FLA mobile. Vous remarquerez que l'exemple ci-dessous représente une seule ligne de code.

```
fl.getDocumentDOM().setMobileSettings("<? xml version='1.0' encoding='UTF-16' standalone='no'
?> <mobileSettings> <contentType id='standalonePlayer' name='Standalone Player'/>
<testDevices> <testDevice id='1170' name='Generic Phone' selected='yes'/> </testDevices>
<outputMsgFiltering info='no' trace='yes' warning='yes'/> <testWindowState height='496'
splitterClosed='No' splitterXPos='400' width='907'/> </mobileSettings>");
```

### Voir aussi

[document.getMobileSettings\(\)](#)

## document.setOvalObjectProperty()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
document.setOvalObjectProperty(propertyName, value)
```

### Paramètres

**propertyName** Chaîne précisant la propriété à définir. La liste des valeurs possibles est indiquée dans le tableau Résumé des propriétés de l'[Objet Oval](#).

**value** Valeur à affecter à la propriété. Les valeurs gérées varient en fonction de la propriété que vous spécifiez dans *propertyName*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit une valeur pour une propriété spécifiée des objets Oval de type primitif.

### Exemple

Reportez-vous aux propriétés individuelles de l'[Objet Oval](#) pour les exemples.

### Voir aussi

[Objet Oval](#), [shape.isOvalObject](#)

## document.setPlayerVersion()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
document.setPlayerVersion(version)
```

### Paramètres

**version** Chaîne représentant la version de Flash Player ciblée par le document spécifié. Les valeurs possibles sont "FlashLite", "FlashLite11", "FlashLite20", "FlashLite30", "1", "2", "3", "4", "5", "6", "7", "8", "9", "FlashPlayer10" et "AdobeAIR1\_1".

### Valeur renvoyée

Valeur `true` si la version du lecteur a été correctement définie, `false` dans le cas contraire.

### Description

Méthode qui définit la version de Flash Player ciblée par le document spécifié. Il s'agit de la même valeur que celle qui a été définie dans la boîte de dialogue Paramètres de publication.

### Exemple

L'exemple suivant cible Flash Player 6 comme version de lecteur pour le document actif :

```
fl.getDocumentDOM().setPlayerVersion("6");
```

### Voir aussi

[document.getPlayerVersion\(\)](#)

## document.setRectangleObjectProperty()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
document.setRectangleObjectProperty(propertyName, value)
```

### Paramètres

**propertyName** Chaîne précisant la propriété à définir. Les valeurs possibles sont indiquées dans le tableau Résumé des propriétés de l'[Objet Rectangle](#).

**value** Valeur à affecter à la propriété. Les valeurs gérées varient en fonction de la propriété que vous spécifiez dans *propertyName*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui spécifie une valeur pour une propriété spécifiée des objets Rectangle de type primitif.

### Exemple

Reportez-vous aux propriétés individuelles de l'[Objet Rectangle](#) pour les exemples.

### Voir aussi

[Objet Rectangle](#), [shape.isRectangleObject](#)

## document.setSelectionBounds()

### Disponibilité

Flash MX 2004 ; paramètre *bContactSensitiveSelection* intégré à Flash 8.

### Utilisation

```
document.setSelectionBounds(boundingBox [, bContactSensitiveSelection])
```

### Paramètres

**boundingRectangle** Rectangle délimitant le nouvel emplacement et la nouvelle taille de la sélection. Pour plus d'informations sur le format de *boundingRectangle*, voir la méthode [document.addNewRectangle\(\)](#).

**bContactSensitiveSelection** Une valeur booléenne indiquant si le mode de sélection Activer au contact est activé (*true*) ou non (*false*) lors de la sélection d'un objet. La valeur par défaut est *false*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui déplace et redimensionne la sélection en une seule opération.

Si une valeur est transmise pour *bContactSensitiveSelection*, elle n'est valide que pour cette méthode et n'affecte pas le mode de sélection Activer au contact du document (voir [fl.contactSensitiveSelection](#)).

### Exemple

L'exemple suivant déplace la sélection aux coordonnées 10, 20 et la redimensionne à 100 sur 200 pixels :

```
var l = 10;  
var t = 20;  
fl.getDocumentDOM().setSelectionBounds({left:l, top:t, right:(100+l), bottom:(200+t)});
```

### Voir aussi

[document.selection](#), [document.setSelectionRect\(\)](#)

# document.setSelectionRect()

## Disponibilité

Flash MX 2004 ; paramètre *bContactSensitiveSelection* intégré à Flash 8.

## Utilisation

```
document.setSelectionRect(rect [, bReplaceCurrentSelection [, bContactSensitiveSelection]])
```

## Paramètres

**rect** Objet rectangle à définir comme sélectionné. Pour plus d'informations sur le format de *rect*, voir [document.addNewRectangle\(\)](#).

**bReplaceCurrentSelection** Valeur booléenne qui indique si la méthode remplace la sélection actuelle (*true*) ou ajoute l'élément à cette dernière (*false*). La valeur par défaut est *true*.

**bContactSensitiveSelection** Une valeur booléenne indiquant si le mode de sélection Activer au contact est activé (*true*) ou non (*false*) lors de la sélection d'un objet. La valeur par défaut est *false*.

## Valeur renvoyée

Aucune.

## Description

Méthode qui trace un cadre de sélection rectangulaire sur la scène, à partir des coordonnées spécifiées. Cette opération est différente de `document.getSelectionRect()`, avec laquelle l'emplacement du rectangle est relatif à l'objet en cours de modification.

Cette méthode revient à faire glisser un rectangle à l'aide de l'outil Sélection. Pour être sélectionnée, une occurrence doit être entièrement enfermée dans ce rectangle.

Si une valeur est transmise pour *bContactSensitiveSelection*, elle n'est valide que pour cette méthode et n'affecte pas le mode de sélection Activer au contact du document (voir [fl.contactSensitiveSelection](#)).

**Remarque :** la répétition de `setSelectionRect()` à l'aide du panneau ou de l'élément de menu Historique répète l'étape qui précède l'opération `setSelectionRect()`.

## Exemple

Dans l'exemple suivant, la seconde sélection remplace la première :

```
fl.getDocumentDOM().setSelectionRect({left:1, top:1, right:200, bottom:200});  
fl.getDocumentDOM().setSelectionRect({left:364.0, top:203.0, right:508.0, bottom:434.0},  
true);
```

Dans l'exemple suivant, la seconde sélection est ajoutée à la première. Cette opération est identique à un appui maintenu sur la touche Maj tout en sélectionnant un second objet.

```
fl.getDocumentDOM().setSelectionRect({left:1, top:1, right:200, bottom:200});  
fl.getDocumentDOM().setSelectionRect({left:364.0, top:203.0, right:508.0, bottom:434.0},  
false);
```

## Voir aussi

[document.getSelectionRect\(\)](#), [document.selection](#), [document.setSelectionBounds\(\)](#)



## document.setStageVanishingPoint()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.setStageVanishingPoint(point)
```

### Paramètres

**point** Point qui spécifie les coordonnées *x* et *y* de l'emplacement auquel doit être défini le point de fuite pour l'affichage des objets 3D.

### Valeur renvoyée

Aucune.

### Description

Spécifie le point de fuite pour l'affichage des objets 3D.

### Exemple

L'exemple suivant définit le point de fuite de la scène :

```
fl.getDocumentDOM().setStageVanishingPoint({x:45, y:45});
```

## document.setStageViewAngle()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.setStageViewAngle(angle)
```

### Paramètres

**angle** Valeur en virgule flottante comprise entre 0.0 et 179.0.

### Valeur renvoyée

Aucune.

### Description

Spécifie l'angle de perspective pour l'affichage des objets 3D.

### Exemple

L'exemple suivant définit l'angle de perspective de la scène sur 70° :

```
fl.getDocumentDOM().setStageViewAngle(70);
```

## document.setStroke()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setStroke(color, size, strokeType)
```

### Paramètres

**color** Couleur du trait, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

**size** Valeur en virgule flottante spécifiant la nouvelle taille du trait de la sélection.

**strokeType** Chaîne spécifiant le nouveau type du trait de la sélection. Les valeurs possibles sont : "hairline" (filet), "solid" (uni), "dashed" (discontinu), "dotted" (pointillé), "ragged" (irrégulier), "stipple" (pointillé fin) et "hatched" (hachuré).

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la couleur, la largeur et le style du trait sélectionné. Pour plus d'informations sur la modification du trait dans le panneau Outils ou dans l'inspecteur Propriétés, voir la méthode [document.setCustomStroke\(\)](#).

### Exemple

L'exemple suivant modifie le trait en lui donnant la couleur rouge, une taille de 3,25 et le type pointillé :

```
fl.getDocumentDOM().setStroke("#ff0000", 3.25, "dashed");
```

## document.setStrokeColor()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setStrokeColor(color)
```

### Paramètres

**color** Couleur du trait, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

### Valeur renvoyée

Aucune.

### Description

Méthode qui modifie la couleur du trait de la sélection avec la couleur spécifiée. Pour plus d'informations sur la modification du trait dans le panneau Outils ou dans l'inspecteur Propriétés, voir la méthode `document.setCustomStroke()`.

### Exemple

Dans l'exemple suivant, les trois instructions définissent la couleur du trait à l'aide des trois différents formats de définition de la couleur.

```
fl.getDocumentDOM().setStrokeColor("#cc00cc");  
fl.getDocumentDOM().setStrokeColor(0xcc00cc);  
fl.getDocumentDOM().setStrokeColor(120000);
```

## document.setStrokeSize()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setStrokeSize(size)
```

### Paramètres

**size** Valeur en virgule flottante comprise entre 0,25 et 10 qui détermine la taille du trait. La méthode ignore toute précision supérieure à deux décimales.

### Valeur renvoyée

Aucune.

### Description

Méthode qui modifie la taille de trait de la sélection avec la taille spécifiée. Pour plus d'informations sur la modification du trait dans le panneau Outils ou dans l'inspecteur Propriétés, voir la méthode `document.setCustomStroke()`.

### Exemple

L'exemple suivant définit sur 5 points la taille du trait de la sélection :

```
fl.getDocumentDOM().setStrokeSize(5);
```

## document.setStrokeStyle()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setStrokeStyle(strokeType)
```

### Paramètres

**strokeType** Chaîne spécifiant le style du trait de la sélection. Les valeurs possibles sont : "hairline" (filet), "solid" (uni), "dashed" (discontinu), "dotted" (pointillé), "ragged" (irrégulier), "stipple" (pointillé fin) et "hatched" (hachuré).

### Valeur renvoyée

Aucune.

### Description

Méthode qui modifie le style du trait de la sélection avec le style spécifié. Pour plus d'informations sur la modification du trait dans le panneau Outils ou dans l'inspecteur Propriétés, voir la méthode [document.setCustomStroke\(\)](#).

### Exemple

L'exemple suivant définit un style de trait discontinu ("dashed") pour la sélection :

```
fl.getDocumentDOM().setStrokeStyle("dashed");
```

## document.setTextRectangle()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setTextRectangle(boundingRectangle)
```

### Paramètres

**boundingRectangle** Rectangle qui spécifie la nouvelle taille dans laquelle l'élément texte doit s'insérer. Pour plus d'informations sur le format de *boundingRectangle*, voir [document.addNewRectangle\(\)](#).

### Valeur renvoyée

Valeur booléenne, `true` si la taille d'au moins un champ de texte est modifiée, `false` dans le cas contraire.

### Description

Méthode qui modifie le rectangle de délimitation de l'élément texte sélectionné en lui donnant la taille spécifiée. Cette méthode provoque un reformatage du texte dans le nouveau rectangle ; l'élément texte n'est ni mis à l'échelle ni transformé. Les valeurs transmises dans *boundingRectangle* sont utilisées comme suit :

- Si le texte est horizontal et statique, la méthode ne prend en compte que la largeur transmise dans *boundingRectangle*. La hauteur est automatiquement calculée pour inclure la totalité du texte.
- Si le texte est vertical, donc statique, la méthode ne prend en compte que la hauteur transmise dans *boundingRectangle*. La largeur est automatiquement calculée pour inclure la totalité du texte.

- Si le texte est dynamique ou saisi, la méthode prend en compte la largeur et la hauteur transmises dans *boundingRectangle*, et la taille du rectangle généré risque d'être supérieure à la taille requise par le texte. Cependant, si les paramètres spécifient une taille trop réduite pour le texte, la méthode ne prend en compte que la largeur transmise dans *boundingRectangle*, la hauteur étant automatiquement calculée pour inclure la totalité du texte.

### Exemple

L'exemple suivant adapte la taille du rectangle de délimitation de texte aux dimensions spécifiées :

```
fl.getDocumentDOM().setTextRectangle({left:0, top:0, right:50, bottom:200})
```

## document.setTextSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setTextSelection(startIndex, endIndex)
```

### Paramètres

**startIndex** Nombre entier spécifiant la position du premier caractère à sélectionner. La position du premier caractère est zéro.

**endIndex** Nombre entier spécifiant la position du dernier caractère de la sélection, *endIndex* non compris. La position du premier caractère est zéro.

### Valeur renvoyée

Valeur booléenne, `true` si la méthode définit bien la sélection du texte, `false` dans le cas contraire.

### Description

Méthode qui définit la sélection de texte du champ de texte sélectionné avec les valeurs spécifiées dans *startIndex* et *endIndex*. Le mode de modification de texte est activé s'il ne l'était pas déjà.

### Exemple

L'exemple suivant sélectionne le texte du 6e au 25e caractère :

```
fl.document.setTextSelection(5, 25);
```

## document.setTextString()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setTextString(text [, startIndex [, endIndex]])
```

### Paramètres

**text** Chaîne de caractères à insérer dans le champ de texte.

**startIndex** Entier qui spécifie le premier caractère à remplacer. La position du premier caractère est zéro. Ce paramètre est facultatif.

**endIndex** Nombre entier spécifiant la position du dernier caractère à remplacer. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si le texte d'au moins une chaîne de texte est modifié, `false` dans le cas contraire.

### Description

Méthode qui insère une chaîne de texte. Si les paramètres facultatifs ne sont pas transmis, la méthode remplace la sélection de texte existante. Si l'objet Text n'est pas en cours de modification, toute la chaîne de texte est remplacée. Si seul le paramètre *startIndex* est transmis, la chaîne transmise est insérée à cette position. Si *startIndex* et *endIndex* sont transmis, la chaîne transmise remplace le segment de texte compris entre *startIndex* et le caractère précédant *endIndex*.

### Exemple

L'exemple suivant remplace la sélection de texte actuelle par la chaîne « Hello World » :

```
var success = fl.getDocumentDOM().setTextString("Hello World!");
```

L'exemple suivant insère « hello » à la position 6 de la sélection de texte actuelle :

```
var pass = fl.getDocumentDOM().setTextString("hello", 6);
```

L'exemple suivant insère « Howdy » entre la position 2 et la position 7 (non comprise) de la sélection de texte actuelle :

```
var ok = fl.getDocumentDOM().setTextString("Howdy", 2, 7);
```

### Voir aussi

[document.getTextString\(\)](#)

## document.setTransformationPoint()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.setTransformationPoint( transformationPoint )
```

### Paramètres

**transformationPoint** Point (par exemple, {x:10,y:20}, où x et y sont des nombres à virgule flottante) spécifiant des valeurs pour le point de transformation de chacun des éléments suivants :

- Formes : *transformationPoint* est défini par rapport au document (0,0 est le coin supérieur gauche de la scène).
- Symboles : *transformationPoint* est défini par rapport au point d'alignement du symbole (0,0 est situé au point d'alignement).
- Texte : *transformationPoint* est défini par rapport au champ de texte (0,0 est le coin supérieur gauche du champ de texte).

- Bitmaps/vidéos : *transformationPoint* est défini par rapport à l'image bitmap ou à la vidéo (0,0 est le coin supérieur gauche de l'image bitmap ou de la vidéo).
- Objets de dessin, ovales et rectangles de type primitif, groupes : *transformationPoint* est défini par rapport au document (0,0 étant le coin supérieur gauche de la scène). Pour définir *transformationPoint* par rapport au point central de l'objet, de la primitive ou du groupe, utilisez la méthode `element.setTransformationPoint()`.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui définit la position du point de transformation de la sélection actuelle.

#### Exemple

L'exemple suivant définit l'emplacement du point de transformation de la sélection sur les coordonnées 100, 200 :

```
fl.getDocumentDOM().setTransformationPoint({x:100, y:200});
```

#### Voir aussi

`document.getTransformationPoint()`, `element.setTransformationPoint()`

## document.silent

#### Disponibilité

Flash MX 2004.

#### Utilisation

`document.silent`

#### Description

Propriété ; valeur booléenne indiquant si l'objet est accessible. Cette propriété est équivalente à l'inverse du paramètre Rendre l'animation accessible du panneau Accessibilité. Ainsi, si `document.silent` est `true`, son effet est similaire à la désactivation de l'option Rendre l'animation accessible. Si le paramètre est `false`, son effet est similaire à l'activation de l'option Rendre l'animation accessible.

#### Exemple

L'exemple suivant définit la valeur de la variable `isSilent` sur la valeur de la propriété `silent` :

```
var isSilent = fl.getDocumentDOM().silent;
```

L'exemple suivant définit la propriété `silent` sur `false` indiquant ainsi que le document est accessible :

```
fl.getDocumentDOM().silent = false;
```

## document.skewSelection()

#### Disponibilité

Flash MX 2004.

### Utilisation

```
document.skewSelection(xSkew, ySkew [, whichEdge])
```

### Paramètres

**xSkew** Valeur en virgule flottante spécifiant la valeur  $x$  d'inclinaison, exprimée en degrés.

**ySkew** Valeur en virgule flottante spécifiant la valeur  $y$  d'inclinaison, exprimée en degrés.

**whichEdge** Chaîne indiquant le côté sur lequel la transformation doit être appliquée. Si ce paramètre est omis, l'inclinaison est appliquée au niveau du point de transformation. Les valeurs gérées sont "top center" (centre haut), "right center" (centre droite), "bottom center" (centre bas) et "left center" (centre gauche). Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui effectue sur la sélection une inclinaison de la valeur spécifiée. L'effet est identique à celui de l'outil Transformation libre utilisé pour incliner l'objet.

### Exemple

Les exemples suivants inclinent l'objet sélectionné de 2° verticalement et de 1,5° horizontalement. Le second exemple transforme l'objet sur le bord centre haut :

```
fl.getDocumentDOM().skewSelection(2.0, 1.5);  
fl.getDocumentDOM().skewSelection(2.0, 1.5, "top center");
```

## document.smoothSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.smoothSelection()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui lisse la courbe de chaque contour de remplissage ou segment incurvé sélectionné. Cette méthode effectue la même opération que le bouton Lisser dans le panneau Outils.

### Exemple

L'exemple suivant lisse la courbe de la sélection :

```
fl.getDocumentDOM().smoothSelection();
```



## document.sourcePath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`document.sourcePath`

### Description

Propriété ; chaîne qui contient la liste des éléments du chemin source ActionScript 3.0 du document, spécifiant l'emplacement des fichiers de classe ActionScript. Les éléments de la chaîne sont délimités par des points-virgules. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Fichier > Paramètres de publication, puis en cliquant sur le bouton Paramètres de Script ActionScript 3.0 dans l'onglet Flash.

### Exemple

L'exemple suivant ajoute le dossier /Class files dans le chemin source du document :

```
var myDoc = fl.getDocumentDOM();  
fl.trace(myDoc.sourcePath);  
myDoc.sourcePath = "./Class files;" + myDoc.sourcePath;  
fl.trace(myDoc.sourcePath);
```

### Voir aussi

[document.externalLibraryPath](#), [document.libraryPath](#), [fl.sourcePath](#)

## document.space()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.space(direction [, bUseDocumentBounds])`

### Paramètres

**direction** Chaîne spécifiant la direction dans laquelle doivent être espacés les objets de la sélection. Les valeurs gérées sont "horizontal" ou "vertical".

**bUseDocumentBounds** Valeur booléenne. Si elle est `true`, la méthode espace les objets sur les limites du document. Sinon, la méthode utilise les limites des objets sélectionnés. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui espace régulièrement les objets sélectionnés.

### Exemple

L'exemple suivant espace les objets horizontalement par rapport aux limites de la scène :

```
fl.getDocumentDOM().space("horizontal",true);
```

L'exemple suivant espace les objets entre eux sur le plan horizontal :

```
fl.getDocumentDOM().space("horizontal");
```

L'exemple suivant espace les objets horizontalement les uns par rapport aux autres, mais avec *bUseDocumentBounds* défini explicitement sur *false* :

```
fl.getDocumentDOM().space("horizontal",false);
```

### Voir aussi

[document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

## document.straightenSelection()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.straightenSelection()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui redresse les traits sélectionnés. Cette méthode est l'équivalent du bouton Redresser du panneau Outils.

### Exemple

L'exemple suivant redresse la courbe de la sélection :

```
fl.getDocumentDOM().straightenSelection();
```

## document.swapElement()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.swapElement(name)
```

#### Paramètres

**name** Chaîne spécifiant le nom de l'élément de bibliothèque à utiliser.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui intervertit la sélection actuelle avec la sélection spécifiée. La sélection doit contenir un graphisme, un bouton, un clip, une vidéo ou un bitmap. S'il n'existe pas de sélection ou si l'objet indiqué est introuvable, un message d'erreur s'affiche.

#### Exemple

L'exemple suivant intervertit la sélection avec l'objet `Symbol 1` de la bibliothèque :

```
fl.getDocumentDOM().swapElement('Symbol 1');
```

## document.swapStrokeAndFill()

#### Disponibilité

Flash 8.

#### Utilisation

```
document.swapStrokeAndFill()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui permute les couleurs du contour et du remplissage.

#### Exemple

L'exemple suivant permute les couleurs du contour et du remplissage dans le document actuel :

```
fl.getDocumentDOM().swapStrokeAndFill();
```

## document.testMovie()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.testMovie([Boolean abortIfErrorsExist])
```

### Paramètres

**abortIfErrorsExist** Boolean ; la valeur par défaut est false. Si cette valeur est définie sur true, la session de test de l'animation ne démarre pas et la fenêtre .swf ne s'ouvre pas en cas d'erreurs du compilateur. Les erreurs du compilateur n'annulent pas la commande. Ce paramètre a été ajouté dans Flash Professional CS5.

### Valeur renvoyée

Aucune.

### Description

Méthode qui exécute une opération de test de l'animation sur le document.

### Exemple

L'exemple suivant teste l'animation pour le document en cours, mais abandonne le test de l'animation en cas d'erreurs du compilateur :

```
fl.getDocumentDOM().testMovie(1);
```

### Voir aussi

[document.canTestMovie\(\)](#), [document.testScene\(\)](#)

## document.testScene()

### Disponibilité

Flash MX 2004.

### Utilisation

```
document.testScene()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui exécute une opération Tester la séquence sur la séquence active dans le document.

### Exemple

L'exemple suivant teste la séquence active du document :

```
fl.getDocumentDOM().testScene();
```

### Voir aussi

[document.canTestScene\(\)](#), [document.testMovie\(\)](#)

## document.timelines

### Disponibilité

Flash MX 2004.

### Utilisation

`document.timelines`

### Description

Propriété en lecture seule ; tableau des objets Timeline (voir [Objet Timeline](#)).

### Exemple

L'exemple suivant récupère le tableau des scénarios du document actif et affiche leurs noms dans le panneau Sortie :

```
var i = 0;
var curTimelines = fl.getDocumentDOM().timelines;
while(i < fl.getDocumentDOM().timelines.length) {
    alert(curTimelines[i].name);
    ++i;
}
```

### Voir aussi

[document.currentTimeline](#), [document.getTimeline\(\)](#)

## document.traceBitmap()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.traceBitmap(threshold, minimumArea, curveFit, cornerThreshold)`

### Paramètres

**threshold** Nombre entier limitant le nombre de couleurs du bitmap faisant l'objet du tracé. Les valeurs gérées sont des entiers compris entre 0 et 500.

**minimumArea** Nombre entier spécifiant le rayon, indiqué en pixels. Les valeurs gérées sont des entiers compris entre 1 et 1000.

**curveFit** Chaîne spécifiant le niveau de lissage des contours. Les valeurs gérées sont "pixels", "very tight" (très serré), "tight" (serré), "normal", "smooth" (lisse) et "very smooth" (très lisse).

**cornerThreshold** Chaîne dont l'effet est similaire à celui de *curveFit*, mais concernant les angles de l'image bitmap. Les valeurs gérées sont "many corners" (nombreux angles), "normal" et "few corners" (peu d'angles).

### Valeur renvoyée

Aucune.

### Description

Méthode qui effectue une opération de tracé de bitmap sur la sélection en cours. Cette méthode est l'équivalent de l'option de menu Modification > Bitmap > Tracer le bitmap.

### Exemple

L'exemple suivant trace le bitmap sélectionné sur la base des paramètres spécifiés :

```
fl.getDocumentDOM().traceBitmap(0, 500, 'normal', 'normal');
```

## document.translate3DCenter()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.translate3DCenter(xyzCoordinate)
```

### Paramètres

**xyzCoordinate** Coordonnées XYZ définissant le point central d'une translation ou d'une rotation 3D.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la position XYZ autour de laquelle est effectuée la translation ou la rotation de la sélection. Cette méthode n'est disponible que pour les clips.

### Exemple

L'exemple suivant spécifie les axes XYZ d'une translation 3D :

```
fl.getDocumentDOM().translate3DCenter({x:180, y:18,z:-30});
```

## document.translate3DSelection()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
document.translate3DSelection(xyzCoordinate, bGlobalTransform)
```

### Paramètres

**xyzCoordinate** Coordonnées XYZ qui spécifient les axes d'une translation 3D.

**bGlobalTransform** Valeur booléenne qui spécifie si le mode de transformation doit être global (`true`) ou local (`false`).

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui applique une translation 3D à la sélection. Cette méthode n'est disponible que pour les clips.

#### Exemple

Dans l'exemple suivant, une translation est d'abord appliquée à la sélection par rapport à la scène (globalement), puis par rapport à elle-même (localement).

```
var myDocument = fl.getDocumentDOM();  
myDocument.translate3DSelection({x:52.0, y:0, z:0}, true);  
myDocument.translate3DSelection({x:52.0, y:0, z:-55.2}, false);
```

#### Voir aussi

[document.translate3DCenter\(\)](#)

## document.transformSelection()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.transformSelection(a, b, c, d)
```

#### Paramètres

- a** Nombre en virgule flottante spécifiant l'élément (0,0) de la matrice de transformation.
- b** Nombre en virgule flottante spécifiant l'élément (0,1) de la matrice de transformation.
- c** Nombre en virgule flottante spécifiant l'élément (1,0) de la matrice de transformation.
- d** Nombre en virgule flottante spécifiant l'élément (1,1) de la matrice de transformation.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui effectue une transformation générale de la sélection, en appliquant la matrice spécifiée dans les arguments. Pour plus d'informations, voir la propriété [element.matrix](#).

#### Exemple

L'exemple suivant étire la sélection d'un facteur 2 dans la direction x :

```
fl.getDocumentDOM().transformSelection(2.0, 0.0, 0.0, 1.0);
```

## document.unGroup()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.unGroup()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui dissocie la sélection.

### Exemple

L'exemple suivant dissocie les éléments de la sélection :

```
fl.getDocumentDOM().unGroup();
```

### Voir aussi

[document.group\(\)](#)

## document.union()

### Disponibilité

Flash 8.

### Utilisation

`document.union()`

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne : `true` en cas de réussite, `false` dans le cas contraire.

### Description

Méthode qui combine toutes les formes sélectionnées dans un objet de dessin.

### Exemple

L'exemple suivant combine toutes les formes sélectionnées dans un objet de dessin :

```
fl.getDocumentDOM().union();
```



#### Voir aussi

[document.crop\(\)](#), [document.deleteEnvelope\(\)](#), [document.intersect\(\)](#), [document.punch\(\)](#),  
[shape.isDrawingObject](#)

## document.unlockAllElements()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.unlockAllElements()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui déverrouille tous les éléments verrouillés dans l'image sélectionnée.

#### Exemple

L'exemple suivant déverrouille tous les éléments verrouillés dans l'image active :

```
fl.getDocumentDOM().unlockAllElements();
```

#### Voir aussi

[element.locked](#)

## document.viewMatrix

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
document.viewMatrix
```

#### Description

Propriété en lecture seule ; objet Matrix. La propriété `viewMatrix` permet de passer de l'espace de l'objet à l'espace du document lorsque ce dernier est en mode de modification. Les coordonnées de la souris, telles qu'elles sont reçues par un outil, sont relatives à l'objet en cours de modification. Voir [Objet Matrix](#).

Par exemple, si vous créez un symbole, double-cliquez sur son entrée pour le modifier et dessinez à l'aide de l'outil Polygone, le point (0,0) se trouve sur le point d'alignement du symbole. Toutefois, l'objet `drawingLayer` attend des valeurs situées dans l'espace du document. Donc, si vous tracez une ligne à partir des coordonnées (0,0) avec `drawingLayer`, ces coordonnées seront relatives au coin supérieur gauche de la scène. La propriété `viewMatrix` permet de passer de l'espace de l'objet modifié à l'espace du document.

### Exemple

L'exemple suivant récupère la valeur de la propriété `viewMatrix` :

```
var mat = fl.getDocumentDOM().viewMatrix;
```

## document.width

### Disponibilité

Flash MX 2004.

### Utilisation

`document.width`

### Description

Propriété ; nombre entier indiquant la largeur en pixels du document (scène).

### Exemple

L'exemple suivant définit la largeur de la scène sur 400 pixels.

```
fl.getDocumentDOM().width= 400;
```

### Voir aussi

[document.height](#)

## document.xmlPanel()

### Disponibilité

Flash MX 2004.

### Utilisation

`document.xmlPanel (fileURI)`

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le chemin d'accès du fichier XML définissant les contrôles apparaissant dans le panneau. Le chemin complet est nécessaire.

### Valeur renvoyée

Un objet dont les propriétés sont définies pour tous les contrôles définis dans le fichier XML. Toutes les propriétés sont renvoyées sous forme de chaînes. L'objet renvoyé comporte une propriété prédéfinie appelée "dismiss" qui possède la valeur de chaîne "accept" ou "cancel".

### Description

Méthode qui transmet une boîte de dialogue XMLUI. Voir [fl.xmlui](#).

### Exemple

L'exemple suivant charge le fichier Test.xml et affiche chacune de ses propriétés :

```
var obj = fl.getDocumentDOM().xmlPanel(fl.configURI + "Commands/Test.xml");  
for (var prop in obj) {  
    fl.trace("property " + prop + " = " + obj[prop]);  
}
```

## document.zoomFactor

### Disponibilité

Flash 8.

### Utilisation

`document.zoomFactor`

### Description

Propriété qui spécifie le pourcentage de zoom de la scène lors de la programmation. Une valeur de 1 équivaut à un zoom de 100 %, 8 à 800 %, 0.5 à 50 %, etc.

### Exemple

L'exemple suivant définit le facteur de zoom de la scène sur 200 %.

```
fl.getDocumentDOM().zoomFactor = 2;
```

# Chapitre 12 : Objet drawingLayer

## Disponibilité

Flash MX 2004.

## Description

L'objet `drawingLayer` est accessible en JavaScript sous forme d'un enfant de l'objet `flash`. L'objet `drawingLayer` est employé par les outils extensibles lorsque l'utilisateur dessine un tracé temporaire à l'aide de la souris, par exemple pour créer un cadre de sélection. Il est conseillé d'appeler `drawingLayer.beginFrame()` avant d'appeler toute autre méthode de `drawingLayer`.

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet `drawingLayer` :

Méthode	Description
<code>drawingLayer.beginDraw()</code>	Active le mode Dessin de Flash.
<code>drawingLayer.beginFrame()</code>	Efface ce qui a été dessiné précédemment avec <code>drawingLayer</code> et prépare le programme pour d'autres commandes de dessin.
<code>drawingLayer.cubicCurveTo()</code>	Dessine une courbe cubique à partir de l'emplacement actuel du curseur, en utilisant les paramètres comme coordonnées du segment cubique.
<code>drawingLayer.curveTo()</code>	Dessine un segment de courbe quadratique commençant à l'emplacement actuel du curseur et se terminant au point spécifié.
<code>drawingLayer.drawPath()</code>	Dessine le tracé spécifié.
<code>drawingLayer.endDraw()</code>	Désactive le mode Dessin.
<code>drawingLayer.endFrame()</code>	Signale la fin d'un groupe de commandes de dessin.
<code>drawingLayer.lineTo()</code>	Dessine une ligne à partir de la position du curseur et jusqu'au point (x,y).
<code>drawingLayer.moveTo()</code>	Définit la position de dessin actuelle.
<code>drawingLayer.newPath()</code>	Renvoie un nouvel <a href="#">Objet Path</a> .
<code>drawingLayer.setColor()</code>	Définit la couleur des prochains dessins.
<code>drawingLayer.setFill()</code>	Cette méthode n'est pas disponible.
<code>drawingLayer.setStroke()</code>	Cette méthode n'est pas disponible.

## drawingLayer.beginDraw()

## Disponibilité

Flash MX 2004.

## Utilisation

```
drawingLayer.beginDraw([persistentDraw])
```

### Paramètres

**`persistentDraw`** Valeur booléenne (facultative) Si sa valeur est `true`, elle indique que le dessin dans le dernier cadre reste sur la scène jusqu'au prochain appel de `beginDraw()` ou `beginFrame()`. (Dans ce contexte, *cadre* fait référence à l'espace du dessin, pas aux images dans le scénario.) Par exemple, lorsque l'utilisateur trace un rectangle, il peut avoir un aperçu du contour de cette forme pendant qu'il déplace sa souris. Si vous souhaitez que ce contour reste visible lorsque l'utilisateur relâche le bouton de sa souris, donnez au paramètre *`persistentDraw`* la valeur `true`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui active le mode Dessin de Flash. Le mode dessin est utilisé pour les tracés temporaires avec le bouton de la souris enfoncé. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant active le mode Dessin de Flash :

```
fl.drawingLayer.beginDraw();
```

## `drawingLayer.beginFrame()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.beginFrame()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui efface ce qui a été dessiné précédemment avec `drawingLayer` et prépare le programme pour d'autres commandes de dessin. Cette méthode doit être appelée après `drawingLayer.beginDraw()`. Tout ce qui est tracé entre un appel de `drawingLayer.beginFrame()` et un appel de `drawingLayer.endFrame()` reste sur la scène jusqu'aux prochains appels de `beginFrame()` et `endFrame()`. (Dans ce contexte, *cadre* fait référence à l'espace du dessin, pas aux images dans le scénario.) En général, cette méthode n'est utilisée que lors de la création d'outils extensibles. Voir [drawingLayer.beginDraw\(\)](#).

## `drawingLayer.cubicCurveTo()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.cubicCurveTo(x1Ctrl, y1Ctrl, x2Ctrl, y2Ctrl, xEnd, yEnd)
```

### Paramètres

**x1Ctrl** Valeur en virgule flottante indiquant la coordonnée  $x$  du premier point de contrôle.

**y1Ctrl** Valeur en virgule flottante indiquant la coordonnée  $y$  du premier point de contrôle.

**x2Ctrl** Valeur en virgule flottante indiquant la coordonnée  $x$  du point de contrôle central.

**y2Ctrl** Valeur en virgule flottante indiquant la coordonnée  $y$  du point de contrôle central.

**xEnd** Valeur en virgule flottante indiquant la coordonnée  $x$  du point de contrôle final.

**yEnd** Valeur en virgule flottante indiquant la coordonnée  $y$  du point de contrôle final.

### Valeur renvoyée

Aucune.

### Description

Méthode qui dessine une courbe cubique à partir de l'emplacement actuel du curseur, en utilisant les paramètres comme coordonnées du segment cubique. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant trace une courbe cubique en utilisant les points de contrôle spécifiés :

```
fl.drawingLayer.cubicCurveTo(0, 0, 1, 1, 2, 0);
```

## **drawingLayer.curveTo()**

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.curveTo(xCtl, yCtl, xEnd, yEnd)
```

### Paramètres

**xCtl** Valeur en virgule flottante indiquant la coordonnée  $x$  du point de contrôle.

**yCtl** Valeur en virgule flottante indiquant la coordonnée  $y$  du point de contrôle.

**xEnd** Valeur en virgule flottante indiquant la coordonnée  $x$  du point de contrôle final.

**yEnd** Valeur en virgule flottante indiquant la coordonnée  $y$  du point de contrôle final.

### Valeur renvoyée

Aucune.

### Description

Méthode qui dessine un segment de courbe quadratique commençant à l'emplacement actuel du curseur et se terminant au point spécifié. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant trace une courbe quadratique en utilisant les points de contrôle spécifiés :

```
fl.drawingLayer.curveTo(0, 0, 2, 0);
```

## `drawingLayer.drawPath()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.drawPath(path)
```

### Paramètres

**path** Objet [Objet Path](#) à dessiner.

### Valeur renvoyée

Aucune.

### Description

Méthode qui dessine le tracé spécifié par le paramètre *path*. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant dessine le tracé spécifié par l'objet Path nommé `gamePath` :

```
fl.drawingLayer.drawPath(gamePath);
```

## `drawingLayer.endDraw()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.endDraw()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui quitte le mode dessin. Le mode dessin est utilisé pour les tracés temporaires avec le bouton de la souris enfoncé. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant quitte le mode dessin :

```
fl.drawingLayer.endDraw();
```

## `drawingLayer.endFrame()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.endFrame();
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui signale la fin d'un groupe de commandes de dessin. Un groupe de commandes de dessin représente tout ce qui est tracé entre les appels de `drawingLayer.beginFrame()` et `drawingLayer.endFrame()`. L'appel suivant de `drawingLayer.beginFrame()` efface tout ce qui a été tracé par ce groupe de commandes de dessin. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

## `drawingLayer.lineTo()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.lineTo(x, y)
```

### Paramètres

**x** Valeur en virgule flottante indiquant la coordonnée *x* du point de contrôle final de la ligne à tracer.

**y** Valeur en virgule flottante indiquant la coordonnée *y* du point de contrôle final de la ligne à tracer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui dessine une ligne à partir de la position actuelle du curseur et jusqu'au point (*x,y*). En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.



### Exemple

L'exemple suivant dessine une ligne à partir de la position actuelle du curseur et jusqu'au point (20,30) :

```
fl.drawingLayer.lineTo(20, 30);
```

## `drawingLayer.moveTo()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.moveTo(x, y)
```

### Paramètres

**x** Valeur en virgule flottante indiquant la coordonnée *x* de la position de début du dessin.

**y** Valeur en virgule flottante indiquant la coordonnée *y* de la position de début du dessin.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la position actuelle de dessin. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant définit la position de début du dessin au point (10,15) :

```
fl.drawingLayer.moveTo(10, 15);
```

## `drawingLayer.newPath()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.newPath()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un objet Path (tracé).

### Description

Méthode qui renvoie un nouvel objet `Path`. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles. Voir l'[Objet Path](#).

### Exemple

L'exemple suivant renvoie un nouvel objet `Path` :

```
fl.drawingLayer.newPath();
```

## `drawingLayer.setColor()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
drawingLayer.setColor(color)
```

### Paramètres

**color** Définit la couleur des objets dessin à venir, dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la couleur des objets dessin à venir. S'applique uniquement aux données persistantes. Pour utiliser cette méthode, le paramètre transmis à `drawingLayer.beginDraw()` doit être défini sur `true`. En général, cette méthode n'est utilisée que lors de la création d'outils extensibles. Voir [drawingLayer.beginDraw\(\)](#).

### Exemple

L'exemple suivant trace une ligne rouge sur la scène :

```
fl.drawingLayer.beginDraw( true );  
fl.drawingLayer.beginFrame();  
fl.drawingLayer.setColor( "#ff0000" );  
fl.drawingLayer.moveTo(0,0);  
fl.drawingLayer.lineTo(100,100);  
fl.drawingLayer.endFrame();  
fl.drawingLayer.endDraw();
```

## **drawingLayer.setFill()**

Cette méthode n'est pas disponible.

## **drawingLayer.setStroke()**

Cette méthode n'est pas disponible.

# Chapitre 13 : Objet Edge

## Disponibilité

Flash MX 2004.

## Description

L'objet Edge représente les limites d'une forme sur la scène.

## Résumé des méthodes

Les méthodes suivantes sont disponibles pour l'objet Edge :

Méthode	Description
<code>edge.getControl()</code>	Récupère un objet point situé à l'emplacement du point de contrôle spécifié du bord.
<code>edge.getHalfEdge()</code>	Renvoie un <a href="#">Objet halfEdge</a> .
<code>edge.setControl()</code>	Définit la position du point de contrôle du bord.
<code>edge.splitEdge()</code>	Fractionne le bord en deux parties.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Edge :

Propriété	Description
<code>edge.cubicSegmentIndex</code>	Nombre entier qui spécifie la valeur d'index d'un segment cubique du bord.
<code>edge.id</code>	En lecture seule ; entier représentant un identificateur unique pour le bord.
<code>edge.isLine</code>	Lecture seule ; entier d'une valeur de 0 ou 1.
<code>edge.stroke</code>	Objet <a href="#">Objet Stroke</a> .

## edge.cubicSegmentIndex

## Disponibilité

Flash CS4 Professional.

## Utilisation

`edge.cubicSegmentIndex`

## Description

Propriété en lecture seule ; entier qui spécifie la valeur d'index d'un segment cubique du bord (voir [shape.getCubicSegmentPoints\(\)](#)).

## Exemple

Le code suivant affiche les valeurs d'index de tous les segments cubiques du bord spécifié :

```
var theShape = fl.getDocumentDOM().selection[0];  
var edgesArray = theShape.edges;  
for (var i=0; i<edgesArray.length; i++) {  
    fl.trace(edgesArray[i].cubicSegmentIndex);  
}
```

## edge.getControl()

### Disponibilité

Flash MX 2004.

### Utilisation

```
edge.getControl(i)
```

### Paramètres

**i** Entier qui spécifie les points de contrôle du bord à renvoyer. Indiquez 0 pour le premier point de contrôle, 1 pour le point de contrôle central ou 2 pour le point de contrôle final. Si la propriété [edge.isLine](#) est `true`, le point de contrôle central est défini sur le milieu du segment joignant les points de contrôle de début et de fin.

### Valeur renvoyée

Le point de contrôle spécifié.

### Description

Méthode qui récupère un objet point situé à l'emplacement du point de contrôle spécifié du bord.

### Exemple

L'exemple suivant stocke dans la variable `pt` le premier point de contrôle de la forme spécifiée :

```
var shape = fl.getDocumentDOM().selection[0];  
var pt = shape.edges[0].getControl(0);
```

## edge.getHalfEdge()

### Disponibilité

Flash MX 2004.

### Utilisation

```
edge.getHalfEdge(index)
```

### Paramètres

**index** Entier qui spécifie le demi-bord à renvoyer. La valeur de *index* doit être 0 pour le premier demi-bord ou 1 pour le second demi-bord.

### Valeur renvoyée

Un objet HalfEdge.

### Description

Méthode qui renvoie un objet [Objet halfEdge](#).

### Exemple

L'exemple suivant enregistre dans les variables hEdge0 et hEdge1 les deux moitiés du bord spécifié :

```
var shape = fl.getDocumentDOM().selection[0];  
var edge = shape.edges[0];  
var hEdge0 = edge.getHalfEdge(0);  
var hEdge1 = edge.getHalfEdge(1);
```

## edge.id

### Disponibilité

Flash MX 2004.

### Utilisation

edge.id

### Description

Propriété en lecture seule ; entier représentant un identifiant unique pour le bord.

### Exemple

L'exemple suivant enregistre dans la variable my\_shape\_id un identificateur unique pour le bord spécifié :

```
var shape = fl.getDocumentDOM().selection[0];  
var my_shape_id = shape.edges[0].id;
```

## edge.isLine

### Disponibilité

Flash MX 2004.

### Utilisation

edge.isLine

### Description

Propriété en lecture seule ; entier ayant la valeur 0 ou 1. La valeur 1 indique que le bord est une ligne droite. Dans ce cas, le point de contrôle central coupe en deux parts égales la ligne qui relie les deux points d'extrémité.

### Exemple

L'exemple suivant détermine si le bord spécifié est une ligne droite, et affiche la valeur 1 (si c'est une ligne droite) ou 0 (si ce n'est pas le cas) dans le panneau Sortie :

```
var shape = fl.getDocumentDOM().selection[0];  
fl.trace(shape.edges[0].isLine);
```

## edge.setControl()

### Disponibilité

Flash MX 2004.

### Utilisation

```
edge.setControl(index, x, y)
```

### Paramètres

**index** Nombre entier qui spécifie le point de contrôle à définir. Les valeurs 0, 1 ou 2 permettent respectivement de spécifier les points de contrôle de début, de milieu et de fin.

**x** Valeur en virgule flottante indiquant la coordonnée horizontale du point de contrôle. Si la scène est en mode de modification ou de modification en place, les coordonnées du point sont calculées par rapport à l'objet en cours de modification. Sinon, les coordonnées du point sont relatives à la scène.

**y** Valeur en virgule flottante indiquant la coordonnée verticale du point de contrôle. Si la scène est en mode de modification ou de modification en place, les coordonnées du point sont calculées par rapport à l'objet en cours de modification. Sinon, les coordonnées du point sont relatives à la scène.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la position du point de contrôle du bord. Il est nécessaire d'appeler `shape.beginEdit()` avant d'utiliser cette méthode. Voir [shape.beginEdit\(\)](#).

### Exemple

L'exemple suivant définit le point de contrôle de début du bord spécifié sur les coordonnées (0, 1) :

```
x = 0; y = 1;
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.edges[0].setControl(0, x, y);
shape.endEdit();
```

## edge.splitEdge()

### Disponibilité

Flash MX 2004.

### Utilisation

```
edge.splitEdge(t)
```

### Paramètres

**t** Valeur en virgule flottante comprise entre 0 et 1 indiquant le point de fractionnement du bord. La valeur 0 représente un point d'extrémité et 1 l'autre extrémité. Par exemple, la valeur 0,5 fractionne le bord en son milieu, c'est-à-dire en son centre exact pour une ligne. Si le bord forme une courbe, 0,5 représente le milieu paramétrique de la courbe.

### Valeur renvoyée

Aucune.

### Description

Méthode qui fractionne le bord en deux parties. Il est nécessaire d'appeler `shape.beginEdit()` avant d'utiliser cette méthode.

### Exemple

L'exemple suivant fractionne le bord spécifié en deux moitiés :

```
var shape = fl.getDocumentDOM().selection[0];  
shape.beginEdit()  
shape.edges[0].splitEdge( 0.5 );  
shape.endEdit()
```

## edge.stroke

### Disponibilité

Flash CS4 Professional.

### Utilisation

`edge.stroke`

### Description

Propriété ; [Objet Stroke](#).

### Exemple

L'exemple suivant affiche la couleur de trait du premier bord de l'objet sélectionné :

```
var shape = fl.getDocumentDOM().selection[0];  
fl.trace(shape.edges[0].stroke.color);
```



# Chapitre 14 : Objet Element

## Disponibilité

Flash MX 2004.

## Description

Tout ce qui apparaît sur la scène est du type Element. Par exemple, le code suivant permet de sélectionner un élément :

```
var el = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
```

## Résumé des méthodes

Les méthodes suivantes sont disponibles pour l'objet Element :

Méthode	Description
<code>element.getPersistentData()</code>	Récupère la valeur des données spécifiées par le paramètre <i>name</i> .
<code>element.getTransformationPoint()</code>	Récupère la valeur du point de transformation de l'élément spécifié.
<code>element.hasPersistentData()</code>	Détermine si les données spécifiées ont été associées à l'élément spécifié.
<code>element.removePersistentData()</code>	Supprime les données persistantes associées à l'objet et correspondant au nom spécifié.
<code>element.setPersistentData()</code>	Stocke des données avec un élément.
<code>element.setTransformationPoint()</code>	Définit la position du point de transformation de l'élément.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Element :

Propriété	Description
<code>element.depth</code>	Lecture seule ; entier ayant une valeur supérieure à 0 et représentant la profondeur de l'objet dans l'affichage.
<code>element.elementType</code>	Lecture seule ; chaîne représentant le type de l'élément spécifié.
<code>element.height</code>	Nombre en virgule flottante spécifiant la hauteur de l'élément, en pixels.
<code>element.layer</code>	Lecture seule ; représente l'objet <a href="#">Objet Layer</a> sur lequel l'élément est situé.
<code>element.left</code>	Lecture seule ; nombre en virgule flottante indiquant la coordonnée gauche de l'élément.
<code>element.locked</code>	Valeur booléenne, <i>true</i> si l'élément est verrouillé, <i>false</i> dans le cas contraire.
<code>element.matrix</code>	<a href="#">Objet Matrix</a> . La matrice possède les propriétés <i>a</i> , <i>b</i> , <i>c</i> , <i>d</i> , <i>tx</i> et <i>ty</i> . Les propriétés <i>a</i> , <i>b</i> , <i>c</i> et <i>d</i> sont des nombres en virgule flottante. Les propriétés <i>tx</i> et <i>ty</i> sont des coordonnées.
<code>element.name</code>	Chaîne spécifiant le nom de l'élément. Cette chaîne est généralement appelée nom d'occurrence.
<code>element.rotation</code>	Valeur en virgule flottante ou entier compris(e) entre -180 et 180 qui indique, en degrés, la rotation de l'objet dans le sens horaire.
<code>element.scaleX</code>	Valeur en virgule flottante qui précise la valeur d'échelle x des rectangles et ovales de type primitif, des symboles et des objets de dessin.

Propriété	Description
<code>element.scaleY</code>	Valeur en virgule flottante qui précise la valeur d'échelle y des rectangles et ovales de type primitif, des symboles et des objets de dessin.
<code>element.selected</code>	Valeur booléenne indiquant si l'objet est sélectionné ou non.
<code>element.skewX</code>	Valeur en virgule flottante, comprise entre -180 et 180, et précisant la valeur d'inclinaison x des rectangles et ovales de type primitif, des symboles et des objets de dessin.
<code>element.skewY</code>	Valeur en virgule flottante, comprise entre -180 et 180, et précisant la valeur d'inclinaison y des rectangles et ovales de type primitif, des symboles et des objets de dessin.
<code>element.top</code>	Lecture seule ; coordonnée supérieure de l'élément.
<code>element.transformX</code>	Nombre à virgule flottante spécifiant la valeur x du point de transformation de l'élément sélectionné dans le système de coordonnées du parent de l'élément.
<code>element.transformY</code>	Nombre à virgule flottante spécifiant la valeur y du point de transformation de l'élément sélectionné dans le système de coordonnées du parent de l'élément.
<code>element.width</code>	Nombre en virgule flottante spécifiant la largeur de l'élément, en pixels.
<code>element.x</code>	Valeur en virgule flottante précisant la valeur x du point d'alignement de l'élément sélectionné.
<code>element.y</code>	Valeur en virgule flottante précisant la valeur y du point d'alignement de l'élément sélectionné.

## element.depth

### Disponibilité

Flash MX 2004.

### Utilisation

`element.depth`

### Description

Propriété en lecture seule ; entier ayant une valeur supérieure à 0 et représentant la profondeur de l'objet dans l'affichage. L'ordre dans lequel les objets sont tracés sur la scène spécifie l'objet qui figure au-dessus des autres. L'ordre des objets peut également être géré par l'option de menu Modification > Réorganisation.

### Exemple

L'exemple suivant affiche la profondeur de l'élément spécifié dans le panneau Sortie :

```
// Select an object and run this script.  
fl.trace("Depth of selected object: " + fl.getDocumentDOM().selection[0].depth);
```

Voir l'exemple relatif à [element.elementType](#).

## element.elementType

### Disponibilité

Flash MX 2004.

### Utilisation

`element.elementType`

### Description

Propriété en lecture seule ; chaîne représentant le type de l'élément spécifié. Valeurs possibles : "shape", "text", "instance" ou "shapeObj". Un "shapeObj" peut être créé à l'aide d'un outil extensible.

### Exemple

L'exemple suivant stocke dans la variable `eType` le type du premier élément :

```
// In a new file, place a movie clip on first frame top layer, and
// then run this line of script.
var eType = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].elementType; //
eType = instance
```

L'exemple suivant affiche plusieurs propriétés pour tous les éléments de l'image ou du calque actif :

```
var tl = fl.getDocumentDOM().getTimeline()
var elts = tl.layers[tl.currentLayer].frames[tl.currentFrame].elements;
for (var x = 0; x < elts.length; x++) {
    var elt = elts[x];
    fl.trace("Element " + x + " Name = " + elt.name + " Type = " + elt.elementType + " location
= " + elt.left + ", " + elt.top + " Depth = " + elt.depth);
}
```

## element.getPersistentData()

### Disponibilité

Flash MX 2004.

### Utilisation

`element.getPersistentData(name)`

### Paramètres

**name** Chaîne identifiant les données à renvoyer.

### Valeur renvoyée

Les données spécifiées par le paramètre *name* ; ou 0 si ces données n'existent pas.

### Description

Méthode qui récupère la valeur des données spécifiées par le paramètre *name*. Le type de données dépend du type des données qui ont été stockées (voir [element.setPersistentData\(\)](#)). Seuls les symboles et les bitmaps acceptent des données persistantes.

### Exemple

L'exemple suivant définit et récupère les données pour l'élément spécifié, affiche ses valeurs dans le panneau Sortie, puis supprime les données :

```
// At least one symbol or bitmap is selected in the first layer, first frame.
var elt = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
elt.setPersistentData("myData","integer", 12);
if (elt.hasPersistentData("myData")){
fl.trace("myData = "+ elt.getPersistentData("myData"));
elt.removePersistentData( "myData" );
fl.trace("myData = "+ elt.getPersistentData("myData"));
}
```

## element.getTransformationPoint()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
element.getTransformationPoint()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un point (par exemple, {x:10,y:20}, sachant que x et y sont des nombres à virgule flottante) qui précise la position du point de transformation (également *point d'origine* ou *point zéro*) dans le système de coordonnées de l'élément.

### Description

Méthode qui récupère la valeur du point de transformation de l'élément spécifié.

les points de transformation sont définis par rapport à différents emplacements, en fonction du type d'élément sélectionné. Pour plus d'informations, voir [element.setTransformationPoint\(\)](#).

### Exemple

L'exemple suivant lit le point de transformation pour le troisième élément de la neuvième image du premier calque dans le document. La propriété `transPoint.x` donne la coordonnée x du point de transformation. La propriété `transPoint.y` donne la coordonnée y du point de transformation.

```
var transPoint =
fl.getDocumentDOM().getTimeline().layers[0].frames[8].elements[2].getTransformationPoint();
```

### Voir aussi

[document.getTransformationPoint\(\)](#), [element.setTransformationPoint\(\)](#), [element.transformX](#),  
[element.transformY](#)

## element.hasPersistentData()

### Disponibilité

Flash MX 2004.

### Utilisation

`element.hasPersistentData(name)`

### Paramètres

**name** Chaîne spécifiant le nom de l'élément de données à tester.

### Valeur renvoyée

Valeur booléenne : `true` si les données spécifiées sont jointes à l'objet, `false` dans le cas contraire.

### Description

Méthode qui détermine si les données spécifiées ont été jointes à l'élément spécifié. Seuls les symboles et les bitmaps acceptent des données persistantes.

### Exemple

Voir `element.getPersistentData()`.

## element.height

### Disponibilité

Flash MX 2004.

### Utilisation

`element.height`

### Description

Propriété ; nombre en virgule flottante spécifiant la hauteur de l'élément, en pixels.

N'utilisez pas cette propriété pour redimensionner un champ de texte. Mais sélectionnez le champ de texte et utilisez `document.setTextRectangle()`. En effet, l'utilisation de cette propriété avec un champ de texte met le texte à l'échelle.

### Exemple

L'exemple suivant fixe la hauteur de l'élément spécifié à 100 pixels :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].height = 100;
```

## element.layer

### Disponibilité

Flash 8.

### Utilisation

`element.layer`

### Description

Propriété en lecture seule ; représente l'objet [Objet Layer](#) sur lequel l'élément est situé.

### Exemple

L'exemple suivant stocke dans la variable `theLayer` l'objet Layer (calque) qui contient l'élément :

```
var theLayer = element.layer;
```

## element.left

### Disponibilité

Flash MX 2004.

### Utilisation

`element.left`

### Description

Propriété en lecture seule ; nombre en virgule flottante indiquant la coordonnée gauche de l'élément. Pour les éléments qui se trouvent dans une séquence, la valeur de `element.left` est calculée par rapport au coin supérieur gauche de la scène. Pour les éléments qui se trouvent dans un symbole, elle est exprimée par rapport au point d'alignement (également *point d'origine* ou *point zéro*) du symbole. Utilisez [document.setSelectionBounds\(\)](#) ou [document.moveSelectionBy\(\)](#) pour définir cette propriété.

### Exemple

L'exemple suivant illustre le changement de valeur de cette propriété lorsqu'un élément est déplacé :

```
// Select an element on the Stage and then run this script.
var sel = fl.getDocumentDOM().selection[0];
fl.trace("Left (before) = " + sel.left);
fl.getDocumentDOM().moveSelectionBy({x:100, y:0});
fl.trace("Left (after) = " + sel.left);
```

Voir l'exemple de [element.elementType](#).

## element.locked

### Disponibilité

Flash MX 2004.

### Utilisation

`element.locked`

### Description

Propriété : valeur booléenne, `true` si l'élément est verrouillé, `false` dans le cas contraire. Si la valeur de [element.elementType](#) correspond à "shape", cette propriété est ignorée.

### Exemple

L'exemple suivant verrouille le premier élément de la première image du calque supérieur :

```
// Similar to Modify > Arrange > Lock:  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].locked = true;
```

## element.matrix

### Disponibilité

Flash MX 2004.

### Utilisation

`element.matrix`

### Description

Propriété ; objet Matrix. Une matrice possède les propriétés `a`, `b`, `c`, `d`, `tx` et `ty`. Les propriétés `a`, `b`, `c` et `d` sont des nombres en virgule flottante. Les propriétés `tx` et `ty` sont des coordonnées. Voir [Objet Matrix](#).

### Exemple

L'exemple suivant déplace l'élément spécifié de 10 pixels sur l'axe *x*, et de 20 pixels sur l'axe *y* :

```
var mat = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].matrix;  
mat.tx += 10;  
mat.ty += 20;  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].matrix = mat;
```

## element.name

### Disponibilité

Flash MX 2004.

### Utilisation

`element.name`

### Description

Propriété ; chaîne spécifiant le nom de l'élément. Cette chaîne est généralement appelée « nom d'occurrence ». Si la valeur de `element.elementType` correspond à "shape", cette propriété est ignorée. Voir [element.elementType](#).

### Exemple

L'exemple suivant définit sur "clip\_mc" le nom d'occurrence du premier élément de l'image 1 au niveau du calque supérieur :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].name = "clip_mc";
```

Voir l'exemple de [element.elementType](#).

## element.removePersistentData()

### Disponibilité

Flash MX 2004.

### Utilisation

```
element.removePersistentData(name)
```

### Paramètres

**name** Chaîne spécifiant le nom des données à supprimer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime les données persistantes associées à l'objet et correspondant au nom spécifié. Seuls les symboles et les bitmaps acceptent des données persistantes.

### Exemple

Voir `element.getPersistentData()`.

## element.rotation

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
element.rotation
```

### Description

Propriété ; entier ou valeur en virgule flottante compris(e) entre -180 et 180 qui spécifie la rotation horaire de l'objet, en degrés.

### Exemple

L'exemple suivant définit sur 45 degrés la rotation de l'élément actuellement sélectionné :

```
var element = fl.getDocumentDOM().selection[0];  
fl.trace("Element rotation = " + element.rotation);  
element.rotation = 45;  
fl.trace("After setting rotation to 45: rotation = " + element.rotation);
```

## element.scaleX

### Disponibilité

Flash CS3 Professional.



### Utilisation

`element.scaleX`

### Description

Propriété ; valeur en virgule flottante qui précise la valeur d'échelle  $x$  des rectangles et ovals de type primitif, des symboles et des objets de dessin. La valeur 1 indique une échelle de 100 %.

### Exemple

L'exemple suivant définit la valeur d'échelle  $x$  de la sélection en cours sur 2 (double sa valeur) :

```
var element = fl.getDocumentDOM().selection[0];  
element.scaleX = 2;
```

### Voir aussi

[element.scaleY](#)

## element.scaleY

### Disponibilité

Flash CS3 Professional.

### Utilisation

`element.scaleY`

### Description

Propriété ; valeur en virgule flottante qui précise la valeur d'échelle  $y$  des rectangles et ovals de type primitif, des symboles et des objets de dessin. La valeur 1 indique une échelle de 100 %.

### Exemple

L'exemple suivant définit la valeur d'échelle  $y$  de la sélection en cours sur 2 (double sa valeur) :

```
var element = fl.getDocumentDOM().selection[0];  
element.scaleY = 2;
```

### Voir aussi

[element.scaleX](#)

## element.selected

### Disponibilité

Flash 8.

### Utilisation

`element.selected`

### Description

Propriété ; valeur booléenne indiquant si l'élément est sélectionné (`true`) ou non (`false`).

### Exemple

L'exemple suivant sélectionne l'élément :

```
element.selected = true;
```

## element.setPersistentData()

### Disponibilité

Flash MX 2004.

### Utilisation

```
element.setPersistentData(name, type, value)
```

### Paramètres

**name** Chaîne spécifiant le nom à associer aux données. Ce nom est utilisé pour récupérer les données.

**type** Chaîne spécifiant le type des données. Les valeurs possibles sont les suivantes : "integer", "integerArray", "double", "doubleArray", "string" et "byteArray".

**value** Spécifie la valeur à associer à l'objet. Le type de données de *value* dépend de la valeur du paramètre *type*. La valeur spécifiée doit être appropriée pour le type de données spécifié par le paramètre *type*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui stocke des données avec un élément. Ces données sont accessibles en JavaScript à la réouverture du fichier FLA contenant l'élément. Seuls les symboles et les bitmaps acceptent des données persistantes.

### Exemple

Voir `element.getPersistentData()`.

## element.setTransformationPoint()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
element.setTransformationPoint(transformationPoint)
```

### Paramètres

**transformationPoint** Point (par exemple, {x:10,y:20}, sachant que x et y sont des nombres à virgule flottante) spécifiant des valeurs pour un point de transformation d'un élément ou d'un groupe.

- Formes : *transformationPoint* est défini par rapport au document (0,0 est le coin supérieur gauche de la scène).
- Symboles : *transformationPoint* est défini par rapport au point d'alignement du symbole (0,0 est situé au point d'alignement).
- Texte : *transformationPoint* est défini par rapport au champ de texte (0,0 est le coin supérieur gauche du champ de texte).
- Bitmaps/vidéos : *transformationPoint* est défini par rapport à la bitmap/vidéo (0,0 est le coin supérieur gauche de l'image bitmap ou de la vidéo).
- Objets de dessin, objets primitifs et groupes : *transformationPoint* est défini par rapport au centre de l'élément ou du groupe (0,0 est le point central de l'élément ou du groupe).

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la position du point de transformation de l'élément.

Cette méthode est pratiquement identique à `document.setTransformationPoint()`, mais diffère cependant sur certains aspects :

- Le point de transformation des objets de dessin, des objets de type primitif et des groupes est calculé par rapport au centre de l'élément ou du groupe, et non par rapport à la scène.
- Il est possible de définir des points de transformation d'éléments sans les sélectionner au préalable.

Cette méthode déplace le point de transformation, mais ne déplace pas l'élément. À l'inverse, les propriétés `element.transformX` et `element.transformY` déplacent l'élément.

### Exemple

L'exemple suivant définit sur 100, 200 le point de transformation du troisième élément sur la scène :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[2].setTransformationPoint({x:100, y:200});
```

### Voir aussi

`document.setTransformationPoint().element.getTransformationPoint()`, `element.transformX`, `element.transformY`

## element.skewX

### Disponibilité

Flash CS3 Professional.

### Utilisation

`element.skewX`

### Description

Propriété ; valeur codée en virgule flottante, comprise entre -180 et 180, et précisant la valeur d'inclinaison *x* des rectangles et ovales de type primitif, des symboles et des objets de dessin.

### Exemple

L'exemple suivant définit la valeur d'inclinaison *x* de la sélection en cours sur 10 :

```
var element = fl.getDocumentDOM().selection[0];  
element.skewX = 10;
```

### Voir aussi

[document.setTransformationPoint\(\)](#), [element.skewY](#)

## element.skewY

### Disponibilité

Flash CS3 Professional.

### Utilisation

`element.skewY`

### Description

Propriété ; valeur en virgule flottante, comprise entre -180 et 180, et précisant la valeur d'inclinaison *y* des rectangles et ovales de type primitif, des symboles et des objets de dessin.

### Exemple

L'exemple suivant définit la valeur d'inclinaison *y* de la sélection en cours sur 10 :

```
var element = fl.getDocumentDOM().selection[0];  
element.skewY = 10;
```

### Voir aussi

[document.setTransformationPoint\(\)](#), [element.skewX](#)

## element.top

### Disponibilité

Flash MX 2004.

### Utilisation

`element.top`

### Description

Propriété en lecture seule ; coordonnée supérieure de l'élément. Pour les éléments qui se trouvent dans une séquence, la valeur de `element.top` est exprimée relativement au coin supérieur gauche de la scène. Pour les éléments qui se trouvent dans un symbole, elle est exprimée relativement au point d'alignement du symbole. Utilisez `document.setSelectionBounds()` ou `document.moveSelectionBy()` pour définir cette propriété.

### Exemple

L'exemple suivant illustre le changement de valeur de cette propriété lorsqu'un élément est déplacé :

```
// Select an element on the Stage and then run this script.  
var sel = fl.getDocumentDOM().selection[0];  
fl.trace("Top (before) = " + sel.top);  
fl.getDocumentDOM().moveSelectionBy({x:0, y:100});  
fl.trace("Top (after) = " + sel.top);
```

Voir l'exemple de `element.elementType`.

## element.transformX

### Disponibilité

Flash CS3 Professional.

### Utilisation

`element.transformX`

### Description

Propriété ; nombre à virgule flottante spécifiant la valeur *x* du point de transformation de l'élément sélectionné dans le système de coordonnées du parent de l'élément. La définition d'une nouvelle valeur pour cette propriété déplace l'élément. À l'inverse, la méthode `element.setTransformationPoint()` déplace le point de transformation, mais pas l'élément.

### Exemple

#### Voir aussi

`element.getTransformationPoint()`, `element.setTransformationPoint()`, `element.transformY`

## element.transformY

### Disponibilité

Flash CS3 Professional.

### Utilisation

`element.transformY`

### Description

Propriété ; nombre à virgule flottante spécifiant la valeur y du point de transformation de l'élément sélectionné dans le système de coordonnées du parent de l'élément. La définition d'une nouvelle valeur pour cette propriété déplace l'élément. A l'inverse, la méthode `element.setTransformationPoint()` déplace le point de transformation, mais pas l'élément.

### Voir aussi

`element.getTransformationPoint()`, `element.setTransformationPoint()`, `element.transformX`

## element.width

### Disponibilité

Flash MX 2004.

### Utilisation

`element.width`

### Description

Propriété ; nombre en virgule flottante spécifiant la largeur de l'élément, en pixels.

N'utilisez pas cette propriété pour redimensionner un champ de texte. Mais sélectionnez le champ de texte et utilisez `document.setTextRectangle()`. En effet, l'utilisation de cette propriété avec un champ de texte met le texte à l'échelle.

### Exemple

L'exemple suivant fixe la largeur de l'élément spécifié à 100 pixels :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].width= 100;
```

## element.x

### Disponibilité

Flash CS3 Professional.

### Utilisation

`element.x`

### Description

Propriété ; valeur en virgule flottante et précisant la valeur x du point d'alignement de l'élément sélectionné.

### Exemple

L'exemple suivant définit la valeur du point d'alignement de l'élément spécifié sur 100, 200 :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].x= 100;  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].y= 200;
```

**Voir aussi**

[element.y](#)

## element.y

**Disponibilité**

Flash CS3 Professional.

**Utilisation**

`element.y`

**Description**

Propriété ; valeur en virgule flottante précisant la valeur *y* du point d'alignement de l'élément sélectionné.

**Exemple**

Voir [element.x](#).

# Chapitre 15 : Objet Fill

## Disponibilité

Flash MX 2004.

## Description

L'objet Fill contient toutes les propriétés du paramètre de couleur de remplissage (Fill color) du panneau Outils ou de la forme sélectionnée. Pour récupérer un objet Fill, utilisez `document.getCustomFill()`.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Fill :

Propriété	Description
<code>fill.bitmapIsClipped</code>	Valeur booléenne qui spécifie si le remplissage bitmap d'une forme plus grande que le bitmap est recadré ou répété.
<code>fill.bitmapPath</code>	Chaîne qui spécifie le chemin et le nom du remplissage bitmap dans la bibliothèque.
<code>fill.color</code>	Chaîne, valeur hexadécimale ou entière représentant la couleur du remplissage.
<code>fill.colorArray</code>	Tableau de couleurs en dégradé.
<code>fill.focalPoint</code>	Entier spécifiant le décalage horizontal du point focal du dégradé par rapport au point de transformation.
<code>fill.linearRGB</code>	Valeur booléenne qui indique si le remplissage doit être effectué sous forme de dégradé RVB linéaire ou radial.
<code>fill.matrix</code>	<a href="#">Objet Matrix</a> qui définit l'emplacement, l'orientation et les échelles des remplissages avec dégradé.
<code>fill.overflow</code>	Chaîne qui indique le comportement d'un débordement de dégradé.
<code>fill.posArray</code>	Tableau de nombres entiers, compris entre zéro et 255, indiquant la position de la couleur correspondante.
<code>fill.style</code>	Chaîne qui spécifie le style du remplissage.

## fill.bitmapsClipped

## Disponibilité

Flash CS4 Professional.

## Utilisation

`fill.bitmapIsClipped`

## Description

Propriété ; valeur booléenne qui spécifie si le remplissage bitmap d'une forme plus grande que le bitmap est recadré (`true`) ou répété (`false`). Cette propriété n'est disponible que si la valeur de la propriété `fill.style` est "bitmap". Si la forme est plus petite que le bitmap, cette valeur est `false`.



### Exemple

L'exemple suivant affiche des informations sur le recadrage du remplissage du bitmap, le cas échéant, dans le panneau Sortie :

```
var fill = fl.getDocumentDOM().getCustomFill();  
if (fill.style == "bitmap")  
    fl.trace("Fill image is clipped: " + fill.bitmapIsClipped);
```

### Voir aussi

[fill.bitmapPath](#)

## fill.bitmapPath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fill.bitmapPath`

### Description

Propriété ; chaîne qui spécifie le chemin et le nom du remplissage bitmap dans la bibliothèque. Cette propriété n'est disponible que si la valeur de la propriété `fill.style` est "bitmap".

### Exemple

L'exemple suivant définit le style du remplissage de l'élément spécifié sur une image bitmap de la bibliothèque :

```
var fill = fl.getDocumentDOM().getCustomFill();  
fill.style = "bitmap";  
fill.bitmapPath = "myBitmap.jpg";  
fl.getDocumentDOM().setCustomFill(fill);
```

### Voir aussi

[fill.bitmapIsClipped](#)

## fill.color

### Disponibilité

Flash MX 2004.

### Utilisation

`fill.color`

### Description

Propriété ; couleur du remplissage, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB

- Entier représentant l'équivalent décimal d'un nombre hexadécimal

### Exemple

L'exemple suivant définit la couleur de remplissage de la sélection :

```
var fill = fl.getDocumentDOM().getCustomFill();  
fill.color = "#FFFFFF";  
fl.getDocumentDOM().setCustomFill( fill );
```

## fill.colorArray

### Disponibilité

Flash MX 2004.

### Utilisation

fill.colorArray

### Description

Propriété ; tableau de couleurs du dégradé, exprimées sous forme d'entiers. Cette propriété n'est disponible que si la valeur de la propriété fill.style est "radialGradient" ou "linearGradient". Voir [fill.style](#)

### Exemple

L'exemple suivant affiche le tableau des couleurs de la sélection dans le panneau Sortie :

```
var fill = fl.getDocumentDOM().getCustomFill();  
if(fill.style == "linearGradient" || fill.style == "radialGradient")  
    alert(fill.colorArray);
```

L'exemple suivant définit le remplissage sur le dégradé linéaire spécifié :

```
var fill = fl.getDocumentDOM().getCustomFill();  
fill.style = "linearGradient";  
fill.colorArray = ["#00ff00", "#ff00ff"];  
fill.posArray = [0, 255];  
fl.getDocumentDOM().setCustomFill(fill);
```

## fill.focalPoint

### Disponibilité

Flash 8.

### Utilisation

fill.focalPoint

### Description

Propriété ; entier spécifiant le décalage horizontal du point focal du dégradé par rapport au point de transformation. Par exemple, une valeur de 10 place le point focal à 10/255 de la distance entre le point de transformation et le bord du dégradé. Une valeur de -255 place le point focal sur le bord gauche du dégradé. La valeur par défaut est 0.

Cette propriété n'est disponible que si la valeur de la propriété `fill.style` est "radialGradient".

### Exemple

L'exemple suivant définit, pour la sélection en cours, le point focal d'un dégradé radial sur 100 pixels à droite du centre de la forme.

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "radialGradient";
fill.colorArray = ["#00ff00", "#ff00ff"];
fill.posArray = [0, 255];
fill.focalPoint = 10100;
fl.getDocumentDOM().setCustomFill(fill);
```

## fill.linearRGB

### Disponibilité

Flash 8.

### Utilisation

`fill.linearRGB`

### Description

Propriété ; valeur booléenne qui indique si le remplissage doit être effectué sous forme de dégradé RVB linéaire ou radial. Cette propriété doit être définie sur `true` pour obtenir une interpolation linéaire du dégradé, sur `false` pour obtenir une interpolation radiale. La valeur par défaut est `false`.

### Exemple

L'exemple suivant spécifie un rendu par interpolation RVB linéaire du dégradé de la sélection en cours :

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.linearRGB style = true"radialGradient";
fill.colorArray = ["#00ff00", "#ff00ff"];
fill.posArray = [0, 255];
fill.focalPoint = 100;
fill.linearRGB = true;
fl.getDocumentDOM().setCustomFill(fill);
```

## fill.matrix

### Disponibilité

Flash MX 2004.

### Utilisation

`fill.matrix`

### Description

Propriété ; [Objet Matrix](#) qui définit l'emplacement, l'orientation et les échelles des remplissages avec dégradés.

### Exemple

L'exemple suivant utilise la propriété `fill.matrix` pour appliquer un remplissage dégradé sur la sélection en cours :

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = 'radialGradient';
fill.colorArray = ['#00ff00', '#ff00ff'];
fill.posArray = [0, 255];
fill.focalPoint = 100;
fill.linearRGB = false;
fill.overflow = 'repeat';
var mat = fl.getDocumentDOM().selection[0].matrix;
mat.a = 0.0167083740234375;
mat.b = -0.0096435546875;
mat.c = 0.0312957763671875;
mat.d = 0.05419921875;
mat.tx = 288.65;
mat.ty = 193.05;
for (i in mat) {
    fl.trace(i+' : '+mat[i]);
}
fl.getDocumentDOM().setCustomFill(fill);
```

## fill.overflow

### Disponibilité

Flash 8.

### Utilisation

`fill.overflow`

### Description

Propriété ; chaîne qui indique le comportement du débordement d'un dégradé. Les valeurs possibles sont "extend" (extension), "repeat" (répétition) et "reflect" (réflexion) ; ces trois chaînes ne sont pas sensibles à la casse. La valeur par défaut est "extend".

### Exemple

L'exemple suivant spécifie un débordement de type "extend" pour la sélection en cours :

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.overflow = "extend";
fl.getDocumentDOM().setCustomFill(fill);
```

## fill.posArray

### Disponibilité

Flash MX 2004.

### Utilisation

`fill.posArray`

### Description

Propriété ; tableau de nombres entiers, compris entre zéro et 255, indiquant la position de la couleur correspondante. Cette propriété n'est disponible que si la valeur de la propriété `fill.style` est "radialGradient" ou "linearGradient".

### Exemple

L'exemple suivant spécifie les couleurs à utiliser pour un dégradé linéaire dans la sélection actuelle :

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray= [0,100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

## fill.style

### Disponibilité

Flash MX 2004. Valeur « bitmap » ajoutée dans Flash CS4 Professional.

### Utilisation

`fill.style`

### Description

Propriété ; chaîne qui spécifie le style du remplissage. Les valeurs possibles sont "bitmap", "solid" (uni), "linearGradient" (dégradé linéaire), "radialGradient" (dégradé radial) et "noFill" (pas de remplissage).

Si cette valeur est "linearGradient" ou "radialGradient", les propriétés `fill.colorArray` et `fill.posArray` sont également disponibles. Si cette valeur est "bitmap", les propriétés `fill.bitmapIsClipped` et `fill.bitmapPath` sont également disponibles.

### Exemple

L'exemple suivant spécifie les couleurs à utiliser pour un dégradé linéaire dans la sélection actuelle :

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style= "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray= [0,100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

# Chapitre 16 : Objet Filter

## Disponibilité

Flash 8.

## Description

L'objet `Filter` contient toutes les propriétés de tous les filtres. La propriété `filter.name` spécifie le type de filtre et détermine les propriétés qui s'appliquent à chaque filtre. Voir [filter.name](#).

Pour renvoyer la liste de filtres associée à un ou plusieurs objets, utilisez `document.getFilters()`. Pour appliquer des filtres à un ou plusieurs objets, utilisez `document.setFilters()`. Voir [document.getFilters\(\)](#) et [document.setFilters\(\)](#).

## Résumé des propriétés

L'objet `Filter` gère les propriétés suivantes :

Propriété	Description
<a href="#">filter.angle</a>	Nombre en virgule flottante spécifiant l'angle de l'ombre ou de la couleur de surbrillance, exprimé en degrés.
<a href="#">filter.blurX</a>	Valeur en virgule flottante et spécifiant la quantité de flou dans la direction x, exprimée en pixels.
<a href="#">filter.blurY</a>	Valeur en virgule flottante et spécifiant la quantité de flou dans la direction y, exprimée en pixels.
<a href="#">filter.brightness</a>	Nombre en virgule flottante spécifiant la luminosité du filtre.
<a href="#">filter.color</a>	Chaîne, valeur hexadécimale ou entier représentant la couleur du filtre.
<a href="#">filter.contrast</a>	Nombre en virgule flottante spécifiant la valeur de contraste du filtre.
<a href="#">filter.distance</a>	Nombre en virgule flottante spécifiant la distance, en pixels, entre l'effet du filtre et un objet.
<a href="#">filter.enabled</a>	Valeur booléenne indiquant si le filtre spécifié est activé ou non.
<a href="#">filter.hideObject</a>	Valeur booléenne indiquant si l'image source est masquée ou non.
<a href="#">filter.highlightColor</a>	Chaîne, valeur hexadécimale ou entier représentant la couleur de surbrillance.
<a href="#">filter.hue</a>	Nombre en virgule flottante spécifiant la teinte du filtre.
<a href="#">filter.inner</a>	Valeur booléenne indiquant si l'ombre est une ombre interne.
<a href="#">filter.knockout</a>	Valeur booléenne indiquant si le filtre est un filtre masque.
<a href="#">filter.name</a>	Lecture seule ; chaîne indiquant le type de filtre.
<a href="#">filter.quality</a>	Chaîne spécifiant la qualité du flou.
<a href="#">filter.saturation</a>	Nombre en virgule flottante spécifiant la valeur de saturation du filtre.
<a href="#">filter.shadowColor</a>	Chaîne, valeur hexadécimale ou entier représentant la couleur de l'ombre.
<a href="#">filter.strength</a>	Entier spécifiant l'intensité du filtre en pourcentage.
<a href="#">filter.type</a>	Chaîne spécifiant le type de biseau ou de rayonnement.

## filter.angle

### Disponibilité

Flash 8.

### Utilisation

`filter.angle`

### Description

Propriété ; nombre en virgule flottante spécifiant l'angle de l'ombre ou de la couleur de surbrillance, exprimé en degrés. Les valeurs gérées sont comprises entre 0 et 360. Cette propriété est définie pour les objets Filter avec une valeur de "bevelFilter", "dropShadowFilter", "gradientBevelFilter", ou "gradientGlowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit un angle de 120 pour les filtres Biseau sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++) {
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].angle = 120;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

`document.setFilterProperty()`

## filter.blurX

### Disponibilité

Flash 8.

### Utilisation

`filter.blurX`

### Description

Propriété ; valeur en virgule flottante et spécifiant la quantité de flou dans la direction x, exprimée en pixels. Les valeurs gérées sont comprises entre 0 et 255. Cette propriété est définie pour des objets Filter avec une valeur de "bevelFilter", "blurFilter", "dropShadowFilter", "glowFilter", "gradientBevelFilter" ou "gradientGlowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la valeur `blurX` sur 30 et la valeur `blurY` sur 20 pour les filtres Flou sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'blurFilter'){
        myFilters[i].blurX = 30;
        myFilters[i].blurY = 20;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

#### Voir aussi

[document.setFilterProperty\(\)](#), [filter.blurY](#)

## filter.blurY

#### Disponibilité

Flash 8.

#### Utilisation

`filter.blurY`

#### Description

Propriété ; valeur en virgule flottante et spécifiant la quantité de flou dans la direction y, exprimée en pixels. Les valeurs gérées sont comprises entre 0 et 255. Cette propriété est définie pour des objets Filter avec une valeur de "bevelFilter", "blurFilter", "dropShadowFilter", "glowFilter", "gradientBevelFilter" ou "gradientGlowFilter" pour la propriété [filter.name](#).

#### Exemple

Voir [filter.blurX](#).

#### Voir aussi

[document.setFilterProperty\(\)](#), [filter.blurX](#)

## filter.brightness

#### Disponibilité

Flash 8.

#### Utilisation

`filter.brightness`

#### Description

Propriété ; nombre en virgule flottante spécifiant la luminosité du filtre. Les valeurs gérées sont comprises entre -100 et 100. Cette propriété est définie pour des objets Filter avec une valeur de "adjustColorFilter" pour la propriété [filter.name](#).



### Exemple

L'exemple suivant définit une luminosité de 30,5 pour les filtres de réglage de la couleur sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].brightness = 30.5;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

## filter.color

### Disponibilité

Flash 8.

### Utilisation

`filter.color`

### Description

Propriété ; couleur du filtre, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

Cette propriété est définie pour des objets Filter avec une valeur de "dropShadowFilter" ou "glowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la couleur sur "#ff00003e" pour les filtres d'ombre portée sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'dropShadowFilter'){
        myFilters[i].color = '#ff00003e';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

`document.setFilterProperty()`

## filter.contrast

### Disponibilité

Flash 8.

### Utilisation

`filter.contrast`

### Description

Propriété ; nombre en virgule flottante spécifiant la valeur de contraste du filtre. Les valeurs gérées sont comprises entre -100 et 100. Cette propriété est définie pour des objets Filter avec une valeur de "adjustColorFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit un contraste de -15,5 pour les filtres de réglage de la couleur sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].contrast = -15.5;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

## filter.distance

### Disponibilité

Flash 8.

### Utilisation

`filter.distance`

### Description

Propriété ; nombre en virgule flottante spécifiant la distance, en pixels, entre l'effet du filtre et un objet. Les valeurs acceptées sont comprises entre -255 et 255. Cette propriété est définie pour les objets Filter avec une valeur de "bevelFilter", "dropShadowFilter", "gradientBevelFilter", ou "gradientGlowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la distance sur 10 pixels pour les filtres d'ombre portée sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'dropShadowFilter'){
        myFilters[i].distance = 10;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

`document.setFilterProperty()`

## filter.enabled

### Disponibilité

Flash CS3 Professional.

### Utilisation

`filter.enabled`

### Description

Propriété; valeur booléenne indiquant si le filtre spécifié est activé (`true`) ou non (`false`).

### Exemple

L'exemple suivant désactive les filtres de couleurs appliqués aux objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].enabled = false;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

## filter.hideObject

### Disponibilité

Flash 8.

### Utilisation

`filter.hideObject`

### Description

Propriété ; valeur booléenne indiquant si l'image source est masquée (`true`) ou affichée (`false`). Cette propriété est définie pour des objets Filter avec une valeur de "dropShadowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la valeur `hideObject` sur `true` pour les filtres d'ombre portée sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'dropShadowFilter'){
        myFilters[i].hideObject = true;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

## filter.highlightColor

### Disponibilité

Flash 8.

### Utilisation

`filter.highlightColor`

### Description

Propriété ; couleur de surbrillance, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

Cette propriété est définie pour des objets Filter avec une valeur de "bevelFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la couleur de surbrillance sur "#ff00003e" pour les filtres de biseau sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].highlightColor = '#ff00003e';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

## filter.hue

### Disponibilité

Flash 8.

### Utilisation

`filter.hue`

### Description

Propriété ; nombre en virgule flottante spécifiant la teinte du filtre. Les valeurs gérées sont comprises entre -180 et 180. Cette propriété est définie pour des objets Filter avec une valeur de "adjustColorFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit une teinte de 120 pour les filtres de réglage de la couleur sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].hue = 120;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

## filter.inner

### Disponibilité

Flash 8.

### Utilisation

`filter.inner`

### Description

Propriété ; valeur booléenne indiquant si l'ombre est intérieure (`true`) ou non (`false`). Cette propriété est définie pour des objets Filter avec une valeur de "dropShadowFilter" ou "glowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la valeur de la propriété `inner` sur `true` pour les filtres de rayonnement sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].inner = true;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

[document.setFilterProperty\(\)](#)

## filter.knockout

### Disponibilité

Flash 8.

### Utilisation

`filter.knockout`

### Description

Propriété ; valeur booléenne indiquant si le filtre masque (`true`) ou non (`false`) l'image source. Cette propriété est définie pour les objets Filter avec une valeur de "bevelFilter", "dropShadowFilter", "glowFilter", "gradientBevelFilter", ou "gradientGlowFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la propriété knockout sur true pour les filtres de rayonnement sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].knockout = true;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

[document.setFilterProperty\(\)](#)

## filter.name

### Disponibilité

Flash 8.

### Utilisation

filter.name

### Description

Propriété en lecture seule ; chaîne indiquant le type de filtre. La valeur de cette propriété identifie les autres propriétés disponibles pour l'objet Filter. La valeur est l'une des suivantes : "adjustColorFilter", "bevelFilter", "blurFilter", "dropShadowFilter", "glowFilter", "gradientBevelFilter" ou "gradientGlowFilter".

### Exemple

L'exemple suivant affiche le nom des filtres et les positions d'index dans le panneau Sortie :

```
var myFilters = fl.getDocumentDOM().getFilters();
var traceStr = "";
for(i=0; i < myFilters.length; i++){
    traceStr = traceStr + " At index " + i + ": " + myFilters[i].name;
}
fl.trace(traceStr);
```

### Voir aussi

[document.getFilters\(\)](#), [document.setFilterProperty\(\)](#)

## filter.quality

### Disponibilité

Flash 8.

### Utilisation

filter.quality

### Description

Propriété ; chaîne spécifiant la qualité du flou. Les valeurs gérées sont "low", "medium" et "high" ("high" est similaire à un flou gaussien). Cette propriété est définie pour les objets Filter avec une valeur de "bevelFilter", "blurFilter", "dropShadowFilter", "glowFilter", "gradientGlowFilter" ou "gradientBevelFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la qualité de flou sur "medium" pour les filtres de flou sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].quality = 'medium';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

`document.setFilterProperty()`

## filter.saturation

### Disponibilité

Flash 8.

### Utilisation

`filter.saturation`

### Description

Propriété ; nombre en virgule flottante spécifiant la valeur de saturation du filtre. Les valeurs acceptées sont comprises entre -100 et 100. Cette propriété est définie pour des objets Filter avec une valeur de "adjustColorFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la saturation sur 100 (niveaux de gris) pour les filtres de réglage de la couleur sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].saturation = 0-100;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

`document.setFilterProperty()`

## filter.shadowColor

### Disponibilité

Flash 8.

### Utilisation

`filter.shadowColor`

### Description

Propriété ; couleur de l'ombre, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVBBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

Cette propriété est définie pour des objets Filter avec une valeur de "bevelFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit la couleur de l'ombre sur "#ff00003e" pour les filtres de biseau sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].shadowColor = '#ff00003e';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

### Voir aussi

`document.setFilterProperty()`

## filter.strength

### Disponibilité

Flash 8.

### Utilisation

`filter.strength`

### Description

Propriété ; entier spécifiant l'intensité du filtre en pourcentage. Les valeurs gérées sont comprises entre 0 et 25 500.

Cette propriété est définie pour les objets Filter avec une valeur de "bevelFilter", "dropShadowFilter", "glowFilter", "gradientGlowFilter" ou "gradientBevelFilter" pour la propriété `filter.name`.

### Exemple

L'exemple suivant définit l'intensité sur 50 pour les filtres de rayonnement sur les objets sélectionnés :



```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].strength = 50;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

#### Voir aussi

[document.setFilterProperty\(\)](#)

## filter.type

#### Disponibilité

Flash 8.

#### Utilisation

`filter.type`

#### Description

Propriété ; chaîne spécifiant le type de filtre de biseau ou de rayonnement. Les valeurs gérées sont "inner", "outer" et "full". Cette propriété est définie pour les objets Filter avec une valeur de "bevelFilter", "gradientGlowFilter" ou "gradientBevelFilter" pour la propriété [filter.name](#).

#### Exemple

L'exemple suivant définit sur "full" le type de tous les filtres de biseau sur les objets sélectionnés :

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].type = 'full';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

#### Voir aussi

[document.setFilterProperty\(\)](#)

# Chapitre 17 : Objet flash (fl)

## Disponibilité

Flash MX 2004.

## Description

L'objet flash représente l'application Flash. Pour vous référer à cet objet, utilisez le nom `flash` ou `fl`. Cette documentation utilise systématiquement le terme `fl` dans les exemples de code.

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet flash :

Méthode	Description
<code>fl.addEventListener()</code>	Enregistre une fonction à appeler lors de la réception d'un événement spécifique.
<code>fl.browseForFileURL()</code>	Ouvre une boîte de dialogue Fichier Ouvrir ou Fichier Enregistrer pour permettre à l'utilisateur d'indiquer un nom de fichier à ouvrir ou à enregistrer.
<code>fl.browseForFolderURL()</code>	Affiche la boîte de dialogue Rechercher un dossier et permet à l'utilisateur de sélectionner un dossier.
« <code>fl.clearPublishCache()</code> » à la page 226	Efface le cache de publication.
<code>fl.clipCopyString()</code>	Copie la chaîne spécifiée dans le Presse-papiers.
<code>fl.closeAll()</code>	Ferme tous les documents ouverts et affiche la boîte de dialogue Enregistrer sous pour les documents qui n'ont pas encore été enregistrés.
<code>fl.closeAllPlayerDocuments()</code>	Ferme tous les fichiers SWF qui ont été ouverts par la commande Contrôle > Tester l'animation.
<code>fl.closeDocument()</code>	Ferme le document spécifié.
<code>fl.createDocument()</code>	Ouvre un nouveau document et le sélectionne.
« <code>fl.exportPublishProfileString()</code> » à la page 234	URI (Uniform Resource Identifier) à partir duquel exporter les paramètres de publication.
<code>fl.fileExists()</code>	Vérifie si un fichier existe déjà sur le disque.
<code>fl.findDocumentDOM()</code>	Permet de cibler un fichier spécifique au moyen de son identificateur unique.
<code>fl.findDocumentIndex()</code>	Renvoie un tableau d'entiers représentant la position d'un document dans le tableau <code>fl.documents</code> .
<code>fl.findObjectInDocByName()</code>	Expose les éléments dont les noms d'occurrence correspondent à un texte particulier.
<code>fl.findObjectInDocByType()</code>	Expose des éléments d'un type spécifié dans un document.
<code>fl.getAppMemoryInfo()</code>	Renvoie un entier représentant le nombre d'octets occupés dans la zone mémoire de Flash.exe spécifiée.
<code>fl.getDocumentDOM()</code>	Récupère le DOM ( <a href="#">Objet Document</a> ) du document actuellement actif.

Méthode	Description
<code>fl.getSwfPanel()</code>	Renvoie l'objet SWFPanel en fonction du nom localisé du panneau ou de son nom de fichier SWF.
<code>fl.isFontInstalled()</code>	Détermine si la police spécifiée est installée.
<code>fl.mapPlayerURL()</code>	Convertit une adresse URL avec séquences d'échappement Unicode en une URL en UTF-8 ou MBCS.
<code>fl.openDocument()</code>	Ouvre dans une nouvelle fenêtre de document un document Flash (FLA) à modifier, et active cette fenêtre.
<code>fl.openScript()</code>	Ouvre un script (JSFL, AS, ASC) ou tout autre fichier (XML, TXT) dans l'éditeur de texte de Flash.
<code>fl.quit()</code>	Ferme Flash en proposant à l'utilisateur d'enregistrer tous les documents modifiés.
<code>fl.reloadTools()</code>	Reconstruit le panneau Outils à partir du contenu du fichier toolconfig.xml. Cette méthode n'est utilisée que lors de la création d'outils extensibles.
<code>fl.removeEventListener()</code>	Annule l'inscription d'une fonction enregistrée à l'aide de <code>fl.addEventListener()</code> .
<code>fl.resetAS3PackagePaths()</code>	Réinitialise le paramètre de chemin de classe global sur la valeur par défaut dans la boîte de dialogue des paramètres d'ActionScript 3.0.
<code>fl.resetPackagePaths()</code>	Réinitialise le paramètre de chemin de classe global sur la valeur par défaut dans la boîte de dialogue des paramètres d'ActionScript 2.0.
<code>fl.runScript()</code>	Exécute un fichier JavaScript.
<code>fl.saveAll()</code>	Enregistre tous les documents ouverts et affiche la boîte de dialogue Enregistrer sous pour ceux qui n'ont pas encore été enregistrés.
<code>fl.saveDocument()</code>	Enregistre le document spécifié sous forme de fichier FLA.
<code>fl.saveDocumentAs()</code>	Affiche la boîte de dialogue Enregistrer sous pour le document spécifié.
<code>fl.selectElement()</code>	Active l'option de sélection ou d'édition d'un élément.
<code>fl.selectTool()</code>	Sélectionne l'outil spécifié dans le panneau Outils.
<code>fl.setActiveWindow()</code>	Définit le document spécifié comme fenêtre active.
<code>fl.showIdleMessage()</code>	Permet de désactiver l'avertissement relatif à un script dont l'exécution est trop longue.
<code>fl.toggleBreakpoint()</code>	Active/désactive un point d'arrêt pour le fichier .as donné à la ligne donnée.
<code>fl.trace()</code>	Envoie une chaîne de texte dans le panneau Sortie.

### Résumé des propriétés

Vous pouvez utiliser les propriétés suivantes avec l'objet flash :

Propriété	Description
<code>fl.actionsPanel</code>	Lecture seule ; <a href="#">Objet actionsPanel</a> .
<code>fl.as3PackagePaths</code>	Chaîne correspondant au paramètre de chemin de classe global dans la boîte de dialogue des paramètres d'ActionScript 3.0.
<code>fl.compilerErrors</code>	Lecture seule ; <a href="#">Objet compilerErrors</a> .
<code>fl.componentsPanel</code>	Lecture seule ; <a href="#">Objet componentsPanel</a> représentant le panneau Composants.
<code>fl.configDirectory</code>	Lecture seule ; chaîne indiquant le chemin d'accès complet du dossier Configuration de l'utilisateur local. Ce chemin est spécifique à chaque plateforme.
<code>fl.configURI</code>	Lecture seule ; chaîne indiquant le chemin d'accès complet du dossier Configuration de l'utilisateur local sous la forme « fichier:///URI ».
<code>fl.contactSensitiveSelection</code>	Valeur booléenne spécifiant si le mode de sélection Activer au contact est activé ou non.
<code>fl.createNewDocList</code>	Lecture seule ; tableau de chaînes représentant les divers types de documents qu'il est possible de créer.
<code>fl.createNewDocListType</code>	Lecture seule ; tableau de chaînes représentant les extensions de fichier des divers types de documents qu'il est possible de créer.
<code>fl.createNewTemplateList</code>	Lecture seule ; tableau de chaînes représentant les divers types de modèles qu'il est possible de créer.
<code>fl.documents</code>	Lecture seule ; tableau d'objets Document (voir <a href="#">Objet Document</a> ) représentant les documents (fichiers FLA) actuellement ouverts pour modification.
<code>fl.drawingLayer</code>	Lecture seule ; <a href="#">Objet drawingLayer</a> que doit utiliser un outil extensible lorsque l'utilisateur dessine un tracé temporaire à l'aide de sa souris.
<code>fl.externalLibraryPath</code>	Chaîne contenant la liste des éléments dans le chemin de bibliothèque externe ActionScript 3.0 global, spécifiant l'emplacement des fichiers SWC utilisés en tant que bibliothèques partagées au moment de l'exécution.
<code>fl.flexSDKPath</code>	Chaîne spécifiant le chemin du dossier du kit de développement Flex, qui contient la corbeille, les structures, les bibliothèques et d'autres dossiers.
<code>fl.installedPlayers</code>	Renvoie un tableau d'objets génériques correspondant à la liste des lecteurs Flash installés dans l'inspecteur des propriétés du document.
<code>fl.languageCode</code>	Renvoie le code à cinq caractères qui identifie les paramètres régionaux de l'interface utilisateur de l'application.
<code>fl.libraryPath</code>	Chaîne qui contient la liste des éléments du chemin de bibliothèque ActionScript 3.0 global, spécifiant l'emplacement des fichiers SWC ou des dossiers contenant des fichiers SWC.
<code>fl.Math</code>	Lecture seule ; <a href="#">Objet Math</a> qui fournit les méthodes nécessaires pour les opérations de matrice et de point.
<code>fl.mruRecentFileList</code>	Lecture seule ; tableau des noms de fichier complets de la liste des fichiers récemment utilisés (MRU, Most Recently Used) gérée par l'outil de création de Flash.
<code>fl.mruRecentFileListType</code>	Lecture seule ; tableau des types de fichiers de la liste des fichiers récemment utilisés (MRU, Most Recently Used) gérée par l'outil de création de Flash.
<code>fl.packagePaths</code>	Chaîne correspondant au paramètre de chemin de classe global dans la boîte de dialogue des paramètres d'ActionScript 2.0.
<code>fl.publishCacheDiskSizeMax</code>	Entier qui définit les préférences de limite de taille du cache disque.

Propriété	Description
<a href="#">fl.publishCacheEnabled</a>	Valeur booléenne qui indique si le cache de publication est activé.
<a href="#">fl.publishCacheMemoryEntrySizeLimit</a>	Entier qui définit la taille maximale de l'entrée du cache mémoire.
<a href="#">fl.publishCacheMemorySizeMax</a>	Entier qui définit les préférences de limite de taille du cache mémoire.
<a href="#">fl.objectDrawingMode</a>	Entier représentant le mode de dessin d'objet qui est activé.
<a href="#">fl.outputPanel</a>	Lecture seule ; référence à l' <a href="#">Objet outputPanel</a> .
<a href="#">fl.presetPanel</a>	Lecture seule ; <a href="#">Objet presetPanel</a> .
<a href="#">fl.scriptURI</a>	Lecture seule ; chaîne représentant le chemin d'accès du script JSFL actuellement en cours d'exécution et exprimée sous la forme « fichier:/// URI ».
<a href="#">fl.sourcePath</a>	Chaîne qui contient la liste des éléments du chemin source ActionScript 3.0 global, spécifiant l'emplacement des fichiers de classes ActionScript.
<a href="#">fl.swfPanels</a>	Tableau des objets swfPanel enregistrés (voir <a href="#">Objet swfPanel</a> ).
<a href="#">fl.tools</a>	Lecture seule ; tableau des objets Tools.
<a href="#">fl.version</a>	Lecture seule ; chaîne longue de la version de l'outil de création de Flash, plateforme comprise.
<a href="#">fl.xmlui</a>	Lecture seule ; <a href="#">Objet XMLUI</a> .

## fl.actionsPanel

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.actionsPanel
```

### Description

Propriété en lecture seule ; objet actionsPanel représentant le panneau Action actuellement affiché. Pour plus d'informations sur l'utilisation de cette propriété, voir la section [Objet actionsPanel](#).

## fl.addEventListener()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.addEventListener(eventType, callbackFunction)
```

### Paramètres

**eventType** Chaîne spécifiant le type d'événement à transmettre à cette fonction de rappel. Les valeurs possibles sont : "documentNew", "documentOpened", "documentClosed", "mouseMove", "documentChanged", "layerChanged" et "frameChanged".

La valeur `documentChanged` ne signifie pas que le contenu d'un document a été modifié, mais qu'un autre document se trouve à présent au premier plan. Ainsi, `fl.getDocumentDOM()` renverra une valeur différente de celle qui a été envoyée avant que cet événement se produise.

**callbackFunction** Nom de la fonction que vous souhaitez exécuter chaque fois que l'événement se produit.

### Valeur renvoyée

Aucune.

### Description

Méthode qui enregistre une fonction à appeler quand un événement particulier se produit.

Si vous décidez d'utiliser cette méthode, sachez que lorsque l'événement se produit fréquemment (comme cela peut être le cas avec `mouseMove`) et que la fonction demande un certain temps pour s'exécuter, votre application peut se bloquer ou éventuellement devenir instable et générer des erreurs.

### Exemple

L'exemple suivant affiche un message dans le panneau Sortie à la fermeture d'un document :

```
myFunction = function () {  
    fl.trace('document was closed'); }  
fl.addEventListener("documentClosed", myFunction);
```

### Voir aussi

[fl.removeEventListener\(\)](#)

## fl.as3PackagePaths

### Disponibilité

Flash CS3 Professional.

### Utilisation

`fl.as3PackagePaths`

### Description

Propriété ; chaîne correspondant au paramètre de chemin de classe global dans la boîte de dialogue des paramètres d'ActionScript 3.0. Les éléments de la chaîne sont délimités par des points-virgules. Pour afficher ou modifier des paramètres de chemin de classe d'ActionScript 2.0, utilisez [fl.packagePaths](#).

### Exemple

L'exemple suivant montre comment modifier les paramètres de chemin de classe d'ActionScript 3.0:

```
fl.trace(fl.as3PackagePaths);  
// Output (assuming started with default value)  
// .;$ (AppConfig)/ActionScript 3.0/Classes  
fl.as3PackagePaths="buying;selling";  
fl.trace(fl.as3PackagePaths);  
// Output  
// buying; selling
```

#### Voir aussi

[fl.resetAS3PackagePaths\(\)](#)

## fl.browseForFileURL()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
fl.browseForFileURL(browseType [, title [, previewArea]])
```

#### Paramètres

**browseType** Chaîne spécifiant le type d'opération à effectuer sur les fichiers. Les valeurs possibles sont "open", "select" ou "save". Les valeurs "open" et "select" ouvrent la boîte de dialogue d'ouverture de fichier. Chaque valeur est prévue pour être compatible avec Dreamweaver. La valeur "save" ouvre la boîte de dialogue d'enregistrement de fichier.

**title** Chaîne indiquant le titre de la boîte de dialogue d'ouverture ou d'enregistrement de fichier. Si ce paramètre est omis, une chaîne par défaut est utilisée. Ce paramètre est facultatif.

**previewArea** Paramètre facultatif qui est ignoré par Flash et Fireworks, et uniquement destiné à la compatibilité avec Dreamweaver.

#### Valeur renvoyée

URL du fichier, exprimée sous la forme « fichier:///URI ». Renvoie la valeur null si l'utilisateur a fermé la boîte de dialogue en cliquant sur le bouton Annuler.

#### Description

Méthode qui ouvre une boîte de dialogue Fichier Ouvrir ou Fichier Enregistrer pour permettre à l'utilisateur d'indiquer un nom de fichier à ouvrir ou à enregistrer.

#### Exemple

L'exemple suivant permet à l'utilisateur de choisir un fichier FLA, et ouvre ce fichier. (La méthode `fl.browseForFileURL()` permet de rechercher tous les types de fichiers, mais `fl.openDocument()` ne permet d'ouvrir que des fichiers FLA.)

```
var fileURL = fl.browseForFileURL("open", "Select file");  
var doc = fl.openDocument(fileURL);
```

#### Voir aussi

[fl.browseForFolderURL\(\)](#)

## fl.browseForFolderURL()

### Disponibilité

Flash 8.

### Utilisation

```
fl.browseForFolderURL([description])
```

### Paramètres

**description** Chaîne facultative spécifiant la description de la boîte de dialogue Rechercher un dossier. Si ce paramètre est omis, rien n'apparaît dans la zone de description.

### Valeur renvoyée

URL du fichier, exprimée sous la forme « fichier:///URI ». Renvoie la valeur `null` si l'utilisateur a fermé la boîte de dialogue en cliquant sur le bouton Annuler.

### Description

Méthode qui affiche la boîte de dialogue Rechercher un dossier et permet à l'utilisateur de sélectionner un dossier.

***Remarque :** le titre de la boîte de dialogue est toujours « Rechercher un dossier ». Utilisez le paramètre `description` pour ajouter sous le titre des informations dans la zone de description, par exemple « Sélectionnez un dossier » ou « Sélectionnez le chemin contenant le profil à importer ».*

### Exemple

L'exemple suivant permet à l'utilisateur de sélectionner un dossier, puis affiche la liste des fichiers qu'il contient :

```
var folderURI = fl.browseForFolderURL("Select a folder.");  
var folderContents = FLfile.listFolder(folderURI);
```

### Voir aussi

[fl.browseForFileURL\(\)](#), [Objet FLfile](#)

## fl.clearPublishCache()

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
fl.clearPublishCache()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.



### Description

Méthode : vide le cache de publication.

### Exemple

Le code suivant vide le cache de publication :

```
fl.clearPublishCache()
```

### Voir aussi

[fl.publishCacheDiskSizeMax](#), [fl.publishCacheEnabled](#), « [fl.publishCacheMemoryEntrySizeLimit](#) » à la page 251, « [fl.publishCacheMemorySizeMax](#) » à la page 251

## fl.clipCopyString()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.clipCopyString(string)
```

### Paramètres

**chaîne** Chaîne à copier dans le Presse-papiers.

### Valeur renvoyée

Aucune.

### Description

Méthode qui copie la chaîne spécifiée dans le Presse-papiers.

Pour copier la sélection actuelle dans le Presse-papiers, utilisez [document.clipCopy\(\)](#).

### Exemple

L'exemple suivant copie le chemin d'accès du document actif dans le Presse-papiers :

```
var documentPath = fl.getDocumentDOM().path;  
fl.clipCopyString(documentPath);
```

## fl.closeAll()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.closeAll([bPromptToSave])
```

### Paramètres

**bPromptToSave** Valeur booléenne facultative indiquant quelle boîte de dialogue doit s'afficher : Enregistrer, pour les fichiers modifiés depuis leur enregistrement ou Enregistrer sous, pour les fichiers qui n'ont jamais été enregistrés. La valeur par défaut est `true`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ferme tous les fichiers ouverts (fichiers FLA, SWF, JSFL, etc.). Si vous souhaitez fermer tous ces fichiers sans enregistrer une seule modification dans l'un d'entre eux, transmettez la valeur `false` pour *bPromptToSave*. Cette méthode ne ferme pas l'application.

### Exemple

Le code suivant ferme tous les fichiers ouverts en invitant l'utilisateur à enregistrer les fichiers nouveaux ou modifiés.

```
fl.closeAll();
```

### Voir aussi

[fl.closeAllPlayerDocuments\(\)](#), [fl.closeDocument\(\)](#)

## fl.closeAllPlayerDocuments()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.closeAllPlayerDocuments();
```

### Paramètres

Aucun.

### Valeur renvoyée

Valeur booléenne, `true` si une ou plusieurs fenêtres d'animation étaient ouvertes, `false` dans le cas contraire.

### Description

Méthode qui ferme tous les fichiers SWF qui ont été ouverts par la commande Contrôle > Tester l'animation.

### Exemple

L'exemple suivant ferme tous les fichiers SWF qui ont été ouverts par la commande Contrôle > Tester l'animation.

```
fl.closeAllPlayerDocuments();
```

### Voir aussi

[fl.closeAll\(\)](#), [fl.closeDocument\(\)](#)

## fl.closeDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.closeDocument (documentObject [, bPromptToSaveChanges])
```

### Paramètres

**documentObject** [Objet Document](#). Si *documentObject* fait référence au document actif, il est possible que la fenêtre Document ne se ferme pas avant la fin de l'exécution du script qui appelle cette méthode.

**bPromptToSaveChanges** Valeur booléenne. Lorsque la valeur de *bPromptToSaveChanges* est *false*, l'utilisateur n'est pas invité à enregistrer le document si celui-ci contient des modifications non enregistrées. Le fichier est simplement fermé et les changements ignorés. Si *bPromptToSaveChanges* prend la valeur *true* et si le document contient des modifications non enregistrées, l'utilisateur en est averti par la boîte de dialogue d'enregistrement standard avec les boutons Oui et Non. La valeur par défaut est *true*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ferme le document spécifié.

### Exemple

L'exemple suivant illustre deux façons de fermer un document.

```
// Closes the specified document and prompts to save changes.
fl.closeDocument(fl.documents[0]);
fl.closeDocument(fl.documents[0] , true); // Use of true is optional.
// Closes the specified document without prompting to save changes.
fl.closeDocument(fl.documents[0] , false);
```

### Voir aussi

[fl.closeAll\(\)](#)

## fl.compilerErrors

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.compilerErrors
```

### Description

Propriété en lecture seule ; objet `compilerErrors` représentant le panneau des erreurs. Pour plus d'informations sur l'utilisation de cette propriété, voir la section [Objet compilerErrors](#).

## fl.componentsPanel

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.componentsPanel`

### Description

Propriété en lecture seule ; [Objet componentsPanel](#) représentant le panneau Composants.

### Exemple

L'exemple suivant stocke un objet componentsPanel dans la variable `comPanel` :

```
var comPanel = fl.componentsPanel;
```

## fl.configDirectory

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.configDirectory`

### Description

Propriété en lecture seule ; chaîne indiquant le chemin d'accès complet du dossier Configuration de l'utilisateur local. Le format de ce chemin est spécifique à chaque plate-forme. Pour spécifier ce chemin sous la forme « fichier:///URI », ce qui n'est pas spécifique à une plate-forme, utilisez [fl.configURI](#).

### Exemple

L'exemple suivant affiche le dossier Configuration dans le panneau Sortie.

```
fl.trace("My local configuration directory is " + fl.configDirectory);
```

## fl.configURI

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.configURI`

### Description

Propriété en lecture seule ; chaîne indiquant le chemin d'accès complet du dossier Configuration de l'utilisateur local sous la forme « fichier:///URI ». Voir aussi [fl.configDirectory](#).

### Exemple

L'exemple suivant exécute le script spécifié. L'utilisation de `fl.configURI` permet de spécifier l'emplacement du script sans savoir sur quelle plate-forme le script est exécuté.

```
// To run a command in your commands menu, change "Test.Jsfl"  
// to the command you want to run in the line below.  
fl.runScript( fl.configURI + "Commands/Test.jsfl" );
```

## fl.contactSensitiveSelection

### Disponibilité

Flash 8.

### Utilisation

```
fl.contactSensitiveSelection
```

### Description

Valeur booléenne précisant si le mode de sélection Activer au contact est activé (`true`) ou non (`false`).

### Exemple

L'exemple suivant montre comment désactiver le mode de sélection Activer au contact avant d'effectuer une sélection, puis comment lui rendre sa valeur initiale après avoir effectué la sélection :

```
var contact = fl.contactSensitiveSelection;  
fl.contactSensitiveSelection = false;  
// Insert selection code here.  
fl.contactSensitiveSelection = contact;
```

## fl.createDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.createDocument ( [docType] )
```

### Paramètres

**docType** Chaîne spécifiant le type de document à créer. Les valeurs possibles sont les suivantes : "timeline", "presentation" et "application". La valeur par défaut est "timeline" et revient à sélectionner Fichier > Nouveau > Fichier Flash (ActionScript 3.0). Ce paramètre est facultatif.

### Valeur renvoyée

En cas de succès de la méthode, l'objet Document associé au nouveau document. En cas d'erreur, elle renvoie la valeur `undefined`.

### Description

Méthode qui ouvre un nouveau document et le sélectionne. Les valeurs de taille, résolution et couleur sont les valeurs par défaut actuelles.

### Exemple

L'exemple suivant crée différents types de documents :

```
// Create two Timeline-based Flash documents.  
fl.createDocument();  
fl.createDocument("timeline");  
// Create a Slide Presentation document.  
fl.createDocument("presentation");  
// Create a Form Application document.  
fl.createDocument("application");
```

## fl.createNewDocList

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.createNewDocList
```

### Description

Propriété en lecture seule ; tableau de chaînes représentant les divers types de documents qu'il est possible de créer.

### Exemple

L'exemple suivant affiche dans le panneau Sortie les types de documents qu'il est possible de créer :

```
fl.trace("Number of choices " + fl.createNewDocList.length);  
for (i = 0; i < fl.createNewDocList.length; i++)  
    fl.trace("choice: " + fl.createNewDocList[i]);
```

## fl.createNewDocListType

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.createNewDocListType
```

### Description

Propriété en lecture seule ; tableau de chaînes représentant les extensions de fichier des divers types de documents qu'il est possible de créer. Les entrées du tableau correspondent directement (à index identique) aux entrées du tableau [fl.createNewDocList](#).

### Exemple

L'exemple suivant affiche dans le panneau Sortie les extensions des types de documents qu'il est possible de créer :

```
fl.trace("Number of types " + fl.createNewDocListType.length);  
for (i = 0; i < fl.createNewDocListType.length; i++) fl.trace("type: " +  
fl.createNewDocListType[i]);
```

## fl.createNewTemplateList

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.createNewTemplateList
```

### Description

Propriété en lecture seule ; tableau de chaînes représentant les divers types de modèles qu'il est possible de créer.

### Exemple

L'exemple suivant affiche dans le panneau Sortie les types de modèles qu'il est possible de créer :

```
fl.trace("Number of template types: " + fl.createNewTemplateList.length); for (i = 0; i <  
fl.createNewTemplateList.length; i++) fl.trace("type: " + fl.createNewTemplateList[i]);
```

## fl.documents

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.documents
```

### Description

Propriété en lecture seule ; tableau des objets Document (voir [Objet Document](#)) représentant les documents (fichiers FLA) actuellement ouverts pour modification.

### Exemple

L'exemple suivant enregistre dans la variable docs un tableau des documents ouverts :

```
var docs = fl.documents;
```

L'exemple suivant affiche dans le panneau Sortie les noms de tous les documents actuellement ouverts :

```
for (doc in fl.documents) {  
    fl.trace(fl.documents[doc].name);  
}
```

## fl.drawingLayer

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.drawingLayer`

### Description

Propriété en lecture seule ; [Objet drawingLayer](#) que doit utiliser un outil extensible lorsque l'utilisateur dessine un tracé temporaire à l'aide de la souris (par exemple, pour créer un rectangle de sélection).

### Exemple

Voir `drawingLayer.setColor()`.

## fl.exportPublishProfileString()

### Disponibilité

Flash Professional CS5

### Utilisation

`fl.exportPublishProfileString( ucfURI [, profileName] )`

### Paramètres

**ucfURI** Chaîne qui spécifie l'URI (Uniform Resource Identifier) du fichier à partir duquel exporter les paramètres de publication.

**profileName** Chaîne qui spécifie le nom du profil à exporter. Ce paramètre est facultatif.

### Valeur renvoyée

Chaîne.

### Description

Renvoie le profil de publication d'un document spécifique sans avoir à ouvrir le fichier. Il est également possible de spécifier le profil de publication, mais cela est facultatif.

### Exemple

L'exemple suivant lit la chaîne du profil de publication :

```
var ppXML = "";
var ucfURI = fl.browseForFileURL("open", "select a FLA");
if (ucfURI && ucfURI.length > 0)
ppXML = fl.exportPublishProfileString(ucfURI);
fl.trace(ppXML);
```



## fl.externalLibraryPath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fl.externalLibraryPath`

### Description

Propriété ; chaîne contenant la liste des éléments dans le chemin de bibliothèque externe ActionScript 3.0 global, spécifiant l'emplacement des fichiers SWC utilisés en tant que bibliothèques partagées au moment de l'exécution. Les éléments de la chaîne sont délimités par des points-virgules. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Modifier > Préférences > ActionScript > Paramètres d'ActionScript 3.0.

### Exemple

L'exemple suivant ajoute le dossier /SWC\_runtime dans le chemin de bibliothèque externe ActionScript 3.0 global :

```
fl.trace(fl.externalLibraryPath);  
fl.externalLibraryPath = "/SWC_runtime;" + fl.externalLibraryPath;  
fl.trace(fl.externalLibraryPath);
```

### Voir aussi

[fl.flexSDKPath](#), [fl.libraryPath](#), [fl.sourcePath](#), [document.externalLibraryPath](#)

## fl.fileExists()

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.fileExists(fileURI)`

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et contenant le chemin d'accès du fichier.

### Valeur renvoyée

Valeur booléenne, `true` si le fichier existe, `false` dans le cas contraire.

### Description

Méthode qui vérifie si un fichier existe déjà sur le disque dur.

### Exemple

Pour chaque fichier spécifié, l'exemple suivant affiche dans le panneau Sortie la valeur `true` si le fichier existe, `false` dans le cas contraire.

```
alert(fl.fileExists("file:///C:/example fla"));  
alert(fl.fileExists("file:///C:/example.jsfl"));  
alert(fl.fileExists(""));
```

## fl.findDocumentDOM()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.findDocumentDOM(id)
```

### Paramètres

**id** Entier représentant un identificateur unique pour un document.

### Valeur renvoyée

Un objet Document, ou la valeur `null` s'il n'existe aucun document doté de l'*id* spécifié.

### Description

Méthode qui permet de cibler un fichier spécifique au moyen de son identificateur unique (au lieu d'utiliser sa valeur d'index, par exemple). Utilisez cette méthode conjointement avec [document.id](#).

### Exemple

L'exemple suivant illustre la lecture de l'ID d'un document, puis son utilisation pour cibler le document en question.

```
var originalDocID = fl.getDocumentDOM().id;  
// other code here, maybe working in different files  
var targetDoc = fl.findDocumentDOM(originalDocID);  
// Set the height of the Stage in the original document to 400 pixels.  
targetDoc.height = 400;
```

### Voir aussi

[fl.findDocumentIndex\(\)](#)

## fl.findDocumentIndex()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.findDocumentIndex(name)
```

### Paramètres

**name** Nom du document dont vous souhaitez connaître l'index. Ce document doit être ouvert.

### Valeur renvoyée

Tableau d'entiers représentant la position du document *name* dans le tableau `fl.documents`.

### Description

Méthode qui renvoie un tableau d'entiers représentant la position du document *name* dans le tableau `fl.documents`. Il se peut que plusieurs documents du même nom soient ouverts (s'ils se trouvent dans des dossiers différents).

### Exemple

L'exemple suivant affiche, dans le panneau Sortie, des informations sur la position d'index de tous les fichiers appelés `test fla` :

```
var filename = "test fla"
var docIndex = fl.findDocumentIndex(filename);
for (var index in docIndex)
    fl.trace(filename + " is open at index " + docIndex[index]);
```

### Voir aussi

`fl.documents`, `fl.findDocumentDOM()`

## fl.findObjectInDocByName()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.findObjectInDocByName(instanceName, document)
```

### Paramètres

**instanceName** Chaîne indiquant le nom d'occurrence d'un élément dans le document spécifié.

**document** [Objet Document](#) dans lequel rechercher l'élément spécifié.

### Valeur renvoyée

Un tableau d'objets génériques. Utilisez la propriété `.obj` de chaque élément du tableau pour obtenir l'objet. L'objet possède les propriétés suivantes : `keyframe`, `layer`, `timeline` et `parent`. Vous pouvez utiliser ces propriétés pour accéder à la hiérarchie de l'objet. Pour plus d'informations sur ces propriétés et leur accès, voir la section [fl.findObjectInDocByType\(\)](#).

Vous pouvez également accéder aux méthodes et aux propriétés des valeurs `layer` et `timeline` ; il s'agit respectivement des équivalents de l'[Objet Layer](#) et de l'[Objet Timeline](#).

### Description

Méthode qui expose les éléments d'un document dont les noms d'occurrences correspondent à un texte particulier.

***Remarque :** dans certains cas, cette méthode ne fonctionne que si elle est exécutée en tant que commande issue d'un fichier FLA, et non pendant l'affichage ou la modification du fichier JSFL.*

### Exemple

L'exemple suivant recherche les éléments nommés « `instance01` » dans le document actuel.

```
var nameToSearchFor = "instance01";
var doc = fl.getDocumentDOM();
var results = fl.findObjectInDocByName(nameToSearchFor, doc);
if (results.length > 0) {
    alert("success, found " + results.length + " objects");
}
else {
    alert("failed, no objects named " + nameToSearchFor + " found");
}
```

#### Voir aussi

[fl.findObjectInDocByType\(\)](#)

## fl.findObjectInDocByType()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.findObjectInDocByType(elementType, document)
```

### Paramètres

**elementType** Chaîne représentant le type d'élément à rechercher. Les valeurs possibles sont indiquées à la section [element.elementType](#).

**document** [Objet Document](#) dans lequel rechercher l'élément spécifié.

### Valeur renvoyée

Un tableau d'objets génériques. Utilisez la propriété `.obj` de chaque élément du tableau pour obtenir l'objet `Element`. Chaque objet possède les propriétés `keyframe`, `layer`, `timeline` et `parent`. Vous pouvez utiliser ces propriétés pour accéder à la hiérarchie de l'objet.

Vous pouvez également accéder aux méthodes et aux propriétés des valeurs `layer` et `timeline` ; il s'agit respectivement des équivalents de l'[Objet Layer](#) et de l'[Objet Timeline](#).

Le deuxième et le troisième exemple de la section Exemples montrent comment accéder à ces propriétés.

### Description

Méthode qui expose des éléments d'un type particulier dans un document.

**Remarque :** dans certains cas, cette méthode ne fonctionne que si elle est exécutée en tant que commande issue d'un fichier FLA, et non pendant l'affichage ou la modification du fichier JSFL.

### Exemple

L'exemple suivant recherche des champs de texte dans le document actif, puis modifie leur contenu :

```
var doc = fl.getDocumentDOM();
var typeToSearchFor = "text";
var results = fl.findObjectInDocByType(typeToSearchFor, doc);
if (results.length > 0) {
    for (var i = 0; i < results.length; i++) {
        results[i].obj.setTextString("new text");
    }
    alert("success, found " + results.length + " objects");
}
else {
    alert("failed, no objects of type " + typeToSearchFor + " found");
}
```

L'exemple suivant montre comment accéder aux propriétés spéciales de l'objet renvoyé par cette méthode :

```
var doc = fl.getDocumentDOM();
var resultsArray = findObjectInDocByType("text", doc);
if (resultsArray.length > 0)
{
    var firstItem = resultsArray[0];

    // firstItem.obj- This is the element object that was found.

    // You can access the following properties of this object:
    // firstItem.keyframe- The keyframe that the element is on.
    // firstItem.layer- The layer that the keyframe is on.
    // firstItem.timeline- The timeline that the layer is on.
    // firstItem.parent- The parent of the timeline. For example,
    // the timeline might be in a symbol instance.
}
```

L'exemple suivant montre comment sauvegarder le DOM pour trouver le nom d'un calque dans lequel un champ de texte a été trouvé, par le biais de l'objet resultArray.obj :

```
var doc = fl.getDocumentDOM();
var typeToSearchFor = "text";
var resultsArray = fl.findObjectInDocByType(typeToSearchFor, doc);
if (resultsArray.length > 0) {
    for (var i = 0; i < resultsArray.length; i++) {
        resultsArray[i].obj.setTextString("new text");
        var firstItem = resultsArray[0];
        firstItemObj = firstItem.obj;
        fl.trace(firstItemObj.layer.name+"layerName");
    }
} else {
    alert("failed, no objects of type " + typeToSearchFor + " found");
}
```

#### Voir aussi

[fl.findObjectInDocByName\(\)](#)

## fl.flexSDKPath

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
fl.flexSDKPath
```

### Description

Propriété ; chaîne spécifiant le dossier du kit de développement Flex, qui contient la corbeille, les structures, les bibliothèques et d'autres dossiers. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Modifier > Préférences > ActionScript > Paramètres d'ActionScript 3.0.

### Exemple

Le code suivant affiche le chemin d'accès du kit de développement Flex dans le panneau Sortie :

```
fl.trace(fl.flexSDKPath);
```

### Voir aussi

[fl.externalLibraryPath](#), [fl.libraryPath](#), [fl.sourcePath](#)

## fl.getAppMemoryInfo()

### Disponibilité

Flash 8 (Windows uniquement).

### Utilisation

```
fl.getAppMemoryInfo(memType)
```

### Paramètres

**memType** Nombre entier précisant la zone d'utilisation mémoire à demander. La liste des valeurs gérées est indiquée dans la description suivante.

### Valeur renvoyée

Entier représentant le nombre d'octets utilisés dans la zone mémoire de Flash.exe spécifiée.

### Description

Méthode (Windows uniquement) qui renvoie un entier représentant le nombre d'octets utilisés dans la zone mémoire de Flash.exe spécifiée. Utilisez le tableau suivant pour déterminer la valeur à transmettre comme *memType* :

memType	Données de ressource
0	PAGEFAULTCOUNT
1	PEAKWORKINGSETSIZE
2	WORKINGSETSIZE

memType	Données de ressource
3	QUOTAPEAKPAGEDPOOLUSAGE
4	QUOTAPAGEDPOOLUSAGE
5	QUOTAPEAKNONPAGEDPOOLUSAGE
6	QUOTANONPAGEDPOOLUSAGE
7	PAGEFILEUSAGE
8	PEAKPAGEFILEUSAGE

### Exemple

L'exemple suivant affiche la consommation actuelle de la mémoire de travail :

```
var memsize = fl.getAppMemoryInfo(2);  
fl.trace("Flash current memory consumption is " + memsize + " bytes or " + memsize/1024 + "  
KB");
```

## fl.getDocumentDOM()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.getDocumentDOM()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un objet Document, ou la valeur `null` si aucun document n'est ouvert.

### Description

Méthode qui récupère le DOM ([Objet Document](#)) du document actuellement actif (fichier FLA). Si un ou plusieurs documents sont ouverts, mais qu'aucun d'eux n'a le focus (si, par exemple, un fichier JSFL a le focus), la méthode lit le DOM du dernier document actif.

### Exemple

L'exemple suivant affiche dans le panneau Sortie le nom du document Flash actif, ou du document Flash qui a été actif le plus récemment.

```
var currentDoc = fl.getDocumentDOM();  
fl.trace(currentDoc.name);
```

## fl.getSwfPanel()

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
fl.getSwfPanel (panelName, [useLocalizedPanelName])
```

### Paramètres

**panelName** Nom localisé du panneau ou nom de fichier racine du fichier SWF du panneau. Transmettez false en tant que second paramètre si vous utilisez le nom de fichier.

**useLocalizedPanelName** Facultatif. Ce paramètre est par défaut défini sur true. S'il est défini sur false, le paramètre panelName est le nom anglais (non localisé) du panneau, qui correspond au nom du fichier SWF sans l'extension de fichier.

### Valeur renvoyée

Objet SWFPanel.

### Description

Méthode : renvoie l'objet SWFPanel en fonction du nom localisé du panneau ou de son nom de fichier SWF (sans l'extension du nom de fichier).

### Exemple

L'exemple suivant affiche le nom du panneau référencé sous le nom « Projet » dans le panneau Sortie :

```
fl.trace('name of panel is: ' + fl.getSwfPanel('Project').name);
```

## fl.installedPlayers

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
fl.installedPlayers()
```

### Paramètres

Aucun.

### Valeur renvoyée

Tableau d'objets génériques correspondant à la liste des lecteurs Flash installés dans l'inspecteur des propriétés du document.

### Description

Propriété en lecture seule : tableau d'objets génériques correspondant à la liste des lecteurs Flash installés dans l'inspecteur des propriétés du document.



Chaque objet du tableau contient les propriétés suivantes :

**name** Nom de chaîne du document.

**version** Peut servir à définir le lecteur actuel pour un document, à l'aide de la fonction `Document.setPlayerVersion()`.

**minASVersion** Version ActionScript minimale requise par le document. Il est possible d'utiliser un entier entre les valeurs `minASVersion` et `maxASVersion` pour définir la version ActionScript du document, à l'aide de la propriété `Document.asVersion`.

**maxASVersion** Version ActionScript maximale prise en charge par le document.

**stageWidth** La largeur par défaut de la scène, exprimée en pixels, pour la cible donnée. Par exemple, pour l'iPhone la taille par défaut est de 320 x 480 pixels. Pour Android, la taille par défaut est de 480 x 800.

**stageHeight** La hauteur par défaut de la scène, exprimée en pixels, pour la cible donnée. Par exemple, pour l'iPhone la taille par défaut est de 320 x 480 pixels. Pour Android, la taille par défaut est de 480 x 800.

### Exemple

L'exemple suivant suit les propriétés de tous les objets du tableau `installedPlayers` sur la fenêtre de sortie :

```
var arr = fl.installedPlayers;
for (var i in arr) fl.trace("name: " + arr[i].name + " version: " + arr[i].version + "
minASVersion: " + arr[i].minASVersion + " maxASVersion: " + arr[i].maxASVersion + " stageWidth:
" + arr[i].stageWidth + " stageHeight: " + arr[i].stageHeight + " ");
```

## fl.isFontInstalled()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
fl.isFontInstalled(fontName)
```

### Paramètres

**fontName** Chaîne qui spécifie le nom d'une police de périphérique.

### Valeur renvoyée

Valeur booléenne, `true` si la police spécifiée est installée, `false` dans le cas contraire.

### Description

Méthode qui détermine si la police spécifiée est installée.

### Exemple

Le code suivant affiche « true » dans le panneau Sortie si la police Times est installée.

```
fl.trace(fl.isFontInstalled("Times"));
```

## fl.languageCode

### Disponibilité

Flash CS5 Professional.

### Utilisation

`fl.languageCode`

### Description

Propriété ; chaîne qui renvoie le code à cinq caractères qui identifie les paramètres régionaux de l'interface utilisateur de l'application.

### Exemple

L'exemple suivant renvoie le code de langue à cinq caractères indiqué par l'interface utilisateur localisée de l'application Flash :

```
locConfigURI = fl.applicationURI + fl.languageCode + "/Configuration";
```

## fl.libraryPath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fl.libraryPath`

### Description

Propriété ; chaîne qui contient la liste des éléments du chemin de bibliothèque ActionScript 3.0 global, spécifiant l'emplacement des fichiers SWC ou des dossiers contenant des fichiers SWC. Les éléments de la chaîne sont délimités par des points-virgules. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Modifier > Préférences > ActionScript > Paramètres d'ActionScript 3.0.

### Exemple

L'exemple suivant ajoute le dossier /SWC au chemin de bibliothèque ActionScript 3.0 global :

```
fl.trace(fl.libraryPath);  
fl.libraryPath = "/SWC;" + fl.libraryPath;  
fl.trace(fl.libraryPath);
```

### Voir aussi

[fl.externalLibraryPath](#), [fl.flexSDKPath](#), [fl.sourcePath](#), [document.libraryPath](#)

## fl.mapPlayerURL()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.mapPlayerURL(URI [, returnMBCS])
```

### Paramètres

**URI** Chaîne contenant l'adresse URL avec séquences d'échappement en Unicode à convertir.

**returnMBCS** Valeur booléenne devant être définie sur `true` pour que le chemin soit renvoyé converti en MBCS avec séquences d'échappement. Sinon, la méthode renvoie une chaîne en UTF-8. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Chaîne qui est l'adresse URL convertie.

### Description

Méthode qui convertit une adresse URL avec séquences d'échappement Unicode en une URL en UTF-8 ou MBCS. Il est conseillé d'appeler cette méthode si la chaîne doit être utilisée dans ActionScript pour accéder à une ressource externe. De plus, il est nécessaire de l'utiliser pour manipuler des caractères multi-octets.

### Exemple

L'exemple suivant convertit une URL en UTF-8 pour permettre à Flash Player de la charger :

```
var url = MMExecute( "fl.mapPlayerURL(" + myURL + ", false);" );  
mc.loadMovie( url );
```

## fl.Math

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.Math
```

### Description

Propriété en lecture seule ; l'[Objet Math](#) fournit les méthodes nécessaires pour les opérations de matrice et de point.

### Exemple

L'exemple suivant montre la matrice de transformation de l'objet sélectionné et son inverse :

```
// Select an element on the Stage and then run this script.  
var mat = fl.getDocumentDOM().selection[0].matrix;  
for(var prop in mat){  
    fl.trace("mat."+prop+" = " + mat[prop]);  
}  
var invMat = fl.Math.invertMatrix( mat );  
for(var prop in invMat) {  
    fl.trace("invMat."+prop+" = " + invMat[prop]);  
}
```

## fl.mruRecentFileList

### Disponibilité

Flash MX 2004.

### Utilisation

fl.mruRecentFileList

### Description

Propriété en lecture seule ; tableau des noms de fichier complets de la liste des fichiers récemment utilisés (MRU, Most Recently Used) gérée par l'outil de création de Flash.

### Exemple

L'exemple suivant affiche dans le panneau Sortie le nombre de fichiers récemment ouverts et le nom de chacun de ces fichiers :

```
fl.trace("Number of recently opened files: " + fl.mruRecentFileList.length);  
for (i = 0; i < fl.mruRecentFileList.length; i++) fl.trace("file: " + fl.mruRecentFileList[i]);
```

## fl.mruRecentFileListType

### Disponibilité

Flash MX 2004.

### Utilisation

fl.mruRecentFileListType

### Description

Propriété en lecture seule ; tableau des types de fichiers de la liste des fichiers récemment utilisés (MRU, Most Recently Used) gérée par l'outil de création de Flash. Ce tableau correspond à celui de la propriété [fl.mruRecentFileList](#).

### Exemple

L'exemple suivant affiche dans le panneau Sortie le nombre de fichiers récemment ouverts et le type de chacun d'eux :

```
fl.trace("Number of recently opened files: " + fl.mruRecentFileListType.length);  
for (i = 0; i < fl.mruRecentFileListType.length; i++) fl.trace("type: " +  
    fl.mruRecentFileListType[i]);
```

## fl.objectDrawingMode

### Disponibilité

Flash 8.

### Utilisation

`fl.objectDrawingMode`

### Description

Propriété ; valeur booléenne déterminant si le mode de dessin d'objet est activé (`true`) ou si le mode de dessin de fusion est activé (`false`).

### Exemple

L'exemple suivant active/désactive le mode Dessin d'objet :

```
var toggleMode = fl.objectDrawingMode;
if (toggleMode) {
    fl.objectDrawingMode = false;
} else {
    fl.objectDrawingMode = true;
}
```

## fl.openDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.openDocument (fileURI)`

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et précisant le nom du fichier à ouvrir.

### Valeur renvoyée

En cas de succès de la méthode, [Objet Document](#) associé au nouveau document ouvert. Si le fichier est introuvable, ou si ce n'est pas un fichier FLA valide, une erreur est affichée et l'exécution du script est annulée.

### Description

Méthode qui ouvre dans une nouvelle fenêtre de document un document Flash (FLA) à modifier et active cette fenêtre. Pour l'utilisateur, l'effet est identique au choix de l'option de menu Fichier > Ouvrir et à la sélection d'un fichier. Si le fichier spécifié est déjà ouvert, la fenêtre qui contient ce document passe au premier plan. La fenêtre qui contient le fichier spécifié devient le document actuellement sélectionné.

### Exemple

L'exemple suivant ouvre un fichier nommé Document fla qui est enregistré dans le répertoire racine du lecteur C. Le code stocke un objet Document représentant ce document dans la variable `doc` et le définit pour qu'il soit le document actuellement sélectionné. En d'autres termes, tant que la fenêtre active reste la même (tant que le focus n'est pas modifié), `fl.getDocumentDOM()` renvoie à ce document.

```
var doc = fl.openDocument("file:///c:/Document fla");
```

## fl.openScript()

### Disponibilité

Flash MX 2004. Paramètres facultatifs ajoutés à Flash Professional CS5.

### Utilisation

```
fl.openScript(fileURI [, createExtension, className])
```

### Paramètres

**fileURI** Chaîne exprimée sous la forme fichier:///URI, qui spécifie le chemin d'accès au fichier JSFL, AS, ASC, XML, TXT ou de tout autre fichier à charger dans l'éditeur de texte de Flash. Ce paramètre peut être nul. S'il est nul, cette méthode ouvre un nouveau script du type spécifié par le paramètre `createExtension`.

**createExtension** Chaîne utilisée pour déterminer le type de document à créer si `fileURI` est défini sur null. Par défaut, 'AS' ; les valeurs autorisées sont les suivantes : 'JSFL', 'AS', 'ASC', 'XML', 'TXT', 'AS3\_CLASS' ou 'AS3\_INTERFACE'. Ce paramètre a été ajouté dans Flash Professional CS5.

**className** Chaîne utilisée pour spécifier le nom qualifié complet de la classe si une classe ou une interface est en cours de création (comme le détermine le paramètre `createExtension`). Ce paramètre a été ajouté dans Flash Professional CS5.

### Valeur renvoyée

Aucune.

### Description

Méthode ; ouvre un fichier existant, ou crée un nouveau script (JSFL, AS, ASC) ou un autre fichier (XML, TXT) dans l'éditeur de texte de Flash.

### Exemple

L'exemple suivant ouvre un fichier nommé my\_test.jsfl, qui est enregistré dans le répertoire /temp du lecteur C.

```
fl.openScript("file:///c:/temp/my_test.jsfl");
```

### Exemple

L'exemple suivant crée un nouveau fichier .as dont la définition de classe AS3 est vide :

```
fl.openScript(null, 'AS3_CLASS');
```

## fl.outputPanel

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.outputPanel`

### Description

Propriété en lecture seule ; référence à l' [Objet outputPanel](#).

### Exemple

Voir [Objet outputPanel](#).

## fl.packagePaths

### Disponibilité

Flash CS3 Professional.

### Utilisation

`fl.packagePaths`

### Description

Propriété ; chaîne correspondant au paramètre de chemin de classe global dans la boîte de dialogue des paramètres d'ActionScript 2.0. Les chemins d'accès de classe sont délimités par des points-virgules (;) dans la chaîne. Pour afficher ou modifier des paramètres de chemin de classe d'ActionScript 3.0, utilisez [fl.as3PackagePaths](#).

### Exemple

L'exemple suivant montre comment modifier les paramètres de chemin de classe d'ActionScript 2.0:

```
fl.trace(fl.packagePaths);  
// Output (assuming started with default value)  
// .;$ (LocalData)/Classes  
fl.packagePaths="buying;selling";  
fl.trace(fl.packagePaths);  
// Output  
// buying; selling
```

### Voir aussi

[fl.resetPackagePaths\(\)](#)

## fl.presetPanel

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fl.presetPanel`

### Description

Propriété en lecture seule ; [Objet presetPanel](#).

## fl.publishCacheDiskSizeMax

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

`fl.publishCacheDiskSizeMax`

### Description

Propriété : entier qui définit la taille maximale, en mégaoctets, du cache de publication sur le disque.

### Exemple

Le code suivant définit la taille maximale du cache de publication du disque sur 1 mégaoctet :

```
fl.publishCacheDiskSizeMax = 1
```

### Voir aussi

[fl.clearPublishCache\(\)](#), [fl.publishCacheEnabled](#), « [fl.publishCacheMemoryEntrySizeLimit](#) » à la page 251, « [fl.publishCacheMemorySizeMax](#) » à la page 251

## fl.publishCacheEnabled

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

`fl.publishCacheEnabled`

### Description

Propriété : valeur booléenne qui indique si le cache de publication est activé.

### Exemple

Le code suivant indique si le cache de publication est activé dans la fenêtre de sortie.

```
fl.trace(fl.publishCacheEnabled);
```

### Voir aussi

[fl.publishCacheDiskSizeMax](#), [fl.clearPublishCache\(\)](#), « [fl.publishCacheMemoryEntrySizeLimit](#) » à la page 251, « [fl.publishCacheMemorySizeMax](#) » à la page 251



## fl.publishCacheMemoryEntrySizeLimit

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

`fl.publishCacheMemoryEntrySizeLimit`

### Description

Propriété : entier qui définit la taille maximale, en kilo-octets, des entrées pouvant être ajoutées au cache de publication en mémoire. Toute entrée dont la taille est inférieure ou égale à cette valeur resteront en mémoire ; les entrées supérieures seront écrites sur le disque.

Les utilisateurs disposant de suffisamment de mémoire peuvent, s'ils le souhaitent, augmenter cette valeur pour accroître les performances ; à l'inverse, les utilisateurs qui disposent de peu de mémoire peuvent réduire cette valeur pour éviter que le cache de publication ne consomme trop de mémoire.

### Exemple

Le code suivant définit la taille maximale des entrées du cache de publication pouvant être stockées en mémoire sur 100 kilo-octets :

```
fl.publishCacheMemoryEntrySizeLimit = 100
```

### Voir aussi

[fl.publishCacheDiskSizeMax](#), [fl.publishCacheEnabled](#), « [fl.clearPublishCache\(\)](#) » à la page 226,  
« [fl.publishCacheMemorySizeMax](#) » à la page 251

## fl.publishCacheMemorySizeMax

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

`fl.publishCacheMemorySizeMax`

### Description

Propriété : entier qui définit la taille maximale, en mégaoctets, du cache de publication en mémoire.

### Exemple

Le code suivant définit la taille maximale du cache de publication en mémoire sur 1 mégaoctet :

```
fl.publishCacheMemorySizeMax = 1
```

### Voir aussi

[fl.publishCacheDiskSizeMax](#), [fl.publishCacheEnabled](#), « [fl.publishCacheMemoryEntrySizeLimit](#) » à la page 251, « [fl.clearPublishCache\(\)](#) » à la page 226

## fl.publishDocument()

### Disponibilité

Flash CS5 Professional.

### Utilisation

```
fl.publishDocument( flaURI [, publishProfile] )
```

### Paramètres

**flaURI** Chaîne, exprimée sous la forme fichier:/// URI, qui spécifie le chemin du fichier FLA devant être publié en silence.

**publishProfile** Chaîne qui spécifie le profil de publication à utiliser lors de la publication. Si ce paramètre est omis, le profil de publication par défaut est utilisé.

### Valeur renvoyée

Boolean

### Description

Méthode ; publie un fichier FLA sans l'ouvrir. Cette API ouvre le fichier FLA dans un mode sans interface utilisateur et publie le fichier SWF (ou tout autre élément sur lequel le profil est défini). Le second paramètre (publishProfile) est facultatif. La valeur renvoyée est une valeur booléenne qui indique si le profil a été trouvé ou non. Dans le cas où le second paramètre n'est pas fourni, la valeur renvoyée est toujours true.

### Exemple

L'exemple suivant invite l'utilisateur à sélectionner un fichier FLA et le publie en silence à l'aide du profil de publication par défaut :

```
var uri = fl.browseForFileURL("select", "select a FLA file to publish");  
var publishProfileName = "Default";  
fl.publishDocument(uri, publishProfileName);
```

## fl.quit()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.quit( [bPromptIfNeeded] )
```

### Paramètres

**bPromptIfNeeded** Valeur booléenne devant prendre la valeur true (par défaut) pour que l'utilisateur soit invité à enregistrer les documents modifiés. Si la valeur de ce paramètre est false, l'utilisateur ne sera pas invité à enregistrer les documents modifiés. Dans ce cas, toute modification des documents ouverts sera annulée, et l'application se fermera immédiatement. Bien que cette méthode soit utile pour le traitement par lots, il est conseillé de l'utiliser avec précaution. Ce paramètre est facultatif.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui ferme Flash en proposant à l'utilisateur d'enregistrer les documents modifiés.

#### Exemple

L'exemple suivant illustre la fermeture du programme en proposant ou non d'enregistrer les documents modifiés :

```
// Quit with prompt to save any modified documents.  
fl.quit();  
fl.quit(true); // True is optional.  
// Quit without saving any files.  
fl.quit(false);
```

## fl.reloadEffects()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
fl.reloadEffects()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui recharge tous les descripteurs d'effets définis dans le dossier Configuration Effects de l'utilisateur. Elle permet de modifier rapidement les scripts durant le développement, donc d'améliorer les effets sans redémarrer l'application. Il est conseillé d'appeler cette méthode à partir d'une commande placée dans le dossier Commands.

#### Exemple

L'exemple suivant est un script d'une ligne qui peut être placé dans le dossier Commands. S'il est nécessaire de recharger des effets, il est ensuite facile d'ouvrir le menu Commandes et d'exécuter le script.

```
fl.reloadEffects();
```

## fl.reloadTools()

#### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.reloadTools()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui reconstruit le panneau Outils à partir du contenu du fichier toolconfig.xml. Cette méthode n'est utilisée que lors de la création d'outils extensibles. Employez-la quand vous devez recharger le panneau Outils, par exemple après avoir modifié le fichier JSFL définissant un outil qui se trouve déjà dans le panneau.

### Exemple

L'exemple suivant est un script d'une ligne qui peut être placé dans le dossier Commands. Lorsque vous souhaitez recharger le panneau Outils, exécutez le script depuis le menu Commandes.

```
fl.reloadTools();
```

## fl.removeEventListener()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.removeEventListener(eventType)
```

### Paramètres

**eventType** Chaîne spécifiant le type d'événement à supprimer de cette fonction de rappel. Les valeurs possibles sont : "documentNew", "documentOpened", "documentClosed", "mouseMove", "documentChanged", "layerChanged" et "frameChanged".

### Valeur renvoyée

Valeur booléenne, `true` si l'écouteur d'événement a bien été supprimé, `false` si la fonction n'a jamais été ajoutée à la liste par la méthode `fl.addEventListener()`.

### Description

Annule l'inscription d'une fonction enregistrée à l'aide de `fl.addEventListener()`.

### Exemple

L'exemple suivant retire l'écouteur d'événement associé à l'événement `documentClosed` :

```
fl.removeEventListener("documentClosed");
```

### Voir aussi

[fl.addEventListener\(\)](#)

## fl.resetAS3PackagePaths()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.resetAS3PackagePaths()
```

### Paramètres

Aucun.

### Description

Méthode qui réinitialise le paramètre de chemin de classe global sur la valeur par défaut dans la boîte de dialogue des paramètres d'ActionScript 3.0. Pour réinitialiser le chemin de classe globale d'ActionScript 2.0, utilisez [fl.resetPackagePaths\(\)](#).

### Exemple

L'exemple suivant réinitialise le paramètre de chemin de classe d'ActionScript 3.0 sur sa valeur par défaut.

```
fl.resetAS3PackagePaths();
```

### Voir aussi

[fl.as3PackagePaths](#)

## fl.resetPackagePaths()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.resetPackagePaths()
```

### Paramètres

Aucun.

### Description

Méthode qui réinitialise le paramètre de chemin de classe global sur la valeur par défaut dans la boîte de dialogue des paramètres d'ActionScript 2.0. Pour réinitialiser le chemin de classe global d'ActionScript 3.0, utilisez [fl.resetAS3PackagePaths\(\)](#).

### Exemple

L'exemple suivant réinitialise le paramètre de chemin de classe d'ActionScript 2.0 sur sa valeur par défaut.

```
fl.resetPackagePaths();
```

### Voir aussi

[fl.packagePaths](#)

## fl.revertDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.revertDocument (documentObject)
```

### Paramètres

**documentObject** [Objet Document](#). Si *documentObject* fait référence au document actif, il est possible que la fenêtre Document ne soit pas rétablie avant la fin de l'exécution du script qui appelle cette méthode.

### Valeur renvoyée

Valeur booléenne, *true* en cas de rétablissement réussi, *false* dans le cas contraire.

### Description

Méthode qui restaure la dernière version enregistrée du document FLA spécifié. Contrairement à l'option de menu Fichier > Rétablir, cette méthode n'affiche pas de boîte de dialogue demandant à l'utilisateur de confirmer l'opération. Voir aussi [document.revert\(\)](#) et [document.canRevert\(\)](#).

### Exemple

L'exemple suivant rétablit la dernière version enregistrée du document FLA actuel ; les éventuelles modifications effectuées depuis le dernier enregistrement sont perdues.

```
fl.revertDocument (fl.getDocumentDOM());
```

## fl.runScript()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.runScript (fileURI [, funcName [, arg1, arg2, ...]])
```

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et précisant le nom du fichier de script à exécuter.

**funcName** Chaîne identifiant une fonction à exécuter dans le fichier JSFL spécifié dans le paramètre *fileURI*. Ce paramètre est facultatif.

**arg** Paramètre facultatif spécifiant un ou plusieurs arguments à transmettre à *funcname*.

### Valeur renvoyée

Une chaîne si *funcName* est spécifié, aucune valeur dans le cas contraire.

### Description

Méthode qui exécute un fichier JavaScript. Si une fonction est spécifiée dans l'un des arguments, la méthode l'exécute, ainsi que tout le code du script qui n'en fait pas partie. Le reste du code du script est exécuté avant la fonction.

### Exemple

Supposons qu'un fichier de script appelé testScript.jsfl réside dans le répertoire racine du lecteur C et qu'il contienne les éléments suivants :

```
function testFunc(num, minNum) {  
    fl.trace("in testFunc: 1st arg: " + num + " 2nd arg: " + minNum);  
}  
for (i=0; i<2; i++) {  
    fl.trace("in for loop i=" + i);  
}  
fl.trace("end of for loop");  
// End of testScript.jsfl
```

Si vous émettez la commande suivante,

```
fl.runScript("file:///C:/testScript.jsfl", "testFunc", 10, 1);
```

les informations suivantes apparaissent dans le panneau Sortie :

```
in for loop i=0  
in for loop i=1  
end of for loop  
in testFunc: 1st arg: 10 2nd arg: 1
```

Il est également possible d'appeler simplement testScript.jsfl sans exécuter de fonction, comme suit :

```
fl.runScript("file:///C:/testScript.jsfl");
```

Le résultat suivant s'affiche dans le panneau Sortie :

```
in for loop i=0  
in for loop i=1  
end of for loop
```

## fl.saveAll()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.saveAll()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui enregistre tous les documents ouverts.

Si le fichier n'a jamais été enregistré ou n'a pas été modifié depuis son dernier enregistrement, il n'est pas enregistré. Pour permettre la sauvegarde d'un fichier qui n'a pas été enregistré, ni modifié, utilisez `fl.saveDocumentAs()`.

### Exemple

L'exemple suivant enregistre tous les documents ouverts qui ont été précédemment enregistrés et modifiés depuis leur dernier enregistrement :

```
fl.saveAll();
```

### Voir aussi

`document.save()`, `document.saveAndCompact()`, `fl.saveDocument()`, `fl.saveDocumentAs()`

## fl.saveDocument()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.saveDocument(document [, fileURI])
```

### Paramètres

**document** [Objet Document](#) qui spécifie le document à enregistrer. Si *document* est `null`, le document actif est enregistré.

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et précisant le nom du document enregistré. Si le paramètre *fileURI* est `null` ou omis, le document est enregistré sous son nom actuel. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` en cas de réussite de l'enregistrement, `false` dans le cas contraire.

Si le fichier n'a jamais été enregistré, ou s'il n'a pas été modifié depuis son dernier enregistrement, il n'est pas enregistré et la valeur `false` est renvoyée. Pour permettre la sauvegarde d'un fichier qui n'a pas été enregistré, ni modifié, utilisez `fl.saveDocumentAs()`.

### Description

Méthode qui enregistre le document spécifié sous forme de fichier FLA.

### Exemple

L'exemple suivant enregistre le document actif et deux autres documents spécifiés :

```
// Save the current document.  
alert(fl.saveDocument(fl.getDocumentDOM()));  
// Save the specified documents.  
alert(fl.saveDocument(fl.documents[0], "file:///C:/example1 fla"));  
alert(fl.saveDocument(fl.documents[1], "file:///C:/example2 fla"));
```



#### Voir aussi

`document.save()`, `document.saveAndCompact()`, `fl.saveAll()`, `fl.saveDocumentAs()`

## fl.saveDocumentAs()

#### Disponibilité

Flash MX 2004.

#### Utilisation

`fl.saveDocumentAs (document)`

#### Paramètres

**document** [Objet Document](#) qui spécifie le document à enregistrer. Si *document* est `null`, le document actif est enregistré.

#### Valeur renvoyée

Valeur booléenne, `true` si l'opération Enregistrer sous est réussie, `false` dans le cas contraire.

#### Description

Méthode qui affiche la boîte de dialogue Enregistrer sous pour le document spécifié.

#### Exemple

L'exemple suivant invite l'utilisateur à enregistrer le document spécifié, puis affiche un message d'alerte qui indique si le document a été enregistré :

```
alert (fl.saveDocumentAs (fl.documents [1]) ) ;
```

#### Voir aussi

`document.save()`, `document.saveAndCompact()`, `fl.saveAll()`, `fl.saveDocument()`

## fl.scriptURI

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

`fl.scriptURI`

#### Description

Propriété en lecture seule ; chaîne représentant le chemin d'accès du script JSFL actuellement en cours d'exécution et exprimée sous la forme « fichier:///URI ». Si le script a été appelé à partir de `fl.runScript()`, cette propriété représente le chemin d'accès du script parent immédiat. Ainsi, de multiples appels à `fl.runScript()` sont évités pour trouver le chemin d'accès du script de l'appel original.

### Exemple

L'exemple suivant affiche dans le panneau Sortie le chemin d'accès du script JSFL actuellement en cours d'exécution :

```
fl.trace(fl.scriptURI);
```

### Voir aussi

[fl.runScript\(\)](#)

## fl.selectElement()

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
fl.selectElement(elementObject, editMode)
```

### Paramètres

**elementObject** [Objet Element](#) à sélectionner.

**editMode** Valeur booléenne indiquant si vous voulez modifier l'élément (`true`) ou uniquement le sélectionner (`false`).

### Valeur renvoyée

Valeur booléenne, `true` si l'élément a bien été sélectionné, `false` dans le cas contraire.

### Description

Méthode qui active l'option de sélection ou d'édition d'un élément. En caractère à récupérer. générale, vous utilisez cette méthode sur des objets renvoyés par [fl.findObjectInDocByName\(\)](#) ou [fl.findObjectInDocByType\(\)](#).

### Exemple

L'exemple suivant sélectionne un élément nommé "second text field" s'il est trouvé dans le document :

```
var nameToSearchFor = "second text field";
var doc = fl.getDocumentDOM();

// Start by viewing Scene 1 (index value of 0).
document.editScene(0);

// Search for element by name.
var results = fl.findObjectInDocByName(nameToSearchFor, doc);
if (results.length > 0) {
    // Select the first element found.
    // Pass false, so the symbolInstance you are searching for is selected.
    // If you pass true, the symbol instance will switch to edit mode.
    fl.selectElement(results[0], false);
    alert("success, found " + results.length + " objects")
}
else {
    alert("failed, no objects with name " + nameToSearchFor + " found");
}
```

#### Voir aussi

[fl.findObjectInDocByName\(\)](#), [fl.findObjectInDocByType\(\)](#)

## fl.selectTool()

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

```
fl.selectTool(toolName)
```

#### Paramètres

**toolName** Chaîne spécifiant le nom de l'outil à sélectionner. Voir la section « Description » ci-dessous pour connaître les valeurs acceptées par ce paramètre.

#### Description

Méthode qui sélectionne l'outil spécifié dans le panneau Outils. Les valeurs possibles par défaut pour *toolName* sont "arrow", "bezierSelect", "freeXform", "fillXform", "lasso", "pen", "penplus", "penminus", "penmodify", "text", "line", "rect", "oval", "rectPrimitive", "ovalPrimitive", "polystar", "pencil", "brush", "inkBottle", "bucket", "eyeDropper", "eraser", "hand" et "magnifier".

Si un utilisateur ou vous-même créez des outils personnalisés, leurs noms peuvent également être transmis comme paramètre *toolName*. La liste des noms d'outil se trouve dans le fichier suivant :

- Windows Vista :

*lecteur d'amorçage*\Users\*nom d'utilisateur*\Local Settings\Application Data\Adobe\Flash  
CS3\langue\Configuration\Tools\toolConfig.xml

- Windows XP :

*lecteur d'amorçage*\Documents and Settings\*nom d'utilisateur*\Local Settings\Application Data\Adobe\Flash  
CS3\langue\Configuration\Tools\toolConfig.xml

- Mac OS X:

Macintosh HD/Utilisateurs/*nom d'utilisateur*/Bibliothèque/Application Support/Adobe/Flash  
CS3/langue/Configuration/Tools/toolConfig.xml

#### Exemple

L'exemple suivant sélectionne l'outil Plume .

```
fl.selectTool("pen") ;
```

#### Voir aussi

[Objet Tools](#), [Objet ToolObj](#)

## fl.setActiveWindow()

### Disponibilité

Flash MX 2004.

### Utilisation

```
fl.setActiveWindow(document [, bActivateFrame])
```

### Paramètres

**document** [Objet Document](#) spécifiant le document à sélectionner comme fenêtre active.

**bActivateFrame** Paramètre facultatif qui est ignoré par Flash et Fireworks, et uniquement destiné à la compatibilité avec Dreamweaver.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit le document spécifié comme fenêtre active. Cette méthode est également prise en charge par Dreamweaver et Fireworks. S'il existe plusieurs fenêtres ouvertes sur le même document (créées à l'aide de l'option Fenêtre > Dupliquer la fenêtre), la fenêtre active la plus récente est sélectionnée.

### Exemple

L'exemple suivant montre deux façons d'activer un document spécifié :

```
fl.setActiveWindow(fl.documents[0]);

var theIndex = fl.findDocumentIndex("myFile.fl");
fl.setActiveWindow(fl.documents[theIndex]);
```

## fl.showIdleMessage()

### Disponibilité

Flash 8.

### Utilisation

```
fl.showIdleMessage(show)
```

### Paramètres

**show** Valeur booléenne indiquant si le message avertissant d'une exécution du script trop longue doit être activé ou désactivé.

### Valeur renvoyée

Aucune.

### Description

Méthode qui permet de désactiver le message avertissant que l'exécution du script est trop longue (transmettez la valeur `false` pour *show*). Il est conseillé d'utiliser cette méthode pour les opérations de traitement par lots qui nécessitent beaucoup de temps. Pour réactiver l'alerte, appelez une nouvelle fois cette méthode en transmettant cette fois la valeur `true` pour *show*.

### Exemple

L'exemple suivant montre comment désactiver puis réactiver le message avertissant que l'exécution du script est trop longue :

```
fl.showIdleMessage(false);  
var result = timeConsumingFunction();  
fl.showIdleMessage(true);  
var result = timeConsumingFunction();
```

## fl.sourcePath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fl.sourcePath`

### Description

Propriété ; chaîne qui contient la liste des éléments du chemin source ActionScript 3.0 global, spécifiant l'emplacement des fichiers de classes ActionScript. Les éléments de la chaîne sont délimités par des points-virgules. Dans l'outil de création, les éléments sont spécifiés en sélectionnant Modifier > Préférences > ActionScript > Paramètres d'ActionScript 3.0.

### Exemple

L'exemple suivant ajoute le dossier /Classes au chemin source ActionScript 3.0 global :

```
fl.trace(fl.sourcePath);  
fl.sourcePath = "/Classes;" + fl.sourcePath;  
fl.trace(fl.sourcePath);
```

### Voir aussi

[fl.flexSDKPath](#), [fl.externalLibraryPath](#), [fl.libraryPath](#), [document.sourcePath](#)

## fl.swfPanels

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fl.swfPanels`

### Description

Propriété en lecture seule ; tableau d'objets `swfPanel` enregistrés (voir [Objet swfPanel](#)). Un objet `swfPanel` est enregistré s'il a été ouvert au moins une fois.

La position du panneau dans le tableau représente l'ordre dans lequel il a été ouvert. Si le premier panneau ouvert se nomme `TraceBitmap` et le second panneau `AnotherFunction`, `fl.swfPanels[0]` est l'objet `TraceBitmap swfPanel`, `fl.swfPanels[1]` est l'objet `AnotherFunction swfPanel`, etc.

### Exemple

Le code suivant affiche dans le panneau Sortie le nom et le chemin des panneaux Window SWF enregistrés :

```
if(fl.swfPanels.length > 0){
    for(x = 0; x < fl.swfPanels.length; x++){
        fl.trace("Panel: " + fl.swfPanels[x].name + " -- Path: " + fl.swfPanels[x].path);
    }
}
```

## fl.toggleBreakpoint()

### Disponibilité

Flash Professional CS5

### Utilisation

`fl.toggleBreakPoint(String fileURI, int line, Boolean enable)`

### Paramètres

**fileURI** Chaîne ; URI du fichier AS dans lequel activer/désactiver le point d'arrêt.

**line** Entier ; numéro de ligne à laquelle activer/désactiver le point d'arrêt.

**enable** Boolean ; si la valeur `true` est sélectionnée, le point d'arrêt est activé. Si la valeur `false` est sélectionnée, le point d'arrêt est désactivé.

### Description

Active/désactive un point d'arrêt pour le fichier `.as` donné à la ligne donnée. Si le paramètre `enable` est défini sur `false`, le point d'arrêt actuellement mémorisé à cette ligne est effacé.

### Exemple

L'exemple suivant active un point d'arrêt à la ligne 10 du fichier AS situé à l'emplacement `C:\AS\breakpointTest.as` :

```
fl.toggleBreakPoint("file:///C:/AS/breakpointTest.as", 10, 1);
```

## fl.tools

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.tools`

### Description

Propriété en lecture seule ; tableau d'objets Tools (voir [Objet Tools](#)). Cette méthode n'est utilisée que lors de la création d'outils extensibles.

## fl.trace()

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.trace(message)`

### Paramètres

**message** Chaîne apparaissant dans le panneau Sortie.

### Valeur renvoyée

Aucune.

### Description

Méthode qui envoie une chaîne de texte au panneau Sortie, suivie d'un code de nouvelle ligne, et affiche le panneau Sortie s'il n'est pas déjà visible. Cette méthode est identique à `outputPanel.trace()` et fonctionne comme l'instruction `trace()` en code ActionScript.

Pour envoyer une ligne vide, utilisez `fl.trace("")` ou `fl.trace("\n")`. Vous pouvez utiliser cette dernière commande en ligne si vous intégrez `\n` à la chaîne *message*.

### Exemple

L'exemple suivant affiche plusieurs lignes de texte dans le panneau Sortie :

```
fl.outputPanel.clear();  
fl.trace("Hello World!!!");  
var myPet = "cat";  
fl.trace("\nI have a " + myPet);  
fl.trace("");  
fl.trace("I love my " + myPet);  
fl.trace("Do you have a " + myPet + "?");
```

## fl.version

### Disponibilité

Flash MX 2004.

### Utilisation

`fl.version`

### Description

Propriété en lecture seule ; chaîne longue de la version de l'outil de création de Flash, plate-forme comprise.

### Exemple

L'exemple suivant affiche la version de l'outil de création de Flash dans le panneau Sortie :

```
alert(fl.version); // For example, WIN 10,0,0,540
```

## fl.xmlui

### Disponibilité

Flash MX 2004.

### Utilisation

fl.xmlui

### Description

Propriétés en lecture seule ; [Objet XMLUI](#). Cette propriété permet de lire et définir les propriétés de XMLUI dans une boîte de dialogue XMLUI, et d'accepter ou d'annuler par programmation la boîte de dialogue.

### Exemple

Voir [Objet XMLUI](#).



# Chapitre 18 : Objet FLfile

## Disponibilité

Flash MX 2004 7.2.

## Description

L'objet FLfile permet de créer des extensions Flash capables d'accéder aux dossiers et fichiers locaux, de les modifier et de les supprimer. L'API FLfile est proposée sous forme d'extension de l'API JavaScript. Cette extension est appelée *bibliothèque partagée* et se trouve dans le dossier suivant :

- Windows Vista :

*lecteur d'amorçage* \Users\*nom d'utilisateur*\Local Settings\Application Data\Adobe\Flash CS3\langue\Configuration\External Libraries\FLfile.dll

- Windows XP :

*lecteur d'amorçage* \Documents and Settings\*nom d'utilisateur*\Local Settings\Application Data\Adobe\Flash CS3\langue\Configuration\External Libraries\FLfile.dll

- Mac OS X:

Macintosh HD/Utilisateurs/*nom d'utilisateur*/Bibliothèque/Application Support/Adobe/FlashCS3/langue/Configuration/External Libraries/FLfile.dll

**Remarque :** *il ne faut pas confondre les bibliothèques partagées qui contiennent les symboles de vos documents Flash avec les bibliothèques partagées de l'API JavaScript. Il s'agit de deux types de composants différents.*

Les méthodes de FLfile s'appliquent aux fichiers et aux dossiers (répertoires) du disque. Par conséquent, chaque méthode prend un ou plusieurs paramètres pour spécifier l'emplacement d'un fichier ou dossier. Cet emplacement de fichier ou de dossier est exprimé sous forme d'une chaîne très semblable à l'adresse URL d'un site Web. Ce type d'adresse, appelé URI de fichier (Uniform Resource Identifier) respecte le format représenté ici (guillemets compris) :

```
"file:///drive|/folder 1/folder 2/.../filename"
```

Par exemple, pour créer un dossier nommé config sur le lecteur C et le placer dans le dossier Program Files/MyApp, utilisez la commande suivante :

```
FLfile.createFolder("file:///C:/Program Files/MyApp/config");
```

Si vous souhaitez ensuite placer dans ce dossier un fichier nommé config.ini, la commande sera la suivante :

```
FLfile.write("file:///C:/Program Files/MyApp/config/config.ini", "");
```

Pour créer un dossier sous Macintosh, vous pouvez utiliser une commande similaire à celle-ci :

```
FLfile.createFolder("file:///Macintosh/MyApp/config");
```

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet FLfile :

Méthode	Description
<code>FLfile.copy()</code>	Copie un fichier.
<code>FLfile.createFolder()</code>	Crée un ou plusieurs dossiers.
<code>FLfile.exists()</code>	Vérifie l'existence d'un fichier ou d'un dossier.
<code>FLfile.getAttributes()</code>	Vérifie les attributs d'un fichier : accessible en écriture, en lecture seule, caché, visible ou dossier système.
<code>FLfile.getCreationDate()</code>	Détermine combien de secondes se sont écoulées entre le 1er janvier 1970 et l'heure à laquelle le fichier ou le dossier a été créé.
<code>FLfile.getCreationDateObj()</code>	Récupère la date de création d'un fichier ou d'un dossier.
<code>FLfile.getModificationDate()</code>	Détermine combien de secondes se sont écoulées entre le 1er janvier 1970 et l'heure de la dernière modification du fichier ou du dossier.
<code>FLfile.getModificationDateObj()</code>	Récupère la date de la dernière modification d'un fichier ou d'un dossier.
<code>FLfile.getSize()</code>	Récupère la taille d'un fichier.
<code>FLfile.listFolder()</code>	Enumère le contenu d'un dossier.
<code>FLfile.platformPathToURI()</code>	Convertit un nom de fichier en un format propre à la plate-forme : fichier:/// URI.
<code>FLfile.read()</code>	Lit le contenu d'un fichier.
<code>FLfile.remove()</code>	Supprime un fichier ou un dossier.
<code>FLfile.setAttributes()</code>	Donne à un fichier ou à un dossier l'attribut lecture seule, lecture et écriture, caché ou visible.
<code>FLfile.uriToPlatformPath()</code>	Convertit un nom de fichier exprimé sous la forme fichier:/// URI en un format propre à la plate-forme.
<code>FLfile.write()</code>	Crée un fichier, écrit du contenu ou en ajoute à la fin d'un fichier.

## FLfile.copy()

### Disponibilité

Flash MX 2004 7.2.

### Utilisation

```
FLfile.copy(fileURI, copyURI)
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui indique le fichier à copier.

**copyURI** Chaîne de type « fichier:///URI » qui spécifie l'emplacement et le nom du fichier copié.

### Valeur renvoyée

Valeur booléenne, `true` si l'opération a réussi, ou `false` dans le cas contraire.

### Description

Méthode qui copie un fichier d'un emplacement à un autre. Cette méthode renvoie *false* si *copyURI* existe déjà.

### Exemple

L'exemple suivant crée une copie de sauvegarde d'un fichier de configuration nommé *config.ini*, et place cette copie dans le même dossier que l'original, avec un nouveau nom :

```
var originalFileURI="file:///C:/Program Files/MyApp/config.ini";  
var newFileURI="file:///C:/Program Files/MyApp/config_backup.ini";  
FLfile.copy(originalFileURI, newFileURI);
```

Il est possible d'exécuter la même tâche avec une seule commande :

```
FLfile.copy("file:///C:/Program Files/MyApp/config.ini", file:///C:/Program  
Files/MyApp/config_backup.ini");
```

## FLfile.createFolder()

### Disponibilité

Flash MX 2004 7.2.

### Utilisation

```
FLfile.createFolder(folderURI)
```

### Paramètres

**folderURI** Adresse URI indiquant le chemin d'accès de la structure de dossiers à créer.

### Valeur renvoyée

Valeur booléenne : *true* si l'opération a réussi, *false* si *folderURI* existe déjà.

### Description

Méthode qui crée un ou plusieurs dossiers à l'emplacement spécifié.

Vous pouvez créer plusieurs dossiers simultanément. Par exemple, la commande suivante crée les dossiers *MyData* et *TempData* s'ils n'existent pas :

```
FLfile.createFolder("file:///c:/MyData/TempData")
```

### Exemple

L'exemple suivant crée un dossier et un sous-dossier sous le dossier de configuration ([fl.configURI](#)) :

```
fl.trace(FLfile.createFolder(fl.configURI+"folder01/subfolder01"));
```

L'exemple suivant tente de créer un dossier appelé *tempFolder* à la racine du lecteur C, puis affiche une alerte informant de la réussite ou de l'échec de l'opération.

```
var folderURI = "file:///c:/tempFolder";
if (FLfile.createFolder(folderURI)) {
    alert("Created " + folderURI);
}
else {
    alert(folderURI + " already exists");
}
```

#### Voir aussi

[FLfile.remove\(\)](#), [FLfile.write\(\)](#)

## FLfile.exists()

#### Disponibilité

Flash MX 2004 7.2.

#### Utilisation

`FLfile.exists(fileURI)`

#### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui désigne le fichier à vérifier.

#### Valeur renvoyée

Valeur booléenne, `true` si l'opération a réussi, ou `false` dans le cas contraire.

#### Description

Méthode qui vérifie si le fichier spécifié existe. Si vous spécifiez un dossier et un nom de fichier, le dossier doit déjà exister. Pour créer des dossiers, voir [FLfile.createFolder\(\)](#).

#### Exemples

L'exemple suivant vérifie l'existence d'un fichier nommé mydata.txt dans le dossier temp, puis affiche une alerte signalant l'existence ou l'absence d'un tel fichier :

```
var fileURI = "file:///c:/temp/mydata.txt";
if (FLfile.exists(fileURI)) {
    alert( fileURI + " exists.");
}
else {
    alert( fileURI + " does not exist.");
}
```

L'exemple suivant vérifie l'existence d'un fichier de configuration requis dans le dossier MyApplication. Si le fichier n'existe pas, il est créé automatiquement.

```
var configFile = "file:///C:/MyApplication/config.ini";
if (!FLfile.exists(configFile)) {
    FLfile.write(configFile, "");
}
```

#### Voir aussi

[FLfile.write\(\)](#)

## FLfile.getAttributes()

#### Disponibilité

Flash MX 2004 7.2.

#### Utilisation

```
FLfile.getAttributes(fileOrFolderURI)
```

#### Paramètres

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui désigne le fichier ou le dossier dont les attributs doivent être récupérés.

#### Valeur renvoyée

Chaîne représentant les attributs du fichier ou dossier spécifié.

si le fichier ou le dossier n'existe pas, les résultats sont imprévisibles. Vous devez utiliser [FLfile.exists\(\)](#) avant cette méthode.

#### Description

Méthode qui renvoie une chaîne représentant les attributs du fichier ou dossier indiqué, ou une chaîne vide si le fichier ne possède pas d'attributs spécifiques (c'est-à-dire s'il n'est pas en lecture seule, caché, etc.). Employez toujours [FLfile.exists\(\)](#) pour vérifier l'existence du fichier ou du dossier avant d'utiliser cette méthode.

Les caractères de cette chaîne représentent les attributs sous la forme suivante :

- **R** — *fileOrFolderURI* est en lecture seule.
- **D** — *fileOrFolderURI* est un dossier (répertoire).
- **H** — *fileOrFolderURI* est caché (Windows uniquement).
- **S** — *fileOrFolderURI* est un fichier ou dossier système (Windows uniquement).
- **A** — *fileOrFolderURI* est prêt pour l'archivage (Windows uniquement).

Par exemple, si *fileOrFolderURI* est un dossier caché, la chaîne renvoyée est "DH".

#### Exemple

L'exemple suivant récupère les attributs du fichier mydata.txt et affiche un message d'alerte si ce fichier est en lecture seule.

```
var URI = "file:///c:/temp/mydata.txt";
if (FLfile.exists(URI)) {
    var attr = FLfile.getAttributes(URI);
    if (attr && (attr.indexOf("R") != -1)) { // Returned string contains R.
        alert(URI + " is read only!");
    }
}
```

**Voir aussi**

`FLfile.setAttributes()`

## FLfile.getCreationDate()

**Disponibilité**

Flash MX 2004 7.2.

**Utilisation**

`FLfile.getCreationDate(fileOrFolderURI)`

**Paramètres**

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui indique le fichier ou le dossier dont la date et l'heure de création doivent être lues, sous forme hexadécimale.

**Valeur renvoyée**

Chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1er janvier 1970 et l'heure à laquelle le fichier ou le dossier a été créé, ou "00000000" si le fichier ou le dossier n'existe pas.

**Description**

Méthode qui détermine combien de secondes se sont écoulées entre le 1er janvier 1970 et l'heure à laquelle le fichier ou le dossier a été créé. Cette méthode est principalement utilisée pour comparer les dates de création ou de modification des fichiers ou des dossiers.

**Exemple**

L'exemple suivant détermine si un fichier a été modifié depuis sa création :

```
// Make sure the specified file exists
var fileURI = "file:///C:/MyApplication/MyApp.flc";
var creationTime = FLfile.getCreationDate(fileURI);
var modificationTime = FLfile.getModificationDate(fileURI);
if ( modificationTime > creationTime ) {
    alert("The file has been modified since it was created.");
}
else {
    alert("The file has not been modified since it was created.");
}
```

**Voir aussi**

`FLfile.getCreationDateObj()`, `FLfile.getModificationDate()`

## FLfile.getCreationDateObj()

**Disponibilité**

Flash MX 2004 7.2.

### Utilisation

`FLfile.getCreationDateObj(fileOrFolderURI)`

### Paramètres

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui indique le fichier ou le dossier dont la date et l'heure de création doivent être lues, sous forme d'un objet Date JavaScript.

### Valeur renvoyée

Objet Date JavaScript qui représente la date et l'heure de création du fichier ou du dossier spécifié. Si le fichier ou le dossier n'existe pas, l'objet contient des informations indiquant qu'il a été créé à minuit (GMT) le 31 décembre 1969.

### Description

Méthode qui renvoie un objet Date JavaScript qui représente la date et l'heure de création du fichier ou du dossier spécifié.

### Exemple

L'exemple suivant affiche dans le panneau Sortie, dans un format lisible par l'opérateur, la date à laquelle un fichier a été créé :

```
// Make sure the specified file exists.  
var file1Date = FLfile.getCreationDateObj("file:///c:/temp/file1.txt");  
fl.trace(file1Date);
```

### Voir aussi

`FLfile.getCreationDate()`, `FLfile.getModificationDateObj()`

## FLfile.getModificationDate()

### Disponibilité

Flash MX 2004 7.2.

### Utilisation

`FLfile.getModificationDate(fileOrFolderURI)`

### Paramètres

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui désigne le fichier dont la date de modification doit être récupérée, sous forme de chaîne hexadécimale.

### Valeur renvoyée

Chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1er janvier 1970 et l'heure de la dernière modification du fichier ou du dossier, ou "00000000" si le fichier n'existe pas.

### Description

Méthode qui détermine combien de secondes se sont écoulées entre le 1er janvier 1970 et l'heure de la dernière modification du fichier ou du dossier. Cette méthode est principalement utilisée pour comparer les dates de création ou de modification des fichiers ou des dossiers.

### Exemple

L'exemple suivant compare les dates de modification de deux fichiers et détermine lequel des deux a été modifié le plus récemment :

```
// Make sure the specified files exist.
file1 = "file:///C:/MyApplication/MyApp fla";
file2 = "file:///C:/MyApplication/MyApp.as";
modificationTime1 = FLfile.getModificationDate(file1);
modificationTime2 = FLfile.getModificationDate(file2) ;
if(modificationTime1 > modificationTime2) {
    alert("File 2 is older than File 1") ;
}
else if(modificationTime1 < modificationTime2) {
    alert("File 1 is older than File 2") ;
}
else {
    alert("File 1 and File 2 were saved at the same time") ;
}
```

### Voir aussi

[FLfile.getCreationDate\(\)](#), [FLfile.getModificationDateObj\(\)](#)

## FLfile.getModificationDateObj()

### Disponibilité

Flash MX 2004 7.2.

### Utilisation

`FLfile.getModificationDateObj(fileOrFolderURI)`

### Paramètres

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui désigne le fichier ou le dossier dont la date et l'heure de modification doivent être récupérées, sous forme d'un objet Date JavaScript.

### Valeur renvoyée

Objet Date JavaScript qui représente la date et l'heure de la dernière modification du fichier ou du dossier spécifié. Si le fichier ou le dossier n'existe pas, l'objet contient des informations indiquant qu'il a été créé à minuit (GMT) le 31 décembre 1969.

### Description

Méthode qui renvoie un objet Date JavaScript représentant la date et l'heure de la dernière modification du fichier ou du dossier spécifié.

### Exemple

L'exemple suivant affiche dans le panneau Sortie, dans un format lisible par l'opérateur, la date de la dernière modification d'un fichier :

```
// Make sure the specified file exists.
var file1Date = FLfile.getModificationDateObj("file:///c:/temp/file1.txt");
trace(file1Date);
```



#### Voir aussi

`FLfile.getCreationDateObj()`, `FLfile.getModificationDate()`

## FLfile.getSize()

#### Disponibilité

Flash MX 2004 7.2.

#### Utilisation

```
FLfile.getSize(fileURI)
```

#### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui indique le fichier dont la taille doit être récupérée.

#### Valeur renvoyée

Un entier qui représente la taille en octets du fichier spécifié, ou 0 si ce fichier n'existe pas.

#### Description

Méthode qui renvoie un entier représentant la taille en octets du fichier spécifié, ou 0 si ce fichier n'existe pas. Si la valeur envoyée est 0, utilisez `FLfile.exists()` pour déterminer si le fichier contient zéro octet ou s'il n'existe pas.

Cette méthode renvoie des valeurs de taille de fichier correctes uniquement pour les fichiers n'excédant pas 2 Go.

#### Exemple

L'exemple suivant stocke la taille du fichier mydata.txt dans la variable `fileSize` :

```
var URL = "file:///c:/temp/mydata.txt";  
var fileSize = FLfile.getSize(URL);
```

## FLfile.listFiles()

#### Disponibilité

Flash MX 2004 7.2.

#### Utilisation

```
FLfile.listFiles(folderURI [, filesOrDirectories])
```

#### Paramètres

**folderURI** Chaîne de type « fichier:///URI » qui indique le dossier dont le contenu doit être récupéré. Il est possible d'insérer des caractères génériques dans *folderURI*. Les caractères génériques utilisables sont \* (un ou plusieurs caractères) et ? (un seul caractère).

**filesOrDirectories** Chaîne facultative qui spécifie si seuls les noms des fichiers ou des dossiers (répertoires) doivent être renvoyés. Si ce paramètre est omis, les noms des dossiers et des fichiers sont renvoyés. Les valeurs gérées sont "files" et "directories".

### Valeur renvoyée

Tableau de chaînes représentant le contenu du dossier. Si le dossier n'existe pas ou si aucun fichier ou dossier ne correspond aux critères spécifiés, renvoie un tableau vide.

### Description

Méthode qui renvoie un tableau de chaînes représentant le contenu du dossier.

### Exemples

L'exemple suivant renvoie trois tableaux. Le premier représente tous les fichiers du dossier C:\temp, le second tous les dossiers du dossier C:\temp et le troisième représente les fichiers et les dossiers du dossier C:\temp :

```
var fileURI = "file:///C:/temp/" ;
var folderURI = "file:///C:/temp" ;
var fileList1 = FLfile.listFolder(fileURI, "files"); // files
var fileList2 = FLfile.listFolder(folderURI, "directories"); //folders
var fileList3 = FLfile.listFolder(folderURI); //files and folders
fl.trace("Files: " + fileList1);
fl.trace("");
fl.trace("Folders:  " + fileList2);
fl.trace("");
fl.trace("Files and folders: " + fileList3);
```

L'exemple suivant renvoie un tableau contenant tous les fichiers texte (.txt) du répertoire « temp » et affiche cette liste dans un message d'alerte :

```
var folderURI = "file:///c:/temp";
var fileMask = "*.txt";
var list = FLfile.listFolder(folderURI + "/" + fileMask, "files");
if (list) {
    alert(folderURI + " contains: " + list.join(" "));
}
```

L'exemple suivant utilise un masque de nom de fichier dans l'adresse *folderURI* spécifiée afin de renvoyer les noms de tous les fichiers exécutables du dossier d'applications Windows :

```
var executables = FLfile.listFolder("file:///C:/WINDOWS/*.exe", "files");
alert(executables.join("\n"));
```

## FLfile.platformPathToURI()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
FLfile.platformPathToURI(fileName)
```

### Paramètres

**fileName** Chaîne, exprimée en un format propre à la plate-forme, qui spécifie le nom du fichier à convertir.

### Valeur renvoyée

Chaîne exprimée sous la forme fichier:/// URI.

### Description

Méthode qui convertit un nom de fichier en un format propre à la plate-forme : fichier:/// URI.

### Exemple

L'exemple suivant convertit le nom de fichier au format spécifique à la plate-forme en format file:/// URI, transmis ensuite à `outputPanel.save()` :

```
var myFilename = "C:\\outputPanel.txt";
var myURI=FLfile.platformPathToURI(myFilename);
fl.outputPanel.save(myURI);
```

### Voir aussi

[FLfile.uriToPlatformPath\(\)](#)

## FLfile.read()

### Disponibilité

Flash MX 2004 7.2.

### Utilisation

`FLfile.read()`

### Paramètres

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui désigne le fichier ou le dossier dont les attributs doivent être récupérés.

### Valeur renvoyée

Le contenu du fichier spécifié sous forme de chaîne, ou `null` si la lecture échoue.

### Description

Méthode qui renvoie le contenu du fichier spécifié sous forme de chaîne, ou `null` si la lecture échoue.

### Exemples

L'exemple suivant lit le fichier `mydata.txt` et, si la lecture réussit, affiche son contenu dans un message d'alerte.

```
var fileURI = "file:///c:/temp/mydata.txt";
var str = FLfile.read( fileURI);
if (str) {
    alert( fileURL + " contains: " + str);
}
```

L'exemple suivant lit le code ActionScript d'un fichier de classe et le stocke dans la variable `code` :

```
var classFileURI = "file:///C:/MyApplication/TextCarousel.as";
var code = FLfile.read(classFileURI);
```

# FLfile.remove()

## Disponibilité

Flash MX 2004 7.2.

## Utilisation

`FLfile.remove(fileOrFolderURI)`

## Paramètres

**fileOrFolderURI** Chaîne de type « fichier:///URI » qui désigne le fichier ou le dossier à supprimer.

## Valeur renvoyée

Valeur booléenne, `true` si l'opération a réussi, ou `false` dans le cas contraire.

## Description

Méthode qui supprime le fichier ou le dossier spécifié. Si le dossier contient des fichiers, ceux-ci sont également supprimés. Il est impossible de supprimer les fichiers ayant l'attribut R (lecture seule).

## Exemples

L'exemple suivant prévient l'utilisateur de l'existence d'un fichier, et supprime celui-ci si l'utilisateur le décide :

```
var fileURI = prompt ("Enter file/folder to be deleted: ", "file:///c:/temp/delete.txt");
if (FLfile.exists(fileURI)) {
    var confirm = prompt("File exists. Delete it? (y/n)", "y");
    if (confirm == "y" || confirm == "Y") {
        if(FLfile.remove(fileURI)) {
            alert(fileURI + " is deleted.");
        }
        else {
            alert("fail to delete " + fileURI);
        }
    }
}
else {
    alert(fileURI + " does not exist");
}
```

L'exemple suivant supprime un fichier de configuration créé par une application :

```
if(FLfile.remove("file:///C:/MyApplication/config.ini")) {
    alert("Configuration file deleted");
}
```

L'exemple suivant supprime le dossier Configuration et tout son contenu :

```
FLfile.remove("file:///C:/MyApplication/Configuration/");
```

## Voir aussi

[FLfile.createFolder\(\)](#), [FLfile.getAttributes\(\)](#)

## FLfile.setAttributes()

### Disponibilité

Flash MX 2004 7.2.

### Utilisation

```
FLfile.setAttributes(fileURI, strAttrs)
```

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et désignant le fichier dont les attributs doivent être définis.

**strAttrs** Chaîne indiquant les valeurs des attributs à définir. La liste des valeurs gérées par *strAttrs* figure à la section Description ci-dessous.

### Valeur renvoyée

Valeur booléenne, `true` si l'opération a réussi.

**Remarque :** si le fichier ou le dossier n'existe pas, les résultats sont imprévisibles. Vous devez utiliser `FLfile.exists()` avant cette méthode.

### Description

Méthode qui désigne les attributs système du fichier spécifié.

Les valeurs suivantes sont possibles pour *strAttrs* :

- **N** — Aucun attribut spécifique (ni lecture seule, ni caché, etc.)
- **A** — Prêt pour l'archivage (Windows uniquement)
- **R** — Lecture seule (sur Macintosh, « Lecture seule » signifie verrouillé)
- **w** — Accessible en écriture (annule **R**)
- **H** — Caché (Windows uniquement)
- **v** — Visible (annule **H**, Windows uniquement)

Si vous insérez à la fois **R** et **w** dans *strAttrs*, le **R** est ignoré et le fichier est paramétré comme étant accessible en écriture. De même, si vous insérez **H** et **v**, le **H** est ignoré et le fichier est paramétré comme étant visible.

Pour vous assurer que l'attribut d'archivage n'est pas défini, utilisez cette commande avec le paramètre **N** avant de définir les attributs. En d'autres termes, le paramètre **A** ne possède pas de pendant qui annule l'attribut Archive.

### Exemples

L'exemple suivant met le fichier mydata.txt en lecture seule et caché. Il n'a aucun effet sur l'attribut d'archivage.

```
var URI = "file:///c:/temp/mydata.txt";  
if (FLfile.exists(URI)) {  
    FLfile.setAttributes(URI, "RH");  
}
```

L'exemple suivant met le fichier mydata.txt en lecture seule et caché. Il fait aussi en sorte que l'attribut d'archivage ne soit pas défini.

```
var URI = "file:///c:/temp/mydata.txt";

if (FLfile.exists(URI)) {
    FLfile.setAttributes(URI, "N");
    FLfile.setAttributes(URI, "RH");
}
```

#### Voir aussi

[FLfile.getAttributes\(\)](#)

## FLfile.uriToPlatformPath()

#### Disponibilité

Flash CS4 Professional.

#### Utilisation

```
FLfile.uriToPlatformPath(fileURI)
```

#### Paramètres

**fileURI** Chaîne, exprimée sous la forme fichier:/// URI, spécifiant le nom du fichier à convertir.

#### Valeur renvoyée

Chaîne représentant un chemin propre à la plate-forme.

#### Description

Méthode qui convertit un nom de fichier exprimé sous la forme fichier:/// URI en un format propre à la plate-forme.

#### Exemple

L'exemple suivant convertit le format file:/// URI en un format spécifique à la plate-forme :

```
var dir = (fl.configDirectory);
var URI = FLfile.platformPathToURI(dir);
fl.trace(URI == fl.configURI); // displays "true"
```

#### Voir aussi

[FLfile.platformPathToURI\(\)](#)

## FLfile.write()

#### Disponibilité

Flash MX 2004 7.2.

#### Utilisation

```
FLfile.write(fileURI, textToWrite, [ , strAppendMode])
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui désigne le fichier dans lequel une chaîne doit être écrite.

**textToWrite** Chaîne représentant le texte à placer dans le fichier.

**strAppendMode** Chaîne facultative ayant la valeur `append`. Elle indique que la chaîne *textToWrite* doit être écrite à la fin du fichier existant. Si ce paramètre est omis, le contenu actuel de *fileURI* est remplacé par le contenu de *textToWrite*.

### Valeur renvoyée

Valeur booléenne, `true` si l'opération a réussi, ou `false` dans le cas contraire.

### Description

Méthode qui écrit la chaîne spécifiée dans le fichier spécifié (au format UTF-8). Si le fichier désigné n'existe pas, il est créé automatiquement. Toutefois, le dossier dans lequel vous placez le fichier doit exister avant l'emploi de cette méthode. Pour créer des dossiers, utilisez `FLfile.createFolder()`.

### Exemple

L'exemple suivant tente d'écrire la chaîne "xxx" dans le fichier mydata.txt, et affiche un message d'alerte si l'écriture a réussi. Il tente ensuite d'écrire la chaîne "aaa" dans le même fichier et affiche un second message d'alerte si l'écriture a réussi. Après l'exécution de ce script, le fichier mydata.txt ne contiendra que le texte "xxxaaa".

```
var URI = "file:///c:/temp/mydata.txt";
if (FLfile.write(URI, "xxx")) {
    alert("Wrote xxx to " + URI);
}
if (FLfile.write(URI, "aaa", "append")) {
    alert("Appended aaa to " + fileURI);
}
```

### Voir aussi

`FLfile.createFolder()`, `FLfile.exists()`

# Chapitre 19 : Objet folderItem

**Héritage**    [Objet Item](#) > Objet folderItem

**Disponibilité**

Flash MX 2004.

**Description**

L'objet folderItem est une sous-classe de l'objet Item. Il n'existe pas de méthodes ni de propriétés particulières pour l'objet folderItem. Voir [Objet Item](#).



# Chapitre 20 : Objet fontItem

**Héritage**    [Objet Item](#) > Objet fontItem

## Disponibilité

Flash MX 2004.

## Description

L'objet fontItem est une sous-classe de l'objet Item (voir [Objet Item](#)).

## Résumé des propriétés

Outre celles de l'objet Item, l'objet fontItem gère les propriétés suivantes :

Propriété	Description
<a href="#">fontItem.bitmap</a>	Spécifie si l'élément Font est mis en bitmap.
<a href="#">fontItem.bold</a>	Spécifie si l'élément Font est en gras.
<a href="#">fontItem.embeddedCharacters</a>	Spécifie les caractères à incorporer.
<a href="#">fontItem.embedRanges</a>	Spécifie les éléments pouvant être sélectionnés dans la boîte de dialogue Incorporation de polices.
<a href="#">fontItem.embedVariantGlyphs</a>	Spécifie si les variantes de glyphes devraient être en sortie dans la police lors de la publication d'un fichier SWF.
<a href="#">fontItem.font</a>	Nom de la police de périphérique associée à l'élément Font.
<a href="#">fontItem.isDefineFont4Symbol</a>	Spécifie le format de la police qui est en sortie lors de la publication d'un fichier SWF.
<a href="#">fontItem.italic</a>	Spécifie si l'élément est en italique.
<a href="#">fontItem.size</a>	Taille de l'élément Font, en points.

## fontItem.bitmap

## Disponibilité

Flash CS4 Professional.

## Utilisation

`fontItem.bitmap`

## Description

Propriété ; valeur booléenne qui spécifie si le l'élément Font est mis en bitmap (`true`) ou non (`false`).

## Exemple

En supposant que le premier élément de la bibliothèque soit un élément Font, le code suivant affiche `true` dans le panneau Sortie s'il est mis en bitmap, `false` dans le cas contraire :

```
var theItem = fl.getDocumentDOM().library.items[0];
fl.trace("bitmap: " + theItem.bitmap);
```

## **fontItem.bold**

### **Disponibilité**

Flash CS4 Professional.

### **Utilisation**

`fontItem.bold`

### **Description**

Propriété ; valeur booléenne qui spécifie si l'élément Font est en gras (`true`) ou non (`false`).

### **Exemple**

En supposant que le premier élément de la bibliothèque soit un élément Font, le code suivant affiche `true` dans le panneau Sortie s'il est en gras, `false` dans le cas contraire, puis le définit sur gras.

```
var theItem = fl.getDocumentDOM().library.items[0];
fl.outputPanel.clear();
fl.trace("bold: " + theItem.bold);
theItem.bold=true;
fl.trace("bold: " + theItem.bold);
```

## **fontItem.embeddedCharacters**

### **Disponibilité**

Flash CS5 Professional.

### **Utilisation**

`fontItem.embeddedCharacters`

### **Description**

Propriété ; valeur de chaîne qui permet de spécifier les caractères à incorporer dans un fichier SWF de façon à ce que ces derniers ne soient pas obligatoirement présents sur les périphériques sur lesquels le fichier SWF sera lu. Cette propriété offre la même fonctionnalité que la boîte de dialogue Incorporation de polices.

Cette propriété est également accessible en lecture afin de savoir quels caractères ont été spécifiés à l'aide de la boîte de dialogue Incorporation de polices pour une option de police donnée.

### **Exemple**

En supposant que le premier élément de la bibliothèque est une police, le code suivant incorpore les caractères a, b et c.

```
fl.getDocumentDOM().library.items[0].embeddedCharacters = "abc";
```

## **fontItem.embedRanges**

### **Disponibilité**

Flash CS5 Professional.

### Utilisation

`fontItem.embedRanges`

### Description

Propriété ; valeur de chaîne qui spécifie une série d'entiers délimités correspondant aux options pouvant être sélectionnées dans la boîte de dialogue Incorporation de polices.

Cette propriété est également accessible en lecture afin de savoir quels caractères ont été spécifiés à l'aide de la boîte de dialogue Incorporation de polices pour une option de police donnée.

**Remarque :** les numéros de plage correspondent au fichier `FontEmbedding/UnicodeTables.xml` trouvé dans le dossier de configuration.

### Exemple

En supposant que le premier élément de la bibliothèque est une police, le code suivant incorpore les plages identifiées par les entiers 1, 3 et 7.

```
fl.getDocumentDOM().library.items[0].embedRanges = "1|3|7";
```

En supposant que le premier élément de la bibliothèque est une police, le code suivant réinitialise les plages à incorporer.

```
fl.getDocumentDOM().library.items[0].embedRanges = "";
```

## fontItem.embedVariantGlyphs

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fontItem.embedVariantGlyphs`

### Description

**Remarque :** bien que cette propriété soit disponible dans Flash CS5 Professional, elle n'a aucun effet lorsqu'elle est appliquée au texte de Text Layout Framework (TLF). À partir de Flash Professional CS5, les variantes de glyphe sont toujours incorporées aux polices utilisées avec le texte de TLF. Le moteur `flash.text.engine` (FTE) référencé ci-dessous est disponible uniquement dans Flash Professional CS4.

Propriété ; valeur booléenne qui spécifie si les variantes de glyphes devraient être en sortie dans la police lors de la publication d'un fichier SWF (`true`) ou non (`false`). Si cette valeur est définie sur `true`, la taille de votre fichier SWF va augmenter. La valeur par défaut est `false`.

Certaines langues substituent dynamiquement les glyphes au fur et à mesure que vous tapez (par exemple, le thaïlandais, l'arabe, l'hébreu et le grec). Si vous préparez ou que vous entrez du texte dans ces types de langue, définissez cette propriété sur `true`.

### Exemples

Les symboles de police qui sont compatibles avec les interfaces de programmation flash.text apparaissent dans la bibliothèque et l'utilisateur peut les gérer directement. Cependant, les symboles de police qui sont compatibles avec les interfaces de programmation flash.text.engine (FTE) n'apparaissent pas dans la bibliothèque ; il vous faut donc vous en occuper manuellement. La fonction suivante ajoute une nouvelle police à la bibliothèque qui peut être utilisée avec les API de FTE.

```
function embedFontSymbol(symbolName, fontName, includeVariants) {
    var doc = fl.getDocumentDOM();
    if (doc) {
        // look up the item. if it exists, delete it.
        var index = doc.library.findItemIndex(symbolName);
        if (index > -1)
            doc.library.deleteItem(symbolName);

        // make a new font symbol in the library
        doc.library.addNewItem('font', symbolName);

        // look up the symbol by its name
        var index = doc.library.findItemIndex(symbolName);
        if (index > -1) {
            // get the item from the library and set the attributes of interest
            var fontObj = doc.library.items[index];
            fontObj.isDefineFont4Symbol = true;
            fontObj.font = fontName;
            fontObj.bold = false;
            fontObj.italic = false;
            fontObj.embedVariantGlyphs = includeVariants;
            // this is what forces the font into the SWF stream
            fontObj.linkageExportForAS = true;
            fontObj.linkageExportInFirstFrame = true;
        }
    }
}
```

La fonction suivante affiche tous les symboles de police dans le panneau Sortie.

```
function dumpFontSymbols()
{
    var doc = fl.getDocumentDOM();
    if (doc) {
        var items = doc.library.items;
        fl.trace("items length = " + items.length);
        var i;
        for(i=0; i<items.length; i++) {
            var item = items[i];
            fl.trace("itemType = " + item.itemType);
            if (item.itemType == 'font') {
                fl.trace("name = " + item.name);
                fl.trace("DF4 symbol = " + item.isDefineFont4Symbol);
                fl.trace("font = " + item.font);
            }
        }
    }
}
```

#### Voir aussi

`fontItem.isDefineFont4Symbol`, `text.embedVariantGlyphs`

## fontItem.font

#### Disponibilité

Flash CS4 Professional.

#### Utilisation

`fontItem.font`

#### Description

Propriété ; chaîne qui spécifie le nom de la police de périphérique associée à l'élément Font. Si vous entrez une chaîne qui ne correspond pas à une police de périphérique installée, un message d'erreur apparaît. Pour déterminer si une police est présente dans le système, utilisez `fl.isFontInstalled()`.

*Remarque : lorsque vous définissez cette valeur, la valeur de la propriété résultante peut différer de la chaîne saisie. Examinez l'exemple suivant :*

#### Exemple

En supposant que le premier élément de la bibliothèque soit un élément Font, le code suivant affiche le nom de la police de périphérique actuellement associée à cet élément, puis le définit sur Times :

```
fl.outputPanel.clear();
var theItem = fl.getDocumentDOM().library.items[0];
fl.trace(theItem.font);
theItem.font = "Times";
// depending on your system, the following may display something like "Times-Roman"
fl.trace(theItem.font);
```

## fontItem.isDefineFont4Symbol

#### Disponibilité

Flash CS4 Professional.

#### Utilisation

`fontItem.isDefineFont4Symbol`

#### Description

Propriété ; valeur booléenne qui spécifie le format de la police qui est en sortie lors de la publication d'un fichier SWF. Si cette valeur est `true`, Flash fournit une police qui peut être utilisée avec les interfaces de programmation de `flash.text.engine` (FTE). Si cette valeur est `false`, la police peut être utilisée avec les interfaces de programmation de `flash.text`, y compris les champs de texte. La valeur par défaut est `false`.

#### Exemple

Voir `fontItem.embedVariantGlyphs`.

## fontItem.italic

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fontItem.italic`

### Description

Propriété ; valeur booléenne qui spécifie si l'élément est en italique (`true`) ou non (`false`).

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément Font, le code suivant affiche `true` dans le panneau Sortie s'il est en italique, `false` dans le cas contraire, puis le définit sur italique.

```
var theItem = fl.getDocumentDOM().library.items[0];
fl.outputPanel.clear();
fl.trace("italic: " + theItem.italic);
theItem.italic=true;
fl.trace("italic: " + theItem.italic);
```

## fontItem.size

### Disponibilité

Flash CS4 Professional.

### Utilisation

`fontItem.size`

### Description

Propriété ; nombre entier représentant la taille de l'élément Font, en points.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément Font, le code suivant affiche la taille en points de l'élément dans le panneau Sortie, puis la définit sur 24.

```
var theItem = fl.getDocumentDOM().library.items[0];
fl.outputPanel.clear();
fl.trace("font size: " + theItem.size);
theItem.size=24;
fl.trace("font size: " + theItem.size);
```

# Chapitre 21 : Objet Frame

## Disponibilité

Flash MX 2004.

## Description

L'objet Frame représente les images du calque.

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet Frame :

Méthode	Description
<code>frame.convertMotionObjectTo2D()</code>	Convertit l'objet de mouvement sélectionné en objet de mouvement 2D.
<code>frame.convertMotionObjectTo3D()</code>	Convertit l'objet de mouvement sélectionné en objet de mouvement 3D.
<code>frame.getCustomEase()</code>	Renvoie un tableau d'objets JavaScript possédant tous une propriété <i>x</i> et <i>y</i> .
<code>frame.getMotionObjectXML()</code>	Renvoie l'élément XML de mouvement à partir de l'objet de mouvement sélectionné.
<code>frame.hasMotionPath()</code>	Vous informe si la sélection actuelle est ou non une interpolation de mouvement.
<code>frame.is3DMotionObject()</code>	Vous informe si la sélection actuelle est ou non un objet de mouvement 3D.
<code>frame.isMotionObject()</code>	Vous informe si la sélection actuelle est ou non un objet de mouvement.
<code>frame.selectMotionPath()</code>	Sélectionne ou désélectionne la trajectoire de l'objet de mouvement actuel.
<code>frame.setCustomEase()</code>	Spécifie une courbe de Bézier cubique à utiliser comme courbe d'accélération personnalisée.
<code>frame.setMotionObjectDuration()</code>	Spécifie la durée (longueur de la plage d'interpolation) de l'objet de mouvement actuellement sélectionné.
<code>frame.setMotionObjectXML()</code>	Applique l'élément XML de mouvement spécifié à l'objet de mouvement sélectionné.

## Résumé des propriétés

Vous pouvez utiliser les propriétés suivantes avec l'objet Frame :

Propriété	Description
<code>frame.actionScript</code>	Chaîne représentant du code ActionScript.
<code>frame.duration</code>	Lecture seule ; entier indiquant le nombre d'images présentes dans une séquence d'images.
<code>frame.elements</code>	Lecture seule ; tableau d'objets Element (voir <a href="#">Objet Element</a> ).
<code>frame.hasCustomEase</code>	Valeur booléenne qui spécifie si l'image extrait ses informations d'accélération de la courbe d'accélération personnalisée.
<code>frame.labelType</code>	Chaîne qui spécifie le type du nom de l'image.
<code>frame.motionTweenOrientToPath</code>	Valeur booléenne qui spécifie si l'élément interpolé doit pivoter ou non pendant son déplacement le long d'un tracé afin de préserver son angle par rapport à chaque point du tracé.

Propriété	Description
<code>frame.motionTweenRotate</code>	Chaîne qui spécifie le type de rotation de l'élément interpolé.
<code>frame.motionTweenRotateTimes</code>	Entier qui spécifie le nombre de rotations de l'élément interpolé entre l'image-clé de départ et la suivante.
<code>frame.motionTweenScale</code>	Valeur booléenne qui spécifie si l'élément interpolé doit subir une mise à l'échelle à la taille de l'objet de la prochaine image-clé, en augmentant ainsi sa taille à chaque image de l'interpolation ( <code>true</code> ), ou s'il ne doit pas subir de mise à l'échelle ( <code>false</code> ).
<code>frame.motionTweenSnap</code>	Valeur booléenne qui spécifie si l'élément interpolé doit être automatiquement accroché au point le plus proche du calque de guide de mouvement associé au calque de cette image ( <code>true</code> ) ou non ( <code>false</code> ).
<code>frame.motionTweenSync</code>	Valeur booléenne qui, lorsqu'elle est définie sur <code>true</code> , synchronise l'animation de l'objet interpolé avec le scénario principal.
<code>frame.name</code>	Chaîne qui spécifie le nom de l'image.
<code>frame.shapeTweenBlend</code>	Chaîne qui spécifie comment l'interpolation d'une forme passe en fondu entre cette forme dans l'image-clé de départ de l'interpolation et la même forme dans l'image-clé suivante.
<code>frame.soundEffect</code>	Chaîne qui spécifie les effets d'un son joint directement à une image ( <code>frame.soundLibraryItem</code> ).
<code>frame.soundLibraryItem</code>	Élément de bibliothèque (voir <a href="#">Objet SoundItem</a> ) utilisé pour créer un son.
<code>frame.soundLoop</code>	Valeur entière qui spécifie le nombre de fois où un son joint directement à une image ( <code>frame.soundLibraryItem</code> ) est lu.
<code>frame.soundLoopMode</code>	Chaîne qui spécifie si un son joint directement à une image ( <code>frame.soundLibraryItem</code> ) doit être lu un certain nombre de fois ou en boucle infinie.
<code>frame.soundName</code>	Chaîne qui spécifie le nom d'un son joint directement à une image ( <code>frame.soundLibraryItem</code> ), tel qu'il est stocké dans la bibliothèque.
<code>frame.soundSync</code>	Chaîne qui spécifie le comportement de synchronisation d'un son joint directement à une image ( <code>frame.soundLibraryItem</code> ).
<code>frame.startFrame</code>	Lecture seule ; index de la première image d'une séquence.
<code>frame.tweenEasing</code>	Entier qui spécifie la quantité d'accélération à appliquer à l'objet interpolé.
<code>frame.tweenInstanceName</code>	Attribue un nom d'occurrence à l'objet de mouvement spécifié.
<code>frame.tweenType</code>	Chaîne qui spécifie le type d'interpolation.
<code>frame.useSingleEaseCurve</code>	Valeur booléenne qui spécifie si une même courbe d'accélération personnalisée est utilisée pour les informations d'accélération de toutes les propriétés.

## frame.convertMotionObjectTo2D()

### Disponibilité

Flash Professional CS5

### Utilisation

```
frame.convertMotionObjectTo2D()
```



### Description

Méthode ; convertit l'objet de mouvement sélectionné en objet de mouvement 2D.

### Exemple

L'exemple suivant convertit l'objet de mouvement sélectionné en objet de mouvement 2D :

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
var layer = my_tl.layers[my_tl.currentLayer];
var frame = layer.frames[my_tl.currentFrame];
return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject() && the()) {
theFrame.convertMotionObjectTo2D();
}else{
fl.trace("It isn't motion or it's already a 2D motion");
}
```

## frame.convertMotionObjectTo3D()

### Disponibilité

Flash Professional CS5

### Utilisation

```
frame.convertMotionObjectTo3D()
```

### Description

Méthode ; convertit l'objet de mouvement sélectionné en objet de mouvement 3D.

### Exemple

L'exemple suivant convertit l'objet de mouvement sélectionné en objet de mouvement 3D :

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
var layer = my_tl.layers[my_tl.currentLayer];
var frame = layer.frames[my_tl.currentFrame];
return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject() && !theFrame.is3DMotionObject()) {
theFrame.convertMotionObjectTo3D();
}else{
fl.trace("It isn't motion or it's already a 3D motion");
}
```

## frame.actionScript

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.actionScript`

### Description

Propriété ; chaîne représentant du code ActionScript. Pour insérer un caractère de nouvelle ligne, utilisez "\n".

### Exemple

L'exemple suivant attribue le code `stop()` à l'action du calque supérieur de la première image :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].actionScript = 'stop();';
```

## frame.duration

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.duration`

### Description

Propriété en lecture seule ; entier indiquant le nombre d'images présentes dans une séquence d'images.

### Exemple

L'exemple suivant enregistre dans la variable `frameSpan` le nombre d'images d'une séquence qui débute à la première image du calque supérieur :

```
var frameSpan = fl.getDocumentDOM().getTimeline().layers[0].frames[0].duration;
```

## frame.elements

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.elements`

### Description

Propriété en lecture seule ; tableau d'objets `Element` (voir [Objet Element](#)). L'ordre des éléments est celui dans lequel ils apparaissent dans le fichier FLA. Si la scène comporte plusieurs formes, et si toutes ces formes sont dissociées, Flash les traite comme un seul élément. Si chaque forme fait partie d'un groupe (si bien qu'il existe plusieurs groupes sur la scène), Flash les traite comme des éléments séparés. Autrement dit, Flash traite les formes non groupées comme un même élément, quel que soit le nombre de formes individuelles figurant sur la scène. Si une image contient trois formes brutes (non groupées), la propriété `elements.length` de cette image renvoie la valeur 1. Pour contourner le problème, vous pouvez sélectionner individuellement toutes les formes et les grouper.

### Exemple

L'exemple suivant enregistre dans la variable `myElements` un tableau des éléments figurant dans le calque supérieur :

```
var myElements = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements;
```

## frame.getCustomEase()

### Disponibilité

Flash 8.

### Utilisation

```
Frame.getCustomEase ( [property] )
```

### Paramètres

**property** Chaîne facultative qui spécifie la propriété dont vous souhaitez lire la valeur d'accélération. Les valeurs gérées sont les suivantes : "all", "position", "rotation", "scale", "color" et "filters". La valeur par défaut est "all".

### Valeur renvoyée

Renvoie un tableau d'objets JavaScript possédant tous une propriété `x` et `y`.

### Description

Méthode qui renvoie un tableau d'objets représentant les points de contrôle de la courbe de Bézier cubique qui définit la courbe d'accélération.

### Exemple

L'exemple suivant renvoie la valeur d'accélération personnalisée de la propriété `position` de la première image du calque supérieur :

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
var easeArray = theFrame.getCustomEase("position");
```

### Voir aussi

[frame.hasCustomEase](#), [frame.setCustomEase\(\)](#), [frame.useSingleEaseCurve](#)

## frame.getMotionObjectXML()

### Disponibilité

Flash Professional CS5

### Utilisation

`Frame.getMotionObjectXML()`

### Description

Renvoie une chaîne de l'élément XML de mouvement à partir de l'objet de mouvement sélectionné.

### Exemple

L'exemple suivant renvoie l'élément XML de mouvement à partir de l'objet de mouvement sélectionné.

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
    var layer = my_tl.layers[my_tl.currentLayer];
    var frame = layer.frames[my_tl.currentFrame];
    return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject()) {
    //fl.trace(theFrame.getMotionObjectXML());
}else{
    fl.trace("It is not motion.");
}
```

## frame.hasCustomEase

### Disponibilité

Flash 8.

### Utilisation

`frame.hasCustomEase`

### Description

Propriété : valeur booléenne. Si elle est `true`, l'image reçoit ses informations d'accélération de la courbe d'accélération personnalisée. Si elle est `false`, l'image reçoit ses informations d'accélération de la valeur d'accélération.

### Exemple

L'exemple suivant spécifie que la première image du calque supérieur doit recevoir ses informations d'accélération de la valeur d'accélération, plutôt que de la courbe d'accélération personnalisée :

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
theFrame.hasCustomEase = false;
```

### Voir aussi

[frame.getCustomEase\(\)](#), [frame.setCustomEase\(\)](#), [frame.useSingleEaseCurve](#)

## frame.hasMotionPath()

### Disponibilité

Flash Professional CS5

### Utilisation

Frame.hasMotionPath()

### Description

Méthode ; valeur booléenne. Vous informe si la sélection actuelle inclut une trajectoire de mouvement.

### Exemple

L'exemple suivant renvoie une instruction trace vous indiquant si la sélection actuelle dispose d'une trajectoire de mouvement.

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function(){
    var layer = my_tl.layers[my_tl.currentLayer];
    var frame = layer.frames[my_tl.currentFrame];
    return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject()){
    if (theFrame.hasMotionPath()){
        fl.trace("There is a motion path");
    }else{
        fl.trace("There is no motion path");
    }
}
```

## frame.is3DMotionObject()

### Disponibilité

Flash Professional CS5

### Utilisation

Frame.is3DMotionObject()

### Description

Méthode ; valeur booléenne. Vous informe si la sélection actuelle est un objet de mouvement 3D.

### Exemple

L'exemple suivant renvoie une instruction trace vous indiquant si la sélection actuelle est un objet de mouvement 3D ou pas.

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
var layer = my_tl.layers[my_tl.currentLayer];
var frame = layer.frames[my_tl.currentFrame];
return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject() && theFrame.is3DMotionObject()) {
fl.trace("This selection is 3D Motion");
} else {
fl.trace("This selection is not 3D motion");
}
```

## frame.isMotionObject()

### Disponibilité

Flash Professional CS5

### Utilisation

Frame.isMotionObject()

### Description

Méthode ; valeur booléenne. Vous informe si la sélection actuelle est un objet de mouvement.

### Exemple

L'exemple suivant renvoie une instruction trace vous indiquant si la sélection actuelle est un objet de mouvement ou pas.

```
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
var layer = my_tl.layers[my_tl.currentLayer];
var frame = layer.frames[my_tl.currentFrame];
return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject()) {
fl.trace("This selection is motion.");
} else {
fl.trace("This selection is not motion.");
}
```

## frame.labelType

### Disponibilité

Flash MX 2004.

### Utilisation

frame.labelType

### Description

Propriété ; chaîne qui spécifie le type de nom d'une image. Les valeurs gérées sont "none" (aucun), "name" (nom), "comment" (commentaire) ou "anchor" (ancre). La définition d'une étiquette sur "none" efface la propriété `frame.name`.

### Exemple

L'exemple suivant définit sur "First Frame" le nom de la première image du calque supérieur, puis sur "comment" son étiquette :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].name = 'First Frame';  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].labelType = 'comment';
```

## frame.motionTweenOrientToPath

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.motionTweenOrientToPath`

### Description

Propriété ; valeur booléenne qui spécifie si l'élément interpolé doit pivoter pendant son déplacement le long d'un tracé afin de préserver son angle par rapport à chaque point du tracé (`true`) ou s'il ne doit pas pivoter (`false`).

Si vous souhaitez spécifier une valeur pour cette propriété, définissez `frame.motionTweenRotate` sur "none".

## frame.motionTweenRotate

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.motionTweenRotate`

### Description

Propriété ; chaîne qui spécifie le type de rotation de l'élément interpolé. Les valeurs gérées sont les suivantes : "none" (aucune), "auto", "clockwise" (sens horaire) et "counter-clockwise" (sens anti-horaire). La valeur "auto" signifie que l'objet subira une rotation dans la direction qui nécessite le moins de mouvement pour correspondre à la rotation de cet objet dans la prochaine image-clé.

Si vous souhaitez spécifier une valeur pour `frame.motionTweenOrientToPath`, définissez cette propriété sur "none".

### Exemple

Voir `frame.motionTweenRotateTimes`.

## frame.motionTweenRotateTimes

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.motionTweenRotateTimes`

### Description

Propriété ; entier qui spécifie le nombre de rotations de l'élément interpolé entre l'image-clé de départ et la suivante.

### Exemple

L'exemple suivant fait effectuer à l'élément qui se trouve dans l'image active trois rotations dans le sens anti-horaire dans le délai nécessaire pour arriver à l'image-clé suivante :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenRotate = "counter-clockwise";  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenRotateTimes = 3;
```

## frame.motionTweenScale

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.motionTweenScale`

### Description

Propriété ; valeur booléenne qui spécifie si l'élément interpolé doit subir une mise à l'échelle à la taille de l'objet de la prochaine image-clé, en augmentant ainsi sa taille à chaque image de l'interpolation (`true`), ou s'il ne doit pas subir de mise à l'échelle (`false`).

### Exemple

L'exemple suivant spécifie que l'élément interpolé doit subir une mise à l'échelle à la taille de l'objet de l'image-clé suivante, en augmentant ainsi sa taille à chaque image de l'interpolation.

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenScale = true;
```

## frame.motionTweenSnap

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.motionTweenSnap`



### Description

Propriété ; valeur booléenne qui spécifie si l'élément interpolé doit être automatiquement accroché au point le plus proche du calque de guide de mouvement associé au calque de cette image (`true`) ou non (`false`).

## frame.motionTweenSync

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.motionTweenSync`

### Description

Propriété ; valeur booléenne qui, lorsqu'elle est définie sur `true`, synchronise l'animation de l'objet interpolé avec le scénario principal.

### Exemple

L'exemple suivant indique que l'objet interpolé doit être synchronisé avec le scénario :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenSync = true;
```

## frame.name

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.name`

### Description

Propriété ; chaîne qui spécifie le nom de l'image.

### Exemple

L'exemple suivant définit sur "First Frame" le nom de la première image du calque supérieur, puis enregistre dans la variable `frameLabel` la valeur de sa propriété `name` :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].name = 'First Frame';  
var frameLabel = fl.getDocumentDOM().getTimeline().layers[0].frames[0].name;
```

## frame.selectMotionPath()

### Disponibilité

Flash Professional CS5

### Utilisation

`Frame.selectMotionPath()`

### Description

Méthode ; valeur booléenne. Sélectionne (true) ou désélectionne (false) la trajectoire de l'objet de mouvement actuel.

### Exemple

Cet exemple sélectionne ou désélectionne la trajectoire de l'objet de mouvement actuel.

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
t    his.getCurrentFrame = function(){
var layer = my_tl.layers[my_tl.c u rrentLayer];
var frame = layer.frames[my_tl.currentFrame];
return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject()){
if (theFrame.hasMotionPath()){
theFrame.selectMotionPath(true);
}
}
else{
fl.trace("There is no motion path");
}
}
else{
fl.trace("It is no motion");
}
}
```

## frame.setCustomEase()

### Disponibilité

Flash 8.

### Utilisation

`frame.setCustomEase(property, easeCurve)`

### Paramètres

**property** Chaîne qui spécifie la propriété pour laquelle la courbe d'accélération doit être utilisée. Les valeurs gérées sont les suivantes : "all", "position", "rotation", "scale", "color" et "filters".

**easeCurve** Tableau d'objets qui définissent la courbe d'accélération. Chaque élément de ce tableau doit être un objet JavaScript avec des propriétés *x* et *y*.

### Valeur renvoyée

Aucune.

### Description

Méthode qui spécifie un tableau de coordonnées de points de contrôle et de points de fin de tangente qui décrivent une courbe de Bézier cubique à utiliser comme courbe d'accélération personnalisée. Ce tableau est basé sur la position horizontale (ordinal : de gauche à droite) des points de contrôle et des points de fin de tangente.

### Exemple

L'exemple suivant définit la courbe d'accélération de toutes les propriétés de la première image du premier calque sur la courbe de Bézier spécifiée par le tableau `easeCurve` :

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0];
var easeCurve = [ {x:0,y:0}, {x:.3,y:.3}, {x:.7,y:.7}, {x:1,y:1} ];
theFrame.setCustomEase( "all", easeCurve );
```

### Voir aussi

[frame.getCustomEase\(\)](#), [frame.hasCustomEase](#), [frame.useSingleEaseCurve](#)

## frame.setMotionObjectDuration()

### Disponibilité

Flash Professional CS5

### Utilisation

`Frame.setMotionObjectDuration( duration [, stretchExistingKeyframes] )`

### Paramètres

**duration** Spécifie le nombre d'images pour la plage d'interpolation de l'objet de mouvement sélectionné.

**stretchExistingKeyframes** Valeur booléenne qui détermine si la plage d'interpolation est étirée ou si des images sont ajoutées à la fin de la dernière image.

### Description

Méthode ; définit la durée (longueur de la plage d'interpolation) de l'objet de mouvement actuellement sélectionné.

### Exemple

L'exemple suivant spécifie une durée de 11 images pour l'objet de mouvement sélectionné.

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
    var layer = my_tl.layers[my_tl.currentLayer];
    var frame = layer.frames[my_tl.currentFrame];
    return frame;
}
var theFrame = getCurrentFrame();
if(theFrame.isMotionObject()) {
    theFrame.setMotionObjectDuration(11);
} else {
    fl.trace("It isn't motion");
}
```

## frame.setMotionObjectXML()

### Disponibilité

Flash Professional CS5

### Utilisation

```
Frame.setMotionObjectXML( xmlstr [, endAtCurrentLocation] )
```

### Paramètres

**xmlstr** Valeur de chaîne qui spécifie la chaîne XML.

**endAtCurrentLocation** Valeur booléenne qui détermine si l'interpolation commence ou se termine à la position actuelle.

### Description

Méthode ; applique l'élément XML de mouvement spécifié à l'objet de mouvement sélectionné.

### Exemple

Cet exemple indique que l'élément XML de mouvement identifié comme `myMotionXML` doit être appliqué à l'objet de mouvement sélectionné.

```
var doc = fl.getDocumentDOM();
var my_tl = doc.getTimeline();
this.getCurrentFrame = function() {
var layer = my_tl.layers[my_tl.currentLayer];
var frame = layer.frames[my_tl.currentFrame];
return frame;
}
var theFrame = getCurrentFrame();
theFrame.setMotionObjectXML(myMotionXML.toString(), false);
```

## frame.shapeTweenBlend

### Disponibilité

Flash MX 2004.

### Utilisation

```
frame.shapeTweenBlend
```

### Description

Propriété ; chaîne qui spécifie comment l'interpolation d'une forme passe en fondu entre cette forme dans l'image-clé de départ de l'interpolation et la même forme dans l'image-clé suivante. Les valeurs gérées sont "distributive" (distributif) ou "angular" (angulaire).

## frame.soundEffect

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.soundEffect`

### Description

Propriété ; chaîne qui spécifie les effets d'un son joint directement à une image ([frame.soundLibraryItem](#)). Les valeurs gérées sont les suivantes : "none", "left channel", "right channel", "fade left to right", "fade right to left", "fade in", "fade out" et "custom".

### Exemple

L'exemple suivant spécifie que le son associé à la première image doit apparaître en fondu à l'ouverture :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundEffect = "fade in";
```

## frame.soundLibraryItem

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.soundLibraryItem`

### Description

Propriété ; élément de bibliothèque (voir [Objet SoundItem](#)) utilisé pour créer un son. Le son est associé directement à l'image.

### Exemple

L'exemple suivant affecte le premier élément de la bibliothèque à la propriété `soundLibraryItem` de la première image :

```
// The first item in the library must be a sound object.  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLibraryItem  
=fl.getDocumentDOM().library.items[0];
```

## frame.soundLoop

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.soundLoop`

### Description

Propriété ; valeur entière qui spécifie le nombre de lecture d'un son joint directement à une image (`frame.soundLibraryItem`). Si vous souhaitez spécifier une valeur pour cette propriété, définissez `frame.soundLoopMode` sur "repeat".

### Exemple

Voir `frame.soundLoopMode`.

## frame.soundLoopMode

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.soundLoopMode`

### Description

Propriété ; chaîne qui spécifie si un son joint directement à une image (`frame.soundLibraryItem`) doit être lu un certain nombre de fois ou en boucle infinie. Les valeurs gérées sont "repeat" (répétition) ou "loop" (en boucle). Pour spécifier le nombre de lectures du son, attribuez une valeur à `frame.soundLoop`.

### Exemple

L'exemple suivant indique qu'un son doit être lu deux fois :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLoopMode = "repeat";  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLoop = 2;
```

## frame.soundName

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.soundName`

### Description

Propriété ; chaîne qui spécifie le nom d'un son joint directement à une image (`frame.soundLibraryItem`), tel que stocké dans la bibliothèque.

### Exemple

L'exemple suivant donne à la propriété `soundName` de la première image la valeur "song1.mp3" ; le fichier song1.mp3 doit exister dans la bibliothèque :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundName = "song1.mp3";
```

## frame.soundSync

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.soundSync`

### Description

Propriété ; chaîne qui spécifie le comportement de synchronisation d'un son joint directement à une image ([frame.soundLibraryItem](#)). Les valeurs gérées sont les suivantes : "event" (événement), "stop", "start" et "stream" (flux).

### Exemple

L'exemple suivant indique qu'un son doit être lu en flux continu :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundSync = 'stream';
```

## frame.startFrame

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.startFrame`

### Description

Propriété en lecture seule ; index de la première image d'une séquence.

### Exemple

Dans l'exemple suivant, `stFrame` est l'index de la première image de la séquence. Dans cet exemple, une séquence d'images s'étend sur six images, de l'image 5 à l'image 10. La valeur de `stFrame` pour n'importe quelle image comprise entre l'image 5 et l'image 10 est donc 4 (n'oubliez pas que les valeurs d'index sont différentes des valeurs de numéro d'image).

```
var stFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[4].startFrame;  
fl.trace(stFrame); // 4  
var stFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[9].startFrame;  
fl.trace(stFrame); // 4
```

## frame.tweenEasing

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.tweenEasing`

### Description

Propriété ; entier qui spécifie la quantité d'accélération à appliquer à l'objet interpolé. Les valeurs gérées vont de -100 à 100. Pour commencer l'interpolation de mouvement lentement et l'accélérer vers la fin de l'animation, utilisez une valeur négative comprise entre -1 et -100. Pour commencer l'interpolation de mouvement rapidement et la ralentir vers la fin de l'animation, utilisez une valeur positive comprise entre 1 et 100.

### Exemple

L'exemple suivant indique que le mouvement de l'objet en cours d'interpolation doit débiter rapidement, puis décélérer vers la fin de l'animation :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].tweenEasing = 50;
```

## frame.tweenInstanceName

### Disponibilité

Flash Professional CS5

### Utilisation

`Frame.tweenInstanceName()`

### Description

Propriété ; chaîne qui attribue un nom d'occurrence à l'objet de mouvement sélectionné.

### Exemple

L'exemple suivant attribue le nom d'occurrence `MyMotionTween` à l'objet de mouvement spécifié.

```
theFrame.tweenInstanceName = "MyMotionTween";
```

## frame.tweenType

### Disponibilité

Flash MX 2004.

### Utilisation

`frame.tweenType`

### Description

Propriété ; chaîne qui spécifie le type d'interpolation. Les valeurs gérées sont les suivantes : "motion" (mouvement), "shape" (forme) ou "none" (aucune). La valeur "none" supprime l'interpolation de mouvement. Utilisez la méthode `timeline.createMotionTween()` pour créer une interpolation de mouvement.



Si vous spécifiez la valeur "motion", l'objet à interpoler doit être un symbole, un champ de texte ou un groupe d'objets. L'interpolation sera effectuée entre son emplacement dans l'image-clé actuelle et son emplacement dans la prochaine image-clé.

Si vous affectez la valeur "shape", l'objet à interpoler doit être une forme. L'interpolation sera effectuée entre la forme dans l'image-clé actuelle et la forme dans la prochaine image-clé.

### Exemple

L'exemple suivant spécifie que l'objet est une interpolation de mouvement. L'interpolation doit donc être effectuée entre son emplacement dans l'image-clé actuelle et son emplacement dans l'image-clé suivante :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].tweenType = "motion";
```

## frame.useSingleEaseCurve

### Disponibilité

Flash 8.

### Utilisation

```
frame.useSingleEaseCurve
```

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, la même courbe d'accélération personnalisée est utilisée pour les informations d'accélération de toutes les propriétés. Si elle est `false`, chaque propriété possède sa propre courbe d'accélération.

Cette propriété n'est pas prise en compte si l'accélération personnalisée n'est pas appliquée à l'image.

### Exemple

L'exemple suivant spécifie qu'une même courbe d'accélération personnalisée doit être utilisée pour toutes les propriétés de la première image du premier calque :

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
theFrame.useSingleEaseCurve = true;
```

### Voir aussi

[frame.getCustomEase\(\)](#), [frame.hasCustomEase](#), [frame.setCustomEase\(\)](#)

# Chapitre 22 : Objet halfEdge

## Disponibilité

Flash MX 2004.

## Description

L'objet halfEdge est le côté dirigé du contour d'un [Objet Shape](#). Un contour se compose de deux demi-contours. Il est possible de suivre les contours d'une forme en « parcourant » ces demi-contours. Par exemple, en partant d'un demi-contour, vous pouvez suivre tous les demi-contours du contour d'une forme, et revenir ainsi au demi-contour d'origine.

Les demi-contours sont ordonnés. Un demi-contour représente un côté du contour, et l'autre demi-contour en représente l'autre côté.

## Résumé des méthodes

L'objet HalfEdge gère les méthodes suivantes :

Méthode	Description
<code>halfEdge.getEdge()</code>	Méthode qui récupère l' <a href="#">Objet Edge</a> de l'objet HalfEdge.
<code>halfEdge.getNext()</code>	Récupère le prochain demi-contour du contour actif.
<code>halfEdge.getOppositeHalfEdge()</code>	Récupère l'objet HalfEdge situé de l'autre côté du contour.
<code>halfEdge.getPrev()</code>	Récupère l'objet HalfEdge précédent du contour actif.
<code>halfEdge.getVertex()</code>	Méthode qui récupère l'objet <a href="#">Objet Vertex</a> au début de l'objet HalfEdge.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet HalfEdge :

Propriété	Description
<code>halfEdge.id</code>	Lecture seule ; entier représentant un identifiant unique pour l'objet HalfEdge.
<code>halfEdge.index</code>	Entier prenant la valeur 0 ou 1 pour déterminer l'index de cet objet HalfEdge dans le contour parent.

## halfEdge.getEdge()

## Disponibilité

Flash MX 2004.

## Utilisation

```
halfEdge.getEdge()
```

## Paramètres

Aucun.

#### Valeur renvoyée

Un [Objet Edge](#).

#### Description

Méthode qui récupère l'objet Edge de l'objet HalfEdge. Voir [Objet Edge](#).

#### Exemple

L'exemple suivant montre comment obtenir un contour et un demi-contour pour la forme spécifiée :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var edge = hEdge.getEdge();
```

## halfEdge.getNext()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
halfEdge.getNext()
```

#### Paramètres

Aucun.

#### Valeur renvoyée

Un objet HalfEdge.

#### Description

Méthode qui récupère le demi-contour suivant du contour actif.

**Remarque :** *bien que les demi-contours possèdent des propriétés de direction et d'ordre séquentiel, ce n'est pas le cas des contours.*

#### Exemple

L'exemple suivant enregistre dans la variable nextHalfEdge le demi-contour suivant du contour spécifié :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge( 0 );  
var nextHalfEdge = hEdge.getNext();
```

## halfEdge.getOppositeHalfEdge()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
halfEdge.getOppositeHalfEdge()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un objet `HalfEdge`.

### Description

Méthode qui récupère l'objet `HalfEdge` situé de l'autre côté du contour.

### Exemple

L'exemple suivant enregistre dans la variable `otherHalfEdge` l'objet `hEdge` du demi-bord opposé :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var otherHalfEdge = hEdge.getOppositeHalfEdge();
```

## halfEdge.getPrev()

### Disponibilité

Flash MX 2004.

### Utilisation

```
halfEdge.getPrev()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un objet `HalfEdge`.

### Description

Méthode qui récupère l'objet `HalfEdge` précédent du contour actif.

**Remarque :** bien que les demi-contours possèdent des propriétés de direction et d'ordre séquentiel, ce n'est pas le cas des contours.

### Exemple

L'exemple suivant enregistre dans la variable `prevHalfEdge` le demi-contour précédent du contour spécifié :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge( 0 );  
var prevHalfEdge = hEdge.getPrev();
```

## halfEdge.getVertex()

### Disponibilité

Flash MX 2004.

### Utilisation

```
halfEdge.getVertex()
```

### Paramètres

Aucun.

### Valeur renvoyée

Un [Objet Vertex](#)

### Description

Méthode qui récupère l'objet Vertex au début de l'objet HalfEdge. Voir [Objet Vertex](#)

### Exemple

L'exemple suivant enregistre dans la variable `vertex` l'objet Vertex situé en tête de l'objet `hEdge` :

```
var shape = fl.getDocumentDOM().selection[0];  
var edge = shape.edges[0];  
var hEdge = edge.getHalfEdge(0);  
var vertex = hEdge.getVertex();
```

## halfEdge.id

### Disponibilité

Flash MX 2004.

### Utilisation

```
halfEdge.id
```

### Description

Propriété en lecture seule ; entier représentant un identifiant unique pour l'objet HalfEdge.

### Exemple

L'exemple suivant affiche dans le panneau Sortie un identificateur unique pour le demi-bord spécifié :

```
var shape = fl.getDocumentDOM().selection[0];  
alert(shape.contours[0].getHalfEdge().id);
```

## halfEdge.index

### Disponibilité

Flash MX 2004.

### Utilisation

`halfEdge.index`

### Description

Propriété en lecture seule ; entier possédant une valeur de 0 ou 1 qui détermine l'index de cet objet HalfEdge dans le contour parent.

### Exemple

L'exemple suivant affiche dans le panneau Sortie la valeur d'index pour le demi-bord spécifié :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var heIndex = hEdge.index;
```

# Chapitre 23 : Objet Instance

**Héritage**    [Objet Element](#) > Objet Instance

## Disponibilité

Flash MX 2004.

## Description

Instance est une sous-classe de l'[Objet Element](#).

## Résumé des propriétés

Outre toutes celles de l'objet Element, l'objet Instance possède les propriétés suivantes :

Propriété	Description
<a href="#">instance.instanceType</a>	Lecture seule ; chaîne représentant le type d'occurrence.
<a href="#">instance.libraryItem</a>	Élément de bibliothèque utilisé pour instancier cette occurrence.

## instance.instanceType

### Disponibilité

Flash MX 2004 ; valeur acceptée "video", ajoutée à Flash 8.

### Utilisation

```
instance.instanceType
```

### Description

Propriété en lecture seule ; chaîne représentant le type d'occurrence. Les valeurs possibles sont "symbol", "bitmap", "embedded video", "linked video", "video" et "compiled clip".

Dans Flash MX 2004, la valeur de `instance.instanceType` pour un élément ajouté à la bibliothèque à l'aide de `library.addItem("video")` est "embedded\_video". Dans Flash 8 et les versions ultérieures, la valeur est "video". Voir [library.addItem\(\)](#).

### Exemple

L'exemple suivant montre que le type d'occurrence d'un clip est symbol :

```
// Select a movie clip and then run this script.
var type = fl.getDocumentDOM().selection[0].instanceType;
fl.trace("This instance type is " + type);
```

## instance.libraryItem

### Disponibilité

Flash MX 2004.

### Utilisation

`instance.libraryItem`

### Description

Propriété ; élément de bibliothèque utilisé pour instancier cette occurrence. Il n'est possible de changer cette propriété qu'en la remplaçant par un autre objet de bibliothèque du même type (par exemple, vous ne pouvez pas faire référence à un bitmap avec une occurrence de type `symbol`). Voir [Objet library](#).

### Exemple

L'exemple suivant modifie le symbole sélectionné pour qu'il référence le premier élément de la bibliothèque :

```
fl.getDocumentDOM().selection[0].libraryItem = fl.getDocumentDOM().library.items[0];
```



# Chapitre 24 : Objet Item

## Disponibilité

Flash MX 2004.

## Description

L'objet Item est une classe de base abstraite. Tous les éléments de la bibliothèque dérivent de l'objet Item. Voir également [Objet library](#).

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet Item :

Méthode	Description
<code>item.addData()</code>	Ajoute les données spécifiées à un élément de la bibliothèque.
<code>item.getData()</code>	Récupère la valeur des données spécifiées.
<code>item.hasData()</code>	Détermine si l'élément de bibliothèque comporte les données indiquées.
<code>item.removeData()</code>	Supprime les données persistantes de l'élément de bibliothèque.

## Résumé des propriétés

Les propriétés suivantes sont disponibles avec l'objet Item :

Propriété	Description
<code>item.itemType</code>	Lecture seule ; chaîne indiquant le type d'élément.
<code>item.linkageBaseClass</code>	Chaîne spécifiant la classe ActionScript 3.0 qui sera associée au symbole.
<code>item.linkageClassName</code>	Chaîne spécifiant la classe ActionScript 2.0 qui sera associée au symbole.
<code>item.linkageExportForAS</code>	Valeur booléenne. Si sa valeur est <code>true</code> , l'élément est exporté pour le code ActionScript.
<code>item.linkageExportForRS</code>	Valeur booléenne. Si sa valeur est <code>true</code> , l'élément est exporté pour le partage à l'exécution.
<code>item.linkageExportInFirstFrame</code>	Valeur booléenne. Si sa valeur est <code>true</code> , l'élément est exporté dans la première image.
<code>item.linkageIdentifier</code>	Chaîne spécifiant le nom qui permettra à Flash d'identifier l'élément au moment de créer une liaison vers le fichier SWF de destination.
<code>item.linkageImportForRS</code>	Valeur booléenne. Si sa valeur est <code>true</code> , l'élément est importé pour le partage à l'exécution.
<code>item.linkageURL</code>	Chaîne qui spécifie l'URL à laquelle se trouve le fichier SWF contenant l'élément partagé.
<code>item.name</code>	Chaîne qui spécifie le nom de l'élément de bibliothèque, structure de l'arborescence comprise.

## item.addData()

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.addData(name, type, data)
```

### Paramètres

**name** Chaîne qui spécifie le nom des données.

**type** Chaîne spécifiant le type des données. Les types valides sont "integer", "integerArray", "double", "doubleArray", "string" et "byteArray".

**data** Données à ajouter à l'élément de bibliothèque spécifié. Le type de données dépend de la valeur du paramètre type. Par exemple, si type est "integer", la valeur des données doit être un entier, etc.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute les données spécifiées à un élément de la bibliothèque.

### Exemple

L'exemple suivant ajoute des données nommées myData, avec une valeur entière de 12, au premier élément de la bibliothèque :

```
fl.getDocumentDOM().library.items[0].addData("myData", "integer", 12);
```

## item.getData()

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.getData(name)
```

### Paramètres

**name** Chaîne spécifiant le nom des données à récupérer.

### Valeur renvoyée

Les données spécifiées par le paramètre *name*. Le type renvoyé dépend du type des données enregistrées.

### Description

Méthode qui récupère la valeur des données spécifiées.

### Exemple

L'exemple suivant récupère les valeurs des données nommées `myData` dans le premier élément de la bibliothèque et les enregistre dans la variable `libData` :

```
var libData = fl.getDocumentDOM().library.items[0].getData("myData");
```

## item.hasData()

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.hasData (name)
```

### Paramètres

**name** Chaîne spécifiant le nom des données à vérifier dans l'élément de bibliothèque.

### Valeur renvoyée

Valeur booléenne : `true` si les données spécifiées existent, `false` dans le cas contraire.

### Description

Méthode qui détermine si l'élément de bibliothèque comporte les données indiquées.

### Exemple

L'exemple suivant affiche un message dans le panneau Sortie si le premier élément de la bibliothèque contient des données nommées `myData` :

```
if (fl.getDocumentDOM().library.items[0].hasData("myData")) {  
    fl.trace("Yep, it's there!");  
}
```

## item.itemType

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.itemType
```

### Description

Propriété en lecture seule ; chaîne indiquant le type d'élément. Valeurs possibles : `"undefined"`, `"component"`, `"movie clip"`, `"graphic"`, `"button"`, `"folder"`, `"font"`, `"sound"`, `"bitmap"`, `"compiled clip"`, `"screen"` ou `"video"`. Si cette propriété est `"video"`, vous pouvez déterminer le type vidéo ; voir [videoItem.videoType](#).

### Exemple

L'exemple suivant affiche le type de l'élément de bibliothèque spécifié dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().library.items[0].itemType);
```

## item.linkageBaseClass

### Disponibilité

Flash CS3 Professional.

### Utilisation

```
item.linkageBaseClass
```

### Description

Propriété ; chaîne spécifiant la classe ActionScript 3.0 qui sera associée au symbole. La valeur spécifiée ici apparaît dans la boîte de dialogue Liaison de l'environnement de programmation, ainsi que dans toutes les autres boîtes de dialogue affichant les contrôles de cette option, par exemple la boîte de dialogue Propriétés du symbole. (Si vous voulez spécifier cette valeur pour une classe ActionScript 2.0, utilisez [item.linkageClassName](#).)

Si la classe de base est la classe par défaut pour le type de symbole (par exemple "flash.display.MovieClip" pour les clips, "flash.display.SimpleButton" pour les boutons, etc.), la valeur de cette propriété est une chaîne vide (""). De même, vous définirez cette valeur sur une chaîne vide pour qu'un élément devienne la classe de base par défaut.

Lorsque vous définissez cette valeur, aucune des vérifications prévues par la boîte de dialogue Liaison n'est effectuée, et aucune erreur n'est renvoyée si Flash est dans l'incapacité de définir la valeur de la classe de base sur celle qui est spécifiée. Par exemple, le fait de définir cette valeur dans la boîte de dialogue Liaison oblige le processus de vérification à contrôler la présence de la classe de base dans le chemin d'accès de classe du fichier FLA. Il s'assure qu'ActionScript 3.0 est choisi dans l'onglet Flash de la boîte de dialogue Paramètres de publication, etc. Ces vérifications ne sont pas effectuées lorsque vous définissez cette propriété dans un script.

### Exemple

Les lignes de code suivantes illustrent plusieurs façons d'utiliser cette propriété :

```
// sets the library item base class to "Sprite"
fl.getDocumentDOM().library.items[0].linkageBaseClass = "flash.display.Sprite";
// sets the library item base class to the default for that item type
fl.getDocumentDOM().library.items[0].linkageBaseClass = "";
// finds and displays the library item's base class
fl.trace(fl.getDocumentDOM().library.items[0].linkageBaseClass);
```

### Voir aussi

[document.docClass](#)

## item.linkageClassName

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.linkageClassName
```

### Description

Propriété ; chaîne spécifiant la classe ActionScript 2.0 qui sera associée au symbole. (Si vous voulez spécifier cette valeur pour une classe ActionScript 3.0, utilisez `item.linkageBaseClass`.)

Pour définir cette propriété, les propriétés `item.linkageExportForAS` et/ou `item.linkageExportForRS` doivent être définies sur `true` et la propriété `item.linkageImportForRS` doit être définie sur `false`.

### Exemple

L'exemple suivant spécifie que le nom de classe ActionScript 2.0 associé au premier élément de la bibliothèque est `myClass` :

```
fl.getDocumentDOM().library.items[0].linkageClassName = "myClass";
```

## item.linkageExportForAS

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.linkageExportForAS
```

### Description

Propriété : valeur booléenne. Si cette propriété a la valeur `true`, l'élément est exporté pour le code ActionScript. Vous pouvez également définir les propriétés `item.linkageExportForRS` et `item.linkageExportInFirstFrame` sur `true`.

Si vous définissez cette propriété sur `true`, la propriété `item.linkageImportForRS` doit être définie sur `false`. Par ailleurs, vous devez spécifier un identificateur (`item.linkageIdentifier`) et une URL (`item.linkageURL`).

### Exemple

L'exemple suivant définit cette propriété pour l'élément de bibliothèque spécifié :

```
fl.getDocumentDOM().library.items[0].linkageExportForAS = true;
```

## item.linkageExportForRS

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.linkageExportForRS
```

### Description

Propriété : valeur booléenne. Si cette propriété a la valeur `true`, l'élément est exporté pour le partage à l'exécution. Vous pouvez également définir les propriétés `item.linkageExportForAS` et `item.linkageExportInFirstFrame` sur `true`.

Si vous définissez cette propriété sur `true`, la propriété `item.linkageImportForRS` doit être définie sur `false`. Par ailleurs, vous devez spécifier un identificateur (`item.linkageIdentifier`) et une URL (`item.linkageURL`).

### Exemple

L'exemple suivant définit cette propriété pour l'élément de bibliothèque spécifié :

```
fl.getDocumentDOM().library.items[0].linkageExportForRS = true;
```

## item.linkageExportInFirstFrame

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.linkageExportInFirstFrame
```

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, l'élément est exporté dans la première image ; si elle est `false`, l'élément est exporté dans l'image de la première occurrence. Si l'élément n'apparaît pas sur la scène, il n'est pas exporté.

Cette propriété peut être définie sur `true` uniquement si les propriétés `item.linkageExportForAS` et/ou `item.linkageExportForRS` sont définies sur `true`.

### Exemple

L'exemple suivant spécifie que l'élément de bibliothèque spécifié doit être exporté dans la première image :

```
fl.getDocumentDOM().library.items[0].linkageExportInFirstFrame = true;
```

## item.linkageIdentifier

### Disponibilité

Flash MX 2004.

### Utilisation

```
item.linkageIdentifier
```

### Description

Propriété ; chaîne spécifiant le nom qui permettra à Flash d'identifier l'élément au moment de créer une liaison vers le fichier SWF de destination. Flash ignore cette propriété si les propriétés `item.linkageImportForRS`, `item.linkageExportForAS` et `item.linkageExportForRS` sont définies sur `false`. À l'inverse, cette propriété doit être définie si l'une de ces propriétés est définie sur `true`.

### Exemple

L'exemple suivant spécifie que la chaîne `my_mc` sera utilisée pour identifier l'élément de bibliothèque au moment de le lier au fichier SWF vers lequel il sera exporté :

```
fl.getDocumentDOM().library.items[0].linkageIdentifier = "my_mc";
```

#### Voir aussi

[item.linkageURL](#)

## item.linkageImportForRS

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
item.linkageImportForRS
```

#### Description

Propriété ; valeur booléenne, si `true`, l'élément est importé pour le partage à l'exécution. Si cette propriété est définie sur `true`, les deux propriétés [item.linkageExportForAS](#) et [item.linkageExportForRS](#) doivent être définies sur `false`. Par ailleurs, vous devez spécifier un identificateur ([item.linkageIdentifier](#)) et une URL ([item.linkageURL](#)).

#### Exemple

L'exemple suivant définit cette propriété sur `true` pour l'élément de bibliothèque spécifié :

```
fl.getDocumentDOM().library.items[0].linkageImportForRS = true;
```

## item.linkageURL

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
item.linkageURL
```

#### Description

Propriété ; chaîne qui spécifie l'URL à laquelle se trouve le fichier SWF contenant l'élément partagé. Flash ignore cette propriété si les propriétés [item.linkageImportForRS](#), [item.linkageExportForAS](#) et [item.linkageExportForRS](#) sont définies sur `false`. A l'inverse, cette propriété doit être définie si l'une de ces propriétés est définie sur `true`. Il est possible d'indiquer une adresse de type URL ou un nom de fichier au format d'une plate-forme précise (c'est-à-dire avec une barre oblique normale (/) ou inversée (\), selon la plate-forme).

#### Exemple

L'exemple suivant spécifie une adresse URL comme lien vers l'élément de bibliothèque spécifié :

```
fl.getDocumentDOM().library.items[0].linkageURL = "theShareSWF.swf";
```

#### Voir aussi

[item.linkageIdentifier](#)

## item.name

### Disponibilité

Flash MX 2004.

### Utilisation

`item.name`

### Description

Méthode (chaîne) qui spécifie le nom de l'élément de bibliothèque, structure de l'arborescence comprise. Par exemple, si Symbol\_1 se trouve à l'intérieur d'un dossier nommé Folder\_1, la propriété `name` de Symbol\_1 reçoit la valeur "Folder\_1/Symbol\_1".

### Exemple

L'exemple suivant affiche le nom de l'élément de bibliothèque spécifié dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().library.items[0].name);
```

## item.removeData()

### Disponibilité

Flash MX 2004.

### Utilisation

`item.removeData(name)`

### Paramètres

**name** Chaîne spécifiant le nom des données à supprimer de l'élément de bibliothèque.

### Valeur renvoyée

Aucune.

### Description

Propriété ; supprime les données persistantes de l'élément de bibliothèque.

### Exemple

L'exemple suivant supprime les données nommées `myData` du premier élément de la bibliothèque :

```
fl.getDocumentDOM().library.items[0].removeData("myData");
```



# Chapitre 25 : Objet Layer

## Disponibilité

Flash MX 2004.

## Description

L'objet Layer représente un calque dans le scénario. La propriété `timeline.layers` contient un tableau d'objets Layer, accessible par le biais de `fl.getDocumentDOM().getTimeline().layers`.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Layer :

Propriété	Description
<code>layer.color</code>	Chaîne, valeur hexadécimale ou entier qui spécifie la couleur utilisée pour indiquer les contours du calque.
<code>layer.frameCount</code>	Lecture seule ; entier précisant le nombre d'images du calque.
<code>layer.frames</code>	Lecture seule ; tableau d'objets Frame.
<code>layer.height</code>	Entier qui spécifie la hauteur du calque en pourcentage ; équivalent au champ Hauteur du calque de la boîte de dialogue Propriétés du calque.
<code>layer.layerType</code>	Chaîne qui spécifie l'utilisation actuelle du calque ; équivalent du champ Type dans la boîte de dialogue Propriétés du calque.
<code>layer.locked</code>	Valeur booléenne spécifiant l'état de verrouillage du calque.
<code>layer.name</code>	Chaîne qui spécifie le nom du calque.
<code>layer.outline</code>	Valeur booléenne spécifiant l'état des contours de tous les objets du calque.
<code>layer.parentLayer</code>	Objet Layer qui représente l'objet parent du calque (dossier de calques, calque de guidage ou calque de masquage).
<code>layer.visible</code>	Valeur booléenne indiquant si les objets du calque sur la scène sont affichés ou masqués.

## layer.color

## Disponibilité

Flash MX 2004.

## Utilisation

`layer.color`

## Description

Propriété ; couleur à utiliser pour délimiter le calque, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB

- Entier représentant l'équivalent décimal d'un nombre hexadécimal

Cette propriété est l'équivalent du champ Couleur de contour dans la boîte de dialogue Propriétés du calque.

#### **Exemple**

L'exemple suivant enregistre dans la variable `colorValue` la valeur du premier calque :

```
var colorValue = fl.getDocumentDOM().getTimeline().layers[0].color;
```

L'exemple suivant présente les trois façons de définir sur « rouge » la couleur du premier calque :

```
fl.getDocumentDOM().getTimeline().layers[0].color=16711680;  
fl.getDocumentDOM().getTimeline().layers[0].color="#ff0000";  
fl.getDocumentDOM().getTimeline().layers[0].color=0xFF0000;
```

## **layer.frameCount**

#### **Disponibilité**

Flash MX 2004.

#### **Utilisation**

`layer.frameCount`

#### **Description**

Propriété en lecture seule ; entier indiquant le nombre d'images du calque.

#### **Exemple**

L'exemple suivant enregistre dans la variable `fcNum` le nombre d'images du premier calque :

```
var fcNum = fl.getDocumentDOM().getTimeline().layers[0].frameCount;
```

## **layer.frames**

#### **Disponibilité**

Flash MX 2004.

#### **Utilisation**

`layer.frames`

#### **Description**

Propriété en lecture seule ; tableau d'objets Frame (voir [Objet Frame](#)).

#### **Exemple**

L'exemple suivant copie dans la variable `frameArray` le tableau d'objets Frame représentant les images du document actif :

```
var frameArray = fl.getDocumentDOM().getTimeline().layers[0].frames;
```

Pour savoir si une image est une image-clé, vérifiez si la propriété `frame.startFrame` correspond à l'index du tableau, comme l'illustre l'exemple suivant :

```
var frameArray = fl.getDocumentDOM().getTimeline().layers[0].frames;
var n = frameArray.length;
for (i=0; i<n; i++) {
    if (i==frameArray[i].startFrame) {
        alert("Keyframe at: " + i);
    }
}
```

## layer.height

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.height`

### Description

Propriété ; entier qui spécifie la hauteur du calque en pourcentage ; équivalent au champ Hauteur du calque de la boîte de dialogue Propriétés du calque. Les valeurs possibles représentent un pourcentage de la hauteur par défaut : 100, 200 ou 300.

### Exemple

L'exemple suivant enregistre la valeur de pourcentage de hauteur du premier calque :

```
var layerHeight = fl.getDocumentDOM().getTimeline().layers[0].height;
```

L'exemple suivant définit la hauteur du premier calque sur 300% :

```
fl.getDocumentDOM().getTimeline().layers[0].height = 300;
```

## layer.layerType

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.layerType`

### Description

Propriété ; chaîne qui spécifie l'utilisation actuelle du calque ; équivalent au champ Type dans la boîte de dialogue Propriétés du calque. Les valeurs gérées sont : "normal", "guide", "guided", "mask", "masked" et "folder".

### Exemple

L'exemple suivant définit le premier calque du scénario sur le type `folder` :

```
fl.getDocumentDOM().getTimeline().layers[0].layerType = "folder";
```

## layer.locked

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.locked`

### Description

Propriété ; valeur booléenne spécifiant l'état de verrouillage du calque. Si cette propriété est `true`, le calque est verrouillé. La valeur par défaut est `false`.

### Exemple

L'exemple suivant enregistre dans la variable `lockStatus` la valeur booléenne d'état de verrouillage du premier calque :

```
var lockStatus = fl.getDocumentDOM().getTimeline().layers[0].locked;
```

L'exemple suivant définit l'état du premier calque sur déverrouillé :

```
fl.getDocumentDOM().getTimeline().layers[0].locked = false;
```

## layer.name

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.name`

### Description

Propriété ; chaîne qui spécifie le nom du calque.

### Exemple

L'exemple suivant définit sur `foreground` le nom du premier calque du document actuel :

```
fl.getDocumentDOM().getTimeline().layers[0].name = "foreground";
```

## layer.outline

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.outline`

### Description

Propriété ; valeur booléenne spécifiant l'état des contours de tous les objets du calque. Si cette propriété est définie sur `true`, seuls les contours des objets du calque sont affichés. Si elle est définie sur `false`, les objets apparaissent tels qu'ils ont été créés.

### Exemple

L'exemple suivant fait afficher uniquement les contours des objets du premier calque :

```
fl.getDocumentDOM().getTimeline().layers[0].outline = true;
```

## layer.parentLayer

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.parentLayer`

### Description

Propriété ; objet Layer qui représente l'objet parent du calque (dossier de calques, calque de guidage ou calque de masquage). Le calque parent doit être un dossier conteneur du calque, un calque de guide ou de masque précédant le calque dans la hiérarchie, ou le calque parent (`parentLayer`) du calque précédent ou suivant. Le paramétrage de l'objet `parentLayer` du calque ne modifie pas la position du calque dans la liste. De ce fait, tenter de définir le `parentLayer` d'un calque sur un calque qui nécessiterait son déplacement est sans effet. Cette propriété utilise la valeur `null` pour un calque de niveau supérieur.

### Exemple

L'exemple suivant utilise deux calques de même niveau dans le même scénario. Le premier calque (`layers[0]`) est converti en dossier, puis défini comme dossier parent du second calque (`layers[1]`). Cette action place le second calque à l'intérieur du premier.

```
var parLayer = fl.getDocumentDOM().getTimeline().layers[0];
parLayer.layerType = "folder";
fl.getDocumentDOM().getTimeline().layers[1].parentLayer = parLayer;
```

## layer.visible

### Disponibilité

Flash MX 2004.

### Utilisation

`layer.visible`

### **Description**

Propriété ; valeur booléenne indiquant si les objets du calque sur la scène sont affichés ou masqués. Si la valeur de cette propriété est `true`, tous les objets du calque sont visibles ; si elle est `false`, ils sont masqués. La valeur par défaut est `true`.

### **Exemple**

L'exemple suivant rend les objets du premier calque invisibles :

```
fl.getDocumentDOM().getTimeline().layers[0].visible = false;
```

# Chapitre 26 : Objet library

## Disponibilité

Flash MX 2004.

## Description

L'objet `library` représente le panneau Bibliothèque. Il s'agit d'une propriété de l'objet `Document` (voir `document.library`) à laquelle vous pouvez accéder par `fl.getDocumentDOM().library`.

L'objet `library` contient un tableau d'éléments de différents types (symboles, bitmaps, sons et vidéo).

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet `library` :

Méthode	Description
<code>library.addItemToDocument()</code>	Ajoute l'élément actif ou un élément spécifié à la scène, à l'emplacement spécifié.
<code>library.addNewItem()</code>	Crée dans le panneau Bibliothèque un nouvel élément du type spécifié, et définit ce nouvel élément sur l'élément sélectionné.
<code>library.deleteItem()</code>	Supprime du panneau Bibliothèque les éléments actifs ou un élément spécifié.
<code>library.duplicateItem()</code>	Copie l'élément sélectionné ou un élément spécifié.
<code>library.editItem()</code>	Ouvre l'élément sélectionné ou un élément spécifié en mode de modification.
<code>library.expandFolder()</code>	Développe ou réduit le dossier sélectionné ou un dossier spécifié dans la bibliothèque.
<code>library.findItemIndex()</code>	Renvoie l'index (de base zéro) de l'élément dans la bibliothèque.
<code>library.getItemProperty()</code>	Récupère la propriété de l'objet sélectionné.
<code>library.getItemType()</code>	Récupère le type de l'objet sélectionné ou d'un objet spécifié par son chemin d'accès dans la bibliothèque.
<code>library.getSelectedItems()</code>	Récupère le tableau de tous les éléments sélectionnés dans la bibliothèque.
<code>library.importEmbeddedSWF()</code>	Importe un fichier SWF dans la bibliothèque, sous forme de clip compilé.
<code>library.itemExists()</code>	Vérifie l'existence de l'élément spécifié dans la bibliothèque.
<code>library.moveToFolder()</code>	Déplace l'élément sélectionné ou un élément de bibliothèque spécifié dans un dossier spécifié.
<code>library.newFolder()</code>	Crée un nouveau dossier portant le nom spécifié (ou le nom par défaut "untitled folder #") si le paramètre <code>folderName</code> n'est pas fourni) dans le dossier sélectionné.
<code>library.renameItem()</code>	Renomme l'élément de bibliothèque sélectionné dans le panneau Bibliothèque.
<code>library.selectAll()</code>	Sélectionne ou désélectionne tous les éléments de la bibliothèque.
<code>library.selectItem()</code>	Sélectionne l'élément de bibliothèque spécifié.

Méthode	Description
<code>library.selectNone()</code>	Retire tous les éléments de bibliothèque de la sélection.
<code>library.setItemProperty()</code>	Définit la propriété de tous les objets de bibliothèque sélectionnés (en ignorant les dossiers).
<code>library.updateItem()</code>	Actualise l'élément de bibliothèque spécifié.

### Résumé des propriétés de l'objet **library**

La propriété suivante est disponible avec l'objet **Library** :

Propriété	Description
<code>library.items</code>	Tableau d'objets <b>Item</b> de la bibliothèque

## **library.addItemToDocument()**

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.addItemToDocument(position [, namePath])
```

### Paramètres

**position** Point qui indique la position *x,y* du centre de l'élément sur la scène.

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Si *namePath* n'est pas spécifié, la sélection actuelle dans la bibliothèque est utilisée. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : *true* si l'élément est bien ajouté au document, *false* en cas d'échec.

### Description

Méthode qui ajoute l'élément actif ou un élément spécifié à la scène, à l'emplacement spécifié.

### Exemple

L'exemple suivant ajoute l'élément sélectionné ou un élément spécifié sur la scène, à la position (3,60).

```
fl.getDocumentDOM().library.addItemToDocument({x:3, y:60});
```

L'exemple suivant ajoute sur la scène l'élément `Symbol1` (qui se trouve dans le dossier `folder1` de la bibliothèque) à la position (550, 485) :

```
fl.getDocumentDOM().library.addItemToDocument({x:550.0, y:485.0}, "folder1/Symbol1");
```



## library.addNewItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.addNewItem(type [, namePath])
```

### Paramètres

**type** Chaîne spécifiant le type d'élément à créer. Les seules valeurs gérées pour *type* sont "video", "movie clip", "button", "graphic", "bitmap", "screen" et "folder" (cette méthode ne permet donc pas d'ajouter un son à la bibliothèque). Spécifier un chemin de dossier revient à utiliser `library.newFolder()` avant d'appeler cette méthode.

**namePath** Chaîne spécifiant le nom de l'élément à ajouter. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si l'élément est bien créé, `false` dans le cas contraire.

### Description

Méthode qui crée dans le panneau Bibliothèque un nouvel élément du type spécifié et active ce nouvel élément. Pour plus d'informations sur l'importation d'éléments dans la bibliothèque, notamment de sons, voir `document.importFile()`.

### Exemple

L'exemple suivant crée un nouvel élément de bouton nommé `start` dans un nouveau dossier nommé `folderTwo` :

```
fl.getDocumentDOM().library.addNewItem("button", "folderTwo/start");
```

## library.deleteItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.deleteItem([namePath])
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément à supprimer. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Si vous transmettez le nom d'un dossier, ce dernier et tout son contenu sont supprimés. Si aucun nom n'est spécifié, Flash supprime le ou les éléments actuellement sélectionnés. Pour supprimer tous les éléments du panneau Bibliothèque, sélectionnez tous les éléments avant d'utiliser cette méthode. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si les éléments sont bien supprimés, `false` dans le cas contraire.

### Description

Méthode qui supprime du panneau Bibliothèque les éléments actifs ou un élément spécifié. Si plusieurs éléments sont sélectionnés, cette méthode les supprime tous.

### Exemple

L'exemple suivant supprime l'élément sélectionné :

```
fl.getDocumentDOM().library.deleteItem();
```

L'exemple suivant supprime l'élément `Symbol_1` du dossier `Folder_1` de la bibliothèque :

```
fl.getDocumentDOM().library.deleteItem("Folder_1/Symbol_1");
```

## library.duplicateItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.duplicateItem( [ namePath ] )
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément à dupliquer. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si l'élément est bien dupliqué, `false` en cas d'échec. Si vous avez sélectionné plusieurs objets, Flash renvoie `false`.

### Description

Méthode qui copie l'élément sélectionné ou un élément spécifié. Le nouvel élément reçoit un nom par défaut (par exemple `item copy`) et devient l'élément sélectionné. Si vous avez sélectionné plusieurs éléments, la commande échoue.

### Exemple

L'exemple suivant crée une copie de l'élément `square` dans le dossier `test` de la bibliothèque:

```
fl.getDocumentDOM().library.duplicateItem("test/square");
```

## library.editItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.editItem( [namePath] )
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Si *namePath* n'est pas spécifié, l'élément de bibliothèque sélectionné s'ouvre en mode de modification. Si aucun élément n'est sélectionné, ou si au contraire plusieurs éléments sont sélectionnés, la première séquence du scénario principal est ouverte en mode de modification. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si l'élément spécifié existe et peut être modifié, `false` dans le cas contraire.

### Description

Méthode qui ouvre l'élément sélectionné ou un élément spécifié en mode de modification.

### Exemple

L'exemple suivant ouvre pour modification l'élément `circle`, qui se trouve dans le dossier `test` de la bibliothèque :

```
fl.getDocumentDOM().library.editItem("test/circle");
```

## library.expandFolder()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.expandFolder(bExpand [, bRecurseNestedParents [, namePath]])
```

### Paramètres

**bExpand** Valeur booléenne : si `true`, le dossier est développé ; si `false` (par défaut), le dossier est réduit.

**bRecurseNestedParents** Valeur booléenne : si `true`, tous les dossiers présents dans le dossier spécifié sont développés ou réduits, selon la valeur de *bExpand*. La valeur par défaut est `false`. Ce paramètre est facultatif.

**namePath** Chaîne spécifiant le nom et, éventuellement, le chemin d'accès du dossier à développer ou à réduire. Si ce paramètre n'est pas spécifié, la méthode s'applique au dossier actuellement sélectionné. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si l'élément est bien développé ou réduit ; `false` en cas d'échec ou si l'élément spécifié n'est pas un dossier.

### Description

Méthode qui développe ou réduit le dossier actif ou un dossier spécifié dans la bibliothèque.

### Exemple

L'exemple suivant réduit le dossier « test », ainsi que tous ses sous-dossiers, le cas échéant :

```
fl.getDocumentDOM().library.expandFolder(false, true, "test");
```

## library.findItemIndex()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.findItemIndex(namePath)
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique.

### Valeur renvoyée

Nombre entier représentant l'index (de base zéro) de l'élément dans la bibliothèque.

### Description

Méthode qui renvoie l'index (de base zéro) de l'élément dans la bibliothèque. L'index de la bibliothèque possède une seule dimension, si bien que les dossiers font partie de l'indexation. Il est possible d'utiliser les chemins d'accès des dossiers pour spécifier un élément imbriqué.

### Exemple

L'exemple suivant enregistre dans la variable `sqIndex` la valeur d'index (de base zéro) de l'élément de bibliothèque "square" qui se trouve dans le dossier "test", puis affiche cette valeur dans une boîte de dialogue :

```
var sqIndex = fl.getDocumentDOM().library.findItemIndex("test/square");  
alert(sqIndex);
```

## library.getItemProperty()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.getItemProperty(property)
```

### Paramètres

**property** Chaîne. Pour obtenir la liste des valeurs utilisables comme paramètre *property*, voir le Résumé des propriétés de l'objet [Objet Item](#), ainsi que les propriétés disponibles pour ses sous-classes.

### Valeur renvoyée

La valeur de chaîne de la propriété.

### Description

Méthode qui récupère la propriété de l'élément sélectionné.

### Exemple

L'exemple suivant affiche dans une boîte de dialogue la valeur de l'identifiant de liaison du symbole tel qu'il est référencé en code ActionScript ou pour le partage à l'exécution :

```
alert (fl.getDocumentDOM().library.getItemProperty("linkageIdentifier")) ;
```

## library.getItemType()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.getItemType ( [namePath] )
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Si *namePath* n'est pas spécifié, Flash renvoie le type de la sélection actuelle. Si plusieurs éléments sont sélectionnés et si le paramètre *namePath* n'est pas spécifié, Flash ignore la commande. Ce paramètre est facultatif.

### Valeur renvoyée

Une valeur de chaîne qui spécifie le type de l'objet. Pour obtenir la liste des valeurs pouvant être renvoyées, voir [item.itemType](#).

### Description

Méthode qui récupère le type de l'objet sélectionné ou spécifié par son chemin d'accès à la bibliothèque.

### Exemple

L'exemple suivant affiche dans une boîte de dialogue le type de l'élément `Symbol_1`, qui se trouve dans le dossier `Folder_1/Folder_2` :

```
alert (fl.getDocumentDOM().library.getItemType ("Folder_1/Folder_2/Symbol_1")) ;
```

## library.getSelectedItems()

### Disponibilité

Flash MX 2004.

### Paramètres

Aucun.

### Valeur renvoyée

Le tableau des valeurs de tous les éléments sélectionnés dans la bibliothèque.

### Description

Méthode qui récupère le tableau de tous les éléments sélectionnés dans la bibliothèque.

### Exemple

L'exemple suivant stocke dans la variable `selItems` le tableau des éléments de la bibliothèque actuellement sélectionnés (ici, plusieurs fichiers audio), puis donne à la propriété `sampleRate` du premier fichier audio du tableau la valeur 11 kHz :

```
var selItems = fl.getDocumentDOM().library.getSelectedItems();  
selItems[0].sampleRate = "11 kHz";
```

## library.importEmbeddedSWF()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.importEmbeddedSWF(linkageName, swfData [, libName])
```

### Paramètres

**linkageName** Chaîne indiquant le nom de la liaison SWF du clip racine.

**swfData** Tableau de données SWF binaires, qui provient d'une bibliothèque externe ou d'une DLL.

**libName** Chaîne spécifiant le nom, dans la bibliothèque, de l'élément créé. Si ce nom est déjà utilisé, la méthode en crée un autre. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui importe un fichier SWF dans la bibliothèque sous forme de clip compilé. À la différence de l'option de menu Fichier > Importer > SWF, cette méthode permet d'incorporer dans la bibliothèque un fichier SWF compilé. Il n'existe dans l'interface utilisateur aucune fonctionnalité correspondante, et cette méthode doit être utilisée avec une bibliothèque externe ou une DLL (voir « [Extensibilité de niveau C](#) » à la page 535).

Le fichier SWF que vous importez doit posséder un clip de niveau supérieur détenant la totalité du contenu. L'identificateur de liaison de ce clip doit être défini sur la même valeur que le paramètre *linkageName* transmis à cette méthode.

### Exemple

L'exemple suivant ajoute à la bibliothèque le fichier SWF ayant la valeur *MyMovie* pour la propriété `linkageName`, sous forme d'un clip compilé nommé *Intro* :

```
fl.getDocumentDOM().library.importEmbeddedSWF("MyMovie", swfData, "Intro");
```

## library.itemExists()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.itemExists(namePath)
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique.

### Valeur renvoyée

Valeur booléenne : `true` si l'élément spécifié est présent dans la bibliothèque ; `false` dans le cas contraire.

### Description

Méthode qui vérifie l'existence de l'élément spécifié dans la bibliothèque.

### Exemple

L'exemple suivant affiche `true` ou `false` dans une boîte de dialogue, selon que l'élément `Symbol_1` existe ou non dans le dossier `Folder_1` de la bibliothèque :

```
alert(fl.getDocumentDOM().library.itemExists('Folder_1/Symbol_1'));
```

## library.items

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.items
```

### Description

Propriété ; tableau des éléments sélectionnés dans la bibliothèque.

### Exemple

L'exemple suivant enregistre dans la variable `itemArray` le tableau de tous les éléments contenus dans la bibliothèque :

```
var itemArray = fl.getDocumentDOM().library.items;
```

## library.moveToFolder()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.moveToFolder(folderPath [, itemToMove [, bReplace]])
```

### Paramètres

**folderPath** Chaîne indiquant le chemin d'accès du dossier sous la forme "nom\_de\_dossier" ou "nom\_de\_dossier1/nom\_de\_dossier2". Pour déplacer un élément vers le niveau supérieur, transmettez une chaîne vide ("") pour le paramètre *folderPath*.

**itemToMove** Chaîne spécifiant le nom de l'élément à déplacer. Si *itemToMove* n'est pas spécifié, les éléments actuellement sélectionnés sont déplacés. Ce paramètre est facultatif.

**bReplace** Valeur booléenne. S'il existe déjà un élément du même nom, le fait de spécifier `true` pour le paramètre *bReplace* remplace l'élément existant par l'élément déplacé. Si la valeur de ce paramètre est `false`, le nom de l'élément déplacé est modifié pour être unique. La valeur par défaut est `false`. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si l'élément est bien déplacé, `false` en cas d'échec.

### Description

Méthode qui déplace l'élément sélectionné ou un élément de bibliothèque spécifié dans un dossier spécifié. Si le paramètre *folderPath* est vide, les éléments sont déplacés vers le niveau supérieur.

### Exemple

L'exemple suivant déplace l'élément `Symbol_1` dans le dossier `new` en remplacement de l'élément qui porte déjà ce nom dans ce dossier :

```
fl.getDocumentDOM().library.moveToFolder("new", "Symbol_1", true);
```

## library.newFolder()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.newFolder([folderPath])
```

### Paramètres

**folderPath** Chaîne spécifiant le nom du dossier à créer. Si cette chaîne contient un chemin et que ce chemin n'existe pas, il est alors créé. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si le dossier est bien créé, `false` en cas d'échec.

### Description

Méthode qui crée un nouveau dossier ayant le nom spécifié (ou le nom par défaut "untitled folder #") si le paramètre *folderName* n'est pas transmis) dans le dossier sélectionné.

### Exemple

L'exemple suivant crée deux nouveaux dossiers de bibliothèque. Le deuxième est un sous-dossier du premier :

```
fl.getDocumentDOM().library.newFolder("first/second");
```



## library.renameItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.renameItem(name)
```

### Paramètres

**name** Chaîne spécifiant le nouveau nom de l'élément de bibliothèque.

### Valeur renvoyée

Valeur booléenne : `true` si le nom de l'élément est bien modifié, `false` dans le cas contraire. Si plusieurs éléments sont sélectionnés, aucun nom n'est modifié et la valeur renvoyée correspond à `false` (comme dans l'interface utilisateur).

### Description

Méthode qui renomme l'élément de bibliothèque sélectionné dans le panneau Bibliothèque.

### Exemple

L'exemple suivant renomme avec le nom "new name" l'élément de bibliothèque actuellement sélectionné :

```
fl.getDocumentDOM().library.renameItem("new name");
```

## library.selectAll()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.selectAll([bSelectAll])
```

### Paramètres

**bSelectAll** Valeur booléenne qui indique si tous les éléments de la bibliothèque doivent être sélectionnés ou désélectionnés. Pour sélectionner tous les éléments de la bibliothèque, omettez ce paramètre ou utilisez la valeur par défaut (`true`) ; la valeur `false` désélectionne tous les éléments. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui sélectionne ou désélectionne tous les éléments de la bibliothèque.

### Exemple

Les exemples suivants sélectionnent tous les éléments de la bibliothèque :

```
fl.getDocumentDOM().library.selectAll();  
fl.getDocumentDOM().library.selectAll(true);
```

Les exemples suivants désélectionnent tous les éléments de la bibliothèque :

```
fl.getDocumentDOM().library.selectAll(false);  
fl.getDocumentDOM().library.selectNone();
```

## library.selectItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.selectItem(namePath [, bReplaceCurrentSelection [, bSelect]])
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique.

**bReplaceCurrentSelection** Valeur booléenne qui indique si le programme doit remplacer la sélection actuelle ou ajouter l'élément à cette sélection. La valeur par défaut est `true` (remplacer la sélection actuelle). Ce paramètre est facultatif.

**bSelect** Valeur booléenne qui indique si l'élément doit être sélectionné ou désélectionné. La valeur par défaut est `true` (sélectionner). Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si l'élément spécifié existe, `false` dans le cas contraire.

### Description

Méthode qui sélectionne l'élément de bibliothèque spécifié.

### Exemple

L'exemple suivant modifie la sélection actuelle dans la bibliothèque en `Symbol_1` au sein du dossier sans titre `Folder_1` :

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1");
```

L'exemple suivant étend la sélection actuelle dans la bibliothèque pour inclure `Symbol_1` au sein du dossier sans titre `Folder_1` :

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1", false);
```

L'exemple suivant désélectionne `Symbol_1` au sein du dossier sans titre `Folder_1`, sans changer les autres éléments sélectionnés :

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1", true, false);
```

## library.selectNone()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.selectNone()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui désélectionne tous les éléments de la bibliothèque.

### Exemple

Les exemples suivants désélectionnent tous les éléments de la bibliothèque :

```
fl.getDocumentDOM().library.selectNone();  
fl.getDocumentDOM().library.selectAll(false);
```

## library.setItemProperty()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.setItemProperty(property, value)
```

### Paramètres

**property** Chaîne indiquant le nom de la propriété à définir. Pour obtenir la liste des propriétés, voir le Résumé des propriétés de l'objet [Objet Item](#) et le résumé des propriétés pour ses sous-classes. Pour savoir quels objets sont des sous-classes de l'objet Item, voir « [Résumé de la structure du DOM](#) » à la page 11.

**value** Valeur à affecter à la propriété spécifiée.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la propriété de tous les objets de bibliothèque sélectionnés (en ignorant les dossiers).

### Exemple

L'exemple suivant affecte la valeur "button" à la propriété `symbolType` des éléments de bibliothèque sélectionnés. Dans ce cas, l'élément doit être un objet [Objet SymbolItem](#) ; `symbolType` étant une propriété valide pour les objets `SymbolItem`.

```
fl.getDocumentDOM().library.setItemProperty("symbolType", "button");
```

## library.updateItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
library.updateItem([namePath])
```

### Paramètres

**namePath** Chaîne spécifiant le nom de l'élément. Si l'élément se trouve dans un dossier, il est possible d'indiquer son nom et son chemin d'accès en utilisant la notation avec barre oblique. Cette opération est identique à un clic droit sur un élément, puis à la sélection de Mettre à jour dans le menu de l'interface utilisateur. Si aucun nom n'est indiqué, c'est la sélection actuelle qui est actualisée. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne : `true` si Flash a bien mis l'élément à jour, `false` dans le cas contraire.

### Description

Méthode qui actualise l'élément de bibliothèque spécifié.

### Exemple

L'exemple suivant affiche une boîte de dialogue qui montre si l'élément actuellement sélectionné a été actualisé (`true`) ou non (`false`) :

```
alert(fl.getDocumentDOM().library.updateItem());
```

# Chapitre 27 : Objet Math

## Disponibilité

Flash MX 2004.

## Description

L'objet Math est disponible sous forme de propriété en lecture seule de l'objet flash ; voir [fl.Math](#). Cet objet comporte des méthodes qui prennent en charge les opérations mathématiques courantes.

## Résumé des méthodes

Les méthodes suivantes sont disponibles pour l'objet Math :

Méthode	Description
<a href="#">Math.concatMatrix()</a>	Effectue une concaténation matricielle et renvoie le résultat.
<a href="#">Math.invertMatrix()</a>	Renvoie l'inverse de la matrice spécifiée.
<a href="#">Math.pointDistance()</a>	Calcule la distance entre deux points.

## Math.concatMatrix()

## Disponibilité

Flash MX 2004.

## Utilisation

```
Math.concatMatrix(mat1, mat2)
```

## Paramètres

**mat1, mat2** Spécifient les objets Matrix à concaténer (voir [Objet Matrix](#)). Chaque paramètre doit être un objet avec les champs a, b, c, d, tx et ty.

## Valeur renvoyée

Une matrice d'objets concaténés.

## Description

Méthode qui effectue une concaténation de matrices et renvoie le résultat.

## Exemple

L'exemple suivant enregistre dans la variable `elt` l'objet sélectionné, multiplie le tableau des objets par le tableau d'affichage et enregistre cette valeur dans la variable `mat` :

```
var elt = fl.getDocumentDOM().selection[0];
var mat = fl.Math.concatMatrix( elt.matrix , fl.getDocumentDOM().viewMatrix );
```

## Math.invertMatrix()

### Disponibilité

Flash MX 2004.

### Utilisation

```
Math.invertMatrix(mat)
```

### Paramètres

**mat** Indique l'objet Matrix à inverser (voir [Objet Matrix](#)). Les champs suivants doivent être présents : a, b, c, d, tx et ty.

### Valeur renvoyée

Un objet Matrix qui est l'inverse de la matrice originale.

### Description

Méthode qui renvoie l'inverse de la matrice spécifiée.

### Exemple

L'exemple suivant enregistre dans la variable `elt` l'objet sélectionné, assigne cette matrice à la variable `mat`, puis enregistre son inverse dans la variable `inv` :

```
var elt = fl.getDocumentDOM().selection[0];  
var mat = elt.matrix;  
var inv = fl.Math.invertMatrix( mat );
```

## Math.pointDistance()

### Disponibilité

Flash MX 2004.

### Utilisation

```
Math.pointDistance(pt1, pt2)
```

### Paramètres

**pt1, pt2** Spécifient les points entre lesquels la distance doit être mesurée.

### Valeur renvoyée

Une valeur en virgule flottante qui représente la distance entre les points.

### Description

Méthode qui calcule la distance entre deux points.

### Exemple

L'exemple suivant stocke la distance entre *pt1* et *pt2* dans la variable `dist` :

```
var pt1 = {x:10, y:20}  
var pt2 = {x:100, y:200}  
var dist = fl.Math.pointDistance(pt1, pt2);
```

# Chapitre 28 : Objet Matrix

## Disponibilité

Flash MX 2004.

## Description

L'objet Matrix représente une matrice de transformation.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Matrix :

Propriété	Description
<code>matrix.a</code>	Nombre en virgule flottante spécifiant l'élément (0,0) de la matrice de transformation.
<code>matrix.b</code>	Nombre en virgule flottante spécifiant l'élément (0,1) de la matrice.
<code>matrix.c</code>	Nombre en virgule flottante spécifiant l'élément (1,0) de la matrice.
<code>matrix.d</code>	Nombre en virgule flottante spécifiant l'élément (1,1) de la matrice.
<code>matrix.tx</code>	Nombre en virgule flottante spécifiant l'emplacement de l'axe x du point d'alignement d'un symbole ou du centre d'une forme.
<code>matrix.ty</code>	Nombre en virgule flottante spécifiant l'emplacement de l'axe y du point d'alignement d'un symbole ou du centre d'une forme.

## matrix.a

## Disponibilité

Flash MX 2004.

## Utilisation

`matrix.a`

## Description

Propriété ; nombre en virgule flottante spécifiant l'élément (0,0) de la matrice de transformation. Cette valeur représente le facteur de redimensionnement de l'axe *x* de l'objet.

## Exemple

Les propriétés *a* et *d* d'une matrice représentent son redimensionnement. Dans l'exemple suivant, les valeurs sont respectivement définies sur 2 et 3, afin de doubler la largeur de l'objet sélectionné et de tripler sa hauteur :

```
var mat = fl.getDocumentDOM().selection[0].matrix;
mat.a = 2;
mat.d = 3;
fl.getDocumentDOM().selection[0].matrix = mat;
```

Il est possible de faire pivoter un objet en définissant les propriétés de matrice *a*, *b*, *c* et *d* l'une par rapport à l'autre, avec  $a = d$  et  $b = -c$ . Par exemple, des valeurs de 0,5, 0,8, -0,8 et 0,5 feront pivoter l'objet de 60° :



```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.a = 0.5;  
mat.b = 0.8;  
mat.c = 0.8*(-1);  
mat.d = 0.5;  
fl.getDocumentDOM().selection[0].matrix = mat;
```

Les valeurs  $a = d = 1$  et  $c = b = 0$  restaureront la forme originale de l'objet.

## matrix.b

### Disponibilité

Flash MX 2004.

### Utilisation

matrix.b

### Description

Propriété ; nombre en virgule flottante spécifiant l'élément (0,1) de la matrice. Cette valeur représente l'inclinaison verticale d'une forme ; elle provoque le déplacement du côté droit de la forme le long de l'axe vertical.

Les propriétés `matrix.b` et `matrix.c` d'une matrice représentent son inclinaison (voir [matrix.c](#)).

### Exemple

Dans l'exemple suivant, vous pouvez donner à `b` et `c` les valeurs -1 et 0, respectivement. Ces valeurs inclinent l'objet d'un angle vertical de 45° :

```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.b = -1;  
mat.c = 0;  
fl.getDocumentDOM().selection[0].matrix = mat;
```

Pour restaurer la forme originale de l'objet, donnez à `b` et `c` la valeur 0.

Voir aussi l'exemple [matrix.a](#).

## matrix.c

### Disponibilité

Flash MX 2004.

### Utilisation

matrix.c

### Description

Propriété ; nombre en virgule flottante spécifiant l'élément (1,0) de la matrice. Cette valeur provoque une inclinaison de l'objet en déplaçant son bord inférieur le long d'un axe horizontal.

Les propriétés `matrix.b` et `matrix.c` d'une matrice représentent son inclinaison.

### Exemple

Voir l'exemple de [matrix.b](#).

## matrix.d

### Disponibilité

Flash MX 2004.

### Utilisation

`matrix.d`

### Description

Propriété ; nombre en virgule flottante spécifiant l'élément (1,1) de la matrice. Cette valeur représente le facteur de redimensionnement de l'axe y de l'objet.

### Exemple

Voir aussi l'exemple de [matrix.a](#).

## matrix.tx

### Disponibilité

Flash MX 2004.

### Utilisation

`matrix.tx`

### Description

Propriété ; nombre en virgule flottante spécifiant l'emplacement de l'axe x du point d'alignement (également *point d'origine* ou *point zéro*) d'un symbole ou du centre d'une forme. Elle définit la translation de la transformation sur l'axe x.

Vous pouvez déplacer un objet en définissant les propriétés `matrix.tx` et `matrix.ty` (voir [matrix.ty](#)).

### Exemple

Dans l'exemple suivant, les paramètres `tx` et `ty` reçoivent la valeur 0 pour amener le point d'alignement de l'objet au point 0,0 du document :

```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.tx = 0;  
mat.ty = 0;  
fl.getDocumentDOM().selection[0].matrix = mat;
```

## matrix.ty

### Disponibilité

Flash MX 2004.

### Utilisation

`matrix.ty`

### Description

Propriété ; nombre en virgule flottante spécifiant l'emplacement de l'axe *y* du point d'alignement d'un symbole ou du centre d'une forme. Elle définit la translation de la transformation sur l'axe *y*.

Vous pouvez déplacer un objet en définissant ses propriétés `matrix.tx` et `matrix.ty`.

### Exemple

Voir l'exemple de `matrix.tx`.

# Chapitre 29 : Objet outputPanel

## Disponibilité

Flash MX 2004.

## Description

L'objet `outputPanel` représente le panneau Sortie, qui affiche des informations facilitant le débogage, par exemple les erreurs de syntaxe. Vous accédez à cet objet via `fl.outputPanel` (ou `flash.outputPanel`). Voir [fl.outputPanel](#).

## Résumé des méthodes

L'objet `outputPanel` utilise les méthodes suivantes :

Méthode	Description
<a href="#">outputPanel.clear()</a>	Efface le contenu du panneau Sortie.
<a href="#">outputPanel.save()</a>	Enregistre le contenu du panneau Sortie dans un fichier texte local.
<a href="#">outputPanel.trace()</a>	Ajoute une ligne au contenu du panneau Sortie, terminé par un code de nouvelle ligne.

## outputPanel.clear()

## Disponibilité

Flash MX 2004.

## Utilisation

```
outputPanel.clear()
```

## Paramètres

Aucun.

## Valeur renvoyée

Aucune.

## Description

Méthode qui efface le contenu du panneau Sortie. Vous pouvez utiliser cette méthode dans une application de traitement par lots pour effacer une liste d'erreurs, ou pour les enregistrer de façon incrémentielle en utilisant cette méthode avec [outputPanel.save\(\)](#).

## Exemple

L'exemple suivant efface le contenu actuel du panneau Sortie :

```
fl.outputPanel.clear();
```

## `outputPanel.save()`

### Disponibilité

Flash MX 2004 ; paramètre *bUseSystemEncoding* ajouté à Flash 8.

### Utilisation

```
outputPanel.save(fileURI [, bAppendToFile [ , bUseSystemEncoding]])
```

### Paramètres

**fileURI** Chaîne de type « fichier:///URI » qui spécifie le nom du fichier local qui doit recevoir le contenu du panneau Sortie.

**bAppendToFile** Valeur booléenne facultative. Si elle est définie sur `true`, le contenu du panneau Sortie est ajouté à la fin du fichier de sortie. Si elle est définie sur `false`, la méthode écrase le fichier de sortie, s'il existe déjà. La valeur par défaut est `false`.

**bUseSystemEncoding** Valeur booléenne facultative. Si elle est définie sur `true`, la méthode enregistre le texte du panneau Sortie en utilisant le codage système. Si la valeur est `false`, le texte du panneau Sortie est enregistré en codage UTF-8, avec des caractères d'ordre d'octet au début du texte. La valeur par défaut est `false`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui enregistre le contenu du panneau Sortie dans un fichier texte local, soit en remplaçant le fichier, soit en ajoutant les informations à la fin du fichier existant.

Si *fileURI* n'est pas spécifié ou n'est pas valide, une erreur est renvoyée.

Cette méthode s'avère pratique pour le traitement par lots. Vous pouvez par exemple créer un fichier JSFL regroupant plusieurs composants compilés. Comme toute éventuelle erreur de compilation sera affichée dans le panneau Sortie, vous pouvez utiliser cette méthode pour enregistrer les erreurs dans un fichier texte, qui sera automatiquement analysé par le système de compilation utilisé.

### Exemple

L'exemple suivant enregistre le contenu du panneau Sortie dans le fichier `batch.log`, qui réside dans le dossier `/tests`, et écrase son contenu le cas échéant :

```
fl.outputPanel.save("file:///c:/tests/batch.log");
```

## `outputPanel.trace()`

### Disponibilité

Flash MX 2004.

### Utilisation

```
outputPanel.trace(message)
```

### Paramètres

**message** Chaîne qui contient le texte à ajouter au panneau Sortie.

### Valeur renvoyée

Aucune.

### Description

Méthode qui envoie une chaîne de texte au panneau Sortie, suivie d'un code de nouvelle ligne, et affiche le panneau Sortie s'il n'est pas déjà visible. Cette méthode est identique à `fl.trace()` et fonctionne de la même manière que l'instruction `trace()` dans `ActionScript`.

Pour envoyer une ligne vide, utilisez `outputPanel.trace("")` ou `outputPanel.trace("\n")`. Vous pouvez utiliser cette dernière commande en ligne si vous intégrez `\n` à la chaîne *message*.

### Exemple

L'exemple suivant affiche plusieurs lignes de texte dans le panneau Sortie :

```
fl.outputPanel.clear();
fl.outputPanel.trace("Hello World!!!");
var myPet = "cat";
fl.outputPanel.trace("\nI have a " + myPet);
fl.outputPanel.trace("");
fl.outputPanel.trace("I love my " + myPet);
fl.outputPanel.trace("Do you have a " + myPet + "?");
```

# Chapitre 30 : Objet Oval

**Héritage**    [Objet Element](#) > [Objet Shape](#) > [Objet Oval](#)

## Disponibilité

Flash CS3 Professional.

## Description

L'objet Oval est une forme tracée à l'aide de l'outil Ovale primitif. Pour savoir si un élément est un objet Oval, utilisez [shape.isOvalObject](#).

## Résumé des propriétés

Outre les propriétés de l'objet [Objet Shape](#), l'objet Oval gère les propriétés suivantes. Pour définir les propriétés d'un objet Oval, utilisez [document.setOvalObjectProperty\(\)](#) :

Propriété	Description
<a href="#">OvalObject.closePath</a>	Lecture seule ; valeur booléenne spécifiant si la case à cocher Fermer le tracé, qui se trouve dans l'inspecteur Propriétés, est sélectionnée ou non.
<a href="#">OvalObject.endAngle</a>	Lecture seule ; valeur en virgule flottante qui indique l'angle de fin de l'objet Oval.
<a href="#">OvalObject.innerRadius</a>	Lecture seule ; valeur en virgule flottante qui précise, en pourcentage, le rayon interne de l'objet Oval.
<a href="#">OvalObject.startAngle</a>	Lecture seule ; valeur en virgule flottante qui indique l'angle de début de l'objet Oval.

## OvalObject.closePath

## Disponibilité

Flash CS3 Professional.

## Utilisation

`OvalObject.closePath`

## Description

Propriété en lecture seule ; valeur booléenne spécifiant si la case à cocher Fermer le tracé, qui se trouve dans l'inspecteur Propriétés, est sélectionnée ou non. Si les valeurs de l'angle de début et de fin sont identiques pour l'objet, la définition de cette propriété n'a aucune incidence tant que ces valeurs demeurent inchangées.

Pour définir cette valeur, utilisez [document.setOvalObjectProperty\(\)](#).

## Exemple

L'exemple suivant désélectionne la propriété `OvalObject.closePath` :

```
fl.getDocumentDOM().setOvalObjectProperty("closePath", false);
```

**Voir aussi**

`document.setOvalObjectProperty(), shape.isOvalObject`

## OvalObject.endAngle

**Disponibilité**

Flash CS3 Professional.

**Utilisation**

`OvalObject.endAngle`

**Description**

Propriété en lecture seule ; valeur en virgule flottante qui indique l'angle de fin de l'objet Oval. Les valeurs acceptées sont comprises entre 0 et 360.

Pour définir cette valeur, utilisez `document.setOvalObjectProperty()`.

**Exemple**

L'exemple suivant définit sur 270 l'angle de fin des objets Oval sélectionnés :

```
fl.getDocumentDOM().setOvalObjectProperty("endAngle", 270);
```

**Voir aussi**

`document.setOvalObjectProperty(), OvalObject.startAngle, shape.isOvalObject`

## OvalObject.innerRadius

**Disponibilité**

Flash CS3 Professional.

**Utilisation**

`OvalObject.innerRadius`

**Description**

Propriétés en lecture seule ; valeur en virgule flottante qui précise, en pourcentage, le rayon interne de l'objet Oval. Les valeurs acceptées sont comprises entre 0 et 99.

Pour définir cette valeur, utilisez `document.setOvalObjectProperty()`.

**Exemple**

L'exemple suivant définit sur 50% le rayon interne des objets Oval sélectionnés :

```
fl.getDocumentDOM().setOvalObjectProperty("innerRadius", 50);
```

**Voir aussi**

`document.setOvalObjectProperty(), shape.isOvalObject`



# OvalObject.startAngle

## Disponibilité

Flash CS3 Professional.

## Utilisation

`OvalObject.startAngle`

## Description

Propriété en lecture seule ; valeur en virgule flottante qui indique l'angle de début de l'objet Oval. Les valeurs acceptées sont comprises entre 0 et 360.

Pour définir cette valeur, utilisez `document.setOvalObjectProperty()`.

## Exemple

L'exemple suivant définit sur 270 l'angle de début des objets Oval sélectionnés :

```
fl.getDocumentDOM().setOvalObjectProperty("startAngle", 270);
```

## Voir aussi

`document.setOvalObjectProperty()`, `OvalObject.endAngle`, `shape.isOvalObject`

# Chapitre 31 : Objet Parameter

## Disponibilité

Flash MX 2004.

## Description

Le type d'objet Parameter est accessible depuis le tableau `componentInstance.parameters` (qui correspond à l'inspecteur des propriétés du composant dans l'outil de création).

## Résumé des méthodes

Les méthodes suivantes sont disponibles pour l'objet Parameter :

Méthode	Description
<code>parameter.insertItem()</code>	Insère un élément dans une liste, un objet ou un tableau.
<code>parameter.removeItem()</code>	Supprime un élément de type liste, objet ou tableau du paramètre d'un écran ou d'un composant.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Parameter :

Propriété	Description
<code>parameter.category</code>	Chaîne indiquant la propriété <code>category</code> du paramètre <code>screen</code> ou <code>componentInstance</code> .
<code>parameter.listIndex</code>	Entier spécifiant la valeur de l'élément de liste sélectionné.
<code>parameter.name</code>	Lecture seule ; chaîne indiquant le nom du paramètre.
<code>parameter.value</code>	Correspond au champ Valeur de l'onglet Paramètres du panneau Inspecteur des composants, de l'onglet Paramètres de l'inspecteur Propriétés ou de la fenêtre inspecteur Propriétés.
<code>parameter.valueType</code>	Lecture seule ; chaîne indiquant le type de paramètre d'écran ou de composant.
<code>parameter.verbose</code>	Détermine où le paramètre est affiché.

## parameter.category

## Disponibilité

Flash MX 2004.

## Utilisation

`parameter.category`

## Description

Propriété ; chaîne indiquant la propriété `category` du paramètre `screen` ou `componentInstance`. Cette propriété représente une autre manière de présenter une liste de paramètres. Cette fonctionnalité n'est pas disponible dans l'interface utilisateur de Flash.

## parameter.insertItem()

### Disponibilité

Flash MX 2004.

### Utilisation

```
parameter.insertItem(index, name, value, type)
```

### Paramètres

**index** Index (de base zéro) sur un entier, qui indique si l'élément sera inséré dans la liste, l'objet ou le tableau. Si l'index est 0, l'élément est inséré au début de la liste. Si l'index est supérieur à la taille de la liste, l'élément est inséré à la fin du tableau.

**name** Chaîne spécifiant le nom de l'élément à insérer. Ce paramètre est obligatoire pour les paramètres d'objet.

**value** Chaîne spécifiant la valeur de l'élément à insérer.

**type** Chaîne spécifiant le type de l'élément à insérer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui insère un élément dans une liste, un objet ou un tableau. Si l'un des paramètres est une liste, un objet ou un tableau, la propriété *value* est un tableau.

### Exemple

L'exemple suivant insère la valeur de New Value dans le paramètre labelPlacement :

```
// Select an instance of a Button component on the Stage.
var parms = fl.getDocumentDOM().selection[0].parameters;
parms[2].insertItem(0, "name", "New Value", "String");
var values = parms[2].value;
for(var prop in values){
    fl.trace("labelPlacement parameter value = " + values[prop].value);
}
```

## parameter.listIndex

### Disponibilité

Flash MX 2004.

### Utilisation

```
parameter.listIndex
```

### Description

Propriété ; valeur de l'élément de liste sélectionné. Cette propriété n'est valide que si `parameter.valueType` est "List".

### Exemple

L'exemple suivant définit le premier paramètre d'un objet diapositive, c'est-à-dire le paramètre `autoKeyNav`. Pour définir le paramètre sur l'une de ses valeurs gérées (`true`, `false` ou `inherit`) `parameter.listIndex` reçoit la valeur de l'index de l'élément dans la liste (0 pour `true`, 1 pour `false`, 2 pour `inherit`).

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;  
parms[0].listIndex = 1;
```

## parameter.name

### Disponibilité

Flash MX 2004.

### Utilisation

`parameter.name`

### Description

Propriété en lecture seule ; chaîne indiquant le nom du paramètre.

### Exemple

L'exemple suivant affiche le nom du cinquième paramètre du composant sélectionné :

```
var parms = fl.getDocumentDOM().selection[0].parameters;  
fl.trace("name: " + parms[4].name);
```

L'exemple suivant affiche le nom du cinquième paramètre de l'écran spécifié :

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters; fl.trace("name: " +  
parms[4].name);
```

## parameter.removeItem()

### Disponibilité

Flash MX 2004.

### Utilisation

`parameter.removeItem(index)`

### Paramètres

**index** Index sous forme de nombre entier (de base zéro) de l'élément à supprimer de la propriété d'écran ou de composant.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime un élément du type liste, objet ou tableau du paramètre d'un écran ou d'un composant.

### Exemple

L'exemple suivant supprime l'élément à l'index1 depuis le paramètre labelPlacement d'un composant :

```
// Select an instance of a Button component on the Stage.
var parms = fl.getDocumentDOM().selection[0].parameters;
var values = parms[2].value;
fl.trace("--Original--");
for(var prop in values){
  fl.trace("labelPlacement value = " + values[prop].value);
}
parms[2].removeItem(1);

var newValues = parms[2].value;
fl.trace("--After Removing Item--");
for(var prop in newValues){
  fl.trace("labelPlacement value = " + newValues[prop].value);
}
```

L'exemple suivant supprime l'élément à l'index1 du paramètre autoKeyNav d'un écran :

```
// Open a presentation document.
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;
var values = parms[0].value;
fl.trace("--Original--");
for(var prop in values){
  fl.trace("autoKeyNav value = " + values[prop].value);
}
parms[0].removeItem(1);

var newValues = parms[0].value;
fl.trace("--After Removing Item--");
for(var prop in newValues){
  fl.trace("autoKeyNav value = " + newValues[prop].value);
}
```

## parameter.value

### Disponibilité

Flash MX 2004.

### Utilisation

parameter.value

### Description

Propriété qui correspond au champ Valeur de l'onglet Paramètres du panneau Inspecteur des composants, de l'onglet Paramètres de l'inspecteur Propriétés ou de la fenêtre inspecteur Propriétés. Le type de la propriété value est déterminé par la propriété valueType du paramètre (voir [parameter.valueType](#)).

## parameter.valueType

### Disponibilité

Flash MX 2004.

### Utilisation

`parameter.valueType`

### Description

Propriété en lecture seule ; chaîne indiquant le type de paramètre d'écran ou de composant. Le type peut prendre l'une des valeurs suivantes : "Default", "Array", "Object", "List", "String", "Number", "Boolean", "Font Name", "Color", "Collection", "Web Service URL" ou "Web Service Operation".

### Voir aussi

[parameter.value](#)

## parameter.verbose

### Disponibilité

Flash MX 2004.

### Utilisation

`parameter.verbose`

### Description

Propriété ; détermine où le paramètre est affiché. Si la valeur de cette propriété est 0 (non détaillée), le paramètre est uniquement affiché dans l'inspecteur des composants. Si elle est définie sur 1 (détaillée), le paramètre est affiché dans l'inspecteur des composants et dans l'onglet Paramètres de l'inspecteur Propriétés.

# Chapitre 32 : Objet Path

## Disponibilité

Flash MX 2004.

## Description

L'objet Path définit une séquence de segments de ligne (droite, courbe ou les deux), qui est particulièrement destinée à la création d'outils extensibles. L'exemple suivant montre le renvoi d'une occurrence d'un objet Path par l'objet flash :

```
path = fl.drawingLayer.newPath();
```

Voir aussi l'[Objet drawingLayer](#).

## Résumé des méthodes

Les méthodes suivantes sont disponibles pour l'objet Path :

Méthode	Description
<code>path.addCubicCurve()</code>	Ajoute un segment de courbe de Bézier cubique au tracé.
<code>path.addCurve()</code>	Ajoute un segment de courbe de Bézier quadratique au tracé.
<code>path.addPoint()</code>	Ajoute un point au tracé.
<code>path.clear()</code>	Supprime tous les points du tracé.
<code>path.close()</code>	Ajoute un point à l'emplacement du premier point du tracé, puis étend le tracé jusqu'à ce point, ce qui ferme le tracé.
<code>path.makeShape()</code>	Crée une forme sur la scène en utilisant les paramètres de trait et de remplissage actuels.
<code>path.newContour()</code>	Ajoute un nouveau contour au tracé.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Path :

Propriété	Description
<code>path.nPts</code>	Lecture seule ; entier représentant le nombre de points du tracé.

## path.addCubicCurve()

### Disponibilité

Flash MX 2004.

### Utilisation

```
path.addCubicCurve(xAnchor, yAnchor, x2, y2, x3, y3, x4, y4)
```

### Paramètres

**xAnchor** Valeur en virgule flottante indiquant la coordonnée *x* du premier point de contrôle.

**yAnchor** Valeur en virgule flottante indiquant la coordonnée *y* du premier point de contrôle.

**x2** Valeur en virgule flottante indiquant la coordonnée *x* du second point de contrôle.

**y2** Valeur en virgule flottante indiquant la coordonnée *y* du second point de contrôle.

**x3** Valeur en virgule flottante indiquant la coordonnée *x* du troisième point de contrôle.

**y3** Valeur en virgule flottante indiquant la coordonnée *y* du troisième point de contrôle.

**x4** Valeur en virgule flottante indiquant la coordonnée *x* du quatrième point de contrôle.

**y4** Valeur en virgule flottante indiquant la coordonnée *y* du quatrième point de contrôle.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui ajoute un segment de courbe de Bézier cubique au tracé.

#### Exemple

L'exemple suivant crée un nouveau tracé, el stocke dans la variable `myPath`, puis lui affecte la courbe :

```
var myPath = fl.drawingLayer.newPath();  
myPath.addCubicCurve(0, 0, 10, 20, 20, 20, 30, 0);
```

## path.addCurve()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
path.addCurve(xAnchor, yAnchor, x2, y2, x3, y3)
```

#### Paramètres

**xAnchor** Valeur en virgule flottante indiquant la coordonnée *x* du premier point de contrôle.

**yAnchor** Valeur en virgule flottante indiquant la coordonnée *y* du premier point de contrôle.

**x2** Valeur en virgule flottante indiquant la coordonnée *x* du second point de contrôle.

**y2** Valeur en virgule flottante indiquant la coordonnée *y* du second point de contrôle.

**x3** Valeur en virgule flottante indiquant la coordonnée *x* du troisième point de contrôle.

**y3** Valeur en virgule flottante indiquant la coordonnée *y* du troisième point de contrôle.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui ajoute un segment de courbe de Bézier quadratique au tracé.



### Exemple

L'exemple suivant crée un nouveau tracé, l'enregistre dans la variable `myPath`, puis affecte la courbe au tracé :

```
var myPath = fl.drawingLayer.newPath();  
myPath.addCurve(0, 0, 10, 20, 20, 0);
```

## path.addPoint()

### Disponibilité

Flash MX 2004.

### Utilisation

```
path.addPoint(x, y)
```

### Paramètres

**x** Nombre en virgule flottante indiquant la coordonnée *x* du point.

**y** Nombre en virgule flottante indiquant la coordonnée *y* du point.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute un point au tracé.

### Exemple

L'exemple suivant crée un nouveau tracé, le stocke dans la variable `myPath`, puis lui affecte le nouveau point :

```
var myPath = fl.drawingLayer.newPath();  
myPath.addPoint(10, 100);
```

## path.clear()

### Disponibilité

Flash MX 2004.

### Utilisation

```
path.clear()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime tous les points du tracé.

### Exemple

L'exemple suivant supprime tous les points d'un tracé stocké dans la variable `myPath` :

```
var myPath = fl.drawingLayer.newPath();  
myPath.clear();
```

## path.close()

### Disponibilité

Flash MX 2004.

### Utilisation

```
path.close();
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ajoute un point au niveau du premier point du tracé, puis étend le tracé jusqu'à ce point, ce qui ferme le tracé. Si le tracé ne comporte pas de points, aucun point n'est ajouté.

### Exemple

L'exemple suivant crée un tracé fermé :

```
var myPath = fl.drawingLayer.newPath();  
myPath.close();
```

## path.makeShape()

### Disponibilité

Flash MX 2004.

### Utilisation

```
path.makeShape([bSupressFill [, bSupressStroke]])
```

### Paramètres

**bSupressFill** Valeur booléenne. Si elle est `true`, la méthode supprime le remplissage qui devrait être appliqué à la forme. La valeur par défaut est `false`. Ce paramètre est facultatif.

**bSuppressStroke** Valeur booléenne. Si elle est `true`, la méthode supprime le trait qui devrait être appliqué à la forme. La valeur par défaut est `false`. Ce paramètre est facultatif.

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui crée une forme sur la scène en utilisant les paramètres de trait et de remplissage actuels. Le tracé est effacé après la création de la forme. Cette méthode comporte deux paramètres facultatifs permettant de supprimer le remplissage et le trait de l'objet forme ainsi obtenu. Si ces paramètres sont omis ou reçoivent la valeur `false`, les valeurs actuelles de remplissage et de trait sont utilisées.

**Exemple**

L'exemple suivant crée une forme dotée du remplissage actuel, mais sans trait :

```
var myPath = fl.drawingLayer.newPath();  
myPath.makeShape(false, true);
```

## path.newContour()

**Disponibilité**

Flash MX 2004.

**Utilisation**

```
path.newContour()
```

**Paramètres**

Aucun.

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui ajoute un nouveau contour au tracé.

**Exemple**

L'exemple suivant crée un carré vide :

```
var myPath = fl.drawingLayer.newPath();
myPath.addPoint(0, 0);
myPath.addPoint(0, 30);
myPath.addPoint(30, 30);
myPath.addPoint(30, 0);
myPath.addPoint(0, 0);

myPath.newContour();
myPath.addPoint(10, 10);
myPath.addPoint(10, 20);
myPath.addPoint(20, 20);
myPath.addPoint(20, 10);
myPath.addPoint(10, 10);

myPath.makeShape();
```

## path.nPts

### Disponibilité

Flash MX 2004.

### Utilisation

path.nPts

### Description

Propriété en lecture seule ; entier représentant le nombre de points du tracé. Un nouveau tracé possède 0 point.

### Exemple

L'exemple suivant affiche dans le panneau Sortie le nombre de points du tracé référencé par la variable `myPath` :

```
var myPath = fl.drawingLayer.newPath();
var numOfPoints = myPath.nPts;
fl.trace("Number of points in the path: " + numOfPoints);
// Displays: Number of points in the path: 0
```

# Chapitre 33 : Objet presetItem

## Disponibilité

Flash CS4 Professional.

## Description

L'objet `presetItem` représente un élément (présélection ou dossier) du panneau Présélections de mouvement (Fenêtre > Présélections de mouvement). Le tableau d'objets `presetItem` est une propriété de l'objet `presetPanel` (`presetPanel.items`).

Toutes les propriétés de l'objet `presetItem` sont en lecture seule. Pour exécuter des tâches telles que supprimer, renommer ou déplacer des éléments, utilisez les méthodes de l'Objet `presetPanel`.

## Résumé des propriétés

L'objet `presetItem` gère les propriétés suivantes :

Propriété	Description
<code>presetItem.isDefault</code>	Spécifie si l'élément est installé avec Flash ou s'il s'agit d'un élément personnalisé que vous (ou quelqu'un d'autre) avez créé.
<code>presetItem.isFolder</code>	Spécifie si l'élément du panneau Présélections de mouvement est un dossier ou une présélection.
<code>presetItem.level</code>	Niveau de l'élément dans la structure de dossiers du panneau Présélections de mouvement.
<code>presetItem.name</code>	Nom de la présélection ou du dossier, sans information sur le chemin d'accès.
<code>presetItem.open</code>	Spécifie si un dossier du panneau Présélections de mouvement est actuellement développé.
<code>presetItem.path</code>	Chemin de l'élément dans l'arborescence de dossiers du panneau Présélections de mouvement et nom de l'élément.

## presetItem.isDefault

## Disponibilité

Flash CS4 Professional.

## Utilisation

`presetItem.isDefault`

## Description

Propriété en lecture seule ; valeur booléenne spécifiant si l'élément est installé avec Flash (`true`) ou s'il s'agit d'un élément personnalisé que vous (ou quelqu'un d'autre) avez créé (`false`). Si cette valeur est `true`, vous pouvez le considérer comme un élément en lecture seule ; il ne peut pas être déplacé, supprimé ni faire l'objet d'une opération similaire.

### Exemple

L'exemple suivant affiche le contenu du panneau Présélections de mouvement et indique si un élément est installé avec Flash :

```
fl.outputPanel.clear();
var presetItemArray=fl.presetPanel.items;
for (i=0;i<presetItemArray.length; i++){
    var presetItem = presetItemArray[i];
    fl.trace(presetItem.name +", default =" + presetItem.isDefault);
}
```

## presetItem.isFolder

### Disponibilité

Flash CS4 Professional.

### Utilisation

`presetItem.isFolder`

### Description

Propriété en lecture seule ; valeur booléenne spécifiant si l'élément du panneau Présélections de mouvement est un dossier (`true`) ou une présélection (`false`).

### Exemple

L'exemple suivant indique que le premier élément du panneau Présélections de mouvement est un dossier et le second une présélection :

```
var presetItemArray=fl.presetPanel.items;
fl.trace(presetItemArray[0].isFolder);
fl.trace(presetItemArray[1].isFolder);
```

## presetItem.level

### Disponibilité

Flash CS4 Professional.

### Utilisation

`presetItem.level`

### Description

Propriété en lecture seule ; nombre entier spécifiant le niveau de l'élément dans la structure de dossiers du panneau Présélections de mouvement. Les dossiers Présélections par défaut et Présélections personnalisées sont de niveau 0.

### Exemple

L'exemple suivant indique que le premier élément du panneau Présélections de mouvement est de niveau 0 et le second de niveau 1 :

```
var presetItemArray=fl.presetPanel.items;  
fl.trace(presetItemArray[0].level);  
fl.trace(presetItemArray[1].level);
```

## presetItem.name

### Disponibilité

Flash CS4 Professional.

### Utilisation

`presetItem.name`

### Description

Propriété en lecture seule ; chaîne représentant le nom de la présélection ou du dossier, sans information sur le chemin.

### Exemple

Voir [presetItem.path](#).

## presetItem.open

### Disponibilité

Flash CS4 Professional.

### Utilisation

`presetItem.open`

### Description

Propriété en lecture seule ; précise si un dossier du panneau Présélections de mouvement est développé (`true`) ou non (`false`).

Cette propriété est `true` si l'élément n'est pas un dossier. Pour déterminer si un élément est un dossier ou une présélection, utilisez [presetItem.isFolder](#).

### Exemple

L'exemple suivant indique si les dossiers du panneau Présélections de mouvement sont développés ou réduits :

```
fl.outputPanel.clear();  
var presetItemArray=fl.presetPanel.items;  
for (i=0;i<presetItemArray.length; i++){  
    var presetItem = presetItemArray[i];  
    if (presetItem.isFolder) {  
        var status = presetItem.open ? "Open" : "Closed"  
        fl.trace(presetItem.level + "-" + presetItem.name + " folder is " + status);  
    }  
}
```

## `presetItem.path`

### Disponibilité

Flash CS4 Professional.

### Utilisation

`presetItem.path`

### Description

Propriété en lecture seule ; chaîne représentant le chemin de l'élément dans l'arborescence de dossiers du panneau Préréglages de mouvement et le nom de l'élément.

### Exemple

L'exemple suivant illustre la différence entre les valeurs de `presetItem.name` et de `presetItem.path`.

```
fl.outputPanel.clear();
var presetItemArray=fl.presetPanel.items;
for (i=0;i<presetItemArray.length; i++){
    var presetItem = presetItemArray[i];
    fl.trace("Name: " + presetItem.name + "\n" + "Path: " + presetItem.path);
    fl.trace("");
}
```



# Chapitre 34 : Objet presetPanel

## Disponibilité

Flash CS4 Professional.

## Description

L'objet `presetPanel` représente le panneau Présélections de mouvement (Fenêtre > Présélections de mouvement). Il s'agit d'une propriété de l'objet flash ([fl.presetPanel](#)).

## Résumé des méthodes

L'objet `presetPanel` gère les méthodes suivantes :

Méthode	Description
<code>presetPanel.addNewItem()</code>	Si une seule interpolation de mouvement est actuellement sélectionnée sur la scène, ajoute ce mouvement dans le panneau Présélections de mouvement.
<code>presetPanel.applyPreset()</code>	Applique la présélection spécifiée ou sélectionnée à l'élément actuellement sélectionné sur la scène.
<code>presetPanel.deleteFolder()</code>	Supprime le dossier spécifié et tous ses sous-dossiers de l'arborescence de dossiers du panneau Présélections de mouvement.
<code>presetPanel.deleteItem()</code>	Supprime la présélection spécifiée du panneau Présélections de mouvement.
<code>presetPanel.expandFolder()</code>	Développe ou réduit le ou les dossiers actuellement sélectionnés dans le panneau Présélections de mouvement.
<code>presetPanel.exportItem()</code>	Exporte la présélection spécifiée ou actuellement sélectionnée dans un fichier XML.
<code>presetPanel.findItemIndex()</code>	Renvoie un nombre entier représentant l'emplacement d'index d'un élément dans le panneau Présélections de mouvement.
<code>presetPanel.getSelectedItems()</code>	Renvoie un tableau d'objets <code>presetItem</code> correspondant aux éléments actuellement sélectionnés dans le panneau Présélections de mouvement.
<code>presetPanel.importItem()</code>	Ajoute une présélection dans le panneau Présélections de mouvement à partir d'un fichier XML spécifié.
<code>presetPanel.moveToFolder()</code>	Déplace l'élément spécifié vers le dossier spécifié.
<code>presetPanel.newFolder()</code>	Crée un dossier dans l'arborescence de dossiers du panneau Présélections de mouvement.
<code>presetPanel.renameItem()</code>	Remplace le nom de la présélection ou du dossier sélectionné(e) par un nom spécifié.
<code>presetPanel.selectItem()</code>	Sélectionne ou désélectionne un élément dans le panneau Présélections de mouvement.

## Résumé des propriétés

L'objet `presetPanel` gère la propriété suivante :

Propriété	Description
<code>presetPanel.items</code>	Tableau des objets <code>presetItem</code> présents dans le panneau Présélections de mouvement.

## `presetPanel.addNewItem()`

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
fl.presetPanel.addNewItem( [namePath] );
```

### Paramètres

**namePath** Chaîne qui spécifie le nom et le chemin de l'élément à ajouter au panneau Présélections de mouvement. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si l'élément a bien été ajouté, `false` dans le cas contraire.

### Description

Méthode qui, si une seule interpolation de mouvement est actuellement sélectionnée sur la scène, ajoute ce mouvement dans le panneau Présélections de mouvement, dans le dossier spécifié avec le nom spécifié. Le chemin spécifié dans *namePath* doit exister dans le panneau.

S'il existe une présélection correspondant à *namePath*, cette méthode n'a aucun effet et renvoie `false`.

Si vous ne transmettez pas de valeur pour *namePath*, l'élément est ajouté au dossier Présélections personnalisées avec le nom « Présélection personnalisée *n* », où *n* est incrémenté à chaque ajout d'un élément de cette manière.

### Exemple

En supposant qu'une seule interpolation de mouvement soit sélectionnée sur la scène, le code suivant ajoute une présélection nommée `Bouncing Ball` dans le dossier Présélections personnalisées :

```
fl.presetPanel.addNewItem("Custom Presets/Bouncing Ball");
```

### Voir aussi

[presetPanel.newFolder\(\)](#)

## `presetPanel.applyPreset()`

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.applyPreset( [presetPath] )
```

### Paramètres

**presetPath** Chaîne qui spécifie le chemin complet et le nom de la présélection à appliquer, telle qu'il apparaît dans le panneau Présélections de mouvement. Ce paramètre est facultatif ; si vous ne transmettez pas de valeur, la présélection actuellement sélectionnée est appliquée.

### Valeur renvoyée

Valeur booléenne, `true` si la présélection a bien été appliquée, `false` dans le cas contraire.

### Description

Méthode qui applique la présélection spécifiée ou actuellement sélectionnée à l'élément sélectionné sur la scène. L'élément doit être une interpolation de mouvement, un symbole ou un élément pouvant être converti en symbole. Si l'élément est une interpolation de mouvement, son mouvement actuel est remplacé par la présélection sélectionnée sans demander la confirmation de l'utilisateur.

Cette méthode échoue dans les cas suivants :

- Le chemin spécifié en tant que `presetPath` n'existe pas.
- Vous ne transmettez pas de valeur pour `presetPath` et aucune présélection n'est sélectionnée.
- Vous ne transmettez pas de valeur pour `presetPath` et plusieurs présélections sont sélectionnées.
- L'élément sélectionné sur la scène n'est pas un symbole et ne peut pas être converti en symbole.

### Exemple

L'exemple suivant applique la présélection `aDribble` à l'élément actuellement sélectionné sur la scène :

```
var result = fl.presetPanel.applyPreset("Custom Presets/Bounces/aDribble");  
fl.trace(result);
```

## presetPanel.deleteFolder()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.deleteFolder( [folderPath] )
```

### Paramètres

**folderPath** Chaîne qui spécifie le dossier à supprimer dans le panneau Présélections de mouvement. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si le ou les dossiers ont bien été supprimés, `false` dans le cas contraire.

### Description

Méthode qui supprime le dossier spécifié et tous ses sous-dossiers de l'arborescence de dossiers du panneau Présélections de mouvement. Toutes les présélections situées dans ces dossiers sont également supprimées. Les dossiers situés dans le dossier Présélections par défaut ne peuvent pas être supprimés.

Si vous ne transmettez pas de valeur pour `folderPath`, tous les dossiers actuellement sélectionnés sont supprimés.

**Remarque :** les dossiers sont supprimés sans demander la confirmation de l'utilisateur et l'action ne peut pas être annulée.

### Exemple

Le code suivant supprime un dossier nommé `Bouncing` du dossier Présélections personnalisées ; tous les sous-dossiers de `Bouncing` sont également supprimés.

```
fl.presetPanel.deleteFolder("Custom Presets/Bouncing");
```

### Voir aussi

[presetPanel.deleteItem\(\)](#)

## presetPanel.deleteItem()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.deleteItem( [namePath] )
```

### Paramètres

**namePath** Chaîne qui spécifie le chemin et le nom de l'élément à supprimer du panneau Présélections de mouvement. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si le ou les éléments ont bien été supprimés, `false` dans le cas contraire.

### Description

Méthode qui supprime la présélection spécifiée du panneau Présélections de mouvement. Si vous ne transmettez pas de valeur pour *namePath*, toutes les présélections actuellement sélectionnées sont supprimées. Les éléments du dossier Présélections par défaut ne peuvent pas être supprimés.

***Remarque :** les éléments sont supprimés sans demander la confirmation de l'utilisateur et l'action ne peut pas être annulée.*

### Exemple

Le code suivant supprime une présélection nommée `aDribble` du dossier Présélections personnalisées :

```
fl.presetPanel.deleteItem("Custom Presets/aDribble");
```

### Voir aussi

[presetPanel.deleteFolder\(\)](#)

## presetPanel.expandFolder()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.expandFolder( [bExpand [, bRecurse [, folderPath] ] ] )
```

### Paramètres

**bExpand** Valeur booléenne qui spécifie si le dossier doit être développé (`true`) ou réduit (`false`). Ce paramètre est facultatif ; la valeur par défaut est `true`.

**bRecurse** Valeur booléenne qui spécifie si les sous-dossiers du dossier doivent être développés (`true`) ou réduits (`false`). Ce paramètre est facultatif ; la valeur par défaut est `false`.

**folderPath** Chaîne qui spécifie le chemin du dossier à développer ou à réduire. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si le ou les dossiers ont bien été développés ou réduits, `false` dans le cas contraire.

### Description

Méthode qui développe ou réduit le ou les dossiers actuellement sélectionnés dans le panneau Présélections de mouvement. Pour développer ou réduire d'autres dossiers que ceux actuellement sélectionnés, transmettez une valeur pour *folderPath*.

### Exemple

L'exemple suivant développe le dossier Présélections personnalisées sans développer ses sous-dossiers :

```
fl.presetPanel.expandFolder(true, false, "Custom Presets");
```

L'exemple suivant développe le dossier Présélections personnalisées et ses sous-dossiers :

```
fl.presetPanel.expandFolder(true, true, "Custom Presets");
```

## presetPanel.exportItem()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.exportItem(fileURI [, namePath] )
```

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et spécifiant le chemin et éventuellement le nom du fichier exporté. Pour plus d'informations, voir la section « Description » ci-dessous.

**namePath** Chaîne qui spécifie le nom et le chemin de l'élément à sélectionner dans le panneau Présélections de mouvement. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si la présélection a bien été exportée, `false` dans le cas contraire.

### Description

Méthode qui exporte la présélection spécifiée ou actuellement sélectionnée dans un fichier XML. Seules des présélections peuvent être exportées ; si vous tentez d'exporter un dossier, la méthode échoue. Elle échoue également si vous tentez de remplacer un fichier du disque.

Si vous ne spécifiez pas de nom de fichier dans *fileURI* (c'est-à-dire, si le dernier caractère de *fileURI* est une barre oblique (/)), le fichier exporté est enregistré sous le nom de la présélection exportée. Si vous ne spécifiez pas de valeur pour *namePath*, la présélection actuellement sélectionnée est exportée. Voir l'exemple ci-dessous.

### Exemple

L'exemple suivant montre quels fichiers sont créés lorsque des paramètres différents sont transmis à cette méthode, et indique si le fichier spécifié a bien été créé. Avant d'exécuter cet exemple, sélectionnez la présélection fly-in-left (entrée rapide-gauche) dans le dossier Présélections par défaut et créez le dossier `My Presets` (Mes présélections) sur le disque.

```
//Exports fly-in-left to C:\My Presets\fly-in-left.xml
fl.presetPanel.exportItem("file:///C:/My Presets/");
//Exports fly-in-left to C:\My Presets\myFavoritePreset.xml
fl.presetPanel.exportItem("file:///C:/My Presets/myFavoritePreset.xml");
// Exports the "pulse" preset to C:\My Presets\pulse.xml
fl.presetPanel.exportItem("file:///C:/My Presets/", "Default Presets/pulse");
// Exports the "pulse" preset to C:\My Presets\thePulsePreset.xml
fl.presetPanel.exportItem("file:///C:/My Presets/thePulsePreset.xml", "Default
Presets/pulse");
```

### Voir aussi

`presetPanel.importItem()`

## presetPanel.findItemIndex()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.findItemIndex([presetName])
```

### Paramètres

**presetName** Chaîne qui spécifie le nom de la présélection dont la valeur d'index est renvoyée. Ce paramètre est facultatif.

### Valeur renvoyée

Nombre entier représentant l'index de la présélection spécifiée dans le tableau `presetPanel.items`. Si vous ne transmettez pas de valeur pour *presetName*, l'index de la présélection actuellement spécifiée est renvoyé. Cette méthode renvoie -1 dans les cas suivants :

- Vous n'avez pas transmis de valeur pour *presetName* et aucune présélection n'est sélectionnée.
- Vous n'avez pas transmis de valeur pour *presetName* et plusieurs présélections sont sélectionnées.
- La valeur que vous avez transmis pour *presetName* ne correspond pas à aucun élément du panneau.

### Description

Méthode qui renvoie un nombre entier représentant l'emplacement d'index d'un élément dans le panneau Présélections de mouvement.

### Exemple

Le code suivant affiche la valeur d'index et le nom de chemin complet de la présélection actuellement sélectionnée :

```
// Select one preset in the Motions Preset panel before running this code
var selectedPreset = fl.presetPanel.findItemIndex();
fl.trace(selectedPreset);
fl.trace(fl.presetPanel.items[selectedPreset].path);
```

## presetPanel.getSelectedItems()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.getSelectedItems()
```

### Paramètres

Aucun.

### Valeur renvoyée

Tableau d'objets `presetItem`.

### Description

Méthode qui renvoie un tableau d'objets `presetItem` correspondant aux éléments actuellement sélectionnés dans le panneau Présélections de mouvement (voir [Objet presetItem](#)). Chaque élément du tableau représente un dossier ou une présélection.

### Exemple

Le code suivant affiche les noms de chemin complets des éléments actuellement sélectionnés dans le panneau Présélections de mouvement :

```
var itemArray = fl.presetPanel.getSelectedItems();
var length = itemArray.length
for (x=0; x<length; x++) {
    fl.trace(itemArray[x].path);
}
```

### Voir aussi

[presetPanel.items](#)

## `presetPanel.importItem()`

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.importItem(fileURI [,namePath ])
```

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:/// URI » et spécifiant le fichier XML à importer en tant que présélection dans le panneau Présélections de mouvement.

**namePath** Chaîne spécifiant dans quel dossier le fichier importé doit être placé et le nom à lui donner. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si le fichier a bien été importé, `false` dans le cas contraire.

### Description

Méthode qui ajoute une présélection dans le panneau Présélections de mouvement à partir d'un fichier XML spécifié. Le chemin spécifié dans *namePath* doit exister dans le panneau.

Pour créer des fichiers XML pouvant être importés, utilisez `presetPanel.exportItem()`.

Si vous ne transmettez pas de valeur pour *namePath*, la présélection importée est placée dans le dossier Présélections personnalisées et nommée comme le fichier importé (sans l'extension XML).

### Exemple

L'exemple suivant importe une présélection dans le dossier Custom Presets/Pulse (Présélections personnalisées/Pulsation), et la nomme *fastPulse*.

```
fl.presetPanel.importItem("file:///C:/My Presets/thePulsePreset.xml", "Custom  
Presets/Pulse/fastPulse");
```

### Voir aussi

[presetPanel.exportItem\(\)](#)

## `presetPanel.items`

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.items
```

### Description

Propriété ; tableau des objets `presetItem` présents dans le panneau Présélections de mouvement (voir [Objet presetItem](#)). Chaque élément du tableau représente un dossier ou une présélection.



### Exemple

Le code suivant affiche les noms de chemin complets des éléments dans le panneau Présélections de mouvement :

```
var itemArray = fl.presetPanel.items;
var length = itemArray.length
for (x=0; x<length; x++) {
    fl.trace(itemArray[x].path);
}
```

### Voir aussi

[presetPanel.getSelectedItems\(\)](#)

## presetPanel.moveToFolder()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.moveToFolder(folderPath [, namePath] )
```

### Paramètres

**folderPath** Chaîne spécifiant le chemin du dossier du panneau Présélections de mouvement vers lequel le ou les éléments sont déplacés.

**namePath** Chaîne spécifiant le chemin et le nom de l'élément à déplacer. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si les éléments ont bien été déplacés, `false` dans le cas contraire.

### Description

Méthode qui déplace l'élément spécifié vers le dossier spécifié.

Si vous transmettez une chaîne vide ("" ) pour *folderPath*, les éléments sont déplacés vers le dossier Présélections personnalisées. Si vous ne transmettez pas de valeur pour *namePath*, les éléments actuellement sélectionnés sont déplacés.

Vous ne pouvez pas déplacer des éléments vers le dossier Présélections par défaut, ni à partir de ce dernier.

### Exemple

Dans l'exemple suivant, les éléments actuellement sélectionnés sont déplacés vers le dossier Custom Presets/Bouncing (Présélections personnalisées/Rebond). La présélection Fast Bounce (Rebond rapide) est ensuite déplacée vers le même dossier :

```
fl.presetPanel.moveToFolder("Custom Presets/Bouncing");
fl.presetPanel.moveToFolder("Custom Presets/Bouncing" , "Custom Presets/Fast Bounce");
```

## `presetPanel.newFolder()`

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.newFolder( [folderPath] )
```

### Paramètres

**folderPath** Chaîne spécifiant où un nouveau dossier doit être ajouté dans le panneau Présélections de mouvement et le nom du nouveau dossier. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur booléenne, `true` si le dossier a bien été ajouté, `false` dans le cas contraire.

### Description

Méthode qui crée un dossier dans l'arborescence de dossiers du panneau Présélections de mouvement. Cette méthode ne permet de créer qu'un seul nouveau niveau de dossiers. C'est-à-dire que si vous transmettez « Présélections personnalisées/Mon premier dossier/Mon second dossier » pour *folderPath*, « Présélections personnalisées/Mon premier dossier » doit exister dans l'arborescence.

Si vous ne transmettez pas de valeur pour *folderPath*, un dossier nommé « Dossier sans nom *n* » est créé au premier niveau sous « Présélections personnalisées », où *n* est incrémenté à chaque ajout d'un dossier de cette manière.

**Remarque :** *vous ne pouvez pas ajouter de dossiers dans le dossier Présélections par défaut.*

### Exemple

L'exemple suivant ajoute un dossier nommé `Bouncing` (Rebond) au-dessous du dossier `Custom Presets` (Présélections personnalisées) :

```
fl.presetPanel.newFolder("Custom Presets/Bouncing");
```

### Voir aussi

[`presetPanel.addNewItem\(\)`](#)

## `presetPanel.renameItem()`

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.renameItem(newName)
```

### Paramètres

**newName** Chaîne qui spécifie le nouveau nom de la présélection ou du dossier.

### Valeur renvoyée

Valeur booléenne, `true` si la présélection ou le dossier a bien été renommé(e), `false` dans le cas contraire.

### Description

Méthode qui remplace le nom de la présélection ou du dossier sélectionné(e) par un nom spécifié. Cette méthode fonctionne uniquement si un seul dossier ou une seule présélection est sélectionné(e) dans le panneau Présélections personnalisées. Cette méthode échoue dans les cas suivants :

- Aucun élément n'est sélectionné.
- Plusieurs éléments sont sélectionnés.
- L'élément sélectionné est situé dans le dossier Présélections par défaut.
- Un élément nommé *newName* existe déjà au même emplacement que l'élément sélectionné.

### Exemple

L'exemple suivant renomme la présélection actuellement sélectionnée dans le dossier Présélections personnalisées en `Bounce Faster` (Rebond plus rapide).

```
var renamed = fl.presetPanel.renameItem("Bounce Faster");  
fl.trace(renamed);
```

## presetPanel.selectItem()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
presetPanel.selectItem(namePath [, bReplaceCurrentSelection [, bSelect] ])
```

### Paramètres

**namePath** Chaîne qui spécifie le nom et le chemin de l'élément à sélectionner dans le panneau Présélections de mouvement.

**bReplaceCurrentSelection** Valeur booléenne indiquant si l'élément spécifié remplace la sélection actuelle (`true`) ou lui est ajouté (`false`). Ce paramètre est facultatif ; la valeur par défaut est `true`.

**bSelect** Valeur booléenne qui spécifie si l'élément doit être sélectionné (`true`) ou désélectionné (`false`). Ce paramètre est facultatif ; la valeur par défaut est `true`. Si vous transmettez `false` pour *bSelect*, la valeur de *bReplaceCurrentSelection* est ignorée.

### Valeur renvoyée

Valeur booléenne, `true` si l'élément a bien été sélectionné ou désélectionné, `false` dans le cas contraire.

### Description

Méthode qui sélectionne ou désélectionne un élément dans le panneau Présélections de mouvement, éventuellement en remplaçant les éléments actuellement sélectionnés.

**Exemple**

Le code suivant ajoute la présélection `fly-in-blur-right` (entrée rapide-flou-droite) aux présélections actuellement sélectionnées (le cas échéant) dans le panneau Présélections de mouvement :

```
fl.presetPanel.selectItem("Default Presets/fly-in-blur-right", false);
```

# Chapitre 35 : Objet Rectangle

**Héritage**    [Objet Element](#) > [Objet Shape](#) > [Objet Rectangle](#)

## Disponibilité

Flash CS3 Professional.

## Description

L'objet Rectangle est une forme tracée à l'aide de l'outil Rectangle primitif. Pour savoir si un élément est un objet Rectangle, utilisez [shape.isRectangleObject](#).

## Résumé des propriétés

Outre celles de l'objet [Objet Shape](#), l'objet Rectangle gère les propriétés suivantes : Pour définir les propriétés d'un objet Rectangle, utilisez [document.setRectangleObjectProperty\(\)](#).

Propriété	Description
<a href="#">RectangleObject.bottomLeftRadius</a>	Lecture seule ; valeur en virgule flottante qui définit le rayon de l'angle inférieur gauche de l'objet Rectangle.
<a href="#">RectangleObject.bottomRightRadius</a>	Lecture seule ; valeur en virgule flottante qui définit le rayon de l'angle inférieur droit de l'objet Rectangle.
<a href="#">RectangleObject.lockFlag</a>	Lecture seule ; valeur booléenne qui indique si les valeurs des rayons des différents angles du rectangle peuvent différer.
<a href="#">RectangleObject.topLeftRadius</a>	Lecture seule ; valeur en virgule flottante qui définit le rayon de tous les angles du rectangle, ou seulement le rayon de l'angle supérieur gauche de l'objet Rectangle.
<a href="#">RectangleObject.topRightRadius</a>	Lecture seule ; valeur en virgule flottante qui définit le rayon de l'angle supérieur droit de l'objet Rectangle.

## RectangleObject.bottomLeftRadius

### Disponibilité

Flash CS3 Professional.

### Utilisation

[RectangleObject.bottomLeftRadius](#)

### Description

Propriété en lecture seule ; valeur en virgule flottante qui définit le rayon de l'angle inférieur gauche de l'objet Rectangle. Si [RectangleObject.lockFlag](#) est `true`, tenter de définir cette valeur n'a aucun effet.

Pour définir cette valeur, utilisez [document.setRectangleObjectProperty\(\)](#).

#### Voir aussi

`document.setRectangleObjectProperty()`, `RectangleObject.bottomRightRadius`,  
`RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`, `RectangleObject.topRightRadius`

## RectangleObject.bottomRightRadius

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

`RectangleObject.bottomRightRadius`

#### Description

Propriété en lecture seule ; valeur en virgule flottante qui définit le rayon de l'angle inférieur droit de l'objet Rectangle. Si `RectangleObject.lockFlag` est défini sur `true`, tenter de définir cette valeur n'a aucun effet.

Pour définir cette valeur, utilisez `document.setRectangleObjectProperty()`.

#### Voir aussi

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`,  
`RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`, `RectangleObject.topRightRadius`

## RectangleObject.lockFlag

#### Disponibilité

Flash CS3 Professional.

#### Utilisation

`RectangleObject.lockFlag`

#### Description

Propriété en lecture seule ; valeur booléenne déterminant si les valeurs des rayons des différents angles du rectangle peuvent être différentes. Si cette valeur est `true`, les angles prennent tous la valeur `RectangleObject.topLeftRadius`. Si elle est `false`, chaque rayon d'angle peut être défini séparément.

Pour définir cette valeur, utilisez `document.setRectangleObjectProperty()`.

#### Voir aussi

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`,  
`RectangleObject.bottomRightRadius`, `RectangleObject.topLeftRadius`,  
`RectangleObject.topRightRadius`

## RectangleObject.topLeftRadius

### Disponibilité

Flash CS3 Professional.

### Utilisation

`RectangleObject.topLeftRadius`

### Description

Propriété en lecture seule ; valeur en virgule flottante qui définit le rayon de tous les angles du rectangle (si la valeur de `RectangleObject.lockFlag` est `true`), ou seulement le rayon de l'angle supérieur gauche (si la valeur de `RectangleObject.lockFlag` est `false`).

Pour définir cette valeur, utilisez `document.setRectangleObjectProperty()`.

### Voir aussi

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`,  
`RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`, `RectangleObject.topRightRadius`

## RectangleObject.topRightRadius

### Disponibilité

Flash CS3 Professional.

### Utilisation

`RectangleObject.topRightRadius`

### Description

Propriété en lecture seule ; valeur en virgule flottante qui définit le rayon de l'angle supérieur droit de l'objet Rectangle. Si `RectangleObject.lockFlag` est `true`, tenter de définir cette valeur n'a aucun effet.

Pour définir cette valeur, utilisez `document.setRectangleObjectProperty()`.

### Voir aussi

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`,  
`RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`

# Chapitre 36 : Objet Shape

**Héritage**    [Objet Element](#) > Objet Shape

## Disponibilité

Flash MX 2004.

## Description

L'objet Shape est une sous-classe de l'objet Element. L'objet Shape offre un contrôle plus précis que les API de dessin pour la manipulation ou la création de formes géométriques sur la scène. Cette précision du contrôle est nécessaire pour permettre aux scripts de créer des effets utiles et autres commandes de dessin (voir [Objet Element](#)).

Pour fonctionner correctement, toutes les méthodes et propriétés de Shape qui modifient une forme ou l'un de ses composants doivent être placées entre des appels de `shape.beginEdit()` et `shape.endEdit()`.

## Résumé des méthodes

Outre celles de l'objet Element, l'objet Shape gère les méthodes suivantes :

Méthode	Description
<code>shape.getCubicSegmentPoints()</code>	Renvoie un tableau de points définissant une courbe cubique.
<code>shape.beginEdit()</code>	Définit le début d'une session de modification.
<code>shape.deleteEdge()</code>	Supprime le bord spécifié.
<code>shape.endEdit()</code>	Définit la fin d'une session de modification de la forme.

## Résumé des propriétés

Outre celles de l'objet Element, l'objet Shape gère les propriétés suivantes :

Propriété	Description
<code>shape.contours</code>	Lecture seule ; tableau des objets Contour de la forme (voir <a href="#">Objet Contour</a> ).
<code>shape.edges</code>	Lecture seule ; tableau des objets Edge (voir <a href="#">Objet Edge</a> ).
<code>shape.isDrawingObject</code>	Lecture seule ; si sa valeur est définie sur true, la forme est un objet de dessin.
<code>shape.isGroup</code>	Lecture seule ; si sa valeur est définie sur true, la forme est un groupe.
<code>shape.isOvalObject</code>	Lecture seule ; si la valeur est définie sur true, la forme est un objet Oval de type primitif (créé à l'aide de l'outil Oval).
<code>shape.isRectangleObject</code>	Lecture seule ; si la valeur est définie sur true, la forme est un objet Rectangle de type primitif (créé à l'aide de l'outil Rectangle).
<code>shape.members</code>	Tableau des objets du groupe actuellement sélectionné.
<code>shape.numCubicSegments</code>	Lecture seule ; nombre de segments cubiques de la forme.
<code>shape.vertices</code>	Lecture seule ; tableau des objets Vertex (voir <a href="#">Objet Vertex</a> ).



## shape.beginEdit()

### Disponibilité

Flash MX 2004.

### Utilisation

```
shape.beginEdit();
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit le début d'une session de modification. Cette méthode doit être appelée avant toute commande modifiant l'objet Shape ou l'un de ses composants.

### Exemple

L'exemple suivant supprime le premier contour du tableau des contours dans la forme actuellement sélectionnée :

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

## shape.contours

### Disponibilité

Flash MX 2004.

### Utilisation

```
shape.contours
```

### Description

Propriété en lecture seule ; tableau d'objets Contour de la forme (voir [Objet Contour](#)).

### Exemple

L'exemple suivant enregistre dans la variable *c* le premier contour du tableau des contours, puis enregistre l'[Objet halfEdge](#) de ce contour dans la variable *he* :

```
var c = fl.getDocumentDOM().selection[0].contours[0];
var he = c.getHalfEdge();
```

## shape.deleteEdge()

### Disponibilité

Flash MX 2004.

### Utilisation

```
shape.deleteEdge(index)
```

### Paramètres

**index** Index de base zéro qui spécifie le bord à supprimer du tableau [shape.edges](#). Cette méthode modifie la longueur du tableau `shape.edges`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime le contour spécifié. Il est nécessaire d'appeler [shape.beginEdit\(\)](#) avant d'utiliser cette méthode.

### Exemple

L'exemple suivant supprime le premier contour du tableau des contours de la forme actuellement sélectionnée :

```
var shape = fl.getDocumentDOM().selection[0];  
shape.beginEdit();  
shape.deleteEdge(0);  
shape.endEdit();
```

## shape.edges

### Disponibilité

Flash MX 2004.

### Utilisation

```
shape.edges
```

### Description

Propriété en lecture seule ; tableau d'objets Edge (voir [Objet Edge](#)).

## shape.endEdit()

### Disponibilité

Flash MX 2004.

### Utilisation

```
shape.endEdit()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la fin d'une session de modification. Toutes les modifications apportées à l'objet Shape ou à l'un de ses composants sont appliquées à la forme. Cette méthode doit être appelée après toute commande modifiant l'objet Shape ou l'un de ses composants.

### Exemple

L'exemple suivant supprime le premier contour du tableau des contours dans la forme actuellement sélectionnée :

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

## shape.getCubicSegmentPoints()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
shape.getCubicSegmentPoints(cubicSegmentIndex)
```

### Paramètres

**cubicSegmentIndex** Nombre entier qui spécifie le segment cubique pour lequel les points sont renvoyés.

### Valeur renvoyée

Tableau de points qui définit une courbe cubique pour l'objet Edge correspondant au *cubicSegmentIndex* spécifié (voir [edge.cubicSegmentIndex](#)).

### Description

Méthode qui renvoie un tableau de points définissant une courbe cubique.

### Exemple

L'exemple suivant affiche les valeurs *x* et *y* de chaque point de la courbe cubique du premier contour de la sélection :

```
var elem = fl.getDocumentDOM().selection[0];
var index = elem.edges[0].cubicSegmentIndex;
var cubicPoints = elem.getCubicSegmentPoints(index);
for (i=0; i<cubicPoints.length; i++) {
    fl.trace("index " + i + " x: " + cubicPoints[i].x + " y: " + cubicPoints[i].y);
}
```

## shape.isDrawingObject

### Disponibilité

Flash 8.

### Utilisation

`shape.isDrawingObject`

### Description

Propriété en lecture seule ; si elle est `true`, la forme est un objet de dessin.

### Exemple

L'exemple suivant stocke dans la variable `sel` le premier objet sélectionné, puis utilise les propriétés `element.elementType` et `shape.isDrawingObject` pour déterminer si l'élément sélectionné est un objet de dessin :

```
var sel = fl.getDocumentDOM().selection[0];  
var shapeDrawingObject = (sel.elementType == "shape") && sel.isDrawingObject;  
fl.trace(shapeDrawingObject);
```

### Voir aussi

`document.crop()`, `document.deleteEnvelope()`, `document.intersect()`, `document.punch()`,  
`document.union()`, `shape.isGroup`

## shape.isGroup

### Disponibilité

Flash MX 2004.

### Utilisation

`shape.isGroup`

### Description

Propriété en lecture seule ; si elle est `true`, la forme est un groupe. Un groupe peut contenir différents types d'éléments, par exemple des éléments de texte et des symboles. Le groupe lui-même est cependant considéré comme une forme, et vous pouvez utiliser la propriété `shape.isGroup` quel que soit le type des éléments du groupe.

### Exemple

L'exemple suivant stocke dans la variable `sel` le premier objet sélectionné, puis utilise les propriétés `element.elementType` et `shape.isGroup` pour déterminer si l'élément sélectionné est un groupe :

```
var sel = fl.getDocumentDOM().selection[0];  
var shapeGroup = (sel.elementType == "shape") && sel.isGroup;  
fl.trace(shapeGroup);
```

### Voir aussi

`shape.isDrawingObject`

## shape.isOvalObject

### Disponibilité

Flash CS3 Professional.

### Utilisation

`shape.isOvalObject`

### Description

Propriété en lecture seule ; si la valeur est `true`, la forme est un objet Oval de type primitif (créé à l'aide de l'outil Primitive ovale).

### Exemple

L'exemple suivant affiche `"true"` si le premier élément sélectionné est un objet Oval de type primitif, `"false"` dans le cas contraire :

```
var sel = fl.getDocumentDOM().selection[0];  
fl.trace(sel.isOvalObject);
```

### Voir aussi

[shape.isRectangleObject](#)

## shape.isRectangleObject

### Disponibilité

Flash CS3 Professional.

### Utilisation

`shape.isRectangleObject`

### Description

Propriété en lecture seule ; si la valeur est `true`, la forme est un objet Rectangle de type primitif (créé à l'aide de l'outil Primitive rectangle).

### Exemple

L'exemple suivant affiche `"true"` si le premier élément sélectionné est un objet Rectangle de type primitif, `"false"` dans le cas contraire :

```
var sel = fl.getDocumentDOM().selection[0];  
fl.trace(sel.isRectangleObject);
```

### Voir aussi

[shape.isOvalObject](#)

## shape.members

### Disponibilité

Flash CS4 Professional.

### Utilisation

`shape.members`

### Description

Propriétés en lecture seule ; tableau des objets du groupe actuellement sélectionné. Cette propriété n'est disponible que si la valeur de `shape.isGroup` est `true`. Les formes brutes du groupe ne sont pas incluses dans le tableau `shape.members`.

Par exemple, si le groupe contient trois objets de dessin et trois formes brutes, le tableau `shape.members` contient trois entrées, une pour chaque objet de dessin. Si le groupe ne contient que des formes brutes, le tableau est vide.

### Exemple

Le code suivant affiche le nombre de segments cubiques de chaque objet de dessin du groupe actuellement sélectionné :

```
var shapesArray = fl.getDocumentDOM().selection[0].members;
for (i=0; i<shapesArray.length; i++) {
    fl.trace(shapesArray[i].numCubicSegments);
}
```

### Voir aussi

[shape.isGroup](#)

## shape.numCubicSegments

### Disponibilité

Flash CS4 Professional.

### Utilisation

`shape.numCubicSegments`

### Description

Propriété en lecture seule ; nombre de segments cubiques de la forme.

### Exemple

En supposant qu'une forme carrée ou rectangulaire soit sélectionnée, le code suivant affiche 4 dans le panneau Sortie :

```
var theShape = fl.getDocumentDOM().selection[0];
fl.trace(theShape.numCubicSegments);
```

## shape.vertices

### Disponibilité

Flash MX 2004.

### Utilisation

`shape.vertices`

### Description

Propriété en lecture seule ; tableau d'objets Vertex (voir [Objet Vertex](#)).

### Exemple

L'exemple suivant stocke le premier objet sélectionné dans la variable `someShape`, puis affiche son nombre de sommets dans le panneau Sortie :

```
var someShape = fl.getDocumentDOM().selection[0];  
fl.trace("The shape has " + someShape.vertices.length + " vertices.");
```

# Chapitre 37 : Objet SoundItem

**Héritage**    [Objet Item](#) > Objet SoundItem

## Disponibilité

Flash MX 2004.

## Description

L'objet `SoundItem` est une sous-classe de l'objet `Item`. Il représente un élément de bibliothèque qui est utilisé pour créer un son. Voir aussi [frame.soundLibraryItem](#) et [Objet Item](#).

## Résumé des méthodes

Outre les méthodes de l'objet `Item`, l'objet `SoundItem` gère la méthode suivante :

Propriété	Description
<code>soundItem.exportToFile()</code>	Exporte l'élément spécifié dans un fichier QuickTime sous Macintosh, ou dans un fichier WAV ou QT sous Windows.

## Résumé des propriétés

Outre celles de l'objet `Item`, l'objet `SoundItem` gère les propriétés suivantes :

Propriété	Description
<code>soundItem.bitRate</code>	Chaîne spécifiant la vitesse de transmission d'un son figurant dans la bibliothèque. Uniquement disponible pour la compression MP3.
<code>soundItem.bits</code>	Chaîne spécifiant la vitesse de transmission d'un son enregistré avec compression en ADPCM et figurant dans la bibliothèque.
<code>soundItem.compressionType</code>	Chaîne spécifiant le type de compression d'un son figurant dans la bibliothèque.
<code>soundItem.convertStereoToMono</code>	Valeur booléenne uniquement disponible pour les types de compression MP3 et Raw.
<code>soundItem.fileLastModifiedDate</code>	Lecture seule ; chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1er janvier 1970 et la date de modification du fichier original (sur disque) au moment où il a été importé dans la bibliothèque.
<code>soundItem.originalCompressionType</code>	Lecture seule ; chaîne indiquant si l'élément spécifié est importé sous forme de fichier mp3.
<code>soundItem.quality</code>	Chaîne spécifiant la qualité de lecture d'un son figurant dans la bibliothèque. Uniquement disponible pour la compression MP3.
<code>soundItem.sampleRate</code>	Chaîne spécifiant la fréquence d'échantillonnage du clip audio.
<code>soundItem.sourceFileExists</code>	Lecture seule ; valeur booléenne qui spécifie si le fichier importé dans la bibliothèque existe toujours à l'emplacement d'où il provient.



Propriété	Description
<code>soundItem.sourceFileIsCurrent</code>	Lecture seule ; valeur booléenne qui spécifie si la date de modification du fichier de l'élément de la bibliothèque correspond à celle trouvée sur le disque du fichier importé.
<code>soundItem.sourceFilePath</code>	Lecture seule ; chaîne de type « fichier:///URI » qui représente le chemin et le nom du fichier importé dans la bibliothèque.
<code>soundItem.useImportedMP3Quality</code>	Valeur booléenne ; si elle est <code>true</code> , toutes les autres propriétés sont ignorées, et la qualité MP3 importée est utilisée.

## soundItem.bitRate

### Disponibilité

Flash MX 2004.

### Utilisation

`soundItem.bitRate`

### Description

Propriété ; chaîne spécifiant la vitesse de transmission d'un son figurant dans la bibliothèque. Cette propriété est uniquement disponible pour la compression MP3. Les valeurs gérées sont "8Kbits/s", "16Kbits/s", "20Kbits/s", "24Kbits/s", "32Kbits/s", "48Kbits/s", "56Kbits/s", "64Kbits/s", "80Kbits/s", "112Kbits/s", "128Kbits/s" et "160Kbits/s". Les sons stéréo exportés à 8 ou 16 Kbits/s sont convertis en mono. Cette propriété a la valeur `undefined` pour les autres types de compression.

Si vous voulez spécifier une valeur pour cette propriété, définissez `soundItem.useImportedMP3Quality` sur `false`.

### Exemple

L'exemple suivant affiche la valeur de la propriété `bitRate` dans le panneau Sortie si l'élément spécifié possède le type de compression MP3 :

```
alert(fl.getDocumentDOM().library.items[0].bitRate);
```

### Voir aussi

`soundItem.compressionType`, `soundItem.convertStereoToMono`

## soundItem.bits

### Disponibilité

Flash MX 2004.

### Utilisation

`soundItem.bits`

### Description

Propriété ; chaîne spécifiant la vitesse de transfert d'un son enregistré avec compression ADPCM et figurant dans la bibliothèque. Les valeurs gérées sont les suivantes : "2bit", "3bit", "4bit" et "5bit".

Si vous souhaitez spécifier une valeur pour cette propriété, définissez `soundItem.useImportedMP3Quality` sur `false`.

### Exemple

L'exemple suivant affiche le nombre de bits de transfert dans le panneau Sortie si l'élément actuellement sélectionné dans la bibliothèque possède le type de compression ADPCM :

```
alert(fl.getDocumentDOM().library.items[0].bits);
```

### Voir aussi

[soundItem.compressionType](#)

## soundItem.compressionType

### Disponibilité

Flash MX 2004.

### Utilisation

`soundItem.compressionType`

### Description

Propriété ; chaîne spécifiant le type de compression d'un son figurant dans la bibliothèque. Les valeurs possibles sont : "Default", "ADPCM", "MP3", "Raw" (brut) et "Speech" (parole).

Si vous souhaitez spécifier une valeur pour cette propriété, définissez `soundItem.useImportedMP3Quality` sur `false`.

### Exemple

L'exemple suivant modifie le type de compression d'un élément de la bibliothèque en lui donnant la valeur `Raw` :

```
fl.getDocumentDOM().library.items[0].compressionType = "Raw";
```

L'exemple suivant modifie le type de compression des éléments de la bibliothèque en leur donnant la valeur `Speech` :

```
fl.getDocumentDOM().library.getSelectedItems().compressionType = "Speech";
```

### Voir aussi

[soundItem.originalCompressionType](#)

## soundItem.convertStereoToMono

### Disponibilité

Flash MX 2004.

### Utilisation

```
soundItem.convertStereoToMono
```

### Description

Propriété ; valeur booléenne uniquement disponible pour les types de compression MP3 et Raw. Si la valeur de cette propriété est `true`, les sons stéréo sont convertis en mono ; si elle est `false` le son stéréo est préservé. Pour le type de compression MP3, si `soundItem.bitRate` est inférieur à 20 Kbps/s, cette propriété n'est pas prise en compte et prend la valeur `true` (voir [soundItem.bitRate](#)).

Si vous souhaitez spécifier une valeur pour cette propriété, définissez `soundItem.useImportedMP3Quality` sur `false`.

### Exemple

L'exemple suivant convertit en mono un élément de la bibliothèque si cet élément possède le type de compression MP3 ou Raw :

```
fl.getDocumentDOM().library.items[0].convertStereoToMono = true;
```

### Voir aussi

[soundItem.compressionType](#)

## soundItem.exportToFile()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
soundItem.exportToFile(fileURI)
```

### Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et spécifiant le chemin et le nom du fichier exporté.

### Valeur renvoyée

Valeur booléenne `true` si le fichier a bien été exporté, `false` dans le cas contraire.

### Description

Méthode qui exporte l'élément spécifié dans un fichier QuickTime sous Macintosh, ou dans un fichier WAV ou QT sous Windows. Les fichiers QuickTime ou QT exportés ne contiennent que du son ; la vidéo n'est pas exportée. Les paramètres d'exportation dépendent de l'élément exporté.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément audio, le code suivant l'exporte sous forme de fichier WAV :

```
var soundFileURL = "file:///C:/out.wav";  
var libItem = fl.getDocumentDOM().library.items[0];  
libItem.exportToFile(soundFileURL);
```

## soundItem.fileLastModifiedDate

### Disponibilité

Flash CS4 Professional.

### Utilisation

`soundItem.fileLastModifiedDate`

### Description

Propriété en lecture seule ; chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1 janvier 1970 et la date de modification du fichier original (sur disque) au moment où le fichier a été importé dans la bibliothèque. Si le fichier n'existe plus, cette valeur est « 00000000 ».

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément audio, le code suivant affiche un nombre hexadécimal tel que décrit ci-dessus :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("Mod date when imported = " + libItem.fileLastModifiedDate);
```

### Voir aussi

[soundItem.sourceFileExists](#), [soundItem.sourceFileIsCurrent](#), [soundItem.sourceFilePath](#),  
[FLfile.getModificationDate\(\)](#)

## soundItem.originalCompressionType

### Disponibilité

Flash CS4 Professional.

### Utilisation

`soundItem.originalCompressionType`

### Description

Propriété en lecture seule ; chaîne indiquant si l'élément spécifié est importé sous forme de fichier mp3. Les valeurs possibles de cette propriété sont « RAW » et « MP3 ».

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément audio, le code suivant affiche « MP3 » si le fichier a été importé dans la bibliothèque sous forme de fichier MP3, ou « RAW » si ce n'est pas le cas :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("Imported compression type = " + libItem.originalCompressionType);
```

### Voir aussi

[soundItem.compressionType](#)

## soundItem.quality

### Disponibilité

Flash MX 2004.

### Utilisation

`soundItem.quality`

### Description

Propriété ; chaîne spécifiant la qualité de lecture d'un son figurant dans la bibliothèque. Cette propriété est uniquement disponible pour la compression MP3. Les valeurs gérées sont : "Fast" (rapide), "Medium" (moyenne) et "Best" (optimale).

Si vous souhaitez spécifier une valeur pour cette propriété, définissez `soundItem.useImportedMP3Quality` sur `false`.

### Exemple

L'exemple suivant définit sur `Best` la qualité de lecture d'un élément de la bibliothèque si cet élément possède le type de compression MP3 :

```
fl.getDocumentDOM().library.items[0].quality = "Best";
```

### Voir aussi

[soundItem.compressionType](#)

## soundItem.sampleRate

### Disponibilité

Flash MX 2004.

### Utilisation

`soundItem.sampleRate`

### Description

Propriété ; chaîne spécifiant la fréquence d'échantillonnage du clip audio. Cette propriété est réservée aux types de compression ADPCM, Raw et Speech. Les valeurs gérées sont "5kHz", "11kHz", "22kHz" et "44kHz".

Si vous souhaitez spécifier une valeur pour cette propriété, définissez `soundItem.useImportedMP3Quality` sur `false`.

### Exemple

L'exemple suivant définit le taux d'échantillonnage d'un élément de la bibliothèque sur 5 kHz si cet élément possède le type de compression ADPCM, Raw ou Speech :

```
fl.getDocumentDOM().library.items[0].sampleRate = "5 kHz";
```

### Voir aussi

[soundItem.compressionType](#)

## soundItem.sourceFileExists

### Disponibilité

Flash CS4 Professional.

### Utilisation

`soundItem.sourceFileExists`

### Description

Propriété en lecture seule : valeur booléenne `true` si le fichier importé dans la bibliothèque existe toujours à l'emplacement d'où il provient ; `false` dans le cas contraire.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément audio, le code suivant affiche « true » si le fichier importé dans la bibliothèque existe toujours :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("sourceFileExists = " + libItem.sourceFileExists);
```

### Voir aussi

[soundItem.sourceFileIsCurrent](#), [soundItem.sourceFilePath](#)

## soundItem.sourceFileIsCurrent

### Disponibilité

Flash CS4 Professional.

### Utilisation

`soundItem.sourceFileIsCurrent`

### Description

Propriété en lecture seule ; valeur booléenne, `true` si la date de modification du fichier de l'élément de la bibliothèque correspond à la date de modification sur disque du fichier importé ; `false` dans le cas contraire.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément audio, le code suivant affiche « true » si le fichier importé n'a pas été modifié sur le disque depuis son importation :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("fileIsCurrent = " + libItem.sourceFileIsCurrent);
```

### Voir aussi

[soundItem.fileLastModifiedDate](#), [soundItem.sourceFilePath](#)

## soundItem.sourceFilePath

### Disponibilité

Flash CS4 Professional.

### Utilisation

`soundItem.sourceFilePath`

### Description

Propriété en lecture seule : chaîne de type « fichier:///URI » qui représente le chemin et le nom du fichier importé dans la bibliothèque.

### Exemple

L'exemple suivant affiche le nom et le chemin du fichier source de tout élément de la bibliothèque de type "sound" :

```
for (idx in fl.getDocumentDOM().library.items) {  
  if (fl.getDocumentDOM().library.items[idx].itemType == "sound") {  
    var myItem = fl.getDocumentDOM().library.items[idx];  
    fl.trace(myItem.name + " source is " + myItem.sourceFilePath);  
  }  
}
```

### Voir aussi

[soundItem.sourceFileExists](#)

## soundItem.useImportedMP3Quality

### Disponibilité

Flash MX 2004.

### Utilisation

`soundItem.useImportedMP3Quality`

### Description

Propriété ; valeur booléenne. Si elle est `true`, toutes les autres propriétés sont ignorées et la qualité du fichier MP3 importé est utilisée.

### Exemple

L'exemple suivant modifie la qualité d'un élément de la bibliothèque en utilisant la qualité du fichier MP3 importé :

```
fl.getDocumentDOM().library.items[0].useImportedMP3Quality = true;
```

### Voir aussi

[soundItem.compressionType](#)

# Chapitre 38 : Objet Stroke

## Disponibilité

Flash MX 2004.

## Description

L'objet Stroke contient tous les paramètres d'un trait, y compris les paramètres personnalisés. Cet objet représente les informations contenues dans l'inspecteur Propriétés. En combinant l'objet Stroke et la méthode `document.setCustomStroke()`, vous pouvez modifier les paramètres du trait pour le panneau Outils, l'inspecteur Propriétés et la sélection actuelle. À l'aide de la méthode `document.getCustomStroke()`, vous pouvez également récupérer les paramètres de trait configurés dans le panneau Outils, l'inspecteur Propriétés et la sélection actuelle.

Cet objet comporte toujours les quatre propriétés suivantes : `style`, `thickness`, `color` et `breakAtCorners`. (Dans Flash CS3, la propriété `breakAtCorners` a été déconseillée en faveur de `stroke.joinType`.) D'autres propriétés peuvent être définies, selon la valeur de la propriété `stroke.style`.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Stroke :

Propriété	Description
<code>stroke.breakAtCorners</code>	Valeur booléenne, identique au paramètre Angles saillants de la boîte de dialogue personnalisée Style de trait.
<code>stroke.capType</code>	Chaîne qui spécifie le type d'extrémité du trait.
<code>stroke.color</code>	Valeur hexadécimale ou entier représentant la couleur du trait.
<code>stroke.curve</code>	Chaîne spécifiant le type de trame du trait.
<code>stroke.dash1</code>	Entier spécifiant la longueur de la partie pleine d'une ligne pointillée.
<code>stroke.dash2</code>	Entier spécifiant la longueur de la partie vide d'une ligne pointillée.
<code>stroke.density</code>	Chaîne spécifiant la densité d'une ligne pointillée fine.
<code>stroke.dotSize</code>	Chaîne spécifiant la taille des points d'une ligne pointillée fine.
<code>stroke.dotSpace</code>	Entier spécifiant l'espacement des points d'une ligne pointillée.
<code>stroke.hatchThickness</code>	Chaîne spécifiant l'épaisseur d'une ligne tramée.
<code>stroke.jiggle</code>	Chaîne spécifiant la propriété de déplacement d'une ligne tramée.
<code>stroke.joinType</code>	Chaîne qui spécifie le type de jointure du trait.
<code>stroke.length</code>	Chaîne spécifiant la longueur d'une ligne tramée.
<code>stroke.miterLimit</code>	Nombre en virgule flottante spécifiant l'angle au-dessus duquel l'extrémité de la pointe sera tronquée par un segment.
<code>stroke.pattern</code>	Chaîne spécifiant le motif d'une ligne irrégulière.
<code>stroke.rotate</code>	Chaîne spécifiant la rotation d'une ligne tramée.
<code>stroke.scaleType</code>	Chaîne spécifiant le type de redimensionnement à appliquer au trait.
<code>stroke.shapeFill</code>	Objet Fill qui représente les paramètres de remplissage du trait.



Propriété	Description
<code>stroke.space</code>	Chaîne spécifiant l'espacement d'une ligne tramée.
<code>stroke.strokeHinting</code>	Valeur booléenne indiquant si les repères de trait doivent être activés pour ce trait.
<code>stroke.style</code>	Chaîne décrivant le style du trait.
<code>stroke.thickness</code>	Nombre entier spécifiant la taille du trait.
<code>stroke.variation</code>	Chaîne spécifiant la variation d'une ligne pointillée fine.
<code>stroke.waveHeight</code>	Chaîne spécifiant la hauteur d'onde d'une ligne irrégulière.
<code>stroke.waveLength</code>	Chaîne spécifiant la longueur d'onde d'une ligne irrégulière.

## stroke.breakAtCorners

### Disponibilité

Flash MX 2004. Délaissée dans Flash 3 au profit de `stroke.joinType`.

### Utilisation

```
stroke.breakAtCorners
```

### Description

Propriété : valeur booléenne. Cette propriété est identique au paramètre Angles saillants de la boîte de dialogue personnalisée Style de trait.

### Exemple

L'exemple suivant définit la propriété `breakAtCorners` sur `true` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.breakAtCorners = true;  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.capType

### Disponibilité

Flash 8.

### Utilisation

```
stroke.capType
```

### Description

Propriété ; chaîne qui spécifie le type d'extrémité du trait. Les valeurs acceptables sont "none" (aucune), "round" (arrondie) et "square" (carrée).

### Exemple

L'exemple suivant donne la valeur `round` au type d'extrémité du trait :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.capType = "round";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.color

### Disponibilité

Flash MX 2004. Dans Flash 8 et les versions ultérieures, cette propriété est délaissée au profit de `stroke.shapeFill.color`.

### Utilisation

`stroke.color`

### Description

Propriété ; couleur du trait, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

### Exemple

L'exemple suivant définit la couleur du trait :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.color = "#000000";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

### Voir aussi

[stroke.shapeFill](#)

## stroke.curve

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.curve`

### Description

Propriété ; chaîne spécifiant le type de trame du trait. Cette propriété ne peut être définie que si la propriété `stroke.style` a la valeur "hatched" (voir [stroke.style](#)). Les valeurs possibles sont "straight", "slight curve", "medium curve" et "very curved".

### Exemple

L'exemple suivant définit la propriété de courbe, entre autres, d'un trait ayant le style hatched :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "hatched";  
myStroke.curve = "straight";  
myStroke.space = "close";  
myStroke.jiggle = "wild";  
myStroke.rotate = "free";  
myStroke.length = "slight";  
myStroke.hatchThickness = "thin";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.dash1

### Disponibilité

Flash MX 2004.

### Utilisation

stroke.dash1

### Description

Propriété ; entier spécifiant la longueur de la partie pleine d'une ligne pointillée. Cette propriété n'est disponible que si la propriété `stroke.style` a la valeur `dashed` (voir [stroke.style](#)).

### Exemple

L'exemple suivant définit les propriétés `dash1` et `dash2` pour le style de trait `dashed` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "dashed";  
myStroke.dash1 = 1;  
myStroke.dash2 = 2;  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.dash2

### Disponibilité

Flash MX 2004.

### Utilisation

stroke.dash2

### Description

Propriété ; entier spécifiant la longueur de la partie vide d'une ligne pointillée. Cette propriété n'est disponible que si la propriété `stroke.style` a la valeur `dashed` (voir [stroke.style](#)).

### Exemple

Voir [stroke.dash1](#).

## stroke.density

### Disponibilité

Flash MX 2004.

### Utilisation

stroke.density

### Description

Propriété ; chaîne spécifiant la densité d'une ligne pointillée fine. Cette propriété est disponible uniquement si la propriété stroke.style a la valeur stipple (voir [stroke.style](#)). Les valeurs possibles sont "very dense", "dense", "sparse" et "very sparse".

### Exemple

L'exemple suivant définit sur sparse la propriété de densité du style de trait stipple :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "stipple";  
myStroke.dotSpace= 3;  
myStroke.variation = "random sizes";  
myStroke.density = "sparse";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.dotSize

### Disponibilité

Flash MX 2004.

### Utilisation

stroke.dotSize

### Description

Propriété ; chaîne spécifiant la taille des points d'une ligne pointillée fine. Cette propriété est disponible uniquement si la propriété stroke.style a la valeur stipple (voir [stroke.style](#)). Les valeurs possibles sont "tiny", "small", "medium" et "large".

L'exemple suivant définit sur tiny la propriété dotSize du style de trait de stipple :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "stipple";  
myStroke.dotSpace= 3;  
myStroke.dotsize = "tiny";  
myStroke.variation = "random sizes";  
myStroke.density = "sparse";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.dotSpace

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.dotSpace`

### Description

Propriété ; entier spécifiant l'espacement des points d'une ligne pointillée. Cette propriété n'est disponible que si la propriété `stroke.style` a pour valeur `dotted`. Voir [stroke.style](#).

### Exemple

L'exemple suivant donne la valeur 3 à la propriété `dotSpace` pour le style de trait `dotted` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "dotted";  
myStroke.dotSpace = 3;  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.hatchThickness

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.hatchThickness`

### Description

Propriété ; chaîne spécifiant l'épaisseur d'une ligne tramée. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `hatched` (voir [stroke.style](#)). Les valeurs possibles sont "hairline", "thin", "medium" et "thick".

### Exemple

L'exemple suivant donne la valeur `thin` à la propriété `hatchThickness` du style de trait `hatched` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "hatched";  
myStroke.curve = "straight";  
myStroke.space = "close";  
myStroke.jiggle = "wild";  
myStroke.rotate = "free";  
myStroke.length = "slight";  
myStroke.hatchThickness = "thin";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.jiggle

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.jiggle`

### Description

Propriété ; chaîne spécifiant la propriété de déplacement d'une ligne tramée. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `hatched` (voir [stroke.style](#)). Les valeurs possibles sont "none", "bounce", "loose" et "wild".

### Exemple

L'exemple suivant donne la valeur `wild` à la propriété `jiggle` du style de trait `hatched` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "hatched";  
myStroke.curve = "straight";  
myStroke.space = "close";  
myStroke.jiggle = "wild";  
myStroke.rotate = "free";  
myStroke.length = "slight";  
myStroke.hatchThickness = "thin";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.joinType

### Disponibilité

Flash 8.

### Utilisation

`stroke.joinType`

### Description

Propriété ; chaîne qui spécifie le type de jointure du trait. Les valeurs gérées sont "miter" (pointe), "round" (arrondie) et "bevel" (biseau).

### Voir aussi

[stroke.capType](#)

## stroke.length

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.length`

### Description

Propriété ; chaîne spécifiant la longueur d'une ligne tramée. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `hatched` (voir [stroke.style](#)). Les valeurs possibles sont les suivantes : `"equal"`, `"slight variation"`, `"medium variation"` et `"random"`. (La valeur `"random"` correspond en fait à la valeur `"medium variation"`.)

### Exemple

L'exemple suivant donne la valeur `slight` à la propriété `length` du style de trait `hatched` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight variation";
myStroke.hatchThickness = "thin";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.miterLimit

### Disponibilité

Flash 8.

### Utilisation

`stroke.miterLimit`

### Description

Propriété ; nombre en virgule flottante spécifiant l'angle au-dessus duquel l'extrémité de la pointe sera tronquée par un segment. Cela signifie que la pointe ne sera tronquée que si son angle est supérieur à la valeur transmise dans `miterLimit`.

### Exemple

L'exemple suivant donne la valeur 3 à la limite de pointe du trait. Si l'angle de la pointe est supérieur à 3, la pointe est tronquée.

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.miterLimit = 3;
var myStroke = fl.getDocumentDOM().setCustomStroke();
```

## stroke.pattern

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.pattern`

### Description

Propriété ; chaîne spécifiant le motif d'une ligne irrégulière. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `ragged` (voir [stroke.style](#)). Les valeurs possibles sont "solid", "simple", "random", "dotted", "random dotted", "triple dotted" et "random triple dotted".

### Exemple

L'exemple suivant donne la valeur `random` à la propriété `pattern` du style de trait `ragged` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "ragged";  
myStroke.pattern = "random";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.rotate

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.rotate`

### Description

Propriété ; chaîne spécifiant la rotation d'une ligne tramée. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `hatched` (voir [stroke.style](#)). Les valeurs possibles sont "none", "slight", "medium" et "free".

### Exemple

L'exemple suivant donne la valeur `free` à la propriété `rotate` du style de trait `hatched` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "hatched";  
myStroke.curve = "straight";  
myStroke.space = "close";  
myStroke.jiggle = "wild";  
myStroke.rotate = "free";  
myStroke.length = "slight";  
myStroke.hatchThickness = "thin";
```

## stroke.scaleType

### Disponibilité

Flash 8.



### Utilisation

`stroke.scaleType`

### Description

Propriété ; chaîne qui spécifie le type de redimensionnement à appliquer au trait. Les valeurs gérées sont "normal", "horizontal", "vertical" et "none".

### Exemple

L'exemple suivant définit le type de dimensionnement du trait sur horizontal :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.scaleType = "horizontal";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.shapeFill

### Disponibilité

Flash 8.

### Utilisation

`stroke.shapeFill`

### Description

Propriété ; objet [Objet Fill](#) qui représente le remplissage du trait.

### Exemple

L'exemple suivant spécifie les paramètres du remplissage et les applique au trait :

```
var fill = fl.getDocumentDOM().getCustomFill();  
fill.linearGradient = true;  
fill.colorArray = [ 00ff00, ff0000, ffffff ];  
var stroke = fl.getDocumentDOM().getCustomStroke();  
stroke.shapeFill = fill;  
fl.getDocumentDOM().setCustomStroke(stroke);
```

## stroke.space

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.space`

### Description

Propriété ; chaîne spécifiant l'espacement d'une ligne tramée. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `hatched` (voir [stroke.style](#)). Les valeurs possibles sont "very close", "close", "distant" et "very distant".

### Exemple

L'exemple suivant donne la valeur `close` à la propriété `space` du style de trait `hatched` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "hatched";  
myStroke.curve = "straight";  
myStroke.space = "close";  
myStroke.jiggle = "wild";  
myStroke.rotate = "free";  
myStroke.length = "slight";  
myStroke.hatchThickness = "thin";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.strokeHinting

### Disponibilité

Flash 8.

### Utilisation

`stroke.strokeHinting`

### Description

Propriété ; valeur booléenne indiquant si les repères de trait doivent être activés pour ce trait.

### Exemple

L'exemple suivant active les repères de trait pour ce trait :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.strokeHinting = true;  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.style

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.style`

### Description

Propriété ; chaîne décrivant le style de trait. Les valeurs possibles sont `"noStroke"`, `"solid"`, `"dashed"`, `"dotted"`, `"ragged"`, `"stipple"` et `"hatched"`. Certaines de ces valeurs nécessitent que des propriétés complémentaires de l'objet Stroke soient définies, comme décrit dans la liste suivante :

- Si cette propriété a la valeur `"solid"` ou `"noStroke"`, il n'existe pas d'autre propriété.
- Si la valeur est `dashed`, deux propriétés supplémentaires sont présentes : `dash1` et `dash2`.
- Si cette propriété a la valeur `dotted`, la propriété supplémentaire `dotSpace` est présente.

- Si la valeur est "ragged", trois propriétés supplémentaires, `pattern`, `waveHeight` et `waveLength`, sont présentes.
- Si la valeur est "stipple", trois propriétés supplémentaires, `dotSize`, `variation` et `density`, sont présentes.
- Si la valeur est "hatched", six propriétés supplémentaires sont présentes : `hatchThickness`, `space`, `jiggle`, `rotate`, `curve` et `length`.

### Exemple

L'exemple suivant donne la valeur `ragged` au style de trait :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "ragged";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.thickness

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.thickness`

### Description

Propriété ; nombre entier spécifiant la taille du trait.

### Exemple

L'exemple suivant donne la valeur 2 à la propriété `thickness` du trait :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.thickness = 2;  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.variation

### Disponibilité

Flash MX 2004.

### Utilisation

`stroke.variation`

### Description

Propriété ; chaîne spécifiant la variation d'une ligne pointillée fine. Cette propriété est disponible uniquement si la propriété `stroke.style` a la valeur `stipple` (voir [stroke.style](#)). Les valeurs possibles sont "one size", "small variation", "varied sizes" et "random sizes".

### Exemple

L'exemple suivant donne la valeur `random sizes` à la propriété de variation du style de trait `stipple` :

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "stipple";  
myStroke.dotSpace= 3;  
myStroke.variation = "random sizes";  
myStroke.density = "sparse";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.waveHeight

### Disponibilité

Flash MX 2004.

### Utilisation

stroke.waveHeight

### Description

Propriété ; chaîne spécifiant la hauteur d'onde d'une ligne irrégulière. Cette propriété est disponible uniquement si la propriété stroke.style a la valeur ragged (voir [stroke.style](#)). Les valeurs possibles sont "flat", "wavy", "very wavy" et "wild".

### Exemple

L'exemple suivant donne la valeur flat à la propriété waveHeight du style de trait ragged:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "ragged";  
myStroke.pattern = "random";  
myStroke.waveHeight = "flat";  
myStroke.waveLength = "short";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

## stroke.waveLength

### Disponibilité

Flash MX 2004.

### Utilisation

stroke.waveLength

### Description

Propriété ; chaîne spécifiant la longueur d'onde d'une ligne irrégulière. Cette propriété est disponible uniquement si la propriété stroke.style a la valeur ragged (voir [stroke.style](#)). Les valeurs possibles sont "very short", "short", "medium" et "long".

### Exemple

L'exemple suivant donne la valeur short à la propriété waveLength du style de trait ragged :

**Objet Stroke**

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "ragged";  
myStroke.pattern = "random";  
myStroke.waveHeight = 'flat';  
myStroke.waveLength = "short";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

# Chapitre 39 : Objet swfPanel

## Disponibilité

Flash CS4 Professional.

## Description

L'objet `swfPanel` représente un panneau Window SWF. Les panneaux Window SWF sont des fichiers SWF qui implémentent les applications que vous pouvez exécuter depuis l'environnement de programmation Flash. Ils sont disponibles depuis le menu Fenêtre > Autres panneaux. Par défaut, les panneaux Window SWF sont stockés dans un sous-dossier du dossier Configuration (voir « [Enregistrement des fichiers JSFL](#) » à la page 2). Par exemple, sous Windows XP, le chemin du dossier est *lecteur d'amorçage*\Documents and Settings\utilisateur\Local Settings\Application Data\Adobe\Flash CS4\langue\Configuration\WindowSWF. Un exemple de panneau Window SWF est disponible ; voir « [Exemple du panneau de tracé de bitmap](#) » à la page 14. Le tableau des panneaux Window SWF enregistrés est stocké dans la propriété `fl.swfPanels`.

## Résumé des méthodes

L'objet `swfPanel` gère la méthode suivante :

Méthode	Description
<code>swfPanel.call()</code>	Fonctionne en combinaison avec les méthodes <code>ActionScript ExternalInterface.addCallback()</code> et <code>MMExecute()</code> pour communiquer avec le panneau SWF depuis l'environnement de programmation.
« <code>swfPanel.setFocus()</code> » à la page 419	Définit le focus clavier sur le panneau SWF spécifié.

## Résumé des propriétés

L'objet `swfPanel` gère les propriétés suivantes :

Propriété	Description
<code>swfPanel.name</code>	Lecture seule ; chaîne représentant le nom du panneau Window SWF spécifié.
<code>swfPanel.path</code>	Lecture seule ; chaîne représentant le chemin du fichier SWF utilisé dans le panneau Window SWF spécifié.

# swfPanel.call()

## Disponibilité

Flash CS4 Professional.

## Utilisation

`swfPanel.call(request)`

## Paramètres

**request** Paramètres à transmettre à la fonction (voir la description et l'exemple ci-dessous).

### Valeur renvoyée

null ou une chaîne renvoyée par l'appel de la fonction. Le résultat de la fonction peut être une chaîne vide.

### Description

Méthode qui fonctionne en combinaison avec les méthodes ActionScript `ExternalInterface.addCallback()` et `MMExecute()` pour communiquer avec le panneau SWF depuis l'environnement de programmation.

### Exemple

L'exemple suivant décrit l'utilisation de code ActionScript et JavaScript pour la création d'un panneau Window SWF et la communication avec ce dernier depuis l'environnement de programmation.

- 1 Créez un fichier FLA ActionScript 3.0 et définissez sa couleur sur gris clair et sa taille sur 400 pixels de large et 250 pixels de haut.
- 2 Placez un champ de texte dynamique au centre de la scène, définissez son nom d'occurrence sur `myTextField` et entrez « Status » dans le champ de texte.
- 3 Définissez les autres propriétés de champs de texte de la façon suivante :
  - Centré
  - 355 pixels de large et 46 pixels de haut
  - Police Times New Roman, 28 points, rouge

- 4 Ajoutez le code ActionScript suivant :

```
// Here's the callback function to be called from JSAPI
function callMeFromJavascript(arg:String):void
{
    try {
        var name:String = String(arg);
        myTextField.text = name;
    } catch (e:Error) {
    }
}

// Expose the callback function as "callMySWF"
ExternalInterface.addCallback("callMySWF", callMeFromJavascript);

// run the JSAPI to wire up the callback
MMExecute("fl.runScript( fl.configURI + \"WindowSWF/fileOp.jsfl\" );");

MMExecute("fl.trace(\"AS3 File Status Panel Initialized\");");
```

- 5 Enregistrez le fichier sous le nom `fileStatus fla`, et publiez le fichier SWF avec les paramètres de publication par défaut.
- 6 Fermez Flash.
- 7 Copiez le fichier `fileStatus.swf` dans le dossier `WindowSWF`, un sous-dossier du dossier `Configuration` (voir « [Enregistrement des fichiers JSFL](#) » à la page 2). Par exemple, sous Windows XP, le chemin du dossier est *lecteur d'amorçage*\\Documents and Settings\\utilisateur\\Local Settings\\Application Data\\Adobe\\Flash CS4\\langue\\Configuration\\WindowSWF.
- 8 Démarrez Flash.
- 9 Créez un fichier JSFL avec le code suivant :

```
function callMyPanel(panelName, arg)
{
    if(fl.swfPanels.length > 0){
        for(x = 0; x < fl.swfPanels.length; x++){
            // look for a SWF panel of the specified name, then call the specified AS3
            function
            // in this example, the panel is named "test" and the AS3 callback is "callMySWF"
            if(fl.swfPanels[x].name == panelName) // name busted?
            {
                fl.swfPanels[x].call("callMySWF",arg);
                break;
            }
        }
    }
    else
        fl.trace("no panels");
}

// define the various handlers for events
documentClosedHandler = function () { callMyPanel("fileStatus", "Document Closed");};
fl.addEventListener("documentClosed", documentClosedHandler );

var dater = "New Document";
documentNewHandler = function () { callMyPanel("fileStatus", dater );};
fl.addEventListener("documentNew", documentNewHandler );

documentOpenedHandler = function () { callMyPanel("fileStatus", "Document Opened");};
fl.addEventListener("documentOpened", documentOpenedHandler );
```

**10** Enregistrez le fichier JSFL dans le même répertoire que le fichier SWF, sous le nom fileOp.jsfl.

**11** Sélectionnez Fenêtre > Autres panneaux > fileStatus.

A présent, lorsque vous créez, ouvrez et fermez des fichiers FLA, le panneau fileStatus affiche un message indiquant l'action effectuée.

## swfPanel.name

### Disponibilité

Flash CS4 Professional.

### Utilisation

swfPanel.name

### Description

Propriété en lecture seule ; chaîne représentant le nom du panneau Window SWF spécifié.

### Exemple

Le code suivant affiche le nom du premier panneau Window SWF enregistré dans le panneau Sortie :

```
fl.trace(fl.swfPanels[0].name);
```



**Voir aussi**

`swfPanel.path`, `fl.swfPanels`

## `swfPanel.path`

**Disponibilité**

Flash CS4 Professional.

**Utilisation**

`swfPanel.path`

**Description**

Propriété en lecture seule ; chaîne représentant le chemin du fichier SWF utilisé dans le panneau Window SWF spécifié.

**Exemple**

Le code suivant affiche dans le panneau Sortie le chemin du fichier SWF utilisé dans le premier panneau Window SWF enregistré :

```
fl.trace(fl.swfPanels[0].path);
```

**Voir aussi**

`swfPanel.name`, `fl.swfPanels`

## `swfPanel.setFocus()`

**Disponibilité**

Flash CS5.5 Professional.

**Utilisation**

`swfPanel.setFocus()`

**Description**

Méthode : définit le focus clavier sur le panneau SWF spécifié.

**Exemple**

Le code suivant définit le focus sur le panneau SWF intitulé « Projet » :

Appliquez les procédures suivantes avant d'exécuter cette commande :

- 1 Détachez le panneau Projet de façon à ce qu'il soit un panneau flottant.
- 2 Ouvrez la boîte de dialogue Créer un fichier à partir du panneau Projet et cliquez sur la scène.
- 3 Appuyez plusieurs fois sur la touche Tabulation pour vous assurer que le panneau Projet ne possède pas le focus.
- 4 Exécutez le script suivant à partir du menu Commandes (placez un fichier JSFL contenant le code ci-dessous dans le répertoire utilisateur/config/Commands) :

**Objet swfPanel**

- 5 Appuyez sur la touche Tabulation. Un curseur d'insertion apparaît dans l'un des champs de texte de la boîte de dialogue Créer un fichier.

```
flash.getSwfPanel ("Project").setFocus();
```

**Voir aussi**

[swfPanel.name](#), [fl.swfPanels](#)

# Chapitre 40 : Objet SymbolInstance

**Héritage**    [Objet Element](#) > [Objet Instance](#) > [Objet SymbolInstance](#)

## Disponibilité

Flash MX 2004.

## Description

SymbolInstance est une sous-classe de l'objet Instance et représente un symbole dans une image (voir [Objet Instance](#)).

## Résumé des propriétés

Outre celles de l'objet Instance, l'objet SymbolInstance possède les propriétés suivantes :

Propriété	Description
<a href="#">symbolInstance.accName</a>	Chaîne équivalente au champ Nom du panneau Accessibilité.
<a href="#">symbolInstance.actionScript</a>	Chaîne qui spécifie les actions affectées au symbole.
<a href="#">symbolInstance.backgroundColor</a>	Chaîne spécifiant la couleur du cache lorsque vous sélectionnez Opaque.
<a href="#">symbolInstance.bitmapRenderMode</a>	Chaîne spécifiant le type d'affichage d'une occurrence de symbole.
<a href="#">symbolInstance.blendMode</a>	Chaîne qui spécifie le mode de fusion à appliquer à un symbole de clip.
<a href="#">symbolInstance.buttonTracking</a>	Chaîne qui définit, pour les symboles de bouton uniquement, la même propriété que le menu contextuel Traiter comme bouton ou Traiter comme élément de menu dans l'inspecteur des propriétés.
<a href="#">symbolInstance.cacheAsBitmap</a>	Valeur booléenne spécifiant si la mise en cache des bitmaps à l'exécution est activée ou non.
<a href="#">symbolInstance.colorAlphaAmount</a>	Entier spécifiant une partie de la transformation de couleur de l'occurrence, en indiquant la valeur des paramètres avancés de l'effet Alpha; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles qui apparaissent à droite dans la boîte de dialogue.
<a href="#">symbolInstance.colorAlphaPercent</a>	Entier spécifiant certaines parties de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue.
<a href="#">symbolInstance.colorBlueAmount</a>	Entier spécifiant une partie de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés.
<a href="#">symbolInstance.colorBluePercent</a>	Entier spécifiant une partie de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés de l'occurrence, suivie du réglage des contrôles qui apparaissent à gauche dans la boîte de dialogue
<a href="#">symbolInstance.colorGreenAmount</a>	Entier spécifiant une partie de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés. Les valeurs autorisées sont comprises entre -255 et 255.

Propriété	Description
<code>symbolInstance.colorGreenPercent</code>	Partie de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue.
<code>symbolInstance.colorMode</code>	Chaîne spécifiant le mode colorimétrique tel qu'il est identifié dans le menu contextuel Couleurs de l'inspecteur Propriétés du symbole.
<code>symbolInstance.colorRedAmount</code>	Entier spécifiant une partie de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés de l'occurrence.
<code>symbolInstance.colorRedPercent</code>	Partie de la transformation de couleur de l'occurrence ; équivalent à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue.
<code>symbolInstance.description</code>	Chaîne équivalente au champ Description du panneau Accessibilité.
<code>symbolInstance.filters</code>	Tableau d'objets Filter (voir <a href="#">Objet Filter</a> ).
<code>symbolInstance.firstFrame</code>	Entier de base zéro indiquant la première image qui doit apparaître dans le scénario du graphique.
<code>symbolInstance.forceSimple</code>	Valeur booléenne qui active ou désactive l'accessibilité aux enfants de l'objet ; équivalente à l'inverse de la fonction Rendre les objets enfant accessibles dans le panneau Accessibilité.
<code>symbolInstance.loop</code>	Chaîne qui définit, pour les symboles graphiques, la même propriété que le menu contextuel Boucle de l'inspecteur Propriétés.
<code>symbolInstance.shortcut</code>	Chaîne équivalente à la touche de raccourci associée au symbole, ainsi qu'au champ Raccourci dans le panneau Accessibilité.
<code>symbolInstance.silent</code>	Valeur booléenne qui active et désactive l'accessibilité à l'objet ; équivalente à l'inverse de la fonction Rendre l'objet accessible dans le panneau Accessibilité.
<code>symbolInstance.symbolType</code>	Chaîne spécifiant le type du symbole, équivalente à la valeur du champ Comportement dans les boîtes de dialogue Créer un nouveau symbole et Convertir en symbole.
<code>symbolInstance.tabIndex</code>	Entier équivalent au champ Index de tabulation du panneau Accessibilité.
« <code>symbolInstance.usesBackgroundColor</code> » à la page 434	Valeur booléenne spécifiant le format d'affichage.
« <code>symbolInstance.visible</code> » à la page 434	Valeur booléenne indiquant si l'occurrence est visible ou pas.

## symbolInstance.accName

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.accName`

### Description

Propriété ; chaîne équivalente au champ Nom du panneau Accessibilité. Les logiciels de lecture vocale identifient l'objet en prononçant ce nom. Cette propriété n'est pas disponible pour les symboles graphiques.

### Exemple

L'exemple suivant enregistre dans la variable `theName` la valeur du nom de l'objet dans le panneau Accessibilité :

```
var theName = fl.getDocumentDOM().selection[0].accName;
```

L'exemple suivant donne la valeur `Home Button` au nom de l'objet dans le panneau Accessibilité :

```
fl.getDocumentDOM().selection[0].accName = "Home Button";
```

## **symbolInstance.actionScript**

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.actionScript`

### Description

Propriété ; chaîne qui spécifie les actions affectées au symbole. Elle ne s'applique qu'aux occurrences de clip et de bouton. Pour une occurrence de symbole graphique, la valeur renvoyée est `undefined`.

### Exemple

L'exemple suivant affecte une action `onClipEvent` au premier élément de la première image du premier calque du scénario :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].actionScript  
= "onClipEvent (enterFrame) {trace('movie clip enterFrame')}";
```

## **symbolInstance.backgroundColor**

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

`symbolInstance.backgroundColor`

### Description

Propriété : chaîne spécifiant la couleur du cache lorsque l'occurrence est définie sur le mode 24 bits. Il s'agit d'une chaîne au format hexadécimal `#rrggbb` ou d'un entier contenant une valeur.

### Exemple

L'exemple suivant affecte à l'occurrence de symbole une couleur d'arrière-plan noire :

```
var bitmapInstance = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];  
bitmapInstance.backgroundColor = "#000000";
```

## symbolInstance.bitmapRenderMode

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

`symbolInstance.bitmapRenderMode`

### Description

Propriété : chaîne qui définit le type d’affichage du symbole.

Les valeurs prises en charge sont les suivantes :

- “none”
- “cache” : définit le symbole devant être mis en cache sous forme de bitmap par Flash Player au moment de l’exécution.
- “export” : définit le symbole à exporter sous forme de bitmap lors de la compilation du fichier SWF.

La propriété précédente, « [symbolInstance.cacheAsBitmap](#) » à la page 425, est semblable à cette propriété, mais offre moins de choix dans la mesure où cette dernière est de type booléen. A l’avenir, la propriété `cacheAsBitmap` pourra devenir obsolète, de façon à faciliter l’accès à cette nouvelle propriété. Les options `true/false` de la propriété booléenne `cacheAsBitmap` sont identiques aux valeurs « cache »/« none » de cette nouvelle propriété.

### Exemple

L’exemple suivant définit la propriété `bitmapRenderMode` du symbole sur la valeur “export” :

```
var symbol = fl.getDocumentDOM().selection[0];  
fl.trace(symbol.bitmapRenderMode);  
symbol.bitmapRenderMode = "export";
```

## symbolInstance.blendMode

### Disponibilité

Flash 8.

### Utilisation

`symbolInstance.blendMode`

### Description

Propriété ; chaîne qui spécifie le mode de fusion à appliquer à un symbole de clip. Les valeurs gérées sont "normal", "layer", "multiply", "screen", "overlay", "hardlight", "lighten", "darken", "difference", "add", "subtract", "invert", "alpha" et "erase".

### Exemple

L'exemple suivant définit le mode de fusion du premier symbole de clip dans la première image du premier niveau sur add :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].blendMode = "add";
```

### Voir aussi

[document.setBlendMode\(\)](#)

## symbolInstance.buttonTracking

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.buttonTracking`

### Description

Propriété ; chaîne qui définit, pour les symboles de bouton uniquement, la même propriété que le menu contextuel Traiter comme bouton ou Traiter comme élément de menu dans l'inspecteur des propriétés. Pour les autres types de symboles, cette propriété est ignorée. Les valeurs gérées sont "button" ou "menu".

### Exemple

L'exemple suivant donne au premier symbole de la première image du premier calque du scénario la valeur Traiter comme élément de menu, dans la mesure où ce symbole est un bouton :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].buttonTracking = "menu";
```

## symbolInstance.cacheAsBitmap

### Disponibilité

Flash 8.

### Utilisation

`symbolInstance.cacheAsBitmap`

### Description

Propriété ; valeur booléenne spécifiant si la mise en cache des bitmaps à l'exécution est activée ou non.

**Remarque :** A compter de Flash Professional CS5.5, les utilisateurs doivent employer la propriété « [symbolInstance.bitmapRenderMode](#) » à la page 424 à la place de cette propriété.

### Exemple

L'exemple suivant active la mise en cache des bitmaps à l'exécution pour le premier élément de la première image du premier calque :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].cacheAsBitmap = true;
```

## **symbolInstance.colorAlphaAmount**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorAlphaAmount`

### **Description**

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence, en indiquant la valeur des paramètres avancés de l'effet Alpha. Cette propriété équivaut à l'utilisation de l'entrée de menu Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles qui apparaissent à droite dans la boîte de dialogue. Cette valeur diminue ou augmente d'un niveau constant les valeurs d'alpha et de teinte. Cette valeur est ajoutée à la valeur actuelle. Cette propriété est plus utile si elle employée avec [symbolInstance.colorAlphaPercent](#). Les valeurs autorisées sont comprises entre -255 et 255.

### **Exemple**

L'exemple suivant soustrait la valeur 100 du paramètre alpha de l'occurrence de symbole sélectionnée :

```
fl.getDocumentDOM().selection[0].colorAlphaAmount = -100;
```

## **symbolInstance.colorAlphaPercent**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorAlphaPercent`

### **Description**

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété équivaut à l'utilisation de la commande Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue. Cette valeur spécifie le nouveau pourcentage de teinte et d'alpha. Les valeurs autorisées sont comprises entre -100 et 100. Voir aussi [symbolInstance.colorAlphaAmount](#).

### **Exemple**

L'exemple suivant définit sur 80 la valeur de la propriété `colorAlphaPercent` de l'occurrence de symbole sélectionnée :

```
fl.getDocumentDOM().selection[0].colorAlphaPercent = 80;
```



## **symbolInstance.colorBlueAmount**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorBlueAmount`

### **Description**

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété est équivalente au paramètre Couleur > Avancé dans l'inspecteur Propriétés de l'occurrence. Les valeurs autorisées sont comprises entre -255 et 255.

## **symbolInstance.colorBluePercent**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorBluePercent`

### **Description**

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété équivaut à l'utilisation de la commande Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue. Cette valeur spécifie le nouveau pourcentage de bleu. Les valeurs autorisées sont comprises entre -100 et 100.

### **Exemple**

L'exemple suivant définit sur 80 la valeur de la propriété `colorBluePercent` de l'occurrence de symbole sélectionnée :

```
fl.getDocumentDOM().selection[0].colorBluePercent = 80;
```

## **symbolInstance.colorGreenAmount**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorGreenAmount`

### **Description**

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété est équivalente au paramètre Couleur > Avancé dans l'inspecteur Propriétés de l'occurrence. Les valeurs autorisées sont comprises entre -255 et 255.

## **symbolInstance.colorGreenPercent**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorGreenPercent`

### **Description**

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété équivaut à l'utilisation de la commande Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue. Cette valeur spécifie le nouveau pourcentage de vert. Les valeurs autorisées sont comprises entre -100 et 100.

### **Exemple**

L'exemple suivant définit sur 70 la valeur de la propriété `colorGreenPercent` de l'occurrence de symbole sélectionnée :

```
fl.getDocumentDOM().selection[0].colorGreenPercent = 70;
```

## **symbolInstance.colorMode**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.colorMode`

### **Description**

Propriété ; chaîne spécifiant le mode colorimétrique tel qu'il est identifié dans le menu contextuel Couleurs de l'inspecteur Propriétés du symbole. Les valeurs gérées sont "none" (aucun), "brightness" (luminosité), "tint" (teinte), "alpha" et "advanced" (avancé).

### **Exemple**

L'exemple suivant donne la valeur alpha à la propriété `colorMode` du premier élément de la première image du premier calque du scénario :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].colorMode = "alpha";
```

## **symbolInstance.colorRedAmount**

### **Disponibilité**

Flash MX 2004.

### Utilisation

`symbolInstance.colorRedAmount`

### Description

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété est équivalente au paramètre Couleur > Avancé dans l'inspecteur Propriétés de l'occurrence. Les valeurs autorisées sont comprises entre -255 et 255.

### Exemple

L'exemple suivant définit sur 255 la valeur de la propriété `colorRedAmount` de l'occurrence de symbole sélectionnée :

```
fl.getDocumentDOM().selection[0].colorRedAmount = 255;
```

## **symbolInstance.colorRedPercent**

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.colorRedPercent`

### Description

Propriété ; entier spécifiant une partie de la transformation de couleur de l'occurrence. Cette propriété équivaut à l'utilisation de la commande Couleur > Paramètres avancés de l'inspecteur Propriétés, suivie du réglage des contrôles de pourcentage qui apparaissent à gauche dans la boîte de dialogue. Cette valeur spécifie le nouveau pourcentage de rouge. Les valeurs autorisées sont comprises entre -100 et 100.

### Exemple

L'exemple suivant définit sur 10 la valeur de la propriété `colorRedPercent` de l'occurrence de symbole sélectionnée :

```
fl.getDocumentDOM().selection[0].colorRedPercent = 10;
```

## **symbolInstance.description**

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.description`

### Description

Propriété ; chaîne équivalente au champ Description du panneau Accessibilité. Cette description est lue par les logiciels de lecture vocale. Cette propriété n'est pas disponible pour les symboles graphiques.

### Exemple

L'exemple suivant enregistre dans la variable `theDescription` la valeur de la description de l'objet dans le panneau Accessibilité :

```
var theDescription = fl.getDocumentDOM().selection[0].description;
```

L'exemple suivant affecte la chaîne `Click the home button to go to home` à la description de l'objet dans le panneau Accessibilité :

```
fl.getDocumentDOM().selection[0].description= "Click the home button to go to home";
```

## symbolInstance.filters

### Disponibilité

Flash 8.

### Utilisation

`symbolInstance.filters`

### Description

Propriété ; tableau d'objets [Filter](#) (voir [Objet Filter](#)). Pour modifier les propriétés relatives aux filtres, n'écrivez pas directement de données dans ce tableau. Vous devez en effet récupérer le tableau, définir chaque propriété, puis redéfinir le tableau de sorte à refléter les nouvelles propriétés.

### Exemple

L'exemple suivant recherche le nom du filtre à l'index 0. S'il s'agit d'un filtre de type `Glow`, la propriété `blurX` correspondante est définie sur 100 et la nouvelle valeur est écrite dans le tableau des filtres.

```
var filterName =  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].filters[0].name;  
fl.trace(filterName);  
var filterArray = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].filters;  
if (filterName == 'glowFilter'){  
    filterArray[0].blurX = 100;  
}  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].filters = filterArray;
```

## symbolInstance.firstFrame

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.firstFrame`

### Description

Propriété ; entier de base zéro indiquant la première image qui doit apparaître dans le scénario du graphisme. Cette propriété ne s'applique qu'aux symboles graphiques et définit le même paramètre que le champ Première image de l'inspecteur Propriétés. Pour les autres types de symboles, cette propriété a la valeur `undefined`.

### Exemple

L'exemple suivant spécifie que l'image 10 doit être la première à apparaître dans le scénario de l'élément spécifié :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].firstFrame = 10;
```

## symbolInstance.forceSimple

### Disponibilité

Flash MX 2004.

### Utilisation

```
symbolInstance.forceSimple
```

### Description

Propriété ; valeur booléenne qui active ou désactive l'accessibilité aux enfants de l'objet. Cette propriété est équivalente à l'inverse du paramètre Rendre les objets enfant accessibles du panneau Accessibilité. Par exemple, si `forceSimple` est `true`, son effet est similaire à la désactivation de l'option Rendre les objets enfant accessibles. Si `forceSimple` est `false`, son effet est similaire à l'activation de l'option Rendre les objets enfant accessibles.

Cette propriété est uniquement disponible pour les objets MovieClip.

### Exemple

L'exemple suivant vérifie que les enfants de l'objet sont accessibles ; si la valeur renvoyée est `false`, les enfants de l'objet sont accessibles :

```
var areChildrenAccessible = fl.getDocumentDOM().selection[0].forceSimple;
```

L'exemple suivant rend les enfants de l'objet accessibles :

```
fl.getDocumentDOM().selection[0].forceSimple = false;
```

## symbolInstance.loop

### Disponibilité

Flash MX 2004.

### Utilisation

```
symbolInstance.loop
```

### Description

Propriété ; chaîne qui définit, pour les symboles graphiques, la même propriété que le menu contextuel Boucle de l'inspecteur Propriétés. Pour les autres types de symboles, cette propriété a la valeur `undefined`. Les valeurs gérées sont "loop", "play once" et "single frame" pour définir l'animation du graphique en conséquence.

### Exemple

L'exemple suivant définit le premier symbole de la première image du premier calque du scénario sur `single frame` (affichage d'une seule image spécifiée dans le scénario graphique), dans la mesure où ce symbole est un graphique :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].loop = 'single frame';
```

## symbolInstance.shortcut

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.shortcut`

### Description

Propriété ; chaîne équivalente à la touche de raccourci associée au symbole. Cette propriété est équivalente au champ Raccourci du panneau Accessibilité. Cette touche est lue par les logiciels de lecture vocale. Cette propriété n'est pas disponible pour les symboles graphiques.

### Exemple

L'exemple suivant enregistre dans la variable `theShortcut` la valeur de la touche de raccourci de l'objet dans le panneau Accessibilité :

```
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;
```

L'exemple suivant définit sur `Ctrl+i` la touche de raccourci de l'objet :

```
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+i";
```

## symbolInstance.silent

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolInstance.silent`

### Description

Propriété ; valeur booléenne qui active ou désactive l'accessibilité de l'objet. Cette propriété est équivalente à l'inverse du paramètre Rendre l'objet accessible du panneau Accessibilité. Par exemple, si `silent` est `true`, son effet est similaire à la désactivation de l'option Rendre l'objet accessible. Si `silent` est `false`, son effet est similaire à l'activation de l'option Rendre l'objet accessible.

Cette propriété n'est pas disponible pour les objets graphiques.

### Exemple

L'exemple suivant vérifie que l'objet est accessible ; si la valeur renvoyée est `false`, l'objet est accessible :

```
var isSilent = fl.getDocumentDOM().selection[0].silent;
```

L'exemple suivant rend l'objet accessible :

```
fl.getDocumentDOM().selection[0].silent = false;
```

## **symbolInstance.symbolType**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.symbolType`

### **Description**

Propriété ; chaîne qui spécifie le type du symbole. Cette propriété est l'équivalent de la valeur du champ Comportement dans les boîtes de dialogue Créer un nouveau symbole et Convertir en symbole. Les valeurs possibles sont "button" (bouton), "movie clip" (clip) et "graphic".

### **Exemple**

L'exemple suivant donne le comportement de symbole graphique au premier symbole de la première image du premier calque du scénario du document actif :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].symbolType = "graphic";
```

## **symbolInstance.tabIndex**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`symbolInstance.tabIndex`

### **Description**

Propriété ; entier équivalent au champ Index de tabulation du panneau Accessibilité. Cette propriété crée l'ordre de tabulation dans lequel les objets seront sélectionnés lorsque l'utilisateur appuiera sur la touche de tabulation. Cette propriété n'est pas disponible pour les symboles graphiques.

### **Exemple**

L'exemple suivant donne la valeur 3 à la propriété `tabIndex` de l'objet `mySymbol`, puis affiche cette valeur dans le panneau Sortie :

```
var mySymbol = fl.getDocumentDOM().selection[0];  
mySymbol.tabIndex = 3;  
fl.trace(mySymbol.tabIndex);
```

## **symbolInstance.usesBackgroundColor**

### **Disponibilité**

Flash CS5.5 Professional.

### **Utilisation**

`symbolInstance.usesBackgroundColor`

### **Description**

Propriété valeur booléenne qui spécifie le mode à utiliser avec l'occurrence (24 bits (true) ou 32 bits avec alpha (false)). Si cette propriété est définie sur true, la propriété `backgroundColor` spécifiée pour l'occurrence est utilisée.

### **Exemple**

L'exemple suivant définit la propriété `usesBackgroundColor` d'une occurrence sur true :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].useTransparentBackground = true;
```

## **symbolInstance.visible**

### **Disponibilité**

Flash CS5.5 Professional.

### **Utilisation**

`symbolInstance.visible`

### **Description**

Propriété : valeur booléenne qui définit la propriété Visible d'un objet sur on (true) ou off (false).

### **Exemple**

L'exemple suivant définit la visibilité du premier élément de la première image du premier calque sur false :

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].visible = false;
```



# Chapitre 41 : Objet SymbolItem

**Héritage**    [Objet Item](#) > Objet SymbolItem

## Disponibilité

Flash MX 2004.

## Description

L'objet SymbolItem est une sous-classe de l'objet [Objet Item](#).

## Résumé des méthodes

Outre celles de l'objet Item, l'objet SymbolItem gère les méthodes suivantes :

Méthode	Description
<code>symbolItem.convertToCompiledClip()</code>	Convertit en clip compilé un élément de symbole dans la bibliothèque.
<code>symbolItem.exportSWC()</code>	Exporte le symbole dans un fichier SWC.
<code>symbolItem.exportSWF()</code>	Exporte le symbole dans un fichier SWF.

## Résumé des propriétés

Outre celles de l'objet Item, l'objet SymbolItem possède les propriétés suivantes :

Propriété	Description
<code>symbolItem.scalingGrid</code>	Valeur booléenne qui spécifie si l'échelle à 9 découpes est activée pour l'élément.
<code>symbolItem.scalingGridRect</code>	Objet Rectangle spécifiant les emplacements des quatre repères pour les 9 découpes.
<code>symbolItem.sourceAutoUpdate</code>	Valeur booléenne qui spécifie si l'élément est actualisé lors de la publication du fichier FLA.
<code>symbolItem.sourceFilePath</code>	Chaîne indiquant le chemin d'accès du fichier source FLA, sous la forme fichier:///URL.
<code>symbolItem.sourceLibraryName</code>	Chaîne qui spécifie le nom de l'élément dans la bibliothèque du fichier source.
<code>symbolItem.symbolType</code>	Chaîne qui spécifie le type de symbole.
<code>symbolItem.timeline</code>	Lecture seule ; <a href="#">Objet Timeline</a> .

## symbolItem.convertToCompiledClip()

## Disponibilité

Flash MX 2004.

## Utilisation

`symbolItem.convertToCompiledClip()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui convertit en clip compilé un élément de symbole dans la bibliothèque.

### Exemple

L'exemple suivant convertit en clip compilé un élément dans la bibliothèque :

```
fl.getDocumentDOM().library.items[3].convertToCompiledClip();
```

## symbolItem.exportSWC()

### Disponibilité

Flash MX 2004.

### Utilisation

```
symbolItem.exportSWC(outputURI)
```

### Paramètres

**outputURI** Chaîne de type fichier:///URI qui désigne le fichier SWC dans lequel la méthode doit exporter le symbole. L'adresse *outputURI* doit faire référence à un fichier local. Flash ne crée pas de dossier si l'adresse *outputURI* n'existe pas.

### Valeur renvoyée

Aucune.

### Description

Méthode qui exporte le symbole dans un fichier SWC.

### Exemple

L'exemple suivant exporte un élément de la bibliothèque dans un fichier SWC nommé `mySymbol.swc` et situé dans le dossier `tests` :

```
fl.getDocumentDOM().library.selectItem("mySymbol");  
var currentSelection = fl.getDocumentDOM().library.getSelectedItems();  
currentSelection[0].exportSWC("file:///Macintosh HD/SWCDirectory/mySymbol.swc");
```

## symbolItem.exportSWF()

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolItem.exportSWF(outputURI)`

### Paramètres

**outputURI** Chaîne de type fichier:///URI qui désigne le fichier SWF dans lequel la méthode doit exporter le symbole. L'adresse *outputURI* doit faire référence à un fichier local. Flash ne crée pas de dossier si l'adresse *outputURI* n'existe pas.

### Valeur renvoyée

Aucune.

### Description

Méthode qui exporte le symbole dans un fichier SWF.

### Exemple

L'exemple suivant exporte un élément de la bibliothèque dans le fichier `my.swf` du dossier `tests` :

```
fl.getDocumentDOM().library.items[0].exportSWF("file:///c:/tests/my.swf");
```

## symbolItem.scalingGrid

### Disponibilité

Flash 8.

### Utilisation

`symbolItem.scalingGrid`

### Description

Propriété ; valeur booléenne spécifiant si l'échelle à 9 découpes est activée pour l'élément.

### Exemple

L'exemple suivant active l'échelle à 9 découpes pour un élément de la bibliothèque :

```
fl.getDocumentDOM().library.items[0].scalingGrid = true;
```

### Voir aussi

[symbolItem.scalingGridRect](#)

## symbolItem.scalingGridRect

### Disponibilité

Flash 8.

### Utilisation

`symbolItem.scalingGridRect`

### Description

Propriété ; objet `Rectangle` spécifiant les emplacements des quatre repères pour les 9 découpes. Pour plus d'informations sur le format du rectangle, voir `document.addNewRectangle()`.

### Exemple

L'exemple suivant spécifie les emplacements des repères pour les 9 découpes :

```
fl.getDocumentDOM().library.items[0].scalingGridRect = {left:338, top:237, right:3859,  
bottom:713};
```

### Voir aussi

`symbolItem.scalingGrid`

## `symbolItem.sourceAutoUpdate`

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolItem.sourceAutoUpdate`

### Description

Propriété ; valeur booléenne qui spécifie si l'élément est actualisé lors de la publication du fichier FLA. La valeur par défaut est `false`. Cette propriété est utilisée pour les symboles de la bibliothèque partagée.

### Exemple

L'exemple suivant définit la propriété `sourceAutoUpdate` pour un élément de la bibliothèque :

```
fl.getDocumentDOM().library.items[0].sourceAutoUpdate = true;
```

## `symbolItem.sourceFilePath`

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolItem.sourceFilePath`

### Description

Propriété ; chaîne indiquant le chemin d'accès du fichier source FLA, sous la forme `fichier:///URI`. Cette adresse doit être un chemin absolu, et non pas un chemin relatif. Cette propriété est utilisée pour les symboles de la bibliothèque partagée.

### Exemple

L'exemple suivant affiche la valeur de la propriété `sourceFilePath` dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().library.items[0].sourceFilePath);
```

## symbolItem.sourceLibraryName

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolItem.sourceLibraryName`

### Description

Propriété ; chaîne qui spécifie le nom de l'élément dans la bibliothèque du fichier source. Cette propriété est utilisée pour les symboles de la bibliothèque partagée.

### Exemple

L'exemple suivant affiche la valeur de la propriété `sourceLibraryName` dans le panneau Sortie :

```
fl.trace(fl.getDocumentDOM().library.items[0].sourceLibraryName);
```

## symbolItem.symbolType

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolItem.symbolType`

### Description

Propriété ; chaîne qui spécifie le type du symbole. Les valeurs possibles sont "button" (bouton), "movie clip" (clip) et "graphic".

### Exemple

L'exemple suivant affiche la valeur actuelle de la propriété `symbolType`, puis lui donne la valeur `button` avant de l'afficher à nouveau :

```
alert(fl.getDocumentDOM().library.items[0].symbolType);  
fl.getDocumentDOM().library.items[0].symbolType = "button";  
alert(fl.getDocumentDOM().library.items[0].symbolType);
```

## symbolItem.timeline

### Disponibilité

Flash MX 2004.

### Utilisation

`symbolItem.timeline`

### Description

Propriété en lecture seule ; [Objet Timeline](#).

### Exemple

L'exemple suivant récupère et affiche le nombre de calques du clip sélectionné dans la bibliothèque :

```
var tl = fl.getDocumentDOM().library.getSelectedItems()[0].timeline;  
alert(tl.layerCount);
```

# Chapitre 42 : Objet Text

**Héritage**    [Objet Element](#) > Objet Text

## Disponibilité

Flash MX 2004.

## Description

L'objet Text représente un élément de texte unique dans un document. Toutes les propriétés du texte concernent l'ensemble du bloc de texte.

Pour savoir comment définir les propriétés d'un segment de texte à l'intérieur du champ de texte, voir le Résumé des propriétés de l'[Objet TextAttrs](#). Pour modifier les propriétés d'une sélection à l'intérieur d'un champ de texte, vous pouvez utiliser `document.setTextElementAttr()` en spécifiant une plage de texte ou la sélection actuelle.

Pour définir les propriétés génériques du champ de texte sélectionné, utilisez `document.setTextElementProperty()`. L'exemple suivant définit sur 50 la valeur x du point d'alignement du champ de texte sélectionné :

```
fl.getDocumentDOM().setTextElementProperty("x", 50);
```

## Résumé des méthodes

Outre celles de l'objet Element, l'objet Text possède les méthodes suivantes :

Méthode	Description
<code>text.getTextAttr()</code>	Récupère l'attribut spécifié pour le texte identifié par les paramètres facultatifs <i>startIndex</i> et <i>endIndex</i> .
<code>text.getTextString()</code>	Récupère la plage de texte spécifiée.
<code>text.setTextAttr()</code>	Définit l'attribut spécifié pour le texte identifié par les paramètres <i>startIndex</i> et <i>endIndex</i> .
<code>text.setTextString()</code>	Modifie la chaîne de texte dans cet objet Text.

## Résumé des propriétés

Outre celles de l'objet Element, l'objet Text possède les propriétés suivantes :

Propriété	Description
<code>text.accName</code>	Chaîne équivalente au champ Nom du panneau Accessibilité.
<code>text.antiAliasSharpness</code>	Nombre en virgule flottante spécifiant la netteté de l'anti-aliasing du texte.
<code>text.antiAliasThickness</code>	Nombre en virgule flottante spécifiant l'épaisseur de l'anti-aliasing du texte.
<code>text.autoExpand</code>	Valeur booléenne qui contrôle l'expansion de la largeur du cadre de délimitation des champs de texte statiques, ou la largeur et la hauteur du cadre de délimitation des champs de texte dynamiques ou de saisie.
<code>text.border</code>	Valeur booléenne qui détermine si Flash affiche ( <i>true</i> ) ou masque ( <i>false</i> ) une bordure autour des champs de texte dynamiques ou de saisie.
<code>text.description</code>	Chaîne équivalente au champ Description du panneau Accessibilité.
<code>text.embeddedCharacters</code>	Chaîne qui spécifie les caractères à incorporer. Equivalent à la saisie de texte dans la boîte de dialogue Intégration de caractères.

Propriété	Description
<code>text.embedRanges</code>	Chaîne composée d'entiers délimités correspondant aux éléments qui peuvent être sélectionnés dans la boîte de dialogue Intégration de caractères.
<code>text.embedVariantGlyphs</code>	Une valeur booléenne qui spécifie s'il faut activer l'intégration de variantes de glyphes.
<code>text.fontRenderingMode</code>	Chaîne qui spécifie le mode de rendu du texte.
<code>text.length</code>	Lecture seule ; entier représentant le nombre de caractères de l'objet Text.
<code>text.lineType</code>	Chaîne définissant le type de ligne comme "single line" (une seule ligne), "multiline" (multiligne), "multiline no wrap" (multiligne sans retour) ou "password" (mot de passe).
<code>text.maxCharacters</code>	Entier spécifiant le nombre maximal de caractères que l'utilisateur peut indiquer dans cet objet Text.
<code>text.orientation</code>	Chaîne spécifiant l'orientation du champ de texte.
<code>text.renderAsHTML</code>	Valeur booléenne qui détermine si Flash affiche le texte en HTML et interprète les balises HTML incorporées.
<code>text.scrollable</code>	Valeur booléenne qui indique si le texte peut défiler ( <code>true</code> ) ou non ( <code>false</code> ).
<code>text.selectable</code>	Valeur booléenne qui indique si le texte peut être sélectionné ( <code>true</code> ) ou non ( <code>false</code> ). Le texte de saisie peut toujours être sélectionné.
<code>text.selectionEnd</code>	Entier de base zéro indiquant le décalage de la fin d'une sous-sélection de texte.
<code>text.selectionStart</code>	Entier de base zéro indiquant le décalage du début d'une sous-sélection de texte.
<code>text.shortcut</code>	Chaîne équivalente au champ Raccourci du panneau Accessibilité.
<code>text.silent</code>	Valeur booléenne indiquant si l'objet est accessible.
<code>text.tabIndex</code>	Nombre entier équivalent au champ Index de tabulation du panneau Accessibilité.
<code>text.textRuns</code>	Lecture seule ; tableau d'objets TextRun.
<code>text.textType</code>	Chaîne qui spécifie le type du champ de texte. Les valeurs possibles sont "static", "dynamic" et "input".
<code>text.useDeviceFonts</code>	Valeur booléenne. Si sa valeur est <code>true</code> , Flash trace le texte à l'aide des polices de périphérique.
<code>text.variableName</code>	Chaîne présentant le contenu de l'objet Text.

## text.accName

### Disponibilité

Flash MX 2004.

### Utilisation

`text.accName`

### Description

Propriété ; chaîne équivalente au champ Nom du panneau Accessibilité. Les logiciels de lecture vocale identifient l'objet en prononçant ce nom. Cette propriété ne peut pas être utilisée avec du texte dynamique.

### Exemple

L'exemple suivant récupère le nom de l'objet :



```
var doc = fl.getDocumentDOM();  
var theName = doc.selection[0].accName;
```

L'exemple suivant définit le nom de l'objet actuellement sélectionné :

```
fl.getDocumentDOM().selection[0].accName = "Home Button";
```

## text.antiAliasSharpness

### Disponibilité

Flash 8.

### Utilisation

```
text.antiAliasSharpness
```

### Description

Propriété ; nombre en virgule flottante spécifiant la netteté de l'anti-aliasing du texte. Cette propriété contrôle la netteté selon laquelle le texte est tracé ; les valeurs plus élevées correspondent à du texte plus net. Une valeur de 0 représente une netteté normale. Cette propriété n'est disponible que si `text.fontRenderingMode` est défini sur `customThicknessSharpness`.

### Exemple

Voir `text.fontRenderingMode`.

### Voir aussi

`text.antiAliasThickness`, `text.fontRenderingMode`

## text.antiAliasThickness

### Disponibilité

Flash 8.

### Utilisation

```
text.antiAliasThickness
```

### Description

Propriété ; nombre en virgule flottante spécifiant l'épaisseur de l'anti-aliasing du texte. Cette propriété détermine l'épaisseur selon laquelle le texte est tracé, une valeur plus élevée représentant un texte plus épais. Une valeur de 0 représente une épaisseur normale. Cette propriété n'est disponible que si `text.fontRenderingMode` est défini sur `customThicknessSharpness`.

### Exemple

Voir `text.fontRenderingMode`.

### Voir aussi

`text.antiAliasSharpness`, `text.fontRenderingMode`

## text.autoExpand

### Disponibilité

Flash MX 2004.

### Utilisation

`text.autoExpand`

### Description

Propriété : valeur booléenne. Pour les champs de texte statiques, la valeur `true` provoque une expansion en largeur du cadre de délimitation afin d'afficher tout le texte. Pour les champs de texte dynamiques ou de saisie, la valeur `true` provoque une expansion en largeur et en hauteur du cadre de délimitation afin d'afficher tout le texte.

### Exemple

L'exemple suivant donne à la propriété `autoExpand` la valeur `true` :

```
fl.getDocumentDOM().selection[0].autoExpand = true;
```

## text.border

### Disponibilité

Flash MX 2004.

### Utilisation

`text.border`

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, Flash trace une bordure autour du texte.

### Exemple

L'exemple suivant donne à la propriété `border` la valeur `true` :

```
fl.getDocumentDOM().selection[0].border = true;
```

## text.description

### Disponibilité

Flash MX 2004.

### Utilisation

`text.description`

### Description

Propriété ; chaîne équivalente au champ Description du panneau Accessibilité. Cette description est lue par les logiciels de lecture vocale.

### Exemple

L'exemple suivant récupère la description de l'objet :

```
var doc = fl.getDocumentDOM();  
var desc = doc.selection[0].description;
```

L'exemple suivant définit la description de l'objet :

```
var doc = fl.getDocumentDOM();  
doc.selection[0].description= "Enter your name here";
```

## text.embeddedCharacters

### Disponibilité

Flash MX 2004.

### Utilisation

`text.embeddedCharacters`

### Description

Propriété ; chaîne qui spécifie les caractères à incorporer. Equivalent à la saisie de texte dans la boîte de dialogue Intégration de caractères.

Elle est disponible uniquement avec les champs de texte dynamiques ou de saisie, et déclenche un avertissement avec un autre type de champ de texte.

**Remarque :** à partir de Flash Professional CS5, l'incorporation de polices est gérée au niveau du document en non plus au niveau de l'objet de texte. Utilisez la propriété « [fontItem.embeddedCharacters](#) » à la page 284 plutôt que la propriété `text.embeddedCharacters`.

### Exemple

L'exemple suivant suppose que le premier ou l'unique élément dans la sélection actuelle est un objet de texte classique et définit la propriété `embeddedCharacters` sur `abc` :

```
fl.getDocumentDOM().selection[0].embeddedCharacters = "abc";
```

## text.embedRanges

### Disponibilité

Flash MX 2004.

### Utilisation

`text.embedRanges`

### Description

Propriété ; chaîne composée d'entiers délimités correspondant aux éléments qui peuvent être sélectionnés dans la boîte de dialogue Intégration de caractères. Cette propriété est disponible uniquement avec les champs de texte dynamiques ou de saisie, et est ignorée avec du texte statique.

cette propriété correspond au fichier XML du dossier Configuration/Font Embedding.

**Remarque :** à partir de Flash Professional CS5, l'incorporation de polices est gérée au niveau du document en non plus au niveau de l'objet de texte. Utilisez la propriété « [fontItem.embedRanges](#) » à la page 284 plutôt que la propriété `text.embedRanges`.

### Exemple

L'exemple suivant suppose que le premier ou l'unique élément dans la sélection actuelle est un objet de texte classique et définit la propriété `embedRanges` sur "1|3|7" :

```
var doc = fl.getDocumentDOM();  
doc.selection[0].embedRanges = "1|3|7";
```

L'exemple suivant réinitialise la propriété :

```
var doc = fl.getDocumentDOM();  
doc.selection[0].embedRanges = "";
```

## text.embedVariantGlyphs

### Disponibilité

Flash CS4 Professional.

### Utilisation

`text.embedVariantGlyphs`

### Description

Propriété; une valeur booléenne qui spécifie s'il faut activer l'intégration de variantes de glyphes (`true`) ou pas (`false`). Cette propriété est disponible uniquement avec les champs de texte dynamiques ou de saisie, et est ignorée avec du texte statique. La valeur par défaut est `false`.

**Remarque :** à partir de Flash Professional CS5, l'incorporation de polices est gérée au niveau du document en non plus au niveau de l'objet de texte. Utilisez la propriété « [fontItem.embedVariantGlyphs](#) » à la page 285 plutôt que la propriété `text.embedVariantGlyphs`. Dans Flash Professional CS5, la propriété `text.embedVariantGlyphs` n'a plus aucun effet, car Flash incorpore toujours les variantes de glyphes pour le texte TLF et ne les incorpore jamais pour le texte classique.

### Exemple

L'exemple suivant active les variantes de glyphes pour qu'elles soient intégrées dans l'objet Text sélectionné :

```
fl.getDocumentDOM().selection[0].embedVariantGlyphs = true;
```

### Voir aussi

[fontItem.embedVariantGlyphs](#)

## text.fontRenderingMode

### Disponibilité

Flash 8.

## Utilisation

`text.fontRenderingMode`

## Description

Propriété ; chaîne qui spécifie le mode de rendu du texte. Cette propriété affecte l’affichage du texte sur la scène et dans Flash Player. Les valeurs gérées sont décrites dans le tableau suivant :

Valeur de la propriété	Rendu du texte
device	Le texte est restitué avec les polices de périphérique.
bitmap	Le texte aliasé est restitué sous la forme d’un bitmap ou comme le ferait une police en mode pixel.
standard	Le texte est restitué par le biais de la méthode d’anti-aliasing standard utilisée par Flash MX 2004. Ce paramètre est particulièrement adapté au texte animé, de très grande taille ou incliné.
advanced	Le texte est restitué par le biais de la technologie de rendu des polices, l’anti-alias avancé mis en oeuvre dans Flash 8. Elle produit un meilleur anti-aliasing et améliore la lisibilité, en particulier dans le cas de texte de petite taille.
customThicknessSharpness	Permet de définir des paramètres personnalisés pour la netteté et l’épaisseur du texte en cas d’emploi de la technologie de rendu des polices par l’anti-alias avancé mis en oeuvre dans Flash 8.

## Exemple

L’exemple suivant illustre une utilisation de la valeur `customThicknessSharpness` pour spécifier la netteté et l’épaisseur du texte :

```
fl.getDocumentDOM().setElementProperty("fontRenderingMode", "customThicknessSharpness");  
fl.getDocumentDOM().setElementProperty("antiAliasSharpness", 400);  
fl.getDocumentDOM().setElementProperty("antiAliasThickness", -200);
```

## Voir aussi

`text.antiAliasSharpness`, `text.antiAliasThickness`

# text.getTextAttr()

## Disponibilité

Flash MX 2004.

## Utilisation

`text.getTextAttr(attrName [, startIndex [, endIndex]])`

## Paramètres

**attrName** Chaîne spécifiant le nom de la propriété TextAttrs à renvoyer. La liste des valeurs possibles de *attrName* figure dans le Résumé des propriétés de l’Objet TextAttrs.

**startIndex** Nombre entier qui spécifie l’index du premier caractère. Ce paramètre est facultatif.

**endIndex** Nombre entier spécifiant la fin de la plage de texte, de *startIndex* compris à *endIndex* non compris. Ce paramètre est facultatif.

### Valeur renvoyée

La valeur de l'attribut spécifié par le paramètre *attrName*.

### Description

Méthode qui récupère l'attribut spécifié par le paramètre *attrName* pour le texte identifié par les paramètres facultatifs *startIndex* et *endIndex*. Si cet attribut n'est pas cohérent pour la plage spécifiée, Flash renvoie la valeur *undefined*. Si vous ne spécifiez pas les paramètres facultatifs *startIndex* et *endIndex*, la méthode utilise la plage de texte entière. Si vous ne précisez que *startIndex*, la plage devient le caractère qui occupe cette position. Si *startIndex* et *endIndex* sont tous deux spécifiés, la plage de caractères commence à *startIndex* et se termine au dernier caractère avant *endIndex*.

### Exemple

L'exemple suivant récupère et affiche la taille de caractères du champ de texte sélectionné :

```
var TheTextSize = fl.getDocumentDOM().selection[0].getTextAttr("size");  
fl.trace(TheTextSize);
```

L'exemple suivant récupère la couleur de remplissage du champ de texte sélectionné :

```
var TheFill = fl.getDocumentDOM().selection[0].getTextAttr("fillColor");  
fl.trace(TheFill);
```

L'exemple suivant récupère la taille du troisième caractère du champ de texte sélectionné :

```
var Char3 = fl.getDocumentDOM().selection[0].getTextAttr("size", 2);  
fl.trace(Char3);
```

L'exemple suivant récupère la couleur du champ de texte sélectionné entre les troisième et huitième caractères :

```
fl.getDocumentDOM().selection[0].getTextAttr("fillColor", 2, 8);
```

## text.getTextString()

### Disponibilité

Flash MX 2004.

### Utilisation

```
text.getTextString([startIndex [, endIndex]])
```

### Paramètres

**startIndex** Nombre entier qui spécifie l'index (de base zéro) du premier caractère. Ce paramètre est facultatif.

**endIndex** Nombre entier spécifiant la fin de la plage de texte, de *startIndex* compris à *endIndex* non compris. Ce paramètre est facultatif.

### Valeur renvoyée

Chaîne de texte de la plage spécifiée.

### Description

Méthode qui récupère la plage de texte spécifiée. Si vous omettez les paramètres facultatifs *startIndex* et *endIndex*, la méthode renvoie tout le texte. Si seul le paramètre *startIndex* est transmis, la méthode renvoie la chaîne qui commence à cet index et se termine à la fin du champ. Si vous spécifiez à la fois *startIndex* et *endIndex*, la méthode renvoie la chaîne comprise entre *startIndex* et le caractère précédant *endIndex*.

### Exemple

L'exemple suivant récupère le texte compris entre le cinquième caractère et la fin de la sélection :

```
var myText = fl.getDocumentDOM().selection[0].getTextString(4);  
fl.trace(myText);
```

L'exemple suivant récupère le texte compris entre les quatrième et neuvième caractères à partir du début du texte sélectionné :

```
var myText = fl.getDocumentDOM().selection[0].getTextString(3, 9);  
fl.trace(myText);
```

## text.length

### Disponibilité

Flash MX 2004.

### Utilisation

`text.length`

### Description

Propriété en lecture seule ; entier indiquant le nombre de caractères présents dans l'objet Text.

### Exemple

L'exemple suivant renvoie le nombre de caractères du texte sélectionné :

```
var textLength = fl.getDocumentDOM().selection[0].length;
```

## text.lineType

### Disponibilité

Flash MX 2004.

### Utilisation

`text.lineType`

### Description

Propriété ; chaîne qui définit le type de ligne. Les valeurs possibles sont "single line", "multiline", "multiline no wrap" et "password".

Cette propriété est uniquement disponible avec les champs de texte dynamiques ou de saisie, et déclenche un avertissement avec du texte statique. La valeur "password" n'est valide que pour les champs de saisie.

### Exemple

L'exemple suivant donne à la propriété `lineType` la valeur `multiline no wrap` :

```
fl.getDocumentDOM().selection[0].lineType = "multiline no wrap";
```

## text.maxCharacters

### Disponibilité

Flash MX 2004.

### Utilisation

`text.maxCharacters`

### Description

Propriété ; entier spécifiant le nombre maximal de caractères que l'utilisateur peut saisir dans cet objet Text.

Elle est disponible uniquement avec les champs de saisie, et déclenche un avertissement avec un autre type de champ de texte.

### Exemple

L'exemple suivant donne la valeur 30 à la propriété maxCharacters :

```
fl.getDocumentDOM().selection[0].maxCharacters = 30;
```

## text.orientation

### Disponibilité

Flash MX 2004.

### Utilisation

`text.orientation`

### Description

Propriété ; chaîne spécifiant l'orientation du champ de texte. Les valeurs possibles sont "horizontal", "vertical left to right" (vertical de gauche à droite) et "vertical right to left" (vertical de droite à gauche).

Cette propriété est disponible uniquement avec les champs de texte statiques, et déclenche un avertissement avec un autre type de champ de texte.

### Exemple

L'exemple suivant donne à l'orientation la valeur vertical right to left :

```
fl.getDocumentDOM().selection[0].orientation = "vertical right to left";
```

## text.renderAsHTML

### Disponibilité

Flash MX 2004.

### Utilisation

`text.renderAsHTML`



### Description

Propriété : valeur booléenne. Si sa valeur est `true`, Flash affiche le texte en HTML et interprète les balises HTML incorporées.

Elle est disponible uniquement avec les champs de texte dynamiques ou de saisie, et déclenche un avertissement avec un autre type de champ de texte.

### Exemple

L'exemple suivant donne à la propriété `renderAsHTML` la valeur `true` :

```
fl.getDocumentDOM().selection[0].renderAsHTML = true;
```

## text.scrollable

### Disponibilité

Flash MX 2004.

### Utilisation

`text.scrollable`

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, il est possible de faire défiler le texte.

Cette propriété est uniquement disponible avec les champs dynamiques ou de saisie, et déclenche un avertissement avec un champ de texte statique.

### Exemple

L'exemple suivant donne à la propriété `scrollable` la valeur `false` :

```
fl.getDocumentDOM().selection[0].scrollable = false;
```

## text.selectable

### Disponibilité

Flash MX 2004.

### Utilisation

`text.selectable`

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, il est possible de sélectionner le texte.

Le texte de saisie peut toujours être sélectionné. Flash génère un avertissement lorsque cette propriété est définie sur `false` et utilisée avec un texte de saisie.

### Exemple

L'exemple suivant donne à la propriété `selectable` la valeur `true` :

```
fl.getDocumentDOM().selection[0].selectable = true;
```

## text.selectionEnd

### Disponibilité

Flash MX 2004.

### Utilisation

`text.selectionEnd`

### Description

Propriété ; entier de base zéro indiquant la fin d'une sous-sélection de texte. Pour plus d'informations, voir [text.selectionStart](#).

## text.selectionStart

### Disponibilité

Flash MX 2004.

### Utilisation

`text.selectionStart`

### Description

Propriété ; entier de base zéro indiquant le début d'une sous-sélection de texte. Cette propriété peut être utilisée avec `text.selectionEnd` pour sélectionner une plage de caractères. Les caractères sont sélectionnés jusqu'à `text.selectionEnd` non compris. Voir [text.selectionEnd](#).

- S'il n'y a pas de sélection, ou si le point d'insertion est actif, `text.selectionEnd` possède la même valeur que `text.selectionStart`.
- Si `text.selectionStart` reçoit une valeur supérieure à celle de `text.selectionEnd`, `text.selectionEnd` prendra la même valeur que `text.selectionStart`, et aucun texte ne sera sélectionné.

### Exemple

L'exemple suivant définit le début de la sous-sélection de texte sur le sixième caractère :

```
fl.getDocumentDOM().selection[0].selectionStart = 5;
```

L'exemple suivant sélectionne les caractères Barbara dans un champ de texte contenant la phrase My name is Barbara, puis met ces caractères en vert gras :

```
fl.getDocumentDOM().selection[0].selectionStart = 11;  
fl.getDocumentDOM().selection[0].selectionEnd = 18;  
var s = fl.getDocumentDOM().selection[0].selectionStart;  
var e = fl.getDocumentDOM().selection[0].selectionEnd;  
fl.getDocumentDOM().setElementTextAttr('bold', true, s, e);  
fl.getDocumentDOM().setElementTextAttr("fillColor", "#00ff00", s, e);
```

## text.setTextAttr()

### Disponibilité

Flash MX 2004.

### Utilisation

```
text.setTextAttr(attrName, attrValue [, startIndex [, endIndex]])
```

### Paramètres

**attrName** Chaîne spécifiant le nom de la propriété TextAttrs à modifier.

**attrValue** Valeur de la propriété TextAttrs.

La liste des valeurs possibles de *attrName* et *attrValue* figure à la section Résumé des propriétés de l'Objet [TextAttrs](#).

**startIndex** Nombre entier qui spécifie l'index (de base zéro) du premier caractère du tableau. Ce paramètre est facultatif.

**endIndex** Nombre entier spécifiant l'index du point d'extrémité de la chaîne de texte sélectionnée, de *startIndex* compris à *endIndex* non compris. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui donne la valeur spécifiée par le paramètre *attrValue* à l'attribut spécifié par le paramètre *attrName* associé au texte identifié par les paramètres facultatifs *startIndex* et *endIndex*. Cette méthode permet de changer des attributs de texte chevauchant des éléments TextRun (voir [Objet TextRun](#)), ou représentant des parties d'éléments TextRun existants. Son utilisation est susceptible de modifier la position et le nombre d'éléments TextRun dans le tableau `text.textRuns` de cet objet (voir [text.textRuns](#)).

Si vous omettez les paramètres facultatifs, la méthode utilise la plage de caractères complète de l'objet Text. Si vous précisez uniquement *startIndex*, la plage devient le caractère qui occupe cette position. Si *startIndex* et *endIndex* sont tous deux spécifiés, la plage de caractères commence à *startIndex* et se termine au dernier caractère avant *endIndex*.

### Exemple

L'exemple suivant met en italique le champ de texte sélectionné :

```
fl.getDocumentDOM().selection[0].setTextAttr("italic", true);
```

L'exemple suivant donne une taille de 10 points au troisième caractère :

```
fl.getDocumentDOM().selection[0].setTextAttr("size", 10, 2);
```

L'exemple suivant donne la couleur rouge au texte compris entre le troisième et le huitième caractère :

```
fl.getDocumentDOM().selection[0].setTextAttr("fillColor", 0xff0000, 2, 8);
```

## text.setTextString()

### Disponibilité

Flash MX 2004.

### Utilisation

```
text.setTextString(text [, startIndex [, endIndex]])
```

### Paramètres

**text** Chaîne composée des caractères à insérer dans l'objet Text.

**startIndex** Nombre entier qui spécifie l'index (de base zéro) du premier caractère de la chaîne dans laquelle le texte sera inséré. Ce paramètre est facultatif.

**endIndex** Nombre entier qui spécifie l'index du dernier caractère de la chaîne sélectionnée. La nouvelle chaîne remplace le texte compris entre *startIndex* et *endIndex* non compris. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Propriété ; modifie la chaîne de texte de cet objet Text. Si vous omettez les paramètres facultatifs, l'objet Text est remplacé dans son intégralité. Si vous précisez uniquement *startIndex*, la chaîne spécifiée est insérée à la position *startIndex*. Si *startIndex* et *endIndex* sont transmis, la chaîne spécifiée remplace le segment de texte compris entre *startIndex* et le dernier caractère avant *endIndex*.

### Exemple

L'exemple suivant affecte la chaîne `this is a string` au champ de texte sélectionné :

```
fl.getDocumentDOM().selection[0].setTextString("this is a string");
```

L'exemple suivant insère la chaîne `abc` à partir du cinquième caractère du champ de texte sélectionné :

```
fl.getDocumentDOM().selection[0].setTextString("01234567890");  
fl.getDocumentDOM().selection[0].setTextString("abc", 4);  
// text field is now "0123abc4567890"
```

L'exemple suivant remplace le texte compris entre le troisième et le huitième caractère du texte sélectionné par la chaîne `abcdefghij`. Les caractères compris entre *startIndex* et *endIndex* sont remplacés. La chaîne de caractères qui débute à *endIndex* est décalée après la chaîne insérée.

```
fl.getDocumentDOM().selection[0].setTextString("01234567890");  
fl.getDocumentDOM().selection[0].setTextString("abcdefghij", 2, 8);  
// text field is now "01abcdefghij890"
```

## text.shortcut

### Disponibilité

Flash MX 2004.

### Utilisation

```
text.shortcut
```

### Description

Propriété ; chaîne équivalente au champ Raccourci du panneau Accessibilité. Ce raccourci est lu par les logiciels de lecture vocale. Cette propriété ne peut pas être utilisée avec du texte dynamique.

### Exemple

L'exemple suivant récupère et affiche la touche de raccourci de l'objet sélectionné :

```
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;  
fl.trace(theShortcut);
```

L'exemple suivant définit la touche de raccourci de l'objet sélectionné :

```
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+i";
```

## text.silent

### Disponibilité

Flash MX 2004.

### Utilisation

`text.silent`

### Description

Propriété ; valeur booléenne indiquant si l'objet est accessible. Cette propriété est équivalente à l'inverse du paramètre Rendre l'objet accessible du panneau Accessibilité. Ainsi, si `silent` est `true`, l'option Rendre l'objet accessible est désactivée. Si cette valeur est `false`, l'option Rendre l'objet accessible est activée.

### Exemple

L'exemple suivant vérifie que l'objet est accessible ; si la valeur renvoyée est `false`, l'objet est accessible :

```
var isSilent = fl.getDocumentDOM().selection[0].silent;
```

L'exemple suivant rend l'objet accessible :

```
fl.getDocumentDOM().selection[0].silent = false;
```

## text.tabIndex

### Disponibilité

Flash MX 2004.

### Utilisation

`text.tabIndex`

### Description

Propriété ; nombre entier équivalent au champ Index de tabulation du panneau Accessibilité. Cette valeur permet de déterminer l'ordre dans lequel les objets seront sélectionnés lorsque l'utilisateur appuiera sur la touche de tabulation.

### Exemple

L'exemple suivant récupère la valeur de la propriété `tabIndex` de l'objet actuellement sélectionné :

```
var theTabIndex = fl.getDocumentDOM().selection[0].tabIndex;
```

L'exemple suivant définit la propriété `tabIndex` de l'objet actuellement sélectionné :

```
fl.getDocumentDOM().selection[0].tabIndex = 1;
```

## text.textRuns

### Disponibilité

Flash MX 2004.

### Utilisation

`text.textRuns`

### Description

Propriété en lecture seule ; tableau d'objets TextRun (voir [Objet TextRun](#)).

### Exemple

L'exemple suivant enregistre dans la variable `myTextRuns` la valeur de la propriété `textRuns` :

```
var myTextRuns = fl.getDocumentDOM().selection[0].textRuns;
```

## text.textType

### Disponibilité

Flash MX 2004.

### Utilisation

`text.textType`

### Description

Propriété ; chaîne qui spécifie le type du champ de texte. Les valeurs possibles sont "static", "dynamic" et "input".

### Exemple

L'exemple suivant donne à la propriété `textType` la valeur `input` :

```
fl.getDocumentDOM().selection[0].textType = "input";
```

## text.useDeviceFonts

### Disponibilité

Flash MX 2004.

### Utilisation

`text.useDeviceFonts`

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, Flash trace le texte à l'aide des polices de périphérique.

### Exemple

L'exemple suivant oblige Flash à utiliser les polices de périphérique lors du tracé du texte :

```
fl.getDocumentDOM().selection[0].useDeviceFonts = true;
```

## text.variableName

### Disponibilité

Flash MX 2004.

### Utilisation

```
text.variableName
```

### Description

Propriété ; chaîne contenant le nom de la variable associée à l'objet Text. Elle est disponible uniquement avec les champs de texte dynamiques ou de saisie, et déclenche un avertissement avec un autre type de champ de texte.

Cette propriété n'est prise en charge que dans ActionScript 1.0 et ActionScript 2.0.

### Exemple

L'exemple suivant définit le nom de la variable du champ de texte sélectionné sur `firstName` :

```
fl.getDocumentDOM().selection[0].variableName = "firstName";
```

# Chapitre 43 : Objet TextAttrs

## Disponibilité

Flash MX 2004.

## Description

L'objet TextAttrs contient toutes les propriétés de texte pouvant être appliquées à une sous-sélection. Cet objet est une propriété de l'objet TextRun (`textRun.textAttrs`).

## Résumé des propriétés

Les propriétés suivantes sont disponibles avec l'objet TextAttrs :

Propriété	Description
<code>textAttrs.aliasText</code>	Valeur booléenne spécifiant que Flash doit tracer le texte à l'aide d'une méthode optimisée pour améliorer la lisibilité du texte en petits caractères.
<code>textAttrs.alignment</code>	Chaîne qui spécifie la justification des paragraphes. Les valeurs possibles sont "left", "center", "right" et "justify".
<code>textAttrs.autoKern</code>	Valeur booléenne qui détermine si Flash utilise ( <code>true</code> ) ou ignore ( <code>false</code> ) les informations de crénage (distance entre deux lettres) des polices de caractères.
<code>textAttrs.bold</code>	Valeur booléenne. La valeur <code>true</code> provoque l'apparition de la version gras de la police.
<code>textAttrs.characterPosition</code>	Chaîne qui détermine la ligne de base du texte.
<code>textAttrs.characterSpacing</code>	Déconseillé en faveur de <code>textAttrs.letterSpacing</code> . Nombre entier représentant l'espacement entre les caractères.
<code>textAttrs.face</code>	Chaîne qui représente le nom de la police, par exemple "Arial".
<code>textAttrs.fillColor</code>	Chaîne, valeur hexadécimale ou entière représentant la couleur de remplissage.
<code>textAttrs.indent</code>	Entier spécifiant l'indentation des paragraphes.
<code>textAttrs.italic</code>	Valeur booléenne. Si sa valeur est <code>true</code> , Flash trace le texte à l'aide de la version italique de la police.
<code>textAttrs.leftMargin</code>	Entier spécifiant la marge gauche des paragraphes.
<code>textAttrs.letterSpacing</code>	Nombre entier représentant l'espacement entre les caractères.
<code>textAttrs.lineSpacing</code>	Entier spécifiant l'interlignage des paragraphes.
<code>textAttrs.rightMargin</code>	Entier spécifiant la marge droite des paragraphes.
<code>textAttrs.rotation</code>	Valeur booléenne. Si sa valeur est <code>true</code> , Flash fait pivoter les caractères du texte à 90°. La valeur par défaut est <code>false</code> .
<code>textAttrs.size</code>	Entier définissant la taille de la police.
<code>textAttrs.target</code>	Chaîne représentant la propriété <code>target</code> du champ de texte.
<code>textAttrs.url</code>	Chaîne représentant la propriété <code>URL</code> du champ de texte.



## textAttrs.aliasText

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.aliasText`

### Description

Propriété ; valeur booléenne spécifiant que Flash doit tracer le texte à l'aide d'une méthode optimisée pour améliorer la lisibilité du texte en petits caractères.

### Exemple

L'exemple suivant donne la valeur `true` à la propriété `aliasText` de tout le texte du champ de texte sélectionné :

```
fl.getDocumentDOM().setElementTextAttr('aliasText', true);
```

## textAttrs.alignment

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.alignment`

### Description

Propriété ; chaîne qui spécifie la justification des paragraphes. Les valeurs possibles sont "left", "center", "right" et "justify".

### Exemple

L'exemple suivant justifie le paragraphe qui contient le texte compris entre le caractère d'index 0 et le caractère d'index 3 non compris : Cette opération peut affecter des caractères situés à l'extérieur de la plage spécifiée s'ils se trouvent dans le même paragraphe.

```
fl.getDocumentDOM().setTextSelection(0, 3);  
fl.getDocumentDOM().setElementTextAttr("alignment", "justify");
```

## textAttrs.autoKern

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.autoKern`

### Description

Propriété ; valeur booléenne qui détermine si Flash utilise (`true`) ou ignore (`false`) les informations de crénage (distance entre deux lettres) des polices de caractères.

### Exemple

L'exemple suivant sélectionne le texte compris entre le caractère d'index 2 et le caractère d'index 6 non compris, puis donne la valeur `true` à la propriété `autoKern` :

```
fl.getDocumentDOM().setTextSelection(3, 6);  
fl.getDocumentDOM().setElementTextAttr('autoKern', true);
```

## textAttrs.bold

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.bold`

### Description

Propriété : valeur booléenne. La valeur `true` provoque l'apparition de la version gras de la police.

### Exemple

L'exemple suivant sélectionne le premier caractère de l'objet Text sélectionné, puis donne la valeur `true` à la propriété `bold` :

```
fl.getDocumentDOM().setTextSelection(0, 1);  
fl.getDocumentDOM().setElementTextAttr('bold', true);
```

## textAttrs.characterPosition

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.characterPosition`

### Description

Propriété ; chaîne qui détermine la ligne de base du texte. Les valeurs possibles sont `"normal"`, `"subscript"` et `"superscript"`. Cette propriété ne s'applique qu'au texte statique.

### Exemple

L'exemple suivant sélectionne le texte compris entre le caractère d'index 2 et le caractère d'index 6 non compris, puis donne la valeur `subscript` à la propriété `characterPosition` :

```
fl.getDocumentDOM().setTextSelection(2, 6);  
fl.getDocumentDOM().setElementTextAttr("characterPosition", "subscript");
```

## `textAttrs.characterSpacing`

### Disponibilité

Flash MX 2004. Délaissée dans Flash 8 au profit de `textAttrs.letterSpacing`.

### Utilisation

`textAttrs.characterSpacing`

### Description

Propriété ; nombre entier représentant l'espace entre les caractères. Les valeurs gérées sont comprises entre -60 et 60.

Cette propriété est disponible uniquement avec les champs de texte statique, elle déclenche un avertissement avec un autre type de champ de texte.

### Exemple

L'exemple suivant donne la valeur 10 à l'espace des caractères du champ de texte sélectionné :

```
fl.getDocumentDOM().setElementTextAttr("characterSpacing", 10);
```

## `textAttrs.face`

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.face`

### Description

Propriété ; chaîne qui représente le nom de la police, par exemple "Arial".

### Exemple

L'exemple suivant donne la valeur `Arial` à la police de caractères du texte sélectionné, à partir du caractère d'index 2 et jusqu'au caractère d'index 8 (non inclus) :

```
fl.getDocumentDOM().selection[0].setTextAttr("face", "Arial", 2, 8);
```

## `textAttrs.fillColor`

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.fillColor`

### Description

Propriété ; couleur du remplissage, exprimée dans l'un des formats suivants :

- Chaîne au format "#RRVVBB" ou "#RRVVBAA"
- Nombre hexadécimal au format 0xRRVVBB
- Entier représentant l'équivalent décimal d'un nombre hexadécimal

### Exemple

L'exemple suivant donne la valeur rouge à la couleur du texte sélectionné, à partir du caractère d'index 2 et jusqu'au caractère d'index 8 (non inclus) :

```
fl.getDocumentDOM().selection[0].setTextAttr("fillColor", 0xff0000, 2, 8);
```

## textAttrs.indent

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.indent`

### Description

Propriété ; entier spécifiant l'indentation des paragraphes. Les valeurs gérées sont comprises entre -720 et 720.

### Exemple

L'exemple suivant donne la valeur 100 à l'indentation du champ de texte sélectionné, à partir du caractère d'index 2 et jusqu'au caractère d'index 8 (non inclus). Cette opération peut affecter des caractères situés à l'extérieur de la plage spécifiée s'ils se trouvent dans le même paragraphe.

```
fl.getDocumentDOM().selection[0].setTextAttr("indent", 100, 2, 8);
```

## textAttrs.italic

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.italic`

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, Flash trace le texte à l'aide de la version italique de la police.

### Exemple

L'exemple suivant met en italique le champ de texte sélectionné :

```
fl.getDocumentDOM().selection[0].setTextAttr("italic", true);
```

## **textAttrs.leftMargin**

### **Disponibilité**

Flash MX 2004.

### **Utilisation**

`textAttrs.leftMargin`

### **Description**

Propriété ; entier spécifiant la marge gauche des paragraphes. Les valeurs gérées sont comprises entre 0 et 720.

### **Exemple**

L'exemple suivant donne la valeur 100 à la propriété `leftMargin` du champ de texte sélectionné, à partir du caractère d'index 2 et jusqu'au caractère d'index 8 (non inclus). Cette opération peut affecter des caractères situés à l'extérieur de la plage spécifiée s'ils se trouvent dans le même paragraphe.

```
fl.getDocumentDOM().selection[0].setTextAttr("leftMargin", 100, 2, 8);
```

## **textAttrs.letterSpacing**

### **Disponibilité**

Flash 8.

### **Utilisation**

`textAttrs.letterSpacing`

### **Description**

Propriété ; nombre entier représentant l'espacement entre les caractères. Les valeurs gérées sont comprises entre -60 et 60.

Cette propriété est disponible uniquement avec les champs de texte statique, elle déclenche un avertissement avec un autre type de champ de texte.

### **Exemple**

Le code suivant sélectionne les caractères de l'index 0 jusqu'à l'index 10 (sans inclure celui-ci) et fixe l'espacement des caractères à 60 :

```
fl.getDocumentDOM().setTextSelection(0, 10);  
fl.getDocumentDOM().setElementTextAttr("letterSpacing", 60);
```

## **textAttrs.lineSpacing**

### **Disponibilité**

Flash MX 2004.

### Utilisation

`textAttrs.lineSpacing`

### Description

Propriété ; entier spécifiant l'interlignage (*leading*) du paragraphe. Les valeurs gérées sont comprises entre -360 et 720.

### Exemple

L'exemple suivant donne la valeur 100 à la propriété `lineSpacing` du champ de texte sélectionné :

```
fl.getDocumentDOM().selection[0].setTextAttr("lineSpacing", 100);
```

## `textAttrs.rightMargin`

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.rightMargin`

### Description

Propriété ; entier spécifiant la marge droite des paragraphes. Les valeurs gérées sont comprises entre 0 et 720.

### Exemple

L'exemple suivant donne la valeur 100 à la propriété `rightMargin` du champ de texte sélectionné, à partir du caractère d'index 2 et jusqu'au caractère d'index 8 (non inclus). Cette opération peut affecter des caractères situés à l'extérieur de la plage spécifiée s'ils se trouvent dans le même paragraphe.

```
fl.getDocumentDOM().selection[0].setTextAttr("rightMargin", 100, 2, 8);
```

## `textAttrs.rotation`

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.rotation`

### Description

Propriété : valeur booléenne. Si sa valeur est `true`, Flash fait pivoter les caractères du texte à 90°. La valeur par défaut est `false`. Cette propriété est disponible uniquement avec les champs de texte statiques ayant une orientation verticale, et déclenche un avertissement avec un autre type de champ de texte.

### Exemple

L'exemple suivant donne la valeur `true` à la rotation du champ de texte sélectionné :

```
fl.getDocumentDOM().setElementTextAttr("rotation", true);
```

## textAttrs.size

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.size`

### Description

Propriété ; entier définissant la taille de la police.

### Exemple

L'exemple suivant lit la taille du caractère ayant l'index 2 et affiche le résultat dans le panneau Sortie :

```
fl.outputPanel.trace(fl.getDocumentDOM().selection[0].getTextAttr("size", 2));
```

## textAttrs.target

### Disponibilité

Flash MX 2004.

### Utilisation

`textAttrs.target`

### Description

Propriété ; chaîne représentant la propriété `target` du champ de texte. Cette propriété ne s'applique qu'au texte statique.

### Exemple

L'exemple suivant récupère la propriété `target` du champ de texte de la première image du calque supérieur de la séquence active, puis l'affiche dans le panneau Sortie :

```
fl.outputPanel.trace(fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].getTextAttr("target"));
```

## textAttrs.url

### Disponibilité

Flash MX 2004.

**Utilisation**`textAttrs.url`**Description**

Propriété ; chaîne représentant la propriété URL du champ de texte. Cette propriété ne s'applique qu'au texte statique.

**Exemple**

L'exemple suivant définit l'URL du champ de texte sélectionné sur `http://www.adobe.com` :

```
fl.getDocumentDOM().setElementTextAttr("url", "http://www.adobe.com");
```



# Chapitre 44 : Objet TextRun

## Disponibilité

Flash MX 2004.

## Description

L'objet TextRun représente un flux de caractères dont les attributs correspondent à toutes les propriétés de l'Objet [TextAttrs](#). Cet objet est une propriété de l'objet Text ([text.textRuns](#)).

## Résumé des propriétés

Outre celles de l'objet Text, l'objet TextRun fournit les propriétés suivantes :

Propriété	Description
<a href="#">textRun.characters</a>	Chaîne représentant le texte contenu dans l'objet TextRun.
<a href="#">textRun.textAttrs</a>	Objet TextAttrs contenant les attributs du segment de texte.

## textRun.textAttrs

### Disponibilité

Flash MX 2004.

### Utilisation

```
textRun.textAttrs
```

### Description

Propriété ; [Objet TextAttrs](#) contenant les attributs du segment de texte.

### Exemple

L'exemple suivant affiche dans le panneau Sortie les propriétés de la première suite de caractères du champ de texte sélectionné.

```
var curTextAttrs = fl.getDocumentDOM().selection[0].textRuns[0].textAttrs;
for (var prop in curTextAttrs) {
    fl.trace(prop + " = " + curTextAttrs[prop]);
}
```

## textRun.characters

### Disponibilité

Flash MX 2004.

### Utilisation

```
textRun.characters
```

**Description**

Propriété ; texte contenu dans l'objet TextRun.

**Exemple**

L'exemple suivant affiche dans le panneau Sortie les caractères qui forment la première suite de caractères du champ de texte sélectionné :

```
fl.trace(fl.getDocumentDOM().selection[0].textRuns[0].characters);
```

# Chapitre 45 : Objet Timeline

## Disponibilité

Flash MX 2004.

## Description

L'objet Timeline représente le scénario de Flash qui est accessible, pour le document actif, par le biais de `fl.getDocumentDOM().getTimeline()`. Cette méthode renvoie le scénario de la séquence actuelle ou du symbole en cours de modification.

Lorsque l'utilisateur travaille sur les séquences, le scénario de chaque séquence possède une valeur d'index qui permet d'y accéder (pour le document actif) via `fl.getDocumentDOM().timelines[i]`. (Dans cet exemple, *i* est l'index de la valeur du scénario.)

Si vous travaillez sur les images en utilisant les méthodes et les propriétés de l'objet Timeline, n'oubliez pas que l'index des images est de base zéro et n'est donc pas égal au numéro de l'image dans la séquence d'images du scénario. Autrement dit, l'image n°1 possède l'index 0.

## Résumé des méthodes

Les méthodes suivantes sont disponibles pour l'objet Timeline :

Méthode	Description
<code>timeline.addMotionGuide()</code>	Ajoute un calque de guide de mouvement au-dessus du calque actif et rattache ce calque actif au nouveau calque de guide de mouvement.
<code>timeline.addNewLayer()</code>	Ajoute un nouveau calque au document et l'active.
<code>timeline.clearFrames()</code>	Supprime tout le contenu d'une image ou d'une plage d'images dans le calque actif.
<code>timeline.clearKeyframes()</code>	Convertit une image-clé en image normale et supprime son contenu du calque actif.
<code>timeline.convertToBlankKeyframes()</code>	Convertit des images en images-clés vierges dans le calque actif.
<code>timeline.convertToKeyframes()</code>	Convertit une plage d'images en images-clés (ou convertit la sélection si aucune image n'est spécifiée) dans le calque actif.
<code>timeline.copyFrames()</code>	Copie une plage d'images du calque actif dans le Presse-papiers.
<code>timeline.copyLayers()</code>	Copie une plage de calques du scénario dans le Presse-papiers.
<code>timeline.copyMotion()</code>	A partir d'une interpolation de mouvement ou d'une animation image par image, copie un mouvement sur les images sélectionnées pour pouvoir l'appliquer à d'autres images.
<code>timeline.copyMotionAsAS3()</code>	A partir d'une interpolation de mouvement ou d'une animation image par image, copie via le Presse-papiers un mouvement dans les images sélectionnées sous forme de code ActionScript 3.0.
<code>timeline.createMotionObject()</code>	Crée un nouvel objet de mouvement à une image de début ou de fin désignée.
<code>timeline.createMotionTween()</code>	Définit la propriété <code>frame.tweenType</code> sur <code>motion</code> pour toutes les images-clés sélectionnées dans le calque actif, et convertit le contenu de chaque image en une occurrence de symbole unique si nécessaire.

Méthode	Description
<code>timeline.cutFrames()</code>	A partir du scénario, coupe une plage d'images du calque actif et la place dans le Presse-papiers.
<code>timeline.cutLayers()</code>	Coupe une plage de calques du scénario et les enregistre dans le Presse-papiers.
<code>timeline.deleteLayer()</code>	Supprime un calque
<code>timeline duplicateLayers()</code>	Duplique les calques sélectionnés ou les calques spécifiés.
<code>timeline.expandFolder()</code>	Développe ou réduit le ou les dossiers spécifiés.
<code>timeline.findLayerIndex()</code>	Recherche le nom de calque spécifié dans un tableau d'index.
<code>timeline.getFrameProperty()</code>	Récupère la valeur de la propriété spécifiée pour les images sélectionnées.
<code>timeline.getGuidelines()</code>	Renvoie une chaîne XML représentant la position actuelle des lignes horizontale et verticale des guides pour un scénario (Affichage > Guides > Afficher les guides).
<code>timeline.getLayerProperty()</code>	Récupère la valeur de la propriété spécifiée pour les calques sélectionnés.
<code>timeline.getSelectedFrames()</code>	Récupère dans un tableau les images actuellement sélectionnées.
<code>timeline.getSelectedLayers()</code>	Récupère les valeurs d'index de base zéro des calques sélectionnés.
<code>timeline.insertBlankKeyframe()</code>	Insère une image-clé vierge à l'index spécifié dans le tableau des images. Si l'index n'est pas spécifié, une image-clé vierge est insérée au niveau de la tête de lecture ou de la sélection.
<code>timeline.insertFrames()</code>	Insère le nombre d'images spécifié au numéro d'image indiqué.
<code>timeline.insertKeyframe()</code>	Insère une image-clé au numéro d'image spécifié.
<code>timeline.pasteFrames()</code>	Colle la plage d'images du Presse-papiers dans les images spécifiées.
<code>timeline.pasteLayers()</code>	Colle les calques copiés sur le scénario au-dessus de l'index du calque spécifié.
<code>timeline.pasteMotion()</code>	Copie la plage d'images de mouvement récupérée par <code>timeline.copyMotion()</code> dans le scénario.
<code>timeline.removeFrames()</code>	Supprime l'image.
<code>timeline.removeMotionObject()</code>	Supprime l'objet de mouvement créé avec <code>timeline.createMotionObject()</code> et convertit la ou les images en images statiques.
<code>timeline.reorderLayer()</code>	Déplace le premier calque spécifié avant ou après le second calque spécifié.
<code>timeline.reverseFrames()</code>	Inverse une plage d'images.
<code>timeline.selectAllFrames()</code>	Sélectionne toutes les images du scénario actif.
<code>timeline setFrameProperty()</code>	Définit la propriété de l'objet Frame pour les images sélectionnées.
<code>timeline.setGuidelines()</code>	Remplace les guides du scénario par les informations spécifiées.
<code>timeline.setLayerProperty()</code>	Donne une valeur spécifiée à la propriété spécifiée pour tous les calques sélectionnés.

Méthode	Description
<code>timeline.setSelectedFrames()</code>	Sélectionne une plage d'images dans le calque actif, ou applique les images sélectionnées au tableau de sélection transmis à cette méthode.
<code>timeline.setSelectedLayers()</code>	Sélectionne le calque spécifié et l'active.
<code>timeline.showLayerMasking()</code>	Affiche les masques de calque pendant la programmation, en verrouillant le masque et les calques masqués.
<code>timeline.startPlayback()</code>	Démarre la lecture automatique du scénario si ce dernier n'est pas en cours de lecture.
<code>timeline.stopPlayback()</code>	Arrête la lecture automatique du scénario si ce dernier est en cours de lecture.

### Résumé des propriétés

Les paramètres suivants sont disponibles avec l'objet Timeline :

Propriété	Description
<code>timeline.currentFrame</code>	Index (de base zéro) de l'image qui se trouve au niveau de la tête de lecture.
<code>timeline.currentLayer</code>	Index (de base zéro) du calque actif.
<code>timeline.frameCount</code>	Lecture seule ; nombre entier indiquant le nombre d'images du calque le plus long du scénario actif.
<code>timeline.layerCount</code>	Lecture seule ; nombre entier indiquant le nombre de calques du scénario spécifié.
<code>timeline.layers</code>	Lecture seule ; tableau d'objets layer.
<code>timeline.libraryItem</code>	Propriété en lecture seule ; indique si le scénario appartient à une séquence.
<code>timeline.name</code>	Chaîne indiquant le nom du scénario actif.

## timeline.addMotionGuide()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.addMotionGuide()
```

### Paramètres

Aucun.

### Valeur renvoyée

Entier représentant l'index (de base zéro) du nouveau calque de guidage. Si le calque actif n'est pas de type "Normal", Flash renvoie -1.

### Description

Méthode qui ajoute un calque de guide de mouvement au-dessus du calque actif et les relie entre eux grâce à la conversion du calque actif en calque de type "Guided".

Cette méthode ne fonctionne qu'avec un calque de type "Normal". Elle n'a aucun d'effet sur un calque de type "Folder", "Mask", "Masked", "Guide" ou "Guided".

### Exemple

L'exemple suivant ajoute un calque de guide de mouvement au-dessus du calque actif et convertit ce dernier en type Guided (guidé) :

```
fl.getDocumentDOM().getTimeline().addMotionGuide();
```

## timeline.addNewLayer()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.addNewLayer([name] [, layerType [, bAddAbove]])
```

### Paramètres

**name** Chaîne spécifiant le nom du nouveau calque. Si ce paramètre est omis, un nouveau nom par défaut est affecté au nouveau calque ("Layer n", où *n* correspond au nombre total de calques). Ce paramètre est facultatif.

**layerType** Chaîne qui spécifie le type du calque à ajouter. Si vous omettez ce paramètre, un calque de type « Normal » est créé. Ce paramètre est facultatif. Les valeurs gérées sont : "normal", "guide", "guided", "mask", "masked" et "folder".

**bAddAbove** Valeur booléenne. Si elle est `true` (valeur par défaut), Flash ajoute le nouveau calque au-dessus du calque actif ; si elle vaut `false`, Flash ajoute le nouveau calque sous le calque actif. Ce paramètre est facultatif.

### Valeur renvoyée

Entier représentant l'index (de base zéro) du nouveau calque.

### Description

Méthode qui ajoute un nouveau calque au document et l'active.

### Exemple

L'exemple suivant ajoute un nouveau calque au scénario, avec un nom par défaut généré par Flash :

```
fl.getDocumentDOM().getTimeline().addNewLayer();
```

L'exemple suivant ajoute un nouveau calque de dossier au-dessus du calque actif et le nomme Folder1 :

```
fl.getDocumentDOM().getTimeline().addNewLayer("Folder1", "folder", true);
```

## timeline.clearFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.clearFrames([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui définit le début de la plage d'images à effacer. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui définit la fin de la plage d'images à effacer. Cette plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime tout le contenu d'une image ou d'une plage d'images dans le calque actif.

### Exemple

L'exemple suivant efface les images à partir de l'image 6 et jusqu'à l'image 11 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().clearFrames(5, 10);
```

L'exemple suivant efface l'image 15 :

```
fl.getDocumentDOM().getTimeline().clearFrames(14);
```

## timeline.clearKeyframes()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.clearKeyframes([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui définit le début de la plage d'images à effacer. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui définit la fin de la plage d'images à effacer. Cette plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui convertit une image-clé en image standard et supprime son contenu du calque actif.

### Exemple

L'exemple suivant efface les images-clés à partir de l'image 5 et jusqu'à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().clearKeyframes(4, 9);
```

L'exemple suivant efface l'image-clé 15 et la convertit en image normale :

```
fl.getDocumentDOM().getTimeline().clearKeyframes(14);
```

## timeline.convertToBlankKeyframes()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.convertToBlankKeyframes([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui spécifie l'image de début à convertir en image-clé. Si le paramètre *startFrameIndex* est omis, la méthode convertit les images actuellement sélectionnées. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image à convertir en image-clé. La plage à convertir s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui convertit des images en images-clés vierges dans le calque actif.

### Exemple

L'exemple suivant convertit en images vides les images-clés à partir de l'image 2 et jusqu'à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().convertToBlankKeyframes(1, 9);
```

L'exemple suivant convertit l'image 5 en image-clé vide :

```
fl.getDocumentDOM().getTimeline().convertToBlankKeyframes(4);
```

## timeline.convertToKeyframes()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.convertToKeyframes([startFrameIndex [, endFrameIndex]])
```



### Paramètres

**startFrameIndex** Index de base zéro qui définit la première image à convertir en image-clé. Si le paramètre *startFrameIndex* est omis, la méthode convertit les images actuellement sélectionnées. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image à convertir en image-clé. La plage à convertir s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui convertit une plage d'images-clés en images standard (ou convertit la sélection si aucune image n'est spécifiée) dans le calque actif.

### Exemple

L'exemple suivant convertit les images sélectionnées en images-clés :

```
fl.getDocumentDOM().getTimeline().convertToKeyframes();
```

L'exemple suivant convertit en images-clés la plage d'images allant de l'image 2 à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().convertToKeyframes(1, 9);
```

L'exemple suivant convertit l'image 5 en image-clé :

```
fl.getDocumentDOM().getTimeline().convertToKeyframes(4);
```

## timeline.copyFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.copyFrames([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui spécifie le début de la plage d'images à copier. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image à copier. La plage d'images à copier s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui copie une plage d'images du calque actif vers le Presse-papiers.

### Exemple

L'exemple suivant copie les images sélectionnées dans le Presse-papiers :

```
fl.getDocumentDOM().getTimeline().copyFrames();
```

L'exemple suivant copie dans le Presse-papiers la plage d'images allant de l'image 2 à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().copyFrames(1, 9);
```

L'exemple suivant copie l'image 5 dans le Presse-papiers :

```
fl.getDocumentDOM().getTimeline().copyFrames(4);
```

## timeline.copyLayers()

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
timeline.copyLayers([startLayerIndex [, endLayerIndex]])
```

### Paramètres

**startLayerIndex** Facultatif. Index de base zéro qui spécifie le début de la plage de calques à copier. Si le paramètre **startLayerIndex** est omis, la méthode utilise la sélection actuelle.

**endLayerIndex** Facultatif. Index de base zéro qui spécifie le dernier calque à copier. La plage de calques s'étend jusqu'à **endLayerIndex** compris. Si seul **startLayerIndex** est transmis, la valeur de **endLayerIndex** prend par défaut la valeur de **startLayerIndex**.

### Valeur renvoyée

Aucune.

### Description

Méthode : copie les calques actuellement sélectionnés dans le scénario, ou les calques dans la plage spécifiée. Il est possible de fournir des arguments facultatifs en vue de spécifier un calque ou une plage de calques à copier.

### Exemple

L'exemple suivant copie les calques de l'index 2 sur l'index 7 dans le scénario :

```
fl.getDocumentDOM().getTimeline().copyLayers(2, 7);
```

### Voir aussi

[timeline.cutLayers\(\)](#), [timeline.pasteLayers\(\)](#), [timeline.duplicateLayers\(\)](#)

## timeline.copyMotion()

### Disponibilité

Flash CS3 Professional.

### Utilisation

`timeline.copyMotion()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui copie un mouvement dans les images sélectionnées, soit d'une interpolation de mouvement, soit d'une animation image par image. Vous pouvez ensuite utiliser `timeline.pasteMotion()` pour appliquer ce mouvement à d'autres images.

Pour savoir comment copier un mouvement sous forme de texte (code) à coller ensuite dans un script, voir `timeline.copyMotionAsAS3()`.

### Exemple

L'exemple suivant copie le mouvement à partir de l'image ou des images sélectionnées :

```
fl.getDocumentDOM().getTimeline().copyMotion();
```

### Voir aussi

`timeline.copyMotionAsAS3()`, `timeline.pasteMotion()`

## timeline.copyMotionAsAS3()

### Disponibilité

Flash CS3 Professional.

### Utilisation

`timeline.copyMotionAsAS3()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui, à partir d'une interpolation de mouvement ou d'une animation image par image, copie sous forme de code ActionScript 3.0. via le Presse-papiers un mouvement dans les images sélectionnées. Vous pouvez ensuite coller ce code dans un script.

Pour savoir comment copier un mouvement dans un format applicable à d'autres images, voir `timeline.copyMotion()`.

### Exemple

L'exemple suivant copie dans le Presse-papiers, sous forme de code ActionScript 3.0, le mouvement à partir de l'image ou des images sélectionnées :

```
fl.getDocumentDOM().getTimeline().copyMotionAsAS3();
```

### Voir aussi

[timeline.copyMotion\(\)](#)

## timeline.createMotionObject()

### Disponibilité

Flash Professional CS5

### Utilisation

```
timeline.createMotionObject([startFrame [,endFrame]])
```

### Paramètres

**startFrame** Spécifie la première image à laquelle créer les objets de mouvement. En cas d'omission du paramètre *startFrame*, la méthode utilise la sélection actuelle. S'il n'existe pas de sélection, toutes les images (de tous les calques) qui se trouvent au niveau de la tête de lecture sont supprimées. Ce paramètre est facultatif.

**endFrame** Spécifie l'image à laquelle arrêter la création d'objets de mouvement ; la plage d'images s'étend jusqu'à *endFrame* (non compris). Si seul le paramètre *startFrame* est spécifié, la valeur de *endFrame* prend par défaut la valeur de *startFrame*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode ; crée un nouvel objet de mouvement. Ces paramètres sont facultatifs. S'ils sont spécifiés, la sélection du scénario est définie sur les images indiquées avant la création de l'objet de mouvement.

### Exemple

L'exemple suivant crée un objet de mouvement à la position actuelle de la tête de lecture sur le calque supérieur :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().createMotionObject();
```

L'exemple suivant crée un objet de mouvement à partir de l'image 5 jusqu'à l'image 15 (non comprise) du calque supérieur dans la séquence actuelle :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().createMotionObject(5, 15);
```

## timeline.createMotionTween()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.createMotionTween([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui spécifie l'image de début de l'interpolation. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image de l'interpolation. La plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la propriété `frame.tweenType` sur `motion` pour toutes les images-clés sélectionnées dans le calque actif, et convertit le contenu de chaque image en une occurrence de symbole unique si nécessaire. Cette propriété est l'équivalent de l'option de menu « Créer une interpolation de mouvement » dans l'interface de Flash.

### Exemple

L'exemple suivant convertit en occurrence de symbole graphique la forme qui se trouve dans la première image jusqu'à l'image 10 non comprise, et donne la valeur `motion` à la propriété `frame.tweenType` (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().createMotionTween(0, 9);
```

## timeline.currentFrame

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.currentFrame
```

### Description

Propriété ; index de base zéro de l'image qui se trouve actuellement au niveau de la tête de lecture.

### Exemple

L'exemple suivant place la tête de lecture face à l'image 10 du scénario actif (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().currentFrame = 9;
```

L'exemple suivant enregistre dans la variable `curFrame` l'emplacement actuel de la tête de lecture :

```
var curFrame = fl.getDocumentDOM().getTimeline().currentFrame;
```

## timeline.currentLayer

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.currentLayer
```

### Description

Propriété ; index (de base zéro) du calque actif. La valeur 0 spécifie le calque supérieur, la valeur 1 indique le calque placé juste en dessous, etc.

### Exemple

L'exemple suivant active le calque supérieur :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
```

L'exemple suivant enregistre dans la variable `curLayer` l'index du calque actif :

```
var curLayer = fl.getDocumentDOM().getTimeline().currentLayer;
```

## timeline.cutFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.cutFrames([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui définit le début de la plage d'images à couper. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui définit la dernière image à couper. La plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui, à partir du scénario, coupe une plage d'images du calque actif et la place dans le Presse-papiers.

### Exemple

L'exemple suivant coupe les images sélectionnées dans le scénario et les place dans le Presse-papiers :

```
fl.getDocumentDOM().getTimeline().cutFrames();
```

L'exemple suivant coupe et place dans le Presse-papiers la plage d'images, de l'image 2 à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image):

```
fl.getDocumentDOM().getTimeline().cutFrames(1, 9);
```

L'exemple suivant coupe l'image 5 dans le scénario et la place dans le Presse-papiers :

```
fl.getDocumentDOM().getTimeline().cutFrames(4);
```

## timeline.cutLayers()

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
timeline.cutLayers([startLayerIndex [, endLayerIndex]])
```

### Paramètres

**startLayerIndex** Facultatif. Index de base zéro qui spécifie le début de la plage de calques à couper. Si le paramètre `startLayerIndex` est omis, la méthode utilise la sélection actuelle.

**endLayerIndex** Facultatif. Index de base zéro qui spécifie le calque à partir duquel la découpe doit s'arrêter. La plage de calques à couper s'étend jusqu'à `endLayerIndex` compris. Si seul `startLayerIndex` est transmis, la valeur de `endLayerIndex` prend par défaut la valeur de `startLayerIndex`.

### Valeur renvoyée

Aucune.

### Description

Méthode : coupe les calques actuellement sélectionnés dans le scénario, ou les calques dans la plage spécifiée. Il est possible de fournir des arguments facultatifs en vue de spécifier un calque ou une plage de calques à couper.

### Exemple

L'exemple suivant coupe les calques de l'index 2 sur l'index 7 dans le scénario :

```
fl.getDocumentDOM().getTimeline().cutLayers(2, 7);
```

### Voir aussi

[timeline.copyLayers\(\)](#), [timeline.pasteLayers\(\)](#), [timeline.duplicateLayers\(\)](#)

## timeline.deleteLayer()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.deleteLayer([index])
```

### Paramètres

**index** Nombre entier de base zéro indiquant le calque à supprimer. S'il n'existe qu'un seul calque dans le scénario, cette méthode n'a aucun effet. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui supprime un calque. S'il s'agit d'un calque de type Dossier, tous les calques de ce dossier sont supprimés. Si l'index du calque n'est pas spécifié, Flash supprime les calques actuellement sélectionnés.

### Exemple

L'exemple suivant supprime le second calque à partir du haut :

```
fl.getDocumentDOM().getTimeline().deleteLayer(1);
```

L'exemple suivant supprime les calques sélectionnés :

```
fl.getDocumentDOM().getTimeline().deleteLayer();
```

## timeline duplicateLayers()

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
timeline.duplicateLayers([startLayerIndex [, endLayerIndex]])
```

### Paramètres

**startLayerIndex** Facultatif. Index de base zéro qui spécifie le début de la plage de calques à copier. Il spécifie également le calque au-dessus duquel les calques du Presse-papiers sont collés. Si le paramètre `startLayerIndex` est omis, la méthode utilise la sélection de calques actuelle.

**endLayerIndex** Facultatif. Index de base zéro qui spécifie le dernier calque à copier. La plage de calques s'étend jusqu'à `endLayerIndex` compris. Si seul `startLayerIndex` est transmis, la valeur de `endLayerIndex` prend par défaut la valeur de `startLayerIndex`.

### Valeur renvoyée

Aucune.

### Description

Méthode : duplique les calques actuellement sélectionnés dans le scénario, ou les calques dans la plage spécifiée. Il est possible de fournir des arguments facultatifs en vue de spécifier un calque ou une plage de calques à dupliquer.

### Exemple

L'exemple suivant duplique le calque actuellement sélectionné dans le scénario :



```
fl.getDocumentDOM().getTimeline().duplicateLayers();
```

L'exemple suivant duplique les calques de l'index 2 sur l'index 7 au-dessus de l'index du calque 2 :

```
fl.getDocumentDOM().getTimeline().duplicatedLayers(2,7);
```

#### Voir aussi

[timeline.copyLayers\(\)](#), [timeline.cutLayers\(\)](#), [timeline.pasteLayers\(\)](#)

## timeline.expandFolder()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
timeline.expandFolder(bExpand [, bRecurseNestedParents [, index]])
```

#### Paramètres

**bExpand** Valeur booléenne. Si elle est `true`, la méthode développe le dossier ; si elle est `false`, la méthode réduit le dossier.

**bRecurseNestedParents** Valeur booléenne. Si elle est `true`, tous les calques du dossier spécifié sont ouverts ou fermés selon la valeur du paramètre *bExpand*. Ce paramètre est facultatif.

**index** Index de base zéro du dossier à développer ou réduire. Utilisez la valeur -1 pour l'appliquer à tous les calques (le paramètre *bRecurseNestedParents* doit également avoir la valeur `true`). Cette propriété est l'équivalent de l'option de menu « Développer/réduire tous les dossiers » dans l'outil de création de Flash. Ce paramètre est facultatif.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui développe ou réduit le ou les dossiers spécifiés. Si vous ne spécifiez pas de calque, cette méthode opère sur le calque actif.

#### Exemple

Les exemples suivants utilisent cette structure de dossiers :

```
Folder 1 ***  
--layer 7  
--Folder 2 ****  
---Layer 5
```

L'exemple suivant développe le dossier Folder 1 uniquement :

```
fl.getDocumentDOM().getTimeline().currentLayer = 1;  
fl.getDocumentDOM().getTimeline().expandFolder(true);
```

L'exemple suivant développe le dossier Folder 1 uniquement (en supposant que Folder 2 était réduit lorsque Folder 1 a été réduit précédemment, sinon Folder 2 est également développé) :

```
fl.getDocumentDOM().getTimeline().expandFolder(true, false, 0);
```

L'exemple suivant réduit tous les dossiers du scénario actuel :

```
fl.getDocumentDOM().getTimeline().expandFolder(false, true, -1);
```

## timeline.findLayerIndex()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.findLayerIndex(name)
```

### Paramètres

**name** Chaîne spécifiant le nom du calque à rechercher.

### Valeur renvoyée

Tableau de valeurs d'index pour le calque spécifié. Si le calque spécifié est introuvable, Flash renvoie `undefined`.

### Description

Méthode qui recherche le nom du calque spécifié dans un tableau d'index. Le tableau des calques possède une seule dimension, si bien que les dossiers font partie de l'index principal.

### Exemple

L'exemple suivant affiche dans le panneau Sortie la valeur d'index de tous les calques nommés Layer 7 :

```
var layerIndex = fl.getDocumentDOM().getTimeline().findLayerIndex("Layer 7");  
fl.trace(layerIndex);
```

L'exemple suivant montre comment transmettre les valeurs renvoyées par cette méthode à

`timeline.setSelectedLayers()` :

```
var layerIndex = fl.getDocumentDOM().getTimeline().findLayerIndex("Layer 1");  
fl.getDocumentDOM().getTimeline().setSelectedLayers(layerIndex[0], true);
```

## timeline.frameCount

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.frameCount
```

### Description

Propriété en lecture seule ; entier indiquant le nombre d'images du calque le plus long du scénario actif.

### Exemple

L'exemple suivant enregistre dans la variable `countNum` le nombre d'images du calque le plus long du document actif :

```
var countNum = fl.getDocumentDOM().getTimeline().frameCount;
```

## timeline.getFrameProperty()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.getFrameProperty(property [, startFrameIndex [, endFrameIndex]])
```

### Paramètres

**property** Chaîne spécifiant le nom de la propriété dont la valeur est recherchée. La liste complète des propriétés est indiquée dans le Résumé des propriétés de l'[Objet Frame](#).

**startFrameIndex** Index de base zéro qui spécifie l'image de début dont la valeur est recherchée. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la fin de la plage d'images à sélectionner. Cette plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Valeur de la propriété spécifiée, ou *undefined* si les images sélectionnées n'ont pas toutes la même valeur pour cette propriété.

### Description

Méthode qui récupère la valeur de la propriété spécifiée pour les images sélectionnées.

### Exemple

L'exemple suivant récupère le nom de la première image du calque supérieur du document actif, et affiche ce nom dans le panneau Sortie :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().setSelectedFrames(0, 0, true);  
var frameName = fl.getDocumentDOM().getTimeline().getFrameProperty("name");  
fl.trace(frameName);
```

## timeline.getGuidelines()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
timeline.getGuidelines()
```

### Paramètres

Aucun.

### Valeur renvoyée

Chaîne XML.

### Description

Méthode qui renvoie une chaîne XML représentant la position actuelle des lignes horizontale et verticale des guides pour un scénario (Affichage > Guides > Afficher les guides). Pour appliquer ces guides à un scénario, utilisez `timeline.setGuidelines()`.

### Exemple

En supposant que le premier scénario contienne des guides, l'exemple suivant les affiche sous forme de chaîne XML dans le panneau Sortie :

```
var currentTimeline = fl.getDocumentDOM().timelines[0];  
fl.trace(currentTimeline.getGuidelines());
```

## timeline.getLayerProperty()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.getLayerProperty(property)
```

### Paramètres

**property** Chaîne spécifiant le nom de la propriété dont la valeur est recherchée. La liste des propriétés est indiquée dans le Résumé des propriétés de l'[Objet Frame](#).

### Valeur renvoyée

Valeur de la propriété spécifiée. Flash examine les propriétés du calque pour en déterminer le type. Si les calques spécifiés n'ont pas tous la même valeur pour cette propriété, Flash renvoie `undefined`.

### Description

Méthode qui récupère la valeur de la propriété spécifiée pour les calques sélectionnés.

### Exemple

L'exemple suivant récupère le nom du calque supérieur du document actif, et l'affiche dans le panneau Sortie :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
var layerName = fl.getDocumentDOM().getTimeline().getLayerProperty("name");  
fl.trace(layerName);
```

## timeline.getSelectedFrames()

### Disponibilité

Flash MX 2004.

### Paramètres

Aucun.

### Valeur renvoyée

Tableau contenant  $3n$  entiers, où  $n$  est le nombre de zones sélectionnées. Le premier entier de chaque groupe est l'index du calque, le second est l'image de départ de la sélection, et le troisième indique l'image de fin de la sélection. L'image de fin n'est pas incluse dans la sélection.

### Description

Méthode qui recherche dans un tableau les images actuellement sélectionnées.

### Exemple

Si le calque supérieur est le calque actif, l'exemple suivant affiche 0, 5, 10, 0, 20, 25 dans le panneau Sortie :

```
var timeline = fl.getDocumentDOM().getTimeline();
timeline.setSelectedFrames(5, 10);
timeline.setSelectedFrames(20, 25, false);
var theSelectedFrames = timeline.getSelectedFrames();
fl.trace(theSelectedFrames);
```

### Voir aussi

[timeline.setSelectedFrames\(\)](#)

## timeline.getSelectedLayers()

### Disponibilité

Flash MX 2004.

### Paramètres

Aucun.

### Valeur renvoyée

Tableau des valeurs d'index de base zéro des calques sélectionnés.

### Description

Méthode qui recherche les valeurs d'index de base zéro des calques sélectionnés.

### Exemple

L'exemple suivant affiche 1, 0 dans le panneau Sortie.

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(0);
fl.getDocumentDOM().getTimeline().setSelectedLayers(1, false);
var layerArray = fl.getDocumentDOM().getTimeline().getSelectedLayers();
fl.trace(layerArray);
```

### Voir aussi

[timeline.setSelectedLayers\(\)](#)

## timeline.insertBlankKeyframe()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.insertBlankKeyframe([frameNumIndex])
```

### Paramètres

**frameNumIndex** Index de base zéro qui définit l'image à laquelle l'image-clé doit être insérée. Si le paramètre *frameNumIndex* est omis, la méthode utilise l'image qui se trouve en face de la tête de lecture. Ce paramètre est facultatif.

Si l'image spécifiée ou l'image sélectionnée est une image normale, l'image-clé est insérée au niveau de celle-ci. Par exemple, pour une plage de 10 images numérotées de 1 à 10, si vous sélectionnez l'image 5, cette méthode transforme l'image 5 en image-clé vide, et la longueur de la plage reste de 10 images. Si l'image 5 est sélectionnée et est une image-clé suivie d'une image normale, cette méthode insère une image-clé vide au niveau de l'image 6. Si l'image 5 est une image-clé et que la suivante est déjà une image-clé, aucune image-clé n'est insérée, mais la tête de lecture est placée en face de l'image 6.

### Valeur renvoyée

Aucune.

### Description

Méthode qui insère une image-clé vierge à l'index spécifié dans le tableau des images ; si l'index n'est pas spécifié, une image-clé vierge est insérée au niveau de la tête de lecture ou de la sélection. Voir aussi [timeline.insertKeyframe\(\)](#).

### Exemple

L'exemple suivant insère une image-clé vide au niveau de l'image 20 (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().insertBlankKeyframe(19);
```

L'exemple suivant insère une image-clé vide au niveau de l'image sélectionnée (ou au niveau de la tête de lecture si aucune image n'est sélectionnée) :

```
fl.getDocumentDOM().getTimeline().insertBlankKeyframe();
```

## timeline.insertFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.insertFrames([numFrames [, bAllLayers [, frameNumIndex]])
```

### Paramètres

**numFrames** Nombre entier qui indique le nombre d'images à insérer. Si ce paramètre est omis, la méthode insère les images au niveau de la sélection actuelle dans le calque actif. Ce paramètre est facultatif.

**bAllLayers** Valeur booléenne. Si elle est `true` (valeur par défaut), la méthode insère dans tous les calques le nombre d'images spécifié dans le paramètre *numFrames* ; si elle a la valeur `false`, la méthode insère ces images dans le calque actif. Ce paramètre est facultatif.

**frameNumIndex** Index de base zéro qui spécifie l'image au niveau de laquelle une nouvelle image doit être insérée. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui insère le nombre d'images spécifié au numéro d'index indiqué.

Si aucun paramètre n'est spécifié, cette méthode fonctionne comme suit :

- Si une ou plusieurs images sont sélectionnées, la méthode insère le nombre d'images sélectionnées à l'emplacement de la première image sélectionnée dans le calque actif. Autrement dit, si les images 6 à 10 sont sélectionnées (soit un total de cinq images), la méthode ajoute cinq images à l'image 6 dans le calque contenant les images sélectionnées.
- Si aucune image n'est sélectionnée, la méthode en insère une dans tous les calques, au niveau de l'image active.  
Si des paramètres sont spécifiés, cette méthode fonctionne comme suit :
- Si seul le paramètre *numFrames* est spécifié, la méthode insère le nombre d'images indiqué au niveau de l'image active dans le calque actif.
- Si le paramètre *numFrames* est spécifié et si le paramètre *bAllLayers* a la valeur `true`, la méthode insère le nombre d'images indiqué au niveau de l'image active dans tous les calques.
- Si les trois paramètres sont spécifiés, la méthode insère le nombre d'images indiqué au niveau de l'index spécifié (*frameIndex*) ; la valeur de *bAllLayers* détermine si les images sont ajoutées au calque actif seulement ou à tous les calques.

Si l'image spécifiée ou l'image sélectionnée est une image normale, l'image est insérée au niveau de celle-ci. Par exemple, pour une plage de 10 images numérotées de 1 à 10, si vous sélectionnez l'image 5 (ou si vous transmettez la valeur 4 pour le paramètre *frameIndex*), cette méthode ajoute une image à l'image 5, et la longueur de la plage devient 11 images. Si l'image 5 est sélectionnée et s'il s'agit d'une image-clé, cette méthode insère une image au niveau de l'image 6, que l'image suivante soit également une image-clé ou non.

### Exemple

L'exemple suivant insère une ou plusieurs images (selon l'étendue de la sélection) au niveau de la sélection actuelle dans le calque actif :

```
fl.getDocumentDOM().getTimeline().insertFrames();
```

L'exemple suivant insère cinq images au niveau de l'image active dans tous les calques :

```
fl.getDocumentDOM().getTimeline().insertFrames(5);
```

**Remarque :** si le document comporte plusieurs calques dotés d'images, et si vous avez sélectionné une image dans un calque lors de l'utilisation de la commande précédente, Flash insère les images dans le calque sélectionné uniquement. Si le document comporte plusieurs calques dotés d'images mais qu'aucune d'elles n'est sélectionnée, Flash insère les images dans tous les calques.

L'exemple suivant insère trois images dans le calque actif uniquement :

```
fl.getDocumentDOM().getTimeline().insertFrames(3, false);
```

L'exemple suivant insère quatre images dans tous les calques, en commençant à la première image :

```
fl.getDocumentDOM().getTimeline().insertFrames(4, true, 0);
```

## timeline.insertKeyframe()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.insertKeyframe([frameNumIndex])
```

### Paramètres

**frameNumIndex** Index de base zéro qui spécifie l'index de l'image au niveau de laquelle l'image-clé doit être insérée dans le calque actif. Si le paramètre *frameNumIndex* est omis, la méthode utilise l'index de l'image qui se trouve en face de la tête de lecture ou de l'image sélectionnée. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui insère une image-clé au numéro d'image indiqué. Si ce paramètre est omis, la méthode insère une image-clé au niveau de la sélection actuelle ou de la tête de lecture.

Cette méthode donne un résultat identique à celui de [timeline.insertBlankKeyframe\(\)](#), si ce n'est que l'image-clé insérée détient le contenu de l'image convertie et n'est donc pas vide.

### Exemple

L'exemple suivant insère une image-clé au niveau de la tête de lecture ou de l'image sélectionnée :

```
fl.getDocumentDOM().getTimeline().insertKeyframe();
```

L'exemple suivant insère une image-clé au niveau de l'image 10 du second calque (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().currentLayer = 1;  
fl.getDocumentDOM().getTimeline().insertKeyframe(9);
```



## timeline.layerCount

### Disponibilité

Flash MX 2004.

### Utilisation

`timeline.layerCount`

### Description

Propriété en lecture seule ; entier indiquant le nombre de calques du scénario spécifié.

### Exemple

L'exemple suivant enregistre dans la variable `NumLayer` le nombre de calques de la séquence active :

```
var NumLayer = fl.getDocumentDOM().getTimeline().layerCount;
```

## timeline.layers

### Disponibilité

Flash MX 2004.

### Utilisation

`timeline.layers`

### Description

Propriété en lecture seule ; tableau d'objets `layer`.

### Exemple

L'exemple suivant enregistre dans la variable `currentLayers` le tableau des objets `layer` du document actif :

```
var currentLayers = fl.getDocumentDOM().getTimeline().layers;
```

## timeline.libraryItem

### Disponibilité

Flash Professional CS5

### Utilisation

`timeline.libraryItem`

### Description

Propriété en lecture seule ; si la propriété `libraryItem` du scénario est définie sur `null`, le scénario appartient à une séquence. Si elle n'est pas définie sur `null`, vous pouvez la considérer comme un objet `LibraryItem`.

### Exemple

L'exemple suivant indique le nom de l'objet `libraryItem` si la valeur de `libraryItem` n'est pas nulle, et le nom de la séquence si la valeur de `libraryItem` est nulle :

```
var item = fl.getDocumentDOM().getTimeline().libraryItem;
if (item)
    fl.trace("libraryItem name: " + item.name);
else
    fl.trace("scene name: " + fl.getDocumentDOM().getTimeline().name);
```

## timeline.name

### Disponibilité

Flash MX 2004.

### Utilisation

`timeline.name`

### Description

Propriété ; chaîne qui spécifie le nom du scénario actif. Ce nom est celui de la séquence, de l'écran (diapositive ou formulaire) ou du symbole qui est en cours de modification.

### Exemple

L'exemple suivant récupère le nom de la séquence active :

```
var sceneName = fl.getDocumentDOM().timelines[0].name;
```

L'exemple suivant donne la valeur `FirstScene` au nom de la première séquence :

```
fl.getDocumentDOM().timelines[0].name = "FirstScene";
```

## timeline.pasteFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

`timeline.pasteFrames([startFrameIndex [, endFrameIndex]])`

### Paramètres

**startFrameIndex** Index de base zéro qui spécifie le début de la plage d'images à coller. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image au niveau de laquelle les images doivent être collées. Cette plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui colle la plage d'images du Presse-papiers dans les images spécifiées.

### Exemple

L'exemple suivant colle la plage d'images du Presse-papiers dans l'image sélectionnée ou, à défaut, à l'emplacement de la tête de lecture :

```
fl.getDocumentDOM().getTimeline().pasteFrames();
```

L'exemple suivant colle les images dans le Presse-papiers à partir de l'image 2 et jusqu'à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().pasteFrames(1, 9);
```

L'exemple suivant colle les images dans le Presse-papiers à partir de l'image 5 :

```
fl.getDocumentDOM().getTimeline().pasteFrames(4);
```

## timeline.pasteLayers()

### Disponibilité

Flash CS5.5 Professional.

### Utilisation

```
timeline.pasteLayers([layerIndex])
```

### Paramètres

**layerIndex** Facultatif. Index de base zéro qui spécifie le calque au-dessus duquel les calques du Presse-papiers sont collés. Si le paramètre layerIndex est omis, la méthode utilise la sélection actuelle.

### Valeur renvoyée

Entier indiquant l'index du calque le plus bas parmi les calques ayant été collés.

### Description

Méthode : colle les calques préalablement coupés ou copiés au-dessus du calque actuellement sélectionné ou au-dessus de l'index du calque spécifié. Si l'index spécifié est un dossier de calques, les calques sont collés dans le dossier. Renvoie l'index du calque le plus bas parmi les calques ayant été collés. Cette action n'a aucune incidence sur le Presse-papiers système.

### Exemple

L'exemple suivant colle les calques du Presse-papiers de calques au-dessus du calque actuellement sélectionné dans le scénario :

```
fl.getDocumentDOM().getTimeline().pasteLayers();
```

L'exemple suivant colle les calques du Presse-papiers de calques au-dessus de l'index du calque 2 :

```
fl.getDocumentDOM().getTimeline().pasteLayers(2);
```

**Voir aussi**

`timeline.cutLayers()`, `timeline.copyLayers()`, `timeline.duplicateLayers()`

## timeline.pasteMotion()

**Disponibilité**

Flash CS3 Professional.

**Utilisation**

`timeline.pasteMotion()`

**Paramètres**

Aucun.

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui copie dans le scénario la plage d'images de mouvement récupérée par `timeline.copyMotion()`. Au besoin, les images existantes sont déplacées (repositionnées à droite) pour faire de la place aux images qui sont collées.

**Exemple**

L'exemple suivant colle le mouvement du Presse-papiers dans l'image sélectionnée ou à l'emplacement de la tête de lecture en déplaçant cette image à droite des images collées :

```
fl.getDocumentDOM().getTimeline().pasteMotion();
```

**Voir aussi**

`timeline.copyMotion()`

## timeline.removeFrames()

**Disponibilité**

Flash MX 2004.

**Utilisation**

`timeline.removeFrames([startFrameIndex [,endFrameIndex]])`

**Paramètres**

**startFrameIndex** Index de base zéro qui spécifie la première image à partir de laquelle les images doivent être supprimées. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. S'il n'existe pas de sélection, toutes les images (de tous les calques) qui se trouvent au niveau de la tête de lecture sont supprimées. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image au niveau de laquelle les images doivent être supprimées ; la plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui supprime l'image.

**Exemple**

L'exemple suivant supprime les images du calque supérieur de la séquence active à partir de l'image 5 et jusqu'à l'image 10 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().removeFrames(4, 9);
```

L'exemple suivant supprime l'image 8 du calque supérieur de la séquence active :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().removeFrames(7);
```

## timeline.removeMotionObject()

**Disponibilité**

Flash Professional CS5

**Utilisation**

```
timeline.removeMotionObject([startFrame [,endFrame]])
```

**Paramètres**

**startFrame** Spécifie la première image à partir de laquelle vous souhaitez commencer la suppression des objets de mouvement. En cas d'omission du paramètre *startFrame*, la méthode utilise la sélection actuelle. S'il n'existe pas de sélection, toutes les images (de tous les calques) qui se trouvent au niveau de la tête de lecture sont supprimées. Ce paramètre est facultatif.

**endFrame** Spécifie l'image à partir de laquelle vous souhaitez cesser la suppression des objets de mouvement ; la plage d'images s'étend jusqu'à *endFrame* (non compris). Si seul le paramètre *startFrame* est spécifié, la valeur de *endFrame* prend par défaut la valeur de *startFrame*. Ce paramètre est facultatif.

**Valeur renvoyée**

Aucune.

**Description**

Méthode ; supprime l'objet de mouvement et reconvertit la ou les images en images statiques. Ces paramètres sont facultatifs. S'ils sont spécifiés, la sélection du scénario est définie sur les images indiquées avant la suppression de l'objet de mouvement.

### Exemple

L'exemple suivant supprime tous les objets de mouvement et reconvertit les images en images statiques à la position actuelle de la tête de lecture sur le calque supérieur :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().removeMotionObject();
```

L'exemple suivant supprime les objets de mouvement à partir de l'image 5 jusqu'à l'image 15 (non comprise) du calque supérieur dans la séquence actuelle :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().removeMotionObject(5, 15);
```

### Voir aussi

« [timeline.createMotionObject\(\)](#) » à la page 478

## timeline.reorderLayer()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.reorderLayer(layerToMove, layerToPutItBy [, bAddBefore])
```

### Paramètres

**layerToMove** Index de base zéro spécifiant le calque à déplacer.

**layerToPutItBy** Index de base zéro indiquant le calque à côté duquel l'autre calque doit être placé. Par exemple, si vous spécifiez 1 pour *layerToMove* et 0 pour *layerToPutItBy*, le second calque est placé à côté du premier.

**bAddBefore** Indique si le calque doit être placé avant ou après *layerToPutItBy*. Si sa valeur est *false*, le calque est placé après *layerToPutItBy*. La valeur par défaut est *true*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui déplace le premier calque spécifié avant ou après le second calque spécifié.

### Exemple

L'exemple suivant place le calque ayant l'index 2 au niveau supérieur (au-dessus du calque ayant l'index 0) :

```
fl.getDocumentDOM().getTimeline().reorderLayer(2, 0);
```

L'exemple suivant place le calque ayant l'index 3 après le calque doté de l'index 5 :

```
fl.getDocumentDOM().getTimeline().reorderLayer(3, 5, false);
```

## timeline.reverseFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.reverseFrames([startFrameIndex [, endFrameIndex]])
```

### Paramètres

**startFrameIndex** Index de base zéro qui spécifie la première image à partir de laquelle les images doivent être inversées. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image au niveau de laquelle les images doivent être inversées ; la plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui inverse une plage d'images.

### Exemple

L'exemple suivant inverse les positions des calques actuellement sélectionnés :

```
fl.getDocumentDOM().getTimeline().reverseFrames();
```

L'exemple suivant inverse les images à partir de l'image 10 et jusqu'à l'image 15 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().reverseFrames(9, 14);
```

## timeline.selectAllFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.selectAllFrames()
```

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui sélectionne toutes les images du scénario actif.

### Exemple

L'exemple suivant sélectionne toutes les images du scénario actif.

```
fl.getDocumentDOM().getTimeline().selectAllFrames();
```

## timeline.setFrameProperty()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.setFrameProperty(property, value [, startFrameIndex [, endFrameIndex]])
```

### Paramètres

**property** Chaîne spécifiant le nom de la propriété à modifier. La liste complète des propriétés et des valeurs est indiquée dans le Résumé des propriétés de l'Objet Frame.

Cette méthode ne permet pas de définir les propriétés en lecture seule, telles que `frame.duration` et `frame.elements`.

**value** Valeur à affecter à la propriété. Pour déterminer le type et les valeurs appropriées, voir le Résumé des propriétés de l'Objet Frame.

**startFrameIndex** Index de base zéro qui spécifie le numéro de l'image de début à modifier. Si le paramètre *startFrameIndex* est omis, la méthode utilise la sélection actuelle. Ce paramètre est facultatif.

**endFrameIndex** Index de base zéro qui spécifie la dernière image à modifier. La plage s'étend jusqu'à *endFrameIndex* non compris. Si seul le paramètre *startFrameIndex* est transmis et que *endFrameIndex* est omis, la valeur de *endFrameIndex* prend par défaut la valeur de *startFrameIndex*. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la propriété de l'objet Frame pour les images sélectionnées.

### Exemple

L'exemple suivant affecte la commande ActionScript `stop()` à la première image du calque supérieur du document actif :

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().setSelectedFrames(0,0,true);  
fl.getDocumentDOM().getTimeline().setFrameProperty("actionScript", "stop();");
```

L'exemple suivant définit une interpolation de mouvement des images comprises entre l'image 2 et l'image 5 non comprise (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
var doc = fl.getDocumentDOM();  
doc.getTimeline().setFrameProperty("tweenType", "motion", 1, 4);
```



## timeline.setGuidelines()

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
timeline.setGuidelines(xmlString)
```

### Paramètres

**xmlString** Chaîne XML contenant des informations sur les guides à appliquer.

### Valeur renvoyée

Valeur booléenne, `true` si les guides ont bien été appliqués, `false` dans le cas contraire.

### Description

Méthode qui remplace les guides du scénario (Affichage > Guides > Afficher les guides) par les informations spécifiées dans *xmlString*. Pour récupérer une chaîne XML pouvant être transmise à cette méthode, utilisez [timeline.getGuidelines\(\)](#).

Pour afficher les guides nouvellement définis, vous devrez peut-être les masquer, puis les réafficher.

### Exemple

L'exemple suivant applique les guides d'un fichier FLA à un autre fichier FLA :

```
var doc0 = fl.documents[0];  
var guides0 = doc0.timelines[0].getGuidelines();  
var doc1 = fl.documents[1];  
doc1.timelines[0].setGuidelines(guides0);
```

## timeline.setLayerProperty()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.setLayerProperty(property, value [, layersToChange])
```

### Paramètres

**property** Chaîne spécifiant la propriété à définir. Pour obtenir la liste des propriétés, voir l'« [Objet Layer](#) » à la page 323.

**value** Valeur à affecter à la propriété. Faites appel au type de valeur utilisé pour définir la propriété dans l'objet Layer.

**layersToChange** Chaîne identifiant les calques à modifier. Les valeurs acceptables sont "selected" (sélectionné), "all" (tous) et "others" (autres). Si vous omettez ce paramètre, la valeur par défaut est "selected". Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui donne une valeur spécifiée à la propriété spécifiée pour tous les calques sélectionnés.

### Exemple

L'exemple suivant rend invisibles le ou les calques sélectionnés :

```
fl.getDocumentDOM().getTimeline().setLayerProperty("visible", false);
```

L'exemple suivant définit sur `selLayer` le nom du ou des calques sélectionnés :

```
fl.getDocumentDOM().getTimeline().setLayerProperty("name", "selLayer");
```

## timeline.setSelectedFrames()

### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.setSelectedFrames(startFrameIndex, endFrameIndex [, bReplaceCurrentSelection])  
timeline.setSelectedFrames(selectionList [, bReplaceCurrentSelection])
```

### Paramètres

**startFrameIndex** Index de base zéro qui spécifie la première image à définir.

**endFrameIndex** Index de base zéro qui spécifie la fin de la sélection ; *endFrameIndex* est l'image qui suit la dernière image de la plage à sélectionner.

**bReplaceCurrentSelection** Valeur booléenne ; si elle est `true`, les images actuellement sélectionnées sont désélectionnées, puis les images spécifiées sont sélectionnées. La valeur par défaut est `true`.

**selectionList** Tableau de trois entiers renvoyés par `timeline.getSelectedFrames()`.

### Valeur renvoyée

Aucune.

### Description

Méthode qui sélectionne une plage d'images dans le calque actif, ou applique les images sélectionnées au tableau de sélection transmis à cette méthode.

### Exemple

Les exemples suivants montrent deux manières de sélectionner dans le calque supérieur la plage d'images allant de l'image 1 à l'image 10 (non comprise) ; puis la plage de l'image 12 à l'image 15 (non comprise) du même calque pour l'ajouter à la sélection en cours (n'oubliez pas que les valeurs d'index diffèrent des valeurs de numéro d'image) :

```
fl.getDocumentDOM().getTimeline().setSelectedFrames(0, 9);  
fl.getDocumentDOM().getTimeline().setSelectedFrames(11, 14, false);  
fl.getDocumentDOM().getTimeline().setSelectedFrames([0, 0, 9]);  
fl.getDocumentDOM().getTimeline().setSelectedFrames([0, 11, 14], false);
```

L'exemple suivant stocke dans la variable `savedSelectionList` le tableau des images sélectionnées, puis utilise ce tableau dans le code pour sélectionner à nouveau ces images après modification de la sélection par le biais d'une commande ou d'une interaction de l'utilisateur :

```
var savedSelectionList = fl.getDocumentDOM().getTimeline().getSelectedFrames();  
// Do something that changes the selection.  
fl.getDocumentDOM().getTimeline().setSelectedFrames(savedSelectionList);
```

#### Voir aussi

[timeline.getSelectedFrames\(\)](#)

## timeline.setSelectedLayers()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
timeline.setSelectedLayers(index [, bReplaceCurrentSelection])
```

#### Paramètres

**index** Index de base zéro indiquant le calque à sélectionner.

**bReplaceCurrentSelection** Valeur booléenne. Si elle est `true`, la méthode remplace la sélection actuelle ; si elle est `false`, la méthode étend la sélection. La valeur par défaut est `true`. Ce paramètre est facultatif.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui définit le calque à sélectionner, et fait du calque spécifié le calque actif. Lorsqu'un calque est sélectionné, toutes ses images sont sélectionnées.

#### Exemple

L'exemple suivant sélectionne le calque supérieur :

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(0);
```

L'exemple suivant ajoute le prochain calque à la sélection :

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(1, false);
```

#### Voir aussi

[timeline.getSelectedLayers\(\)](#)

## timeline.showLayerMasking()

#### Disponibilité

Flash MX 2004.

### Utilisation

```
timeline.showLayerMasking([layer])
```

### Paramètres

**layer** Index de base zéro d'un calque de masque ou d'un calque masqué à afficher durant la programmation. Ce paramètre est facultatif.

### Valeur renvoyée

Aucune.

### Description

Méthode qui affiche les masques de calque pendant la programmation, en verrouillant le calque de masque et les calques masqués. Si aucun calque n'est spécifié, la méthode s'applique au calque actif. Si cette méthode est utilisée sur un calque qui n'est pas de type Masque ou Masqué, Flash affiche une erreur dans le panneau Sortie.

### Exemple

L'exemple suivant spécifie que le masquage du premier calque doit apparaître pendant la programmation.

```
fl.getDocumentDOM().getTimeline().showLayerMasking(0);
```

## timeline.startPlayback()

### Disponibilité

Flash Professional CS5

### Utilisation

```
timeline.startPlayback()
```

### Valeur renvoyée

Aucune.

### Description

Méthode ; démarre la lecture automatique du scénario si ce dernier est en cours de lecture. Il est possible d'utiliser cette méthode avec les panneaux SWF en vue de gérer la lecture du scénario dans l'environnement de création.

### Exemple

L'exemple suivant démarre la lecture du scénario.

```
fl.getDocumentDOM().getTimeline().startPlayback();
```

## timeline.stopPlayback()

### Disponibilité

Flash Professional CS5

### **Utilisation**

```
timeline.stopPlayback()
```

### **Valeur renvoyée**

Aucune.

### **Description**

Méthode ; arrête la lecture automatique du scénario si ce dernier est en cours de lecture. Il est possible d'utiliser cette méthode avec les panneaux SWF en vue de gérer la lecture du scénario dans l'environnement de création.

### **Exemple**

L'exemple suivant arrête la lecture du scénario.

```
fl.getDocumentDOM().getTimeline().stopPlayback();
```

# Chapitre 46 : Objet ToolObj

## Disponibilité

Flash MX 2004.

## Description

Un objet `toolObj` représente un outil individuel dans le panneau Outils. Pour accéder à un objet `ToolObj`, utilisez les propriétés de l'Objet Tools : le tableau `tools.toolObjs` ou `tools.activeTool`.

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet `ToolObj` :

*Remarque : ces méthodes ne sont utilisées que lors de la création d'outils extensibles.*

Méthode	Description
<code>toolObj.enablePIControl()</code>	Active ou désactive le contrôle spécifié dans un inspecteur Propriétés. Cette méthode n'est utilisée que lors de la création d'outils extensibles.
<code>toolObj.setIcon()</code>	Identifie un fichier PNG à utiliser comme icône d'outil dans le panneau Outils de Flash.
<code>toolObj.setMenuString()</code>	Définit la chaîne qui doit apparaître dans le menu contextuel pour identifier l'outil.
<code>toolObj.setOptionsFile()</code>	Associe un fichier XML à l'outil.
<code>toolObj.setPI()</code>	Définit un inspecteur Propriétés spécifique à utiliser lorsque l'outil est activé.
<code>toolObj.setToolName()</code>	Affecte un nom à l'outil pour la configuration du panneau Outils.
<code>toolObj.setToolTip()</code>	Définit l'info-bulle qui doit apparaître lorsque la souris est amenée sur l'icône de l'outil.
<code>toolObj.showPIControl()</code>	Affiche ou masque un contrôle dans l'inspecteur Propriétés.
<code>toolObj.showTransformHandles()</code>	Appelée dans la méthode <code>configureTool()</code> du fichier JavaScript de l'outil extensible pour indiquer que les poignées de transformation libre doivent apparaître lorsque l'outil est actif.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet `ToolObj` :

Propriété	Description
<code>toolObj.depth</code>	Entier qui définit la profondeur de l'outil dans le menu contextuel du panneau Outils.
<code>toolObj.iconID</code>	Entier qui définit l'ID de ressource de l'outil.
<code>toolObj.position</code>	Lecture seule ; entier indiquant la position de l'outil dans le panneau Outils.

## toolObj.depth

### Disponibilité

Flash MX 2004.

### Utilisation

`toolObj.depth`

### Description

Propriété en lecture seule ; entier qui définit la profondeur de l'outil dans le menu contextuel du panneau Outils. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant spécifie que l'outil possède une profondeur de 1, ce qui signifie qu'il se trouve un niveau en dessous d'un outil dans le panneau Outils :

```
fl.tools.activeTool.depth = 1;
```

## toolObj.enablePIControl()

### Disponibilité

Flash MX 2004.

### Utilisation

`toolObj.enablePIControl(control, bEnable)`

### Paramètres

**control** Chaîne spécifiant le nom du contrôle à activer ou désactiver. Les valeurs autorisées dépendent de l'inspecteur Propriétés appelé par cet outil ; voir [toolObj.setPI\(\)](#).

L'inspecteur Propriétés d'une forme possède les contrôles suivants :

stroke	fill
--------	------

L'inspecteur Propriétés d'un texte possède les contrôles suivants :

type	font	pointsize
color	bold	italic
direction	alignLeft	alignCenter
alignRight	alignJustify	spacing
position	autoKern	small
rotation	format	lineType

selectable	html	border
deviceFonts	varEdit	options
link	maxChars	target

L'inspecteur Propriétés d'un film possède les contrôles suivants :

size	publish	background
framerate	player	profile

**enable** Valeur booléenne indiquant si le contrôle doit être activé (`true`) ou non (`false`).

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui active ou désactive le contrôle spécifié dans un inspecteur Propriétés. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

#### Exemple

Si la commande suivante est insérée dans le fichier JavaScript d'un outil extensible, Flash n'affiche pas les options de trait dans l'inspecteur Propriétés de cet outil :

```
theTool.enablePIControl("stroke", false);
```

## toolObj.iconID

#### Disponibilité

Flash MX 2004.

#### Utilisation

`toolObj.iconID`

#### Description

Propriété en lecture seule ; entier de valeur -1. Cette propriété s'utilise uniquement lors de la création d'outils extensibles. Une valeur `iconID` de -1 signifie que Flash ne tentera pas de trouver une icône pour l'outil. Au contraire, le script de l'outil doit définir l'icône à afficher dans le panneau Outils ; voir [toolObj.setIcon\(\)](#).

#### Exemple

L'exemple suivant attribue une valeur de -1 (l'ID d'icône de l'outil actuel) à la variable `toolIconID` :

```
var toolIconID = fl.tools.activeTool.iconID
```



## toolObj.position

### Disponibilité

Flash MX 2004.

### Utilisation

`toolObj.position`

### Description

Propriété en lecture seule ; entier indiquant la position de l'outil dans le panneau Outils. Cette propriété n'est utilisée que lors de la création d'outils extensibles.

### Exemple

Si les commandes suivantes sont insérées dans la méthode `mouseDown()` du fichier JavaScript d'un outil extensible, la position de ce dernier dans le panneau Outils sera affichée sous forme d'entier dans le panneau Sortie :

```
myToolPos = fl.tools.activeTool.position;  
fl.trace(myToolPos);
```

## toolObj.setIcon()

### Disponibilité

Flash MX 2004.

### Utilisation

`toolObj.setIcon(file)`

### Paramètres

**file** Chaîne spécifiant le nom du fichier PNG à utiliser comme icône. Le fichier PNG doit être placé dans le même dossier que le fichier JSFL.

### Valeur renvoyée

Aucune.

### Description

Méthode qui identifie un fichier PNG à utiliser comme icône d'outil dans le panneau Outils. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant indique que l'image enregistrée dans le fichier « PolyStar.png » doit être utilisée comme icône de l'outil PolyStar. Ce code est extrait du fichier d'exemple PolyStar.jsfl (voir « [Exemple d'outil PolyStar](#) » à la page 14) :

```
theTool = fl.tools.activeTool;  
theTool.setIcon("PolyStar.png");
```

## toolObj.setMenuString()

### Disponibilité

Flash MX 2004.

### Utilisation

```
toolObj.setMenuString(menuStr)
```

### Paramètres

**menuStr** Chaîne spécifiant le nom qui doit apparaître dans le menu contextuel pour identifier l'outil.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit la chaîne qui devant apparaître dans le menu contextuel pour identifier l'outil. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant indique que l'outil `theTool` doit afficher le nom « PolyStarTool » dans son menu contextuel. Ce code est extrait du fichier d'exemple `PolyStar.jsfl` (voir « [Exemple d'outil PolyStar](#) » à la page 14) :

```
theTool = fl.tools.activeTool;  
theTool.setMenuString("PolyStar Tool");
```

## toolObj.setOptionsFile()

### Disponibilité

Flash MX 2004.

### Utilisation

```
toolObj.setOptionsFile(xmlFile)
```

### Paramètres

**xmlFile** Chaîne spécifiant le nom du fichier XML qui contient la description des options de l'outil. Le fichier XML doit être placé dans le même dossier que le fichier JSFL.

### Valeur renvoyée

Aucune.

### Description

Méthode qui associe un fichier XML à l'outil. Le fichier spécifie les options qui s'afficheront dans un panneau modal appelé par un bouton Options dans l'inspecteur Propriétés. Vous utiliserez en caractère à récupérer, générale cette méthode dans la fonction `configureTool()` intégrée à votre fichier JSFL. Voir [configureTool\(\)](#).

Le fichier `PolyStar.xml` spécifie trois options associées à l'outil Polygon :

```
<properties>
  <property name="Style"
    variable="style"
    list="polygon,star"
    defaultValue="0"
    type="Strings"/>

  <property name="Number of Sides"
    variable="nsides"
    min="3"
    max="32"
    defaultValue="5"
    type="Number" />

  <property name="Star point size"
    variable="pointParam"
    min="0"
    max="1"
    defaultValue=".5"
    type="Double" />

</properties>
```

### Exemple

L'exemple suivant spécifie que le fichier « PolyStar.xml » est associé à l'outil actif. Ce code est extrait du fichier d'exemple PolyStar.jsfl (voir « [Exemple d'outil PolyStar](#) » à la page 14) :

```
theTool = fl.tools.activeTool;
theTool.setOptionsFile("PolyStar.xml");
```

## toolObj.setPI()

### Disponibilité

Flash MX 2004.

### Utilisation

```
toolObj.setPI(pi)
```

### Paramètres

**pi** Chaîne spécifiant l'inspecteur Propriétés à appeler pour cet outil.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit un inspecteur Propriétés spécifique à utiliser lorsque l'outil est activé. Cette méthode n'est utilisée que lors de la création d'outils extensibles. Les valeurs gérées sont "shape" (forme, valeur par défaut), "text" et "movie" (film).

### Exemple

L'exemple suivant spécifie que l'inspecteur Propriétés des formes doit être utilisé lorsque l'outil est activé. Ce code est extrait du fichier d'exemple PolyStar.jsfl (voir « [Exemple d'outil PolyStar](#) » à la page 14) :

```
theTool = fl.tools.activeTool;  
theTool.setPI("shape");
```

## toolObj.setToolName()

### Disponibilité

Flash MX 2004.

### Utilisation

```
toolObj.setToolName(name)
```

### Paramètres

**name** Chaîne spécifiant le nom de l'outil.

### Valeur renvoyée

Aucune.

### Description

Méthode qui affecte un nom à l'outil pour la configuration du panneau Outils. Cette méthode n'est utilisée que lors de la création d'outils extensibles. Ce nom est uniquement utilisé par le fichier XML de disposition que Flash lit pour construire le panneau Outils. Il n'apparaît donc pas dans l'interface utilisateur de Flash.

### Exemple

L'exemple suivant affecte le nom polystar à l'outil nommé theTool. Ce code est extrait du fichier d'exemple PolyStar.jsfl (voir « [Exemple d'outil PolyStar](#) » à la page 14) :

```
theTool = fl.tools.activeTool;  
theTool.setToolName("polystar");
```

## toolObj.setToolTip()

### Disponibilité

Flash MX 2004.

### Utilisation

```
toolObj.setToolTip(toolTip)
```

### Paramètres

**toolTip** Chaîne spécifiant l'info-bulle à utiliser pour cet outil.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit l'info-bulle qui doit apparaître lorsque la souris est amenée sur l'icône de l'outil. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

### Exemple

L'exemple suivant spécifie que l'info-bulle de l'outil doit être `PolyStarTool`. Ce code est extrait du fichier d'exemple `PolyStar.jsfl` (voir « [Exemple d'outil PolyStar](#) » à la page 14) :

```
theTool = fl.tools.activeTool;  
theTool.setToolTip("PolyStar Tool");
```

## toolObj.showPIControl()

### Disponibilité

Flash MX 2004.

### Utilisation

```
toolObj.showPIControl(control, bShow)
```

### Paramètres

**control** Chaîne spécifiant le nom du contrôle à afficher ou masquer. Cette méthode n'est utilisée que lors de la création d'outils extensibles. Les valeurs autorisées dépendent de l'inspecteur des propriétés appelé par cet outil (voir `toolObj.setPI()`).

L'inspecteur Propriétés d'une forme possède les contrôles suivants :

stroke	fill
--------	------

L'inspecteur Propriétés d'un texte possède les contrôles suivants :

type	font	pointsize
color	bold	italic
direction	alignLeft	alignCenter
alignRight	alignJustify	spacing
position	autoKern	small
rotation	format	lineType
selectable	html	border
deviceFonts	varEdit	options
link	maxChars	target

L'inspecteur Propriétés d'un film possède les contrôles suivants :

size	publish	background
framerate	player	profile

**bShow** Valeur booléenne indiquant si le contrôle spécifié doit être affiché (`true`) ou masqué (`false`).

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui affiche ou masque un contrôle dans l'inspecteur Propriétés. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

**Exemple**

Si la commande suivante est insérée dans le fichier JavaScript d'un outil extensible, Flash n'affiche pas les options de remplissage dans l'inspecteur Propriétés de cet outil :

```
fl.tools.activeTool.showPControl("fill", false);
```

## toolObj.showTransformHandles()

**Disponibilité**

Flash MX 2004.

**Utilisation**

```
toolObj.showTransformHandles(bShow)
```

**Paramètres**

**bShow** Valeur booléenne indiquant si les poignées de transformation libre de l'outil actif doivent être affichées (`true`) ou masquées (`false`).

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui est appelée par la méthode `configureTool()` du fichier JavaScript d'un outil extensible pour indiquer que les poignées de transformation libre doivent apparaître lorsque l'outil est actif. Cette méthode n'est utilisée que lors de la création d'outils extensibles.

**Exemple**

Voir `configureTool()`.

# Chapitre 47 : Objet Tools

## Disponibilité

Flash MX 2004.

## Description

L'objet Tools est accessible à partir de l'objet flash (`fl.tools`). La propriété `tools.toolObjs` contient un tableau d'objets ToolObj, et la propriété `tools.activeTool` renvoie l'objet ToolObj pour l'outil actif. (Voir aussi [Objet ToolObj](#) et la liste d'outils extensibles dans « [Fonctions et méthodes de niveau supérieur](#) » à la page 15.)

*Remarque : les méthodes et propriétés suivantes sont uniquement utilisées lors de la création d'outils extensibles.*

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet Tools :

Méthode	Description
<code>tools.constrainPoint()</code>	Récupère deux points et renvoie un nouveau point ajusté ou <i>imposé</i> .
<code>tools.getKeyDown()</code>	Renvoie la touche la plus récemment appuyée.
« <code>tools.setCreatingBbox()</code> » à la page 517	PLACEHOLDER
<code>tools.setCursor()</code>	Définit l'apparence du pointeur.
<code>tools.snapPoint()</code>	Reçoit un point et en renvoie un nouveau qui peut être ajusté ou <i>accroché</i> à l'objet géométrique le plus proche.

## Résumé des propriétés

Les propriétés suivantes sont disponibles avec l'objet Tools :

Propriété	Description
<code>tools.activeTool</code>	Lecture seule ; renvoie l' <a href="#">Objet ToolObj</a> pour l'outil actif.
<code>tools.altIsDown</code>	Lecture seule ; valeur booléenne qui indique si la touche Alt est enfoncée.
<code>tools.ctrlIsDown</code>	Lecture seule ; valeur booléenne qui indique si la touche Ctrl est enfoncée.
<code>tools.mouseIsDown</code>	Lecture seule ; valeur booléenne qui indique si le bouton gauche de la souris est enfoncé.
<code>tools.penDownLoc</code>	Lecture seule ; point qui représente la position du dernier événement de bouton de souris enfoncé sur la scène.
<code>tools.penLoc</code>	Lecture seule ; point représentant l'emplacement actuel de la souris.
<code>tools.shiftIsDown</code>	Lecture seule ; valeur booléenne qui indique si la touche Maj est actuellement enfoncée.
<code>tools.toolObjs</code>	Lecture seule ; tableau d'objets ToolObj.

## tools.activeTool

### Disponibilité

Flash MX 2004.

### Utilisation

`tools.activeTool`

### Description

Propriété en lecture seule ; renvoie l'Objet `ToolObj` pour l'outil actif.

### Exemple

L'exemple suivant enregistre un objet qui représente l'outil actif dans la variable `theTool` :

```
var theTool = fl.tools.activeTool;
```

## tools.altIsDown

### Disponibilité

Flash MX 2004.

### Utilisation

`tools.altIsDown`

### Description

Propriété en lecture seule ; valeur booléenne qui indique si la touche Alt est actuellement enfoncée. Sa valeur est `true` si la touche Alt est enfoncée, ou `false` dans le cas contraire.

### Exemple

L'exemple suivant détermine si la touche Alt est enfoncée :

```
var isAltDown = fl.tools.altIsDown;
```

## tools.constrainPoint()

### Disponibilité

Flash MX 2004.

### Utilisation

`tools.constrainPoint(pt1, pt2)`

### Paramètres

**pt1**, **pt2** Points indiquant respectivement le point auquel le clic a eu lieu et le point jusqu'auquel le bouton de la souris est resté enfoncé.



### Valeur renvoyée

Nouveau point ajusté ou imposé.

### Description

Méthode qui reçoit deux points et renvoie un nouveau point ajusté ou imposé. Si la touche Maj est enfoncée lors de l'exécution de la commande, le point renvoyé est imposé, soit pour contraindre un angle à 45° (par exemple pour une ligne terminée par une flèche), soit pour obliger un objet à conserver son format (par exemple pour tracer un carré parfait à l'aide de l'outil Rectangle).

### Exemple

L'exemple suivant renvoie un point imposé :

```
pt2 = fl.tools.constrainPoint(pt1, tempPt);
```

## tools.ctllsDown

### Disponibilité

Flash MX 2004.

### Utilisation

```
tools.ctllsDown
```

### Description

Propriété en lecture seule ; valeur booléenne définie sur `true` si la touche Ctrl est enfoncée, `false` dans le cas contraire.

### Exemple

L'exemple suivant détermine si la touche Ctrl est enfoncée :

```
var isCtrlDown = fl.tools.ctrlIsDown;
```

## tools.getKeyDown()

### Disponibilité

Flash MX 2004.

### Utilisation

```
tools.getKeyDown()
```

### Paramètres

Aucun.

### Valeur renvoyée

La valeur entière de la touche.

### Description

Méthode qui renvoie la touche la plus récemment appuyée.

### Exemple

L'exemple suivant affiche la valeur d'entier de la touche la plus récemment enfoncée :

```
var theKey = fl.tools.getKeyDown();  
fl.trace(theKey);
```

## tools.mouselsDown

### Disponibilité

Flash MX 2004.

### Utilisation

`tools.mouseIsDown`

### Description

Propriété en lecture seule ; valeur booléenne définie sur `true` si le bouton gauche de la souris est actuellement enfoncé, `false` dans le cas contraire.

### Exemple

L'exemple suivant détermine si le bouton gauche de la souris est enfoncé.

```
var isMouseDown = fl.tools.mouseIsDown;
```

## tools.penDownLoc

### Disponibilité

Flash MX 2004.

### Utilisation

`tools.penDownLoc`

### Description

Propriété en lecture seule ; point qui représente la position du dernier événement de bouton de souris enfoncé sur la scène. La propriété `tools.penDownLoc` comprend deux propriétés, `x` et `y`, qui correspondent à la position `x, y` du pointeur de la souris.

### Exemple

L'exemple suivant détermine la position du dernier événement de bouton de souris enfoncé sur la scène et affiche les valeurs `x` et `y` dans le panneau Sortie :

```
var pt1 = fl.tools.penDownLoc;  
fl.trace("x,y location of last mouseDown event was " + pt1.x + ", " + pt1.y)
```

**Voir aussi**

[tools.penLoc](#)

## tools.penLoc

**Disponibilité**

Flash MX 2004.

**Utilisation**

`tools.penLoc`

**Description**

Propriété en lecture seule ; point représentant l'emplacement actuel du pointeur de la souris. La propriété `tools.penLoc` comprend deux propriétés, *x* et *y*, qui correspondent à la position *x,y* du pointeur de la souris.

**Exemple**

L'exemple suivant détermine l'emplacement actuel de la souris :

```
var tempPt = fl.tools.penLoc;
```

**Voir aussi**

[tools.penDownLoc](#)

## tools.setCreatingBbox()

**Disponibilité**

Flash 11.

**Utilisation**

`tools.setCreatingBbox()`

**Paramètres**

**Espace réservé** Entier définissant l'apparence du pointeur, comme décrit dans la liste suivante :

**Valeur renvoyée**

Aucune.

**Description**

Méthode qui donne l'apparence spécifiée au pointeur.

**Exemple**

L'exemple suivant donne l'aspect d'une flèche noire au pointeur.

```
fl.tools.setCursor(1);
```

## tools.setCursor()

### Disponibilité

Flash MX 2004.

### Utilisation

```
tools.setCursor(cursor)
```

### Paramètres

**cursor** Entier définissant l'apparence du pointeur, décrite dans la liste suivante :

- 0 = curseur Plus (+)
- 1 = flèche noire
- 2 = flèche blanche
- 3 = flèche à quatre directions
- 4 = flèche bidirectionnelle horizontale
- 5 = flèche bidirectionnelle verticale
- 6 = X
- 7 = curseur main

### Valeur renvoyée

Aucune.

### Description

Méthode qui donne l'apparence spécifiée au pointeur.

### Exemple

L'exemple suivant donne l'aspect d'une flèche noire au pointeur.

```
fl.tools.setCursor(1);
```

## tools.shiftIsDown

### Disponibilité

Flash MX 2004.

### Utilisation

```
tools.shiftIsDown
```

### Description

Propriété en lecture seule ; valeur booléenne définie sur `true` si la touche Maj est enfoncée, `false` dans le cas contraire.

### Exemple

L'exemple suivant détermine si la touche Maj est enfoncée.

```
var isShiftDown = fl.tools.shiftIsDown;
```

## tools.snapPoint()

### Disponibilité

Flash MX 2004.

### Utilisation

```
tools.snapPoint(pt)
```

### Paramètres

**pt** Spécifie l'emplacement du point pour lequel un point d'accrochage doit être renvoyé.

### Valeur renvoyée

Nouveau point qui peut être ajusté ou accroché à l'objet géométrique le plus proche.

### Description

Méthode qui reçoit un point et en renvoie un nouveau qui peut être ajusté ou *accroché* à l'objet géométrique le plus proche. Si l'accrochage a été désactivé dans le menu Affichage de l'interface utilisateur de Flash, le point renvoyé est le point d'origine.

### Exemple

L'exemple suivant renvoie un nouveau point qui peut être accroché à l'objet géométrique le plus proche :

```
var theSnapPoint = fl.tools.snapPoint(pt1);
```

## tools.toolObjs

### Disponibilité

Flash MX 2004.

### Utilisation

```
tools.toolObjs
```

### Description

Propriété en lecture seule ; tableau d'objets ToolObj (voir [Objet ToolObj](#)).

# Chapitre 48 : Objet Vertex

## Disponibilité

Flash MX 2004.

## Description

L'objet Vertex est la partie de la structure de données d'une forme qui contient les coordonnées.

## Résumé des méthodes

Vous pouvez utiliser les méthodes suivantes avec l'objet Vertex :

Méthode	Description
<code>vertex.getHalfEdge()</code>	Récupère un objet <a href="#">Objet halfEdge</a> qui partage ce sommet.
<code>vertex.setLocation()</code>	Définit l'emplacement du sommet.

## Résumé des propriétés

Les propriétés suivantes sont disponibles pour l'objet Vertex :

Propriété	Description
<code>vertex.x</code>	Lecture seule ; emplacement de la coordonnée x du sommet, en pixels.
<code>vertex.y</code>	Lecture seule ; emplacement de la coordonnée y du sommet, en pixels.

## vertex.getHalfEdge()

## Disponibilité

Flash MX 2004.

## Utilisation

```
vertex.getHalfEdge()
```

## Paramètres

Aucun.

## Valeur renvoyée

Objet [Objet halfEdge](#).

## Description

Méthode qui récupère un objet [Objet halfEdge](#) qui partage ce sommet.

## Exemple

L'exemple suivant montre comment obtenir les autres demi-contours qui partagent le même sommet :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var theVertex = hEdge.getVertex();  
var someHEdge = theVertex.getHalfEdge(); // Not necessarily the same half edge  
var theSameVertex = someHEdge.getVertex();  
fl.trace('the same vertex: ' + theSameVertex);
```

## vertex.setLocation()

### Disponibilité

Flash MX 2004.

### Utilisation

```
vertex.setLocation(x, y)
```

### Paramètres

**x** Valeur en virgule flottante indiquant la coordonnée *x* de la position souhaitée pour le sommet, en pixels.

**y** Valeur en virgule flottante indiquant la coordonnée *y* de la position souhaitée pour le sommet, en pixels.

### Valeur renvoyée

Aucune.

### Description

Méthode qui définit l'emplacement du sommet. Il est nécessaire d'appeler `shape.beginEdit()` avant d'utiliser cette méthode.

### Exemple

L'exemple suivant définit le sommet sur le point d'origine :

```
var shape = fl.getDocumentDOM().selection[0];  
shape.beginEdit();  
var hEdge = shape.edges[0].getHalfEdge(0);  
var vertex = hEdge.getVertex();  
var someHEdge = vertex.getHalfEdge();  
var vertex = someHEdge.getVertex();  
// Move the vertex to the origin.  
vertex.setLocation(0.0, 0.0);  
shape.endEdit();
```

## vertex.x

### Disponibilité

Flash MX 2004.

### Utilisation

```
vertex.x
```

**Description**

Propriété en lecture seule ; emplacement de la coordonnée  $x$  du sommet, en pixels.

**Exemple**

L'exemple suivant affiche dans le panneau Sortie les coordonnées  $x$  et  $y$  du sommet :

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var vertex = hEdge.getVertex();  
  
fl.trace('x location of vertex is: ' + vertex.x);  
fl.trace('y location of vertex is: ' + vertex.y);
```

## vertex.y

**Disponibilité**

Flash MX 2004.

**Utilisation**

vertex.y

**Description**

Propriété en lecture seule ; emplacement de la coordonnée  $y$  du sommet, en pixels.

**Exemple**

Voir [vertex.x](#).



# Chapitre 49 : Objet VideoItem

**Héritage**    [Objet Item](#) > Objet VideoItem

## Disponibilité

Flash MX 2004.

## Description

L'objet VideoItem est une sous-classe de l'objet [Objet Item](#).

## Résumé des méthodes

Outre les méthodes de l'objet Item, l'objet VideoItem gère la méthode suivante :

Propriété	Description
<code>videoItem.exportToFLV()</code>	Exporte l'élément spécifié dans un fichier FLV.

## Résumé des propriétés

Outre celles de l'objet Item, l'objet VideoItem gère les propriétés suivantes :

Propriété	Description
<code>videoItem.fileLastModifiedDate</code>	Lecture seule ; chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1er janvier 1970 et la date de modification du fichier original (sur disque) au moment où il a été importé dans la bibliothèque.
<code>videoItem.sourceFileExists</code>	Lecture seule ; valeur booléenne qui spécifie si le fichier importé dans la bibliothèque existe toujours à l'emplacement d'où il provient.
<code>videoItem.sourceFileIsCurrent</code>	Lecture seule ; valeur booléenne qui spécifie si la date de modification du fichier de l'élément de la bibliothèque correspond à celle trouvée sur le disque du fichier importé.
<code>videoItem.sourceFilePath</code>	Lecture seule ; chaîne représentant le chemin d'accès à l'élément vidéo.
<code>videoItem.videoType</code>	Lecture seule ; chaîne spécifiant le type de la vidéo représentée par l'élément.

## videoItem.exportToFLV()

## Disponibilité

Flash CS4 Professional.

## Utilisation

```
videoItem.exportToFLV(fileURI)
```

## Paramètres

**fileURI** Chaîne exprimée sous la forme « fichier:///URI » et spécifiant le chemin et le nom du fichier exporté.

### Valeur renvoyée

Valeur booléenne, `true` si le fichier a bien été exporté, `false` dans le cas contraire.

### Description

Méthode qui exporte l'élément spécifié dans un fichier FLV.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément vidéo, le code suivant l'exporte sous forme de fichier FLV :

```
var videoFileURL = "file:///C:/out.flv";  
var libItem = fl.getDocumentDOM().library.items[0];  
libItem.exportToFLV(videoFileURL);
```

## videoItem.fileLastModifiedDate

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
videoItem.fileLastModifiedDate
```

### Description

Propriété en lecture seule ; chaîne contenant un nombre hexadécimal qui représente le nombre de secondes écoulées entre le 1 janvier 1970 et la date de modification du fichier original (sur disque) au moment où le fichier a été importé dans la bibliothèque. Si le fichier n'existe plus, cette valeur est « 00000000 ».

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément vidéo, le code suivant affiche un nombre hexadécimal tel que décrit ci-dessus :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("Mod date when imported = " + libItem.fileLastModifiedDate);
```

### Voir aussi

[videoItem.sourceFileExists](#), [videoItem.sourceFileIsCurrent](#), [videoItem.sourceFilePath](#), [FLfile.getModificationDate\(\)](#)

## videoItem.sourceFileExists

### Disponibilité

Flash CS4 Professional.

### Utilisation

```
videoItem.sourceFileExists
```

### Description

Propriété en lecture seule : valeur booléenne `true` si le fichier importé dans la bibliothèque existe toujours à l'emplacement d'où il provient ; `false` dans le cas contraire.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément vidéo, le code suivant affiche « `true` » si le fichier importé dans la bibliothèque existe toujours :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("sourceFileExists = " + libItem.sourceFileExists);
```

### Voir aussi

[videoItem.sourceFileIsCurrent](#), [videoItem.sourceFilePath](#)

## videoItem.sourceFileIsCurrent

### Disponibilité

Flash CS4 Professional.

### Utilisation

`videoItem.sourceFileIsCurrent`

### Description

Propriété en lecture seule ; valeur booléenne, `true` si la date de modification du fichier de l'élément de la bibliothèque correspond à la date de modification sur disque du fichier importé ; `false` dans le cas contraire.

### Exemple

En supposant que le premier élément de la bibliothèque soit un élément vidéo, le code suivant affiche « `true` » si le fichier importé n'a pas été modifié sur le disque depuis son importation :

```
var libItem = fl.getDocumentDOM().library.items[0];  
fl.trace("fileIsCurrent = " + libItem.sourceFileIsCurrent);
```

### Voir aussi

[videoItem.fileLastModifiedDate](#), [videoItem.sourceFilePath](#)

## videoItem.sourceFilePath

### Disponibilité

Flash 8.

### Utilisation

`videoItem.sourceFilePath`

### Description

Propriété en lecture seule ; chaîne de type « fichier:///URI » qui représente le chemin d'accès à l'élément vidéo.

### Exemple

L'exemple suivant affiche le nom et le chemin d'accès source de tout élément de la bibliothèque de type video :

```
for (idx in fl.getDocumentDOM().library.items) {  
  if (fl.getDocumentDOM().library.items[idx].itemType == "video") {  
    var myItem = fl.getDocumentDOM().library.items[idx];  
    fl.trace(myItem.name + " source is " + myItem.sourceFilePath);  
  }  
}
```

### Voir aussi

[videoItem.sourceFileExists](#)

## videoItem.videoType

### Disponibilité

Flash 8.

### Utilisation

videoItem.videoType

### Description

Propriété en lecture seule ; chaîne spécifiant le type de vidéo représenté par l'élément. Les valeurs gérées sont "embeddedvideo", "linkedvideo" et "video".

### Exemple

L'exemple suivant affiche le nom et le type de tout élément de la bibliothèque de type video :

```
for (idx in fl.getDocumentDOM().library.items) {  
  if (fl.getDocumentDOM().library.items[idx].itemType == "video") {  
    var myItem = fl.getDocumentDOM().library.items[idx];  
    fl.trace(myItem.name + " is " + myItem.videoType);  
  }  
}
```

# Chapitre 50 : Objet XMLUI

## Disponibilité

Flash MX 2004.

## Description

Flash 8 prend en charge des boîtes de dialogue personnalisées écrites à l'aide d'un sous-ensemble du langage XUL (XML User Interface Language). Une boîte de dialogue XMLUI (XML User Interface) peut être utilisée par plusieurs fonctionnalités de Flash, telles que les commandes et les comportements, pour offrir une interface utilisateur à de nouvelles fonctionnalités créées avec l'API d'extensibilité. L'objet XMLUI permet de lire et définir les propriétés d'une boîte de dialogue XMLUI, et d'en accepter ou d'en annuler une. Les méthodes XMLUI peuvent être utilisées dans des fonctions de rappel, telles que les gestionnaires `oncommand` des boutons.

Il est possible de créer un fichier `dialog.xml` et de l'appeler à partir de l'API JavaScript avec la méthode `document.xmlPanel()`. Pour récupérer un objet représentant la boîte de dialogue XMLUI en cours, utilisez `fl.xmlui`.

## Résumé des méthodes

Les méthodes suivantes sont disponibles avec l'objet XMLUI :

Méthode	Description
<code>xmlui.accept()</code>	Ferme la boîte de dialogue XMLUI actuelle avec l'état d'acceptation.
<code>xmlui.cancel()</code>	Ferme la boîte de dialogue XMLUI active avec l'état d'annulation.
<code>xmlui.get()</code>	Récupère la valeur de la propriété spécifiée de la boîte de dialogue XMLUI actuelle.
<code>xmlui.getControlItemElement()</code>	Renvoie l'élément de contrôle actif pour le contrôle spécifié.
<code>xmlui.setEnabled()</code>	Renvoie une valeur booléenne qui spécifie si le contrôle est activé ou désactivé (affiché en grisé).
<code>xmlui.setVisible()</code>	Renvoie une valeur booléenne qui spécifie si le contrôle est visible ou masqué.
<code>xmlui.set()</code>	Modifie la valeur de la propriété spécifiée de la boîte de dialogue XMLUI actuelle.
<code>xmlui.setControlItemElement()</code>	Définit le libellé et la valeur de l'élément actif.
<code>xmlui.setControlItemElements()</code>	Définit les paires libellé et valeur de l'élément actif.
<code>xmlui.setEnabled()</code>	Active ou désactive (affiche en grisé) un contrôle.
<code>xmlui.setVisible()</code>	Affiche ou masque un contrôle.

## xmlui.accept()

## Disponibilité

Flash MX 2004.

### Utilisation

`xmlui.accept()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ferme la boîte de dialogue XMLUI avec un état d'acceptation, ce qui est équivalent à un clic sur le bouton OK.

### Voir aussi

`fl.xmlui, document.xmlPanel(), xmlui.cancel()`

## xmlui.cancel()

### Disponibilité

Flash MX 2004.

### Utilisation

`xmlui.cancel()`

### Paramètres

Aucun.

### Valeur renvoyée

Aucune.

### Description

Méthode qui ferme la boîte de dialogue XMLUI avec un état d'annulation, ce qui est équivalent à un clic sur le bouton Annuler.

### Voir aussi

`fl.xmlui, document.xmlPanel(), xmlui.accept()`

## xmlui.get()

### Disponibilité

Flash MX 2004.

### Utilisation

`xmlui.get(controlPropertyName)`

### Paramètres

**controlPropertyName** Chaîne spécifiant le nom de la propriété XMLUI dont vous recherchez la valeur.

### Valeur renvoyée

Chaîne représentant la valeur de la propriété spécifiée. Dans les cas où une valeur booléenne `true` ou `false` est attendue, cette méthode renvoie la chaîne `"true"` ou `"false"`.

### Description

Méthode qui récupère la valeur de la propriété spécifiée de la boîte de dialogue XMLUI active.

### Exemple

L'exemple suivant renvoie la valeur de la propriété nommée `URL` :

```
fl.xmlui.get("URL");
```

### Voir aussi

```
fl.xmlui, document.xmlPanel(), xmlui.getControlItemElement(), xmlui.set()
```

## xmlui.getControlItemElement()

### Disponibilité

Flash 8.

### Utilisation

```
xmlui.getControlItemElement(controlPropertyName)
```

### Paramètres

**controlPropertyName** Chaîne spécifiant la propriété dont vous souhaitez récupérer l'élément de contrôle.

### Valeur renvoyée

Objet représentant l'élément de contrôle actif pour le contrôle spécifié par *controlPropertyName*.

### Description

Méthode qui renvoie le libellé et la valeur de la ligne sélectionnée dans un contrôle de type zone de liste (ListBox) ou zone déroulante (ComboBox) pour le contrôle spécifié par le paramètre *controlPropertyName*.

### Exemple

L'exemple suivant renvoie le libellé et la valeur de la ligne actuellement sélectionnée pour le contrôle `myListBox` :

```
var elem = new Object();  
elem = fl.xmlui.getControlItemElement("myListBox");  
fl.trace("label = " + elem.label + " value = " + elem.value);
```

### Voir aussi

```
fl.xmlui, document.xmlPanel(), xmlui.get(), xmlui.setControlItemElement(),  
xmlui.setControlItemElements()
```

## xmlui.setEnabled()

### Disponibilité

Flash 8.

### Utilisation

```
xmlui.setEnabled(controlID)
```

### Paramètres

**controlID** Chaîne spécifiant l'attribut ID du contrôle dont vous recherchez l'état.

### Valeur renvoyée

Valeur booléenne, `true` si le contrôle est activé, `false` dans le cas contraire.

### Description

Méthode qui renvoie une valeur booléenne indiquant si le contrôle est activé ou désactivé (affiché en grisé).

### Exemple

L'exemple suivant renvoie une valeur indiquant si le contrôle dont l'attribut ID correspond à `myListBox` est activé :

```
var isEnabled = fl.xmlui.setEnabled("myListBox");  
fl.trace(isEnabled);
```

### Voir aussi

[fl.xmlui](#), [document.xmlPanel\(\)](#), [xmlui.setEnabled\(\)](#)

## xmlui.getVisible()

### Disponibilité

Flash 8.

### Utilisation

```
xmlui.getVisible(controlID)
```

### Paramètres

**controlID** Chaîne spécifiant l'attribut ID du contrôle dont vous recherchez l'état de visibilité.

### Valeur renvoyée

Valeur booléenne, `true` si le contrôle est visible, ou `false` s'il est invisible (masqué).

### Description

Méthode qui renvoie une valeur booléenne indiquant si le contrôle est visible ou masqué.

### Exemple

L'exemple suivant renvoie une valeur indiquant si le contrôle dont l'attribut ID correspond à `myListBox` est visible :



```
var isVisible = fl.xmlui.getVisible("myListBox");  
fl.trace(isVisible);
```

#### Voir aussi

`xmlui.setVisible()`

## xmlui.set()

#### Disponibilité

Flash MX 2004.

#### Utilisation

```
xmlui.set(controlPropertyName, value)
```

#### Paramètres

**controlPropertyName** Chaîne spécifiant le nom de la propriété XMLUI à modifier.

**value** Chaîne spécifiant la valeur à donner à la propriété XMLUI.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui modifie la valeur de la propriété spécifiée de la boîte de dialogue XMLUI active.

#### Exemple

L'exemple suivant définit la valeur d'une propriété nommée URL sur `www.adobe.com` :

```
fl.xmlui.set("URL", "www.adobe.com");
```

#### Voir aussi

`fl.xmlui`, `document.xmlPanel()`, `xmlui.get()`, `xmlui.setControlItemElement()`,  
`xmlui.setControlItemElements()`

## xmlui.setControlItemElement()

#### Disponibilité

Flash 8.

#### Utilisation

```
xmlui.setControlItemElement(controlPropertyName, elementItem)
```

#### Paramètres

**controlPropertyName** Chaîne spécifiant l'élément de contrôle à définir.

**elementItem** Objet JavaScript doté d'une propriété de chaîne nommée `label` et d'une propriété de chaîne facultative nommée `value`. Si la propriété `value` n'existe pas, elle est créée et reçoit la même valeur que `label`.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui définit le libellé et la valeur de la ligne actuellement sélectionnée dans le contrôle de type zone de liste (ListBox) ou zone déroulante (ComboBox) spécifié par le paramètre *controlPropertyName*.

#### Exemple

L'exemple suivant définit le libellé et la valeur de l'élément actif de la propriété de contrôle `PhoneNumber` :

```
var elem = new Object();  
elem.label = "Fax";  
elem.value = "707-555-5555";  
fl.xmlui.setControlItemElement("PhoneNumber", elem);
```

#### Voir aussi

```
fl.xmlui, document.xmlPanel(), xmlui.getControlItemElement(), xmlui.set(),  
xmlui.setControlItemElements()
```

## xmlui.setControlItemElements()

#### Disponibilité

Flash 8.

#### Utilisation

```
xmlui.setControlItemElements(controlID, elementItemArray)
```

#### Paramètres

**controlID** Chaîne spécifiant l'attribut ID du contrôle à définir.

**elementItemArray** Tableau constitué d'objets JavaScript dotés chacun d'une propriété de chaîne nommée `label` et d'une propriété de chaîne facultative nommée `value`. Si la propriété `value` n'existe pas, elle est créée et reçoit la même valeur que `label`.

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui efface les valeurs du contrôle de type zone de liste ou zone déroulante spécifié par *controlID* et remplace les éléments de liste ou les commandes de menu par les paires `label`, `value` spécifiées par *elementItemArray*.

#### Exemple

L'exemple suivant définit le libellé et la valeur des éléments associés au contrôle possédant l'attribut d'ID `myControlID` sur les paires `label`, `value` spécifiées :

```
var nameArray = new Array("January", "February", "March");
var monthArray = new Array();
for (i=0;i<nameArray.length;i++){
    elem = new Object();
    elem.label = nameArray[i];
    elem.value = i;
    monthArray[i] = elem;
}
fl.xmlui.setControlItemElements("myControlID", monthArray);
```

#### Voir aussi

[xmlui.getControlItemElement\(\)](#), [xmlui.set\(\)](#), [xmlui.setControlItemElement\(\)](#)

## xmlui.setEnabled()

#### Disponibilité

Flash 8.

#### Utilisation

```
xmlui.setEnabled(controlID, enable)
```

#### Paramètres

**controlID** Chaîne spécifiant l'attribut ID du contrôle à activer ou désactiver.

**enable** Valeur booléenne, *true* si vous souhaitez activer le contrôle, *false* pour le désactiver (l'afficher en grisé).

#### Valeur renvoyée

Aucune.

#### Description

Méthode qui active ou désactive (affiche en grisé) un contrôle.

#### Exemple

L'exemple suivant affiche en grisé le contrôle dont l'attribut ID est `myControl` :

```
fl.xmlui.setEnabled("myControl", false);
```

#### Voir aussi

[xmlui.setEnabled\(\)](#)

## xmlui.setVisible()

#### Disponibilité

Flash 8.

#### Utilisation

```
xmlui.setVisible(controlID, visible)
```

### Paramètres

**controlID** Chaîne spécifiant l'attribut ID du contrôle à afficher ou masquer.

**visible** Valeur booléenne, `true` si vous voulez afficher le contrôle, et `false` pour le masquer.

### Valeur renvoyée

Aucune.

### Description

Méthode qui affiche ou masque un contrôle.

### Exemple

L'exemple suivant masque le contrôle dont l'attribut ID est `myControl` :

```
fl.xmlui.setVisible("myControl", false);
```

### Voir aussi

[xmlui.getVisible\(\)](#)

# Chapitre 51 : Extensibilité de niveau C

Ce chapitre décrit le mécanisme d'extensibilité de niveau C qui permet d'implémenter des fichiers d'extension Adobe Flash CS4 Professional à l'aide d'une combinaison de codes JavaScript et C personnalisé. Ce mécanisme n'a pas été modifié dans cette version de Flash.

## A propos de l'extensibilité

Pour implémenter l'extensibilité, les fonctions sont définies en C, puis regroupées dans une DLL ou une bibliothèque partagée qui est ensuite enregistrée dans le répertoire approprié. Ces fonctions peuvent alors être appelées en JavaScript à l'aide de l'API JavaScript d'Adobe Flash.

Par exemple, il est possible de définir en C une fonction effectuant des calculs complexes beaucoup plus rapidement qu'en JavaScript, ou de créer des outils ou des effets très sophistiqués.

Ce mécanisme d'extensibilité est un sous-ensemble de l'API d'Adobe Dreamweaver CS3. Si vous connaissez déjà cette API, les fonctions de l'API du mécanisme d'extensibilité de niveau C ne vous sembleront pas vraiment étranges. Toutefois, cette API diffère de l'API de Dreamweaver sur certains points :

- Cette API ne contient pas toutes les commandes de l'API de Dreamweaver.
- Toutes les déclarations qui sont de type `wchar_t` et `char` dans l'API de Dreamweaver sont implémentées comme déclarations `unsigned short` dans celle-ci, afin d'assurer la prise en charge d'Unicode pour le transfert de chaînes.
- La fonction `JSVal JS_BytesToValue()` de cette API ne figure pas dans l'API de Dreamweaver.
- L'emplacement où les DLL et les fichiers de bibliothèque partagée doivent être stockés est différent (voir « [Intégration de fonctions en C](#) » à la page 535).

## Intégration de fonctions en C

Le mécanisme d'extensibilité de niveau C permet d'implémenter des fichiers d'extension Flash à l'aide d'une combinaison de codes JavaScript et C. Le processus d'implémentation de cette possibilité se résume ainsi :

- 1 Définir des fonctions en langage C ou C++.
- 2 Regrouper ces fonctions dans une DLL (Windows) ou une bibliothèque partagée (Macintosh).
- 3 Enregistrer le fichier de la DLL ou de la bibliothèque à l'emplacement approprié :
  - Windows 7 :  
*lecteur de démarrage\Utilisateurs\nom d'utilisateur\AppData\Adobe\Flash CS5 ou CS5.5\langue\Configuration\External Libraries*
  - Windows Vista :  
*lecteur de démarrage\Utilisateurs\nom d'utilisateur\Local Settings\Application Data\Adobe\Flash CS5 ou CS5.5\langue\Configuration\External Libraries*
  - Windows XP :

*lecteur de démarrage\Documents and Settings\nom d'utilisateur\Local Settings\Application Data\Adobe\Flash CS5 ou CS5.5\langue\Configuration\External Libraries*

- Mac OS X :

*Macintosh HD/Utilisateurs/nom d'utilisateur/Bibliothèque/Application Support/Adobe/Flash CS5 ou CS5.5/langue/Configuration/External Libraries*

4 Créez un fichier JSFL qui appelle ces fonctions.

5 Exécutez le fichier JSFL à partir du menu Commandes dans l'environnement de programmation Flash.

Pour plus d'informations, voir « [Exemple d'implémentation d'une DLL](#) » à la page 539.

## Extensibilité de niveau C et interpréteur JavaScript

L'interaction entre l'API JavaScript de Flash et le code C contenu dans la DLL ou la bibliothèque partagée comporte trois étapes :

- Au démarrage, pour enregistrer les fonctions de la bibliothèque
- Lors de l'appel de la fonction en C, pour transmettre les arguments passés de JavaScript en C
- Avant le retour de la fonction C, pour transmettre la valeur renvoyée

Pour accomplir ces tâches, l'interpréteur définit plusieurs types de données et expose une API. Les définitions des types de données et des fonctions présentées dans cette section apparaissent dans le fichier `mm_jsapi.h`. Pour permettre à votre bibliothèque de fonctionner correctement, vous devez indiquer en "include" le fichier `mm_jsapi.h` au début de chaque fichier de votre bibliothèque, à l'aide de la ligne suivante :

```
#include "mm_jsapi.h"
```

Cette inclusion du fichier `mm_jsapi.h` comprend également le fichier `mm_jsapi_environment.h` qui définit la structure `MM_Environment`.

Pour obtenir le fichier `mm_jsapi.h`, vous pouvez l'extraire du fichier d'exemple ZIP ou SIT (voir « [Exemple d'implémentation d'une DLL](#) » à la page 539) ou copier le code ci-dessous dans un fichier nommé `mm_jsapi.h` :

```
#ifndef _MM_JSAPI_H_
#define _MM_JSAPI_H_

/*****
 * Public data types
 *****/

typedef struct JSContext JSContext;
typedef struct JSObject JSObject;
typedef long jsval;
#ifndef JSBool
typedef long JSBool;
#endif

typedef JSBool (*JSNative)(JSContext *cx, JSObject *obj, unsigned int argc,
jsval *argv, jsval *rval);

/* Possible values for JSBool */
#define JS_TRUE 1
#define JS_FALSE 0
```

```
/* *****  
 * Public functions  
 * ***** */  
  
/* JSBool JS_DefineFunction(unsigned short *name, JSNative call, unsigned int nargs) */  
#define JS_DefineFunction(n, c, a) \  
(mmEnv.defineFunction ? (*(mmEnv.defineFunction))(mmEnv.libObj, n, c, a) \  
: JS_FALSE)  
  
/* unsigned short *JS_ValueToString(JSContext *cx, jsval v, unsigned int *pLength) */  
#define JS_ValueToString(c, v, l) \  
(mmEnv.valueToString? (*(mmEnv.valueToString))(c, v, l) : (char *)0)  
  
/* unsigned char *JS_ValueToBytes(JSContext *cx, jsval v, unsigned int *pLength) */  
#define JS_ValueToBytes(c, v, l) \  
(mmEnv.valueToBytes? (*(mmEnv.valueToBytes))(c, v, l) : (unsigned char *)0)  
  
/* JSBool JS_ValueToInteger(JSContext *cx, jsval v, long *lp); */  
#define JS_ValueToInteger(c, v, l) \  
(mmEnv.valueToInteger ? (*(mmEnv.valueToInteger))(c, v, l) : JS_FALSE)  
  
/* JSBool JS_ValueToDouble(JSContext *cx, jsval v, double *dp); */  
#define JS_ValueToDouble(c, v, d) \  
(mmEnv.valueToDouble? (*(mmEnv.valueToDouble))(c, v, d) : JS_FALSE)  
  
/* JSBool JS_ValueToBoolean(JSContext *cx, jsval v, JSBool *bp); */  
#define JS_ValueToBoolean(c, v, b) \  
(mmEnv.valueToBoolean ? (*(mmEnv.valueToBoolean))(c, v, b) : JS_FALSE)  
  
/* JSBool JS_ValueToObject(JSContext *cx, jsval v, JSObject **op); */  
#define JS_ValueToObject(c, v, o) \  
(mmEnv.valueToObject? (*(mmEnv.valueToObject))(c, v, o) : JS_FALSE)  
  
/* JSBool JS_StringToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp); */  
#define JS_StringToValue(c, b, s, v) \  
(mmEnv.stringToValue? (*(mmEnv.stringToValue))(c, b, s, v) : JS_FALSE)  
  
/* JSBool JS_BytesToValue(JSContext *cx, unsigned char *bytes, uint sz, jsval *vp); */  
#define JS_BytesToValue(c, b, s, v) \  
(mmEnv.bytesToValue? (*(mmEnv.bytesToValue))(c, b, s, v) : JS_FALSE)  
  
/* JSBool JS_DoubleToValue(JSContext *cx, double dv, jsval *vp); */  
#define JS_DoubleToValue(c, d, v) \  
(mmEnv.doubleToValue? (*(mmEnv.doubleToValue))(c, d, v) : JS_FALSE)  
  
/* jsval JS_IntegerToValue(long lv); */  
#define JS_IntegerToValue(lv) (((jsval)(lv) << 1) | 0x1)  
  
/* jsval JS_BooleanToValue(JSBool bv); */  
#define JS_BooleanToValue(bv) (((jsval)(bv) << 3) | 0x6)  
  
/* jsval JS_ObjectToValue(JSObject *obj); */  
#define JS_ObjectToValue(ov) ((jsval)(ov))  
  
/* unsigned short *JS_ObjectType(JSObject *obj); */  
#define JS_ObjectType(o) \
```

```
(mmEnv.objectType ? (*(mmEnv.objectType))(o) : (char *)0)

/* JSObject *JS_NewArrayObject(JSContext *cx, unsigned int length, jsval *v) */
#define JS_NewArrayObject(c, l, v) \
(mmEnv.newArrayObject ? (*(mmEnv.newArrayObject))(c, l, v) : (JSObject *)0)

/* long JS_GetArrayLength(JSContext *cx, JSObject *obj) */
#define JS_GetArrayLength(c, o) \
(mmEnv.getArrayLength ? (*(mmEnv.getArrayLength))(c, o) : -1)

/* JSBool JS_GetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp) */
#define JS_GetElement(c, o, i, v) \
(mmEnv.getElement ? (*(mmEnv.getElement))(c, o, i, v) : JS_FALSE)

/* JSBool JS_SetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp) */
#define JS_SetElement(c, o, i, v) \
(mmEnv.setElement ? (*(mmEnv.setElement))(c, o, i, v) : JS_FALSE)

/* JSBool JS_ExecuteScript(JSContext *cx, JSObject *obj, unsigned short *script,
    * unsigned int sz, jsval *rval) */
#define JS_ExecuteScript(c, o, s, z, r) \
(mmEnv.executeScript ? (*(mmEnv.executeScript))(c, o, s, z, (LPCTSTR) __FILE__, \
    __LINE__, r) : JS_FALSE)

/* JSBool JS_ReportError(JSContext *cx, unsigned short *error, unsigned int sz) */
#define JS_ReportError(c, e, s) \
(mmEnv.reportError ? (*(mmEnv.reportError))(c, e, s) : JS_FALSE)

/*****
    * Private data types, macros, and globals
    *****/

typedef struct {
JSObject *libObj;
JSBool (*defineFunction)(JSObject *libObj, unsigned short *name, JSNative call,
    unsigned int nargs);
unsigned short *(*valueToString)(JSContext *cx, jsval v, unsigned int *pLength);
unsigned char *(*valueToBytes)(JSContext *cx, jsval v, unsigned int *pLength);
JSBool (*valueToInteger)(JSContext *cx, jsval v, long *lp);
JSBool (*valueToDouble)(JSContext *cx, jsval v, double *dp);
JSBool (*valueToBoolean)(JSContext *cx, jsval v, JSBool *bp);
JSBool (*valueToObject)(JSContext *cx, jsval v, JSObject **op);
JSBool (*stringToValue)(JSContext *cx, unsigned short *b, unsigned int sz, jsval *vp);
JSBool (*bytesToValue)(JSContext *cx, unsigned char *b, unsigned int sz, jsval *vp);
JSBool (*doubleToValue)(JSContext *cx, double dv, jsval *vp);
unsigned short *(*objectType)(JSObject *obj);
JSObject *(*newArrayObject)(JSContext *cx, unsigned int length, jsval *vp);
long (*getArrayLength)(JSContext *cx, JSObject *obj);
JSBool (*getElement)(JSContext *cx, JSObject *obj, unsigned int idx,
    jsval *vp);
JSBool (*setElement)(JSContext *cx, JSObject *obj, unsigned int idx,
    jsval *vp);
JSBool (*executeScript)(JSContext *cx, JSObject *obj, unsigned short *script,
    unsigned int sz, unsigned short *file, unsigned int lineNum, jsval *rval);
JSBool (*reportError)(JSContext *cx, unsigned short *error, unsigned int sz);
```



```
} MM_Environment;

extern MM_Environment mmEnv;

// Declare the external entry point and linkage
#ifdef _WIN32
# ifndef _MAC
// Windows
__declspec( dllexport ) void MM_InitWrapper( MM_Environment *env, unsigned int envSize );
# endif
#else
extern void MM_InitWrapper( MM_Environment *env, unsigned int envSize );
#endif

#define MM_STATE\
/* Definitions of global variables */ \
MM_Environment mmEnv; \
\
void\
MM_InitWrapper(MM_Environment *env, unsigned int envSize) \
{ \
extern void MM_Init();\
\
char **envPtr = (char **)env; \
char **mmPtr = (char **)(&mmEnv);\
char **envEnd = (char **)((char *)envPtr + envSize);\
char **mmEnd = (char **)((char *)mmPtr + sizeof(MM_Environment)); \
\
/* Copy fields from env to mmEnv, one pointer at a time */\
while (mmPtr < mmEnd && envPtr < envEnd)\
*mmPtr++ = *envPtr++; \
\
/* If env doesn't define all of mmEnv's fields, set extras to NULL */ \
while (mmPtr < mmEnd) \
*mmPtr++ = (char *)0; \
\
/* Call user's MM_Init function */\
MM_Init();\
} \

#endif /* _MM_JSAPI_H_ */
```

## Exemple d'implémentation d'une DLL

Cette section montre comment créer une implémentation de DLL simple. Pour découvrir le fonctionnement du processus sans réellement créer la DLL, vous pouvez installer les exemples de fichier DLL fournis dans le fichier Samples.zip ; les fichiers sont situés dans le dossier ExtendingFlash/dllSampleComputeSum. (Pour plus d'informations sur le téléchargement du fichier Samples.zip, voir « [Exemples d'implémentation](#) » à la page 13.) Extrayez les fichiers d'exemple du fichier dllSampleComputeSum.dmg ou dllSampleComputeSum.zip, puis procédez comme suit :

- Enregistrez le fichier Sample.jsfl dans le répertoire Configuration/Commands (voir « [Enregistrement des fichiers JSFL](#) » à la page 2).

- Enregistrez le fichier Sample.dll dans le répertoire Configuration/External Libraries (voir « [Intégration de fonctions en C](#) » à la page 535).
- Dans l'environnement de programmation de Flash, choisissez Commandes > Echantillon. L'instruction trace du fichier JSFL affiche dans le panneau Sortie les résultats de la fonction définie dans Sample.dll.

La suite de cette section présente le développement de l'exemple. Dans ce cas précis, la DLL ne contient qu'une fonction, qui additionne deux chiffres. Le code C est présenté dans l'exemple suivant :

```
// Source code in C
// Save the DLL or shared library with the name "Sample".
#include <windows.h>
#include <stdlib.h>

#include "mm_jsapi.h"

// A sample function
// Every implementation of a JavaScript function must have this signature.
JSBool computeSum(JSContext *cx, JSObject *obj, unsigned int argc, jsval *argv, jsval *rval)
{
    long a, b, sum;

    // Make sure the right number of arguments were passed in.
    if (argc != 2)
        return JS_FALSE;

    // Convert the two arguments from jsvals to longs.
    if (JS_ValueToInteger(cx, argv[0], &a) == JS_FALSE ||
        JS_ValueToInteger(cx, argv[1], &b) == JS_FALSE)
        return JS_FALSE;

    /* Perform the actual work. */
    sum = a + b;

    /* Package the return value as a jsval. */
    *rval = JS_IntegerToValue(sum);

    /* Indicate success. */
    return JS_TRUE;
}
```

Après l'écriture de ce code, constituez le fichier DLL ou la bibliothèque partagée pour l'enregistrer ensuite dans le répertoire approprié, Configuration/ExternalLibraries (voir « [Intégration de fonctions en C](#) » à la page 535). Créez ensuite un fichier JSFL contenant le code suivant et enregistrez-le dans le répertoire Configuration/Commands (voir « [Enregistrement des fichiers JSFL](#) » à la page 2).

```
// JSFL file to run C function defined above.
var a = 5;
var b = 10;
var sum = Sample.computeSum(a, b);
fl.trace("The sum of " + a + " and " + b + " is " + sum );
```

Pour exécuter la fonction définie dans la DLL, choisissez Commandes > Echantillon dans l'environnement de programmation Flash.

## Types de données

L'interpréteur JavaScript définit les types de données décrits dans cette section.

### **typedef struct JSContext JSContext**

Un pointeur sur ce type de données opaque est transmis à la fonction de niveau C. Certaines fonctions de l'API acceptent ce pointeur parmi leurs arguments.

### **typedef struct JSObject JSObject**

Un pointeur sur ce type de données opaque est transmis à la fonction de niveau C. Ce type de données représente un objet, qui peut être de type tableau ou autre.

### **typedef struct jsval jsval**

Structure de données opaque pouvant contenir un entier ou un pointeur vers un nombre en virgule flottante, une chaîne ou un objet. Certaines fonctions de l'API peuvent lire les valeurs des arguments de la fonction dans une structure `jsval`, et certaines permettent d'écrire la valeur renvoyée par la fonction dans une structure `jsval`.

### **typedef enum { JS\_FALSE = 0, JS\_TRUE = 1 } JSBool**

Type de données simple contenant une valeur booléenne.

## API de niveau C

L'API d'extensibilité de niveau C se compose de la signature de fonction `JSBool (*JSNative)` et des fonctions suivantes :

- `JSBool JS_DefineFunction()`
- `unsigned short *JS_ValueToString()`
- `JSBool JS_ValueToInteger()`
- `JSBool JS_ValueToInteger()`
- `JSBool JS_ValueToBoolean()`
- `JSBool JS_ValueToObject()`
- `JSBool JS_StringToValue()`
- `JSBool JS_DoubleToValue()`
- `JSVal JS_BooleanToValue()`
- `JSVal JS_BytesToValue()`
- `JSVal JS_IntegerToValue()`
- `JSVal JS_ObjectToValue()`
- `unsigned short *JS_ObjectType()`
- `JSObject *JS_NewArrayObject()`
- `long JS_GetArrayLength()`

- `JSBool JS_GetElement()`
- `JSBool JS_SetElement()`
- `JSBool JS_ExecuteScript()`

**typedef JSBool (\*JSNative)(JSContext \*cx, JSObject \*obj, unsigned int argc, jsval \*argv, jsval \*rval)**

### Description

Méthode qui décrit les implémentations de niveau C des fonctions JavaScript dans les situations suivantes :

- Le pointeur *cx* pointe sur une structure `JSContext` opaque, qui doit être transmise à certaines fonctions de l'API JavaScript. Cette variable contient le contexte d'exécution de l'interpréteur.
- Le pointeur *obj* pointe sur l'objet dans le contexte duquel le script s'exécute. Pendant l'exécution du script, le mot-clé `this` est égal à cet objet.
- L'entier *argc* correspond au nombre d'arguments transmis à la fonction.
- Le pointeur *argv* pointe sur un tableau de structures `jsval`. La longueur du tableau correspond au nombre d'éléments défini par *argc*.
- Le pointeur *rval* pointe sur une seule structure `jsval`. La valeur renvoyée par la fonction doit être écrite dans *\*rval*.

La fonction renvoie la valeur `JS_TRUE` si elle s'exécute correctement, et `JS_FALSE` dans le cas contraire. Si la fonction renvoie `JS_FALSE`, l'exécution du script est interrompue et un message d'erreur s'affiche.

## JSBool JS\_DefineFunction()

### Utilisation

`JSBool JS_DefineFunction(unsigned short *name, JSNative call, unsigned int nargs)`

### Description

Cette méthode enregistre une fonction de niveau C auprès de l'interpréteur JavaScript de Flash. Lorsque la fonction `JS_DefineFunction()` a enregistré la fonction de niveau C spécifiée dans l'argument *call*, vous pouvez appeler cette dernière dans un script JavaScript à l'aide du nom indiqué dans l'argument *name*. L'argument *name* est sensible à la casse.

En règle générale, cette fonction est appelée depuis la fonction `MM_Init()`, que Flash appelle au démarrage.

### Arguments

`unsigned short *name, JSNative call, unsigned int nargs`

- L'argument *name* est le nom de la fonction tel qu'il est exposé à JavaScript.
- L'argument *call* est un pointeur vers une fonction de niveau C. La fonction doit renvoyer une valeur `JSBool` qui indique son succès ou son échec.
- L'argument *nargs* indique le nombre d'arguments attendus par la fonction.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## unsigned short \*JS\_ValueToString()

### Utilisation

```
unsigned short *JS_ValueToString(JSContext *cx, jsval v, unsigned int *pLength)
```

### Description

Cette méthode extrait un argument de fonction d'une structure `jsval`, le convertit en chaîne si possible, puis renvoie la valeur ainsi convertie à la routine appelante.

**Remarque :** ne modifiez pas le pointeur sur le tampon renvoyé ; vous risqueriez d'altérer les structures de données de l'interpréteur JavaScript. Pour modifier la chaîne, copiez ses caractères dans un autre tampon et créez une nouvelle chaîne en JavaScript.

### Arguments

`JSContext *cx`, `jsval v`, `unsigned int *pLength`

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `v` est la structure `jsval` dont la chaîne doit être extraite.
- L'argument `pLength` est un pointeur vers un entier non signé. Cette fonction définit `*pLength` sur la longueur de la chaîne (en octets).

### Valeur renvoyée

Un pointeur vers une chaîne terminée par un octet nul en cas de succès, ou la valeur `null` en cas d'échec. La routine appelante ne doit pas libérer cette chaîne à la fin de son exécution.

## JSBool JS\_ValueToInteger()

### Utilisation

```
JSBool JS_ValueToInteger(JSContext *cx, jsval v, long *lp);
```

### Description

Cette méthode extrait un argument de fonction d'une structure `jsval`, le convertit en entier si possible, puis renvoie la valeur ainsi convertie à la routine appelante.

### Arguments

`JSContext *cx`, `jsval v`, `long *lp`

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `v` est la structure `jsval` dont l'entier doit être extrait.
- L'argument `lp` est un pointeur vers un entier de 4 octets. Cette fonction enregistre la valeur convertie dans `*lp`.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## JSBool JS\_ValueToInteger()

### Utilisation

```
JSBool JS_ValueToDouble(JSContext *cx, jsval v, double *dp);
```

### Description

Cette méthode extrait un argument de fonction d'une structure `jsval`, le convertit en double si possible, puis renvoie la valeur ainsi convertie à la routine appelante.

### Arguments

`JSContext *cx`, `jsval v`, `double *dp`

- L'argument `cx` est le pointeur opaque vers `JSContext` qui est transmis à la fonction JavaScript.
- L'argument `v` est la structure `jsval` dont le double doit être extrait.
- L'argument `dp` est un pointeur vers un nombre double de 8 octets. Cette fonction stocke la valeur convertie dans `*dp`.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## JSBool JS\_ValueToBoolean()

### Utilisation

```
JSBool JS_ValueToBoolean(JSContext *cx, jsval v, JSBool *bp);
```

### Description

Cette méthode extrait un argument de fonction d'une structure `jsval`, le convertit en booléen si possible, puis renvoie la valeur ainsi convertie à la routine appelante.

### Arguments

`JSContext *cx`, `jsval v`, `JSBool *bp`

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `v` est la structure `jsval` dont la valeur booléenne doit être extraite.
- L'argument `bp` est un pointeur vers une valeur booléenne `JSBool`. Cette fonction stocke la valeur convertie dans `*bp`.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## JSBool JS\_ValueToObject()

### Utilisation

```
JSBool JS_ValueToObject(JSContext *cx, jsval v, JSObject **op);
```

### Description

Cette méthode extrait un argument de fonction d'une structure `jsval`, le convertit en objet si possible, puis renvoie la valeur ainsi convertie à la routine appelante. Si l'objet est un tableau, utilisez `JS_GetArrayLength()` et `JS_GetElement()` pour lire son contenu.

### Arguments

`JSContext *cx, jsval v, JSObject **op`

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `v` est la structure `jsval` dont l'objet doit être extrait.
- L'argument `op` est un pointeur vers un pointeur `JSObject`. Cette fonction stocke la valeur convertie dans `*op`.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## JSBool JS\_StringToValue()

### Utilisation

```
JSBool JS_StringToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp);
```

### Description

Cette méthode stocke la valeur de renvoi d'une chaîne dans une structure `jsval`. Elle alloue un nouvel objet chaîne JavaScript.

### Arguments

`JSContext *cx, unsigned short *bytes, size_t sz, jsval *vp`

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `bytes` est la chaîne à stocker dans la structure `jsval`. Les données de la chaîne sont copiées. Il est donc conseillé de libérer la chaîne lorsqu'elle n'est plus nécessaire. Si la taille de la chaîne n'est pas spécifiée (voir l'argument `sz`), la chaîne doit se terminer par un octet de valeur nulle.
- L'argument `sz` est la taille de la chaîne, en octets. Si la valeur de `sz` est 0, la longueur de la chaîne terminée par un octet de valeur nulle est calculée automatiquement.
- L'argument `vp` est un pointeur vers la structure `jsval` dans laquelle le contenu de la chaîne doit être copié.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## JSBool JS\_DoubleToValue()

### Utilisation

```
JSBool JS_DoubleToValue(JSContext *cx, double dv, jsval *vp);
```

### Description

Cette méthode stocke la valeur de renvoi d'un nombre en virgule flottante dans une structure `jsval`.

### Arguments

`JSContext *cx, double dv, jsval *vp`

- L'argument *cx* est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument *dv* est un nombre en virgule flottante de 8 octets.
- L'argument *vp* est un pointeur vers la structure `jsval` dans laquelle le contenu du double doit être copié.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

## JSVal JS\_BooleanToValue()

### Utilisation

```
jsval JS_BooleanToValue(JSBool bv);
```

### Description

Cette méthode stocke la valeur booléenne de renvoi dans une structure `jsval`.

### Arguments

`JSBool bv`

- L'argument *bv* est une valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.

### Valeur renvoyée

Structure `JSVal` qui contient la valeur booléenne transmise en tant qu'argument à la fonction.

## JSVal JS\_BytesToValue()

### Utilisation

```
JSBool JS_BytesToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp);
```

### Description

Cette méthode convertit des octets en valeur JavaScript.

### Arguments

`JSContext *cx, unsignedshort *bytes, uintsz, jsval *vp`

- L'argument *cx* est le contexte JavaScript.
- L'argument *bytes* est la chaîne d'octets à convertir en objet JavaScript.
- L'argument *sz* est le nombre d'octets à convertir.
- L'argument *vp* est la valeur JavaScript.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.



## JSVal JS\_IntegerToValue()

### Utilisation

```
jsval JS_IntegerToValue(long lv);
```

### Description

Cette méthode convertit un entier long en structure JSVal.

### Arguments

*lv*

L'argument *lv* est la valeur de l'entier long à convertir en structure jsval.

### Valeur renvoyée

Structure JSVal qui contient l'entier transmis en tant qu'argument à la fonction.

## JSVal JS\_ObjectToValue()

### Utilisation

```
jsval JS_ObjectToValue(JSObject *obj);
```

### Description

Cette méthode stocke la valeur de renvoi d'un objet dans une structure JSVal. Utilisez JS\_NewArrayObject() pour créer un objet array, et JS\_SetElement() pour définir son contenu.

### Arguments

JSObject \**obj*

L'argument *obj* est un pointeur vers l'objet JSObject à convertir en structure JSVal.

### Valeur renvoyée

Structure JSVal qui contient l'objet transmis en tant qu'argument à la fonction.

## unsigned short \*JS\_ObjectType()

### Utilisation

```
unsigned short *JS_ObjectType(JSObject *obj);
```

### Description

Cette méthode reçoit une référence à un objet et renvoie le nom de la classe de cet objet. Par exemple, si l'objet est de type DOM, la fonction renvoie "Document". Si l'objet est un nœud du document, la fonction renvoie "Element". Pour un objet tableau, la fonction renvoie "Array".

**Remarque :** ne modifiez pas le pointeur sur le tampon renvoyé ; vous risqueriez d'altérer les structures de données de l'interpréteur JavaScript.

### Arguments

JSObject \**obj*

En général, cet argument est transmis et converti à l'aide de la fonction `JS_ValueToObject()`.

#### Valeur renvoyée

Un pointeur sur une chaîne terminée par un octet nul. La routine appelante ne doit pas libérer cette chaîne à la fin de son exécution.

## JSObject \*JS\_NewArrayObject()

#### Utilisation

```
JSObject *JS_NewArrayObject(JSContext *cx, unsigned int length [, jsval *v])
```

#### Description

Cette méthode crée un nouvel objet qui contient un tableau de `JSVals`.

#### Arguments

```
JSContext *cx, unsigned int length, jsval *v
```

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `length` est le nombre maximal d'éléments du tableau.
- L'argument `v` est un pointeur facultatif vers les structures `jsval` à stocker dans le tableau. Si la valeur renvoyée n'est pas `null`, `v` est un tableau qui contient des éléments `length`. Si la valeur renvoyée est `null`, le contenu initial de l'objet tableau est non défini (`undefined`). Il peut être défini à l'aide de la fonction `JS_SetElement()`.

#### Valeur renvoyée

Un pointeur vers un nouvel objet tableau, ou la valeur `null` en cas d'échec.

## long JS\_GetArrayLength()

#### Utilisation

```
long JS_GetArrayLength(JSContext *cx, JSObject *obj)
```

#### Description

Cette méthode reçoit un pointeur sur un objet tableau et extrait le nombre d'éléments de ce tableau.

#### Arguments

```
JSContext *cx, JSObject *obj
```

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `obj` est un pointeur vers un objet tableau.

#### Valeur renvoyée

Le nombre d'éléments du tableau, ou -1 en cas d'échec.

## JSBool JS\_GetElement()

### Utilisation

```
JSBool JS_GetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp)
```

### Description

Cette méthode lit un seul élément dans un objet tableau.

### Arguments

```
JSContext *cx, JSObject *obj, jsint idx, jsval *vp
```

- L'argument *cx* est le pointeur *JSContext* opaque qui est transmis à la fonction JavaScript.
- L'argument *obj* est un pointeur vers un objet tableau.
- L'argument *idx* est un index d'entiers dans le tableau. Le premier élément est l'index 0 et le dernier, l'index (*length* - 1).
- L'argument *vp* est un pointeur vers une structure *jsval* dans laquelle le contenu de la structure *jsval* du tableau doit être copié.

### Valeur renvoyée

Valeur booléenne : *JS\_TRUE* indique le succès de l'opération ; *JS\_FALSE* indique son échec.

## JSBool JS\_SetElement()

### Utilisation

```
JSBool JS_SetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp)
```

### Description

Cette méthode écrit un seul élément dans un objet tableau.

### Arguments

```
JSContext *cx, JSObject *obj, jsint idx, jsval *vp
```

- L'argument *cx* est le pointeur *JSContext* opaque qui est transmis à la fonction JavaScript.
- L'argument *obj* est un pointeur vers un objet tableau.
- L'argument *idx* est un index d'entiers dans le tableau. Le premier élément est l'index 0 et le dernier, l'index (*length* - 1).
- L'argument *vp* est un pointeur vers une structure *jsval* dont le contenu doit être copié dans la structure *jsval* du tableau.

### Valeur renvoyée

Valeur booléenne : *JS\_TRUE* indique le succès de l'opération ; *JS\_FALSE* indique son échec.

## JSBool JS\_ExecuteScript()

### Utilisation

```
JS_ExecuteScript(JSContext *cx, JSObject *obj, unsigned short *script, unsigned int sz, jsval *rval)
```

### Description

Cette méthode compile et exécute une chaîne en JavaScript. Si le script génère une valeur de renvoi, celle-ci est renvoyée dans `*rval`.

### Arguments

`JSContext *cx, JSObject *obj, unsigned short *script, unsigned intsz, jsval *rval`

- L'argument `cx` est le pointeur `JSContext` opaque qui est transmis à la fonction JavaScript.
- L'argument `obj` pointe vers l'objet dans le contexte duquel le script s'exécute. Pendant l'exécution du script, le mot-clé `this` est égal à cet objet. En général, ce pointeur est le pointeur `JSObject` transmis à la fonction JavaScript.
- L'argument `script` est une chaîne qui contient du code JavaScript. Si la taille de la chaîne n'est pas spécifiée (voir l'argument `sz`), la chaîne doit se terminer par un octet de valeur nulle.
- L'argument `sz` est la taille de la chaîne, en octets. Si la valeur de `sz` est 0, la longueur de la chaîne terminée par un octet de valeur nulle est automatiquement calculée.
- L'argument `rval` pointe vers une structure `jsval`. La valeur renvoyée par la fonction est stockée dans `*rval`.

### Valeur renvoyée

Valeur booléenne : `JS_TRUE` indique le succès de l'opération ; `JS_FALSE` indique son échec.