Ampliación de
ADOBE® FLASH® CS4 PROFESSIONAL

Adobe®

© 2008 Adobe Systems Incorporated. Todos los derechos reservados.

Ampliación de Flash® para Windows® y Mac OS

Si esta guía se distribuye con software que incluye un contrato de usuario final, la guía, así como el software descrito en ella, se proporciona con una licencia y sólo puede usarse o copiarse en conformidad con los términos de dicha licencia. Con la excepción de lo permitido por la licencia, ninguna parte de esta guía puede ser reproducida, almacenada en un sistema de recuperación de datos ni transmitida de ninguna forma ni por ningún medio, ya sea electrónico, mecánico, de grabación o de otro tipo, sin el consentimiento previo por escrito de Adobe Systems Incorporated. Tenga en cuenta que el contenido de esta guía está protegido por las leyes de derechos de autor aunque no se distribuya con software que incluya un contrato de licencia de usuario final.

El contenido de esta guía se proporciona exclusivamente con fines informativos, está sujeto a cambios sin previo aviso y no debe interpretarse como un compromiso de Adobe Systems Incorporated. Adobe Systems Incorporated no asume ninguna responsabilidad por los errores o imprecisiones que puedan existir en el contenido informativo de esta guía.

Recuerde que las ilustraciones o imágenes existentes que desee incluir en su proyecto pueden estar protegidas por las leyes de derechos de autor. La incorporación no autorizada de este material en sus trabajos puede infringir los derechos del propietario de los derechos de autor. Asegúrese de obtener los permisos necesarios del propietario de los derechos de autor.

Cualquier referencia a nombres de compañías en plantillas de ejemplo sólo se hace con propósitos de demostración y no está relacionada con ninguna organización real.

Adobe, the Adobe logo, ActionScript, Dreamweaver, Fireworks, Flash, Flash Lite, and Version Cue are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Windows and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries. Macintosh and Mac OS are trademarks of Apple Inc., registered in the United States and other countries. All other trademarks are the property of their respective owners.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

MPEG Layer-3 audio compression technology licensed by Fraunhofer IIS and Thomson Multimedia (http://www.mp3licensing.com).

Speech compression and decompression technology licensed from Nellymoser, Inc. (www.nellymoser.com).

Video compression and decompression is powered by On2 TrueMotion video technology. © 1992-2005 On2 Technologies, Inc. All Rights Reserved. http://www.on2.com.

This product includes software developed by the OpenSymphony Group (http://www.opensymphony.com/).

This product contains either BSAFE and/or TIPEM software by RSA Security, Inc.

Sorenson Spark™ video compression and decompression technology licensed from Sorenson Media, Inc.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA

Notice to U.S. government end users. The software and documentation are “Commercial Items,” as that term is defined at 48 C.F.R. §2.101, consisting of “Commercial Computer Software” and “Commercial Computer Software Documentation,” as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. Consistent with 48 C.F.R. §12.212 or 48 C.F.R. §§227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished-rights reserved under the copyright laws of the United States. Adobe Systems Incorporated, 345 Park Avenue, San Jose, CA 95110-2704, USA. For U.S. Government End Users, Adobe agrees to comply with all applicable equal opportunity laws including, if appropriate, the provisions of Executive Order 11246, as amended, Section 402 of the Vietnam Era Veterans Readjustment Assistance Act of 1974 (38 USC 4212), and Section 503 of the Rehabilitation Act of 1973, as amended, and the regulations at 41 CFR Parts 60-1 through 60-60, 60-250 ,and 60-741. The affirmative action clause and regulations contained in the preceding sentence shall be incorporated by reference.

Contenido

Capítulo 1: Introducción
Trabajo con la API JavaScript 1
Novedades de la API JavaScript 5
Objetos de la API JavaScript 8
Implementaciones de muestra 13

Capítulo 2: Funciones y métodos de nivel superior
activate() 16
alert() 17
configureTool() 17
confirm() 18
deactivate() 19
keyDown() 19
keyUp() 20
mouseDoubleClick() 21
mouseDown() 21
mouseMove() 22
mouseUp() 23
notifySettingsChanged() 23
prompt() 24
setCursor() 25

Capítulo 3: Objeto actionsPanel
actionsPanel.getClassForObject() 26
actionsPanel.getScriptAssistMode() 27
actionsPanel.getSelectedText() 27
actionsPanel.getText() 28
actionsPanel.hasSelection() 29
actionsPanel.replaceSelectedText() 29
actionsPanel.setScriptAssistMode() 30
actionsPanel.setSelection() 31
actionsPanel.setText() 31

Capítulo 4: Objeto BitmapInstance
bitmapInstance.getBits() 33
bitmapInstance.hPixels 34
bitmapInstance.setBits() 35
bitmapInstance.vPixels 35

Capítulo 5: Objeto BitmapItem
bitmapItem.allowSmoothing 37
bitmapItem.compressionType 38
bitmapItem.exportToFile() 38
bitmapItem.fileLastModifiedDate 39

bitmapItem.originalCompressionType 39
bitmapItem.quality 40
bitmapItem.sourceFileExists 40
bitmapItem.sourceFileIsCurrent 41
bitmapItem.sourceFilePath 41
bitmapItem.useDeblocking 42
bitmapItem.useImportedJPEGQuality 42

Capítulo 6: Objeto CompiledClipInstance
compiledClipInstance.accName 44
compiledClipInstance.actionScript 45
compiledClipInstance.description 45
compiledClipInstance.forceSimple 46
compiledClipInstance.shortcut 46
compiledClipInstance.silent 46
compiledClipInstance.tabIndex 47

Capítulo 7: Objeto compilerErrors
compilerErrors.clear() 48
compilerErrors.save() 49

Capítulo 8: Objeto ComponentInstance
componentInstance.parameters 50

Capítulo 9: Objeto componentsPanel
componentsPanel.addItemToDocument() 51
componentsPanel.reload() 52

Capítulo 10: Objeto Contour
contour.fill 53
contour.getHalfEdge() 54
contour.interior 55
contour.orientation 55

Capítulo 11: Objeto Document
document.accName 64
document.addDataToDocument() 65
document.addDataToSelection() 66
document.addFilter() 66
document.addItem() 67
document.addNewLine() 68
document.addNewOval() 68
document.addNewPrimitiveOval() 69
document.addNewPrimitiveRectangle() 70
document.addNewPublishProfile() 71
document.addNewRectangle() 72
document.addNewScene() 73
document.addNewText() 73

document.align() 74
document.allowScreens() 75
document.arrange() 75
document.as3AutoDeclare 76
document.as3Dialect 76
document.as3ExportFrame 77
document.as3StrictMode 77
document.as3WarningsMode 78
document.asVersion 78
document.autoLabel 79
document.backgroundColor 80
document.breakApart() 80
document.canEditSymbol() 81
document.canRevert() 81
document.canSaveAVersion() 82
document.canTestMovie() 82
document.canTestScene() 83
document.changeFilterOrder() 83
document.clipCopy() 84
document.clipCut() 85
document.clipPaste() 85
document.close() 86
document.convertLinesToFills() 87
document.convertToSymbol() 87
document.crop() 88
document.currentPublishProfile 89
document.currentTimeline 89
document.deleteEnvelope() 90
document.deletePublishProfile() 90
document.deleteScene() 91
document.deleteSelection() 92
document.description 92
document.disableAllFilters() 93
document.disableFilter() 93
document.disableOtherFilters() 94
document.distribute() 94
document.distributeToLayers() 95
document.docClass 96
document.documentHasData() 96
document.duplicatePublishProfile() 97
document.duplicateScene() 97
document.duplicateSelection() 98
document.editScene() 98
document.enableAllFilters() 99
document.enableFilter() 100
document.enterEditMode() 100

document.exitEditMode() 101
document.exportPNG() 101
document.exportPublishProfile() 102
document.exportPublishProfileString() 103
document.exportSWF() 103
document.externalLibraryPath 104
document.forceSimple 105
document.frameRate 105
document.getAlignToDocument() 106
document.getBlendMode() 106
document.getCustomFill() 107
document.getCustomStroke() 108
document.getDataFromDocument() 109
document.getElementProperty() 109
document.getElementTextAttr() 110
document.getFilters() 111
document.getMetadata() 111
document.getMobileSettings() 112
document.getPlayerVersion() 113
document.getSelectionRect() 113
document.getTextString() 114
document.getTimeline() 115
document.getTransformationPoint() 116
document.group() 116
document.height 117
document.id 117
document.importFile() 118
document.importPublishProfile() 118
document.importPublishProfileString() 119
document.importSWF() 120
document.intersect() 120
document.library 121
document.libraryPath 121
document.livePreview 122
document.match() 122
document.mouseClick() 123
document.mouseDblClk() 124
document.moveSelectedBezierPointsBy() 124
document.moveSelectionBy() 125
document.name 126
document.optimizeCurves() 126
document.path 127
document.pathURI 127
document.publish() 128
document.publishProfiles 128
document.punch() 129

document.removeAllFilters() 129
document.removeDataFromDocument() 130
document.removeDataFromSelection() 130
document.removeFilter() 131
document.renamePublishProfile() 131
document.renameScene() 132
document.reorderScene() 133
document.resetOvalObject() 133
document.resetRectangleObject() 134
document.resetTransformation() 134
document.revert() 135
document.revertToLastVersion() 135
document.rotate3DSelection() 136
document.rotateSelection() 137
document.save() 137
document.saveAndCompact() 138
document.saveAVersion() 139
document.scaleSelection() 139
document.screenOutline 140
document.selectAll() 141
document.selection 141
document.selectNone() 143
document.setAlignToDocument() 144
document.setBlendMode() 144
document.setCustomFill() 145
document.setCustomStroke() 146
document.setElementProperty() 146
document.setElementTextAttr() 147
document.setFillColor() 148
document.setFilterProperty() 148
document.setFilters() 149
document.setInstanceAlpha() 150
document.setInstanceBrightness() 150
document.setInstanceTint() 151
document.setMetadata() 151
document.setMobileSettings() 153
document.setOvalObjectProperty() 154
document.setPlayerVersion() 154
document.setRectangleObjectProperty() 155
document.setSelectionBounds() 156
document.setSelectionRect() 156
document.setStageVanishingPoint() 157
document.setStageViewAngle() 158
document.setStroke() 158
document.setStrokeColor() 159
document.setStrokeSize() 160

document.setStrokeStyle() 160
document.setTextRectangle() 161
document.setTextSelection() 161
document.setTextString() 162
document.setTransformationPoint() 163
document.silent 164
document.skewSelection() 164
document.smoothSelection() 165
document.sourcePath 165
document.space() 166
document.straightenSelection() 167
document.swapElement() 167
document.swapStrokeAndFill() 168
document.synchronizeWithHeadVersion() 168
document.testMovie() 169
document.testScene() 169
document.timelines 170
document.traceBitmap() 170
document.translate3DCenter() 171
document.translate3DSelection() 172
document.transformSelection() 172
document.unGroup() 173
document.union() 173
document.unlockAllElements() 174
document.viewMatrix 175
document.width 175
document.xmlPanel() 176
document.zoomFactor 176

Capítulo 12: Objeto drawingLayer
drawingLayer.beginDraw() 177
drawingLayer.beginFrame() 178
drawingLayer.cubicCurveTo() 179
drawingLayer.curveTo() 179
drawingLayer.drawPath() 180
drawingLayer.endDraw() 180
drawingLayer.endFrame() 181
drawingLayer.lineTo() 181
drawingLayer.moveTo() 182
drawingLayer.newPath() 182
drawingLayer.setColor() 183
drawingLayer.setFill() 184
drawingLayer.setStroke() 184

Capítulo 13: Objeto Edge
edge.cubicSegmentIndex 185
edge.getControl() 186

edge.getHalfEdge() 186
edge.id 187
edge.isLine 187
edge.setControl() 188
edge.splitEdge() 188
edge.stroke 189

Capítulo 14: Objeto Element

element.depth 191
element.elementType 192
element.getPersistentData() 192
element.getTransformationPoint() 193
element.hasPersistentData() 194
element.height 194
element.layer 194
element.left 195
element.locked 195
element.matrix 196
element.name 196
element.removePersistentData() 197
element.rotation 197
element.scaleX 197
element.scaleY 198
element.selected 198
element.setPersistentData() 199
element.setTransformationPoint() 199
element.skewX 200
element.skewY 201
element.top 201
element.transformX 202
element.transformY 202
element.width 203
element.x 203
element.y 204

Capítulo 15: Objeto Fill

fill.bitmapIsClipped 205
fill.bitmapPath 206
fill.color 206
fill.colorArray 207
fill.focalPoint 207
fill.linearRGB 208
fill.matrix 208
fill.overflow 209
fill.posArray 210
fill.style 210

Capítulo 16: Objeto Filter

filter.angle 212
filter.blurX 212
filter.blurY 213
filter.brightness 213
filter.color 214
filter.contrast 214
filter.distance 215
filter.enabled 216
filter.hideObject 216
filter.highlightColor 217
filter.hue 217
filter.inner 218
filter.knockout 218
filter.name 219
filter.quality 219
filter.saturation 220
filter.shadowColor 221
filter.strength 221
filter.type 222

Capítulo 17: Objeto flash (fl)

fl.actionsPanel 226
fl.addEventListener() 226
fl.as3PackagePaths 227
fl.browseForFileURL() 228
fl.browseForFolderURL() 228
fl.clipCopyString() 229
fl.closeAll() 230
fl.closeAllPlayerDocuments() 230
fl.closeDocument() 231
fl.compilerErrors 232
fl.componentsPanel 232
fl.configDirectory 232
fl.configURI 233
fl.contactSensitiveSelection 233
fl.createDocument() 233
fl.createNewDocList 234
fl.createNewDocListType 235
fl.createNewTemplateList 235
fl.documents 235
fl.downloadLatestVersion() 236
fl.drawingLayer 237
fl.externalLibraryPath 237
fl.fileExists() 237
fl.findDocumentDOM() 238
fl.findDocumentIndex() 239

fl.findObjectInDocByName() 239
fl.findObjectInDocByType() 240
fl.flexSDKPath 242
fl.getAppMemoryInfo() 242
fl.getDocumentDOM() 243
fl.isFontInstalled() 244
fl.libraryPath 244
fl.mapPlayerURL() 245
fl.Math 246
fl.mruRecentFileList 246
fl.mruRecentFileListType 246
fl.objectDrawingMode 247
fl.openDocument() 247
fl.openScript() 248
fl.outputPanel 248
fl.packagePaths 249
fl.presetPanel 249
fl.quit() 250
fl.reloadEffects() 250
fl.reloadTools() 251
fl.removeEventListener() 251
fl.resetAS3PackagePaths() 252
fl.resetPackagePaths() 253
fl.revertDocument() 253
fl.revertDocumentToLastVersion() 254
fl.runScript() 254
fl.saveAll() 255
fl.saveAVersionOfDocument() 256
fl.saveDocument() 257
fl.saveDocumentAs() 257
fl.scriptURI 258
fl.selectElement() 258
fl.selectTool() 259
fl.setActiveWindow() 260
fl.showIdleMessage() 261
fl.sourcePath 261
fl.swfPanels 262
fl.synchronizeDocumentWithHeadVersion() 263
fl.tools 263
fl.trace() 263
fl.version 264
fl.xmlui 265

Capítulo 18: Objeto FLfile

FLfile.copy() 267
FLfile.createFolder() 268

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

Capítulo 19: Objeto folderItem

Capítulo 20: Objeto fontItem
fontItem.bitmap 282
fontItem.bold 283
fontItem.embedVariantGlyphs 283
fontItem.font 285
fontItem.isDefineFont4Symbol 285
fontItem.italic 286
fontItem.size 286

Capítulo 21: Objeto Frame
frame.actionScript 288
frame.duration 289
frame.elements 289
frame.getCustomEase() 289
frame.hasCustomEase 290
frame.labelType 291
frame.motionTweenOrientToPath 291
frame.motionTweenRotate 291
frame.motionTweenRotateTimes 292
frame.motionTweenScale 292
frame.motionTweenSnap 293
frame.motionTweenSync 293
frame.name 293
frame.setCustomEase() 294
frame.shapeTweenBlend 294
frame.soundEffect 295
frame.soundLibraryItem 295
frame.soundLoop 296
frame.soundLoopMode 296
frame.soundName 296

frame.soundSync 297
frame.startFrame 297
frame.tweenEasing 298
frame.tweenType 298
frame.useSingleEaseCurve 299

Capítulo 22: Objeto HalfEdge
halfEdge.getEdge() 300
halfEdge.getNext() 301
halfEdge.getOppositeHalfEdge() 301
halfEdge.getPrev() 302
halfEdge.getVertex() 302
halfEdge.id 303
halfEdge.index 303

Capítulo 23: Objeto Instance
instance.instanceType 305
instance.libraryItem 306

Capítulo 24: Objeto Item
item.addData() 308
item.getData() 308
item.hasData() 309
item.itemType 309
item.linkageBaseClass 310
item.linkageClassName 310
item.linkageExportForAS 311
item.linkageExportForRS 311
item.linkageExportInFirstFrame 312
item.linkageIdentifier 312
item.linkageImportForRS 313
item.linkageURL 313
item.name 314
item.removeData() 314

Capítulo 25: Objeto Layer
layer.color 315
layer.frameCount 316
layer.frames 316
layer.height 317
layer.layerType 317
layer.locked 318
layer.name 318
layer.outline 318
layer.parentLayer 319
layer.visible 319

Capítulo 26: Objeto library

library.addItemToDocument() 322
library.addNewItem() 323
library.deleteItem() 323
library.duplicateItem() 324
library.editItem() 324
library.expandFolder() 325
library.findItemIndex() 326
library.getItemProperty() 326
library.getItemType() 327
library.getSelectedItems() 327
library.importEmbeddedSWF() 328
library.itemExists() 328
library.items 329
library.moveToFolder() 329
library.newFolder() 330
library.renameItem() 331
library.selectAll() 331
library.selectItem() 332
library.selectNone() 333
library.setItemProperty() 333
library.updateItem() 334

Capítulo 27: Objeto Math

Math.concatMatrix() 335
Math.invertMatrix() 336
Math.pointDistance() 336

Capítulo 28: Objeto Matrix

matrix.a 338
matrix.b 339
matrix.c 339
matrix.d 340
matrix.tx 340
matrix.ty 341

Capítulo 29: Objeto outputPanel

outputPanel.clear() 342
outputPanel.save() 343
outputPanel.trace() 343

Capítulo 30: Objeto Oval

OvalObject.closePath 345
OvalObject.endAngle 346
OvalObject.innerRadius 346
OvalObject.startAngle 347

Capítulo 31: Objeto Parameter

parameter.category 348
parameter.insertItem() 349
parameter.listIndex 349
parameter.name 350
parameter.removeItem() 350
parameter.value 351
parameter.valueType 352
parameter.verbose 352

Capítulo 32: Objeto Path

path.addCubicCurve() 353
path.addCurve() 354
path.addPoint() 355
path.clear() 355
path.close() 356
path.makeShape() 356
path.newContour() 357
path.nPts 358

Capítulo 33: Objeto presetItem

presetItem.isDefault 359
presetItem.isFolder 360
presetItem.level 360
presetItem.name 361
presetItem.open 361
presetItem.path 362

Capítulo 34: Objeto presetPanel

presetPanel.addNewItem() 364
presetPanel.applyPreset() 364
presetPanel.deleteFolder() 365
presetPanel.deleteItem() 366
presetPanel.expandFolder() 366
presetPanel.exportItem() 367
presetPanel.findItemIndex() 368
presetPanel.getSelectedItems() 369
presetPanel.importItem() 370
presetPanel.items 370
presetPanel.moveToFolder() 371
presetPanel.newFolder() 372
presetPanel.renameItem() 372
presetPanel.selectItem() 373

Capítulo 35: Objeto Rectangle

RectangleObject.bottomLeftRadius 375
RectangleObject.bottomRightRadius 376
RectangleObject.lockFlag 376

RectangleObject.topLeftRadius 377
RectangleObject.topRightRadius 377

Capítulo 36: Objeto Screen
screen.accName 378
screen.childScreens 379
screen.description 379
screen.forceSimple 380
screen.hidden 380
screen.instanceName 381
screen.name 381
screen.nextScreen 382
screen.parameters 382
screen.parentScreen 383
screen.prevScreen 383
screen.silent 384
screen.tabIndex 384
screen.timeline 385

Capítulo 37: Objeto ScreenOutline
screenOutline.copyScreenFromFile() 387
screenOutline.currentScreen 387
screenOutline.deleteScreen() 388
screenOutline.duplicateScreen() 388
screenOutline.getSelectedScreens() 389
screenOutline.insertNestedScreen() 390
screenOutline.insertScreen() 390
screenOutline.moveScreen() 391
screenOutline.renameScreen() 392
screenOutline.rootScreen 392
screenOutline.screens 393
screenOutline.setCurrentScreen() 393
screenOutline.setScreenProperty() 394
screenOutline.setSelectedScreens() 394

Capítulo 38: Objeto Shape
shape.beginEdit() 397
shape.contours 397
shape.deleteEdge() 398
shape.edges 398
shape.endEdit() 398
shape.getCubicSegmentPoints() 399
shape.isDrawingObject 400
shape.isGroup 400
shape.isOvalObject 401
shape.isRectangleObject 401
shape.members 402

shape.numCubicSegments 402
shape.vertices 403

Capítulo 39: Objeto SoundItem

soundItem.bitRate 405
soundItem.bits 405
soundItem.compressionType 406
soundItem.convertStereoToMono 406
soundItem.exportToFile() 407
soundItem.fileLastModifiedDate 408
soundItem.originalCompressionType 408
soundItem.quality 409
soundItem.sampleRate 409
soundItem.sourceFileExists 410
soundItem.sourceFileIsCurrent 410
soundItem.sourceFilePath 411
soundItem.useImportedMP3Quality 411

Capítulo 40: Objeto Stroke

stroke.breakAtCorners 413
stroke.capType 413
stroke.color 414
stroke.curve 414
stroke.dash1 415
stroke.dash2 415
stroke.density 416
stroke.dotSize 416
stroke.dotSpace 417
stroke.hatchThickness 417
stroke.jiggle 418
stroke.joinType 418
stroke.length 418
stroke.miterLimit 419
stroke.pattern 419
stroke.rotate 420
stroke.scaleType 420
stroke.shapeFill 421
stroke.space 421
stroke.strokeHinting 422
stroke.style 422
stroke.thickness 423
stroke.variation 423
stroke.waveHeight 424
stroke.waveLength 424

Capítulo 41: Objeto swfPanel

swfPanel.call() 426
swfPanel.name 428
swfPanel.path 429

Capítulo 42: Objeto SymbolInstance

symbolInstance.accName 431
symbolInstance.actionScript 432
symbolInstance.blendMode 432
symbolInstance.buttonTracking 432
symbolInstance.cacheAsBitmap 433
symbolInstance.colorAlphaAmount 433
symbolInstance.colorAlphaPercent 434
symbolInstance.colorBlueAmount 434
symbolInstance.colorBluePercent 434
symbolInstance.colorGreenAmount 435
symbolInstance.colorGreenPercent 435
symbolInstance.colorMode 436
symbolInstance.colorRedAmount 436
symbolInstance.colorRedPercent 436
symbolInstance.description 437
symbolInstance.filters 437
symbolInstance.firstFrame 438
symbolInstance.forceSimple 438
symbolInstance.loop 439
symbolInstance.shortcut 439
symbolInstance.silent 440
symbolInstance.symbolType 440
symbolInstance.tabIndex 441

Capítulo 43: Objeto SymbolItem

symbolItem.convertToCompiledClip() 442
symbolItem.exportSWC() 443
symbolItem.exportSWF() 443
symbolItem.scalingGrid 444
symbolItem.scalingGridRect 444
symbolItem.sourceAutoUpdate 445
symbolItem.sourceFilePath 445
symbolItem.sourceLibraryName 446
symbolItem.symbolType 446
symbolItem.timeline 446

Capítulo 44: Objeto Text

text.accName 449
text.antiAliasSharpness 450
text.antiAliasThickness 450
text.autoExpand 451

text.border 451
text.description 451
text.embeddedCharacters 452
text.embedRanges 452
text.embedVariantGlyphs 453
text.fontRenderingMode 453
text.getTextAttr() 454
text.getTextString() 455
text.length 456
text.lineType 456
text.maxCharacters 457
text.orientation 457
text.renderAsHTML 457
text.scrollable 458
text.selectable 458
text.selectionEnd 459
text.selectionStart 459
text.setTextAttr() 460
text.setTextString() 460
text.shortcut 461
text.silent 462
text.tabIndex 462
text.textRuns 463
text.textType 463
text.useDeviceFonts 463
text.variableName 464

Capítulo 45: Objeto TextAttrs

textAttrs.aliasText 466
textAttrs.alignment 466
textAttrs.autoKern 466
textAttrs.bold 467
textAttrs.characterPosition 467
textAttrs.characterSpacing 468
textAttrs.face 468
textAttrs.fillColor 468
textAttrs.indent 469
textAttrs.italic 469
textAttrs.leftMargin 470
textAttrs.letterSpacing 470
textAttrs.lineSpacing 470
textAttrs.rightMargin 471
textAttrs.rotation 471
textAttrs.size 472
textAttrs.target 472
textAttrs.url 472

Capítulo 46: Objeto TextRun

textRun.textAttrs 474
textRun.characters 474

Capítulo 47: Objeto Timeline

timeline.addMotionGuide() 478
timeline.addNewLayer() 478
timeline.clearFrames() 479
timeline.clearKeyframes() 480
timeline.convertToBlankKeyframes() 480
timeline.convertToKeyframes() 481
timeline.copyFrames() 482
timeline.copyMotion() 483
timeline.copyMotionAsAS3() 483
timeline.createMotionTween() 484
timeline.currentFrame 485
timeline.currentLayer 485
timeline.cutFrames() 485
timeline.deleteLayer() 486
timeline.expandFolder() 487
timeline.findLayerIndex() 488
timeline.frameCount 488
timeline.getFrameProperty() 489
timeline.getGuidelines() 489
timeline.getLayerProperty() 490
timeline.getSelectedFrames() 491
timeline.getSelectedLayers() 491
timeline.insertBlankKeyframe() 492
timeline.insertFrames() 493
timeline.insertKeyframe() 494
timeline.layerCount 495
timeline.layers 495
timeline.name 495
timeline.pasteFrames() 496
timeline.pasteMotion() 496
timeline.removeFrames() 497
timeline.reorderLayer() 498
timeline.reverseFrames() 498
timeline.selectAllFrames() 499
timeline.setFrameProperty() 500
timeline.setGuidelines() 500
timeline.setLayerProperty() 501
timeline.setSelectedFrames() 502
timeline.setSelectedLayers() 503
timeline.showLayerMasking() 503

Capítulo 48: Objeto ToolObj

toolObj.depth 506
toolObj.enablePIControl() 506
toolObj.iconID 507
toolObj.position 508
toolObj.setIcon() 508
toolObj.setMenuString() 509
toolObj.setOptionsFile() 509
toolObj.setPI() 510
toolObj.setToolName() 511
toolObj.setToolTip() 511
toolObj.showPIControl() 512
toolObj.showTransformHandles() 513

Capítulo 49: Objeto Tools

tools.activeTool 515
tools.altIsDown 515
tools.constrainPoint() 515
tools.ctlIsDown 516
tools.getKeyDown() 516
tools.mouseIsDown 517
tools.penDownLoc 517
tools.penLoc 518
tools.setCursor() 518
tools.shiftIsDown 519
tools.snapPoint() 519
tools.toolObjs 520

Capítulo 50: Objeto Vertex

vertex.getHalfEdge() 521
vertex.setLocation() 522
vertex.x 522
vertex.y 523

Capítulo 51: Objeto VideoItem

videoItem.exportToFLV() 524
videoItem.fileLastModifiedDate 525
videoItem.sourceFileExists 525
videoItem.sourceFileIsCurrent 526
videoItem.sourceFilePath 526
videoItem.videoType 527

Capítulo 52: Objeto XMLUI

xmlui.accept() 528
xmlui.cancel() 529
xmlui.get() 529
xmlui.getControlItemElement() 530
xmlui.getEnabled() 531

xmlui.getVisible() 531
xmlui.set() 532
xmlui.setControlItemElement() 532
xmlui.setControlItemElements() 533
xmlui.setEnabled() 534
xmlui.setVisible() 535

Capítulo 53: Extensibilidad de nivel C
Extensibilidad 536
Integración de funciones de C 536
Tipos de datos 541
La API de nivel C 542

Capítulo 1: Introducción

Como usuario de Adobe® Flash® CS4 Professional, tal vez Adobe® ActionScript® le resulte familiar: permite crear scripts que se ejecutan en tiempo de ejecución en Adobe® Flash® Player. La interfaz de programación de aplicaciones JavaScript (API JavaScript) de Flash descrita en este documento es una herramienta de programación complementaria que le permite crear scripts que se ejecutan en el entorno de edición.

En este documento se describen los objetos, métodos y propiedades disponibles en la API JavaScript. Se da por sentado que conoce la forma de utilizar los comandos que se describen en este documento cuando trabaja en el entorno de edición. Si tiene alguna duda sobre la función de un determinado comando, consulte otros documentos de la Ayuda de Flash, como el manual *Utilización de Flash*, donde podrá buscar dicha información.

En este documento también se da por sentado que el lector conoce la sintaxis de JavaScript o de ActionScript, además de conceptos básicos de programación como funciones, parámetros y tipos de datos.

Trabajo con la API JavaScript

Con la API JavaScript de Flash se pueden escribir scripts para realizar diversas acciones en el entorno de edición de Flash (es decir, mientras el usuario tiene abierto el programa Flash). Esta funcionalidad es diferente del lenguaje ActionScript, que permite escribir scripts para realizar acciones en el entorno de Flash Player (es decir, mientras se reproduce un archivo SWF), y diferente también de los comandos JavaScript, que se pueden utilizar en las páginas mostradas en un navegador Web.

La API JavaScript permite escribir scripts para aplicaciones Flash que ayudan a agilizar el proceso de edición. Por ejemplo, se pueden escribir scripts para automatizar tareas repetitivas o añadir herramientas personalizadas al panel Herramientas.

La API JavaScript de Flash es similar a la API JavaScript de Adobe® Dreamweaver® y Adobe® Fireworks® (que, a su vez, se diseñaron basándose en la API JavaScript de Netscape). La API JavaScript de Flash se basa en un modelo de objetos de documento (DOM o Document Object Model), que permite acceder a los documentos de Flash empleando objetos JavaScript. La API JavaScript de Flash incluye todos los elementos de la API JavaScript de Netscape, además del DOM de Flash. En este documento se describen estos objetos añadidos y sus métodos y propiedades. Puede utilizar cualquiera de los elementos del lenguaje JavaScript nativo en un script de Flash, pero sólo tendrán efecto los elementos que tengan sentido en el contexto de un documento de Flash.

La API JavaScript también contiene métodos que permiten implementar extensibilidad utilizando una combinación de código JavaScript y C personalizado. Para más información, consulte “Extensibilidad de nivel C” en la página 536.

El intérprete de JavaScript en Flash es el motor Mozilla SpiderMonkey, versión 1.5, disponible en la Web en www.mozilla.org/js/spidermonkey/. SpiderMonkey es una de las dos implementaciones de referencia del lenguaje JavaScript desarrollado por Mozilla.org. Es el mismo motor que se incorpora al navegador de Mozilla.

SpiderMonkey implementa el lenguaje JavaScript básico que se define en la especificación ECMAScript (ECMA-262) edición 3 y es totalmente compatible con la especificación. Sólo son incompatibles los objetos host específicos del navegador que no forman parte de la especificación ECMA-262. Del mismo modo, un gran número de guías de referencia de JavaScript distinguen entre JavaScript básico y de cliente (relacionado con el navegador). Sólo JavaScript básico se aplica al intérprete de JavaScript de Flash.

Creación de archivos JSFL

Puede utilizar Adobe Flash CS4 Professional o el editor de texto que prefiera para escribir y editar archivos JavaScript de Flash (JSFL). Si utiliza Flash, estos archivos usan la extensión .jsfl de forma predeterminada. Para escribir un script, seleccione Archivo > Nuevo > Archivo JavaScript de Flash.

También puede crear un archivo JSFL mediante los comandos del panel Historial. Seguidamente, haga clic en el botón Guardar del panel Historial o seleccione Guardar como comando en el menú del panel. El archivo de comando (JSFL) se guardará en la carpeta de comandos (consulte "Almacenamiento de archivos JSFL" en la página 2). A continuación, podrá abrir el archivo y editarlo de la misma forma que cualquier otro archivo de script.

El panel Historial ofrece también otras opciones que resultan muy útiles. Se pueden copiar los comandos seleccionados en el portapapeles, así como ver los comandos JavaScript que se generan mientras se está trabajando con Flash.

Para copiar comandos del panel Historial en el portapapeles:

1 Seleccione uno o varios comandos en el panel Historial.

2 Realice uno de los siguientes pasos:

- Haga clic en el botón Copiar.
- Seleccione Copiar pasos en el menú del panel.

Para ver los comandos JavaScript en el panel Historial:

- Seleccione Ver > JavaScript en el menú del panel.

Almacenamiento de archivos JSFL

Puede tener disponibles scripts JSFL dentro del entorno de edición de Flash; para ello, debe almacenarlos en una de las distintas carpetas dentro de la carpeta Configuration. De forma predeterminada, la carpeta Configuration se encuentra en la siguiente ubicación:

- Windows® Vista™:

  unidad de arranque\Users*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\

- Windows XP:

  unidad de arranque\Documents and Settings*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\

- Mac OS® X:

  Macintosh HD/Usuarios/*nombre de usuario*/Librería/Application Support/Adobe/Flash CS4/*idioma*/Configuration/

Para determinar la ubicación de la carpeta Configuration, utilice `fl.configDirectory` o `fl.configURI`, como se muestra en el siguiente ejemplo:

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

En la carpeta Configuration, las siguientes carpetas pueden contener scripts accesibles desde el entorno de edición: Behaviors (para que la interfaz de usuario admita comportamientos); Commands (para los scripts que aparecen en el menú Comandos); JavaScript (para los scripts utilizados por el asistente de script para llenar los controles de la interfaz de usuario); Tools (para herramientas ampliables en el panel Herramientas) y WindowSWF (para paneles que aparecen en el menú Ventana). Este documento se centra en los scripts utilizados para comandos y herramientas.

Si se edita un script en la carpeta Commands, el nuevo script quedará disponible de inmediato en Flash. Si se edita un script para una herramienta ampliable, se debe cerrar y reiniciar Flash, o bien utilizar el comando `fl.reloadTools()`. Sin embargo, si se ha utilizado un script para añadir una herramienta ampliable al panel Herramientas y se edita después el script, se deberá quitar para volver a añadir la herramienta al panel Herramientas, o bien cerrar y reiniciar Flash para que la herramienta revisada quede disponible.

Puede guardar los archivos de comandos y herramientas en dos lugares en los que se encontrarán accesibles en el entorno de edición.

- Para los scripts que aparecen como elementos en el menú Comandos, guarde el archivo JSFL en la carpeta Commands en la siguiente ubicación:

Sistema operativo	Ubicación
Windows Vista	*unidad de arranque*\Users*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\Commands
Windows XP	*unidad de arranque*\Documents and Settings*usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\Commands
Mac OS X	Macintosh HD/Usuarios/*nombre de usuario*/Librería/Application Support/Adobe/Flash CS4/*idioma*/Configuration/Commands

- Para los scripts que aparecen como herramientas ampliables en el panel Herramienta, guarde el archivo JSFL en la carpeta Tools en la ubicación siguiente:

Sistema operativo	Ubicación
Windows Vista	*unidad de arranque*\Users*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\Tools
Windows XP	*unidad de arranque*\Documents and Settings*usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\Tools
Mac OS X	Macintosh HD/Usuarios/*nombre de usuario*/Librería/Application Support/Adobe/Flash CS4/*idioma*/Configuration/Tools

Si un archivo JSFL va acompañado de otros archivos, por ejemplo, de archivos XML, deben almacenarse en el mismo directorio que el archivo JSFL.

Ejecución de scripts

Existen varias formas de ejecutar scripts. En esta sección se explican las más frecuentes.

Para ejecutar un script que esté editando o visualizando en ese momento:

- Haga clic con el botón derecho (o clic con la tecla Comando pulsada en Macintosh) y seleccione Ejecutar script.
- Haga clic en el icono Ejecutar script de la barra de herramientas de la ventana Script.

Esta opción permite ejecutar un script antes de guardarlo. También permite ejecutar un script incluso si no hay ningún archivo FLA abierto.

Para ejecutar un script que se encuentra en la carpeta Commands, siga uno de los estos procedimientos:

- En el entorno de edición, seleccione Comandos > *Nombre de script*.
- Utilice el método abreviado de teclado que haya asignado al script. Para asignar un método abreviado, utilice Edición > Métodos abreviados de teclado y seleccione Menú de comandos de dibujo en el menú desplegable. Expanda el nodo Comandos en el árbol de menús para ver una lista de los scripts disponibles.

Para ejecutar un script de comando que no se encuentra en la carpeta Commands, realice uno de los siguientes pasos:

- En el entorno de edición, seleccione Comandos > Ejecutar comando y, a continuación, seleccione el script que desea ejecutar.
- Dentro del script, utilice el comando `fl.runScript()`.
- En el sistema de archivos, haga doble clic en el archivo de script.

Para añadir al panel Herramientas una herramienta implementada en un archivo JSFL:

1. Copie en la carpeta de herramientas el archivo JSFL de la herramienta y los demás archivos asociados (consulte "Almacenamiento de archivos JSFL" en la página 2).
2. Seleccione Edición > Personalizar panel de herramientas (Windows) o Flash > Personalizar panel de herramientas (Macintosh).
3. Añada la herramienta a la lista de herramientas disponibles.
4. Haga clic en Aceptar.

Puede añadir comandos individuales de la API JavaScript a los archivos de ActionScript utilizando la función `MMExecute()`, que se describe en *Referencia del lenguaje y componentes ActionScript 3.0*. Sin embargo, la función `MMExecute()` sólo tiene efecto cuando se utiliza en el contexto de un elemento de la interfaz de usuario personalizada, como un inspector de propiedades de componentes o un panel SWF en el entorno de edición. Aunque se llamen desde ActionScript, los comandos API JavaScript no tienen efecto en Flash Player ni fuera del entorno de edición.

Para enviar un comando desde un script de ActionScript:

- Utilice la siguiente sintaxis (puede concatenar varios comandos en una cadena):

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

  También se puede ejecutar un script desde la línea de comandos.

Para ejecutar un script desde la línea de comandos de Windows:

- Utilice la siguiente sintaxis (añada información de la ruta según sea necesario):

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

Para ejecutar un script desde la aplicación "Terminal" de Macintosh:

- Utilice la siguiente sintaxis (añada información de la ruta según sea necesario):

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

  El comando `osascript` también puede ejecutar AppleScript en un archivo. Por ejemplo, podría incluir el siguiente texto en un archivo denominado myScript:

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

  A continuación, para ejecutar el script, puede utilizar este comando:

```
osascript myScript
```

Novedades de la API JavaScript

En Flash CS4, se han añadido determinados objetos, métodos y propiedades, mientras que otros han sido eliminados. A continuación resumimos estos cambios.

Si no ha utilizado la API JavaScript con anterioridad, puede que desee omitir esta sección e ir directamente a "Objetos de la API JavaScript" en la página 8.

Nuevos objetos

Los objetos siguientes aparecen por primera vez en Flash CS4:

Objeto presetPanel

Objeto presetItem

Objeto swfPanel

Nuevos métodos y propiedades

Los siguientes métodos y propiedades aparecen por primera vez en Flash CS4:

- Objeto BitmapItem
  - `bitmapItem.exportToFile()`
  - `bitmapItem.fileLastModifiedDate`
  - `bitmapItem.originalCompressionType`
  - `bitmapItem.sourceFileExists`
  - `bitmapItem.sourceFileIsCurrent`
  - `bitmapItem.sourceFilePath`
  - `bitmapItem.useDeblocking`
- Objeto Contour
  - `contour.fill`
- Objeto Document
  - `document.addNewPrimitiveOval()`
  - `document.addNewPrimitiveRectangle()`
  - `document.exportPublishProfileString()`
  - `document.externalLibraryPath`
  - `document.importPublishProfileString()`
  - `document.libraryPath`
  - `document.pathURI`
  - `document.rotate3DSelection`
  - `document.setStageVanishingPoint()`

- `document.setStageViewAngle()`
- `document.sourcePath`
- `document.translate3DCenter()`
- `document.translate3DSelection()`

- Objeto Edge
  - `edge.cubicSegmentIndex`
  - `edge.stroke`
- Objeto Fill
  - `fill.bitmapIsClipped`
  - `fill.bitmapPath`
- Objeto flash (fl)
  - `fl.externalLibraryPath`
  - `fl.flexSDKPath`
  - `fl.isFontInstalled()`
  - `fl.libraryPath`
  - `fl.presetPanel`
  - `fl.sourcePath`
  - `fl.swfPanels`
- Objeto FLfile
  - `FLfile.platformPathToURI()`
  - `FLfile.uriToPlatformPath()`
- Objeto fontItem
  - `fontItem.bitmap`
  - `fontItem.bold`
  - `fontItem.embedVariantGlyphs`
  - `fontItem.font`
  - `fontItem.isDefineFont4Symbol`
  - `fontItem.italic`
  - `fontItem.size`
- Objeto Shape
  - `shape.getCubicSegmentPoints()`
  - `shape.members`
  - `shape.numCubicSegments`
- Objeto SoundItem
  - `soundItem.exportToFile()`
  - `soundItem.fileLastModifiedDate`
  - `soundItem.originalCompressionType`

  - `soundItem.sourceFileExists`
  - `soundItem.sourceFileIsCurrent`
  - `soundItem.sourceFilePath`
- Objeto Text
  - `text.embedVariantGlyphs`
- Objeto Timeline
  - `timeline.getGuidelines()`
  - `timeline.setGuidelines()`
- Objeto VideoItem
  - `videoItem.exportToFLV()`
  - `videoItem.fileLastModifiedDate`
  - `videoItem.sourceFileExists`
  - `videoItem.sourceFileIsCurrent`

Otros cambios

La siguiente propiedad tiene un nuevo valor admitido en Flash CS4:

- `fill.style`

Los siguientes objetos, métodos y propiedades ya no están disponibles en Flash CS4:

- Objeto Project
- Objeto ProjectItem
- `fl.openProject()`
- `fl.closeProject()`
- `fl.createProject()`
- `fl.getProject()`
- Objeto Effect
- `configureEffect()`
- `executeEffect()`
- `removeEffect()`
- `fl.activeEffect`
- `fl.effects`
- `fl.enableImmediateUpdates()`
- `fl.reloadEffects()`

Objetos de la API JavaScript

En esta sección se ofrece un resumen de los objetos disponibles en la API JavaScript de Flash y cómo comenzar a trabajar con ellos. Todos los comandos JavaScript estándar también se encuentran disponibles al utilizar la API JavaScript.

En la siguiente tabla se describen brevemente todos los objetos de la API JavaScript. Los objetos aparecen en orden alfabético.

Objeto	Descripción
Objeto actionsPanel	El objeto actionsPanel representa el panel Acciones que se muestra actualmente.
Objeto BitmapInstance	El objeto BitmapInstance es una subclase del objeto Instance y representa un mapa de bits en un fotograma.
Objeto BitmapItem	El objeto BitmapItem hace referencia a un mapa de bits en la biblioteca de un documento. El objeto BitmapItem es una subclase del objeto Item.
Objeto CompiledClipInstance	El objeto CompiledClipInstance es una subclase del objeto Instance.
Objeto compilerErrors	El objeto compilerErrors representa el panel de errores del compilador. Se trata de una propiedad del objeto flash (fl.compilerErrors).
Objeto ComponentInstance	El objeto ComponentInstance es una subclase del objeto SymbolInstance y representa un componente en un fotograma.
Objeto componentsPanel	El objeto componentsPanel, que representa el panel Componentes, es una propiedad del objeto flash (`fl.componentsPanel`).
Objeto Contour	Un objeto Contour representa un trazado cerrado de bordes partidos en el límite de una forma.
Objeto Document	El objeto Document representa el escenario.
Objeto drawingLayer	Al objeto drawingLayer se puede acceder desde JavaScript como elemento secundario del objeto flash.
Objeto Edge	El objeto Edge representa un borde de una forma en el escenario.
Objeto Element	Todo lo que aparece en el escenario es de tipo Element.
Objeto Fill	El objeto Fill contiene todas las propiedades de la configuración de color de relleno del panel Herramientas o de una forma seleccionada.
Objeto Filter	El objeto Filter contiene todas las propiedades para todos los filtros.
Objeto flash (fl)	El objeto flash representa la aplicación Flash.
Objeto FLfile	El objeto FLfile permite escribir extensiones de Flash que pueden acceder, modificar y eliminar archivos y carpetas en el sistema de archivos local.
Objeto folderItem	El objeto folderItem es una subclase del objeto Item.
Objeto fontItem	El objeto fontItem es una subclase del objeto Item.
Objeto Frame	El objeto Frame representa fotogramas en la capa.
Objeto HalfEdge	Lado dirigido del borde de un objeto Shape.
Objeto Instance	El objeto Instance es una subclase del objeto Element.
Objeto Item	El objeto Item es una clase base abstracta.
Objeto Layer	El objeto Layer representa una capa en la línea de tiempo.
Objeto library	El objeto Library representa el panel Biblioteca.

Objeto	Descripción
Objeto Math	El objeto Math está disponible como propiedad de sólo lectura del objeto flash (`fl.Math`).
Objeto Matrix	El objeto Matrix representa una matriz de transformación.
Objeto outputPanel	El objeto outputPanel representa el panel Salida, que muestra información sobre solución de problemas, como errores de sintaxis. Se trata de una propiedad del objeto flash (`fl.outputPanel`).
Objeto Oval	El objeto Oval es una forma que se dibuja con la herramienta Óvalo. Para determinar si un elemento es un objeto Rectangle, utilice `shape.isOvalObject`.
Objeto Parameter	El acceso al tipo de objeto Parameter se realiza desde el conjunto `screen.parameters` (que corresponde al inspector de propiedades de la pantalla en la herramienta de edición de Flash) o el conjunto `componentInstance.parameters` (que corresponde al inspector de propiedades del componente en la herramienta de edición).
Objeto Path	El objeto Path define una secuencia de segmentos de línea (recta, curva o ambas) que suele emplearse para crear herramientas ampliables.
Objeto presetItem	El objeto presetItem representa un elemento (valor predefinido o carpeta) en el panel Configuración predefinida de movimiento.
Objeto presetPanel	El objeto presetPanel representa el panel Configuración predefinida de movimiento (Ventana > Configuración predefinida de movimiento). Se trata de una propiedad del objeto flash (`fl.presetPanel`).
Objeto Rectangle	El objeto Rectangle es una forma que se dibuja con la herramienta Rectángulo. Para determinar si un elemento es un objeto Rectangle, utilice `shape.isRectangleObject`.
Objeto Screen	El objeto Screen representa una pantalla única en un documento de diapositivas o formularios.
Objeto ScreenOutline	El objeto ScreenOutline representa el grupo de pantallas en un documento de diapositivas o formularios.
Objeto Shape	El objeto Shape es una subclase del objeto Element. El objeto Shape proporciona un control más preciso que las API de dibujo para manipular o crear geometría en el escenario.
Objeto SoundItem	El objeto SoundItem es una subclase del objeto Item. Representa un elemento de biblioteca utilizado para crear un sonido.
Objeto Stroke	El objeto Stroke contiene toda la configuración de un trazo, incluida la configuración personalizada.
Objeto swfPanel	El objeto swfPanel representa un panel SWF de ventana. Los paneles SWF de ventana son archivos SWF que implementan aplicaciones que se pueden ejecutar desde el entorno de edición de Flash. El conjunto de objetos swfPanel es una propiedad del objeto flash (`fl.swfPanels`).
Objeto SymbolInstance	El objeto SymbolInstance es una subclase del objeto Instance y representa un símbolo en un fotograma.
Objeto SymbolItem	El objeto SymbolItem es una subclase del objeto Item.
Objeto Text	El objeto Text representa un elemento de texto único en un documento.
Objeto TextAttrs	El objeto TextAttrs contiene todas las propiedades de texto que se pueden aplicar a una subselección. Este objeto es una subclase del objeto Text.
Objeto TextRun	El objeto TextRun representa una serie de caracteres que tienen atributos que coinciden con todas las propiedades del objeto TextAttrs.
Objeto Timeline	El objeto Timeline representa la línea de tiempo de Flash, a la que se puede acceder para el documento actual mediante `fl.getDocumentDOM().getTimeline()`.
Objeto ToolObj	Un objeto ToolObj representa una herramienta individual en el panel Herramientas.

Objeto	Descripción
Objeto Tools	Se puede acceder al objeto Tools desde el objeto Flash (`fl.tools`).
Objeto Vertex	El objeto Vertex forma parte de la estructura de datos de formas que contiene los datos de coordenadas.
Objeto VideoItem	El objeto VideoItem es una subclase del objeto Item.
Objeto XMLUI	El objeto XMLUI permite obtener y definir propiedades de un cuadro de diálogo XMLUI, así como aceptar o cancelar una.

El modelo de objetos de documento de Flash

El modelo de objetos de documento (DOM) de la API JavaScript de Flash se compone de una serie de funciones de nivel superior (consulte "Funciones y métodos de nivel superior" en la página 16) y dos objetos de nivel superior: FLfile y Flash (fl). Cada uno de estos objetos se encuentran disponibles en todo momento en un script porque siempre están presentes cuando se abre el entorno de edición de Flash. Para más información, consulte Objeto FLfile y Objeto flash (fl).

Para hacer referencia al objeto Flash, puede utilizar `flash` o `fl`. Por ejemplo, para cerrar todos los archivos FLA abiertos, puede utilizar cualquiera de las sentencias siguientes:

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

El objeto Flash contiene los siguientes objetos *objetos secundarios*:

Objeto	Modo de acceso
Objeto actionsPanel	Utilice `fl.actionsPanel` para acceder al objeto actionsPanel. Este objeto corresponde al panel Acciones en el entorno de edición de Flash.
Objeto compilerErrors	Utilice `fl.compilerErrors` para acceder al objeto compilerErrors. Este objeto corresponde al panel Errores de compilador en el entorno de edición de Flash.
Objeto componentsPanel	Utilice `fl.componentsPanel` para acceder al objeto componentsPanel. Este objeto corresponde al panel Componentes en el entorno de edición de Flash.
Objeto Document	Utilice `fl.documents` para recuperar un conjunto de todos los documentos abiertos; use `fl.documents[index]` para acceder a un documento concreto; utilice `fl.getDocumentDOM()` para acceder al documento activo (el documento seleccionado).
Objeto drawingLayer	Utilice `fl.drawingLayer` para acceder al objeto drawingLayer.
Objeto Math	Utilice `fl.Math` para acceder al objeto Math.
Objeto outputPanel	Utilice `fl.outputPanel` para acceder al objeto outputPanel. Este objeto corresponde al panel Salida en el entorno de edición de Flash.
Objeto presetPanel	Utilice `fl.presetPanel` para acceder al objeto presetPanel. Este objeto corresponde al panel Configuración predefinida de movimiento (Ventana > Configuración predefinida de movimiento).
Objeto swfPanel	Utilice `fl.swfPanels` para acceder a un conjunto de objetos swfPanel. Estos objetos corresponden a los paneles SWF de la ventana.
Objeto Tools	Utilice `fl.tools` para acceder a un conjunto de objetos Tools.
Objeto XMLUI	Utilice `fl.xmlui` para acceder a un objeto XMLUI (Interfaz de usuario XML). El objeto XMLUI permite obtener y establecer las propiedades de un cuadro de diálogo XMLUI.

El objeto Document

Una propiedad importante del objeto Flash de nivel superior es la propiedad `fl.documents`. Esta propiedad contiene un conjunto de objetos Document donde cada uno representa uno de los archivos FLA abiertos actualmente en el entorno de edición. Las propiedades de cada objeto Document representan la mayoría de los elementos que puede contener un archivo FLA. Por tanto, gran parte del DOM se compone de objetos y propiedades secundarios del objeto Document. Para más información, consulte Objeto Document.

Para hacer referencia al primer documento abierto, por ejemplo, utilice la sentencia `flash.documents[0]` o `fl.documents[0]`. El primer documento es el primer documento de Flash que se abrió en la sesión actual en el entorno de edición. Cuando se cierra el primer documento que se abrió, se reducen los índices de los otros documentos abiertos.

Para buscar el índice de un determinado documento, utilice `flash.findDocumentIndex(`*nameOfDocument*`)` o `fl.findDocumentIndex(`*nameOfDocument*`)`. Consulte `fl.findDocumentIndex()`.

Para acceder al documento seleccionado actualmente, utilice la sentencia `flash.getDocumentDOM()` o `fl.getDocumentDOM()`. Consulte `fl.getDocumentDOM()`. El segundo es la sintaxis empleada en la mayoría de los ejemplos de este documento.

Para buscar un determinado documento en el conjunto `fl.documents`, repita a través del conjunto y pruebe en cada documento su propiedad `document.name`. Consulte `fl.documents` y `document.name`.

El acceso a todos los objetos del DOM que no figuran en la tabla anterior (consulte "El modelo de objetos de documento de Flash" en la página 10) se realiza desde el objeto Document. Por ejemplo, para acceder a la biblioteca de un documento, se emplea la propiedad `document.library`, que recupera un objeto Library:

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

Para acceder al conjunto de elementos de la biblioteca, utilice la propiedad `library.items`; cada elemento del conjunto es un objeto Item:

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

Para acceder a un determinado elemento de la biblioteca, deberá especificar un miembro del conjunto `library.items`:

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

En otras palabras, el objeto Library es un elemento secundario del objeto Document, y Item es un elemento secundario del objeto Library. Para más información, consulte `document.library`, Objeto library, library.items`library.items` y Objeto Item.

Especificación del destino de una acción

A menos que se indique lo contrario, los métodos afectan al enfoque o la selección actual. Por ejemplo, el script siguiente dobla el tamaño de la selección actual porque no se especifica ningún objeto concreto:

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

En algunos casos conviene que una acción se realice específicamente sobre el elemento seleccionado actualmente en el documento de Flash. Para ello, utilice el conjunto que contiene la propiedad `document.selection` (consulte `document.selection`). El primer elemento del conjunto representa el elemento seleccionado actualmente, como se muestra en el ejemplo siguiente:

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

El script siguiente dobla el tamaño del primer elemento en el escenario almacenado en el conjunto de elementos, en lugar de la selección actual:

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

También puede realizar acciones como establecer bucles a través de todos los elementos del escenario o incrementar la anchura y altura con un valor determinado, como se muestra en el ejemplo siguiente:

```
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;
    }
```

Resumen de la estructura del DOM

La siguiente lista muestra la estructura del DOM en formato de esquema. Los números al comienzo de cada línea representan el nivel de un objeto. Por ejemplo, un objeto precedido de “03” es un elemento secundario del siguiente objeto más alto “02”, que, a su vez, es un elemento secundario del siguiente objeto más alto “01”.

En algunos casos, un objeto está disponible especificando una propiedad de su objeto principal. Por ejemplo, la propiedad `document.timelines` contiene un conjunto de objetos Timeline. Estas propiedades se incluyen en el siguiente esquema.

Algunos objetos son subclases de otros objetos, en lugar de ser elementos secundarios de otros objetos. Un objeto que es una subclase de otro objeto tiene métodos y/o propiedades propias además de los métodos y propiedades del objeto principal (la superclase). Las subclases comparten el mismo nivel en la jerarquía que su superclase. Por ejemplo, el objeto Item es una superclase del objeto BitmapItem. Estas relaciones se ilustran en el siguiente ejemplo:

```
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
  03 ScreenOutline object
    04 Screen object (screenOutline.screens array)
      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
```

Implementaciones de muestra

Hay disponibles varias implementaciones JSFL de ejemplo para Adobe Flash CS4 Professional. Puede revisar e instalar estos archivos para familiarizarse con la API de JavaScript. Las muestras se encuentran en la carpeta Samples/ExtendingFlash en el archivo Samples.zip ubicado en www.adobe.com/go/learn_fl_samples_es.

Comando de forma de muestra

El script de ejemplo de la API JavaScript denominado Shape.jsfl se ubica en la carpeta ExtendingFlash/Shape (consulte “Implementaciones de muestra” más arriba). Este script muestra información sobre los contornos de la forma en el panel Salida.

Para instalar y ejecutar el script Shape:

1 Copie el archivo Shape.jsfl en la carpeta Configuration/Commands (consulte "Almacenamiento de archivos JSFL" en la página 2).

2 En un documento de Flash (archivo FLA), seleccione un objeto Shape.

3 Seleccione Comandos > Forma para ejecutar el script.

Comando de filtros de obtener y establecer de muestra

El script de ejemplo de la API JavaScript denominado GetSet.jsfl se ubica en la carpeta ExtendingFlash/filtersGetSet (consulte "Implementaciones de muestra" más arriba). Este script añade filtros a un objeto seleccionado y muestra información sobre los filtros que se añaden en el panel Salida.

Para instalar y ejecutar el script filtersGetSet:

1 Copie el archivo filtersGetSet.jsfl en la carpeta Configuration/Commands (consulte "Almacenamiento de archivos JSFL" en la página 2).

2 En un documento de Flash (archivo FLA), seleccione un texto, clip de película u objeto Button.

3 Seleccione Comandos > filtersGetSet para ejecutar el script.

Herramienta PolyStar de muestra

El script de muestra de la API JavaScript denominado PolyStar.jsfl se ubica en la carpeta ExtendingFlash/PolyStar (consulte "Implementaciones de muestra" más arriba).

PolyStar.jsfl replica la herramienta PolyStar que se puede encontrar en el panel Herramientas de Flash. El script muestra cómo crear la herramienta PolyStar con la API JavaScript e incluye comentarios detallados que describen lo que está haciendo el código. Lea este archivo para conocer mejor cómo funciona la API JavaScript. También se recomienda que lea el archivo PolyStar.xml en el directorio Tools para conocer mejor cómo crear su propia herramienta.

Panel Trazar mapa de bits de muestra

Un conjunto de archivos denominado TraceBitmap.fla y TraceBitmap.swf se ubican en la carpeta ExtendingFlash/TraceBitmapPanel (consulte "Implementaciones de muestra" más arriba). Estos archivos muestran cómo diseñar y crear un panel para controlar las funciones de Flash. También muestran el uso de la función `MMExecute()` para llamar a los comandos JavaScript desde un script de ActionScript.

Para ejecutar la muestra TraceBitmap:

1 Si se está ejecutando Flash, salga de la aplicación.

2 Copie el archivo TraceBitmap.swf en la carpeta WindowSWF, que es un subdirectorio de la carpeta Configuration (consulte "Almacenamiento de archivos JSFL" en la página 2). Por ejemplo, en Windows, la carpeta se encuentra en *unidad de arranque*\Documents and Settings*usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\WindowSWF.

3 Inicie Flash.

4 Cree o abra un documento de Flash (archivo FLA) e importe una imagen de mapa de bits o JPEG al archivo.

Puede utilizar el archivo flower.jpg incluido en la carpeta TraceBitmapPanel u otra imagen que elija.

5 Con la imagen importada seleccionada, elija Ventana > Otros paneles > Trazar mapa de bits.

6 Haga clic en Enviar.

La imagen se convierte en un grupo de formas.

DLL de muestra

Una implementación de DLL de muestra se ubica en la carpeta ExtendingFlash/dllSampleComputeSum (consulte "Implementaciones de muestra" más arriba). Para obtener más información sobre la creación de las DLL, consulte "Extensibilidad de nivel C" en la página 536.

Capítulo 2: Funciones y métodos de nivel superior

Información sobre esta sección

En esta sección se describen las funciones y los métodos de nivel superior disponibles cuando se utiliza la interfaz de programación de aplicaciones JavaScript (API JavaScript) de Adobe Flash. Para obtener información sobre dónde almacenar los archivos de la API JavaScript, consulte "Almacenamiento de archivos JSFL" en la página 2.

Métodos globales

Los métodos siguientes se pueden llamar desde cualquier script de la API JavaScript:

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

Herramientas extensibles

Las funciones siguientes están disponibles en scripts que crean herramientas ampliables:

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

activate()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando se activa la herramienta ampliable (es decir, cuando se selecciona la herramienta en el panel Herramientas). Utilice esta función para realizar las tareas de inicialización que necesita la herramienta.

Ejemplo

El ejemplo siguiente establece el valor de `tools.activeTool` cuando se selecciona la herramienta ampliable en el panel Herramientas:

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

Véase también

`tools.activeTool`

alert()

Disponibilidad

Flash MX 2004

Uso

```
alert ( alertText )
```

Parámetros

`alertText` Una cadena que especifica el mensaje que desea mostrar en el cuadro de diálogo Alerta.

Valor devuelto

Ninguno.

Descripción

Método; muestra una cadena en un cuadro de diálogo modal Alerta, junto con un botón Aceptar.

Ejemplo

El ejemplo siguiente muestra el mensaje “Process Complete” en un cuadro de diálogo Alerta:

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

Véase también

`confirm()`, `prompt()`

configureTool()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando se abre Flash y se carga la herramienta ampliable en el panel Herramientas. Utilice esta función para definir la información que Flash necesita conocer sobre la herramienta.

Ejemplo

Los ejemplos siguientes muestran dos implementaciones posibles de esta función:

```
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()

Disponibilidad

Flash 8

Uso

```
confirm ( strAlert )
```

Parámetros

`strAlert` Una cadena que especifica el mensaje que desea mostrar en el cuadro de diálogo Alerta.

Valor devuelto

Un valor booleano: `true` si el usuario hace clic en Aceptar; `false` si el usuario hace clic en Cancelar.

Descripción

Método; muestra una cadena en un cuadro de diálogo modal Alerta, junto con botones Aceptar y Cancelar.

Nota: *si no hay documentos (archivos FLA) abiertos, este método falla con una condición de error.*

Ejemplo

El ejemplo siguiente muestra el mensaje "Sort data?" en un cuadro de diálogo Alerta:

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

Véase también

`alert()`, `prompt()`

deactivate()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable se desactiva (es decir, cuando la herramienta activa cambia de esta herramienta a otra). Utilice esta función para realizar la limpieza que necesita la herramienta.

Ejemplo

El siguiente ejemplo muestra un mensaje en el panel Salida cuando la herramienta se vuelve inactiva:

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

keyDown()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y el usuario presiona una tecla. El script debe llamar a `tools.getKeyDown()` para determinar qué tecla se ha presionado.

Ejemplo

El ejemplo siguiente muestra información sobre qué tecla se ha presionado cuando la herramienta ampliable está activa y el usuario presiona una tecla.

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

Véase también

`keyUp()`, `tools.getKeyDown()`

keyUp()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y se suelta una tecla.

Ejemplo

El ejemplo siguiente muestra un mensaje en el panel Salida cuando la herramienta ampliable está activa y se suelta una tecla.

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

Véase también

keyDown()

mouseDoubleClick()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y se hace doble clic en el botón del ratón en el escenario.

Ejemplo

El ejemplo siguiente muestra un mensaje en el panel Salida cuando la herramienta ampliable está activa y se hace doble clic en el botón del ratón.

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

mouseDown()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`pt` Un punto que especifica la ubicación del ratón cuando se presiona el botón. Se transfiere a la función cuando se presiona el botón del ratón. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y se presiona el botón del ratón mientras el puntero se encuentra sobre el escenario.

Ejemplo

Los siguientes ejemplos muestran cómo se puede emplear esta función cuando la herramienta ampliable está activa. El primer ejemplo muestra un mensaje en el panel Salida de que se ha presionado el botón del ratón. El segundo ejemplo muestra las coordenadas*x*e*y* de la ubicación del ratón cuando se presionó el botón.

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

mouseMove()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`pt` Un punto que especifica la ubicación actual del ratón. Se transfiere a la función cuando se mueve el ratón, realizando un seguimiento de la ubicación del ratón. Si el escenario se encuentra en modo de edición o de edición en contexto, las coordenadas del punto serán relativas al objeto que se está editando. En caso contrario, las coordenadas del punto serán relativas al escenario. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Función; se llama cada vez que la herramienta ampliable está activa y el ratón se mueve sobre un punto especificado en el escenario. El botón del ratón puede estar presionado o no presionado.

Ejemplo

Los ejemplos siguientes muestran el uso de esta función. El primer ejemplo muestra un mensaje en el panel Salida de que se está moviendo el ratón. El segundo ejemplo muestra las coordenadas*x*e*y* de la ubicación del ratón a medida que se mueve.

```
function mouseMove() {
    fl.trace("moving");
}

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

mouseUp()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y se suelta el botón del ratón después de presionarse en el escenario.

Ejemplo

El ejemplo siguiente muestra un mensaje en el panel Salida cuando la herramienta ampliable está activa y se suelta el botón del ratón.

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

notifySettingsChanged()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y el usuario cambia sus opciones en el inspector de propiedades. Puede utilizar la propiedad `tools.activeTool` para consultar los valores actuales de las opciones (consulte `tools.activeTool`).

Ejemplo

El ejemplo siguiente muestra un mensaje en el panel Salida cuando la herramienta ampliable está activa y el usuario cambia sus opciones en el inspector de propiedades.

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

prompt()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`promptMsg` Una cadena que se mostrará en el cuadro de diálogo Mensaje (limitado a 256 caracteres en Mac OS X).

`text` Una cadena opcional que se mostrará como valor predeterminado para el campo de texto.

Valor devuelto

La cadena que el usuario haya escrito si éste hace clic en Aceptar; `null` si hace clic en Cancelar.

Descripción

Método; muestra un mensaje y texto opcional en un cuadro de diálogo modal Alerta, junto con botones Aceptar y Cancelar.

Ejemplo

El ejemplo siguiente pide al usuario que introduzca un nombre de usuario. Si el usuario escribe un nombre y hace clic en Aceptar, el nombre aparece en el panel Salida.

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

Véase también

`alert()`, `confirm()`

setCursor()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Función; se llama cuando la herramienta ampliable está activa y se mueve el ratón, para permitir que el script establezca punteros personalizados. El script debe llamar a `tools.setCursor()` para especificar el puntero que se va a utilizar. Para obtener una lista de qué punteros corresponden a qué valores enteros, consulte `tools.setCursor()`.

Ejemplo

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

Capítulo 3: Objeto actionsPanel

Disponibilidad
Flash CS3 Professional

Descripción
El objeto actionsPanel, que representa el panel Acciones mostrado actualmente, es una propiedad del objeto Flash (consulte fl.actionsPanel).

Resumen de métodos
Pueden emplearse los métodos siguientes con el objeto actionsPanel:

Método	Descripción
`actionsPanel.getClassForObject()`	Devuelve la clase de una variable especificada.
`actionsPanel.getScriptAssistMode()`	Especifica si se activa el modo de asistente de script.
`actionsPanel.getSelectedText()`	Devuelve el texto seleccionado actualmente en el panel Acciones.
`actionsPanel.getText()`	Devuelve el texto en el panel Acciones.
`actionsPanel.hasSelection()`	Especifica si hay texto seleccionado actualmente en el panel Acciones.
`actionsPanel.replaceSelectedText()`	Reemplaza el texto seleccionado actualmente por el especificado.
`actionsPanel.setScriptAssistMode()`	Activa o desactiva el modo de asistente de script.
`actionsPanel.setSelection()`	Selecciona un conjunto de caracteres especificado en el panel Acciones.
`actionsPanel.setText()`	Borra cualquier texto del panel Acciones y añade el especificado.

actionsPanel.getClassForObject()

Disponibilidad
Flash CS3 Professional

Uso
```
actionsPanel.getClassForObject(ASvariableName)
```

Parámetros
`ASvariableName` Una cadena que representa el nombre de una variable de ActionScript.

Valor devuelto
Una cadena que representa la clase de la que es miembro *ASvariableName*.

Descripción
Método; devuelve la clase de la variable especificada, que deberá definirse en el panel Acciones mostrado actualmente. Además, el cursor o texto seleccionado en el panel Acciones deberá estar situado después de la definición de la variable.

Ejemplo

El ejemplo siguiente muestra la clase asignada a la variable myVar, si el cursor se coloca después de la sentencia ar myVar:ActivityEvent; en el panel Acciones.

```
// 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()

Disponibilidad

Flash CS3 Professional

Uso

```
actionsPanel.getScriptAssistMode()
```

Parámetros

Ninguno.

Valor devuelto

Un valor booleano que especifica si el modo de asistente de script está activado (`true`) o no (`false`).

Descripción

Método; especifica si se activa el modo de asistente de script.

Ejemplo

El ejemplo siguiente muestra un mensaje si no se activa el modo de asistente de script.

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

Véase también

`actionsPanel.setScriptAssistMode()`

actionsPanel.getSelectedText()

Disponibilidad

Flash CS3 Professional

Uso

```
actionsPanel.getSelectedText()
```

Parámetros
Ninguno.

Valor devuelto
Una cadena que contiene el texto seleccionado actualmente en el panel Acciones.

Descripción
Método; devuelve el texto seleccionado actualmente en el panel Acciones.

Ejemplo
El ejemplo siguiente muestra el texto seleccionado actualmente en el panel Acciones.

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

Véase también
`actionsPanel.getText()`, `actionsPanel.hasSelection()`, `actionsPanel.replaceSelectedText()`, `actionsPanel.setSelection()`

actionsPanel.getText()

Disponibilidad
Flash CS3 Professional

Uso
```
actionsPanel.getText()
```

Parámetros
Ninguno.

Valor devuelto
Una cadena que contiene todo el texto en el panel Acciones.

Descripción
Método; devuelve el texto en el panel Acciones.

Ejemplo
El ejemplo siguiente muestra el texto del panel Acciones.

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

Véase también
`actionsPanel.getSelectedText()`, `actionsPanel.setText()`

actionsPanel.hasSelection()

Disponibilidad
Flash CS3 Professional

Uso
`actionsPanel.hasSelection()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano que especifica si hay texto seleccionado en el panel Acciones (`true`) o no (`false`).

Descripción
Método; especifica si hay texto seleccionado actualmente en el panel Acciones.

Ejemplo
El ejemplo siguiente muestra el texto seleccionado actualmente en el panel Acciones. Si no hay texto seleccionado, muestra todo el texto en el panel Acciones.

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

Véase también
`actionsPanel.getSelectedText()`, `actionsPanel.getText()`, `actionsPanel.replaceSelectedText()`, `actionsPanel.setSelection()`

actionsPanel.replaceSelectedText()

Disponibilidad
Flash CS3 Professional

Uso
`actionsPanel.replaceSelectedText(replacementText)`

Parámetros
`replacementText` Una cadena que representa el texto que reemplazará al texto seleccionado en el panel Acciones.

Valor devuelto
Un valor booleano de `true` si se encuentra el panel Acciones, y de `false` en caso contrario.

Descripción

Método; reemplaza el texto seleccionado actualmente por el texto especificado en *replacementText*. Si *replacementText* contiene más caracteres que el texto seleccionado, los caracteres que siguen al texto seleccionado seguirán ahora a *replacementText*; es decir, no se sobrescriben.

Ejemplo

El ejemplo siguiente reemplaza el texto seleccionado actualmente en el panel Acciones.

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

Véase también

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

actionsPanel.setScriptAssistMode()

Disponibilidad

Flash CS3 Professional

Uso

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

Parámetros

`bScriptAssist` Un valor booleano que especifica si se activa o desactiva el modo de asistente de script.

Valor devuelto

Un valor booleano que especifica si se activó o desactivó correctamente el asistente de script.

Descripción

Método; activa o desactiva el modo de asistente de script.

Ejemplo

El ejemplo siguiente cambia el estado del modo de asistente 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());
```

Véase también

`actionsPanel.getScriptAssistMode()`

actionsPanel.setSelection()

Disponibilidad

Flash CS3 Professional

Uso

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

Parámetros

`startIndex` Un entero basado en cero que especifica el primer carácter que se va a seleccionar.

`numberOfChars` Un entero que especifica cuántos caracteres se van a seleccionar.

Valor devuelto

Un valor booleano que especifica si los caracteres solicitados se pueden seleccionar (`true`) o no (`false`).

Descripción

Método; selecciona un conjunto de caracteres especificado en el panel Acciones.

Ejemplo

El ejemplo siguiente reemplaza los caracteres "2006" en el panel Acciones por el texto especificado.

```
// 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");
```

Véase también

`actionsPanel.getSelectedText()`, `actionsPanel.hasSelection()`, `actionsPanel.replaceSelectedText()`

actionsPanel.setText()

Disponibilidad

Flash CS3 Professional

Uso

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

Parámetros

`replacementText` Una cadena que representa el texto que se reemplazará en el panel Acciones.

Valor devuelto

Un valor booleano de `true` si el texto especificado se colocó en el panel Acciones y de `false` en caso contrario.

Descripción

Método; borra todo el texto del panel Acciones y añade el especificado en *replacementText*.

Ejemplo

El ejemplo siguiente reemplaza todo el texto que hay actualmente en el panel Acciones por el texto especificado.

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

Véase también

`actionsPanel.getText()`, `actionsPanel.replaceSelectedText()`

Capítulo 4: Objeto BitmapInstance

Herencia Objeto Element > Objeto Instance > Objeto BitmapInstance

Disponibilidad
Flash MX 2004

Descripción
El objeto BitmapInstance es una subclase del objeto Instance y representa un mapa de bits en un fotograma (consulte Objeto Instance).

Resumen de métodos
Además de los métodos del Objeto Instance, puede emplear los métodos siguientes con el objeto BitmapInstance:

Método	Descripción
`bitmapInstance.getBits()`	Permite crear efectos de mapa de bits tomando los bits del mapa, manipulándolos y devolviéndolos a Flash.
`bitmapInstance.setBits()`	Define los bits de un elemento de mapa de bits existente.

Resumen de propiedades
Además de las propiedades del Objeto Instance, puede utilizar las siguientes propiedades con el objeto BitmapInstance:

Propiedad	Descripción
`bitmapInstance.hPixels`	De sólo lectura; un entero que representa la anchura del mapa de bits, en píxeles.
`bitmapInstance.vPixels`	De sólo lectura; un entero que representa la altura del mapa de bits, en píxeles.

bitmapInstance.getBits()

Disponibilidad
Flash MX 2004

Uso
```
bitmapInstance.getBits()
```

Parámetros
Ninguno.

Valor devuelto

Un objeto que contiene las propiedades `width`, `height`, `depth`, `bits` y, si el mapa de bits tiene una tabla de colores, `cTab`. El elemento `bits` es un conjunto de bytes. El elemento `cTab` es un conjunto de valores de color con el formato `"#RRGGBB"`. El conjunto tiene la misma longitud que la tabla de colores.

El conjunto de bytes sólo tiene sentido cuando una DLL o biblioteca compartida hace referencia a él. Sólo suele utilizarse para crear un efecto o una herramienta ampliable. Para más información sobre la creación de DLL para su uso con JavaScript de Flash, consulte "Extensibilidad de nivel C" en la página 536.

Descripción

Método; permite crear efectos de mapa de bits tomando los bits del mapa, manipulándolos y devolviéndolos a Flash.

Ejemplo

El código siguiente crea una referencia al objeto seleccionado actualmente; comprueba si el objeto es un mapa de bits y traza la altura, anchura y la profundidad en bits del mapa:

```
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);
}
```

Véase también

`bitmapInstance.setBits()`

bitmapInstance.hPixels

Disponibilidad

Flash MX 2004

Uso

`bitmapInstance.hPixels`

Descripción

Propiedad de sólo lectura; un entero que representa la anchura del mapa de bits, es decir, el número de píxeles en la dimensión horizontal.

Ejemplo

El código siguiente recupera la anchura del mapa de bits en píxeles:

```
// 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;
}
```

Véase también

`bitmapInstance.vPixels`

bitmapInstance.setBits()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

bitmap Un objeto que contiene las propiedades `height`, `width`, `depth`, `bits` y `cTab`. Las propiedades `height`, `width` y `depth` son enteros. La propiedad `bits` es un conjunto de bytes. La propiedad `cTab` sólo es necesaria para mapas de bits con una profundidad en bits de 8 o menos y es una cadena que representa un valor de color con el formato `"#RRGGBB"`.

Nota: *el conjunto de bytes sólo tiene sentido cuando una biblioteca externa hace referencia a él. Sólo suele utilizarse para crear un efecto o una herramienta ampliable.*

Valor devuelto

Ninguno.

Descripción

Método; define los bits de un elemento de mapa de bits existente. Permite crear efectos de mapa de bits tomando los bits del mapa, manipulándolos y devolviéndolos a Flash.

Ejemplo

El código siguiente comprueba si la selección actual es un mapa de bits y, a continuación, establece la altura del mapa de bits en 150 píxeles:

```
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);
}
```

Véase también

`bitmapInstance.getBits()`

bitmapInstance.vPixels

Disponibilidad

Flash MX 2004

Uso

`bitmapInstance.vPixels`

Descripción

Propiedad de sólo lectura; un entero que representa la altura del mapa de bits, es decir, el número de píxeles en la dimensión vertical.

Ejemplo

El código siguiente obtiene la altura del mapa de bits en píxeles:

```
// 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;
}
```

Véase también

`bitmapInstance.hPixels`

Capítulo 5: Objeto BitmapItem

Herencia Objeto Item > Objeto BitmapItem

Disponibilidad
Flash MX 2004

Descripción
Un objeto BitmapItem hace referencia a un mapa de bits en la biblioteca de un documento. El objeto BitmapItem es una subclase del objeto Item (consulte Objeto Item).

Resumen de propiedades
Además de las propiedades del Objeto Item, el objeto BitmapItem tiene las siguientes propiedades:

Propiedad	Descripción
`bitmapItem.allowSmoothing`	Un valor booleano que especifica si se permite el suavizado de un mapa de bits.
`bitmapItem.compressionType`	Cadena que determina el tipo de compresión de imagen que se aplica al mapa de bits.
`bitmapItem.fileLastModifiedDate`	Número de segundos que han transcurrido entre el 1 de enero de 1970 y la fecha de modificación del archivo original.
`bitmapItem.originalCompressionType`	Especifica si el elemento se importó como archivo jpeg.
`bitmapItem.sourceFileExists`	Especifica si el archivo importado en la biblioteca sigue existiendo en su ubicación original.
`bitmapItem.sourceFileIsCurrent`	Especifica si la fecha de modificación del archivo en el elemento de la biblioteca coincide con la fecha de modificación en disco del archivo importado.
`bitmapItem.sourceFilePath`	La ruta y el nombre del archivo importado en la biblioteca.
`bitmapItem.useDeblocking`	Especifica si el desbloqueo está activado.
`bitmapItem.useImportedJPEGQuality`	Valor booleano que especifica si se utiliza la calidad JPEG importada predeterminada.

Resumen de métodos
Además de las propiedades del Objeto Item, el objeto BitmapItem tiene los siguientes métodos:

Método	Descripción
`bitmapItem.exportToFile()`	Exporta el elemento especificado a un archivo PNG o JPG.

bitmapItem.allowSmoothing

Disponibilidad
Flash MX 2004

Uso

```
bitmapItem.allowSmoothing
```

Descripción

Propiedad; un valor booleano que especifica si se permite el suavizado de un mapa de bits (`true`) o no (`false`).

Ejemplo

El código siguiente define la propiedad `allowSmoothing` del primer elemento de la biblioteca del documento actual como `true`:

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

bitmapItem.compressionType

Disponibilidad

Flash MX 2004

Uso

```
bitmapItem.compressionType
```

Descripción

Propiedad; una cadena que determina el tipo de compresión de imagen que se aplica al mapa de bits. Los valores aceptables son `"photo"` o `"lossless"`. Si el valor de `bitmapItem.useImportedJPEGQuality` es `false`, `"photo"` corresponde a JPEG con una calidad de 0 a 100; si `bitmapItem.useImportedJPEGQuality` es `true`, `"photo"` corresponde a JPEG con un valor de calidad de documento predeterminada. El valor `"lossless"` corresponde al formato GIF o PNG (consulte bitmapItem.useImportedJPEGQuality).

Ejemplo

El código siguiente define la propiedad `compressionType` del primer elemento de la biblioteca del documento actual como `"photo"`:

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

bitmapItem.exportToFile()

Disponibilidad

Flash CS4 Professional

Uso

```
bitmapItem.exportToFile(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta y el nombre del archivo exportado.

Valor devuelto

Un valor booleano de `true` si se exporta correctamente el archivo, y de `false` en caso contrario.

Descripción

Método; exporta el elemento especificado a un archivo PNG o JPG.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de mapa de bits, el siguiente código lo exporta como un archivo JPG:

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

bitmapItem.fileLastModifiedDate

Disponibilidad

Flash CS4 Professional

Uso

```
bitmapItem.fileLastModifiedDate
```

Descripción

Propiedad de sólo lectura; una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la fecha de modificación del archivo original en el momento en que se importó a la biblioteca. Si el archivo ya no existe, el valor es "00000000".

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de mapa de bits, el siguiente código muestra un número hexadecimal, tal como se describe más arriba.

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

Véase también

`bitmapItem.sourceFileExists`, `bitmapItem.sourceFileIsCurrent`, `bitmapItem.sourceFilePath`, `FLfile.getModificationDate()`

bitmapItem.originalCompressionType

Disponibilidad

Flash CS4 Professional

Uso

```
bitmapItem.originalCompressionType
```

Descripción

Propiedad de sólo lectura; una cadena que especifica si el elemento especificado se ha importado como un archivo JPEG. Los valores posibles de esta propiedad son “photo” (para archivos JPEG) y “lossless” (para tipos de archivos sin compresión, como GIF y PNG).

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de mapa de bits, el siguiente código muestra "photo" si el archivo se importó en la biblioteca como archivo JPEG, o muestra "lossless" en caso contrario:

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

Véase también

`bitmapItem.compressionType`

bitmapItem.quality

Disponibilidad

Flash MX 2004

Uso

`bitmapItem.quality`

Descripción

Propiedad; un entero que especifica la calidad del mapa de bits. Para utilizar la calidad de documento predeterminada, especifique -1; en caso contrario, especifique un entero de 0 a 100. Sólo está disponible para compresión JPEG.

Ejemplo

El código siguiente define la propiedad `quality` del primer elemento de la biblioteca del documento actual como 65:

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

bitmapItem.sourceFileExists

Disponibilidad

Flash CS4 Professional

Uso

`bitmapItem.sourceFileExists`

Descripción

Propiedad de sólo lectura; un valor booleano `true` si el archivo importado en la biblioteca sigue existiendo en su ubicación original; `false` en caso contrario.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de mapa de bits, el siguiente código muestra "true" si el archivo se importó en la biblioteca sigue existiendo.

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

Véase también

`bitmapItem.sourceFileIsCurrent`,

`bitmapItem.sourceFilePath`

bitmapItem.sourceFileIsCurrent

Disponibilidad

Flash CS4 Professional

Uso

```
bitmapItem.sourceFileIsCurrent
```

Descripción

Propiedad de sólo lectura; un valor booleano `true` si la fecha de modificación del archivo en el elemento de la biblioteca coincide con la fecha de modificación en disco del archivo importado; `false` en caso contrario.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de mapa de bits, el siguiente código muestra "true" si el archivo importado no se ha modificado en el disco desde su importación:

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

Véase también

`bitmapItem.fileLastModifiedDate`, `bitmapItem.sourceFilePath`

bitmapItem.sourceFilePath

Disponibilidad

Flash CS4 Professional

Uso

```
bitmapItem.sourceFilePath
```

Descripción

Propiedad de sólo lectura; una cadena, especificada como URI file:/// que especifica la ruta y el nombre del archivo importado en la biblioteca.

Ejemplo

El ejemplo siguiente muestra el nombre y la ruta del archivo de origen de todos los elementos de la biblioteca que sean del tipo `"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);
    }
}
```

Véase también

`bitmapItem.sourceFileExists`

bitmapItem.useDeblocking

Disponibilidad

Flash CS4 Professional

Uso

```
bitmapItem.useDeblocking
```

Descripción

Propiedad; un valor booleano que especifica si el desbloqueo está activado (`true`) o no (`false`).

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de mapa de bits, el siguiente código activa el desbloqueo del elemento:

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

bitmapItem.useImportedJPEGQuality

Disponibilidad

Flash MX 2004

Uso

```
bitmapItem.useImportedJPEGQuality
```

Descripción

Propiedad; un valor booleano que especifica si se utiliza la calidad JPEG importada predeterminada (`true`) o no (`false`). Sólo está disponible para compresión JPEG.

Ejemplo

El código siguiente define la propiedad `useImportedJPEGQuality` del primer elemento de la biblioteca del documento actual como `true`:

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

Capítulo 6: Objeto CompiledClipInstance

Herencia Objeto Element > Objeto Instance > Objeto CompiledClipInstance

Disponibilidad
Flash MX 2004

Descripción
El objeto CompiledClipInstance es una subclase del objeto Instance. Es básicamente una instancia de un clip de película que se ha convertido en un elemento de biblioteca de clip compilado (consulte Objeto Instance).

Resumen de propiedades
Además de las propiedades del Objeto Instance, el objeto CompiledClipInstance tiene las propiedades siguientes:

Propiedad	Descripción
`compiledClipInstance.accName`	Una cadena que equivale al campo Nombre del panel Accesibilidad.
`compiledClipInstance.actionScript`	Una cadena que representa el ActionScript para esta instancia; equivale a `symbolInstance.actionScript`.
`compiledClipInstance.description`	Una cadena que equivale al campo Descripción del panel Accesibilidad.
`compiledClipInstance.forceSimple`	Un valor booleano que activa y desactiva los elementos secundarios del objeto para que sea accesible.
`compiledClipInstance.shortcut`	Una cadena que equivale al campo Método abreviado del panel Accesibilidad.
`compiledClipInstance.silent`	Un valor booleano que activa o desactiva la accesibilidad del objeto; equivale a la lógica inversa de la opción Hacer que el objeto sea accesible del panel Accesibilidad.
`compiledClipInstance.tabIndex`	Un entero que equivale al campo Índice de fichas del panel Accesibilidad.

compiledClipInstance.accName

Disponibilidad
Flash MX 2004

Uso
```
compiledClipInstance.accName
```

Descripción
Propiedad; una cadena que equivale al campo Nombre del panel Accesibilidad. Los lectores de pantalla identifican los objetos mediante la lectura del nombre en voz alta.

Ejemplo
El ejemplo siguiente obtiene y define el nombre de accesibilidad del primer objeto seleccionado:

```
// 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

Disponibilidad

Flash MX 2004

Uso

`compiledClipInstance.actionScript`

Descripción

Propiedad; una cadena que representa el ActionScript para esta instancia; equivale a `symbolInstance.actionScript`.

Ejemplo

El código siguiente asigna ActionScript a los elementos especificados:

```
// 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

Disponibilidad

Flash MX 2004

Uso

`compiledClipInstance.description`

Descripción

Propiedad; una cadena que equivale al campo Descripción del panel Accesibilidad. El lector de pantalla lee esta descripción.

Ejemplo

El ejemplo siguiente ilustra la obtención y definición de la propiedad `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

Disponibilidad
Flash MX 2004

Uso
`compiledClipInstance.forceSimple`

Descripción
Propiedad; un valor booleano que activa y desactiva los elementos secundarios del objeto para que sea accesible. Equivale a la lógica inversa de la opción Hacer que los objetos secundarios sean accesibles del panel Accesibilidad. Si `forceSimple` es `true`, equivale a la opción desactivada Hacer que los objetos secundarios sean accesibles. Si `forceSimple` es `false`, equivale a la opción activada Hacer que los objetos secundarios sean accesibles.

Ejemplo
El ejemplo siguiente ilustra la obtención y definición de la propiedad `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

Disponibilidad
Flash MX 2004

Uso
`compiledClipInstance.shortcut`

Descripción
Propiedad; una cadena que equivale al campo Método abreviado del panel Accesibilidad. El lector de pantalla lee este método abreviado. Esta propiedad no está disponible para campos de texto dinámicos.

Ejemplo
El ejemplo siguiente ilustra la obtención y definición de la propiedad `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

Disponibilidad
Flash MX 2004

Uso

```
compiledClipInstance.silent
```

Descripción

Propiedad; un valor booleano que activa o desactiva la accesibilidad del objeto; equivale a la lógica inversa de la opción Hacer que el objeto sea accesible del panel Accesibilidad. Es decir, si `silent` es `true`, estará desactivada la opción Hacer que el objeto sea accesible. Si `silent` es `false`, estará la activada la opción Hacer que el objeto sea accesible.

Ejemplo

El ejemplo siguiente ilustra la obtención y definición de la propiedad `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

Disponibilidad

Flash MX 2004

Uso

```
compiledClipInstance.tabIndex
```

Descripción

Propiedad; un entero que equivale al campo Índice de fichas del panel Accesibilidad. Crea un orden de tabulación con el que se accede a los objetos cuando el usuario presiona la tecla Tabulador.

Ejemplo

El ejemplo siguiente ilustra la obtención y definición de la propiedad `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;
```

Capítulo 7: Objeto compilerErrors

Disponibilidad
Flash CS3 Professional

Descripción
El objeto compilerErrors, que representa el panel de errores del compilador, es una propiedad del objeto flash (fl) a la que puede acceder `fl.compilerErrors` (consulte Objeto flash (fl)).

Resumen de métodos
Con el objeto compilerErrors pueden utilizarse los siguientes métodos:

Método	Descripción
`compilerErrors.clear()`	Borra el contenido del panel de errores del compilador.
`compilerErrors.save()`	Guarda el contenido del panel de errores del compilador en un archivo de texto local.

compilerErrors.clear()

Disponibilidad
Flash CS3 Professional

Uso
```
compilerErrors.clear()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; borra el contenido del panel de errores del compilador.

Ejemplo
El siguiente ejemplo borra el contenido del panel de errores del compilador:

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

Véase también
`compilerErrors.save()`

compilerErrors.save()

Disponibilidad

Flash CS3 Professional

Uso

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

Parámetros

`fileURI` Cadena que se expresa en forma de URI file:/// y que especifica el nombre del archivo guardado. Si *fileURI* ya existe y no ha especificado el valor `true` para *bAppendToFile*, *fileURI* se sobrescribirá sin advertencia.

`bAppendToFile` Valor booleano opcional que especifica si el contenido del panel de errores del compilador debe añadirse a *fileURI* (`true`) o no (`false`). El valor predeterminado es `false`.

`bUseSystemEncoding` Valor booleano opcional que especifica si debe guardarse el texto del panel de errores del compilador utilizando la codificación del sistema. Si este valor es `false` (el valor predeterminado), el texto del panel de errores del compilador se guardará empleando la codificación UTF-8, con caracteres de marca de orden de bytes al comienzo del texto. El valor predeterminado es `false`.

Valor devuelto

Ninguno.

Descripción

Método; guarda el contenido del panel de errores del compilador en un archivo de texto local.

Ejemplo

El siguiente ejemplo guarda el contenido del panel de errores del compilador en un archivo llamado errors.log situado en la carpeta C:\tests:

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

Véase también

`compilerErrors.clear()`

Capítulo 8: Objeto ComponentInstance

Herencia Objeto Element > Objeto Instance > Objeto SymbolInstance > Objeto ComponentInstance

Disponibilidad
Flash MX 2004

Descripción
El objeto ComponentInstance es una subclase del objeto SymbolInstance y representa un componente en un fotograma (consulte Objeto SymbolInstance).

Resumen de propiedades
Además de todas las propiedades del Objeto SymbolInstance, el objeto ComponentInstance tiene la propiedad siguiente:

Propiedad	Descripción
`componentInstance.parameters`	De sólo lectura; un conjunto de las propiedades de ActionScript 2.0 que son accesibles desde el inspector de propiedades o de componentes.

componentInstance.parameters

Disponibilidad
Flash MX 2004

Uso
```
componentInstance.parameters
```

Descripción
Propiedad de sólo lectura; un conjunto de las propiedades de ActionScript 2.0 que son accesibles desde el inspector de propiedades o de componentes. Consulte Objeto Parameter.

Ejemplo
El ejemplo siguiente ilustra la obtención y definición de la propiedad `parameters`:

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

Véase también
Objeto Parameter

Capítulo 9: Objeto componentsPanel

Disponibilidad
Flash MX 2004

Descripción
El objeto componentsPanel, que representa el panel Componentes, es una propiedad del objeto Flash (fl) y se puede acceder a él a través de `fl.componentsPanel` (consulte Objeto flash (fl)).

Resumen de métodos
Puede emplear los métodos siguientes con el objeto componentsPanel:

Método	Descripción
`componentsPanel.addItemToDocument()`	Añade el componente especificado al documento en la posición especificada.
`componentsPanel.reload()`	Actualiza la lista de componentes del panel Componentes.

componentsPanel.addItemToDocument()

Disponibilidad
Flash MX 2004

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

Parámetros
`position` Un punto (por ejemplo, `{x:0, y:100}`) que especifica la ubicación donde se añadirá el componente. Especifique *position* en relación con el punto central del componente, no el punto de registro del componente (también *punto de origen* o *punto cero*).

`categoryName` Una cadena que especifica el nombre de la categoría del componente (por ejemplo, `"Data"`). El panel Componentes muestra los nombres de categoría válidos.

`componentName` Una cadena que especifica el nombre del componente en la categoría especificada (por ejemplo, "`WebServiceConnector`"). El panel Componentes muestra los nombres de componente válidos.

Valor devuelto
Ninguno.

Descripción
Añade el componente especificado al documento en la posición especificada.

Ejemplo
El ejemplo siguiente ilustra algunas formas de utilizar este método:

```
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()

Disponibilidad

Flash 8

Uso

```
componentsPanel.reload()
```

Parámetros

Ninguno.

Valor devuelto

Un valor booleano de `true` si se actualiza la lista del panel Componentes, y de `false` en caso contrario.

Descripción

Método; actualiza la lista de componentes del panel Componentes.

Ejemplo

El ejemplo siguiente actualiza el panel Componentes:

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

Capítulo 10: Objeto Contour

Disponibilidad
Flash MX 2004

Descripción
Un objeto Contour representa un trazado cerrado de bordes partidos en el límite de una forma.

Resumen de métodos
Puede emplear el método siguiente con el objeto Contour:

Método	Descripción
`contour.getHalfEdge()`	Devuelve un Objeto HalfEdge en el contorno de la selección.

Resumen de propiedades
Puede emplear las propiedades siguientes con el objeto Contour:

Propiedad	Descripción
`contour.fill`	Un Objeto Fill.
`contour.interior`	Sólo lectura; el valor es `true` si el contorno encierra un área y `false` en caso contrario.
`contour.orientation`	De sólo lectura; un entero que indica la orientación del contorno.

contour.fill

Disponibilidad
Flash CS4 Professional

Uso
`contour.fill`

Descripción
Propiedad; un Objeto Fill.

Ejemplo
Suponiendo que haya un contorno con un relleno seleccionado, el siguiente ejemplo muestra el color de relleno del contorno en el panel Salida:

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

contour.getHalfEdge()

Disponibilidad

Flash MX 2004

Uso

```
contour.getHalfEdge()
```

Parámetros

Ninguno.

Valor devuelto

Un Objeto HalfEdge.

Descripción

Método; devuelve un Objeto HalfEdge en el contorno de la selección.

Ejemplo

Este ejemplo atraviesa todos los contornos de la forma seleccionada y muestra las coordenadas de los vértices del panel Salida:

```
// 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

Disponibilidad
Flash MX 2004

Uso
`contour.interior`

Descripción
Propiedad de sólo lectura; el valor es `true` si el contorno encierra un área y `false` en caso contrario.

Ejemplo
Este ejemplo atraviesa todos los contornos de la forma seleccionada y muestra el valor de la propiedad `interior` para cada contorno del panel Salida:

```
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

Disponibilidad
Flash MX 2004

Uso
`contour.orientation`

Descripción
Propiedad de sólo lectura; un entero que indica la orientación del contorno. El valor del entero es -1 si la orientación es en el sentido contrario a las agujas del reloj, 1 si es en el sentido de las agujas del reloj y 0 si es un contorno sin área.

Ejemplo
El ejemplo siguiente atraviesa todos los contornos de la forma seleccionada y muestra el valor de la propiedad `orientation` para cada contorno del panel Salida:

```
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();
```

Capítulo 11: Objeto Document

Disponibilidad

Flash MX 2004

Descripción

El objeto Document representa el escenario. Es decir, únicamente los archivos FLA se consideran documentos. Para volver al objeto Document para el documento actual, utilice `fl.getDocumentDOM()`.

Resumen de métodos

Puede emplear los métodos siguientes con el objeto Document:

Método	Descripción
`document.addDataToDocument()`	Almacena datos especificados con un documento.
`document.addDataToSelection()`	Almacena datos especificados con el objeto u objetos seleccionados.
`document.addFilter()`	Aplica un filtro a los objetos seleccionados.
`document.addItem()`	Añade un elemento desde cualquier documento o biblioteca abierta al objeto Document especificado.
`document.addNewLine()`	Añade un nuevo trazado entre dos puntos.
`document.addNewOval()`	Añade un nuevo objeto Oval al rectángulo de delimitación especificado.
`document.addNewPrimitiveOval()`	Añade un nuevo óvalo simple dentro de la delimitación especificada.
`document.addNewPrimitiveRectangle()`	Añade un nuevo rectángulo simple dentro de la delimitación especificada.
`document.addNewPublishProfile()`	Añade un nuevo perfil de publicación y lo convierte en el actual.
`document.addNewRectangle()`	Añade un nuevo rectángulo o rectángulo redondeado, ajustándolo a los límites especificados.
`document.addNewScene()`	Añade una nueva escena (Objeto Timeline) después de la escena seleccionada y la convierte en la seleccionada actualmente.
`document.addNewText()`	Inserta un nuevo campo de texto vacío.
`document.align()`	Alinea la selección.
`document.allowScreens()`	Utilice este método antes de usar la propiedad `document.screenOutline`.
`document.arrange()`	Organiza la selección en el escenario.
`document.breakApart()`	Realiza una operación de separación en la selección actual.
`document.canEditSymbol()`	Indica si el menú Editar símbolos y las funciones están activados.

Método	Descripción
`document.canRevert()`	Determina si puede utilizar correctamente el método `document.revert()` o `fl.revertDocument()`.
`document.canSaveAVersion()`	Determina si una versión del documento especificado se puede guardar en el servidor Version Cue.
`document.canTestMovie()`	Determina si puede utilizar correctamente el método `document.testMovie()`.
`document.canTestScene()`	Determina si puede utilizar correctamente el método `document.testScene()`.
`document.changeFilterOrder()`	Cambia el índice del filtro de la lista Filtro.
`document.clipCopy()`	Copia la selección actual desde el documento hasta el Portapapeles.
`document.clipCut()`	Corta la selección actual del documento y la escribe en el Portapapeles.
`document.clipPaste()`	Pega el contenido del Portapapeles en el documento.
`document.close()`	Cierra el documento especificado.
`document.convertLinesToFills()`	Convierte líneas en rellenos en los objetos seleccionados.
`document.convertToSymbol()`	Convierte el elemento o elemento de escenario seleccionados en un símbolo nuevo.
`document.crop()`	Utiliza el objeto de dibujo seleccionado en la parte superior para recortar todos los objetos de dibujo seleccionados por debajo.
`document.deleteEnvelope()`	Elimina la envoltura (recuadro de delimitación que contiene uno o varios objetos) del objeto seleccionado.
`document.deletePublishProfile()`	Elimina el perfil activo actualmente, si hay más de uno.
`document.deleteScene()`	Elimina la escena actual (Objeto Timeline) y, si la escena eliminada no era la última, establece la siguiente como el objeto Timeline actual.
`document.deleteSelection()`	Elimina la selección actual en el escenario.
`document.disableAllFilters()`	Desactiva todos los filtros en los objetos seleccionados.
`document.disableFilter()`	Desactiva el filtro especificado en la lista Filtros.
`document.disableOtherFilters()`	Desactiva todos los filtros salvo el que se encuentra en la posición especificada en la lista Filtros.
`document.distribute()`	Distribuye la selección.
`document.distributeToLayers()`	Realiza una operación de distribución en capas en la selección actual; equivale a seleccionar Distribuir en capas.
`document.documentHasData()`	Comprueba si el documento contiene datos persistentes con el nombre especificado.
`document.duplicatePublishProfile()`	Duplica el perfil activo y selecciona la versión duplicada.
`document.duplicateScene()`	Realiza una copia de la escena seleccionada, asignando un nombre exclusivo a la nueva escena y convirtiéndola en la actual.
`document.duplicateSelection()`	Duplica la selección en el escenario.

Método	Descripción
`document.editScene()`	Convierte la escena especificada en la escena seleccionada actualmente para editar.
`document.enableAllFilters()`	Activa todos los filtros de la lista Filtros para el objeto u objetos seleccionados.
`document.enableFilter()`	Activa el filtro especificado para el objeto u objetos seleccionados.
`document.enterEditMode()`	Cambia la herramienta de edición al modo de edición especificado por el parámetro.
`document.exitEditMode()`	Sale del modo de edición de símbolos y vuelve a seleccionar el siguiente nivel superior desde el modo de edición.
`document.exportPNG()`	Exporta el documento como uno o varios archivos PNG.
`document.exportPublishProfile()`	Exporta el perfil activo actualmente a un archivo XML.
`document.exportPublishProfileString()`	Devuelve una cadena que especifica el perfil indicado en formato XML.
`document.exportSWF()`	Exporta el documento en formato SWC de Flash.
`document.getAlignToDocument()`	Equivale a recuperar el valor del botón En escenario en el panel Alinear.
`document.getBlendMode()`	Devuelve una cadena que especifica el modo de mezcla para el objeto u objetos seleccionados.
`document.getCustomFill()`	Recupera el objeto de relleno de la forma seleccionada, o bien, el panel Herramientas y el inspector de propiedades, si se especifica.
`document.getCustomStroke()`	Recupera el objeto de trazo de la forma seleccionada, o bien, el panel Herramientas y el inspector de propiedades, si se especifica.
`document.getDataFromDocument()`	Recupera el valor de los datos especificados.
`document.getElementProperty()`	Obtiene la propiedad `Element` especificada para la selección actual.
`document.getElementTextAttr()`	Obtiene una propiedad `TextAttrs` específica de los objetos de texto seleccionados.
`document.getFilters()`	Devuelve un conjunto que contiene la lista de filtros aplicados al objeto u objetos seleccionados actualmente.
`document.getMetadata()`	Devuelve una cadena que contiene los metadatos XML asociados al documento.
`document.getMobileSettings()`	Devuelve la cadena transmitida a `document.setMobileSettings()`.
`document.getPlayerVersion()`	Devuelve una cadena que representa la versión del reproductor identificado del documento especificado.
`document.getSelectionRect()`	Obtiene el rectángulo de delimitación de la selección actual.
`document.getTextString()`	Obtiene el texto seleccionado actualmente.
`document.getTimeline()`	Recupera el Objeto Timeline actual en el documento.
`document.getTransformationPoint()`	Obtiene la ubicación del punto de transformación de la selección actual.

Método	Descripción
`document.group()`	Convierte la selección actual en un grupo.
`document.importFile()`	Importa un archivo en el documento.
`document.importPublishProfile()`	Importa un perfil desde un archivo.
`document.importPublishProfileString()`	Importa una cadena XML que representa un perfil de publicación y lo establece como perfil actual.
`document.importSWF()`	Importa un archivo SWF en el documento.
`document.intersect()`	Crea un objeto de dibujo de intersección a partir de todos los objetos de dibujo seleccionados.
`document.match()`	Iguala el tamaño de los objetos seleccionados.
`document.mouseClick()`	Ejecuta un clic de ratón desde la herramienta Selección.
`document.mouseDblClk()`	Ejecuta un doble clic de ratón desde la herramienta Selección.
`document.moveSelectedBezierPointsBy()`	Si la selección contiene como mínimo un trazado con al menos un punto Bézier seleccionado, este método mueve todos los puntos Bézier seleccionados en todos los trazados seleccionados con la cantidad especificada.
`document.moveSelectionBy()`	Mueve los objetos seleccionados una distancia especificada.
`document.optimizeCurves()`	Optimiza el suavizado de la selección actual, permitiendo varias pasadas, si se especifican, para obtener un suavizado óptimo; equivale a seleccionar Modificar > Forma > Optimizar.
`document.publish()`	Publica el documento según la configuración de publicación activa (Archivo > Configuración de publicación); equivale a seleccionar Archivo > Publicar.
`document.punch()`	Utiliza el objeto de dibujo seleccionado en la parte superior para perforar todos los objetos de dibujo seleccionados por debajo.
`document.removeAllFilters()`	Elimina todos los filtros del objeto u objetos seleccionados.
`document.removeDataFromDocument()`	Elimina datos persistentes con el nombre especificado que se han asociado al documento.
`document.removeDataFromSelection()`	Elimina datos persistentes con el nombre especificado que se han asociado a la selección.
`document.removeFilter()`	Elimina el filtro especificado de la lista Filtros del objeto u objetos seleccionados.
`document.renamePublishProfile()`	Cambia el nombre del perfil actual.
`document.renameScene()`	Cambia el nombre de la escena seleccionada actualmente en el panel Escenas.
`document.reorderScene()`	Mueve la escena especificada delante de otra escena especificada.
`document.resetOvalObject()`	Establece todos los valores del inspector de propiedades con la configuración predeterminada del objeto Oval.
`document.resetRectangleObject()`	Establece todos los valores del inspector de propiedades con la configuración predeterminada del objeto Rectangle.
`document.resetTransformation()`	Restablece la matriz de transformación; equivale a seleccionar Modificar > Transformar > Eliminar transformación.

Método	Descripción
`document.revert()`	Devuelve el documento especificado a su versión guardada con anterioridad; equivale a seleccionar Archivo > Descartar cambios.
`document.revertToLastVersion()`	Devuelve el documento especificado a la versión almacenada en el servidor Version Cue y registra todos los errores en el panel Salida.
`document.rotate3DSelection()`	Aplica un giro 3D a la selección.
`document.rotateSelection()`	Gira la selección el número de grados especificado.
`document.save()`	Guarda el documento en su ubicación predeterminada; equivale a seleccionar Archivo > Guardar.
`document.saveAndCompact()`	Guarda y compacta el archivo; equivale a seleccionar Archivo > Guardar y compactar.
`document.saveAVersion()`	Guarda una versión del documento especificado en el servidor Version Cue.
`document.scaleSelection()`	Escala la selección en la cantidad especificada; equivale al uso de la herramienta Transformación libre para aplicar escala al objeto.
`document.selectAll()`	Selecciona todos los elementos en el escenario; equivale a presionar Control+A (Windows), Comando+A (Macintosh) o seleccionar Edición > Seleccionar todo.
`document.selectNone()`	Anula la selección de los elementos seleccionados.
`document.setAlignToDocument()`	Define las preferencias de `document.align()`, `document.distribute()`, `document.match()` y `document.space()` para que actúen en el documento; equivale a la activación del botón En escenario del panel Alinear.
`document.setBlendMode()`	Establece el modo de mezcla para los objetos seleccionados.
`document.setCustomFill()`	Establece la configuración de relleno para el panel Herramientas, el inspector de propiedades y cualquier forma seleccionada.
`document.setCustomStroke()`	Establece la configuración de trazo para el panel Herramientas, el inspector de propiedades y cualquier forma seleccionada.
`document.setElementProperty()`	Establece la propiedad `Element` especificada en el objeto u objetos seleccionados en el documento.
`document.setElementTextAttr()`	Establece la propiedad `textAttrs` especificada de los elementos de texto seleccionados con el valor especificado.
`document.setFillColor()`	Cambia el color de relleno de la selección al especificarlo.
`document.setFilterProperty()`	Establece una propiedad de filtro especificada para los objetos seleccionados actualmente.
`document.setFilters()`	Aplica filtros a los objetos seleccionados.
`document.setInstanceAlpha()`	Establece la opacidad de la instancia.
`document.setInstanceBrightness()`	Establece el brillo de la instancia.
`document.setInstanceTint()`	Establece la tinta de la instancia.

Método	Descripción
`document.setMetadata()`	Establece los metadatos XML para el documento especificado, sobrescribiendo los metadatos existentes.
`document.setMobileSettings()`	Establece el valor de una cadena de configuración XML en un archivo FLA móvil.
`document.setOvalObjectProperty()`	Especifica un valor para la propiedad indicada de objetos Oval simples.
`document.setPlayerVersion()`	Establece la versión de Flash Player del documento especificado.
`document.setRectangleObjectProperty()`	Especifica un valor para la propiedad indicada de objetos Rectangle simples.
`document.setSelectionBounds()`	Mueve y cambia el tamaño de la selección en una única operación.
`document.setSelectionRect()`	Dibuja un recuadro de delimitación rectangular en relación con el escenario, empleando las coordenadas especificadas.
`document.setStageVanishingPoint()`	Especifica el punto de desvanecimiento para visualizar objetos 3D.
`document.setStageViewAngle()`	Especifica el ángulo de perspectiva para visualizar objetos 3D.
`document.setStroke()`	Establece el color, el ancho y el estilo de los trazos seleccionados.
`document.setStrokeColor()`	Cambia el color de trazo de la selección al especificado.
`document.setStrokeSize()`	Cambia el tamaño de trazo de la selección al especificado.
`document.setStrokeStyle()`	Cambia el estilo de trazo de la selección al especificado.
`document.setTextRectangle()`	Cambia el rectángulo de delimitación para el elemento de texto seleccionado al tamaño especificado.
`document.setTextSelection()`	Establece la selección de texto del campo de texto seleccionado actualmente con los valores especificados por los valores ***`startIndex`*** y ***`endIndex`***.
`document.setTextString()`	Inserta una cadena de texto.
`document.setTransformationPoint()`	Mueve el punto de transformación de la selección actual.
`document.skewSelection()`	Sesga la selección en la cantidad especificada.
`document.smoothSelection()`	Suaviza la curva de cada línea curva o contorno de relleno seleccionado.
`document.space()`	Distribuye los objetos de la selección de manera uniforme.
`document.straightenSelection()`	Endereza los trazos seleccionados actualmente; equivale al uso del botón Enderezar del panel Herramientas.
`document.swapElement()`	Cambia la selección actual por la especificada.
`document.swapStrokeAndFill()`	Intercambia los colores de Trazo y Relleno.
`document.synchronizeWithHeadVersion()`	Sincroniza el documento especificado con la versión más actual en el servidor Version Cue y registra cualquier error en el panel Salida.
`document.testMovie()`	Ejecuta una operación Probar película en el documento.

Método	Descripción
`document.testScene()`	Ejecuta una operación Probar escena en la escena actual del documento.
`document.traceBitmap()`	Realiza un mapa de bits de traza en la selección actual; equivale a seleccionar Modificar > Mapa de bits > Trazar mapa de bits.
`document.transformSelection()`	Realiza una transformación general en la selección actual aplicando la matriz especificada en los argumentos.
`document.translate3DCenter()`	Define la posición XYZ alrededor de la cual se transforma y se gira la selección.
`document.translate3DSelection()`	Aplica una transformación 3D a la selección.
`document.unGroup()`	Desagrupa la selección actual.
`document.union()`	Combina todas las formas seleccionadas en un objeto de dibujo.
`document.unlockAllElements()`	Desbloquea todos los elementos bloqueados en el fotograma seleccionado actualmente.
`document.xmlPanel()`	Envía un cuadro de diálogo XMLUI.

Resumen de propiedades

Puede emplear las propiedades siguientes con el objeto Document.

Propiedad	Descripción
`document.accName`	Una cadena que equivale al campo Nombre del panel Accesibilidad.
`document.as3AutoDeclare`	Valor booleano que describe si las instancias situadas en el escenario se añaden automáticamente a las clases de línea de tiempo definidas por el usuario.
`document.as3Dialect`	Cadena que describe el “dialecto” de ActionScript 3.0 empleado en el documento especificado.
`document.as3ExportFrame`	Entero que especifica el fotograma al que se exportarán las clases de ActionScript 3.0.
`document.as3StrictMode`	Valor booleano que especifica si el compilador de ActionScript 3.0 debe compilar con la opción Modo estricto activada o desactivada.
`document.as3WarningsMode`	Valor booleano que especifica si el compilador de ActionScript 3.0 debe compilar con la opción Modo de advertencias activada o desactivada.
`document.asVersion`	Entero que especifica qué versión de ActionScript se está utilizando en el archivo especificado.
`document.autoLabel`	Valor booleano que equivale a la casilla de verificación Etiquetado automático del panel Accesibilidad.
`document.backgroundColor`	Cadena, valor hexadecimal o entero que representa el color de fondo.
`document.currentPublishProfile`	Cadena que especifica el nombre del perfil de publicación activo para el documento especificado.
`document.currentTimeline`	Entero que especifica el índice de la línea de tiempo activa.
`document.description`	Una cadena que equivale al campo Descripción del panel Accesibilidad.
`document.docClass`	Especifica la clase de ActionScript 3.0 de nivel superior asociada al documento.

Propiedad	Descripción
`document.externalLibraryPath`	Cadena que contiene una lista de elementos de la ruta de biblioteca externa de ActionScript 3.0 del documento, donde se especifica la ubicación de los archivos SWC utilizados como bibliotecas compartidas de tiempo de ejecución.
`document.forceSimple`	Valor booleano que especifica si los elementos secundarios del objeto especificado son accesibles.
`document.frameRate`	Valor flotante que especifica el número de fotogramas mostrados por segundo cuando se reproduce el archivo SWF; el valor predeterminado es 12.
`document.height`	Entero que especifica la altura del documento (escenario) en píxeles.
`document.id`	Entero exclusivo (asignado automáticamente) que identifica a un documento durante una sesión de Flash.
`document.library`	Sólo lectura; el Objeto library para un documento.
`document.libraryPath`	Cadena que contiene una lista de elementos de la ruta de biblioteca de ActionScript 3.0 del documento, donde se especifica la ubicación de los archivos SWC o de las carpetas que contienen los archivos SWC.
`document.livePreview`	Valor booleano que especifica si está activada la opción Vista previa dinámica.
`document.name`	Sólo lectura; cadena que representa el nombre de un documento (archivo FLA).
`document.path`	Sólo lectura; cadena que representa la ruta del documento con el formato específico de la plataforma.
`document.pathURI`	Sólo lectura; cadena que representa la ruta del documento, expresada como URI file:///.
`document.publishProfiles`	Sólo lectura; conjunto de los nombres del perfil de publicación para el documento.
`document.screenOutline`	Sólo lectura; el objeto actual Objeto ScreenOutline para el documento.
`document.selection`	Conjunto de los objetos seleccionados en el documento.
`document.silent`	Un valor booleano que especifica si el objeto es accesible.
`document.sourcePath`	Cadena que contiene una lista de elementos de la ruta de origen de ActionScript 3.0 del documento, donde se especifica la ubicación de los archivos de clases de ActionScript.
`document.timelines`	Sólo lectura; conjunto de objetos Timeline (consulte Objeto Timeline).
`document.viewMatrix`	Sólo lectura; un Objeto Matrix.
`document.width`	Entero que especifica la anchura del documento (escenario) en píxeles.
`document.zoomFactor`	Especifica el porcentaje de zoom del escenario en tiempo de edición.

document.accName

Disponibilidad

Flash MX 2004

Uso

```
document.accName
```

Descripción

Propiedad; una cadena que equivale al campo Nombre del panel Accesibilidad. Los lectores de pantalla identifican los objetos mediante la lectura del nombre en voz alta.

Ejemplo

El ejemplo siguiente establece el nombre de accesibilidad del documento como `"Main Movie"`:

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

El ejemplo siguiente obtiene el nombre de accesibilidad del documento:

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

document.addDataToDocument()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`name` Una cadena que especifica el nombre de los datos que se van a añadir.

`type` Una cadena que define el tipo de los datos que se van a añadir. Los valores aceptables son `"integer"`, `"integerArray"`, `"double"`, `"doubleArray"`, `"string"` y `"byteArray"`.

`data` El valor que se va a añadir. Los tipos válidos dependen del parámetro *type*.

Valor devuelto

Ninguno.

Descripción

Método; almacena datos especificados con un documento. Los datos se escriben en el archivo FLA y están disponibles en JavaScript cuando se vuelve a abrir el archivo.

Ejemplo

El ejemplo siguiente añade un valor de entero 12 al documento actual:

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

El ejemplo siguiente devuelve el valor de los datos con el nombre `"myData"` y muestra el resultado en el panel Salida:

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

Véase también

`document.getDataFromDocument()`, `document.removeDataFromDocument()`

document.addDataToSelection()

Disponibilidad
Flash MX 2004

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

Parámetros
`name` Una cadena que especifica el nombre de los datos persistentes.

`type` Define el tipo de datos. Los valores aceptables son `"integer"`, `"integerArray"`, `"double"`, `"doubleArray"`, `"string"` y `"byteArray"`.

`data` El valor que se va a añadir. Los tipos válidos dependen del parámetro *type*.

Valor devuelto
Ninguno.

Descripción
Método; almacena datos especificados con el objeto u objetos seleccionados. Los datos se escriben en el archivo FLA y están disponibles en JavaScript cuando se vuelve a abrir el archivo. Sólo los símbolos y mapas de bits admiten datos persistentes.

Ejemplo
El ejemplo siguiente añade un valor entero de 12 al objeto seleccionado:

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

Véase también
`document.removeDataFromSelection()`

document.addFilter()

Disponibilidad
Flash 8

Uso
`document.addFilter(filterName)`

Parámetros
`filterName` Cadena que especifica el filtro que se va a añadir a la lista Filtros y que se activará para el objeto u objetos seleccionados. Los valores aceptables son `"adjustColorFilter"`, `"bevelFilter"`, `"blurFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientBevelFilter"` y `"gradientGlowFilter"`.

Valor devuelto
Ninguno.

Descripción

Método; aplica un filtro a los objetos seleccionados y coloca el filtro al final de la lista Filtros.

Ejemplo

El ejemplo siguiente aplica un filtro de iluminado al objeto u objetos seleccionados:

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

Véase también

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

document.addItem()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`position` Un punto que especifica las coordenadas *x* e *y* de la ubicación en la que se desea añadir el elemento. Utiliza el centro de un símbolo o la esquina superior izquierda de un mapa de bits o de un vídeo.

`item` Un objeto Item que especifica el elemento que se va a añadir y la biblioteca desde la que se va a añadir (consulte Objeto Item).

Valor devuelto

Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción

Método; añade un elemento desde cualquier documento o biblioteca abierta al objeto Document especificado.

Ejemplo

El ejemplo siguiente añade el primer elemento de la biblioteca al primer documento en la ubicación especificada para el símbolo, mapa de bits o vídeo seleccionado.

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

El ejemplo siguiente añade el símbolo `myMovieClip` desde la biblioteca del documento actual hasta el documento actual:

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

El ejemplo siguiente añade el símbolo `myMovieClip` desde el segundo documento del conjunto de documentos hasta el tercer documento de dicho conjunto:

```
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()

Disponibilidad
Flash MX 2004

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

Parámetros
`startpoint` Un par de números de coma flotante que especifican las coordenadas *x* e *y* donde comienza la línea.

`endpoint` Un par de números de coma flotante que especifican las coordenadas *x* e *y* donde finaliza la línea.

Valor devuelto
Ninguno.

Descripción
Método; añade un nuevo trazado entre dos puntos. El método utiliza los atributos de trazo actuales del documento y añade el trazado en el fotograma y la capa actuales. Este método equivale a hacer clic en la herramienta Línea y dibujar una línea.

Ejemplo
El ejemplo siguiente añade una línea entre el punto de partida y el punto final especificados:

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

document.addNewOval()

Disponibilidad
Flash MX 2004

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

Parámetros
`boundingRectangle` Un rectángulo que especifica los límites del óvalo que desea añadir. Para obtener más información sobre el formato de *boundingRectangle*, consulte `document.addNewRectangle()`.

`bSuppressFill` Un valor booleano que, si se define como `true`, hace que el método cree la forma sin relleno. El valor predeterminado es `false`. Este parámetro es opcional.

`bSuppressStroke` Un valor booleano que, si se define como `true`, hace que el método cree la forma sin trazo. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; añade un nuevo objeto Oval al rectángulo de delimitación especificado. Este método realiza la misma operación que la herramienta Óvalo. El método utiliza los atributos predeterminados de trazo y relleno del documento y añade el óvalo en el fotograma y la capa actuales. Si tanto *bSuppressFill* como *bSuppressStroke* se definen como `true`, el método no tiene ningún efecto.

Ejemplo
El ejemplo siguiente añade un nuevo óvalo dentro de las coordenadas especificadas; 164 píxeles de ancho por 178 píxeles de alto:

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

El ejemplo siguiente dibuja el óvalo sin relleno:

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

El ejemplo siguiente dibuja el óvalo sin trazo:

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

Véase también
`document.addNewPrimitiveOval()`

document.addNewPrimitiveOval()

Disponibilidad
Flash CS4 Professional

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

Parámetros
`boundingRectangle` Un rectángulo que especifica los límites en los que se añade el nuevo óvalo. Para más información sobre el formato de *boundingRectangle*, consulte document.addNewRectangle().

`bSuppressFill` Un valor booleano que, si se define como `true`, hace que el método cree el óvalo sin relleno. El valor predeterminado es `false`. Este parámetro es opcional.

`bSuppressStroke` Un valor booleano que, si se define como `true`, hace que el método cree el óvalo sin trazo. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción

Método; añade un nuevo óvalo simple dentro de la delimitación especificada. Este método realiza la misma operación que la herramienta Óvalo simple. El óvalo simple utiliza los atributos predeterminados de trazo y relleno del documento, y se añade en el fotograma y la capa actuales. Si tanto *bSuppressFill* como *bSuppressStroke* se definen como `true`, el método no tiene ningún efecto.

Ejemplo

El siguiente ejemplo añade óvalos simples dentro de las coordenadas especificadas con y sin relleno y trazo:

```
// 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);
```

Véase también

`document.addNewOval()`

document.addNewPrimitiveRectangle()

Disponibilidad

Flash CS4 Professional

Uso

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

Parámetros

`rect` Un rectángulo que especifica los límites en los que se añade el nuevo rectángulo simple. Para más información sobre el formato de *boundingRectangle*, consulte document.addNewRectangle().

`roundness` Un entero entre 0 y 999 que representa el número de puntos utilizados para especificar el grado de redondez de las esquinas.

`bSuppressFill` Un valor booleano que, si se define como `true`, hace que el método cree el rectángulo sin relleno. El valor predeterminado es `false`. Este parámetro es opcional.

`bSuppressStroke` Un valor booleano que, si se define como `true`, hace que el método cree el rectángulo sin trazo. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; añade un nuevo rectángulo simple dentro de la delimitación especificada. Este método realiza la misma operación que la herramienta Rectángulo simple. El rectángulo simple utiliza los atributos predeterminados de trazo y relleno del documento, y se añade en el fotograma y la capa actuales. Si tanto *bSuppressFill* como *bSuppressStroke* se definen como `true`, el método no tiene ningún efecto.

Ejemplo

El siguiente ejemplo añade rectángulos simples dentro de las coordenadas especificadas con y sin relleno y trazo:

```
// 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);
```

Véase también

`document.addNewRectangle()`

document.addNewPublishProfile()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`profileName` El nombre exclusivo del nuevo perfil. Si no especifica un nombre, se suministrará un nombre predeterminado. Este parámetro es opcional.

Valor devuelto

Un entero que es el índice del nuevo perfil en la lista de perfiles. Devuelve -1 si no se puede crear un perfil nuevo.

Descripción

Método; añade un nuevo perfil de publicación y lo convierte en el actual.

Ejemplo

El ejemplo siguiente añade un nuevo perfil de publicación con un nombre predeterminado y, a continuación, muestra el nombre del perfil en el panel Salida:

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

El ejemplo siguiente añade un nuevo perfil de publicación con el nombre "`my profile`":

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

Véase también

`document.deletePublishProfile()`

document.addNewRectangle()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`boundingRectangle` Un rectángulo que especifica los límites dentro de los cuales se añade el nuevo rectángulo, con el formato `{left:value1,top:value2,right:value3,bottom:value4}`. Los valores `left` y `top` especifican la ubicación de la esquina superior izquierda (por ej., `left:0,top:0` representa la esquina superior izquierda del escenario), y los valores `right` y `bottom` especifican la ubicación de la esquina inferior derecha. Por tanto, la anchura del rectángulo es la diferencia de valor entre `left` y `right`, mientras que la altura es la diferencia entre `top` y `bottom`.

En otras palabras, no todos los límites del rectángulo corresponden a los valores mostrados en el inspector de propiedades. Los valores `left` y `top` corresponden a los valores X e Y del inspector de propiedades, respectivamente. Sin embargo, los valores `right` y `bottom` no corresponden a los valores de anchura y altura del inspector de propiedades. Por ejemplo, considere un rectángulo con los siguientes límites:

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

Este rectángulo mostraría los siguientes valores en el inspector de propiedades:

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

`roundness` Un valor entero de 0 a 999 que especifica la redondez que se va a utilizar para las esquinas. El valor se expresa como número de puntos. Cuanto mayor sea el valor, mayor será la redondez.

`bSuppressFill` Un valor booleano que, si se define como `true`, hace que el método cree la forma sin relleno. El valor predeterminado es `false`. Este parámetro es opcional.

`bSuppressStroke` Un valor booleano que, si se define como `true`, hace que el método cree el rectángulo sin trazo**.** El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; añade un nuevo rectángulo o rectángulo redondeado, ajustándolo a los límites especificados. Este método realiza la misma operación que la herramienta Rectángulo. El método utiliza los atributos predeterminados de trazo y relleno del documento y añade el rectángulo en el fotograma y la capa actuales. Si tanto *bSuppressFill* como *bSuppressStroke* se definen como `true`, el método no tiene ningún efecto.

Ejemplo

El ejemplo siguiente añade un rectángulo nuevo sin esquinas redondeadas dentro de las coordenadas especificadas; 100 píxeles de anchura y altura:

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

El ejemplo siguiente añade un rectángulo nuevo sin esquinas redondeadas y sin relleno; 100 píxeles de anchura y 200 de altura:

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

El ejemplo siguiente añade un rectángulo nuevo sin esquinas redondeadas y sin trazo; 200 píxeles de anchura y 100 de altura:

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

Véase también

`document.addNewPrimitiveRectangle()`

document.addNewScene()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`name` Especifica el nombre de la escena. Si no especifica un nombre, se generará un nombre de escena nuevo.

Valor devuelto

Valor booleano: `true` si la escena se añade correctamente, y `false` en caso contrario.

Descripción

Método; añade una nueva escena (`Objeto Timeline`) después de la escena seleccionada y la convierte en la seleccionada actualmente. Si el nombre de la escena especificada ya existe, la escena no se añade y el método devuelve un error.

Ejemplo

El ejemplo siguiente añade una nueva escena llamada `myScene` después de la escena actual en el documento actual. La variable `success` será `true` cuando se cree la nueva escena, y `false` en caso contrario.

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

El ejemplo siguiente añade una nueva escena utilizando la convención de asignación de nombres predeterminada. Si sólo existe una escena, la escena recién creada se llamará `"Scene 2"`.

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

document.addNewText()

Disponibilidad

Flash MX 2004; parámetro *text* opcional añadido en Flash CS3 Professional.

Uso

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

Parámetros

`boundingRectangle` Especifica el tamaño y la ubicación del campo de texto. Para obtener más información sobre el formato de *boundingRectangle*, consulte `document.addNewRectangle()`.

`text` Una cadena opcional que especifica el texto que se colocará en el campo. Si omite este parámetro, la selección en el panel Herramientas cambia a la herramienta Texto. Por tanto, si no desea cambiar la herramienta seleccionada, pase un valor para *text*.

Valor devuelto

Ninguno.

Descripción

Método; inserta un nuevo campo de texto y, opcionalmente, coloca texto en el campo. Si omite el parámetro *text*, puede llamar a `document.setTextString()` para rellenar el campo de texto.

Ejemplo

El ejemplo siguiente crea un nuevo campo de texto en la esquina superior izquierda del escenario y establece la cadena de texto como `"Hello World"`:

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

Véase también

`document.setTextString()`

document.align()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`alignmode` Una cadena que especifica cómo se alinea la selección. Los valores aceptables son `"left"`, `"right"`, `"top"`, `"bottom"`, `"vertical center"` y `"horizontal center"`.

`bUseDocumentBounds` Un valor booleano que, si se define como `true`, hace que el método alinee los límites del documento. En caso contrario, el método utiliza los límites de los objetos seleccionados. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; alinea la selección.

Ejemplo

El ejemplo siguiente alinea los objetos a la izquierda y con el escenario. Equivale a activar la opción En escenario en el panel Alienar y a hacer clic en el botón Alinear a la izquierda:

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

Véase también

`document.distribute()`, `document.getAlignToDocument()`, `document.setAlignToDocument()`

document.allowScreens()

Disponibilidad

Flash MX 2004

Uso

```
document.allowScreens()
```

Parámetros

Ninguno.

Valor devuelto

Un valor booleano: `true` si se puede usar `document.screenOutline` con seguridad; `false` en caso contrario.

Descripción

Método; se utiliza antes de usar la propiedad `document.screenOutline`. Si este método devuelve el valor `true`, podrá acceder de forma segura a `document.screenOutline`; Flash muestra un error si accede a `document.screenOutline` en un documento sin pantallas.

Ejemplo

El ejemplo siguiente determina si se pueden emplear métodos `screens` en el documento actual:

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

Véase también

`document.screenOutline`

document.arrange()

Disponibilidad

Flash MX 2004

Uso

```
document.arrange(arrangeMode)
```

Parámetros

`arrangeMode` Especifica la dirección en la que se mueve la selección. Los valores aceptables son `"back"`, `"backward"`, `"forward"` y `"front"`. Ofrece las mismas posibilidades que las opciones del menú Modificar > Organizar.

Valor devuelto

Ninguno.

Descripción

Método; organiza la selección en el escenario. Este método sólo se aplica a objetos sin forma.

Ejemplo

El ejemplo siguiente mueve la selección actual a un primer plano:

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

document.as3AutoDeclare

Disponibilidad

Flash CS3 Professional

Uso

```
document.as3AutoDeclare
```

Descripción

Propiedad; un valor booleano que describe si las instancias situadas en el escenario se añaden automáticamente a las clases de línea de tiempo definidas por el usuario. El valor predeterminado es `true`.

Ejemplo

El ejemplo siguiente especifica que las instancias situadas en el escenario del documento actual se deben añadir manualmente a las clases de línea de tiempo definidas por el usuario.

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

document.as3Dialect

Disponibilidad

Flash CS3 Professional

Uso

```
document.as3Dialect
```

Descripción

Propiedad; una cadena que describe el "dialecto" de ActionScript 3.0 empleado en el documento especificado. El valor predeterminado es `"AS3"`. Si desea permitir clases de prototipo, como las permitidas en las especificaciones anteriores de ECMAScript, establezca este valor como `"ES"`.

Ejemplo

El ejemplo siguiente especifica que el dialecto utilizado en el documento actual es ECMAScript:

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

Véase también

`document.asVersion`

document.as3ExportFrame

Disponibilidad

Flash CS3 Professional

Uso

```
document.as3ExportFrame
```

Descripción

Propiedad; un entero que especifica el fotograma al que se exportarán las clases de ActionScript 3.0. De forma predeterminada, las clases se exportan al fotograma 1.

Ejemplo

El ejemplo siguiente cambia el fotograma al que se exportan las clases de 1 (el predeterminado) a 5.

```
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

Disponibilidad

Flash CS3 Professional

Uso

```
document.as3StrictMode
```

Descripción

Propiedad; un valor booleano que especifica si el compilador de ActionScript 3.0 debe compilar con la opción Modo estricto activada (`true`) o desactivada (`false`). El Modo estricto provoca que las advertencias se registren como errores, lo que significa que no se realizará la compilación si existen esos errores. El valor predeterminado es `true`.

Ejemplo

El ejemplo siguiente desactiva la opción Modo estricto del compilador.

```
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);
```

Véase también

`document.as3WarningsMode`

document.as3WarningsMode

Disponibilidad

Flash CS3 Professional

Uso

```
document.as3WarningsMode
```

Descripción

Propiedad; un valor booleano que especifica si el compilador de ActionScript 3.0 debe compilar con la opción Modo de advertencias activada (`true`) o desactivada (`false`). El Modo de advertencias provoca que se registren advertencias adicionales que resultan útiles para descubrir incompatibilidades al actualizar el código de ActionScript 2.0 a ActionScript 3.0. El valor predeterminado es `true`.

Ejemplo

El ejemplo siguiente desactiva la opción Modo de advertencias del compilador.

```
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);
```

Véase también

`document.as3StrictMode`

document.asVersion

Disponibilidad

Flash CS3 Professional

Uso

`document.asVersion`

Descripción

Propiedad; un entero que especifica qué versión de ActionScript se está utilizando en el documento indicado. Los valores aceptables son 1, 2 y 3.

Para determinar la versión del reproductor de destino para el documento especificado, utilice `document.getPlayerVersion()`; este método devuelve una cadena, de modo que lo pueden utilizar los reproductores Flash® Lite™.

Ejemplo

El ejemplo siguiente establece la versión de ActionScript del documento actual como ActionScript 2.0 si lo está actualmente como ActionScript 1.0.

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

Véase también

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

document.autoLabel

Disponibilidad

Flash MX 2004

Uso

`document.autoLabel`

Descripción

Propiedad; un valor booleano que equivale a la casilla de verificación Etiquetado automático del panel Accesibilidad. Puede utilizar esta propiedad para indicar a Flash que etiquete objetos automáticamente en el escenario con el texto asociado a ellos.

Ejemplo

El ejemplo siguiente obtiene el valor de la propiedad `autoLabel` y muestra el resultado en el panel Salida:

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

El ejemplo siguiente define la propiedad `autoLabel` como `true`, indicándole a Flash que etiquete automáticamente los objetos del escenario:

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

document.backgroundColor

Disponibilidad
Flash MX 2004

Uso
`document.backgroundColor`

Descripción
Propiedad; el color del fondo, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

Ejemplo
El ejemplo siguiente establece el color de fondo como negro:

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

document.breakApart()

Disponibilidad
Flash MX 2004

Uso
`document.breakApart()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; realiza una operación de separación en la selección actual.

Ejemplo
El ejemplo siguiente separa la selección actual:

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

document.canEditSymbol()

Disponibilidad
Flash MX 2004

Uso
`document.canEditSymbol()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si el menú Editar símbolos y las funciones están disponibles para utilizar; `false` en caso contrario.

Descripción
Método; indica si el menú Editar símbolos y las funciones están activados. No indica si la selección se puede editar. Este método no se debe utilizar para comprobar si se permite `fl.getDocumentDOM().enterEditMode()`.

Ejemplo
El ejemplo siguiente muestra en el panel Salida el estado del menú Editar símbolos y las funciones:

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

document.canRevert()

Disponibilidad
Flash MX 2004

Uso
`document.canRevert()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si puede utilizar correctamente los métodos `document.revert()` o `fl.revertDocument()`, y `false` en caso contrario.

Descripción
Método; determina si puede utilizar correctamente el método `document.revert()` o `fl.revertDocument()`.

Ejemplo
El ejemplo siguiente comprueba si el documento actual puede volver a la versión guardada anteriormente. Si es así, `fl.getDocumentDOM().revert()` restaura la versión guardada anteriormente.

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

document.canSaveAVersion()

Disponibilidad

Flash CS3 Professional

Uso

`document.canSaveAVersion()`

Parámetros

Ninguno.

Valor devuelto

Un valor booleano de `true` si se puede guardar una versión del archivo en el servidor Version Cue; `false` en caso contrario.

Descripción

Método; determina si una versión del documento especificado se puede guardar en el servidor Version Cue.

Ejemplo

El ejemplo siguiente comprueba si se puede utilizar `document.saveAVersion()`. Si es así, llama al método.

```
if(fl.getDocumentDOM().canSaveAVersion()){
    fl.getDocumentDOM().saveAVersion;
}
```

Véase también

`document.revertToLastVersion()`, `document.saveAVersion()`

document.canTestMovie()

Disponibilidad

Flash MX 2004

Uso

`document.canTestMovie()`

Parámetros

Ninguno.

Valor devuelto

Un valor booleano: `true` si puede utilizar correctamente el método `document.testMovie()`; `false` en caso contrario.

Descripción

Método; determina si puede utilizar correctamente el método `document.testMovie()`.

Ejemplo

El ejemplo siguiente comprueba si puede utilizarse `fl.getDocumentDOM().testMovie()`. Si es así, llama al método.

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

Véase también

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

document.canTestScene()

Disponibilidad

Flash MX 2004

Uso

```
document.canTestScene()
```

Parámetros

Ninguno.

Valor devuelto

Un valor booleano: `true` si puede utilizar correctamente el método `document.testScene()`; `false` en caso contrario.

Descripción

Método; determina si puede utilizar correctamente el método `document.testScene()`.

Ejemplo

El ejemplo siguiente comprueba en primer lugar si se puede utilizar correctamente fl.getDocumentDOM().testScene(). Si es así, llama al método.

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

Véase también

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

document.changeFilterOrder()

Disponibilidad

Flash 8

Uso

```
document.changeFilterOrder(oldIndex, newIndex)
```

Parámetros

oldIndex Un entero que representa la posición actual del índice basado en cero del filtro que desea reubicar en la lista Filtros.

newIndex Un entero que representa la nueva posición del índice del filtro en la lista.

Valor devuelto

Ninguno.

Descripción

Método; cambia el índice del filtro de la lista Filtros. Todos los filtros por encima o por debajo de *newIndex* se cambian hacia arriba o hacia abajo según corresponda. Por ejemplo, al utilizar los filtros mostrados a continuación, si emite el comando fl.getDocumentDOM().changeFilterOrder `(3, 0)`, los filtros se reorganizarán de este modo:

Antes	Después
`blurFilterdropShadowFilterglowFiltergradientBevelFilter`	`gradientBevelFilterblurFilterdropShadowFilterglowFilter`

Si emite después el comando fl.getDocumentDOM().changeFilterOrder `(0, 2)`, los filtros se reorganizan de la manera siguiente:

Antes	Después
`gradientBevelFilterblurFilterdropShadowFilterglowFilter`	`blurFilterdropShadowFiltergradientBevelFilterglowFilter`

Ejemplo

El ejemplo siguiente mueve a la primera posición el filtro que se encuentra actualmente en la segunda posición de la lista Filtros:

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

Véase también

`document.addFilter()`, `document.disableFilter()`, `document.enableFilter()`, `document.getFilters()`, `document.removeFilter()`, Objeto Filter

document.clipCopy()

Disponibilidad

Flash MX 2004

Uso

```
document.clipCopy()
```

Parámetros

Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; copia la selección actual desde el documento hasta el Portapapeles.

Para copiar una cadena en el Portapapeles, utilice `fl.clipCopyString()`.

Ejemplo
El ejemplo siguiente copia la selección actual desde el documento hasta el Portapapeles:

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

Véase también
`document.clipCut()`, `document.clipPaste()`

document.clipCut()

Disponibilidad
Flash MX 2004

Uso
`document.clipCut()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; corta la selección actual del documento y la escribe en el Portapapeles.

Ejemplo
El ejemplo siguiente corta la selección actual del documento y la escribe en el Portapapeles:

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

Véase también
`document.clipCopy()`, `document.clipPaste()`, `fl.clipCopyString()`

document.clipPaste()

Disponibilidad
Flash MX 2004

Uso

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

Parámetros

`bInPlace` Un valor booleano que, cuando se define como `true`, hace que el método realice una operación Pegar in situ. El valor predeterminado es `false`, lo que hace que el método realice una operación de pegado en el centro del documento. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; pega el contenido del Portapapeles en el documento.

Ejemplo

El ejemplo siguiente pega el contenido del Portapapeles en el centro del documento:

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

El ejemplo siguiente pega el contenido del Portapapeles en el documento actual:

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

Véase también

`document.clipCopy()`, `document.clipCut()`, `fl.clipCopyString()`

document.close()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`bPromptToSaveChanges` Un valor booleano que, cuando se define como `true`,, hace que el método presente al usuario un cuadro de diálogo si hay cambios sin guardar en el documento. Si *bPromptToSaveChanges* se define como `false`, no se pregunta al usuario si desea guardar los documentos modificados. El valor predeterminado es `true`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; cierra el documento especificado.

Ejemplo

El ejemplo siguiente cierra el documento actual y muestra al usuario un cuadro de diálogo para guardar los cambios:

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

El ejemplo siguiente cierra el documento actual sin guardar los cambios:

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

document.convertLinesToFills()

Disponibilidad
Flash MX 2004

Uso
`document.convertLinesToFills()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; convierte líneas en rellenos en los objetos seleccionados.

Ejemplo
El ejemplo siguiente convierte las líneas seleccionadas actualmente en rellenos:

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

document.convertToSymbol()

Disponibilidad
Flash MX 2004

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

Parámetros
`type` Una cadena que especifica el tipo de símbolo que se va a crear. Los valores aceptables son `"movie clip"`, `"button"` y `"graphic"`.

`name` Una cadena que especifica el nombre del nuevo símbolo, que debe ser exclusivo. Puede enviar una cadena vacía para que este método cree un nombre de símbolo único.

`registrationPoint` Especifica el punto que representa la ubicación 0,0 del símbolo. Los valores admitidos son: `"top left"`, `"top center"`, `"top right"`, `"center left"`, `"center"`, `"center right"`, `"bottom left"`, `"bottom center"` y `"bottom right"`.

Valor devuelto

Un objeto para el símbolo recién creado o `null` si no puede crear el símbolo.

Descripción

Método; convierte el elemento o elemento de escenario seleccionados en un símbolo nuevo. Para obtener información sobre la definición de propiedades de vinculación y elementos compartidos para un símbolo, consulte Objeto Item.

Ejemplo

Los ejemplos siguientes crean un símbolo de clip de película con un nombre especificado, un símbolo de botón con un nombre especificado o un símbolo de clip de película con un nombre predeterminado:

```
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()

Disponibilidad

Flash 8

Uso

```
document.crop()
```

Parámetros

Ninguno.

Valor devuelto

Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción

Método; utiliza el objeto de dibujo seleccionado en la parte superior para recortar todos los objetos de dibujo seleccionados por debajo. Este método devuelve `false` si no hay objetos de dibujo seleccionados o si alguno de los elementos seleccionados no es un objeto de dibujo.

Ejemplo

El ejemplo siguiente recorta los objetos seleccionados actualmente:

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

Véase también

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

document.currentPublishProfile

Disponibilidad
Flash MX 2004

Uso
`document.currentPublishProfile`

Descripción
Propiedad; una cadena que especifica el nombre del perfil de publicación activo para el documento especificado.

Ejemplo
El ejemplo siguiente añade un nuevo perfil de publicación con el nombre predeterminado y, a continuación, muestra el nombre del perfil en el panel Salida:

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

El ejemplo siguiente cambia el perfil de publicación seleccionado a `"Default"`:

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

document.currentTimeline

Disponibilidad
Flash MX 2004

Uso
`document.currentTimeline`

Descripción
Propiedad; un entero que especifica el índice de la línea de tiempo activa. Puede establecer la línea de tiempo activa cambiando el valor de esta propiedad; el efecto es prácticamente equivalente a llamar a `document.editScene()`. La única diferencia es que no aparece un mensaje de error si el índice de la línea de tiempo no es válido, la propiedad simplemente no se establece, lo que provoca un error sin mensaje.

Ejemplo
El ejemplo siguiente muestra el índice de la línea de tiempo actual:

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

El ejemplo siguiente cambia la línea de tiempo activa desde la línea de tiempo principal hasta una escena llamada `"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;
}
```

Véase también

`document.getTimeline()`

document.deleteEnvelope()

Disponibilidad

Flash 8

Uso

`document.deleteEnvelope()`

Parámetros

Ninguno.

Valor devuelto

Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción

Método; elimina la envoltura (recuadro de delimitación que contiene uno o varios objetos) de los objetos seleccionados.

Ejemplo

El ejemplo siguiente elimina la envoltura de los objetos seleccionados:

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

Véase también

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

document.deletePublishProfile()

Disponibilidad

Flash MX 2004

Uso

`document.deletePublishProfile()`

Parámetros
Ninguno.

Valor devuelto
Un entero que es el índice del nuevo perfil actual. Si no hay un nuevo perfil disponible, el método deja el perfil actual sin modificar y devuelve su índice.

Descripción
Método; elimina el perfil activo actualmente, si hay más de uno. Debe quedar un perfil como mínimo.

Ejemplo
El ejemplo siguiente elimina el perfil activo actualmente, si hay más de uno, y muestra el índice del nuevo perfil activo:

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

Véase también
`document.addNewPublishProfile()`

document.deleteScene()

Disponibilidad
Flash MX 2004

Uso
`document.deleteScene()`

Parámetros
Ninguno.

Valor devuelto
Valor booleano: `true` si la escena se elimina correctamente, y `false` en caso contrario.

Descripción
Método; elimina la escena actual (Objeto Timeline) y, si la escena eliminada no era la última, establece la siguiente como el objeto Timeline actual. Si la escena eliminada era la última, establece el primer objeto como objeto Timeline actual. Si sólo existe un objeto Timeline (escena), devuelve el valor `false`.

Ejemplo
Suponiendo que haya tres escenas (`Scene0`, `Scene1` y `Scene2`) en el documento actual, el ejemplo siguiente convierte `Scene2` en la escena actual y, a continuación, la elimina:

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

document.deleteSelection()

Disponibilidad
Flash MX 2004

Uso
```
document.deleteSelection()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; elimina la selección actual en el escenario. Muestra un mensaje de error si no hay ninguna selección.

Ejemplo
El ejemplo siguiente elimina la selección actual en el documento:

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

document.description

Disponibilidad
Flash MX 2004

Uso
```
document.description
```

Descripción
Propiedad; una cadena que equivale al campo Descripción del panel Accesibilidad. El lector de pantalla lee esta descripción.

Ejemplo
El ejemplo siguiente establece la descripción del documento:

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

El ejemplo siguiente obtiene la descripción del documento y la muestra en el panel Salida:

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

document.disableAllFilters()

Disponibilidad
Flash 8

Uso
`document.disableAllFilters()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; desactiva todos los filtros en los objetos seleccionados.

Ejemplo
El ejemplo siguiente desactiva todos los filtros en los objetos seleccionados:

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

Véase también
`document.addFilter()`, `document.changeFilterOrder()`, `document.disableFilter()`, `document.disableOtherFilters()`, `document.enableAllFilters()`, `document.getFilters()`, `document.removeAllFilters()`, Objeto Filter

document.disableFilter()

Disponibilidad
Flash 8

Uso
`document.disableFilter(filterIndex)`

Parámetros
`filterIndex` Un entero que especifica el índice basado en cero del filtro en la lista Filtros.

Valor devuelto
Ninguno.

Descripción
Método; desactiva el filtro especificado en la lista Filtros.

Ejemplo

El ejemplo siguiente desactiva el primer y tercer filtros (valores de índice de 0 y 2) de la lista Filtros en el objeto u objetos seleccionados:

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

Véase también

`document.addFilter()`, `document.changeFilterOrder()`, `document.disableAllFilters()`, `document.disableOtherFilters()`, `document.enableFilter()`, `document.getFilters()`, `document.removeFilter()`, Objeto Filter

document.disableOtherFilters()

Disponibilidad

Flash 8

Uso

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

Parámetros

`enabledFilterIndex` Un entero que representa el índice basado en cero del filtro que debe permanecer activado cuando se desactiven todos los demás filtros.

Valor devuelto

Ninguno.

Descripción

Método; desactiva todos los filtros salvo el que se encuentra en la posición especificada en la lista Filtros.

Ejemplo

El ejemplo siguiente desactiva todos los filtros salvo el segundo de la lista (valor de índice de 1):

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

Véase también

`document.addFilter()`, `document.changeFilterOrder()`, `document.disableAllFilters()`, `document.disableFilter()`, `document.enableFilter()`, `document.getFilters()`, `document.removeFilter()`, Objeto Filter

document.distribute()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

`distributemode` Una cadena que especifica dónde se distribuyen los objetos seleccionados. Los valores aceptables son `"left edge"`, `"horizontal center"`, `"right edge"`, `"top edge"`, `"vertical center"` y `"bottom edge"`.

`bUseDocumentBounds` Un valor booleano que, si se define como `true`, distribuye los objetos seleccionados empleando los límites del documento. En caso contrario, el método utiliza los límites de los objetos seleccionados. El valor predeterminado es `false`.

Valor devuelto

Ninguno.

Descripción

Método; distribuye la selección.

Ejemplo

El ejemplo siguiente distribuye los objetos seleccionados por sus bordes superiores:

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

El ejemplo siguiente distribuye los objetos seleccionados por sus bordes superiores y establece expresamente el parámetro *bUseDcoumentBounds*:

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

El ejemplo siguiente distribuye los objetos seleccionados por sus bordes superiores, empleando los límites del documento:

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

Véase también

`document.getAlignToDocument()`, `document.setAlignToDocument()`

document.distributeToLayers()

Disponibilidad

Flash MX 2004

Uso

```
document.distributeToLayers()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; realiza una operación de distribución en capas en la selección actual; equivale a seleccionar Distribuir en capas. Este método muestra un error si no hay ninguna selección.

Ejemplo

El ejemplo siguiente distribuye la selección actual a las capas:

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

document.docClass

Disponibilidad

Flash CS3 Professional

Uso

```
document.docClass
```

Descripción

Propiedad; una cadena que especifica la clase de ActionScript 3.0 de nivel superior asociada al documento. Si el documento no está configurado para utilizar ActionScript 3.0, se omite esta propiedad.

Ejemplo

El ejemplo siguiente especifica que la clase de ActionScript 3.0 asociada al documento es com.mycompany.ManagerClass, definida en 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);
```

Véase también

item.linkageBaseClass

document.documentHasData()

Disponibilidad

Flash MX 2004

Uso

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

Parámetros

name Una cadena que especifica el nombre de los datos que se van a comprobar.

Valor devuelto
Un valor booleano: `true` si el documento tiene datos persistentes; `false` en caso contrario.

Descripción
Método; comprueba si el documento contiene datos persistentes con el nombre especificado.

Ejemplo
El ejemplo siguiente comprueba si el documento contiene datos persistentes con el nombre `"myData"`:

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

Véase también
`document.addDataToDocument()`, `document.getDataFromDocument()`, `document.removeDataFromDocument()`

document.duplicatePublishProfile()

Disponibilidad
Flash MX 2004

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

Parámetros
`profileName` Una cadena que especifica el nombre exclusivo del perfil duplicado. Si no especifica un nombre, el método utiliza el nombre predeterminado. Este parámetro es opcional.

Valor devuelto
Un entero que es el índice del nuevo perfil en la lista de perfiles. Devuelve -1 si no se puede duplicar el perfil.

Descripción
Método; duplica el perfil activo y selecciona la versión duplicada.

Ejemplo
El ejemplo siguiente duplica el perfil activo actualmente y muestra el índice del nuevo perfil en el panel Salida:

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

document.duplicateScene()

Disponibilidad
Flash MX 2004

Uso
```
document.duplicateScene()
```

Parámetros
Ninguno.

Valor devuelto
Valor booleano: `true` si la escena se duplica correctamente, y `false` en caso contrario.

Descripción
Método; realiza una copia de la escena seleccionada, asignando un nombre exclusivo a la nueva escena y convirtiéndola en la actual.

Ejemplo
El ejemplo siguiente duplica la segunda escena del documento actual:

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

document.duplicateSelection()

Disponibilidad
Flash MX 2004

Uso
```
document.duplicateSelection()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; duplica la selección en el escenario.

Ejemplo
El ejemplo siguiente duplica la selección actual, lo que equivale a hacer clic mientras se presiona la tecla Alt y, a continuación, se arrastra un elemento:

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

document.editScene()

Disponibilidad
Flash MX 2004

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

Parámetros

`index` Un entero basado en cero que especifica la escena que desea editar.

Valor devuelto

Ninguno.

Descripción

Método; convierte la escena especificada en la escena seleccionada actualmente para editar.

Ejemplo

Suponiendo que haya tres escenas (`Scene0`, `Scene1` y `Scene2`) en el documento actual, el ejemplo siguiente convierte `Scene2` en la escena actual y, a continuación, la elimina:

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

document.enableAllFilters()

Disponibilidad

Flash 8

Uso

```
document.enableAllFilters()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; activa todos los filtros de la lista Filtros para el objeto u objetos seleccionados.

Ejemplo

El ejemplo siguiente activa todos los filtros de la lista Filtros para el objeto u objetos seleccionados:

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

Véase también

`document.addFilter()`, `document.changeFilterOrder()`, `document.disableAllFilters()`, `document.enableFilter()`, `document.getFilters()`, `document.removeAllFilters()`, Objeto Filter

document.enableFilter()

Disponibilidad
Flash 8

Uso
`document.enableFilter(filterIndex)`

Parámetros
`filterIndex` Un entero que especifica el índice basado en cero del filtro en la lista Filtro que se desea activar.

Valor devuelto
Ninguno.

Descripción
Método; activa el filtro especificado para el objeto u objetos seleccionados.

Ejemplo
El ejemplo siguiente activa el segundo filtro del objeto u objetos seleccionados:

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

Véase también
`document.addFilter()`, `document.changeFilterOrder()`, `document.disableFilter()`, `document.enableAllFilters()`, `document.getFilters()`, `document.removeFilter()`, Objeto Filter

document.enterEditMode()

Disponibilidad
Flash MX 2004

Uso
`document.enterEditMode([editMode])`

Parámetros
`editMode` Una cadena que especifica el modo de edición. Los valores aceptables son `"inPlace"` o `"newWindow"`. Si no se especifica ningún parámetro, la opción predeterminada es el modo de edición de símbolos. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; cambia la herramienta de edición al modo de edición especificado por el parámetro. Si no se especifica ningún parámetro, el método utiliza de forma predeterminada el modo de edición de símbolos, que equivale a hacer clic con el botón derecho del ratón en el símbolo para activar el menú contextual y seleccionar Edición.

Ejemplo

El ejemplo siguiente sitúa a Flash en modo de edición en contexto para el símbolo seleccionado actualmente:

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

El ejemplo siguiente sitúa a Flash en modo de edición en una nueva ventana para el símbolo seleccionado actualmente:

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

Véase también

`document.exitEditMode()`

document.exitEditMode()

Disponibilidad

Flash MX 2004

Uso

```
document.exitEditMode()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; sale del modo de edición de símbolos y vuelve a seleccionar el siguiente nivel superior desde el modo de edición. Por ejemplo, si está editando un símbolo dentro de otro, este método sube un nivel desde el símbolo que está editando hasta el símbolo principal.

Ejemplo

El ejemplo siguiente sale del modo de edición de símbolos:

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

Véase también

`document.enterEditMode()`

document.exportPNG()

Disponibilidad

Flash 8

Uso

```
document.exportPNG([fileURI [, bCurrentPNGSettings [, bCurrentFrame]]])
```

Parámetros

`fileURI` Cadena que se expresa en forma de URI file:/// y que especifica el nombre del archivo exportado. Si *fileURI* es una cadena vacía o no está especificada, Flash mostrará el cuadro de diálogo Exportar película.

`bCurrentPNGSettings` Un valor booleano que especifica si se utiliza la configuración de publicación actual de PNG (`true`) o se muestra el cuadro de diálogo Exportar PNG (`false`). Este parámetro es opcional. El valor predeterminado es `false`.

`bCurrentFrame` Un valor booleano que especifica si se exporta sólo el fotograma actual (`true`) o todos los fotogramas, con cada fotograma como archivo PNG independiente (`false`). Este parámetro es opcional. El valor predeterminado es `false`.

Valor devuelto

Un valor booleano de `true` si el archivo se exporta correctamente como archivo PNG, y de `false` en caso contrario.

Descripción

Método; exporta el documento como uno o varios archivos PNG. Si se especifica *fileURI* y el archivo ya existe, se sobrescribe sin mostrar ninguna advertencia.

Ejemplo

El siguiente ejemplo exporta el fotograma actual del documento actual a myFile.png, con la configuración de publicación actual de PNG:

```
fl.getDocumentDOM().exportPNG("file:///C|/myProject/myFile.png", true, true);
```

document.exportPublishProfile()

Disponibilidad

Flash MX 2004

Uso

```
document.exportPublishProfile(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta del archivo XML al que se exporta el perfil.

Valor devuelto

Ninguno.

Descripción

Método; exporta el perfil activo actualmente a un archivo XML.

Ejemplo

El ejemplo siguiente exporta el perfil activo actualmente al archivo llamado profile.xml en la carpeta /Documents and Settings/nombredeusuario/Escritorio de la unidad C:

```
fl.getDocumentDOM().exportPublishProfile('file:///C|/Documents and
Settings/username/Desktop/profile.xml');
```

Véase también

`document.exportPublishProfileString()`, `document.importPublishProfile()`

document.exportPublishProfileString()

Disponibilidad

Flash CS4 Professional

Uso

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

Parámetros

`profileName` Una cadena que especifica el nombre del perfil que se va a exportar a una cadena XML. Este parámetro es opcional.

Valor devuelto

Una cadena XML.

Descripción

Método; devuelve una cadena que especifica el perfil indicado en formato XML. Si no transfiere un valor para *profileName*, se exportará el perfil actual.

Ejemplo

El siguiente ejemplo almacena una cadena XML que representa el perfil actual en una variable denominada `profileXML` y después la muestra en el panel Salida:

```
var profileXML=fl.getDocumentDOM().exportPublishProfileString();
fl.trace(profileXML);
```

Véase también

`document.exportPublishProfile()`, `document.importPublishProfileString()`

document.exportSWF()

Disponibilidad

Flash MX 2004

Uso

```
document.exportSWF([fileURI [, bCurrentSettings]])
```

Parámetros

`fileURI` Cadena que se expresa en forma de URI file:/// y que especifica el nombre del archivo exportado. Si *fileURI* está vacío o no se especifica, Flash mostrará el cuadro de diálogo Exportar película. Este parámetro es opcional.

bCurrentSettings Un valor booleano que, cuando se define como `true`, hace que Flash utilice la configuración de publicación actual de SWF. En caso contrario, Flash muestra el cuadro de diálogo Exportar Flash Player. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; exporta el documento en formato SWC de Flash.

Ejemplo
El ejemplo siguiente exporta el documento a la ubicación de archivo especificada con la configuración de publicación actual:

```
fl.getDocumentDOM().exportSWF("file:///C|/Documents and
Settings/joe_user/Desktop/qwerty.swf");
```

El ejemplo siguiente muestra los cuadros de diálogo Exportar película y Exportar Flash Player y, a continuación, exporta el documento de acuerdo con la configuración especificada:

```
fl.getDocumentDOM().exportSWF("", true);
```

El ejemplo siguiente muestra el cuadro de diálogo Exportar película y, a continuación, exporta el documento según la configuración especificada:

```
fl.getDocumentDOM().exportSWF();
```

document.externalLibraryPath

Disponibilidad
Flash CS4 Professional

Uso
```
document.externalLibraryPath
```

Descripción
Propiedad; una cadena que contiene una lista de elementos de la ruta de biblioteca externa de ActionScript 3.0 del documento, donde se especifica la ubicación de los archivos SWC utilizados como bibliotecas compartidas de tiempo de ejecución. Los elementos de la cadena se delimitan con punto y coma. En la herramienta de edición, puede especificar elementos desde Archivo > Configuración de publicación y seleccionar Configuración de ActionScript 3.0 en la ficha Flash.

Ejemplo
El ejemplo siguiente define la ruta de biblioteca externa del documento en "." y en "../mySWCLibrary":

```
var myDocument = fl.getDocumentDOM();
myDocument.externalLibraryPath = ".;../mySWCLibrary";
fl.trace(myDocument.externalLibraryPath);
```

Véase también

`document.libraryPath`, `document.sourcePath`, `fl.externalLibraryPath`

document.forceSimple

Disponibilidad

Flash MX 2004

Uso

```
document.forceSimple
```

Descripción

Propiedad; un valor booleano que especifica si los elementos secundarios del objeto especificado son accesibles. Equivale a la lógica inversa de la opción Hacer que los objetos secundarios sean accesibles del panel Accesibilidad. Es decir, si `forceSimple` es `true`, equivale a la opción desactivada Hacer que los objetos secundarios sean accesibles. Si `forceSimple` es `false`, equivale a la opción activada Hacer que los objetos secundarios sean accesibles.

Ejemplo

El ejemplo siguiente define la variable `areChildrenAccessible` con el valor de la propiedad `forceSimple`: El valor `false` indica que los elementos secundarios son accesibles.

```
var areChildrenAccessible = fl.getDocumentDOM().forceSimple;
```

El ejemplo siguiente define la propiedad `forceSimple` para permitir que los elementos secundarios del documento sean accesibles:

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

document.frameRate

Disponibilidad

Flash MX 2004

Uso

```
document.frameRate
```

Descripción

Un valor flotante que especifica el número de fotogramas mostrados por segundo cuando se reproduce el archivo SWF; el valor predeterminado es 12. Ajustar esta propiedad es lo mismo que establecer la velocidad de fotogramas predeterminada en el cuadro de diálogo Propiedades del documento (Modificar > Documento) del archivo FLA.

Ejemplo

El ejemplo siguiente establece la velocidad de reproducción en 25,5 fotogramas por segundo:

```
fl.getDocumentDOM().frameRate = 25.5;
```

document.getAlignToDocument()

Disponibilidad
Flash MX 2004

Uso
`document.getAlignToDocument()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si en las preferencias se define la alineación de objetos con el escenario; `false` en caso contrario.

Descripción
Método; equivale a recuperar el valor del botón En escenario en el panel Alinear. Obtiene la preferencia que puede emplearse para los métodos `document.align()`, `document.distribute()`, `document.match()` y `document.space()` en el documento.

Ejemplo
El ejemplo siguiente recupera el valor del botón En escenario en el panel Alinear. Si el valor devuelto es `true`, el botón En escenario está activo; en caso contrario, no lo está.

```
var isAlignToDoc = fl.getDocumentDOM().getAlignToDocument();
fl.getDocumentDOM().align("left", isAlignToDoc);
```

Véase también
`document.setAlignToDocument()`

document.getBlendMode()

Disponibilidad
Flash 8

Uso
`document.getBlendMode()`

Parámetros
Ninguno.

Valor devuelto
Una cadena que especifica el modo de mezcla para el objeto u objetos seleccionados. Si hay más de un objeto seleccionado y tienen distintos modos de mezcla, la cadena refleja el modo de mezcla del objeto con la profundidad mayor.

Nota: *el valor devuelto es impredecible si la selección contiene objetos que no admiten modos de mezcla o tienen el valor* `"normal"` *de modo de mezcla.*

Descripción

Método; devuelve una cadena que especifica el modo de mezcla para el objeto u objetos seleccionados.

Ejemplo

El ejemplo siguiente muestra el nombre del modo de mezcla en el panel Salida:

```
fl.trace(fl.getDocumentDom().getBlendMode());
```

document.getCustomFill()

Disponibilidad

Flash MX 2004

Uso

```
document.getCustomFill([objectToFill])
```

Parámetros

`objectToFill` Una cadena que especifica la ubicación del objeto de relleno. Los valores siguientes son válidos:

- `"toolbar"` devuelve el objeto de relleno del panel Herramientas y del inspector de propiedades.
- `"selection"` devuelve el objeto de relleno de la selección.

  Si omite este parámetro, el valor predeterminado es `"selection"`. Si no hay selección, el método devuelve `undefined`. Este parámetro es opcional.

Valor devuelto

El Objeto Fill especificado por el parámetro *objectToFill*, si es correcto; en caso contrario, devuelve `undefined`.

Descripción

Método; recupera el objeto de relleno de la forma seleccionada o, si se especifica, del panel Herramientas y del inspector de propiedades.

Ejemplo

El ejemplo siguiente obtiene el objeto de relleno de la selección y, a continuación, cambia a blanco el color de la selección:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = '#FFFFFF';
fill.style = "solid";
fl.getDocumentDOM().setCustomFill(fill);
```

El ejemplo siguiente devuelve el objeto de relleno del panel Herramientas y del inspector de propiedades y, a continuación, cambia la muestra de color a un degradado lineal:

```
var fill = fl.getDocumentDOM().getCustomFill("toolbar");
fill.style = "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray = [0, 100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

Véase también

`document.setCustomFill()`

document.getCustomStroke()

Disponibilidad

Flash MX 2004

Uso

`document.getCustomStroke([locationOfStroke])`

Parámetros

`locationOfStroke` Una cadena que especifica la ubicación del objeto de trazo. Los valores siguientes son válidos:

- `"toolbar"`, si se define, devuelve el objeto de trazo del panel Herramientas y del inspector de propiedades.
- `"selection"`, si se define, devuelve el objeto de trazo de la selección.

  Si omite este parámetro, el valor predeterminado es `"selection"`. Si no hay selección, devuelve `undefined`. Este parámetro es opcional.

Valor devuelto

El Objeto Stroke especificado por el parámetro *locationOfStroke*, si es correcto; en caso contrario, devuelve `undefined`.

Descripción

Devuelve el objeto de trazo de la forma seleccionada o, si se especifica, del panel Herramientas y del inspector de propiedades.

Ejemplo

El ejemplo siguiente devuelve la configuración de trazo actual de la selección y cambia el grosor del trazo a 2:

```
var stroke = fl.getDocumentDOM().getCustomStroke("selection");
stroke.thickness = 2;
fl.getDocumentDOM().setCustomStroke(stroke);
```

El ejemplo siguiente devuelve la configuración de trazo actual del panel Herramientas y del inspector de propiedades y establece el color de trazo en rojo:

```
var stroke = fl.getDocumentDOM().getCustomStroke("toolbar");
stroke.color = "#FF0000";
fl.getDocumentDOM().setCustomStroke(stroke);
```

Véase también

`document.setCustomStroke()`

document.getDataFromDocument()

Disponibilidad
Flash MX 2004

Uso
`document.getDataFromDocument(name)`

Parámetros
`name` Una cadena que especifica el nombre de los datos que se van a devolver.

Valor devuelto
Los datos especificados.

Descripción
Método; recupera el valor de los datos especificados. El tipo devuelto depende del tipo de datos que estaba almacenado.

Ejemplo
El ejemplo siguiente añade un valor entero de 12 al documento actual y utiliza este método para mostrar el valor en el panel Salida:

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

Véase también
`document.addDataToDocument()`, `document.documentHasData()`, `document.removeDataFromDocument()`

document.getElementProperty()

Disponibilidad
Flash MX 2004

Uso
`document.getElementProperty(propertyName)`

Parámetros
`propertyName` Una cadena que especifica el nombre de la propiedad Element para la que se va a recuperar el valor.

Valor devuelto
Valor de la propiedad especificada. Devuelve `null` si la propiedad es un estado indeterminado, como cuando se seleccionan múltiples elementos con distintos valores de propiedad. Devuelve `undefined` si la propiedad no es válida para el elemento seleccionado.

Descripción
Método; obtiene la propiedad `Element` especificada para la selección actual. Para ver una lista de los valores aceptables, consulte la tabla del resumen de propiedades del Objeto Element.

Ejemplo

El ejemplo siguiente obtiene el `nombre` de la propiedad Element para la selección actual:

```
// elementName = the instance name of the selected object.
var elementName = fl.getDocumentDOM().getElementProperty("name");
```

Véase también

`document.setElementProperty()`

document.getElementTextAttr()

Disponibilidad

Flash MX 2004

Uso

```
document.getElementTextAttr(attrName [, startIndex [, endIndex]])
```

Parámetros

`attrName` Una cadena que especifica el nombre de la propiedad del objeto `TextAttrs` que se va a devolver. Para ver una lista de nombres de propiedades y valores esperados, consulte la tabla del resumen de propiedades del Objeto TextAttrs.

`startIndex` Un entero que especifica el índice del primer carácter, con un 0 (cero) que especifica la primera posición. Este parámetro es opcional.

`endIndex` Un entero que especifica el índice del último carácter. Este parámetro es opcional.

Valor devuelto

Si hay un campo de texto seleccionado, la propiedad se devuelve si sólo se utiliza un valor en el texto. Devuelve `undefined` si se utilizan varios valores en el campo de texto. Si hay varios campos de texto seleccionados y todos los valores de alineación de texto son iguales, el método devuelve este valor. Si hay varios campos de texto seleccionados, pero no todos los valores de alineación de texto son iguales, el método devuelve `undefined`. Si no se transfieren los argumentos opcionales, estas reglas se aplican al rango de texto seleccionado actualmente o a todo el campo de texto si no se está editando el texto. Si sólo se transfiere *startIndex*, se devuelve la propiedad del carácter a la derecha del índice, si todos los objetos de texto seleccionado coinciden con los valores. Si se transfieren *startIndex* y *endIndex*, el valor devuelto refleja toda la gama de caracteres desde *startIndex* hasta *endIndex* (no incluido).

Descripción

Método; obtiene una propiedad `TextAttrs` específica de los objetos de texto seleccionados. Se ignorarán los objetos seleccionados que no sean campos de texto. Para ver una lista de nombres de propiedades y valores esperados, consulte la tabla del resumen de propiedades del Objeto TextAttrs. Consulte también `document.setElementTextAttr()`.

Ejemplo

El ejemplo siguiente obtiene el tamaño de los campos de texto seleccionados:

```
fl.getDocumentDOM().getElementTextAttr("size");
```

El ejemplo siguiente obtiene el color del carácter en el índice 3 en los campos de texto seleccionados:

```
fl.getDocumentDOM().getElementTextAttr("fillColor", 3);
```

El ejemplo siguiente obtiene el nombre de la fuente del texto desde el índice 2 hasta el índice 10 (no incluido) de los campos de texto seleccionados:

```
fl.getDocumentDOM().getElementTextAttr("face", 2, 10);
```

document.getFilters()

Disponibilidad
Flash 8

Uso
`document.getFilters()`

Parámetros
Ninguno.

Valor devuelto
Un conjunto que contiene la lista de filtros aplicados al objeto u objetos seleccionados actualmente.

Descripción
Método; devuelve un conjunto que contiene la lista de filtros aplicados al objeto u objetos seleccionados actualmente. Si se seleccionan varios objetos y no tienen filtros idénticos, este método devuelve la lista de filtros aplicados al primer objeto seleccionado.

Ejemplo
Consulte `document.setFilters()`.

Véase también
`document.addFilter()`, `document.changeFilterOrder()`, `document.setFilters()`, Objeto Filter

document.getMetadata()

Disponibilidad
Flash 8

Uso
`document.getMetadata()`

Parámetros
Ninguno.

Valor devuelto
Una cadena que contiene los metadatos XML asociados al documento, o una cadena vacía si no hay metadatos.

Descripción

Método; devuelve una cadena que contiene los metadatos XML asociados al documento, o una cadena vacía si no hay metadatos.

Ejemplo

El ejemplo siguiente muestra metadatos XML del documento actual en el panel Salida:

```
fl.trace("XML Metadata is :" + fl.getDocumentDOM().getMetadata());
```

Véase también

`document.setMetadata()`

document.getMobileSettings()

Disponibilidad

Flash CS3 Professional

Uso

```
document.getMobileSettings()
```

Parámetros

Ninguno.

Valor devuelto

Una cadena que representa la configuración XML del documento. Si no se ha establecido ningún valor, devuelve una cadena vacía.

Descripción

Método; devuelve la configuración XML móvil del documento.

Ejemplo

El ejemplo siguiente muestra la configuración XML del documento actual:

```
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>"
```

Véase también

`document.setMobileSettings()`

document.getPlayerVersion()

Disponibilidad

Flash CS3 Professional

Uso

`document.getPlayerVersion()`

Parámetros

Ninguno.

Valor devuelto

Una cadena que representa la versión de Flash Player especificada mediante `document.setPlayerVersion()`. Si no se ha establecido ningún valor, devuelve el valor especificado en el cuadro de diálogo Configuración de publicación.

Descripción

Método; devuelve una cadena que representa la versión del reproductor de destino del documento especificado. Para obtener una lista de valores que puede devolver este método, consulte `document.setPlayerVersion()`.

Para determinar qué versión de ActionScript tiene el archivo especificado, utilice `document.asVersion`.

Ejemplo

El ejemplo siguiente ilustra la búsqueda de versiones especificadas del reproductor para el documento actual y la recuperación de dichos valores:

```
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""
```

Véase también

`document.setPlayerVersion()`

document.getSelectionRect()

Disponibilidad

Flash MX 2004

Uso

`document.getSelectionRect()`

Parámetros

Ninguno.

Valor devuelto

El rectángulo de delimitación de la selección actual o 0 si no hay nada seleccionado. Para más información sobre el formato del valor devuelto, consulte `document.addNewRectangle()`.

Descripción

Método; obtiene el rectángulo de delimitación de la selección actual. Si la selección no es rectangular, se devuelve el rectángulo más pequeño que abarque la selección completa. El rectángulo se basa en el espacio de documento o, cuando se encuentra en modo de edición, el punto de registro (también *punto de origen* o *punto cero*) del símbolo que se está editando.

Ejemplo

El ejemplo siguiente obtiene el rectángulo de delimitación para la selección actual y, a continuación, muestra sus propiedades:

```
var newRect = fl.getDocumentDOM().getSelectionRect();
var outputStr = "left: " + newRect.left + " top: " + newRect.top + " right: " + newRect.right
+ " bottom: " + newRect.bottom;
alert(outputStr);
```

Véase también

`document.selection`, `document.setSelectionRect()`

document.getTextString()

Disponibilidad

Flash MX 2004

Uso

```
document.getTextString([startIndex [, endIndex]])
```

Parámetros

`startIndex` Un entero que es un índice del primer carácter que se va a obtener. Este parámetro es opcional.

`endIndex` Un entero que es un índice del último carácter que se va a obtener. Este parámetro es opcional.

Valor devuelto

Una cadena que contiene el texto seleccionado.

Descripción

Método; obtiene el texto seleccionado actualmente. Si no se transfieren los parámetros opcionales, se utilizará la selección de texto actual. Si el texto no está abierto para editar, se devolverá la cadena de texto completa. Si sólo se transfiere *startIndex*, se devolverá la cadena que comienza en ese índice y que termina al final del campo. Si se transfieren *startIndex* y *endIndex*, se devolverá la cadena que comienza desde *startIndex* hasta *endIndex* (no incluido).

Si hay varios campos de texto seleccionados, se devolverá la concatenación de todas las cadenas.

Ejemplo

El ejemplo siguiente obtiene la cadena de los campos de texto seleccionados:

```
fl.getDocumentDOM().getTextString();
```

El ejemplo siguiente obtiene la cadena en el índice de carácter 5 de los campos de texto seleccionados:

```
fl.getDocumentDOM().getTextString(5);
```

El ejemplo siguiente obtiene la cadena desde el índice de carácter 2 hasta el índice de carácter 10 (no incluido):

```
fl.getDocumentDOM().getTextString(2, 10);
```

Véase también

`document.setTextString()`

document.getTimeline()

Disponibilidad

Flash MX 2004

Uso

```
document.getTimeline()
```

Parámetros

Ninguno.

Valor devuelto

El objeto Timeline actual.

Descripción

Método; recupera el Objeto Timeline actual en el documento. El objeto de línea de tiempo actual puede ser la escena actual, el símbolo que se está editando o la pantalla actual.

Ejemplo

El ejemplo siguiente obtiene el objeto Timeline y devuelve el número de fotogramas en la capa más larga:

```
var longestLayer = fl.getDocumentDOM().getTimeline().frameCount;
fl.trace("The longest layer has" + longestLayer + "frames");
```

El ejemplo siguiente entra en el modo de edición en contexto para el símbolo seleccionado en el escenario e inserta un fotograma en la línea de tiempo del símbolo.

```
fl.getDocumentDOM().enterEditMode("inPlace");
fl.getDocumentDOM().getTimeline().insertFrames();
```

El ejemplo siguiente obtiene el objeto Timeline y muestra su nombre:

```
var timeline = fl.getDocumentDOM().getTimeline();
alert(timeline.name);
```

Véase también

`document.currentTimeline`, `document.timelines`, `symbolItem.timeline`

document.getTransformationPoint()

Disponibilidad
Flash MX 2004

Uso
`document.getTransformationPoint()`

Parámetros
Ninguno.

Valor devuelto
Un punto (por ejemplo, `{x:10, y:20}`, donde `x` e `y` son números de coma flotante) que especifica la posición del punto de transformación (también *punto de origen* o *punto cero*) en el sistema de coordenadas del elemento seleccionado.

Descripción
Método; obtiene la ubicación del punto de transformación de la selección actual. Puede emplear el punto de transformación para transformaciones como rotar y sesgar.

Nota: *los puntos de transformación tendrán distintas ubicaciones, según el tipo de elemento seleccionado. Para más información, consulte* `document.setTransformationPoint()`.

Ejemplo
El ejemplo siguiente obtiene el punto de transformación para la selección actual. La propiedad `transPoint.x` proporciona la coordenada *x* del punto de transformación. La propiedad `transPoint.y` proporciona la coordenada *y* del punto de transformación.

```
var transPoint = fl.getDocumentDOM().getTransformationPoint();
```

Véase también
`document.setTransformationPoint()`, `element.getTransformationPoint()`

document.group()

Disponibilidad
Flash MX 2004

Uso
`document.group()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; convierte la selección actual en un grupo.

Ejemplo
El ejemplo siguiente convierte los objetos de la selección actual en un grupo:

```
fl.getDocumentDOM().group();
```

Véase también
`document.unGroup()`

document.height

Disponibilidad
Flash MX 2004

Uso
`document.height`

Descripción
Propiedad; un entero que especifica la altura del documento (escenario) en píxeles.

Ejemplo
El ejemplo siguiente establece la altura del escenario en 400 píxeles:

```
fl.getDocumentDOM().height = 400;
```

Véase también
`document.width`

document.id

Disponibilidad
Flash CS3 Professional

Uso
`document.id`

Descripción
Propiedad de sólo lectura; un entero exclusivo (asignado automáticamente) que identifica a un documento durante una sesión de Flash. Utilice esta propiedad junto con `fl.findDocumentDOM()` para especificar la acción de un documento determinado.

Ejemplo
El ejemplo siguiente muestra el ID del documento actual:

```
fl.trace("Current doc's internal ID is: " + fl.getDocumentDOM().id);
```

Véase también

`fl.findDocumentDOM()`

document.importFile()

Disponibilidad

Flash 8

Uso

```
document.importFile(fileURI [, importToLibrary])
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta del archivo que se va a importar.

`importToLibrary` Un valor booleano que especifica si se importará el archivo sólo a la biblioteca del documento (`true`) o se colocará también una copia en el escenario (`false`). El valor predeterminado es `false`.

Valor devuelto

Ninguno.

Descripción

Método; importa un archivo a un documento. Este método realiza la misma operación que el comando del menú Importar a biblioteca o Importar a escenario. Para importar un perfil de publicación, utilice `document.importPublishProfile()`.

Ejemplo

El siguiente ejemplo permite al usuario localizar el archivo que se importará al escenario:

```
var dom = fl.getDocumentDOM();
var URI = fl.browseForFileURL("select", "Import File");
dom.importFile(URI);
```

Véase también

`document.importSWF()`, `fl.browseForFileURL()`

document.importPublishProfile()

Disponibilidad

Flash MX 2004

Uso

```
document.importPublishProfile( fileURI )
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta del archivo XML que define el perfil que se va a importar.

Valor devuelto

Un entero que es el índice del archivo importado en la lista de perfiles. Devuelve -1 si no se puede importar el perfil.

Descripción

Método; importa un perfil desde un archivo.

Ejemplo

El ejemplo siguiente importa el perfil que contiene el archivo profile.xml y muestra su índice en la lista de perfiles:

```
alert(fl.getDocumentDOM().importPublishProfile('file:///C|/Documents and
Settings/janeUser/Desktop/profile.xml'));
```

document.importPublishProfileString()

Disponibilidad

Flash CS4 Professional

Uso

```
document.importPublishProfileString(xmlString)
```

Parámetros

`xmlString` Una cadena que contiene datos XML para importarlos como perfil actual.

Valor devuelto

Un valor booleano de `true` si la cadena se importa correctamente, y de `false` en caso contrario.

Descripción

Método; importa una cadena XML que representa un perfil de publicación y lo establece como perfil actual. Para generar una cadena XML e importarla, utilice `document.exportPublishProfileString()` antes de utilizar este método.

Ejemplo

En el siguiente ejemplo, el perfil predeterminado se exporta como una cadena XML. El comando `replace` estándar de JavaScript se utiliza para modificar la cadena XML. La cadena se importa y la configuración de salida predeterminada de ActionScript 3 se establece en 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()

Disponibilidad
Flash MX 2004

Uso
`document.importSWF(fileURI)`

Parámetros
`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo SWF que desea importar.

Valor devuelto
Ninguno.

Descripción
Método; importa un archivo SWF en el documento. Este método equivale a utilizar el comando de menú Importar para especificar un archivo SWF. En Flash 8 y posterior, también puede utilizar `document.importFile()` para importar un archivo SWF (así como otros tipos de archivos).

Ejemplo
El ejemplo siguiente importa el archivo `"mySwf.swf"` desde la carpeta Configuration de Flash:

```
fl.getDocumentDOM().importSWF(fl.configURI+"mySwf.swf");
```

Véase también
`document.importFile()`

document.intersect()

Disponibilidad
Flash 8

Uso
`document.intersect()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción
Método; crea un objeto de dibujo de intersección a partir de todos los objetos de dibujo seleccionados. Este método devuelve `false` si no hay objetos de dibujo seleccionados o si alguno de los elementos seleccionados no es un objeto de dibujo.

Ejemplo

El ejemplo siguiente crea un objeto de dibujo de intersección a partir de todos los objetos de dibujo seleccionados:

```
fl.getDocumentDOM().intersect();
```

Véase también

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

document.library

Disponibilidad

Flash MX 2004

Uso

```
document.library
```

Descripción

Propiedad de sólo lectura; el Objeto library para un documento.

Ejemplo

El ejemplo siguiente obtiene la biblioteca para el documento seleccionado actualmente:

```
var myCurrentLib = fl.getDocumentDOM().library;
```

Suponiendo que el documento seleccionado actualmente no es `fl.documents[1]`, el ejemplo siguiente obtiene la biblioteca para una biblioteca no seleccionada o una que se abrió a través de Archivo > Abrir como biblioteca externa:

```
var externalLib = fl.documents[1].library;
```

document.libraryPath

Disponibilidad

Flash CS4 Professional

Uso

```
document.libraryPath
```

Descripción

Propiedad; una cadena que contiene una lista de elementos de la ruta de biblioteca de ActionScript 3.0 del documento, donde se especifica la ubicación de los archivos SWC o de las carpetas que contienen los archivos SWC. Los elementos de la cadena se delimitan con punto y coma. En la herramienta de edición, puede especificar elementos desde Archivo > Configuración de publicación y seleccionar Configuración de ActionScript 3.0 en la ficha Flash.

Ejemplo

El siguiente ejemplo añade la carpeta ../Files a la ruta de biblioteca del documento y muestra la ruta de la biblioteca en el panel Salida:

```
var myDoc = fl.getDocumentDOM()
fl.trace(myDoc.libraryPath);
myDoc.libraryPath = "../Files;" + myDoc.libraryPath;
fl.trace(myDoc.libraryPath);
```

Véase también

`document.externalLibraryPath`,`document.sourcePath`, `fl.libraryPath`

document.livePreview

Disponibilidad

Flash MX 2004

Uso

```
document.livePreview
```

Descripción

Propiedad; un valor booleano que especifica si está activada la opción Vista previa dinámica. Si se define como `true`, los componentes aparecen en el escenario tal como aparecerán en el contenido publicado, incluido su tamaño aproximado. Si se define como `false`, los componentes sólo aparecerán como contornos. El valor predeterminado es `true`.

Ejemplo

El ejemplo siguiente define Vista previa dinámica como `false`:

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

document.match()

Disponibilidad

Flash MX 2004

Uso

```
document.match(bWidth, bHeight [, bUseDocumentBounds])
```

Parámetros

`bWidth` Un valor booleano que, cuando se define como `true`, hace que el método iguale la anchura de los elementos seleccionados.

`bHeight` Un valor booleano que, cuando se define como `true`, hace que el método iguale la altura de los elementos seleccionados.

`bUseDocumentBounds` Un valor booleano que, si se define como `true`, hace que el método iguale el tamaño de los objetos con los límites del documento. En caso contrario, el método utiliza los límites del objeto mayor. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción
Método; iguala el tamaño de los objetos seleccionados.

Ejemplo
El ejemplo siguiente sólo iguala la anchura de los objetos seleccionados:

```
fl.getDocumentDOM().match(true,false);
```

El ejemplo siguiente sólo iguala la altura:

```
fl.getDocumentDOM().match(false,true);
```

El ejemplo siguiente sólo iguala la anchura a los límites del documento:

```
fl.getDocumentDOM().match(true,false,true);
```

Véase también
`document.getAlignToDocument()`, `document.setAlignToDocument()`

document.mouseClick()

Disponibilidad
Flash MX 2004

Uso
```
document.mouseClick(position, bToggleSel, bShiftSel)
```

Parámetros
`position` Un par de valores de coma flotante que especifican las coordenadas *x* e *y* del clic en píxeles.

`bToggleSel` Un valor booleano que especifica el estado de la tecla Mayús: es `true` si está pulsada; `false` si no lo está.

`bShiftSel` Un valor booleano que especifica el estado de la preferencia de selección de la tecla Mayús en la aplicación: `true` si está activada; `false` si está desactivada.

Valor devuelto
Ninguno.

Descripción
Método; ejecuta un clic de ratón desde la herramienta Selección.

Ejemplo
El ejemplo siguiente ejecuta un clic de ratón en la ubicación especificada:

```
fl.getDocumentDOM().mouseClick({x:300, y:200}, false, false);
```

Véase también
`document.mouseDblClk()`

document.mouseDblClk()

Disponibilidad
Flash MX 2004

Uso
`document.mouseDblClk(position, bAltDown, bShiftDown, bShiftSelect)`

Parámetros
`position` Un par de valores de coma flotante que especifican las coordenadas *x* e *y* del clic en píxeles.

`bAltdown` Un valor booleano que registra si la tecla Alt está presionada en el momento del evento: `true` si está pulsada; `false` si no lo está.

`bShiftDown` Un valor booleano que registra si la tecla Mayús está presionada en el momento del evento: `true` si está pulsada; `false` si no lo está.

`bShiftSelect` Un valor booleano que indica el estado de la preferencia de selección de la tecla Mayús en la aplicación: `true` si está activada; `false` si está desactivada.

Valor devuelto
Ninguno.

Descripción
Método; ejecuta un doble clic de ratón desde la herramienta Selección.

Ejemplo
El ejemplo siguiente ejecuta un doble clic de ratón en la ubicación especificada:

```
fl.getDocumentDOM().mouseDblClk({x:392.9, y:73}, false, false, true);
```

Véase también
`document.mouseClick()`

document.moveSelectedBezierPointsBy()

Disponibilidad
Flash MX 2004

Uso
`document.moveSelectedBezierPointsBy(delta)`

Parámetros
`delta` Un par de valores de coma flotante que especifican las coordenadas *x* e *y* en píxeles a las que se mueven los puntos Bézier seleccionados. Por ejemplo, (`{x:1,y:2}`) especifica una ubicación que se encuentra un píxel hacia la derecha y dos píxeles hacia abajo respecto a la ubicación actual.

Valor devuelto
Ninguno.

Descripción
Método; si la selección contiene como mínimo un trazado con al menos un punto Bézier seleccionado, este método mueve todos los puntos Bézier seleccionados en todos los trazados seleccionados con la cantidad especificada.

Ejemplo
El ejemplo siguiente mueve los puntos Bézier seleccionados 10 píxeles hacia la derecha y 5 píxeles hacia abajo:

```
fl.getDocumentDOM().moveSelectedBezierPointsBy({x:10, y:5});
```

document.moveSelectionBy()

Disponibilidad
Flash MX 2004

Uso
```
document.moveSelectionBy(distanceToMove)
```

Parámetros
`distanceToMove` Un par de valores de coma flotante que especifican las coordenadas *x* e *y* a las que el método mueve la selección. Por ejemplo, (`{x:1,y:2}`) especifica una ubicación que se encuentra un píxel hacia la derecha y dos píxeles hacia abajo respecto a la ubicación actual.

Valor devuelto
Ninguno.

Descripción
Método; mueve los objetos seleccionados una distancia especificada.

Nota: *cuando el usuario utiliza las teclas de flecha para mover el elemento, el panel Historial combina todas las pulsaciones de la tecla de flecha como un solo paso. Cuando el usuario presiona las teclas de flecha repetidamente, en lugar de dar varios pasos en el panel Historial, este método ejecuta un paso, y los argumentos se actualizan para reflejar las teclas de flecha repetidas.*

Para más información sobre cómo realizar una selección, consulte `document.setSelectionRect()`, `document.mouseClick()`, `document.mouseDblClk()` y el Objeto Element.

Ejemplo
El ejemplo siguiente mueve el elemento seleccionado 62 píxeles hacia la derecha y 84 píxeles hacia abajo:

```
fl.getDocumentDOM().moveSelectionBy({x:62, y:84});
```

document.name

Disponibilidad
Flash MX 2004

Uso
`document.name`

Descripción
Propiedad de sólo lectura; una cadena que representa el nombre de un documento (archivo FLA).

Ejemplo
El ejemplo siguiente establece la variable `fileName` con el nombre de archivo del primer documento del conjunto de documentos:

```
var fileName = flash.documents[0].name;
```

El ejemplo siguiente muestra los nombres de todos los documentos abiertos en el panel Salida:

```
var openDocs = fl.documents;
for(var i=0;i < openDocs.length; i++){
    fl.trace(i + " " + openDocs[i].name +"\n");
}
```

document.optimizeCurves()

Disponibilidad
Flash MX 2004

Uso
`document.optimizeCurves(smoothing, bUseMultiplePasses)`

Parámetros
`smoothing` Un entero entre 0 y 100, donde 0 especifica sin suavizado y 100, suavizado máximo.

`bUseMultiplePasses` Un valor booleano que, cuando se define como `true`, indica que el método debe emplear varias pasadas, lo que resulta más lento aunque ofrece mejores resultados. Este parámetro equivale a hacer clic en el botón Utilizar varias pasadas en el cuadro de diálogo Optimizar curvas.

Valor devuelto
Ninguno.

Descripción
Método; optimiza el suavizado para la selección actual, lo que permite múltiples pasadas, si se especifican, para un suavizado óptimo. Este método equivale a seleccionar Modificar > Forma > Optimizar.

Ejemplo
El ejemplo siguiente optimiza la curva de la selección actual a 50º de suavizado con varias pasadas:

```
fl.getDocumentDOM().optimizeCurves(50, true);
```

document.path

Disponibilidad

Flash MX 2004

Uso

```
document.path
```

Descripción

Propiedad de sólo lectura; una cadena que representa la ruta del documento con el formato específico de la plataforma. Si el documento no se ha guardado nunca, esta propiedad es `undefined`.

Ejemplo

El ejemplo siguiente muestra la ruta del primer documento del conjunto de documentos en el panel Salida. Debe guardar el documento antes de ejecutar el script. En el ejemplo, el archivo se llama test.fla y se guarda en la carpeta Mis documentos de un ordenador con Windows.

```
var filePath = flash.documents[0].path;
fl.trace(filePath);
// displays C:\Documents and Settings\<user name>\My Documents\test.fla
```

Véase también

`document.pathURI`

document.pathURI

Disponibilidad

Flash CS4 Professional

Uso

```
document.pathURI
```

Descripción

Propiedad de sólo lectura; una cadena que representa la ruta del documento, expresada como URI file:///. Si el documento no se ha guardado nunca, esta propiedad es `undefined`.

Ejemplo

El ejemplo siguiente muestra la ruta del primer documento del conjunto de documentos como una cadena URI file:/// en el panel Salida. Debe guardar el documento antes de ejecutar el script. En el ejemplo, el archivo se llama test.fla y se guarda en la carpeta Mis documentos de un ordenador con Windows.

```
var filePathURI = flash.documents[0].pathURI;
fl.trace(filePathURI);
// displays file:///C|/Documents%20and%20Settings/<userName>/My%20Documents/test.fla
```

Véase también

`document.path`

document.publish()

Disponibilidad

Flash MX 2004

Uso

```
document.publish()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; publica el documento de acuerdo con la Configuración de publicación activa (Archivo > Configuración de publicación). Este método equivale a seleccionar Archivo > Publicar.

Ejemplo

El ejemplo siguiente publica el documento actual:

```
fl.getDocumentDOM().publish();
```

document.publishProfiles

Disponibilidad

Flash MX 2004

Uso

```
document.publishProfiles
```

Descripción

Propiedad de sólo lectura; un conjunto de los nombres del perfil de publicación para el documento.

Ejemplo

El ejemplo siguiente muestra los nombres de los perfiles de publicación para el documento:

```
var myPubProfiles = fl.getDocumentDOM().publishProfiles;
for (var i=0; i < myPubProfiles.length; i++){
     fl.trace(myPubProfiles[i]);
}
```

document.punch()

Disponibilidad
Flash 8

Uso
`document.punch()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción
Método; utiliza el objeto de dibujo seleccionado en la parte superior para perforar todos los objetos de dibujo seleccionados por debajo. Este método devuelve `false` si no hay objetos de dibujo seleccionados o si alguno de los elementos seleccionados no es un objeto de dibujo.

Ejemplo
El ejemplo siguiente perfora los objetos de dibujo situados por debajo del objeto de dibujo seleccionado:

```
fl.getDocumentDOM().punch();
```

Véase también
`document.crop()`, `document.deleteEnvelope()`, `document.intersect()`, `document.union()`, `shape.isDrawingObject`

document.removeAllFilters()

Disponibilidad
Flash 8

Uso
`document.removeAllFilters()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; elimina todos los filtros del objeto u objetos seleccionados.

Ejemplo

El ejemplo siguiente elimina todos los filtros del objeto u objetos seleccionados:

```
fl.getDocumentDOM().removeAllFilters();
```

Véase también

`document.addFilter()`, `document.changeFilterOrder()`, `document.disableAllFilters()`, `document.getFilters()`, `document.removeFilter()`, Objeto Filter

document.removeDataFromDocument()

Disponibilidad

Flash MX 2004

Uso

```
document.removeDataFromDocument(name)
```

Parámetros

`name` Una cadena que especifica el nombre de los datos que se van a eliminar.

Valor devuelto

Ninguno.

Descripción

Método; elimina datos persistentes con el nombre especificado que se han asociado al documento.

Ejemplo

El ejemplo siguiente elimina del documento los datos persistentes con el nombre `"myData"`:

```
fl.getDocumentDOM().removeDataFromDocument("myData");
```

Véase también

`document.addDataToDocument()`, `document.documentHasData()`, `document.getDataFromDocument()`

document.removeDataFromSelection()

Disponibilidad

Flash MX 2004

Uso

```
document.removeDataFromSelection(name)
```

Parámetros

`name` Una cadena que especifica el nombre de los datos persistentes que se van a eliminar.

Valor devuelto
Ninguno.

Descripción
Método; elimina datos persistentes con el nombre especificado que se han asociado a la selección.

Ejemplo
El ejemplo siguiente elimina de la selección los datos persistentes con el nombre `"myData"`:

```
fl.getDocumentDOM().removeDataFromSelection("myData");
```

Véase también
`document.addDataToSelection()`

document.removeFilter()

Disponibilidad
Flash 8

Uso
```
document.removeFilter(filterIndex)
```

Parámetros
`filterIndex` Un entero que especifica el índice basado en cero del filtro que se va a eliminar del objeto u objetos seleccionados.

Valor devuelto
Ninguno.

Descripción
Método; elimina el filtro especificado de la lista Filtros del objeto u objetos seleccionados.

Ejemplo
El ejemplo siguiente elimina el primer filtro (valor de índice de 0) de la lista Filtros del objeto u objetos seleccionados:

```
fl.getDocumentDOM().removeFilter(0);
```

Véase también
`document.addFilter()`, `document.changeFilterOrder()`, `document.disableFilter()`, `document.getFilters()`, `document.removeAllFilters()`, Objeto Filter

document.renamePublishProfile()

Disponibilidad
Flash MX 2004

Uso

```
document.renamePublishProfile([profileNewName])
```

Parámetros

`profileNewName` Un parámetro opcional que especifica el nuevo nombre del perfil. El nombre nuevo debe ser único. Si no se especifica el nombre, se suministra un nombre predeterminado.

Valor devuelto

Valor booleano: `true` si el nombre se modifica correctamente, y `false` en caso contrario.

Descripción

Método; cambia el nombre del perfil actual.

Ejemplo

El ejemplo siguiente cambia el nombre del perfil actual por un nombre predeterminado y lo muestra:

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

document.renameScene()

Disponibilidad

Flash MX 2004

Uso

```
document.renameScene(name)
```

Parámetros

`name` Una cadena que especifica el nuevo nombre de la escena.

Valor devuelto

Valor booleano: `true` si el nombre se modifica correctamente, y `false` en caso contrario. Si el nuevo nombre no es único, por ejemplo, el método devuelve `false`.

Descripción

Método; cambia el nombre de la escena seleccionada actualmente en el panel Escenas. El nuevo nombre de la escena seleccionada debe ser único.

Ejemplo

El ejemplo siguiente cambia el nombre de la escena actual por `"new name"`:

```
var success = fl.getDocumentDOM().renameScene("new name");
```

document.reorderScene()

Disponibilidad
Flash MX 2004

Uso
```
document.reorderScene(sceneToMove, sceneToPutItBefore)
```

Parámetros
`sceneToMove` Un entero que especifica qué escena se va a mover, donde 0 (cero) es la primera escena.

`sceneToPutItBefore` Un entero que especifica la escena antes de la cual desea mover la escena especificada por *sceneToMove*. Especifique 0 (cero) para la primera escena. Por ejemplo, si especifica 1 para *sceneToMove* y 0 para *sceneToPutItBefore*, la segunda escena se situará delante de la primera. Especifique -1 para mover la escena al final.

Valor devuelto
Ninguno.

Descripción
Método; mueve la escena especificada delante de otra escena especificada.

Ejemplo
El ejemplo siguiente mueve la segunda escena delante de la primera:

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

document.resetOvalObject()

Disponibilidad
Flash CS3 Professional

Uso
```
document.resetOvalObject()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; establece todos los valores del inspector de propiedades con la configuración predeterminada del objeto Oval. Si hay algún objeto Oval seleccionado, sus propiedades se restablecen también con los valores predeterminados.

Ejemplo
El ejemplo siguiente restablece las propiedades del objeto Oval del documento actual con los valores predeterminados:

```
fl.getDocumentDOM().resetOvalObject();
```

Véase también

document.resetRectangleObject()

document.resetRectangleObject()

Disponibilidad

Flash CS3 Professional

Uso

```
document.resetRectangleObject()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; establece todos los valores del inspector de propiedades con la configuración predeterminada del objeto Rectangle. Si hay algún objeto Rectangle seleccionado, sus propiedades se restablecen también con los valores predeterminados.

Ejemplo

El ejemplo siguiente restablece las propiedades del objeto Rectangle del documento actual con los valores predeterminados:

```
fl.getDocumentDOM().resetRectangleObject();
```

Véase también

document.resetOvalObject()

document.resetTransformation()

Disponibilidad

Flash MX 2004

Uso

```
document.resetTransformation()
```

Parámetros

Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; restablece la matriz de transformación. Este método equivale a seleccionar Modificar > Transformar > Quitar transformación.

Ejemplo
El ejemplo siguiente restablece la matriz de transformación para la selección actual:

```
fl.getDocumentDOM().resetTransformation();
```

document.revert()

Disponibilidad
Flash MX 2004

Uso
`document.revert()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; devuelve el documento especificado a la versión guardada anteriormente. Este método equivale a seleccionar Archivo > Descartar cambios.

Ejemplo
El ejemplo siguiente devuelve el documento actual a la versión guardada anteriormente:

```
fl.getDocumentDOM().revert();
```

Véase también
`document.canRevert()`, `fl.revertDocument()`

document.revertToLastVersion()

Disponibilidad
Flash CS3 Professional

Uso
`document.revertToLastVersion()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano de `true` si se descartan los cambios del documento, y de `false` en caso contrario.

Descripción
Método; si se pueden descartar los cambios del archivo, se muestra un cuadro de diálogo para que el usuario pueda confirmar que se descartarán los cambios. Si el usuario lo confirma, este método devuelve el archivo a la versión guardada en el servidor Version Cue y registra cualquier error en el panel Salida.

Ejemplo
El ejemplo siguiente devuelve el documento actual a la versión almacenada en el servidor Version Cue:

```
fl.getDocumentDOM().revertToLastVersion();
```

Véase también
`document.canSaveAVersion()`, `document.saveAVersion()`, `document.synchronizeWithHeadVersion()`, `fl.revertDocumentToLastVersion()`

document.rotate3DSelection()

Disponibilidad
Flash CS4 Professional

Uso
```
document.rotate3DSelection(xyzCoordinate, bGlobalTransform)
```

Parámetros
`xyzCoordinate` Un punto de coordenadas XYZ que especifica los ejes para la rotación 3D.

`bGlobalTransform` Un valor booleano que especifica si el modo de transformación debe ser global (`true`) o local (`false`).

Valor devuelto
Ninguno.

Descripción
Método; aplica un giro 3D a la selección. Este método sólo está disponible para clips de película.

Ejemplo
En el siguiente ejemplo, la selección se gira primero en relación al escenario (globalmente) y, después, con respecto a sí misma (localmente).

```
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()

Disponibilidad
Flash MX 2004

Uso
```
document.rotateSelection(angle [, rotationPoint])
```

Parámetros
`angle` Un valor de coma flotante que especifica el ángulo de rotación.

`rotationPoint` Una cadena que especifica qué lado del recuadro de delimitación va a rotar. Los valores aceptables son `"top right"`, `"top left"`, `"bottom right"`, `"bottom left"`, `"top center"`, `"right center"`, `"bottom center"` y `"left center"`. Si no se especifica, el método utiliza el punto de transformación. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; gira la selección el número de grados especificado. Se obtiene el mismo efecto que con la herramienta Transformación libre para girar el objeto.

Ejemplo
El ejemplo siguiente gira la selección 45º alrededor del punto de transformación:

```
fl.getDocumentDOM().rotateSelection(45);
```

El ejemplo siguiente gira la selección 45º alrededor de la esquina inferior izquierda:

```
fl.getDocumentDOM().rotateSelection(45, "bottom left");
```

document.save()

Disponibilidad
Flash MX 2004

Uso
```
document.save([bOkToSaveAs])
```

Parámetros
`bOkToSaveAs` Un parámetro opcional que, si se define como `true` o se omite y no se ha guardado nunca el archivo, abre el cuadro de diálogo Guardar como. Si se define como `false` y el archivo no se ha guardado nunca, el archivo no se guardará.

Valor devuelto
Un valor booleano: `true` si la operación de guardado se realiza correctamente; `false` en caso contrario.

Descripción

Método; guarda el documento en su ubicación predeterminada. Este método equivale a seleccionar Archivo > Guardar.

Para especificar un nombre para el archivo (en lugar de guardarlo con el mismo nombre), utilice `fl.saveDocument()`.

Nota: *si el archivo es nuevo y no se ha modificado ni guardado, o bien no se ha modificado desde la última vez que se guardó, este método no tiene ningún efecto y el valor devuelto será* `false`*. Para permitir que se guarde un archivo no guardado o no modificado, utilice* `document.saveAndCompact()` *o* `fl.saveDocumentAs()`*.*

Ejemplo

El ejemplo siguiente guarda el documento actual en su ubicación predeterminada.

```
fl.getDocumentDOM().save();
```

Véase también

`document.saveAndCompact()`, `fl.saveAll()`, `fl.saveDocument()`, `fl.saveDocumentAs()`

document.saveAndCompact()

Disponibilidad

Flash MX 2004

Uso

```
document.saveAndCompact([bOkToSaveAs])
```

Parámetros

`bOkToSaveAs` Un parámetro opcional que, si se define como `true` o se omite y no se ha guardado nunca el archivo, abre el cuadro de diálogo Guardar como. Si se define como `false` y el archivo no se ha guardado nunca, el archivo no se guardará. El valor predeterminado es `true`.

Valor devuelto

Un valor booleano: `true` si la operación de guardado y compactado se realiza correctamente; `false` en caso contrario.

Descripción

Método; guarda y compacta el archivo. Este método equivale a seleccionar Archivo > Guardar y compactar.

Nota: *si el archivo no se ha guardado nunca, este método devuelve* `true` *incluso si el usuario cancela el cuadro de diálogo Guardar como. Para determinar con precisión si se guardó el archivo, utilice* `fl.saveDocumentAs()`*.*

Ejemplo

El ejemplo siguiente guarda y compacta el documento actual:

```
fl.getDocumentDOM().saveAndCompact();
```

Véase también

`document.save()`, `fl.saveDocumentAs()`, `fl.saveDocument()`, `fl.saveAll()`

document.saveAVersion()

Disponibilidad
Flash CS3 Professional

Uso
`document.saveAVersion()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano de `true` si se guarda correctamente una versión del documento en el servidor Version Cue; `false` en caso contrario.

Descripción
Método; si el archivo se puede guardar en el servidor Version Cue, muestra un cuadro de diálogo para permitir que el usuario introduzca comentarios de la versión, guarda en el servidor una versión del documento especificado y registra cualquier error en el panel Salida.

Nota: *si Flash no puede guardar el archivo debido a que no se han almacenado en caché las credenciales del servidor en la sesión de aplicación actual, se muestra un error de fallo de autenticación en el panel Salida. Si se produce este error, el usuario debe utilizar el cuadro de diálogo Archivo > Abrir para abrir el espacio de trabajo de Version Cue con las credenciales correctas. A continuación, tendrán lugar las subsiguientes llamadas de la API JavaScript a este servidor.*

Ejemplo
`See document.canSaveAVersion().`

Véase también
`document.canSaveAVersion()`, `document.revertToLastVersion()`, `document.synchronizeWithHeadVersion()`

document.scaleSelection()

Disponibilidad
Flash MX 2004

Uso
`document.scaleSelection(xScale, yScale [, whichCorner])`

Parámetros
`xScale` Un valor de coma flotante que especifica la cantidad de *x* que se va a escalar.

`yScale` Un valor de coma flotante que especifica la cantidad de *y* que se va a escalar.

`whichCorner` Un valor de cadena que especifica el borde sobre el que se produce la transformación. Si se omite, el escalado se produce sobre el punto de transformación. Los valores aceptables son: `"bottom left"`, `"bottom right"`,

`"top right"`, `"top left"`, `"top center"`, `"right center"`, `"bottom center"` y `"left center"`. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; escala la selección en la cantidad especificada. Este método equivale a utilizar la herramienta Transformación libre para escalar el objeto.

Ejemplo
El ejemplo siguiente amplía la anchura de la selección actual hasta el doble de la anchura original y reduce la altura a la mitad:

```
fl.getDocumentDOM().scaleSelection(2.0, 0.5);
```

El ejemplo siguiente voltea la selección en vertical:

```
fl.getDocumentDOM().scaleSelection(1, -1);
```

El ejemplo siguiente voltea la selección en horizontal:

```
fl.getDocumentDOM().scaleSelection(-1, 1);
```

El ejemplo siguiente escala la selección en vertical a 1,9 desde la parte superior central:

```
fl.getDocumentDOM().scaleSelection(1, 1.90, 'top center');
```

document.screenOutline

Disponibilidad
Flash MX 2004

Uso
`document.screenOutline`

Descripción
Propiedad de sólo lectura; el objeto ScreenOutline actual para el documento. Antes de acceder al objeto por primera vez, asegúrese de utilizar `document.allowScreens()` para determinar si existe la propiedad.

Ejemplo
El ejemplo siguiente muestra el conjunto de valores en la propiedad `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);
```

Véase también
`document.allowScreens()`, Objeto ScreenOutline

document.selectAll()

Disponibilidad
Flash MX 2004

Uso
`document.selectAll()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; selecciona todos los elementos del escenario. Este método equivale a presionar Control+A (Windows) o Comando+A (Macintosh) o a seleccionar Edición > Seleccionar todo.

Ejemplo
El ejemplo siguiente selecciona todo lo que está visible actualmente para el usuario:

```
fl.getDocumentDOM().selectAll();
```

Véase también
`document.selection`, `document.selectNone()`

document.selection

Disponibilidad
Flash MX 2004

Uso
`document.selection`

Descripción
Propiedad; un conjunto de los objetos seleccionados en el documento. Si no hay nada seleccionado, devuelve un conjunto de longitud cero. Si no hay ningún documento abierto, devuelve `null`.

Para añadir objetos al conjunto, deberá seleccionarlos en primer lugar de una de estas dos formas:

- Seleccione manualmente los objetos en el escenario.
- Utilice uno de los métodos de selección, como `document.setSelectionRect()`, `document.setSelectionBounds()`, `document.mouseClick()`, `document.mouseDblClk()` o `document.selectAll()`.
- Seleccione manualmente uno o varios fotogramas.

- Utilice uno de los métodos del Objeto Timeline para seleccionar uno o varios fotogramas, por ejemplo `timeline.getSelectedFrames()`, `timeline.setSelectedFrames()` o `timeline.selectAllFrames()`.
- Especifique todos los elementos de un fotograma determinado (consulte Objeto Element). Consulte el primer ejemplo siguiente.
- Cree un conjunto de uno o varios elementos y asígnelo al conjunto `document.selection`. Consulte el tercer ejemplo siguiente.

Ejemplo

El ejemplo siguiente asigna todos los elementos del fotograma 11 a la selección actual (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().currentFrame = 10;
fl.getDocumentDOM().selection =
fl.getDocumentDOM().getTimeline().layers[0].frames[10].elements;
```

En el ejemplo siguiente se crea un rectángulo en la esquina superior izquierda del escenario y una cadena de texto bajo el rectángulo. A continuación, selecciona ambos objetos empleando `document.setSelectionRect()` y los añade a conjunto `document.selection` Por último, muestra el contenido de `document.selection` en el panel Salida.

```
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]);
}
```

A continuación, se ofrece un ejemplo avanzado. Muestra cómo se realiza un bucle a través del conjunto de capas y el conjunto de elementos para localizar instancias de un determinado símbolo y seleccionarlas. Puede ampliar este ejemplo para incluir bucles para varios fotogramas o escenas. Este ejemplo asigna todas las instancias del clip de película `myMovieClip` del primer fotograma a la selección actual:

```
// 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()

Disponibilidad

Flash MX 2004

Uso

`document.selectNone()`

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; desactiva la selección de los elementos seleccionados.

Ejemplo

El ejemplo siguiente desactiva la selección de los elementos que estén seleccionados:

```
fl.getDocumentDOM().selectNone();
```

Véase también

`document.selectAll()`, `document.selection`

document.setAlignToDocument()

Disponibilidad

Flash MX 2004

Uso

```
document.setAlignToDocument(bToStage)
```

Parámetros

`bToStage` Un valor booleano que, si se define como `true`,alinea los objetos con el escenario. Si se define como `false`, no los alinea.

Valor devuelto

Ninguno.

Descripción

Método; establece las preferencias de `document.align()`, `document.distribute()`, `document.match()` y `document.space()` para que actúen en el documento. Este método equivale a activar el botón En escenario en el panel Alinear.

Ejemplo

El ejemplo siguiente activa el botón En escenario del panel Alinear para alinear objetos con el escenario:

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

Véase también

`document.getAlignToDocument()`

document.setBlendMode()

Disponibilidad

Flash 8

Uso

```
document.setBlendMode(mode)
```

Parámetros

`mode` Una cadena que representa el modo de mezcla deseado para los objetos seleccionados. Los valores aceptables son `"normal"`, `"layer"`, `"multiply"`, `"screen"`, `"overlay"`, `"hardlight"`, `"lighten"`, `"darken"`, `"difference"`, `"add"`, `"subtract"`, `"invert"`, `"alpha"` y `"erase"`.

Valor devuelto
Ninguno.

Descripción
Método; establece el modo de mezcla para los objetos seleccionados.

Ejemplo
El ejemplo siguiente establece el modo de mezcla para el objeto seleccionado como `"add"`.

```
fl.getDocumentDOM().setBlendMode("add");
```

Véase también
`document.addFilter()`, `document.setFilterProperty()`, `symbolInstance.blendMode`

document.setCustomFill()

Disponibilidad
Flash MX 2004

Uso
```
document.setCustomFill(fill)
```

Parámetros
`fill` Un objeto Fill que especifica la configuración de relleno que se va a utilizar. Consulte Objeto Fill.

Valor devuelto
Ninguno.

Descripción
Método; establece la configuración de relleno para el panel Herramientas, el inspector de propiedades y cualquier forma seleccionada. Permite que un script establezca la configuración de relleno antes de dibujar el objeto, en lugar de dibujar el objeto, seleccionándolo y cambiando la configuración de relleno. También permite que un script cambie la configuración de relleno del panel Herramientas y del inspector de propiedades.

Ejemplo
El ejemplo siguiente cambia a blanco el color de la muestra de color de relleno en el panel Herramientas, el inspector de propiedades y las formas seleccionadas:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = '#FFFFFF';
fill.style = "solid";
fl.getDocumentDOM().setCustomFill(fill);
```

Véase también
`document.getCustomFill()`

document.setCustomStroke()

Disponibilidad
Flash MX 2004

Uso
`document.setCustomStroke(stroke)`

Parámetros
`stroke` Un Objeto Stroke.

Valor devuelto
Ninguno.

Descripción
Método; establece la configuración de trazo para el panel Herramientas, el inspector de propiedades y cualquier forma seleccionada. Permite que un script establezca la configuración de trazo antes de dibujar el objeto, en lugar de dibujar el objeto, seleccionándolo y cambiando la configuración de relleno. También permite que un script cambie la configuración de trazo del panel Herramientas y del inspector de propiedades.

Ejemplo
El ejemplo siguiente cambia la configuración de grosor del trazo en el panel Herramientas, el inspector de propiedades y las formas seleccionadas:

```
var stroke = fl.getDocumentDOM().getCustomStroke();
stroke.thickness += 2;
fl.getDocumentDOM().setCustomStroke(stroke);
```

Véase también
`document.getCustomStroke()`

document.setElementProperty()

Disponibilidad
Flash MX 2004

Uso
`document.setElementProperty(property, value)`

Parámetros
`property` Una cadena que especifica el nombre de la propiedad del objeto `Element` que se va a definir. Para ver una lista completa de propiedades y valores, consulte la tabla del resumen de propiedades del Objeto Element.

No es posible utilizar este método para establecer valores para propiedades de sólo lectura, como `element.elementType`, `element.top` o `element.left`.

`value` Un entero que especifica el valor que se va a establecer en la propiedad Element especificada.

Valor devuelto
Ninguno.

Descripción
Método; establece la propiedad `Element` especificada en el objeto u objetos seleccionados en el documento. Este método no tiene ningún efecto si no hay selección.

Ejemplo
El ejemplo siguiente establece la anchura de todos los objetos seleccionados en 100 y la altura en 50:

```
fl.getDocumentDOM().setElementProperty("width", 100);
fl.getDocumentDOM().setElementProperty("height", 50);
```

document.setElementTextAttr()

Disponibilidad
Flash MX 2004

Uso

```
document.setElementTextAttr(attrName, attrValue [, startIndex [, endIndex]])
```

Parámetros
`attrName` Una cadena que especifica el nombre de la propiedad del objeto `TextAttrs` que se va a modificar.

`attrValue` El valor con el que se define la propiedad `TextAttrs`. Para ver una lista de nombres de propiedades y valores esperados, consulte la tabla del resumen de propiedades del Objeto TextAttrs.

`startIndex` Un valor entero que especifica el índice del primer carácter que está afectado. Este parámetro es opcional.

`endIndex` Un valor entero que especifica el índice del último carácter que está afectado. Este parámetro es opcional.

Valor devuelto
Un valor booleano: `true` si ha cambiado al menos una propiedad de atributo de texto; `false` en caso contrario.

Descripción
Método; establece la propiedad `textAttrs` especificada de los elementos de texto seleccionados con el valor especificado. Para ver una lista de nombres de propiedades y valores admitidos, consulte la tabla del resumen de propiedades del Objeto TextAttrs. Si no se transfieren los parámetros opcionales, el método establece el estilo del rango de texto seleccionado actualmente o de todo el campo de texto si no hay texto seleccionado. Si sólo se transfiere *startIndex*, el método establece los atributos del carácter. Si se transfieren *startIndex* y *endIndex*, el método establece los atributos de los caracteres comenzando desde *startIndex* hasta *endIndex* (no incluido). Si se especifican estilos de párrafo, se verán afectados todos los que pertenezcan al rango.

Ejemplo
Los ejemplos siguientes establecen los atributos de texto `fillColor, italic y bold` para los elementos de texto seleccionados:

```
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()

Disponibilidad
Flash MX 2004

Uso
`document.setFillColor(color)`

Parámetros
`color` El color del relleno, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

  Si se define como `null`, no se establece ningún color de relleno, lo que equivale a definir la muestra de color de relleno en la interfaz de usuario como Sin relleno.

Valor devuelto
Ninguno.

Descripción
Método; cambia el color de relleno de la selección al especificarlo. Para obtener información sobre cómo cambiar el color de relleno en el panel Herramientas y el inspector de propiedades, consulte `document.setCustomFill()`.

Ejemplo
Las tres primeras sentencias del ejemplo siguiente establecen el color de relleno utilizando cada uno de los distintos formatos para especificar color. La cuarta sentencia establece el relleno como Sin relleno.

```
fl.getDocumentDOM().setFillColor("#cc00cc");
fl.getDocumentDOM().setFillColor(0xcc00cc);
fl.getDocumentDOM().setFillColor(120000);
fl.getDocumentDOM().setFillColor(null);
```

document.setFilterProperty()

Disponibilidad
Flash 8

Uso
`document.setFilterProperty(property, filterIndex, value)`

Parámetros
`property` Una cadena que especifica la propiedad que se va a establecer. Los valores aceptables son `"blurX"`, `"blurY"`, `"quality"`, `angle"`, `"distance"`, `"strength"`, `"knockout"`, `"inner"`, `"bevelType"`, `"color"`, `"shadowColor"` y `"highlightColor"`.

`filterIndex` Un entero que especifica el índice basado en cero del filtro en la lista Filtros.

value Un número o cadena que especifica el valor que se va a establecer para la propiedad de filtro especificada. Los valores válidos dependen de la propiedad y del filtro que se definen.

Valor devuelto
Ninguno.

Descripción
Método; establece una propiedad de filtro especificada para los objetos seleccionados actualmente (asumiendo que el objeto admite el filtro especificado).

Ejemplo
El ejemplo siguiente define la propiedad `quality` como 2 para el segundo filtro (valor de índice de 1) de la lista Filtros de los objetos seleccionados y, a continuación, define la propiedad `shadowColor` del primer filtro de la lista Filtros en el objeto u objetos seleccionados:

```
fl.getDocumentDOM().setFilterProperty("quality", 1, 2);
fl.getDocumentDOM().setFilterProperty("shadowColor", 0, "#FF00FF");
```

Véase también
`document.addFilter()`, `document.getFilters()`, `document.setBlendMode()`, `document.setFilters()`, Objeto Filter

document.setFilters()

Disponibilidad
Flash 8

Uso
`document.setFilters(filterArray)`

Parámetros
filterArray El conjunto de filtros especificado actualmente.

Valor devuelto
Ninguno.

Descripción
Método; aplica filtros a los objetos seleccionados. Utilice este método después de llamar `document.getFilters()` y realizar los cambios deseados en los filtros.

Ejemplo
El ejemplo siguiente obtiene los filtros del objeto seleccionado y define la propiedad `blurX` de todos los filtros de desenfoque como 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);
```

Véase también

`document.addFilter()`, `document.getFilters()`, `document.setFilterProperty()`, Objeto Filter

document.setInstanceAlpha()

Disponibilidad

Flash MX 2004

Uso

```
document.setInstanceAlpha(opacity)
```

Parámetros

`opacity` Un entero entre 0 (transparente) y 100 (completamente saturado) que ajusta la transparencia de la instancia.

Valor devuelto

Ninguno.

Descripción

Método; establece la opacidad de la instancia.

Ejemplo

El ejemplo siguiente establece la opacidad de la tinta con un valor de 50:

```
fl.getDocumentDOM().setInstanceAlpha(50);
```

document.setInstanceBrightness()

Disponibilidad

Flash MX 2004

Uso

```
document.setInstanceBrightness(brightness)
```

Parámetros

`brightness` Un entero que especifica el brillo como un valor entre -100 (negro) y 100 (blanco).

Valor devuelto

Ninguno.

Descripción
Método; establece el brillo de la instancia.

Ejemplo
El ejemplo siguiente establece el brillo de la instancia con un valor de 50:

```
fl.getDocumentDOM().setInstanceBrightness(50);
```

document.setInstanceTint()

Disponibilidad
Flash MX 2004

Uso
```
document.setInstanceTint( color, strength )
```

Parámetros
`color` El color de tinta, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

`strength` Un entero entre 0 y 100 que especifica la opacidad de la tinta.

Valor devuelto
Ninguno.

Descripción
Método; establece la tinta de la instancia.

Ejemplo
El ejemplo siguiente define la tinta para la instancia seleccionada como roja con un valor de opacidad de 50:

```
fl.getDocumentDOM().setInstanceTint(0xff0000, 50);
```

document.setMetadata()

Disponibilidad
Flash 8

Uso
```
document.setMetadata(strMetadata)
```

Parámetros

`strMetadata` Una cadena que contiene los metadatos XML que se van a asociar al documento. Para más información, consulte la siguiente descripción.

Valor devuelto

Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción

Método; establece los metadatos XML para el documento especificado, sobrescribiendo los metadatos existentes. El XML transferido como *strMetadata* se valida y se puede reescribir antes de almacenar. Si no se puede validar como código XML válido o infringe reglas específicas, los metadatos XML no se definen y se devuelve `false`. (Si se devuelve `false`, no hay forma de obtener información más detallada sobre el error.)

Nota: *aunque se devuelva* `true`*, es posible que el código XML definido no sea exactamente el mismo que la cadena que se transfirió. Para obtener el valor exacto con el que se definió XML, utilice* `document.getMetadata()`.

El formato de los metadatos es RDF compatible con la especificación XMP. Para más información sobre RDF y XMP, consulte las siguientes fuentes:

- RDF Primer en www.w3.org/TR/rdf-primer/
- La especificación RDF en www.w3.org/TR/1999/REC-rdf-syntax-19990222/
- La página de inicio de XMP en www.adobe.com/products/xmp/

Ejemplo

Los ejemplos siguientes muestran distintas formas válidas de representar los mismos datos. En todos estos casos salvo en el segundo, si los datos se enviaran a `Document.setMetadata()`, no se reescribirían (aparte de eliminar los saltos de línea).

En el primer ejemplo, los metadatos están en etiquetas, con distintos esquemas colocados en etiquetas `rdf:Description` independientes:

```
<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>
```

En el segundo ejemplo, los metadatos están en etiquetas, pero con distintos esquemas en una etiqueta `rdf:Description`. Este ejemplo también incluye comentarios, que `Document.setMetadata()` ignorará y descartará:

```
<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>
```

En el tercer ejemplo, los metadatos están en atributos y los distintos esquemas en una sola etiqueta `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>
```

Véase también

`document.getMetadata()`

document.setMobileSettings()

Disponibilidad

Flash CS3 Professional

Uso

`document.setMobileSettings(xmlString)`

Parámetros

`xmlString` Una cadena que describe la configuración XML en un archivo FLA móvil.

Valor devuelto

Un valor de `true` si se estableció correctamente la configuración; `false` en caso contrario.

Descripción

Método; establece el valor de una cadena de configuración XML en un archivo FLA móvil. (La mayoría de los archivos FLA tienen una cadena XML que describe la configuración en el documento.)

Ejemplo

El ejemplo siguiente establece la cadena de configuración XML de un archivo FLA. Tenga en cuenta que el ejemplo a continuación representa una sola línea de código.

```
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>");
```

Véase también

`document.getMobileSettings()`

document.setOvalObjectProperty()

Disponibilidad

Flash CS3 Professional

Uso

```
document.setOvalObjectProperty(propertyName, value)
```

Parámetros

`propertyName` Una cadena que especifica la propiedad que se va a establecer. Para ver una lista de los valores aceptables, consulte la tabla del resumen de propiedades del Objeto Oval.

`value` El valor que se va a asignar a la propiedad. Los valores aceptables varían según la propiedad especificada en *propertyName*.

Valor devuelto

Ninguno.

Descripción

Método; especifica un valor para la propiedad indicada de objetos Oval simples.

Ejemplo

Consulte propiedades individuales en Objeto Oval para ver ejemplos.

Véase también

Objeto Oval, `shape.isOvalObject`

document.setPlayerVersion()

Disponibilidad

Flash CS3 Professional

Uso

```
document.setPlayerVersion(version)
```

Parámetros

`version` Una cadena que representa la versión de Flash Player del documento especificado. Los valores admitidos son `"FlashLite"`, `"FlashLite11"`, `"FlashLite20"`, `"FlashLite30"`, `"1"`, `"2"`, `"3"`, `"4"`, `"5"`, `"6"`, `"7"`, `"8"`, `"9"`, `"FlashPlayer10"` y `"AdobeAIR1_1"`.

Valor devuelto

Un valor de `true` si se estableció correctamente la versión del reproductor; `false` en caso contrario.

Descripción

Método; establece la versión de Flash Player del documento especificado. Se trata del mismo valor que el definido en el cuadro de diálogo Configuración de publicación.

Ejemplo

El ejemplo siguiente tiene Flash Player 6 como versión del reproductor para el documento actual:

```
fl.getDocumentDOM().setPlayerVersion("6");
```

Véase también

`document.getPlayerVersion()`

document.setRectangleObjectProperty()

Disponibilidad

Flash CS3 Professional

Uso

```
document.setRectangleObjectProperty(propertyName, value)
```

Parámetros

`propertyName` Una cadena que especifica la propiedad que se va a establecer. Para ver una lista de los valores aceptables, consulte la tabla del resumen de propiedades del Objeto Rectangle.

`value` El valor que se va a asignar a la propiedad. Los valores aceptables varían según la propiedad especificada en *propertyName*.

Valor devuelto

Ninguno.

Descripción

Método; especifica un valor para la propiedad indicada de objetos Rectangle simples.

Ejemplo

Consulte propiedades individuales en Objeto Rectangle para ver ejemplos.

Véase también

Objeto Rectangle, shape.isRectangleObject

document.setSelectionBounds()

Disponibilidad

Flash MX 2004; parámetro *bContactSensitiveSelection* añadido en Flash 8.

Uso

```
document.setSelectionBounds(boundingRectangle [, bContactSensitiveSelection])
```

Parámetros

`boundingRectangle` Un rectángulo que especifica la nueva ubicación y el tamaño de la selección. Para obtener más información sobre el formato de *boundingRectangle*, consulte `document.addNewRectangle()`.

`bContactSensitiveSelection` Un valor booleano que especifica si el modo de selección Por contacto está activado (`true`) o desactivado (`false`) durante la selección de objetos. El valor predeterminado es `false`.

Valor devuelto

Ninguno.

Descripción

Método; mueve y cambia el tamaño de la selección en una única operación.

Si transmite un valor para *bContactSensitiveSelection*, sólo es válido para este método y no afecta al modo de selección Por contacto del documento (consulte `fl.contactSensitiveSelection`).

Ejemplo

El ejemplo siguiente mueve la selección actual a 10, 20 y cambia el tamaño a 100, 200:

```
var l = 10;
var t = 20;
fl.getDocumentDOM().setSelectionBounds({left:l, top:t, right:(100+l), bottom:(200+t)});
```

Véase también

`document.selection`, `document.setSelectionRect()`

document.setSelectionRect()

Disponibilidad

Flash MX 2004; parámetro *bContactSensitiveSelection* añadido en Flash 8.

Uso

```
document.setSelectionRect(rect [, bReplaceCurrentSelection [, bContactSensitiveSelection]])
```

Parámetros

`rect` Un objeto rectangle para definir como seleccionado. Para más información sobre el formato de *rect*, consulte document.addNewRectangle().

`bReplaceCurrentSelection` Un valor booleano que especifica si el método reemplaza la selección actual (`true`) o se añade a la selección actual (`false`). El valor predeterminado es `true`.

bContactSensitiveSelection Un valor booleano que especifica si el modo de selección Por contacto está activado (`true`) o desactivado (`false`) durante la selección de objetos. El valor predeterminado es `false`.

Valor devuelto

Ninguno.

Descripción

Método; dibuja un recuadro de delimitación rectangular en relación con el escenario, empleando las coordenadas especificadas. Es distinto de `document.getSelectionRect()`, en el que el rectángulo es relativo al objeto que se está editando.

Este método equivale a arrastrar un rectángulo con la herramienta Selección. Para que se seleccione, una instancia debe estar completamente encerrada por el rectángulo.

Si pasa un valor para *bContactSensitiveSelection*, sólo es válido para este método y no afecta al modo de selección Por contacto del documento (consulte `fl.contactSensitiveSelection`

Nota: *si repite `setSelectionRect()` empleando el panel Historial o el elemento de menú, se repetirá el paso anterior a la operación `setSelectionRect()` .*

Ejemplo

En el ejemplo siguiente, la segunda selección reemplaza a la primera:

```
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);
```

En el ejemplo siguiente, la segunda selección se añade a la primera. Tiene el mismo efecto que la operación manual de mantener presionada la tecla Mayús y seleccionar un segundo objeto.

```
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);
```

Véase también

`document.getSelectionRect()`, `document.selection`, `document.setSelectionBounds()`

document.setStageVanishingPoint()

Disponibilidad

Flash CS4 Professional

Uso

```
document.setStageVanishingPoint(point)
```

Parámetros

point Un punto que especifica las coordenadas *x* e *y* de la ubicación en la que se establece el punto de desvanecimiento para visualizar objetos 3D.

Valor devuelto
Ninguno.

Descripción
Especifica el punto de desvanecimiento para visualizar objetos 3D.

Ejemplo
El ejemplo siguiente define el punto de desvanecimiento del escenario:

```
fl.getDocumentDOM().setStageVanishingPoint({x:45, y:45});
```

document.setStageViewAngle()

Disponibilidad
Flash CS4 Professional

Uso
```
document.setStageViewAngle(angle)
```

Parámetros
`angle` Valor de coma flotante entre 0,0 y 179,0.

Valor devuelto
Ninguno.

Descripción
Especifica el ángulo de perspectiva para visualizar objetos 3D.

Ejemplo
El ejemplo siguiente define el ángulo de perspectiva del escenario en 70 grados:

```
fl.getDocumentDOM().setStageViewAngle(70);
```

document.setStroke()

Disponibilidad
Flash MX 2004

Uso
```
document.setStroke(color, size, strokeType)
```

Parámetros
`color` El color del trazo, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`

- Un entero que representa el equivalente decimal del número hexadecimal

size Un valor de coma flotante que especifica el nuevo tamaño de trazo para la selección.

strokeType Una cadena que especifica el nuevo tipo de trazo para la selección. Los valores aceptables son `"hairline"`, `"solid"`, `"dashed"`, `"dotted"`, `"ragged"`, `"stipple"` y `"hatched"`.

Valor devuelto
Ninguno.

Descripción
Método; establece el color, el ancho y el estilo del trazo seleccionado. Para obtener información sobre cómo cambiar el trazo en el panel Herramientas y el inspector de propiedades, consulte `document.setCustomStroke()`.

Ejemplo
El ejemplo siguiente define el color del trazo como rojo, el tamaño como 3,25 y el tipo como líneas discontinuas:

```
fl.getDocumentDOM().setStroke("#ff0000", 3.25, "dashed");
```

document.setStrokeColor()

Disponibilidad
Flash MX 2004

Uso
```
document.setStrokeColor(color)
```

Parámetros
color El color del trazo, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

Valor devuelto
Ninguno.

Descripción
Método; cambia el color de trazo de la selección al especificado. Para obtener información sobre cómo cambiar el trazo en el panel Herramientas y el inspector de propiedades, consulte `document.setCustomStroke()`.

Ejemplo
Las tres sentencias del ejemplo siguiente establecen el color de trazo utilizando cada uno de los distintos formatos para especificar color:

```
fl.getDocumentDOM().setStrokeColor("#cc00cc");
fl.getDocumentDOM().setStrokeColor(0xcc00cc);
fl.getDocumentDOM().setStrokeColor(120000);
```

document.setStrokeSize()

Disponibilidad
Flash MX 2004

Uso
`document.setStrokeSize(size)`

Parámetros
`size` Un valor de coma flotante de 0,25 a 10 que especifica el tamaño de trazo. El método ignora las precisiones mayores que dos posiciones decimales.

Valor devuelto
Ninguno.

Descripción
Método; cambia el tamaño de trazo de la selección al especificado. Para obtener información sobre cómo cambiar el trazo en el panel Herramientas y el inspector de propiedades, consulte `document.setCustomStroke()`.

Ejemplo
El ejemplo siguiente cambia el tamaño de trazo de la selección a 5:

```
fl.getDocumentDOM().setStrokeSize(5);
```

document.setStrokeStyle()

Disponibilidad
Flash MX 2004

Uso
`document.setStrokeStyle(strokeType)`

Parámetros
`strokeType` Una cadena que especifica el estilo de trazo para la selección actual. Los valores aceptables son `"hairline"`, `"solid"`, `"dashed"`, `"dotted"`, `"ragged"`, `"stipple"` y `"hatched"`.

Valor devuelto
Ninguno.

Descripción
Método; cambia el estilo de trazo de la selección al especificado. Para obtener información sobre cómo cambiar el trazo en el panel Herramientas y el inspector de propiedades, consulte `document.setCustomStroke()`.

Ejemplo
El ejemplo siguiente cambia el estilo de trazo de la selección a `"dashed"`:

```
fl.getDocumentDOM().setStrokeStyle("dashed");
```

document.setTextRectangle()

Disponibilidad
Flash MX 2004

Uso
```
document.setTextRectangle(boundingRectangle)
```

Parámetros
`boundingRectangle` Un rectángulo que especifica el nuevo tamaño dentro del cual el elemento de texto debe fluir. Para más información sobre el formato de *boundingRectangle*, consulte document.addNewRectangle().

Valor devuelto
Un valor booleano: `true` si ha cambiado el tamaño de al menos un campo de texto; `false` en caso contrario.

Descripción
Método; cambia el rectángulo de delimitación para el elemento de texto seleccionado al tamaño especificado. Este método hace que el texto vuelva a fluir dentro del nuevo rectángulo. El elemento de texto no se escala ni transforma. Los valores transferidos en *boundingRectangle* se emplean de la forma siguiente:

- Si el texto es horizontal y estático, el método sólo tiene en cuenta el valor de anchura transferido en *boundingRectangle*; la altura se calcula automáticamente para que quepa todo el texto.
- Si el texto es vertical (y, por tanto, estático), el método sólo tiene en cuenta el valor de altura transferido en *boundingRectangle*; la anchura se calcula automáticamente para que quepa todo el texto.
- Si el texto es dinámico o de entrada, el método tiene en cuenta los valores de anchura y altura transferidos en *boundingRectangle* y el rectángulo resultante podría ser mayor de lo necesario para que encaje todo el texto. Sin embargo, si los parámetros especifican un tamaño de rectángulo que es demasiado pequeño para que encaje todo el texto, el método sólo tiene en cuenta el valor de anchura transferido en *boundingRectangle* (la altura se calcula automáticamente para que quepa todo el texto).

Ejemplo
El ejemplo siguiente cambia el tamaño del rectángulo de texto de delimitación a las dimensiones especificadas:

```
fl.getDocumentDOM().setTextRectangle({left:0, top:0, right:50, bottom:200})
```

document.setTextSelection()

Disponibilidad
Flash MX 2004

Uso
```
document.setTextSelection(startIndex, endIndex)
```

Parámetros
`startIndex` Un entero que especifica la posición del primer carácter que se va a seleccionar. La posición del primer carácter es 0 (cero).

endIndex Un entero que especifica la posición final de la selección hasta *endIndex* (no incluido). La posición del primer carácter es 0 (cero).

Valor devuelto
Un valor booleano: `true` si el método puede establecer correctamente la selección de texto; `false` en caso contrario.

Descripción
Método; establece la selección de texto del campo de texto seleccionado actualmente con los valores especificados por los valores *startIndex* y *endIndex*. Se activará la edición de texto, si aún no lo está.

Ejemplo
El ejemplo siguiente selecciona el texto desde el carácter 6º hasta el carácter 25º.

```
fl.document.setTextSelection(5, 25);
```

document.setTextString()

Disponibilidad
Flash MX 2004

Uso

```
document.setTextString(text [, startIndex [, endIndex]])
```

Parámetros
text Una cadena de caracteres que se va a insertar en el campo de texto.

startIndex Un entero que especifica el primer carácter que se va a reemplazar. La posición del primer carácter es 0 (cero). Este parámetro es opcional.

endIndex Un entero que especifica el último carácter que se va a reemplazar. Este parámetro es opcional.

Valor devuelto
Un valor booleano: `true` si está establecido el texto de al menos una cadena de texto; `false` en caso contrario.

Descripción
Método; inserta una cadena de texto. Si no se transfieren los parámetros opcionales, se reemplazará la selección de texto existente; si el objeto de texto no se está editando actualmente, se reemplazará la cadena de texto completa. Si sólo se transfiere *startIndex*, la cadena transferida se insertará en esta posición. Si se transfieren *startIndex* y *endIndex*, la cadena transferida reemplazará el segmento de texto que comienza desde *startIndex* hasta *endIndex* (no incluido).

Ejemplo
El ejemplo siguiente reemplaza la selección de texto actual con “Hello World”:

```
var success = fl.getDocumentDOM().setTextString("Hello World!");
```

El ejemplo siguiente inserta “hello” en la posición 6 de la selección de texto actual:

```
var pass = fl.getDocumentDOM().setTextString("hello", 6);
```

El ejemplo siguiente inserta “Howdy” comenzando en la posición 2 hasta la posición 7 (no incluida) de la selección de texto actual:

```
var ok = fl.getDocumentDOM().setTextString("Howdy", 2, 7);
```

Véase también

`document.getTextString()`

document.setTransformationPoint()

Disponibilidad

Flash MX 2004

Uso

```
document.setTransformationPoint( transformationPoint )
```

Parámetros

`transformationPoint` Un punto (por ejemplo, `{x:10, y:20}`, donde `x` e `y` son números de coma flotante) que especifica los valores para el punto de transformación de los siguientes elementos:

- Formas: *transformationPoint* se establece en relación al documento (0,0 es la esquina superior izquierda del escenario).
- Símbolos: *transformationPoint* se establece en relación al punto de registro del símbolo (0,0 se ubica en el punto de registro).
- Texto: *transformationPoint* se establece en relación al campo de texto (0,0 es la esquina superior izquierda del campo de texto).
- Mapas de bits/vídeos: *transformationPoint* se establece en relación al mapa de bits o vídeo (0,0 es la esquina superior izquierda del mapa de bits o vídeo).
- Objetos de dibujo, óvalos y rectángulos simples, y grupos: *transformationPoint* se establece en relación al documento (0,0 es la esquina superior izquierda del escenario). Para establecer *transformationPoint* en relación al punto central del objeto, óvalo o rectángulo simple, o grupo, utilice `element.setTransformationPoint()`.

Valor devuelto

Ninguno.

Descripción

Método; establece la posición del punto de transformación de la selección actual.

Ejemplo

El ejemplo siguiente establece el punto de transformación de la selección actual en 100, 200:

```
fl.getDocumentDOM().setTransformationPoint({x:100, y:200});
```

Véase también

`document.getTransformationPoint()`, `element.setTransformationPoint()`

document.silent

Disponibilidad
Flash MX 2004

Uso
`document.silent`

Descripción
Propiedad; un valor booleano que especifica si el objeto es accesible. Equivale a la lógica inversa de la opción Permitir acceso a la película del panel Accesibilidad. Es decir, si `document.silent` es `true`, equivale a desactivar la opción Permitir acceso a la película. Si es `false`, equivale a la opción activada Permitir acceso a la película.

Ejemplo
El ejemplo siguiente define la variable `isSilent` con el valor de la propiedad `silent`:

```
var isSilent = fl.getDocumentDOM().silent;
```

El ejemplo siguiente establece la propiedad `silent` como `false`, lo que indica que el documento es accesible:

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

document.skewSelection()

Disponibilidad
Flash MX 2004

Uso
`document.skewSelection(xSkew, ySkew [, whichEdge])`

Parámetros
`xSkew` Un número de coma flotante que especifica la cantidad que se va a sesgar respecto del eje *x*, medida en grados.

`ySkew` Un número de coma flotante que especifica la cantidad que se va a sesgar respecto del eje *y*, medida en grados.

`whichEdge` Una cadena que especifica el borde donde se produce la transformación. Si se omite, el sesgo se produce en el punto de transformación. Los valores aceptables son `"top center"`, `"right center"`, `"bottom center"` y `"left center"`. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; sesga la selección en la cantidad especificada. Se obtiene el mismo efecto que con la herramienta Transformación libre para sesgar el objeto.

Ejemplo
Los ejemplos siguientes sesgan el objeto seleccionado 2,0 en vertical y 1,5 en horizontal. El segundo ejemplo transforma el objeto en el borde superior central:

```
fl.getDocumentDOM().skewSelection(2.0, 1.5);
fl.getDocumentDOM().skewSelection(2.0, 1.5, "top center");
```

document.smoothSelection()

Disponibilidad
Flash MX 2004

Uso
```
document.smoothSelection()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; suaviza la curva de cada línea curva o contorno de relleno seleccionado. Este método realiza la misma acción que el botón Suavizar del panel Herramientas.

Ejemplo
El ejemplo siguiente suaviza la curva de la selección actual:

```
fl.getDocumentDOM().smoothSelection();
```

document.sourcePath

Disponibilidad
Flash CS4 Professional

Uso
```
document.sourcePath
```

Descripción
Propiedad; una cadena que contiene una lista de elementos de la ruta de origen de ActionScript 3.0 del documento, donde se especifica la ubicación de los archivos de clases de ActionScript. Los elementos de la cadena se delimitan con punto y coma. En la herramienta de edición, puede especificar elementos desde Archivo > Configuración de publicación y seleccionar Configuración de ActionScript 3.0 en la ficha Flash.

Ejemplo
El ejemplo siguiente añade los archivos de la carpeta ./Class a la ruta de origen del documento:

```
var myDoc = fl.getDocumentDOM();
fl.trace(myDoc.sourcePath);
myDoc.sourcePath = "./Class files;" + myDoc.sourcePath;
fl.trace(myDoc.sourcePath);
```

Véase también

`document.externalLibraryPath`,`document.libraryPath`, `fl.sourcePath`

document.space()

Disponibilidad

Flash MX 2004

Uso

```
document.space(direction [, bUseDocumentBounds])
```

Parámetros

`direction` Una cadena que especifica la dirección en la que se distribuyen los objetos en la selección. Los valores aceptables son `"horizontal"` o `"vertical"`.

`bUseDocumentBounds` Un valor booleano que, si se define como `true`, distribuye los objetos en los límites del documento. En caso contrario, el método utiliza los límites de los objetos seleccionados. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; distribuye los objetos de la selección de manera uniforme.

Ejemplo

El ejemplo siguiente distribuye los objetos horizontalmente en relación con el escenario:

```
fl.getDocumentDOM().space("horizontal",true);
```

El ejemplo siguiente distribuye los objetos horizontalmente en relación mutua:

```
fl.getDocumentDOM().space("horizontal");
```

El ejemplo siguiente distribuye los objetos horizontalmente en relación mutua, con *bUseDcoumentBounds* definido expresamente como `false`:

```
fl.getDocumentDOM().space("horizontal",false);
```

Véase también

`document.getAlignToDocument()`, `document.setAlignToDocument()`

document.straightenSelection()

Disponibilidad
Flash MX 2004

Uso
```
document.straightenSelection()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; endereza los trazos seleccionados actualmente. Este método equivale a utilizar el botón Enderezar del panel Herramientas.

Ejemplo
El ejemplo siguiente endereza la curva de la selección actual:

```
fl.getDocumentDOM().straightenSelection();
```

document.swapElement()

Disponibilidad
Flash MX 2004

Uso
```
document.swapElement(name)
```

Parámetros
`name` Una cadena que especifica el nombre del elemento de biblioteca que se va a utilizar.

Valor devuelto
Ninguno.

Descripción
Método; cambia la selección actual por la especificada. La selección debe contener un gráfico, botón, clip de película, vídeo o mapa de bits. Este método muestra un mensaje de error si no se selecciona ningún objeto o no se encuentra el objeto dado.

Ejemplo
El ejemplo siguiente cambia la selección actual por `Symbol 1` de la biblioteca:

```
fl.getDocumentDOM().swapElement('Symbol 1');
```

document.swapStrokeAndFill()

Disponibilidad
Flash 8

Uso
```
document.swapStrokeAndFill()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; intercambia los colores de Trazo y Relleno.

Ejemplo
El siguiente ejemplo intercambia los colores de Trazo y Relleno en el documento actual:

```
fl.getDocumentDOM().swapStrokeAndFill();
```

document.synchronizeWithHeadVersion()

Disponibilidad
Flash CS3 Professional

Uso
```
fldocument.getDocumentDOMsynchronizeWithHeadVersion().swapStrokeAndFill();
```

Parámetros
Ninguno.

Valor devuelto
Un valor booleano de `true` si el archivo especificado se sincronizó correctamente con el servidor Version Cue, `false` en caso contrario.

Descripción
Método; sincroniza el documento especificado con la versión más actual en el servidor Version Cue y registra cualquier error en el panel Salida.

Este método sólo funciona con documentos que están actualmente abiertos. Para recuperar la última versión de un archivo que no está abierto, utilice `fl.downloadLatestVersion()`.

Ejemplo
En el ejemplo siguiente se sincroniza el documento actual con la versión del servidor Version Cue:

```
fl.getDocumentDOM().synchronizeWithHeadVersion();
```

Véase también

`document.canSaveAVersion()`, `fl.downloadLatestVersion()`, `document.revertToLastVersion()`, `document.saveAVersion()`, `fl.synchronizeDocumentWithHeadVersion()`

document.testMovie()

Disponibilidad

Flash MX 2004

Uso

```
document.testMovie()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; ejecuta una operación Probar película en el documento.

Ejemplo

El ejemplo siguiente prueba la película para el documento actual:

```
fl.getDocumentDOM().testMovie();
```

Véase también

`document.canTestMovie()`, `document.testScene()`

document.testScene()

Disponibilidad

Flash MX 2004

Uso

```
document.testScene()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; ejecuta una operación Probar escena en la escena actual del documento.

Ejemplo

El ejemplo siguiente prueba la escena actual en el documento:

```
fl.getDocumentDOM().testScene();
```

Véase también

`document.canTestScene()`, `document.testMovie()`

document.timelines

Disponibilidad

Flash MX 2004

Uso

```
document.timelines
```

Descripción

Propiedad de sólo lectura; un conjunto de objetos Timeline (consulte Objeto Timeline).

Ejemplo

El ejemplo siguiente obtiene el conjunto de las líneas de tiempo actuales en el documento activo y muestra sus nombres en el panel Salida:

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

Véase también

`document.currentTimeline`, `document.getTimeline()`

document.traceBitmap()

Disponibilidad

Flash MX 2004

Uso

```
document.traceBitmap(threshold, minimumArea, curveFit, cornerThreshold)
```

Parámetros

`threshold` Un entero que controla el número de colores del mapa de bits trazado. Los valores aceptables son enteros comprendidos entre 0 y 500.

`minimumArea` Un entero que especifica el radio medido en píxeles. Los valores aceptables son enteros comprendidos entre 1 y 1000.

`curveFit` Una cadena que especifica con qué suavidad se dibujan los contornos. Los valores aceptables son `"pixels"`, `"very tight"`, `"tight"`, `"normal"`, `"smooth"` y `"very smooth"`.

`cornerThreshold` Una cadena que es similar a *curveFit*, aunque corresponde a las esquinas de la imagen de mapa de bits. Los valores aceptables son `"many corners"`, `"normal"` y `"few corners"`.

Valor devuelto

Ninguno.

Descripción

Método; realiza un mapa de bits de traza en la selección actual. Este método equivale a seleccionar Modificar > Mapa de bits > Trazar mapa de bits.

Ejemplo

El ejemplo siguiente traza el mapa de bits seleccionado empleando los parámetros especificados:

```
fl.getDocumentDOM().traceBitmap(0, 500, 'normal', 'normal');
```

document.translate3DCenter()

Disponibilidad

Flash CS4 Professional

Uso

```
document.translate3DCenter(xyzCoordinate)
```

Parámetros

`xyzCoordinate` Una coordenada XYZ que especifica el centro para la rotación y la transformación 3D.

Valor devuelto

Ninguno.

Descripción

Método; define la posición XYZ alrededor de la cual se transforma y se gira la selección. Este método sólo está disponible para clips de película.

Ejemplo

El siguiente ejemplo especifica los ejes XYZ para la transformación 3D:

```
fl.getDocumentDOM().translate3DCenter({x:180, y:18,z:-30});
```

document.translate3DSelection()

Disponibilidad
Flash CS4 Professional

Uso
```
document.translate3DSelection(xyzCoordinate, bGlobalTransform)
```

Parámetros
`xyzCoordinate` Una coordenada XYZ que especifica los ejes para la transformación 3D.

`bGlobalTransform` Un valor booleano que especifica si el modo de transformación debe ser global (`true`) o local (`false`).

Valor devuelto
Ninguno.

Descripción
Método; aplica una transformación 3D a la selección. Este método sólo está disponible para clips de película.

Ejemplo
En el siguiente ejemplo, la selección se transforma primero en relación al escenario (globalmente) y, después, con respecto a sí misma (localmente).

```
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);
```

Véase también
document.translate3DCenter()

document.transformSelection()

Disponibilidad
Flash MX 2004

Uso
```
document.transformSelection(a, b, c, d)
```

Parámetros
`a` Un número de coma flotante que especifica el elemento (0,0) de la matriz de transformación.

`b` Un número de coma flotante que especifica el elemento (0,1) de la matriz de transformación.

`c` Un número de coma flotante que especifica el elemento (1,0) de la matriz de transformación.

`d` Un número de coma flotante que especifica el elemento (1,1) de la matriz de transformación.

Valor devuelto
Ninguno.

Descripción
Método; realiza una transformación general en la selección actual aplicando la matriz especificada en los argumentos. Para más información, consulte la propiedad `element.matrix`.

Ejemplo
El ejemplo siguiente amplía la selección en un factor de 2 en la dirección x:

```
fl.getDocumentDOM().transformSelection(2.0, 0.0, 0.0, 1.0);
```

document.unGroup()

Disponibilidad
Flash MX 2004

Uso
`document.unGroup()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; desagrupa la selección actual.

Ejemplo
El ejemplo siguiente desagrupa los elementos de la selección actual:

```
fl.getDocumentDOM().unGroup();
```

Véase también
`document.group()`

document.union()

Disponibilidad
Flash 8

Uso
`document.union()`

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si es correcto y `false` en caso contrario.

Descripción
Método; combina todas las formas seleccionadas en un objeto de dibujo.

Ejemplo
El ejemplo siguiente combina todas las formas seleccionadas en un objeto de dibujo:

```
fl.getDocumentDOM().union();
```

Véase también
`document.crop()`, `document.deleteEnvelope()`, `document.intersect()`, `document.punch()`, `shape.isDrawingObject`

document.unlockAllElements()

Disponibilidad
Flash MX 2004

Uso
```
document.unlockAllElements()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; desbloquea todos los elementos bloqueados en el fotograma seleccionado actualmente.

Ejemplo
En el siguiente ejemplo se desbloquean todos los objetos bloqueados del fotograma actual:

```
fl.getDocumentDOM().unlockAllElements();
```

Véase también
`element.locked`

document.viewMatrix

Disponibilidad

Flash MX 2004

Uso

`document.viewMatrix`

Descripción

Propiedad de sólo lectura; un objeto Matrix. `viewMatrix` se emplea para cambiar del espacio de objeto al espacio de documento cuando el documento se encuentra en modo de edición. La ubicación del ratón, como la recibe una herramienta, es relativa al objeto que se está editando. Consulte Objeto Matrix.

Por ejemplo, si crea un símbolo, hace doble clic en él para editarlo y dibuja con la herramienta PolyStar, el punto (0,0) estará en el punto de registro del símbolo. Sin embargo, el objeto drawingLayer espera valores en el espacio de documento, por lo que si dibuja una línea desde (0,0) empleando drawingLayer, comenzará en la esquina superior izquierda del escenario. La propiedad `viewMatrix` permite cambiar del espacio del objeto que se está editando al espacio de documento.

Ejemplo

El ejemplo siguiente obtiene el valor de la propiedad `viewMatrix`:

```
var mat = fl.getDocumentDOM().viewMatrix;
```

document.width

Disponibilidad

Flash MX 2004

Uso

`document.width`

Descripción

Propiedad; un entero que especifica la anchura del documento (escenario) en píxeles.

Ejemplo

El ejemplo siguiente establece la anchura del escenario en 400 píxeles:

```
fl.getDocumentDOM().width= 400;
```

Véase también

`document.height`

document.xmlPanel()

Disponibilidad
Flash MX 2004

Uso
`document.xmlPanel(fileURI)`

Parámetros
`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta del archivo XML que define los controles del panel. Se necesita la ruta completa.

Valor devuelto
Un objeto que tiene propiedades definidas para todos los controles definidos en el archivo XML. Todas las propiedades se devuelven como cadenas. El objeto devuelto tendrá una propiedad predefinida llamada `"dismiss"` que tendrá el valor de cadena `"accept"` o `"cancel"`.

Descripción
Método; envía un cuadro de diálogo XMLUI. Consulte `fl.xmlui`.

Ejemplo
El ejemplo siguiente carga el archivo Test.xml y muestra cada una de las propiedades que contiene:

```
var obj = fl.getDocumentDOM().xmlPanel(fl.configURI + "Commands/Test.xml");
for (var prop in obj) {
    fl.trace("property " + prop + " = " + obj[prop]);
}
```

document.zoomFactor

Disponibilidad
Flash 8

Uso
`document.zoomFactor`

Descripción
Propiedad; especifica el porcentaje de zoom del escenario en tiempo de edición. Un valor de 1 es igual a un zoom del 100%, 8 es igual a 800%, 0,5 es igual a 50%, y así sucesivamente.

Ejemplo
El ejemplo siguiente establece el factor de zoom del escenario en 200%.

```
fl.getDocumentDOM().zoomFactor = 2;
```

Capítulo 12: Objeto drawingLayer

Disponibilidad
Flash MX 2004

Descripción
Al objeto drawingLayer se puede acceder desde JavaScript como elemento secundario del objeto flash. El objeto drawingLayer se utiliza para herramientas ampliables cuando el usuario desea dibujar temporalmente mientras arrastra el ratón; por ejemplo, al crear un recuadro de delimitación. Debe llamar a `drawingLayer.beginFrame()` antes de que pueda llamar a cualquier otro método drawingLayer.

Resumen de métodos
Los siguientes métodos están disponibles para el objeto drawingLayer:

Método	Descripción
`drawingLayer.beginDraw()`	Sitúa a Flash en modo de dibujo.
`drawingLayer.beginFrame()`	Borra lo que se había dibujado anteriormente utilizando drawingLayer y prepara para más comandos de dibujo.
`drawingLayer.cubicCurveTo()`	Dibuja una curva cúbica desde la ubicación actual de la pluma empleando los parámetros como coordenadas del segmento cúbico.
`drawingLayer.curveTo()`	Dibuja un segmento de curva cuadrática comenzando en la posición de dibujo actual y terminando en un punto especificado.
`drawingLayer.drawPath()`	Dibuja el trazado especificado.
`drawingLayer.endDraw()`	Sale del modo de dibujo.
`drawingLayer.endFrame()`	Señala el final de un grupo de comandos de dibujo.
`drawingLayer.lineTo()`	Dibuja una línea desde la posición de dibujo actual hasta el punto (*x*,*y*).
`drawingLayer.moveTo()`	Establece la posición de dibujo actual.
`drawingLayer.newPath()`	Devuelve un nuevo Objeto Path.
`drawingLayer.setColor()`	Establece el color de los datos dibujados a continuación.
`drawingLayer.setFill()`	Este método no está disponible.
`drawingLayer.setStroke()`	Este método no está disponible.

drawingLayer.beginDraw()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.beginDraw([persistentDraw])
```

Parámetros

`persistentDraw` Un valor booleano (opcional). Si se define como `true`, indica que el dibujo del último fotograma permanece en el escenario hasta que se realice una nueva llamada `beginDraw()` o `beginFrame()`. (En este contexto, *frame* hace referencia a dónde comienza y termina el dibujo, no a fotogramas de la línea de tiempo.) Por ejemplo, cuando los usuarios dibujan un rectángulo, pueden obtener una vista previa del contorno de la forma mientras arrastran el ratón. Si desea que la vista previa de la forma permanezca cuando el usuario libere el botón del ratón, defina *persistentDraw* como `true`.

Valor devuelto

Ninguno.

Descripción

Método; sitúa a Flash en modo de dibujo. El modo de dibujo se emplea para dibujar temporalmente mientras se presiona el botón del ratón. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo

El ejemplo siguiente sitúa a Flash en modo de dibujo:

```
fl.drawingLayer.beginDraw();
```

drawingLayer.beginFrame()

Disponibilidad

Flash MX 2004

Uso

```
drawingLayer.beginFrame()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; borra lo que se había dibujado anteriormente utilizando drawingLayer y prepara para más comandos de dibujo. Deberá llamarse después de `drawingLayer.beginDraw()`. Todo lo que haya dibujado entre `drawingLayer.beginFrame()` y `drawingLayer.endFrame()` permanecerá en el escenario hasta que llame a los siguientes `beginFrame()` y `endFrame()`. (En este contexto, *frame* hace referencia a dónde comienza y termina el dibujo, no a fotogramas de la línea de tiempo.) Este método sólo suele utilizarse cuando se crean herramientas ampliables. Consulte `drawingLayer.beginDraw()`.

drawingLayer.cubicCurveTo()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.cubicCurveTo(x1Ctrl, y1Ctrl, x2Ctl, y2Ctl, xEnd, yEnd)
```

Parámetros
x1Ctl Un valor de coma flotante que es la ubicación *x* del primer punto de control.

y1Ctl Un valor de coma flotante que es la ubicación *y* del primer punto de control.

x2Ctl Un valor de coma flotante que es la posición *x* del punto intermedio de control.

y2Ctl Un valor de coma flotante que es la posición *y* del punto intermedio de control.

xEnd Un valor de coma flotante que es la posición *x* del punto final de control.

yEnd Un valor de coma flotante que es la posición *y* del punto final de control.

Valor devuelto
Ninguno.

Descripción
Método; dibuja una curva cúbica desde la ubicación actual de la pluma empleando los parámetros como coordenadas del segmento cúbico. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente dibuja una curva cúbica utilizando los puntos de control especificados:

```
fl.drawingLayer.cubicCurveTo(0, 0, 1, 1, 2, 0);
```

drawingLayer.curveTo()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.curveTo(xCtl, yCtl, xEnd, yEnd)
```

Parámetros
xCtl Un valor de coma flotante que es la posición *x* del punto de control.

yCtl Un valor de coma flotante que es la posición *y* del punto de control.

xEnd Un valor de coma flotante que es la posición *x* del punto final de control.

yEnd Un valor de coma flotante que es la posición *y* del punto final de control.

Valor devuelto
Ninguno.

Descripción
Método; dibuja un segmento de curva cuadrática comenzando en la posición de dibujo actual y terminando en un punto especificado. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente dibuja una curva cuadrática utilizando los puntos de control especificados:

```
fl.drawingLayer.curveTo(0, 0, 2, 0);
```

drawingLayer.drawPath()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.drawPath(path)
```

Parámetros
path Un Objeto Path que se va a dibujar.

Valor devuelto
Ninguno.

Descripción
Método; dibuja el trazado especificado por el parámetro *path*. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente dibuja un trazado especificado por el objeto Path llamado `gamePath`:

```
fl.drawingLayer.drawPath(gamePath);
```

drawingLayer.endDraw()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.endDraw()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; sale del modo de dibujo. El modo de dibujo se utiliza cuando se desea dibujar temporalmente mientras se presiona el botón del ratón. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente sale del modo de dibujo:

```
fl.drawingLayer.endDraw();
```

drawingLayer.endFrame()

Disponibilidad
Flash MX 2004

Uso
`drawingLayer.endFrame()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; señala el final de un grupo de comandos de dibujo. Un grupo de comandos de dibujo hace referencia a todo lo que se dibuja entre `drawingLayer.beginFrame()` y `drawingLayer.endFrame()`. La siguiente llamada a `drawingLayer.beginFrame()` borrará todo lo que se haya dibujado en este grupo de comandos de dibujo. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

drawingLayer.lineTo()

Disponibilidad
Flash MX 2004

Uso
`drawingLayer.lineTo(x, y)`

Parámetros
x Un valor de coma flotante que es la coordenada *x* del punto final de la línea que se va a dibujar.

y Un valor de coma flotante que es la coordenada *y* del punto final de la línea que se va a dibujar.

Valor devuelto
Ninguno.

Descripción
Método; dibuja una línea desde la posición de dibujo actual hasta el punto (*x,y*). Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente dibuja una línea desde la posición de dibujo actual hasta el punto (20,30):

```
fl.drawingLayer.lineTo(20, 30);
```

drawingLayer.moveTo()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.moveTo(x, y)
```

Parámetros
x Un valor de coma flotante que especifica la coordenada *x* de la posición en la que se comienza a dibujar.

y Un valor de coma flotante que especifica la coordenada *y* de la posición en la que se comienza a dibujar.

Valor devuelto
Ninguno.

Descripción
Método; establece la posición de dibujo actual. Este método sólo suele utilizarse cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente establece la posición de dibujo actual en el punto (10,15):

```
fl.drawingLayer.moveTo(10, 15);
```

drawingLayer.newPath()

Disponibilidad
Flash MX 2004

Uso
```
drawingLayer.newPath()
```

Parámetros
Ninguno.

Valor devuelto
Un objeto Path.

Descripción
Método; devuelve un nuevo objeto Path. Este método sólo suele utilizarse cuando se crean herramientas ampliables. Consulte Objeto Path.

Ejemplo
El ejemplo siguiente devuelve un objeto Path nuevo:

```
fl.drawingLayer.newPath();
```

drawingLayer.setColor()

Disponibilidad
Flash MX 2004

Uso
`drawingLayer.setColor(color)`

Parámetros
`color` El color de los datos dibujados a continuación en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

Valor devuelto
Ninguno.

Descripción
Método; establece el color de los datos dibujados a continuación. Sólo se aplica a datos persistentes. Para utilizar este método, el parámetro transferido a `drawingLayer.beginDraw()` debe definirse como `true`. Este método sólo suele utilizarse cuando se crean herramientas ampliables. Consulte `drawingLayer.beginDraw()`.

Ejemplo
El ejemplo siguiente dibuja una línea roja en el escenario:

```
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()

Este método no está disponible.

drawingLayer.setStroke()

Este método no está disponible.

Capítulo 13: Objeto Edge

Disponibilidad
Flash MX 2004

Descripción
El objeto Edge representa un borde de una forma en el escenario.

Resumen de métodos
Los métodos siguientes están disponibles para el objeto Edge:

Método	Descripción
`edge.getControl()`	Obtiene un objeto point definido en la ubicación del punto de control especificado del borde.
`edge.getHalfEdge()`	Devuelve un Objeto HalfEdge.
`edge.setControl()`	Establece la posición del punto de control del borde.
`edge.splitEdge()`	Divide el borde en dos partes.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto Edge:

Propiedad	Descripción
`edge.cubicSegmentIndex`	Sólo lectura; un entero que especifica el valor de índice de un segmento cúbico del borde.
`edge.id`	Sólo lectura; un entero que representa un identificador exclusivo para el borde.
`edge.isLine`	Sólo lectura; un entero con un valor de 0 ó 1.
`edge.stroke`	Un Objeto Stroke.

edge.cubicSegmentIndex

Disponibilidad
Flash CS4 Professional

Uso
```
edge.cubicSegmentIndex
```

Descripción
Propiedad de sólo lectura; un entero que especifica el valor de índice de un segmento cúbico del borde (consulte shape.getCubicSegmentPoints()).

Ejemplo
El siguiente código muestra los valores de índice de todos los segmentos cúbicos del borde especificado:

```
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()

Disponibilidad
Flash MX 2004

Uso
`edge.getControl(i)`

Parámetros
i Un entero que especifica qué punto de control del borde se va a devolver. Especifique 0 para el primer punto de control, 1 para el punto de control medio o 2 para el punto de control final. Si la propiedad edge.isLine es `true`, el punto de control medio se definirá como el punto medio del segmento que une los puntos de control inicial y final.

Valor devuelto
El punto de control especificado.

Descripción
Método; obtiene un objeto point definido en la ubicación del punto de control especificado del borde.

Ejemplo
El ejemplo siguiente almacena el primer punto de control de la forma especificada en la variable `pt`:

```
var shape = fl.getDocumentDOM().selection[0];
var pt = shape.edges[0].getControl(0);
```

edge.getHalfEdge()

Disponibilidad
Flash MX 2004

Uso
`edge.getHalfEdge(index)`

Parámetros
index Un entero que especifica qué borde partido se va a devolver. El valor de *index* debe ser 0 para el primer borde partido o 1 para el segundo.

Valor devuelto
Un objeto HalfEdge.

Descripción
Método; devuelve un Objeto HalfEdge.

Ejemplo
El ejemplo siguiente almacena los bordes partidos del borde especificado en las variables `hEdge0` y `hEdge1`:

```
var shape = fl.getDocumentDOM().selection[0];
var edge = shape.edges[0];
var hEdge0 = edge.getHalfEdge(0);
var hEdge1 = edge.getHalfEdge(1);
```

edge.id

Disponibilidad
Flash MX 2004

Uso
`edge.id`

Descripción
Propiedad de sólo lectura; un entero que representa un identificador exclusivo para el borde.

Ejemplo
El ejemplo siguiente almacena un identificador exclusivo para el borde especificado en la variable `my_shape_id`:

```
var shape = fl.getDocumentDOM().selection[0];
var my_shape_id = shape.edges[0].id;
```

edge.isLine

Disponibilidad
Flash MX 2004

Uso
`edge.isLine`

Descripción
Propiedad de sólo lectura; un entero con un valor de 0 o 1. Un valor de 1 indica que el borde es una línea recta. En ese caso, el punto de control medio divide en dos partes iguales la línea que une los dos puntos finales.

Ejemplo
El ejemplo siguiente determina si el borde especificado es una línea recta y muestra un valor de 1 (es una línea recta) o 0 (no es una línea recta) en el panel Salida:

```
var shape = fl.getDocumentDOM().selection[0];
fl.trace(shape.edges[0].isLine);
```

edge.setControl()

Disponibilidad
Flash MX 2004

Uso
```
edge.setControl(index, x, y)
```

Parámetros
index Un entero que especifica qué punto de control se va a definir. Utilice los valores 0, 1 o 2 para especificar los puntos de control inicial, medio y final respectivamente.

x Un valor de coma flotante que especifica la ubicación horizontal del punto de control. Si el escenario se encuentra en modo de edición o de edición en contexto, la coordenada del punto será relativa al objeto editado. En caso contrario, la coordenada del punto será relativa al escenario.

y Un valor de coma flotante que especifica la ubicación vertical del punto de control. Si el escenario se encuentra en modo de edición o de edición en contexto, la coordenada del punto será relativa al objeto editado. En caso contrario, la coordenada del punto será relativa al escenario.

Valor devuelto
Ninguno.

Descripción
Método; establece la posición del punto de control del borde. Deberá llamar a `shape.beginEdit()` antes de utilizar este método. Consulte `shape.beginEdit()`.

Ejemplo
El ejemplo siguiente define el punto de control inicial del borde especificado en las coordenadas (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()

Disponibilidad
Flash MX 2004

Uso
```
edge.splitEdge(t)
```

Parámetros
t Un valor de coma flotante entre 0 y 1 que especifica dónde se divide el borde. Un valor de 0 representa un punto final y un valor 1 representa el otro. Por ejemplo, si se transfiere un valor de 0,5 se dividirá el borde por la mitad, lo que para una línea es exactamente el centro. Si el borde representa una curva, 0,5 representa el medio paramétrico de la curva.

Valor devuelto
Ninguno.

Descripción
Método; divide el borde en dos partes. Deberá llamar a `shape.beginEdit()` antes de utilizar este método.

Ejemplo
El ejemplo siguiente divide el borde especificado por la mitad:

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit()
shape.edges[0].splitEdge( 0.5 );
shape.endEdit()
```

edge.stroke

Disponibilidad
Flash CS4 Professional

Uso
`edge.stroke`

Descripción
Propiedad; un Objeto Stroke.

Ejemplo
En el siguiente ejemplo se muestra el color de trazo del primer borde del objeto seleccionado:

```
var shape = fl.getDocumentDOM().selection[0];
fl.trace(shape.edges[0].stroke.color);
```

Capítulo 14: Objeto Element

Disponibilidad

Flash MX 2004

Descripción

Todo lo que aparece en el escenario es de tipo Element. El ejemplo del código siguiente permite seleccionar un elemento:

```
var el = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
```

Resumen de métodos

Los métodos siguientes están disponibles para el objeto Element:

Método	Descripción
`element.getPersistentData()`	Recupera el valor de los datos especificados por el parámetro ***name***.
`element.getTransformationPoint()`	Obtiene el valor del punto de transformación del elemento especificado.
`element.hasPersistentData()`	Determina si los datos especificados se han asociado al elemento especificado.
`element.removePersistentData()`	Elimina datos persistentes con el nombre especificado que se han asociado al objeto.
`element.setPersistentData()`	Almacena datos con un elemento.
`element.setTransformationPoint()`	Establece la posición del punto de transformación del elemento.

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto Element:

Propiedad	Descripción
`element.depth`	De sólo lectura; un entero que tiene un valor mayor que 0 para la profundidad del objeto en la vista.
`element.elementType`	De sólo lectura; una cadena que representa el tipo de elemento especificado.
`element.height`	Un valor flotante que especifica la altura del elemento en píxeles.
`element.layer`	De sólo lectura; representa el Objeto Layer en el que se encuentra el elemento.
`element.left`	De sólo lectura; un valor flotante que representa el lado izquierdo del elemento.
`element.locked`	Un valor booleano; `true` si el elemento está bloqueado; en caso contrario, `false`.
`element.matrix`	Un Objeto Matrix. La matriz tiene propiedades `a`, `b`, `c`, `d`, `tx` y `ty`. `a`, `b`, `c` y `d` son valores de coma flotante; `tx` y `ty` son coordenadas.
`element.name`	Una cadena que especifica el nombre del elemento, que suele denominarse nombre de instancia.
`element.rotation`	Un valor flotante o entero comprendido entre -180 y 180 que especifica el giro a la derecha, en grados, del objeto.

Propiedad	Descripción
`element.scaleX`	Un valor flotante que especifica el valor de escala *x* de símbolos, objetos de dibujo y rectángulos y óvalos simples.
`element.scaleY`	Un valor flotante que especifica el valor de escala *y* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples.
`element.selected`	Un valor booleano que especifica si el elemento se selecciona o no.
`element.skewX`	Un valor flotante comprendido entre -180 y 180 que especifica el valor de sesgo *x* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples.
`element.skewY`	Un valor flotante comprendido entre -180 y 180 que especifica el valor de sesgo *y* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples.
`element.top`	De sólo lectura; parte superior del elemento.
`element.transformX`	Un número de coma flotante que especifica el valor *x* del punto de transformación del elemento seleccionado, en el sistema de coordenadas del elemento principal del objeto.
`element.transformY`	Un número de coma flotante que especifica el valor *y* del punto de transformación del elemento seleccionado, en el sistema de coordenadas del elemento principal del objeto.
`element.width`	Un valor flotante que especifica el ancho del elemento en píxeles.
`element.x`	Un valor flotante que especifica el valor *x* del punto de registro del elemento seleccionado.
`element.y`	Un valor flotante que especifica el valor *y* del punto de registro del elemento seleccionado.

element.depth

Disponibilidad

Flash MX 2004

Uso

`element.depth`

Descripción

Propiedad de sólo lectura; un entero que tiene un valor mayor que 0 para la profundidad del objeto en la vista. El orden de dibujo de los objetos del escenario especifica cuál está sobre los demás. El orden de los objetos también se puede gestionar con el elemento de menú Modificar > Organizar.

Ejemplo

El ejemplo siguiente muestra la profundidad del elemento especificado en el panel Salida:

```
// Select an object and run this script.
fl.trace("Depth of selected object: " + fl.getDocumentDOM().selection[0].depth);
```

Consulte el ejemplo de `element.elementType`.

element.elementType

Disponibilidad
Flash MX 2004

Uso
`element.elementType`

Descripción
Propiedad de sólo lectura; una cadena que representa el tipo de elemento especificado. El valor es uno de los siguientes: `"shape"`, `"text"`, `"instance"` o `"shapeObj"`. Se crea un `"shapeObj"` con una herramienta ampliable.

Ejemplo
El ejemplo siguiente almacena el tipo del primer elemento en la variable `eType`:

```
// 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
```

El ejemplo siguiente muestra varias propiedades para todos los elementos de la capa o el fotograma actual:

```
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()

Disponibilidad
Flash MX 2004

Uso
`element.getPersistentData(name)`

Parámetros
`name` Una cadena que identifica los datos que se van a devolver.

Valor devuelto
Los datos especificados por el parámetro *name* o 0 si no existen los datos.

Descripción
Método; recupera el valor de los datos especificados por el parámetro *name*. El tipo de datos depende del tipo de datos que se han almacenado (consulte `element.setPersistentData()`). Sólo los símbolos y mapas de bits admiten datos persistentes.

Ejemplo

El ejemplo siguiente define y obtiene datos para el elemento especificado, muestra su valor en el panel Salida y, a continuación, elimina los datos:

```
// 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()

Disponibilidad

Flash CS3 Professional

Uso

`element.getTransformationPoint()`

Parámetros

Ninguno.

Valor devuelto

Un punto (por ejemplo, `{x:10, y:20}`, donde `x` e `y` son números de coma flotante) que especifica la posición del punto de transformación (también *punto de origen* o *punto cero*) en el sistema de coordenadas del elemento.

Descripción

Método; obtiene el valor del punto de transformación del elemento especificado.

Los puntos de transformación tendrán distintas ubicaciones, según el tipo de elemento seleccionado. Para más información, consulte `element.setTransformationPoint()`.

Ejemplo

El ejemplo siguiente obtiene el punto de transformación para el tercer elemento del noveno fotograma en la primera capa del documento. La propiedad `transPoint.x` proporciona la coordenada *x* del punto de transformación. La propiedad `transPoint.y` proporciona la coordenada *y* del punto de transformación.

```
var transPoint =
fl.getDocumentDOM().getTimeline().layers[0].frames[8].elements[2].getTransformationPoint();
```

Véase también

`document.getTransformationPoint()`, `element.setTransformationPoint()`, `element.transformX`, `element.transformY`

element.hasPersistentData()

Disponibilidad
Flash MX 2004

Uso
`element.hasPersistentData(name)`

Parámetros
`name` Una cadena que especifica el nombre del elemento de datos que se va a comprobar.

Valor devuelto
Un valor booleano: `true` si los datos especificados están asociados al objeto; `false` en caso contrario.

Descripción
Método; determina si los datos especificados se han asociado al elemento especificado. Sólo los símbolos y mapas de bits admiten datos persistentes.

Ejemplo
Consulte `element.getPersistentData()`.

element.height

Disponibilidad
Flash MX 2004

Uso
`element.height`

Descripción
Propiedad; un valor flotante que especifica la altura del elemento en píxeles.

No utilice esta propiedad para cambiar el tamaño de un campo de texto. Seleccione el campo de texto y utilice `document.setTextRectangle()`. El uso de esta propiedad con un campo de texto cambia la escala del texto.

Ejemplo
El ejemplo siguiente establece la altura del elemento especificado en 100:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].height = 100;
```

element.layer

Disponibilidad
Flash 8

Uso

```
element.layer
```

Descripción

Propiedad de sólo lectura; representa el Objeto Layer en el que se encuentra el elemento.

Ejemplo

El ejemplo siguiente almacena el objeto Layer que contiene el elemento de la variable `theLayer`:

```
var theLayer = element.layer;
```

element.left

Disponibilidad

Flash MX 2004

Uso

```
element.left
```

Descripción

Propiedad de sólo lectura; un valor flotante que representa el lado izquierdo del elemento. El valor de `element.left` es relativo a la esquina superior izquierda del escenario para elementos que están en una escena, y relativo al punto de registro del símbolo (también *punto de origen* o *punto cero*) si el elemento se almacena en un símbolo. Utilice `document.setSelectionBounds()` o `document.moveSelectionBy()` para establecer esta propiedad.

Ejemplo

El ejemplo siguiente ilustra cómo cambia el valor de esta propiedad cuando se mueve un elemento:

```
// 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);
```

Consulte el ejemplo de `element.elementType`.

element.locked

Disponibilidad

Flash MX 2004

Uso

```
element.locked
```

Descripción

Propiedad; un valor booleano: `true` si el elemento está bloqueado; en caso contrario, `false`. Si el valor de `element.elementType` es `"shape"`, se omitirá esta propiedad.

Ejemplo

El ejemplo siguiente bloquea el primer elemento del primer fotograma de la capa superior:

```
// Similar to Modify > Arrange > Lock:
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].locked = true;
```

element.matrix

Disponibilidad

Flash MX 2004

Uso

`element.matrix`

Descripción

Propiedad; un objeto Matrix. Una matriz tiene propiedades `a`, `b`, `c`, `d`, `tx` y `ty`. Las propiedades `a`, `b`, `c` y `d` son valores de coma flotante; las propiedades `tx` y `ty` son coordenadas. Consulte Objeto Matrix.

Ejemplo

El ejemplo siguiente mueve el elemento especificado 10 píxeles en *x* y 20 píxeles en *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

Disponibilidad

Flash MX 2004

Uso

`element.name`

Descripción

Propiedad; una cadena que especifica el nombre del elemento, que suele denominarse nombre de instancia. Si el valor de `element.elementType` es `"shape"`, se omitirá esta propiedad. Consulte `element.elementType`.

Ejemplo

El ejemplo siguiente define el nombre de instancia del primer elemento del Fotograma 1 en la capa superior como `"clip_mc"`:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].name = "clip_mc";
```

Consulte el ejemplo de `element.elementType`.

element.removePersistentData()

Disponibilidad
Flash MX 2004

Uso
`element.removePersistentData(name)`

Parámetros
`name` Una cadena que especifica el nombre de los datos que se van a eliminar.

Valor devuelto
Ninguno.

Descripción
Método; elimina datos persistentes con el nombre especificado que se han asociado al objeto. Sólo los símbolos y mapas de bits admiten datos persistentes.

Ejemplo
Consulte `element.getPersistentData()`.

element.rotation

Disponibilidad
Flash CS3 Professional

Uso
`element.rotation`

Descripción
Propiedad; un valor flotante o entero comprendido entre -180 y 180 que especifica el giro a la derecha, en grados, del objeto.

Ejemplo
El ejemplo siguiente establece en 45 grados el giro del elemento actualmente seleccionado:

```
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

Disponibilidad
Flash CS3 Professional

Uso

```
element.scaleX
```

Descripción

Propiedad; un valor flotante que especifica el valor de escala *x* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples. El valor de 1 indica una escala de 100%.

Ejemplo

El ejemplo siguiente establece el valor de escala *x* de la selección actual en 2 (dobla su valor):

```
var element = fl.getDocumentDOM().selection[0];
element.scaleX = 2;
```

Véase también

element.scaleY

element.scaleY

Disponibilidad

Flash CS3 Professional

Uso

```
element.scaleY
```

Descripción

Propiedad; un valor flotante que especifica el valor de escala *y* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples. El valor de 1 indica una escala de 100%.

Ejemplo

El ejemplo siguiente establece el valor de escala *y* de la selección actual en 2 (dobla su valor):

```
var element = fl.getDocumentDOM().selection[0];
element.scaleY = 2;
```

Véase también

element.scaleX

element.selected

Disponibilidad

Flash 8

Uso

```
element.selected
```

Descripción
Propiedad; un valor booleano que especifica si el elemento está seleccionado (`true`) o no (`false`).

Ejemplo
El ejemplo siguiente selecciona el elemento:

```
element.selected = true;
```

element.setPersistentData()

Disponibilidad
Flash MX 2004

Uso
```
element.setPersistentData(name, type, value)
```

Parámetros
`name` Una cadena que especifica el nombre que se va a asociar a los datos. Este nombre se utiliza para recuperar los datos.

`type` Una cadena que define el tipo de datos. Los valores válidos son `"integer"`, `"integerArray"`, `"double"`, `"doubleArray"`, `"string"` y `"byteArray"`.

`value` Especifica el valor que se va a asociar al objeto. El tipo de datos de *value* depende del valor del parámetro *type*. El valor especificado deberá ser adecuado al tipo de datos indicado por el parámetro *type*.

Valor devuelto
Ninguno.

Descripción
Método; almacena datos con un elemento. Los datos estarán disponibles cuando se abra el archivo FLA que contiene el elemento. Sólo los símbolos y mapas de bits admiten datos persistentes.

Ejemplo
Consulte `element.getPersistentData()`.

element.setTransformationPoint()

Disponibilidad
Flash CS3 Professional

Uso
```
element.setTransformationPoint(transformationPoint)
```

Parámetros

`transformationPoint` Un punto (por ejemplo, `{x:10, y:20}`, donde `x` e `y` son números de coma flotante) que especifica los valores del punto de transformación de un elemento o grupo.

- Formas: *transformationPoint* se establece en relación al documento (0,0 es la esquina superior izquierda del escenario).
- Símbolos: *transformationPoint* se establece en relación al punto de registro del símbolo (0,0 se ubica en el punto de registro).
- Texto: *transformationPoint* se establece en relación al campo de texto (0,0 es la esquina superior izquierda del campo de texto).
- Mapas de bits/vídeos: *transformationPoint* se establece en relación al mapa de bits o vídeo (0,0 es la esquina superior izquierda del mapa de bits o vídeo).
- Objetos de dibujo, objetos simples y grupos: *transformationPoint* se establece en relación al centro del elemento o grupo (0,0 es el punto central del elemento o grupo).

Valor devuelto

Ninguno.

Descripción

Método; establece la posición del punto de transformación del elemento.

Este método es prácticamente igual que `document.setTransformationPoint()`. Se diferencia en lo siguiente:

- El punto de transformación de los objetos de dibujo, objetos sencillos y grupos se establece en relación al centro del elemento o grupo, no en relación al escenario.
- Se pueden establecer puntos de transformación de elementos sin seleccionarlos primero.

  Este método mueve el punto de transformación, pero no desplaza el elemento. Por el contrario, las propiedades `element.transformX` y `element.transformY` son las que mueven el elemento.

Ejemplo

El ejemplo siguiente establece el punto de transformación del tercer elemento del escenario en 100, 200:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[2].setTransformationPoint({x:
100, y:200});
```

Véase también

`document.setTransformationPoint()`, `element.getTransformationPoint()`, `element.transformX`, `element.transformY`

element.skewX

Disponibilidad

Flash CS3 Professional

Uso

```
element.skewX
```

Descripción

Propiedad; un valor flotante comprendido entre -180 y 180 que especifica el valor de sesgo *x* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples.

Ejemplo

El ejemplo siguiente establece el valor de sesgo *x* de la selección actual en 10:

```
var element = fl.getDocumentDOM().selection[0];
element.skewX = 10;
```

Véase también

`document.setTransformationPoint()`, `element.skewY`

element.skewY

Disponibilidad

Flash CS3 Professional

Uso

```
element.skewY
```

Descripción

Propiedad; un valor flotante comprendido entre -180 y 180 que especifica el valor de sesgo *y* de símbolos, objetos de dibujo, así como de rectángulos y óvalos simples.

Ejemplo

El ejemplo siguiente establece el valor de sesgo *y* de la selección actual en 10:

```
var element = fl.getDocumentDOM().selection[0];
element.skewY = 10;
```

Véase también

`document.setTransformationPoint()`, `element.skewX`

element.top

Disponibilidad

Flash MX 2004

Uso

```
element.top
```

Descripción

Propiedad de sólo lectura; parte superior del elemento. El valor de `element.top` es relativo a la esquina superior izquierda del escenario para elementos que están en una escena, y relativo al punto de registro del símbolo si el elemento se almacena en un símbolo. Utilice `document.setSelectionBounds()` o `document.moveSelectionBy()` para establecer esta propiedad.

Ejemplo

El ejemplo siguiente muestra cómo cambia el valor de esta propiedad cuando se mueve un elemento:

```
// 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);
```

Consulte el ejemplo de `element.elementType`.

element.transformX

Disponibilidad

Flash CS3 Professional

Uso

`element.transformX`

Descripción

Propiedad; un número de coma flotante que especifica el valor *x* del punto de transformación del elemento seleccionado, en el sistema de coordenadas del elemento principal del objeto. Si se establece esta propiedad con un nuevo valor, se mueve el elemento. Por el contrario, el método `element.setTransformationPoint()` mueve el punto de transformación pero no el elemento.

Ejemplo

Véase también

`element.getTransformationPoint()`, `element.setTransformationPoint()`, `element.transformY`

element.transformY

Disponibilidad

Flash CS3 Professional

Uso

`element.transformY`

Descripción

Propiedad; un número de coma flotante que especifica el valor *y* del punto de transformación del elemento seleccionado, en el sistema de coordenadas del elemento principal del objeto. Si se establece esta propiedad con un nuevo valor, se mueve el elemento. Por el contrario, el método `element.setTransformationPoint()` mueve el punto de transformación pero no el elemento.

Véase también

`element.getTransformationPoint()`, `element.setTransformationPoint()`, `element.transformX`

element.width

Disponibilidad

Flash MX 2004

Uso

```
element.width
```

Descripción

Propiedad; un valor flotante que especifica el ancho del elemento en píxeles.

No utilice esta propiedad para cambiar el tamaño de un campo de texto. Seleccione el campo de texto y utilice `document.setTextRectangle()`. El uso de esta propiedad con un campo de texto cambia la escala del texto.

Ejemplo

El ejemplo siguiente establece el ancho del elemento especificado en 100:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].width= 100;
```

element.x

Disponibilidad

Flash CS3 Professional

Uso

```
element.x
```

Descripción

Propiedad; un valor flotante que especifica el valor *x* del punto de registro del elemento seleccionado.

Ejemplo

El ejemplo siguiente establece el valor del punto de registro del elemento especificado en 100, 200:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].x= 100;
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].y= 200;
```

Véase también

`element.y`

element.y

Disponibilidad

Flash CS3 Professional

Uso

```
element.y
```

Descripción

Propiedad; un valor flotante que especifica el valor *y* del punto de registro del elemento seleccionado.

Ejemplo

Consulte `element.x`

Capítulo 15: Objeto Fill

Disponibilidad
Flash MX 2004

Descripción
Este objeto contiene todas las propiedades de la configuración del color de relleno del panel Herramientas o de una forma seleccionada. Para recuperar un objeto Fill, utilice `document.getCustomFill()`.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto Fill:

Propiedad	Descripción
`fill.bitmapIsClipped`	Valor booleano que especifica si el relleno de mapa de bits para una forma más grande que el mapa de bits se recorta o se repite.
`fill.bitmapPath`	Cadena que especifica la ruta y el nombre del relleno de mapa de bits en la biblioteca.
`fill.color`	Una cadena, valor hexadecimal o entero que representa el color de relleno.
`fill.colorArray`	Conjunto de colores en degradado.
`fill.focalPoint`	Un entero que especifica el desplazamiento horizontal del punto focal del degradado desde el punto de transformación.
`fill.linearRGB`	Un valor booleano que especifica si se debe representar el relleno como un degradado RGB radial o lineal.
`fill.matrix`	Un Objeto Matrix que define la ubicación, la orientación y las escalas de los rellenos con degradado.
`fill.overflow`	Cadena que especifica el comportamiento del desbordamiento de un degradado.
`fill.posArray`	Conjunto de enteros, cada uno en el intervalo 0-255, que indica la posición del color correspondiente.
`fill.style`	Cadena que especifica el estilo de relleno.

fill.bitmapIsClipped

Disponibilidad
Flash CS4 Professional

Uso
```
fill.bitmapIsClipped
```

Descripción
Propiedad; valor booleano que especifica si el relleno de mapa de bits para una forma más grande que el mapa de bits se recorta (`true`) o se repite (`false`). Esta propiedad sólo está disponible si el valor de la propiedad `fill.style` es `"bitmap"`. Si la forma es más pequeña que el mapa de bits, este valor es `false`.

Ejemplo

El ejemplo siguiente muestra información que indica si el mapa de bits está recortado, si es necesario, en el panel Salida:

```
var fill = fl.getDocumentDOM().getCustomFill();
if (fill.style == "bitmap")
    fl.trace("Fill image is clipped: " + fill.bitmapIsClipped);
```

Véase también

`fill.bitmapPath`

fill.bitmapPath

Disponibilidad

Flash CS4 Professional

Uso

`fill.bitmapPath`

Descripción

Cadena que especifica la ruta y el nombre del relleno de mapa de bits en la biblioteca. Esta propiedad sólo está disponible si el valor de la propiedad `fill.style` es `"bitmap"`.

Ejemplo

El siguiente ejemplo define el estilo de relleno del elemento especificado como una imagen de mapa de bits en la biblioteca:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "bitmap";
fill.bitmapPath = "myBitmap.jpg";
fl.getDocumentDOM().setCustomFill(fill);
```

Véase también

`fill.bitmapIsClipped`

fill.color

Disponibilidad

Flash MX 2004

Uso

`fill.color`

Descripción

Propiedad; el color del relleno, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`

- Un entero que representa el equivalente decimal del número hexadecimal

Ejemplo

El ejemplo siguiente define el color de relleno de la selección actual:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = "#FFFFFF";
fl.getDocumentDOM().setCustomFill( fill );
```

fill.colorArray

Disponibilidad

Flash MX 2004

Uso

```
fill.colorArray
```

Descripción

Propiedad; un conjunto de colores en degradado, expresado en enteros. Esta propiedad sólo está disponible si el valor de la propiedad `fill.style` es `"radialGradient"` o `"linearGradient"`. Consulte `fill.style`.

Ejemplo

El ejemplo siguiente muestra el conjunto de colores de la selección actual, si es necesario, en el panel Salida:

```
var fill = fl.getDocumentDOM().getCustomFill();
if(fill.style == "linearGradient" || fill.style == "radialGradient")
    alert(fill.colorArray);
```

El ejemplo siguiente establece el relleno con el degradado lineal especificado:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "linearGradient";
fill.colorArray = ["#00ff00","#ff00ff"];
fill.posArray = [0, 255];
fl.getDocumentDOM().setCustomFill(fill);
```

fill.focalPoint

Disponibilidad

Flash 8

Uso

```
fill.focalPoint
```

Descripción

Propiedad; un entero que especifica el desplazamiento horizontal del punto focal del degradado desde el punto de transformación. Un valor de 10, por ejemplo, situaría el punto focal a 10/255 de la distancia desde el punto de transformación hasta el borde del degradado. Un valor de -255 situaría el punto focal en el límite izquierdo del degradado. El valor predeterminado es 0.

Esta propiedad sólo está disponible si el valor de la propiedad `fill.style` es `"radialGradient"`.

Ejemplo

El ejemplo siguiente define el punto focal de un degradado radial para la selección actual en 100 píxeles hacia la derecha del centro de la forma:

```
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

Disponibilidad

Flash 8

Uso

```
fill.linearRGB
```

Descripción

Propiedad; un valor booleano que especifica si se debe representar el relleno como un degradado RGB radial o lineal. Defina esta propiedad como `true` para especificar una interpolación lineal de un degradado, o como `false` para especificar una interpolación radial de un degradado. El valor predeterminado es `false`.

Ejemplo

El ejemplo siguiente especifica que el degradado de la selección actual se debe representar con un valor RGB lineal:

```
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

Disponibilidad

Flash MX 2004

Uso

`fill.matrix`

Descripción

Propiedad, un Objeto Matrix que define la ubicación, la orientación y las escalas de los rellenos con degradado.

Ejemplo

El ejemplo siguiente utiliza la propiedad `fill.matrix` para especificar un relleno con degradado para la selección actual:

```
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

Disponibilidad

Flash 8

Uso

`fill.overflow`

Descripción

Propiedad; una cadena que especifica el comportamiento del desbordamiento de un degradado. Los valores aceptables son `"extend"`, `"repeat"` y `"reflect"`; en las cadenas no se distingue el uso de mayúsculas y minúsculas. El valor predeterminado es `"extend"`.

Ejemplo

El ejemplo siguiente especifica que el comportamiento del desbordamiento para la selección actual debe ser `"extend"`:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.overflow = "extend";
fl.getDocumentDOM().setCustomFill(fill);
```

fill.posArray

Disponibilidad

Flash MX 2004

Uso

`fill.posArray`

Descripción

Propiedad; un conjunto de enteros, cada uno en el intervalo 0-255, que indica la posición del color correspondiente. Esta propiedad sólo está disponible si el valor de la propiedad `fill.style` es `"radialGradient"` o `"linearGradient"`.

Ejemplo

El ejemplo siguiente especifica los colores que se van a utilizar en un degradado lineal para la selección actual:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray= [0,100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

fill.style

Disponibilidad

Flash MX 2004 Valor "bitmap" añadido en Flash CS4 Professional.

Uso

`fill.style`

Descripción

Propiedad; una cadena que especifica el estilo de relleno. Los valores aceptables son `"bitmap"`, `"solid"`, `"linearGradient"`, `"radialGradient"` y `"noFill"`.

Si este valor es `"linearGradient"` o `"radialGradient"`, también estarán disponibles las propiedades `fill.colorArray` y `fill.posArray`. Si este valor es `"bitmap"`, también estarán disponibles las propiedades `fill.bitmapIsClipped` y `fill.bitmapPath`.

Ejemplo

El ejemplo siguiente especifica los colores que se van a utilizar en un degradado lineal para la selección actual:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style= "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray= [0,100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

Capítulo 16: Objeto Filter

Disponibilidad

Flash 8

Descripción

Este objeto contiene todas las propiedades para todos los filtros. La propiedad `filter.name` especifica el tipo de filtro y determina qué propiedades se pueden aplicar a cada filtro. Consulte `filter.name`.

Para devolver la lista de filtros de un objeto u objetos, utilice `document.getFilters()`. Para aplicar los filtros a un objeto u objetos, utilice `document.setFilters()`. Consulte `document.getFilters()` y `document.setFilters()`.

Resumen de propiedades

Pueden emplearse las propiedades siguientes con el objeto Filter:

Propiedad	Descripción
`filter.angle`	Un valor flotante que especifica el ángulo del color de la sombra o de resaltado, en grados.
`filter.blurX`	Un valor flotante que especifica la cantidad de desenfoque en la dirección x, expresado en píxeles.
`filter.blurY`	Un valor flotante que especifica la cantidad de desenfoque en la dirección y.
`filter.brightness`	Un valor flotante que especifica el brillo del filtro.
`filter.color`	Una cadena, valor hexadecimal o entero que representa el color del filtro.
`filter.contrast`	Un valor flotante que especifica el valor de contraste del filtro.
`filter.distance`	Un valor flotante que especifica la distancia entre el efecto del filtro y un objeto, en píxeles.
`filter.enabled`	Un valor booleano que especifica si está activado el filtro especificado.
`filter.hideObject`	Un valor booleano que especifica si está oculta la imagen de origen.
`filter.highlightColor`	Una cadena, valor hexadecimal o entero que representa el color de resaltado.
`filter.hue`	Un valor flotante que especifica el matiz del filtro.
`filter.inner`	Un valor booleano que especifica si la sombra es interior.
`filter.knockout`	Un valor booleano que especifica si el filtro es extractor.
`filter.name`	De sólo lectura; una cadena que especifica el tipo de filtro.
`filter.quality`	Una cadena que especifica la calidad del desenfoque.
`filter.saturation`	Un valor flotante que especifica el valor de saturación del filtro.
`filter.shadowColor`	Una cadena, valor hexadecimal o entero que representa el color de sombra.
`filter.strength`	Un entero que especifica el porcentaje de intensidad del filtro.
`filter.type`	Una cadena que especifica el tipo de bisel o iluminado.

filter.angle

Disponibilidad

Flash 8

Uso

`filter.angle`

Descripción

Propiedad; un valor flotante que especifica el ángulo del color de la sombra o de resaltado, en grados. Los valores aceptables oscilan entre 0 y 360. Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"dropShadowFilter"`, `"gradientBevelFilter"` o `"gradientGlowFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el ángulo en 120 para los filtros de bisel en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.blurX

Disponibilidad

Flash 8

Uso

`filter.blurX`

Descripción

Propiedad; un valor flotante que especifica la cantidad de desenfoque en la dirección*x*, expresado en píxeles. Los valores aceptables oscilan entre 0 y 255. Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"blurFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientBevelFilter"` o `"gradientGlowFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el valor `blurX` en 30 y el valor `blurY` en 20 para los filtros de desenfoque en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`, `filter.blurY`

filter.blurY

Disponibilidad

Flash 8

Uso

`filter.blurY`

Descripción

Propiedad; un valor flotante que especifica la cantidad de desenfoque en la dirección *y*, expresado en píxeles. Los valores aceptables oscilan entre 0 y 255. Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"blurFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientBevelFilter"` o `"gradientGlowFilter"` para la propiedad `filter.name`.

Ejemplo

Consulte `filter.blurX`.

Véase también

`document.setFilterProperty()`, `filter.blurX`

filter.brightness

Disponibilidad

Flash 8

Uso

`filter.brightness`

Descripción

Propiedad; un valor flotante que especifica el brillo del filtro. Los valores aceptables oscilan entre -100 y 100. Esta propiedad se define para los objetos Filter con un valor de `"adjustColorFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el brillo en 30,5 para los filtros de ajuste de color en el objeto u objetos seleccionados:

```
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

Disponibilidad

Flash 8

Uso

`filter.color`

Descripción

Propiedad; el color del filtro, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

  Esta propiedad se define para los objetos Filter con un valor de `"dropShadowFilter"` o `"glowFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el color en `"#ff00003e"` para los filtros de sombra en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.contrast

Disponibilidad

Flash 8

Uso

`filter.contrast`

Descripción

Propiedad; un valor flotante que especifica el valor de contraste del filtro. Los valores aceptables oscilan entre -100 y 100. Esta propiedad se define para los objetos Filter con un valor de `"adjustColorFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el valor de contraste en -15,5 para los filtros de ajuste de color en el objeto u objetos seleccionados:

```
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

Disponibilidad

Flash 8

Uso

`filter.distance`

Descripción

Propiedad; un valor flotante que especifica la distancia entre el efecto del filtro y un objeto, en píxeles. Los valores aceptables son de -255 a 255. Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"dropShadowFilter"`, `"gradientBevelFilter"` o `"gradientGlowFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece la distancia en 10 píxeles para los filtros de sombra en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.enabled

Disponibilidad
Flash CS3 Professional

Uso
`filter.enabled`

Descripción
Propiedad; valor booleano que especifica si el filtro especificado está activado (`true`) o desactivado (`false`).

Ejemplo
El ejemplo siguiente desactiva los filtros de color en el objeto u objetos seleccionados:

```
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

Disponibilidad
Flash 8

Uso
`filter.hideObject`

Descripción
Propiedad; valor booleano que especifica si la imagen de origen está oculta (`true`) o se muestra (`false`). Esta propiedad se define para los objetos Filter con un valor de `"dropShadowFilter"` para la propiedad `filter.name`.

Ejemplo
El ejemplo siguiente establece el valor de `hideObject` como `true` para los filtros de sombra en el objeto u objetos seleccionados:

```
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

Disponibilidad
Flash 8

Uso
`filter.highlightColor`

Descripción
Propiedad; el color del resaltado, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

  Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"` para la propiedad `filter.name`.

Ejemplo
El ejemplo siguiente establece el color de resaltado en `"#ff00003e"` para los filtros de bisel en el objeto u objetos seleccionados:

```
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

Disponibilidad
Flash 8

Uso
`filter.hue`

Descripción
Propiedad; un valor flotante que especifica el matiz del filtro. Los valores aceptables oscilan entre -180 y 180. Esta propiedad se define para los objetos Filter con un valor de `"adjustColorFilter"` para la propiedad `filter.name`.

Ejemplo
El ejemplo siguiente establece el matiz en 120 para los filtros de ajuste de color en el objeto u objetos seleccionados:

```
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

Disponibilidad

Flash 8

Uso

`filter.inner`

Descripción

Propiedad; valor booleano que especifica si la sombra es interior (`true`) o no (`false`). Esta propiedad se define para los objetos Filter con un valor de `"dropShadowFilter"` o `"glowFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el valor de la propiedad `inner` como `true` para los filtros de iluminado en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.knockout

Disponibilidad

Flash 8

Uso

`filter.knockout`

Descripción

Propiedad; valor booleano que especifica si el filtro es extractor (`true`) o no (`false`). Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientBevelFilter"` o `"gradientGlowFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece la propiedad `knockout` como `true` para los filtros de iluminado en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.name

Disponibilidad

Flash 8

Uso

```
filter.name
```

Descripción

Propiedad de sólo lectura; una cadena que especifica el tipo de filtro. El valor de esta propiedad determina las demás propiedades del objeto Filter que están disponibles. Este valor puede ser: `"adjustColorFilter"`, `"bevelFilter"`, `"blurFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientBevelFilter"` o `"gradientGlowFilter"`.

Ejemplo

El ejemplo siguiente muestra los nombres del filtro y las posiciones de índice en el panel Salida:

```
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);
```

Véase también

`document.getFilters()`, `document.setFilterProperty()`

filter.quality

Disponibilidad

Flash 8

Uso

`filter.quality`

Descripción

Propiedad; una cadena que especifica la calidad del desenfoque. Los valores aceptables son `"low"`, `"medium"` y `"high"` (`"high"` es similar a un desenfoque gaussiano). Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"blurFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientGlowFilter"` o `"gradientBevelFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece la calidad del desenfoque en `"medium"` para los filtros de iluminado en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.saturation

Disponibilidad

Flash 8

Uso

`filter.saturation`

Descripción

Propiedad; un valor flotante que especifica el valor de saturación del filtro. Los valores aceptables son de -100 a 100. Esta propiedad se define para los objetos Filter con un valor de `"adjustColorFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el valor de saturación en -100 (escala de grises) para los filtros de ajuste de color en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.shadowColor

Disponibilidad

Flash 8

Uso

`filter.shadowColor`

Descripción

Propiedad; el color de la sombra, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

  Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el color de sombra en `"#ff00003e"` para los filtros de bisel en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.strength

Disponibilidad

Flash 8

Uso

`filter.strength`

Descripción

Propiedad; un entero que especifica el porcentaje de intensidad del filtro. Los valores aceptables oscilan entre 0 y 25.500. Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"dropShadowFilter"`, `"glowFilter"`, `"gradientGlowFilter"` o `"gradientBevelFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece la intensidad en 50 para los filtros de iluminado en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

filter.type

Disponibilidad

Flash 8

Uso

`filter.type`

Descripción

Propiedad; una cadena que especifica el tipo de bisel o iluminado. Los valores aceptables son `"inner"`, `"outer"` y `"full"`. Esta propiedad se define para los objetos Filter con un valor de `"bevelFilter"`, `"gradientGlowFilter"` o `"gradientBevelFilter"` para la propiedad `filter.name`.

Ejemplo

El ejemplo siguiente establece el tipo en `"full"` para los filtros de bisel en el objeto u objetos seleccionados:

```
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);
```

Véase también

`document.setFilterProperty()`

Capítulo 17: Objeto flash (fl)

Disponibilidad

Flash MX 2004

Descripción

El objeto flash representa la aplicación Flash. Puede utilizar `flash` o `fl` para hacer referencia a este objeto. Esta documentación utiliza `fl` en los ejemplos de código.

Resumen de métodos

Pueden emplearse los métodos siguientes con el objeto flash:

Método	Descripción
`fl.addEventListener()`	Registra una función que se llamará cuando tenga lugar un evento concreto.
`fl.browseForFileURL()`	Abre el cuadro de diálogo del sistema Abrir archivo o Guardar archivo y permite que el usuario especifique un archivo para abrir o guardar.
`fl.browseForFolderURL()`	Muestra el cuadro de diálogo Buscar carpeta y permite que el usuario seleccione una carpeta.
`fl.clipCopyString()`	Copia la cadena especificada en el Portapapeles.
`fl.closeAll()`	Cierra todos los documentos abiertos y aparece el cuadro de diálogo Guardar como para todos los documentos que no se guardaron previamente.
`fl.closeAllPlayerDocuments()`	Cierra todos los archivos SWF que estaban abiertos con Control > Probar película.
`fl.closeDocument()`	Cierra el documento especificado.
`fl.createDocument()`	Abre un documento nuevo y lo selecciona.
`fl.downloadLatestVersion()`	Descarga del servidor Version Cue la última versión de un archivo que no esté actualmente abierto.
`fl.fileExists()`	Comprueba si el archivo ya existe en el disco.
`fl.findDocumentDOM()`	Permite buscar un archivo específico utilizando su identificador único.
`fl.findDocumentIndex()`	Devuelve un conjunto de enteros que representan la posición de un documento en el conjunto `fl.documents`.
`fl.findObjectInDocByName()`	Expone los elementos con nombres de instancias que coinciden con texto especificado.
`fl.findObjectInDocByType()`	Busca en un documento elementos de un tipo especificado.
`fl.getAppMemoryInfo()`	Devuelve un entero que representa el número de bytes que se están utilizando en un área especificada de memoria Flash.exe.
`fl.getDocumentDOM()`	Recupera el DOM (Objeto Document) del documento activo actualmente.
`fl.isFontInstalled()`	Determina si está instalada una fuente especificada.
`fl.mapPlayerURL()`	Asigna una URL Unicode de escape a una URL UTF-8 o MBCS.

Método	Descripción
`fl.openDocument()`	Abre un documento de Flash (archivo FLA) para editarlo en una nueva ventana de documento de Flash y lo selecciona.
`fl.openScript()`	Abre un archivo de script (JSFL, AS, ASC) o de otro tipo (XML, TXT) en el editor de texto de Flash.
`fl.quit()`	Sale de Flash y pregunta al usuario si desea guardar los documentos modificados.
`fl.reloadTools()`	Recrea el panel Herramientas a partir del archivo toolconfig.xml. Sólo se utiliza para crear herramientas ampliables.
`fl.removeEventListener()`	Anula el registro de una función que se registró con `fl.addEventListener()`.
`fl.resetAS3PackagePaths()`	Restablece la configuración de ruta de clases global en el cuadro de diálogo Configuración de ActionScript 3.0 con el valor predeterminado.
`fl.resetPackagePaths()`	Restablece la configuración de ruta de clases global en el cuadro de diálogo Configuración de ActionScript 2.0 con el valor predeterminado.
`fl.revertDocumentToLastVersion()`	Devuelve el documento especificado a la versión del servidor Version Cue.
`fl.runScript()`	Ejecuta un archivo JavaScript.
`fl.saveAll()`	Guarda todos los documentos abiertos y aparece el cuadro de diálogo Guardar como para todos los documentos que no se guardaron previamente.
`fl.saveAVersionOfDocument()`	Guarda una versión del documento especificado en el servidor Version Cue.
`fl.saveDocument()`	Guarda el documento especificado como documento FLA.
`fl.saveDocumentAs()`	Muestra el cuadro de diálogo Guardar como para el documento especificado.
`fl.selectElement()`	Activa la selección o edición de un elemento.
`fl.selectTool()`	Selecciona la herramienta especificada en el panel Herramientas.
`fl.setActiveWindow()`	Establece el documento especificado como ventana activa.
`fl.showIdleMessage()`	Permite desactivar la advertencia sobre un script con una ejecución demasiado larga.
`fl.synchronizeDocumentWithHeadVersion()`	Sincroniza el documento especificado con la versión más reciente en el servidor Version Cue.
`fl.trace()`	Envía una cadena de texto al panel Salida.

Resumen de propiedades

Pueden emplearse las propiedades siguientes con el objeto flash.

Propiedad	Descripción
`fl.actionsPanel`	Sólo lectura; un Objeto actionsPanel.
`fl.as3PackagePaths`	Una cadena que corresponde a la configuración de rutas de clases global en el cuadro de diálogo Configuración de ActionScript 3.0.
`fl.compilerErrors`	Sólo lectura; un Objeto compilerErrors.
`fl.componentsPanel`	Sólo lectura; un Objeto componentsPanel que representa el panel Componentes.
`fl.configDirectory`	Sólo lectura; cadena que especifica la ruta completa de la carpeta Configuration del usuario local con la ruta específica de la plataforma.
`fl.configURI`	Sólo lectura; cadena que especifica la ruta completa de la carpeta Configuration del usuario local como URI file:///.
`fl.contactSensitiveSelection`	Un valor booleano que especifica si el modo de selección Por contacto está activado.
`fl.createNewDocList`	Sólo lectura; conjunto de cadenas que representa los distintos tipos de documentos que se pueden crear.
`fl.createNewDocListType`	Sólo lectura; conjunto de cadenas que representa las extensiones de archivo de los tipos de documentos que se pueden crear.
`fl.createNewTemplateList`	Sólo lectura; conjunto de cadenas que representa los distintos tipos de plantillas que se pueden crear.
`fl.documents`	Sólo lectura; conjunto de objetos Document (consulte Objeto Document) que representa los documentos (archivos FLA) que están abiertos para editar.
`fl.drawingLayer`	Sólo lectura; el Objeto drawingLayer que una herramienta ampliable debe utilizar cuando el usuario desee dibujar temporalmente mientras arrastra el ratón.
`fl.externalLibraryPath`	Cadena que contiene una lista de elementos de la ruta de biblioteca externa global de ActionScript 3.0, donde se especifica la ubicación de los archivos SWC utilizados como bibliotecas compartidas de tiempo de ejecución.
`fl.flexSDKPath`	Cadena que especifica la ruta de la carpeta Flex SDK, que contiene, entre otras, las carpetas bin, frameworks y lib.
`fl.libraryPath`	Cadena que contiene una lista de elementos de la ruta de biblioteca global de ActionScript 3.0, donde se especifica la ubicación de los archivos SWC o de las carpetas que contienen los archivos SWC.
`fl.Math`	Sólo lectura; el Objeto Math proporciona métodos para operaciones de matrices y puntos.
`fl.mruRecentFileList`	Sólo lectura; conjunto de los nombres completos de archivo de la lista Usados recientemente (MRU) que gestiona la herramienta de edición de Flash.
`fl.mruRecentFileListType`	Sólo lectura; conjunto de los tipos de archivo de la lista MRU que gestiona la herramienta de edición de Flash.
`fl.packagePaths`	Cadena que corresponde a la configuración de rutas de clases global en el cuadro de diálogo Configuración de ActionScript 2.0.
`fl.objectDrawingMode`	Un entero que representa el modo de dibujo del objeto que está activado.
`fl.outputPanel`	Sólo lectura; referencia al Objeto outputPanel.
`fl.presetPanel`	Sólo lectura; un Objeto presetPanel.
`fl.scriptURI`	Sólo lectura; cadena que representa la ruta del script JSFL que se ejecuta actualmente, expresada como URI file:///.

Propiedad	Descripción
`fl.sourcePath`	Cadena que contiene una lista de elementos de la ruta de origen global de ActionScript 3.0, donde se especifica la ubicación de los archivos de clases de ActionScript.
`fl.swfPanels`	Conjunto de objetos swfPanel registrados (consulte Objeto swfPanel).
`fl.tools`	Sólo lectura; conjunto de objetos Tools.
`fl.version`	Sólo lectura; la versión de cadena larga de la herramienta de edición de Flash, incluida la plataforma.
`fl.xmlui`	Sólo lectura; un Objeto XMLUI.

fl.actionsPanel

Disponibilidad
Flash CS3 Professional

Uso
```
fl.actionsPanel
```

Descripción
Propiedad de sólo lectura; un objeto actionsPanel que representa el panel Acciones mostrado actualmente. Para obtener información sobre la utilización de esta propiedad, consulte Objeto actionsPanel.

fl.addEventListener()

Disponibilidad
Flash CS3 Professional

Uso
```
fl.addEventListener(eventType, callbackFunction)
```

Parámetros
`eventType` Una cadena que especifica el tipo de evento que va a pasar a esta función callback. Los valores aceptables son `"documentNew"`, `"documentOpened"`, `"documentClosed"`, `"mouseMove"`, `"documentChanged"`, `"layerChanged"` y `"frameChanged"`.

El valor `documentChanged` no implica que el contenido del documento haya cambiado; significa que ahora hay un documento diferente en primer plano. Es decir, `fl.getDocumentDOM()` devolverá un valor distinto que antes de que se produjese el evento.

`callbackFunction` Nombre de la función que desea ejecutar cada vez que tiene lugar el evento.

Valor devuelto
Ninguno.

Descripción

Método; registra una función que se llamará cuando tenga lugar un evento concreto.

Al utilizar este método, tenga en cuenta que si el evento se produce con frecuencia (como podría ser el caso de `mouseMove`) y la función tarda mucho tiempo en ejecutarse, la aplicación podría colgarse o entrar en un estado de error.

Ejemplo

El siguiente ejemplo muestra un mensaje en el panel Salida cuando se cierra un documento:

```
myFunction = function () {
    fl.trace('document was closed'); }
fl.addEventListener("documentClosed", myFunction);
```

Véase también

`fl.removeEventListener()`

fl.as3PackagePaths

Disponibilidad

Flash CS3 Professional

Uso

```
fl.as3PackagePaths
```

Descripción

Propiedad; una cadena que corresponde a la configuración de rutas de clases global en el cuadro de diálogo Configuración de ActionScript 3.0. Los elementos de la cadena se delimitan con punto y coma. Para ver o cambiar la configuración de rutas de clases de ActionScript 2.0, utilice `fl.packagePaths`.

Ejemplo

El ejemplo siguiente ilustra el cambio de la configuración de rutas de clases de 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
```

Véase también

`fl.resetAS3PackagePaths()`

fl.browseForFileURL()

Disponibilidad
Flash MX 2004

Uso
```
fl.browseForFileURL(browseType [, title [, previewArea]])
```

Parámetros
`browseType` Una cadena que especifica el tipo de operación de búsqueda de archivo. Los valores aceptables son `"open"`, `"select"` y `"save"`. Los valores `"open"` y `"select"` abren el cuadro de diálogo del sistema Abrir archivo. Cada valor se suministra por compatibilidad con Dreamweaver. El valor `"save"` abre el cuadro de diálogo del sistema Guardar archivo.

`title` Una cadena que especifica el título del cuadro de diálogo Abrir archivo o Guardar archivo. Si se omite este parámetro se empleará un valor predeterminado. Este parámetro es opcional.

`previewArea` Un parámetro opcional que omiten Flash y Fireworks y sólo está presente por compatibilidad con Dreamweaver.

Valor devuelto
La URL del archivo, expresada como URI file:///; devuelve `null` si el usuario cancela el cuadro de diálogo.

Descripción
Método; abre el cuadro de diálogo del sistema Abrir archivo o Guardar archivo y permite que el usuario especifique un archivo para abrir o guardar.

Ejemplo
El ejemplo siguiente permite que el usuario elija un archivo FLA para abrir y, a continuación, abre el archivo. (El método `fl.browseForFileURL()` puede buscar cualquier tipo de archivo, pero `fl.openDocument()` sólo puede abrir archivos FLA.)

```
var fileURL = fl.browseForFileURL("open", "Select file");
var doc = fl.openDocument(fileURL);
```

Véase también
`fl.browseForFolderURL()`

fl.browseForFolderURL()

Disponibilidad
Flash 8

Uso
```
fl.browseForFolderURL([description])
```

Parámetros

`description` Una cadena opcional que especifica la descripción del cuadro de diálogo Buscar carpeta. Si se omite este parámetro, no se mostrará nada en el área de descripción.

Valor devuelto

La URL de la carpeta, expresada como URI file:///; devuelve `null` si el usuario cancela el cuadro de diálogo.

Descripción

Método; muestra el cuadro de diálogo Buscar carpeta y permite que el usuario seleccione una carpeta.

Nota: *el título del cuadro de diálogo es siempre Buscar carpeta. Utilice el parámetro `description` para añadir más detalles en el área de descripción debajo del título, como "Seleccione una carpeta" o "Seleccione la ruta que contiene el perfil que desea importar".*

Ejemplo

El ejemplo siguiente permite que el usuario seleccione una carpeta y, a continuación, muestra la lista de archivos de esa carpeta:

```
var folderURI = fl.browseForFolderURL("Select a folder.");
var folderContents = FLfile.listFolder(folderURI);
```

Véase también

`fl.browseForFileURL()`, Objeto FLfile

fl.clipCopyString()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.clipCopyString(string)
```

Parámetros

`string` Una cadena que se va a copiar en el Portapapeles.

Valor devuelto

Ninguno.

Descripción

Método; copia la cadena especificada en el Portapapeles.

Para copiar la selección actual en el Portapapeles, utilice `document.clipCopy()`.

Ejemplo

El ejemplo siguiente copia la ruta del documento actual en el Portapapeles:

```
var documentPath = fl.getDocumentDOM().path;
fl.clipCopyString(documentPath);
```

fl.closeAll()

Disponibilidad
Flash MX 2004

Uso
```
fl.closeAll([bPromptToSave])
```

Parámetros
`bPromptToSave` Valor booleano opcional que indica si se muestra el cuadro de diálogo Guardar para todos los archivos modificados desde el proceso de guardado anterior, o el cuadro de diálogo Guardar como para los archivos que nunca se hayan guardado. El valor predeterminado es `true`.

Valor devuelto
Ninguno.

Descripción
Método; cierra todos los archivos abiertos (archivos FLA, SWF, JSFL, etc.). Si quiere cerrar todos los archivos abiertos sin guardar los cambios en ninguno de ellos, transfiera `false` a *bPromptToSave*. Este método no cierra la aplicación.

Ejemplo
El código siguiente abre todos los archivos y pide al usuario que guarde los nuevos o los modificados.

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

Véase también
`fl.closeAllPlayerDocuments()`, `fl.closeDocument()`

fl.closeAllPlayerDocuments()

Disponibilidad
Flash CS3 Professional

Uso
```
fl.closeAllPlayerDocuments()
```

Parámetros
Ninguno.

Valor devuelto
Un valor booleano: `true` si había abiertas una o varias ventanas de película; `false` en caso contrario.

Descripción
Método; cierra todos los archivos SWF que estaban abiertos con Control > Probar película.

Ejemplo

En el ejemplo siguiente se cierran todos los archivos SWF que estaban abiertos con Control > Probar película.

```
fl.closeAllPlayerDocuments();
```

Véase también

`fl.closeAll()`, `fl.closeDocument()`

fl.closeDocument()

Disponibilidad

Flash MX 2004

Uso

```
fl.closeDocument(documentObject [, bPromptToSaveChanges])
```

Parámetros

`documentObject` Un Objeto Document. Si *documentObject* hace referencia al documento activo, es posible que la ventana Documento no se cierre hasta que termine de ejecutarse el script que llama a este método.

`bPromptToSaveChanges` Un valor booleano. Cuando *bPromptToSaveChanges* tiene el valor `false`, no se mostrará un mensaje al usuario si el documento contiene cambios que no se han guardado; es decir, se cerrará el archivo y se descartarán los cambios. Si *bPromptToSaveChanges* tiene el valor `true` y el documento contiene cambios que no se han guardado, el usuario recibirá un mensaje con el cuadro de diálogo estándar con Sí y No. El valor predeterminado es `true`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; cierra el documento especificado.

Ejemplo

El ejemplo siguiente ilustra dos formas de cerrar un documento.

```
// 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);
```

Véase también

`fl.closeAll()`

fl.compilerErrors

Disponibilidad
Flash CS3 Professional

Uso
`fl.compilerErrors`

Descripción
Propiedad de sólo lectura; un objeto compilerErrors que representa el panel de errores. Para obtener información sobre la utilización de esta propiedad, consulte Objeto compilerErrors.

fl.componentsPanel

Disponibilidad
Flash MX 2004

Uso
`fl.componentsPanel`

Descripción
Propiedad de sólo lectura; un Objeto componentsPanel que representa el panel Componentes.

Ejemplo
El ejemplo siguiente almacena un objeto componentsPanel en la variable `comPanel`:

```
var comPanel = fl.componentsPanel;
```

fl.configDirectory

Disponibilidad
Flash MX 2004

Uso
`fl.configDirectory`

Descripción
Propiedad de sólo lectura; una cadena que especifica la ruta completa de la carpeta Configuration del usuario local con el formato específico de la plataforma. Para especificar esta ruta como una URI file:///, que no es específica de la plataforma, utilice `fl.configURI`.

Ejemplo
El ejemplo siguiente muestra el directorio Configuration en el panel Salida:

```
fl.trace("My local configuration directory is " + fl.configDirectory);
```

fl.configURI

Disponibilidad
Flash MX 2004

Uso
`fl.configURI`

Descripción
Propiedad de sólo lectura; una cadena que especifica la ruta completa de la carpeta Configuration del usuario local como URI file:///. Véase también `fl.configDirectory`.

Ejemplo
El ejemplo siguiente ejecuta un script especificado. El uso de `fl.configURI` permite especificar la ubicación del script sin necesidad de saber en qué plataforma se está ejecutando el script.

```
// 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

Disponibilidad
Flash 8

Uso
`fl.contactSensitiveSelection`

Descripción
Un valor booleano que especifica si el modo de selección Por contacto está activado (`true`) o no (`false`).

Ejemplo
El siguiente ejemplo muestra cómo desactivar el modo de selección Por contacto antes de realizar una selección y cómo restablecerlo a su valor original después de realizar la selección.

```
var contact = fl.contactSensitiveSelection;
fl.contactSensitiveSelection = false;
// Insert selection code here.
fl.contactSensitiveSelection = contact;
```

fl.createDocument()

Disponibilidad
Flash MX 2004

Uso

```
fl.createDocument([docType])
```

Parámetros

`docType` Una cadena que especifica el tipo de documento que se va a crear. Los valores aceptables son `"timeline"`, `"presentation"` y `"application"`. El valor predeterminado es `"timeline"`, que tiene el mismo efecto que seleccionar Archivo > Nuevo > Archivo de Flash (ActionScript 3.0). Este parámetro es opcional.

Valor devuelto

El objeto Document para el documento recién creado, si el método es correcto. Si se produce un error, el valor es `undefined`.

Descripción

Método; abre un documento nuevo y lo selecciona. Los valores de tamaño, resolución y color son los predeterminados.

Ejemplo

El ejemplo siguiente crea distintos tipos de documentos:

```
// 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

Disponibilidad

Flash MX 2004

Uso

```
fl.createNewDocList
```

Descripción

Propiedad de sólo lectura; un conjunto de cadenas que representa los distintos tipos de documentos que se pueden crear.

Ejemplo

El ejemplo siguiente muestra los tipos de documentos que se pueden crear en el panel Salida:

```
fl.trace("Number of choices " + fl.createNewDocList.length);
for (i = 0; i < fl.createNewDocList.length; i++)
    fl.trace("choice: " + fl.createNewDocList[i]);
```

fl.createNewDocListType

Disponibilidad
Flash MX 2004

Uso
`fl.createNewDocListType`

Descripción
Propiedad de sólo lectura; un conjunto de cadenas que representa las extensiones de archivo de los tipos de documentos que se pueden crear. Las entradas del conjunto corresponden directamente (por índice) a las entradas del conjunto `fl.createNewDocList`.

Ejemplo
El ejemplo siguiente muestra las extensiones de los tipos de documentos que se pueden crear en el panel Salida:

```
fl.trace("Number of types " + fl.createNewDocListType.length);
for (i = 0; i < fl.createNewDocListType.length; i++) fl.trace("type: " +
fl.createNewDocListType[i]);
```

fl.createNewTemplateList

Disponibilidad
Flash MX 2004

Uso
`fl.createNewTemplateList`

Descripción
Propiedad de sólo lectura; un conjunto de cadenas que representa los distintos tipos de plantillas que se pueden crear.

Ejemplo
El ejemplo siguiente muestra los tipos de plantillas que se pueden crear en el panel Salida:

```
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

Disponibilidad
Flash MX 2004

Uso
`fl.documents`

Descripción

Propiedad de sólo lectura; un conjunto de objetos Document (consulte Objeto Document) que representa los documentos (archivos FLA) que están abiertos para editar.

Ejemplo

El ejemplo siguiente almacena un conjunto de documentos abiertos en la variable `docs`:

```
var docs = fl.documents;
```

El ejemplo siguiente muestra los nombres de los documentos abiertos en el panel Salida:

```
for (doc in fl.documents) {
    fl.trace(fl.documents[doc].name);
}
```

fl.downloadLatestVersion()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.downloadLatestVersion(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta local del archivo que se va a descargar del servidor Version Cue. Sólo se pueden descargar los archivos que no están abiertos. Si el archivo especificado por *fileURI* está abierto, este método no tiene ningún efecto.

Valor devuelto

Un valor booleano de `true` si se descarga correctamente el archivo, y de `false` en caso contrario.

Descripción

Método; descarga del servidor Version Cue la última versión de un archivo que no esté actualmente abierto. Para descargar la última versión de un archivo abierto, utilice `document.synchronizeWithHeadVersion()`.

Ejemplo

El ejemplo siguiente descarga el archivo myFile.fla del servidor Version Cue:

```
fl.downloadLatestVersion("file:///C|/MyFiles/Version Cue/docs/myFile.fla");
```

Véase también

`document.synchronizeWithHeadVersion()`, `fl.revertDocumentToLastVersion()`, `fl.saveAVersionOfDocument()`, `fl.synchronizeDocumentWithHeadVersion()`

fl.drawingLayer

Disponibilidad
Flash MX 2004

Uso
`fl.drawingLayer`

Descripción
Propiedad de sólo lectura; el Objeto drawingLayer que una herramienta ampliable deberá utilizar cuando el usuario desee dibujar temporalmente mientras arrastra el ratón (por ejemplo, para crear un recuadro de delimitación).

Ejemplo
Consulte `drawingLayer.setColor()`.

fl.externalLibraryPath

Disponibilidad
Flash CS4 Professional

Uso
`fl.externalLibraryPath`

Descripción
Propiedad; una cadena que contiene una lista de elementos de la ruta de biblioteca externa global de ActionScript 3.0, donde se especifica la ubicación de los archivos SWC utilizados como bibliotecas compartidas de tiempo de ejecución. Los elementos de la cadena se delimitan con punto y coma. En la herramienta de edición, los elementos se especifican seleccionando Edición > Preferencias > ActionScript > Configuración de ActionScript 3.0.

Ejemplo
El ejemplo siguiente añade la carpeta /SWC_runtime a la ruta de biblioteca externa global de ActionScript 3.0.

```
fl.trace(fl.externalLibraryPath);
fl.externalLibraryPath = "/SWC_runtime;" + fl.externalLibraryPath;
fl.trace(fl.externalLibraryPath);
```

Véase también
`fl.flexSDKPath`, `fl.libraryPath`, `fl.sourcePath`, `document.externalLibraryPath`

fl.fileExists()

Disponibilidad
Flash MX 2004

Uso

```
fl.fileExists(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que contiene la ruta al archivo.

Valor devuelto

Un valor booleano: `true` si el archivo se encuentra en el disco; `false` en caso contrario.

Descripción

Método; comprueba si el archivo ya existe en el disco.

Ejemplo

El ejemplo siguiente muestra `true` o `false` en el panel Salida para cada archivo especificado, en función de si existe o no el archivo.

```
alert(fl.fileExists("file:///C|/example.fla"));
alert(fl.fileExists("file:///C|/example.jsfl"));
alert(fl.fileExists(""));
```

fl.findDocumentDOM()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.findDocumentDOM(id)
```

Parámetros

`id` Un entero que representa un identificador exclusivo de un documento.

Valor devuelto

Un objeto Document o `null` si no existe ningún documento con el *id* especificado.

Descripción

Método; permite buscar un archivo específico mediante su identificador exclusivo (en lugar de su valor de índice, por ejemplo). Utilice este método junto con `document.id`.

Ejemplo

El ejemplo siguiente ilustra la lectura de un ID de documento y el uso de éste para buscar el documento:

```
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;
```

Véase también

`fl.findDocumentIndex()`

fl.findDocumentIndex()

Disponibilidad

Flash MX 2004

Uso

```
fl.findDocumentIndex(name)
```

Parámetros

`name` El nombre del documento para el que desea buscar el índice. El documento debe estar abierto.

Valor devuelto

Un conjunto de enteros que representan la posición del *nombre* del documento en el conjunto `fl.documents`.

Descripción

Método; devuelve un conjunto de enteros que representan la posición del *nombre* del documento en el conjunto `fl.documents`. Se pueden abrir varios documentos con el mismo nombre (si están ubicados en distintas carpetas).

Ejemplo

El ejemplo siguiente muestra información sobre la posición del índice de todos los archivos abiertos denominados test.fla en el panel Salida:

```
var filename = "test.fla"
var docIndex = fl.findDocumentIndex(filename);
for (var index in docIndex)
    fl.trace(filename + " is open at index " + docIndex[index]);
```

Véase también

`fl.documents`, `fl.findDocumentDOM()`

fl.findObjectInDocByName()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.findObjectInDocByName(instanceName, document)
```

Parámetros

`instanceName` Una cadena que especifica el nombre de instancia de un elemento en el documento especificado.

`document` El Objeto Document en el que se realizará la búsqueda del elemento especificado.

Valor devuelto

Un conjunto de objetos genéricos. Utilice la propiedad `.obj` de cada elemento del conjunto para obtener el objeto. El objeto tiene las siguientes propiedades: `keyframe`, `layer`, `timeline` y `parent`. Estas propiedades se pueden utilizar para acceder a la jerarquía del objeto. Para más información sobre estas propiedades y la forma de acceder a ellas, consulte `fl.findObjectInDocByType()`.

También se puede acceder a los métodos y propiedades de los valores `layer` y `timeline`; son equivalentes al Objeto Layer y al Objeto Timeline, respectivamente.

Descripción

Método; expone los elementos en un documento con nombres de instancias que coinciden con texto especificado.

Nota: *en algunos casos, este método sólo funciona cuando se ejecuta como comando desde un archivo FLA y no cuando está viendo o editando actualmente el archivo JSFL.*

Ejemplo

En el siguiente ejemplo se buscan en el documento actual los elementos denominados `"instance01"`.

```
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");
}
```

Véase también

`fl.findObjectInDocByType()`

fl.findObjectInDocByType()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.findObjectInDocByType(elementType, document)
```

Parámetros

`elementType` Una cadena que representa el tipo de elemento de búsqueda. Para saber qué valores se aceptan, consulte `element.elementType`.

`document` El Objeto Document en el que se realizará la búsqueda del elemento especificado.

Valor devuelto

Un conjunto de objetos genéricos. Utilice la propiedad `.obj` de cada elemento del conjunto para obtener el objeto Element. Cada objeto tiene las siguientes propiedades: `keyframe`, `layer`, `timeline` y `parent`. Estas propiedades se pueden utilizar para acceder a la jerarquía del objeto.

También se puede acceder a los métodos y propiedades de los valores `layer` y `timeline`; son equivalentes al Objeto Layer y al Objeto Timeline, respectivamente.

El segundo y tercer ejemplo de la sección Ejemplos muestra la forma de tener acceso a estas propiedades.

Descripción

Método; busca en un documento elementos de un tipo especificado.

Nota: *en algunos casos, este método sólo funciona cuando se ejecuta como comando desde un archivo FLA y no cuando está viendo o editando actualmente el archivo JSFL.*

Ejemplo

El ejemplo siguiente busca campos de texto en el documento actual y luego cambia su contenido:

```
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");
}
```

El ejemplo siguiente muestra cómo acceder a las propiedades especiales del objeto devuelto por este método:

```
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.
}
```

El ejemplo siguiente muestra la forma de realizar una copia de seguridad de DOM para buscar el nombre de una capa en la que se encontró el campo de texto, mediante el objeto 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");
}
```

Véase también

`fl.findObjectInDocByName()`

fl.flexSDKPath

Disponibilidad

Flash CS4 Professional

Uso

`fl.flexSDKPath`

Descripción

Propiedad; una cadena que especifica la ruta de la carpeta Flex SDK, que contiene, entre otras, las carpetas bin, frameworks y lib. En la herramienta de edición, los elementos se especifican seleccionando Edición > Preferencias > ActionScript > Configuración de ActionScript 3.0.

Ejemplo

El código siguiente muestra la ruta del SDK de Flex en el panel Salida:

```
fl.trace(fl.flexSDKPath);
```

Véase también

`fl.externalLibraryPath`, `fl.libraryPath`, `fl.sourcePath`

fl.getAppMemoryInfo()

Disponibilidad

Flash 8 (sólo Windows).

Uso

`fl.getAppMemoryInfo(memType)`

Parámetros

`memType` Un entero que especifica el área de uso de la memoria que se va a consultar. Para ver una lista de los valores aceptables, consulte la descripción siguiente.

Valor devuelto

Un entero que representa el número de bytes que se están utilizando en un área especificada de memoria Flash.exe.

Descripción

Método (sólo Windows); devuelve un entero que representa el número de bytes que se están utilizando en un área especificada de memoria Flash.exe. Utilice la tabla siguiente para determinar qué valor desea pasar como *memType*:

memType	Datos de recursos
0	`PAGEFAULTCOUNT`
1	`PEAKWORKINGSETSIZE`
2	`WORKINGSETSIZE`
3	`QUOTAPEAKPAGEDPOOLUSAGE`
4	`QUOTAPAGEDPOOLUSAGE`
5	`QUOTAPEAKNONPAGEDPOOLUSAGE`
6	`QUOTANONPAGEDPOOLUSAGE`
7	`PAGEFILEUSAGE`
8	`PEAKPAGEFILEUSAGE`

Ejemplo

El ejemplo siguiente muestra el consumo actual de memoria de trabajo:

```
var memsize = fl.getAppMemoryInfo(2);
fl.trace("Flash current memory consumption is " + memsize + " bytes or " + memsize/1024 + "
KB");
```

fl.getDocumentDOM()

Disponibilidad

Flash MX 2004

Uso

`fl.getDocumentDOM()`

Parámetros

Ninguno.

Valor devuelto

Un objeto Document o `null` si no hay documentos abiertos.

Descripción

Método; recupera el DOM (Objeto Document) del documento activo actualmente (archivo FLA). Si hay uno o varios documentos abiertos, pero no hay uno seleccionado (por ejemplo, hay un archivo JSFL seleccionado), recupera el DOM del documento que más recientemente ha estado activo.

Ejemplo

El ejemplo siguiente muestra el nombre del documento activo actualmente o que más recientemente ha estado activo en el panel Salida:

```
var currentDoc = fl.getDocumentDOM();
fl.trace(currentDoc.name);
```

fl.isFontInstalled()

Disponibilidad

Flash CS4 Professional

Uso

```
fl.isFontInstalled(fontName)
```

Parámetros

`fontName` Cadena que especifica el nombre de la fuente del dispositivo.

Valor devuelto

Un valor booleano de `true` si la fuente especificada está instalada, y de `false` en caso contrario.

Descripción

Método; determina si está instalada una fuente especificada.

Ejemplo

El siguiente código muestra “true” en el panel Salida si la fuente Times está instalada.

```
fl.trace(fl.isFontInstalled("Times"));
```

fl.libraryPath

Disponibilidad

Flash CS4 Professional

Uso

```
fl.libraryPath
```

Descripción

Propiedad; una cadena que contiene una lista de elementos de la ruta de biblioteca global de ActionScript 3.0, donde se especifica la ubicación de los archivos SWC o de las carpetas que contienen los archivos SWC. Los elementos de la cadena se delimitan con punto y coma. En la herramienta de edición, los elementos se especifican seleccionando Edición > Preferencias > ActionScript > Configuración de ActionScript 3.0.

Ejemplo

El ejemplo siguiente añade la carpeta /SWC a la ruta de biblioteca global de ActionScript 3.0:

```
fl.trace(fl.libraryPath);
fl.libraryPath = "/SWC;" + fl.libraryPath;
fl.trace(fl.libraryPath);
```

Véase también

`fl.externalLibraryPath`, `fl.flexSDKPath`, `fl.sourcePath`, `document.libraryPath`

fl.mapPlayerURL()

Disponibilidad

Flash MX 2004

Uso

```
fl.mapPlayerURL(URI [, returnMBCS])
```

Parámetros

`URI` Una cadena que contiene la URL Unicode de escape que se va a asignar.

`returnMBCS` Un valor booleano que debe definir como `true` si desea que se devuelva una ruta MBCS de escape. En caso contrario, el método devuelve UTF-8. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Una cadena que es la URL convertida.

Descripción

Método; asigna una URL Unicode de escape a una URL UTF-8 o MBCS. Utilice este método cuando la cadena se va a utilizar en ActionScript para acceder a un recurso externo. Deberá emplear este método si necesita gestionar caracteres de múltiples bytes.

Ejemplo

El ejemplo siguiente convierte una URL a UTF-8 para que el reproductor pueda cargarla:

```
var url = MMExecute( "fl.mapPlayerURL(" + myURL + ", false);" );
mc.loadMovie( url);
```

fl.Math

Disponibilidad
Flash MX 2004

Uso
`fl.Math`

Descripción
Propiedad de sólo lectura; el Objeto Math proporciona métodos para operaciones de matrices y puntos.

Ejemplo
En el siguiente ejemplo se muestra la matriz de transformación del objeto seleccionado y su contrario:

```
// 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

Disponibilidad
Flash MX 2004

Uso
`fl.mruRecentFileList`

Descripción
Propiedad de sólo lectura; un conjunto de los nombres completos de archivo de la lista Usados recientemente (MRU) que gestiona la herramienta de edición de Flash.

Ejemplo
El ejemplo siguiente muestra el número de archivos abiertos recientemente y el nombre de cada uno en el panel Salida:

```
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

Disponibilidad
Flash MX 2004

Uso

```
fl.mruRecentFileListType
```

Descripción

Propiedad de sólo lectura; un conjunto de los tipos de archivo de la lista MRU que gestiona la herramienta de edición de Flash. Este conjunto corresponde al conjunto de la propiedad `fl.mruRecentFileList`.

Ejemplo

El ejemplo siguiente muestra el número de archivos abiertos recientemente y el tipo de cada uno en el panel Salida:

```
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

Disponibilidad

Flash 8

Uso

```
fl.objectDrawingMode
```

Descripción

Propiedad; un valor booleano que especifica si el modo de dibujo de objeto está activado (`true`) o lo está el modo de dibujo de fusión (`false`).

Ejemplo

El siguiente ejemplo cambia el estado del modo de dibujo de objeto:

```
var toggleMode = fl.objectDrawingMode;
if (toggleMode) {
    fl.objectDrawingMode = false;
} else {
    fl.objectDrawingMode = true;
}
```

fl.openDocument()

Disponibilidad

Flash MX 2004

Uso

```
fl.openDocument(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el nombre del archivo que se va a abrir.

Valor devuelto
El Objeto Document para el documento recién abierto, si el método es correcto. Si no se encuentra el archivo o no es un archivo FLA válido, se presenta un error y se cancela el script.

Descripción
Método; abre un documento de Flash (archivo FLA) para editarlo en una nueva ventana de documento de Flash y lo selecciona. Para un usuario, el efecto equivale a seleccionar Archivo > Abrir y, a continuación, seleccionar un archivo. Si el archivo especificado ya está abierto, la ventana que contiene el documento se coloca en primer plano. La ventana que contiene el archivo especificado se convierte en el documento seleccionado actualmente.

Ejemplo
El ejemplo siguiente abre un archivo llamado Document.fla que se almacena en el directorio raíz de la unidad C. El código almacena un objeto Document que representa ese documento en la variable `doc` y establece el documento como el seleccionado actualmente. Es decir, hasta que cambie el enfoque, `fl.getDocumentDOM()` hace referencia a este documento.

```
var doc = fl.openDocument("file:///c|/Document.fla");
```

fl.openScript()

Disponibilidad
Flash MX 2004

Uso
```
fl.openScript(fileURI)
```

Parámetros
`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta del archivo JSFL, AS, ASC, XML, TXT o de otro tipo que se debe cargar en el editor de texto de Flash.

Valor devuelto
Ninguno.

Descripción
Método; abre un archivo de script (JSFL, AS, ASC) o de otro tipo (XML, TXT) en el editor de texto de Flash.

Ejemplo
El ejemplo siguiente abre un archivo llamado my_test.jsfl que se almacena en el directorio /temp de la unidad C:

```
fl.openScript("file:///c|/temp/my_test.jsfl");
```

fl.outputPanel

Disponibilidad
Flash MX 2004

Uso

```
fl.outputPanel
```

Descripción

Propiedad de sólo lectura; referencia al Objeto outputPanel.

Ejemplo

Consulte Objeto outputPanel.

fl.packagePaths

Disponibilidad

Flash CS3 Professional

Uso

```
fl.packagePaths
```

Descripción

Propiedad; una cadena que corresponde a la configuración de rutas de clases global en el cuadro de diálogo Configuración de ActionScript 2.0. Las rutas de clases de una cadena se delimitan con punto y coma (;). Para ver o cambiar la configuración de rutas de clases de ActionScript 3.0, utilice `fl.as3PackagePaths`.

Ejemplo

El ejemplo siguiente ilustra el cambio de la configuración de rutas de clases de 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
```

Véase también

`fl.resetPackagePaths()`

fl.presetPanel

Disponibilidad

Flash CS4 Professional

Uso

```
fl.presetPanel
```

Descripción

Sólo lectura; un Objeto presetPanel.

fl.quit()

Disponibilidad
Flash MX 2004

Uso
`fl.quit([bPromptIfNeeded])`

Parámetros
`bPromptIfNeeded` Un valor booleano que es `true` (predeterminado) si desea que el usuario reciba un mensaje para guardar los documentos modificados. Defina este parámetro como `false` si no desea que el usuario reciba un mensaje para guardar los documentos modificados. En este último caso se descartarán las modificaciones realizadas en los documentos abiertos y la aplicación se cerrará inmediatamente. Aunque resulta útil para el procesamiento por lotes, utilice este método con precaución. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; sale de Flash y pregunta al usuario si desea guardar los documentos modificados.

Ejemplo
El ejemplo siguiente ilustra el cierre con la pregunta sobre si desea guardar los documentos modificados y sin ella:

```
// 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()

Disponibilidad
Flash MX 2004

Uso
`fl.reloadEffects()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; vuelve a cargar todos los descriptores de efectos definidos en la carpeta Configuration Effects del usuario. Permite cambiar rápidamente los script durante el desarrollo y proporciona un mecanismo para mejorar los efectos sin volver a iniciar la aplicación. Este método funciona mejor si se utiliza en un comando situado en la carpeta Commands.

Ejemplo
El ejemplo siguiente es un script de una línea que puede situar en la carpeta Commands. Cuando necesite volver a cargar los efectos, acceda al menú Comandos y ejecute el script.

```
fl.reloadEffects();
```

fl.reloadTools()

Disponibilidad
Flash MX 2004

Uso
```
fl.reloadTools()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; recrea el panel Herramientas a partir del archivo toolconfig.xml. Este método sólo se utiliza para crear herramientas ampliables. Utilice este método cuando necesite volver a cargar el panel Herramientas, por ejemplo, después de modificar el archivo JSFL que define una herramienta que ya se encuentra en el panel.

Ejemplo
El ejemplo siguiente es un script de una línea que puede situar en la carpeta Commands. Cuando necesite volver a cargar el panel Herramientas, ejecute el script desde el menú Comandos.

```
fl.reloadTools();
```

fl.removeEventListener()

Disponibilidad
Flash CS3 Professional

Uso
```
fl.removeEventListener(eventType)
```

Parámetros

`eventType` Una cadena que especifica el tipo de evento que va a eliminar de esta función callback. Los valores aceptables son `"documentNew"`, `"documentOpened"`, `"documentClosed"`, `"mouseMove"`, `"documentChanged"`, `"layerChanged"` y `"frameChanged"`.

Valor devuelto

Un valor booleano de `true` si se eliminó correctamente el detector de eventos, `false` si la función nunca se añadió a la lista con el método `fl.addEventListener()`.

Descripción

Anula el registro de una función que se registró con `fl.addEventListener()`.

Ejemplo

El ejemplo siguiente elimina el detector de eventos asociado al evento `documentClosed`:

```
fl.removeEventListener("documentClosed");
```

Véase también

`fl.addEventListener()`

fl.resetAS3PackagePaths()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.resetAS3PackagePaths()
```

Parámetros

Ninguno.

Descripción

Método; restablece la configuración de ruta de clases global en el cuadro de diálogo Configuración de ActionScript 3.0 con el valor predeterminado. Para restablecer la ruta de clases global de ActionScript 2.0, utilice `fl.resetPackagePaths()`.

Ejemplo

El ejemplo siguiente muestra el restablecimiento de la configuración de ruta de clases de ActionScript 3.0 con el valor predeterminado.

```
fl.resetAS3PackagePaths();
```

Véase también

`fl.as3PackagePaths`

fl.resetPackagePaths()

Disponibilidad
Flash CS3 Professional

Uso
`fl.resetPackagePaths()`

Parámetros
Ninguno.

Descripción
Método; restablece la configuración de ruta de clases global en el cuadro de diálogo Configuración de ActionScript 2.0 con el valor predeterminado. Para restablecer la ruta de clases global de ActionScript 3.0, utilice `fl.resetAS3PackagePaths()`.

Ejemplo
El ejemplo siguiente muestra el restablecimiento de la configuración de ruta de clases de ActionScript 2.0 con el valor predeterminado.

```
fl.resetPackagePaths();
```

Véase también
`fl.packagePaths`

fl.revertDocument()

Disponibilidad
Flash MX 2004

Uso
`fl.revertDocument(documentObject)`

Parámetros
`documentObject` Un Objeto Document. Si *documentObject* hace referencia al documento activo, es posible que no se descarten los cambios en la ventana Documento hasta que termine de ejecutarse el script que llama a este método.

Valor devuelto
Un valor booleano: `true` si la operación Descartar cambios se realiza correctamente; `false` en caso contrario.

Descripción
Método; devuelve el documento FLA especificado a la última versión guardada. A diferencia de la opción del menú Archivo > Descartar cambios, este método no muestra una ventana de advertencia para preguntar al usuario si confirma la operación. Véase también `document.revert()` y `document.canRevert()`.

Para devolver el documento a la versión del servidor Version Cue, utilice `fl.revertDocumentToLastVersion()`.

Ejemplo

El ejemplo siguiente devuelve el documento FLA actual a la última versión guardada; se perderán los cambios realizados desde la última vez que lo guardó.

```
fl.revertDocument(fl.getDocumentDOM());
```

fl.revertDocumentToLastVersion()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.revertDocumentToLastVersion(documentObject)
```

Parámetros

`documentObject` Un Objeto Document.

Valor devuelto

Un valor booleano de `true` si se descartan los cambios del documento, y de `false` en caso contrario.

Descripción

Método; si se pueden descartar los cambios del archivo, devuelve el documento especificado a la versión guardada en el servidor Version Cue y registra cualquier error en el panel Salida.

Para cambiar un documento a la última versión guardada localmente, utilice `fl.revertDocument()`.

Ejemplo

El ejemplo siguiente devuelve el documento actual a la versión almacenada en el servidor Version Cue:

```
fl.revertDocumentToLastVersion(fl.getDocumentDOM());
```

Véase también

`document.revertToLastVersion()`, `fl.downloadLatestVersion()`, `fl.saveAVersionOfDocument()`, `fl.synchronizeDocumentWithHeadVersion()`

fl.runScript()

Disponibilidad

Flash MX 2004

Uso

```
fl.runScript(fileURI [, funcName [, arg1, arg2, ...]])
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el nombre del archivo de script que se va a ejecutar.

funcName Una cadena que identifica una función que se va a ejecutar en el archivo JSFL especificada en *fileURI*. Este parámetro es opcional.

arg Un parámetro opcional que especifica uno o más argumentos que se van a transferir a *funcname*.

Valor devuelto

El resultado de la función como cadena, si se especifica *funcName*; de lo contrario, no devuelve nada.

Descripción

Método; ejecuta un archivo JavaScript. Si se especifica una función como uno de los argumentos, ejecuta la función y el código del script que no está dentro de la función. El resto del código del script se ejecuta antes de que se ejecute la función.

Ejemplo

Supongamos que hay un archivo de script llamado testScript.jsfl en el directorio raíz de la unidad C y su contenido es el siguiente:

```
function testFunct(num, minNum) {
    fl.trace("in testFunct: 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 emite el comando siguiente,

```
fl.runScript("file:///C|/testScript.jsfl", "testFunct", 10, 1);
```

aparecerá la información siguiente en el panel Salida:

```
in for loop i=0
in for loop i=1
end of for loop
in testFunct: 1st arg: 10 2nd arg: 1
```

También puede llamar a testScript.jsfl sin ejecutar una función, de este modo:

```
fl.runScript("file:///C|/testScript.jsfl");
```

Esto produce lo siguiente en el panel Salida:

```
in for loop i=0
in for loop i=1
end of for loop
```

fl.saveAll()

Disponibilidad

Flash MX 2004

Uso

```
fl.saveAll()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; guarda todos los documentos abiertos.

Si un archivo no se ha guardado nunca o no se ha modificado desde la última vez que se guardó, no se guardará. Para permitir que se guarde un archivo no guardado o no modificado, utilice `fl.saveDocumentAs()`.

Ejemplo
El ejemplo siguiente guarda todos los documentos abiertos guardados previamente y que se han modificado desde la última vez que se guardaron:

```
fl.saveAll();
```

Véase también
`document.save()`, `document.saveAndCompact()`, `fl.saveDocument()`, `fl.saveDocumentAs()`

fl.saveAVersionOfDocument()

Disponibilidad
Flash CS3 Professional

Uso
`fl.saveAVersionOfDocument(document)`

Parámetros
`document` Un Objeto Document.

Valor devuelto
Un valor booleano de `true` si se guarda correctamente una versión del documento en el servidor Version Cue; `false` en caso contrario.

Descripción
Método; si el archivo se puede guardar en el servidor Version Cue, muestra un cuadro de diálogo para permitir que el usuario introduzca comentarios de la versión, guarda en el servidor una versión del documento especificado y registra cualquier error en el panel Salida.

Ejemplo
El ejemplo siguiente guarda el documento actual en el servidor Version Cue:

```
f1.saveAVersionOfDocument(f1.getDocumentDOM());
```

Véase también
`document.saveAVersion()`

fl.saveDocument()

Disponibilidad
Flash MX 2004

Uso
```
fl.saveDocument(document [, fileURI])
```

Parámetros
`document` Un Objeto Document que especifica el documento que se va a guardar. Si *document* es `null`, se guardará el documento activo.

`fileURI` Cadena que se expresa en forma de URI file:/// y que especifica el nombre del documento guardado. Si el parámetro *fileURI* es `null` o se omite, el documento se guardará con su nombre actual. Este parámetro es opcional.

Valor devuelto
Un valor booleano: `true` si la operación de guardado se realiza correctamente; `false` en caso contrario.

Si el archivo no se ha guardado nunca o no se ha modificado desde la última vez que se guardó, no se guardará y el valor devuelto será `false`. Para permitir que se guarde un archivo no guardado o no modificado, utilice `fl.saveDocumentAs()`.

Descripción
Método; guarda el documento especificado como documento FLA.

Ejemplo
El ejemplo siguiente guarda el documento actual y dos documentos especificados:

```
// 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"));
```

Véase también
`document.save()`, `document.saveAndCompact()`, `fl.saveAll()`, `fl.saveDocumentAs()`

fl.saveDocumentAs()

Disponibilidad
Flash MX 2004

Uso
```
fl.saveDocumentAs(document)
```

Parámetros
`document` Un Objeto Document que especifica el documento que se va a guardar. Si *document* es `null`, se guardará el documento activo.

Valor devuelto

Un valor booleano: `true` si la operación Guardar como se realiza correctamente; `false` en caso contrario.

Descripción

Método; muestra el cuadro de diálogo Guardar como para el documento especificado.

Ejemplo

El ejemplo siguiente pregunta al usuario si desea guardar el documento especificado y, a continuación, muestra un mensaje de alerta en el que se indica si el documento estaba guardado:

```
alert(fl.saveDocumentAs(fl.documents[1]));
```

Véase también

`document.save()`, `document.saveAndCompact()`, `fl.saveAll()`, `fl.saveDocument()`

fl.scriptURI

Disponibilidad

Flash CS3 Professional

Uso

```
fl.scriptURI
```

Descripción

Propiedad de sólo lectura; una cadena que representa la ruta del script JSFL que se ejecuta actualmente, expresada como URI file:///. Si el script se llamó desde `fl.runScript()`, esta propiedad representa la ruta del script principal inmediato. Es decir, no recorre varias llamadas a `fl.runScript()` para buscar la ruta del script original que realiza la llamada.

Ejemplo

El ejemplo siguiente muestra la ruta del script JSFI que se ejecuta actualmente en el panel Salida:

```
fl.trace(fl.scriptURI);
```

Véase también

`fl.runScript()`

fl.selectElement()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.selectElement(elementObject, editMode)
```

Parámetros

elementObject El Objeto Element que desea seleccionar.

editMode Un valor booleano que especifica si desea editar el elemento (`true`) o solamente seleccionarlo (`false`).

Valor devuelto

Un valor booleano de `true` si el elemento se selecciona correctamente, y de `false` en caso contrario.

Descripción

Método; activa la selección o edición de un elemento. Normalmente, este método se utilizará en objetos devueltos por `fl.findObjectInDocByName()` o `fl.findObjectInDocByType()`.

Ejemplo

El ejemplo siguiente selecciona un elemento denominado "`second text field`" si se encuentra uno en el documento:

```
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");
    }
```

Véase también

`fl.findObjectInDocByName()`, `fl.findObjectInDocByType()`

fl.selectTool()

Disponibilidad

Flash CS3 Professional

Uso

```
fl.selectTool(toolName)
```

Parámetros

toolName Una cadena que especifica el nombre de la herramienta que se va a seleccionar. Consulte “Descripción” abajo para más información sobre los valores aceptables para este parámetro.

Descripción

Método; selecciona la herramienta especificada en el panel Herramientas. Los valores predeterminados aceptables para *toolName* son `"arrow"`, `"bezierSelect"`, `"freeXform"`, `"fillXform"`, `"lasso"`, `"pen"`, `"penplus"`, `"penminus"`, `"penmodify"`, `"text"`, `"line"`, `"rect"`, `"oval"`, `"rectPrimitive"`, `"ovalPrimitive"`, `"polystar"`, `"pencil"`, `"brush"`, `"inkBottle"`, `"bucket"`, `"eyeDropper"`, `"eraser"`, `"hand"` y `"magnifier"`.

Si el usuario crea herramientas personalizadas, los nombres de éstas se pueden pasar también como parámetro *toolName*. La lista de los nombres de herramientas se ubica en el siguiente archivo:

- Windows Vista:

  unidad de arranque\Users*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS3*idioma*\Configuration\Tools\toolConfig.xml

- Windows XP:

  unidad de arranque\Documents and Settings*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS3*idioma*\Configuration\Tools\toolConfig.xml

- Mac OS X:

  Macintosh HD/Usuarios/*nombre de usuario*/Librería/Application Support/Adobe/Flash CS3/*idioma*/Configuration/Tools/toolConfig.xml

Ejemplo

El ejemplo siguiente selecciona la herramienta Pluma:

```
fl.selectTool("pen");
```

Véase también

Objeto Tools, Objeto ToolObj

fl.setActiveWindow()

Disponibilidad

Flash MX 2004

Uso

```
fl.setActiveWindow(document [, bActivateFrame])
```

Parámetros

`document` Un Objeto Document que especifica el documento que se va a seleccionar en la ventana activa.

`bActivateFrame` Un parámetro opcional que omiten Flash y Fireworks y sólo está presente por compatibilidad con Dreamweaver.

Valor devuelto

Ninguno.

Descripción
Método; establece el documento especificado como ventana activa. Este método también se admite en Dreamweaver y Fireworks. Si el documento tiene múltiples vistas (creadas con Ventana > Duplicar ventana), se seleccionará la vista activa más reciente.

Ejemplo
El ejemplo siguiente muestra dos formas de activar un documento especificado:

```
fl.setActiveWindow(fl.documents[0]);

var theIndex = fl.findDocumentIndex("myFile.fla");
fl.setActiveWindow(fl.documents[theIndex]);
```

fl.showIdleMessage()

Disponibilidad
Flash 8

Uso
`fl.showIdleMessage(show)`

Parámetros
`show` Un valor booleano que especifica si se activa o desactiva la advertencia sobre un script con una ejecución demasiado larga.

Valor devuelto
Ninguno.

Descripción
Método; permite desactivar la advertencia sobre un script con una ejecución demasiado larga (transfiere `false` para *show*). Puede utilizarlo cuando procese operaciones por lotes que tarden demasiado tiempo en terminar. Para volver a activar la alerta, emita de nuevo el comando, transfiriendo esta vez `true` para *show*.

Ejemplo
El ejemplo siguiente ilustra cómo se desactiva y vuelve a activar la advertencia sobre un script con una ejecución demasiado larga:

```
fl.showIdleMessage(false);
var result = timeConsumingFunction();
fl.showIdleMessage(true); ;
var result = timeConsumingFunction();
```

fl.sourcePath

Disponibilidad
Flash CS4 Professional

Uso

```
fl.sourcePath
```

Descripción

Propiedad; una cadena que contiene una lista de elementos de la ruta de origen global de ActionScript 3.0, donde se especifica la ubicación de los archivos de clases de ActionScript. Los elementos de la cadena se delimitan con punto y coma. En la herramienta de edición, los elementos se especifican seleccionando Edición > Preferencias > ActionScript > Configuración de ActionScript 3.0.

Ejemplo

El ejemplo siguiente añade la carpeta /Classes a la ruta de origen global de ActionScript 3.0:

```
fl.trace(fl.sourcePath);
fl.sourcePath = "/Classes;" + fl.sourcePath;
fl.trace(fl.sourcePath);
```

Véase también

`fl.flexSDKPath`, `fl.externalLibraryPath`, `fl.libraryPath`, `document.sourcePath`

fl.swfPanels

Disponibilidad

Flash CS4 Professional

Uso

```
fl.swfPanels
```

Descripción

Propiedad de sólo lectura; un conjunto de objetos swfPanel registrados (consulte Objeto swfPanel). Se considera que un objeto swfPanel está registrado si se ha abierto al menos una vez.

Una posición del panel en el conjunto representa el orden en que se abrió. Si el primer panel abierto se llama TraceBitmap y el segundo panel que se abre se llama AnotherFunction, `fl.swfPanels[0]` es el objeto swfPanel TraceBitmap, `fl.swfPanels[1]` es el otro objeto swfPanel AnotherFunction, etc.

Ejemplo

El siguiente código muestra el nombre y la ruta de cualquier panel SWF de ventana registrado en el panel Salida:

```
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.synchronizeDocumentWithHeadVersion()

Disponibilidad
Flash CS3 Professional

Uso
`fl.synchronizeDocumentWithHeadVersion(documentObject)`

Parámetros
`documentObject` Un Objeto Document.

Valor devuelto
Un valor booleano de `true` si el archivo especificado se sincronizó correctamente con el servidor Version Cue, `false` en caso contrario.

Descripción
Método; sincroniza el documento especificado con la versión más actual en el servidor Version Cue y registra cualquier error en el panel Salida. Este método es idéntico a `document.synchronizeWithHeadVersion()`.

Ejemplo
El ejemplo siguiente sincroniza el documento actual con el servidor Version Cue:

```
fl.synchronizeWithHeadVersion(fl.getDocumentDOM());
```

Véase también
`fl.downloadLatestVersion()`, `fl.revertDocumentToLastVersion()`, `fl.saveAVersionOfDocument()`

fl.tools

Disponibilidad
Flash MX 2004

Uso
`fl.tools`

Descripción
Propiedad de sólo lectura; un conjunto de objetos Tools (consulte Objeto Tools). Esta propiedad sólo se utiliza para crear herramientas ampliables.

fl.trace()

Disponibilidad
Flash MX 2004

Uso

```
fl.trace(message)
```

Parámetros

`message` Una cadena que aparece en el panel Salida.

Valor devuelto

Ninguno.

Descripción

Método; envía una cadena de texto al panel Salida, terminada con una línea nueva y muestra el panel Salida si aún no está visible. Este método es idéntico a `outputPanel.trace()` y funciona del mismo modo que la declaración `trace()` en ActionScript.

Para enviar una línea en blanco, utilice `fl.trace("")` o `fl.trace("\n")`. Para utilizar estos comandos en línea, convierta `\n` en parte de la cadena *message*.

Ejemplo

El ejemplo siguiente muestra varias líneas de texto en el panel Salida:

```
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

Disponibilidad

Flash MX 2004

Uso

```
fl.version
```

Descripción

Propiedad de sólo lectura; la versión de cadena larga de la herramienta de edición de Flash, incluida la plataforma.

Ejemplo

El ejemplo siguiente muestra la versión de la herramienta de edición de Flash en el panel Salida:

```
alert(fl.version); // For example, WIN 10,0,0,540
```

fl.xmlui

Disponibilidad

Flash MX 2004

Uso

```
fl.xmlui
```

Descripción

Propiedad de sólo lectura; un Objeto XMLUI. Esta propiedad permite obtener y establecer propiedades XMLUI en un cuadro de diálogo XMLUI y permite aceptar o cancelar el cuadro de diálogo de forma programada.

Ejemplo

Consulte Objeto XMLUI.

Capítulo 18: Objeto FLfile

Disponibilidad
Flash MX 2004 7.2.

Descripción
El objeto FLfile permite escribir extensiones de Flash que pueden acceder, modificar y eliminar archivos y carpetas en el sistema local de archivos. La API FLfile se proporciona en forma de una extensión en la API JavaScript. Esta extensión se denomina *shared library* (biblioteca compartida) y se ubica en la siguiente carpeta:

- Windows Vista:

  unidad de arranque\Users*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS3*idioma*\Configuration\External Libraries\FLfile.dll

- Windows XP:

  unidad de arranque\Documents and Settings*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS3*idioma*\Configuration\External Libraries\FLfile.dll

- Mac OS X:

  Macintosh HD/Usuarios/*nombre de usuario* /Librería/Application Support/Adobe/Flash CS3/*idioma* /Configuration/External Libraries/FLfile.dll

Nota: *no confunda las bibliotecas compartidas que contienen símbolos en los documentos de Flash con las bibliotecas compartidas de la API JavaScript. Se trata de dos cosas diferentes.*

Los métodos FLfile trabajan con archivos o carpetas (directorios) en disco. Por lo tanto, cada método adopta uno o varios parámetros para especificar la ubicación de un archivo o carpeta. La ubicación del archivo o carpeta se expresa como una cadena de forma muy similar a una dirección URL de un sitio Web. Se denomina un URI (Identificador uniforme de recursos) de archivo y se le aplica formato tal y como se muestra a continuación (incluyendo las comillas):

```
"file:///drive|/folder 1/folder 2/.../filename"
```

Por ejemplo, si desea crear una carpeta en la unidad C denominada config y situarla en la carpeta Program Files/MyApp, utilice el siguiente comando:

```
FLfile.createFolder("file:///C|/Program Files/MyApp/config");
```

Si desea ubicar un archivo llamado config.ini en esa carpeta, utilice el siguiente comando:

```
FLfile.write("file:///C|/Program Files/MyApp/config/config.ini", "");
```

Para crear una carpeta en Macintosh, puede utilizar el siguiente comando:

```
FLfile.createFolder("file:///Macintosh/MyApp/config");
```

Resumen de métodos
Se pueden emplear los siguientes métodos con el objeto FLfile:

Método	Descripción
`FLfile.copy()`	Copia un archivo.
`FLfile.createFolder()`	Crea una o varias carpetas.
`FLfile.exists()`	Determina la existencia de un archivo o carpeta.
`FLfile.getAttributes()`	Averigua si un archivo se puede modificar, es de sólo lectura, está oculto o visible o está en una carpeta del sistema.
`FLfile.getCreationDate()`	Especifica cuántos segundos han transcurrido entre el 1 de enero de 1970 y la hora de creación del archivo o carpeta.
`FLfile.getCreationDateObj()`	Obtiene la fecha de creación de un archivo o carpeta.
`FLfile.getModificationDate()`	Especifica cuántos segundos han transcurrido entre el 1 de enero de 1970 y la hora de modificación del archivo o carpeta.
`FLfile.getModificationDateObj()`	Obtiene la fecha en la que se modificó un archivo o carpeta por última vez.
`FLfile.getSize()`	Obtiene el tamaño de un archivo.
`FLfile.listFolder()`	Enumera el contenido de una carpeta.
`FLfile.platformPathToURI()`	Convierte un nombre de archivo en un formato específico de plataforma en una URI file:///.
`FLfile.read()`	Lee el contenido de un archivo.
`FLfile.remove()`	Elimina un archivo o carpeta.
`FLfile.setAttributes()`	Hace que un archivo o carpeta se pueda modificar, que sea de sólo lectura o se encuentre oculto o visible.
`FLfile.uriToPlatformPath()`	Convierte un nombre de archivo expresado como URI file:/// en un formato concreto de la plataforma.
`FLfile.write()`	Crea, se añade o escribe en un archivo.

FLfile.copy()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.copy(fileURI, copyURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo que desea copiar.

`copyURI` Una cadena, expresada como URI file:///, que especifica la ubicación y el nombre del archivo copiado.

Valor devuelto

Un valor booleano de `true` si es correcto, y de `false` en caso contrario.

Descripción

Método; copia un archivo de una ubicación a otra. Este método devuelve `false` si ya existe *copyURI*.

Ejemplo

El ejemplo siguiente realiza una copia de seguridad de un archivo de configuración llamado config.ini y lo sitúa en la misma carpeta en la que se encuentra, pero con otro nombre:

```
var originalFileURI="file:///C|/Program Files/MyApp/config.ini";
var newFileURI="file:///C|/Program Files/MyApp/config_backup.ini";
FLfile.copy(originalFileURI, newFileURI);
```

Si lo prefiere, puede realizar la misma tarea con un solo comando:

```
FLfile.copy("file:///C|:/Program Files/MyApp/config.ini", file:///C|/Program
Files/MyApp/config_backup.ini");
```

FLfile.createFolder()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.createFolder(folderURI)
```

Parámetros

`folderURI` Una URI de carpeta que especifica la carpeta que desea crear.

Valor devuelto

Un valor booleano de `true` si es correcto y de `false` si *folderURI* ya existe.

Descripción

Método; crea una o varias carpetas en la ubicación especificada.

Puede crear varias carpetas de una vez. Por ejemplo, el comando siguiente crea las carpetas MyData y TempData si aún no existen:

```
FLfile.createFolder("file:///c|/MyData/TempData")
```

Ejemplo

El ejemplo siguiente crea una carpeta y una subcarpeta en la carpeta de configuración (fl.configURI):

```
fl.trace(FLfile.createFolder(fl.configURI+"folder01/subfolder01"));
```

El ejemplo siguiente intenta crear una carpeta llamada tempFolder en el nivel de la raíz de la unidad C y muestra un cuadro de alerta que indica si la operación ha tenido éxito:

```
var folderURI = "file:///c|/tempFolder";
if (FLfile.createFolder(folderURI)) {
    alert("Created " + folderURI);
}
else {
    alert(folderURI + " already exists");
}
```

Véase también

`FLfile.remove()`, `FLfile.write()`

FLfile.exists()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.exists(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo que desea verificar.

Valor devuelto

Un valor booleano de `true` si es correcto, y de `false` en caso contrario.

Descripción

Método; determina si existe un archivo especificado. Si especifica una carpeta y un nombre de archivo, la carpeta ya debe existir. Para crear carpetas, consulte `FLfile.createFolder()`.

Ejemplos

El ejemplo siguiente comprueba si hay un archivo llamado mydata.txt en la carpeta temporal y muestra un cuadro de alerta en el que se indica si el archivo existe.

```
var fileURI = "file:///c|/temp/mydata.txt";
if (FLfile.exists(fileURI)) {
    alert( fileURI + " exists.");
}
else {
    alert( fileURI + " does not exist.");
}
```

El ejemplo siguiente comprueba si existe un archivo de configuración necesario en la carpeta MyApplication. Si el archivo no existe, lo crea.

```
var configFile = "file:///C|/MyApplication/config.ini";
if (!FLfile.exists(configFile)) {
    FLfile.write(configFile,"");

}
```

Véase también

`FLfile.write()`

FLfile.getAttributes()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.getAttributes(fileOrFolderURI)
```

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo o la carpeta cuyos atributos desea recuperar.

Valor devuelto

Una cadena que representa los atributos del archivo o carpeta que se ha especificado.

los resultados son impredecibles si el archivo o la carpeta no existen. Deberá utilizar `FLfile.exists()` antes de emplear este método.

Descripción

Método; devuelve una cadena que representa los atributos del archivo o la carpeta que se ha especificado, o bien una cadena vacía si el archivo no tiene atributos específicos (es decir, no es de sólo lectura, no está oculto, etc.). Debe utilizar siempre `FLfile.exists()` para probar la existencia de un archivo o carpeta antes de emplear este método.

Los caracteres de la cadena representan los atributos siguientes:

- `R` — *fileOrFolderURI* es de sólo lectura
- `D` — *fileOrFolderURI* es una carpeta (directorio)
- `H` — *fileOrFolderURI* está oculto (sólo Windows)
- `S` — *fileOrFolderURI* es un archivo o carpeta del sistema (sólo Windows)
- `A` — *fileOrFolderURI* está listo para archivar (sólo Windows)

Por ejemplo, si *fileOrFolderURI* es una carpeta oculta, la cadena devuelta es`"DH"`.

Ejemplo

El ejemplo siguiente obtiene los atributos del archivo mydata.txt y muestra un cuadro de alerta si el archivo es de sólo lectura.

```
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!");
    }
}
```

Véase también

`FLfile.setAttributes()`

FLfile.getCreationDate()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.getCreationDate(fileOrFolderURI)
```

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo o carpeta cuya fecha y hora de creación desea recuperar como cadena hexadecimal.

Valor devuelto

Una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la hora de creación del archivo o carpeta, o bien, `"00000000"` si no existe el archivo o carpeta.

Descripción

Especifica cuántos segundos han transcurrido entre el 1 de enero de 1970 y la hora de creación del archivo o carpeta. Este método se utiliza principalmente para comparar las fechas de creación o modificación de los archivos o carpetas.

Ejemplo

El ejemplo siguiente determina si un archivo se ha modificado desde que se creó:

```
// Make sure the specified file exists
var fileURI = "file:///C|/MyApplication/MyApp.fla";
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.");
}
```

Véase también

`FLfile.getCreationDateObj()`, `FLfile.getModificationDate()`

FLfile.getCreationDateObj()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.getCreationDateObj(fileOrFolderURI)
```

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo o carpeta cuya fecha y hora de creación desea recuperar como objeto Date de JavaScript.

Valor devuelto

Un objeto Date de JavaScript que representa la fecha y hora de creación de un archivo o carpeta especificada. Si el archivo no existe, el objeto contiene información que indica que el archivo o carpeta se creó a medianoche GMT el 31 de diciembre de 1969.

Descripción

Método; devuelve un objeto Date de JavaScript que representa la fecha y hora de creación de un archivo o carpeta especificada.

Ejemplo

En el ejemplo siguiente se muestra (con formato legible para el usuario) la fecha de creación de un archivo en el panel Salida:

```
// Make sure the specified file exists.
var file1Date = FLfile.getCreationDateObj("file:///c|/temp/file1.txt");
fl.trace(file1Date);
```

Véase también

`FLfile.getCreationDate()`, `FLfile.getModificationDateObj()`

FLfile.getModificationDate()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.getModificationDate(fileOrFolderURI)
```

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo cuya fecha y hora de modificación desea recuperar como cadena hexadecimal.

Valor devuelto

Una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la hora de última modificación del archivo o carpeta, o bien, `"00000000"` si no existe el archivo o carpeta.

Descripción

Especifica cuántos segundos han transcurrido entre el 1 de enero de 1970 y la hora de modificación del archivo o carpeta. Este método se utiliza principalmente para comparar las fechas de creación o modificación de los archivos o carpetas.

Ejemplo

En el ejemplo siguiente se comparan las fechas de modificación de dos archivos y se determina cuál de los dos se modificó más recientemente:

```
// 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") ;
}
```

Véase también

`FLfile.getCreationDate()`, `FLfile.getModificationDateObj()`

FLfile.getModificationDateObj()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.getModificationDateObj(fileOrFolderURI)
```

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo o carpeta cuya fecha y hora de modificación desea recuperar como objeto Date de JavaScript.

Valor devuelto

Un objeto Date de JavaScript que representa la fecha y hora de la última modificación de un archivo o carpeta especificada. Si el archivo o carpeta no existe, el objeto contiene información que indica que el archivo o carpeta se creó a medianoche GMT el 31 de diciembre de 1969.

Descripción

Método; devuelve un objeto Date de JavaScript que representa la fecha y hora de la última modificación de un archivo o carpeta especificada.

Ejemplo

En el ejemplo siguiente se muestra (con formato legible para el usuario) la fecha de la última modificación de un archivo en el panel Salida:

```
// Make sure the specified file exists.
var file1Date = FLfile.getModificationDateObj("file:///c|/temp/file1.txt");
trace(file1Date);
```

Véase también

`FLfile.getCreationDateObj()`, `FLfile.getModificationDate()`

FLfile.getSize()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.getSize(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo cuyo tamaño desea recuperar.

Valor devuelto

Un entero que representa el tamaño del archivo especificado, en bytes, o bien 0 si no existe el archivo.

Descripción

Método; devuelve un entero que representa el tamaño del archivo especificado, en bytes, o bien 0 si no existe el archivo. Si el valor devuelto es 0, puede utilizar `FLfile.exists()` para determinar si el archivo es de byte cero o no existe.

Este método devuelve los valores de tamaño correctos solamente de los archivos que tienen un tamaño inferior o igual a 2 GB.

Ejemplo

El ejemplo siguiente almacena el tamaño del archivo mydata.txt en la variable `fileSize`:

```
var URL = "file:///c|/temp/mydata.txt";
var fileSize = FLfile.getSize(URL);
```

FLfile.listFolder()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.listFolder(folderURI [, filesOrDirectories])
```

Parámetros

`folderURI` Una cadena, expresada como URI file:///, que especifica la carpeta cuyo contenido desea recuperar. Puede incluir una máscara de comodín como parte de *folderURI*. Los comodines válidos son: * (equivale a uno o varios caracteres) y ? (equivale a un solo carácter).

`filesOrDirectories` Una cadena opcional que especifica si sólo se devuelven nombres de archivo o sólo nombres de carpeta (directorio). Si se omite, se devuelven nombres de archivo y de carpeta. Los valores aceptables son `"files"` y `"directories"`.

Valor devuelto

Un conjunto de cadenas que representa el contenido de la carpeta. Si la carpeta no existe o si ningún archivo o carpeta coinciden con los criterios especificados, devuelve un conjunto vacío.

Descripción

Método; devuelve un conjunto de cadenas que representa el contenido de la carpeta.

Ejemplos

El siguiente ejemplo devuelve tres conjuntos. El primero representa todos los archivos de la carpeta C:\temp, el segundo todas las carpetas de la carpeta C:\temp y el tercero los archivos y carpetas de la carpeta 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);
```

El ejemplo siguiente devuelve un conjunto de todos los archivos de texto (.txt) de la carpeta temp y muestra la lista en un cuadro de alerta:

```
var folderURI = "file:///c|/temp";
var fileMask = "*.txt";
var list = FLfile.listFolder(folderURI + "/" + fileMask, "files");
if (list) {
    alert(folderURI + " contains: " + list.join(" "));
}
```

El ejemplo siguiente utiliza una máscara de archivo en la *folderURI* especificada para devolver los nombres de todos los archivos ejecutables de la carpeta de aplicación de Windows:

```
var executables = FLfile.listFolder("file:///C|/WINDOWS/*.exe","files");
alert(executables.join("\n"));
```

FLfile.platformPathToURI()

Disponibilidad

Flash CS4 Professional

Uso

```
FLfile.platformPathToURI(fileName)
```

Parámetros

`fileName` Cadena; expresada en un formato concreto de la plataforma. Especifica el nombre de archivo que se quiere convertir.

Valor devuelto

Una cadena expresada como URI file:///.

Descripción

Convierte un nombre de archivo en un formato concreto de la plataforma en una URI file:///.

Ejemplo

En el siguiente ejemplo se convierte un nombre de archivo de un formato específico de plataforma a una URI file:///, que se transmite a `outputPanel.save()`:

```
var myFilename = "C:\\outputPanel.txt";
var myURI=FLfile.platformPathToURI(myFilename);
fl.outputPanel.save(myURI);
```

Véase también

`FLfile.uriToPlatformPath()`

FLfile.read()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.read()
```

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo o la carpeta cuyos atributos desea recuperar.

Valor devuelto

El contenido del archivo especificado como una cadena o `null` si se produce un error de lectura.

Descripción

Método; devuelve el contenido del archivo especificado como una cadena o `null` si se produce un error de lectura.

Ejemplos

El ejemplo siguiente lee el archivo mydata.txt y, si es correcto, muestra un cuadro de alerta con el contenido del archivo.

```
var fileURI = "file:///c|/temp/mydata.txt";
var str = FLfile.read( fileURI);
if (str) {
    alert( fileURL + " contains: " + str);
}
```

El ejemplo siguiente lee el código de ActionScript de un archivo de clase y lo almacena en la variable `code`:

```
var classFileURI = "file:///C|/MyApplication/TextCarousel.as";
var code = FLfile.read(classFileURI);
```

FLfile.remove()

Disponibilidad

Flash MX 2004 7.2.

Uso

`FLfile.remove(fileOrFolderURI)`

Parámetros

`fileOrFolderURI` Una cadena, expresada como URI file:///, que especifica el archivo o la carpeta que desea eliminar (quitar).

Valor devuelto

Un valor booleano de `true` si es correcto, y de `false` en caso contrario.

Descripción

Método; elimina el archivo o la carpeta especificada. Si la carpeta contiene archivos, también se eliminarán esos archivos. No se pueden eliminar los archivos con el atributo R (de sólo lectura).

Ejemplos

El ejemplo siguiente advierte al usuario si existe un archivo y, a continuación, lo elimina si el usuario así lo desea:

```
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");
}
```

El ejemplo siguiente elimina un archivo de configuración creado por una aplicación:

```
if(FLfile.remove("file:///C|/MyApplication/config.ini")) {
    alert("Configuration file deleted");
}
```

El ejemplo siguiente elimina la carpeta Configuration y su contenido:

```
FLfile.remove("file:///C|/MyApplication/Configuration/");
```

Véase también

`FLfile.createFolder()`, `FLfile.getAttributes()`

FLfile.setAttributes()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.setAttributes(fileURI, strAttrs)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo cuyos atributos desea configurar.

`strAttrs` Una cadena que especifica valores para los atributos que desea definir. Para obtener valores aceptables de *strAttrs*, consulte la sección "Descripción".

Valor devuelto

Un valor booleano de `true` si es correcto.

Nota: *Los resultados son impredecibles si el archivo o la carpeta no existen. Deberá utilizar* `FLfile.exists()` *antes de emplear este método.*

Descripción

Método; especifica atributos de nivel del sistema para el archivo especificado.

Los valores siguientes son válidos para *strAttrs*:

- `N` — Sin atributos específicos (no es de sólo lectura, no está oculto, etc.)
- `A` — Listo para archivar (sólo Windows)
- `R` — De sólo lectura (en Macintosh, de sólo lectura significa “bloqueado”)
- `W` — De escritura (anula `R`)
- `H` — Oculto (sólo Windows)
- `V` — Visible (anula `H`, sólo Windows)

Si incluye `R` y `W` en *strAttrs*, se ignorará `R` y el archivo se definirá como de escritura. De forma similar, si transfiere`H` y `V`, se ignorará `H` y el archivo se definirá como visible.

Si desea asegurarse de que el atributo de archivo no está definido, utilice este comando con el parámetro `N` antes de establecer los atributos. Es decir, no hay contrapartida directa para `A` que desactive el atributo de archivo.

Ejemplos

El ejemplo siguiente define el archivo mydata.text como de sólo lectura y oculto. No tiene efecto en el atributo de archivo.

```
var URI = "file:///c|/temp/mydata.txt";
if (FLfile.exists(URI)) {
    FLfile.setAttributes(URI, "RH");
}
```

El ejemplo siguiente define el archivo mydata.text como de sólo lectura y oculto. También garantiza que no se define el atributo de archivo.

```
var URI = "file:///c|/temp/mydata.txt";

if (FLfile.exists(URI)) {
    FLfile.setAttributes(URI, "N");
    FLfile.setAttributes(URI, "RH");
}
```

Véase también

`FLfile.getAttributes()`

FLfile.uriToPlatformPath()

Disponibilidad

Flash CS4 Professional

Uso

`FLfile.uriToPlatformPath(fileURI)`

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el nombre de archivo que desea convertir.

Valor devuelto

Una cadena que representa una ruta específica de la plataforma.

Descripción

Convierte un nombre de archivo expresado como URI file:/// en un formato concreto de la plataforma.

Ejemplo

En el siguiente ejemplo se convierte una URI file:/// a un formato específico de plataforma:

```
var dir =(fl.configDirectory);
var URI = FLfile.platformPathToURI(dir);
fl.trace(URI == fl.configURI); // displays "true"
```

Véase también

`FLfile.platformPathToURI()`

FLfile.write()

Disponibilidad

Flash MX 2004 7.2.

Uso

```
FLfile.write(fileURI, textToWrite, [ , strAppendMode])
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo en el que desea escribir.

`textToWrite` Una cadena que representa el texto que desea situar en el archivo.

`strAppendMode` Una cadena opcional con el valor `"append"` que indica que quiere adjuntar *textToWrite* al archivo existente. Si se omite, *fileURI* se sobrescribe con *textToWrite*.

Valor devuelto

Un valor booleano de `true` si es correcto, y de `false` en caso contrario.

Descripción

Método; escribe la cadena especificada en el archivo especificado (como UTF-8). Si no existe el archivo especificado se crea. Sin embargo, debe existir la carpeta en la que está ubicando el archivo antes de utilizar este método. Para crear carpetas, utilice `FLfile.createFolder()`.

Ejemplo

El ejemplo siguiente intenta escribir la cadena "xxx" en el archivo mydata.txt y muestra un mensaje de alerta si la escritura se realiza correctamente. A continuación, intenta añadir la cadena "aaa" al archivo y muestra un segundo mensaje de alerta si la escritura se realiza correctamente. Después de ejecutar este script, el archivo mydata.txt sólo contendrá el texto "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);
}
```

Véase también

`FLfile.createFolder()`, `FLfile.exists()`

Capítulo 19: Objeto folderItem

Herencia Objeto Item > Objeto folderItem

Disponibilidad
Flash MX 2004

Descripción
El objeto folderItem es una subclase del objeto Item. folderItem no tiene métodos ni propiedades exclusivos. Consulte Objeto Item.

Capítulo 20: Objeto fontItem

Herencia Objeto Item > objeto fontItem

Disponibilidad
Flash MX 2004

Descripción
El objeto fontItem es una subclase del objeto Item (consulte Objeto Item).

Resumen de propiedades
Además de las propiedades del objeto Item, el objeto fontItem dispone de las siguientes:

Propiedad	Descripción
`fontItem.bitmap`	Especifica si el elemento Font es de mapa de bits.
`fontItem.bold`	Especifica si el elemento Font está en negrita.
`fontItem.embedVariantGlyphs`	Especifica si los pictogramas variantes deben aparecer en la fuente al publicar un archivo SWF.
`fontItem.font`	Nombre de la fuente de dispositivo asociada al objeto Font.
`fontItem.isDefineFont4Symbol`	Especifica el formato de la fuente que aparece al publicar un archivo SWF.
`fontItem.italic`	Especifica si el elemento Font está en cursiva.
`fontItem.size`	Tamaño del elemento Font, en puntos.

fontItem.bitmap

Disponibilidad
Flash CS4 Professional

Uso
`fontItem.bitmap`

Descripción
Propiedad; un valor booleano que especifica si el elemento Font es de mapa de bits (`true`) o no (`false`).

Ejemplo
Si damos por hecho que el primer elemento de la biblioteca es un elemento Font, el siguiente código muestra `true` en el panel Salida si es de mapa de bits, y `false` si no lo es:

```
var theItem = fl.getDocumentDOM().library.items[0];
fl.trace("bitmap: "+ theItem.bitmap);
```

fontItem.bold

Disponibilidad
Flash CS4 Professional

Uso
`fontItem.bold`

Descripción
Propiedad; un valor booleano que especifica si el elemento Font está en negrita (`true`) o no (`false`).

Ejemplo
Si damos por hecho que el primer elemento de la biblioteca es un elemento Font, el siguiente código muestra `true` en el panel Salida si está en negrita, y `false` si no lo es (y lo pone en negrita).

```
var theItem = fl.getDocumentDOM().library.items[0];
fl.outputPanel.clear();
fl.trace("bold: "+ theItem.bold);
theItem.bold=true;
fl.trace("bold: "+ theItem.bold);
```

fontItem.embedVariantGlyphs

Disponibilidad
Flash CS4 Professional

Uso
`fontItem.embedVariantGlyphs`

Descripción
Propiedad; valor booleano que especifica si los pictogramas variantes deben aparecer en la fuente al publicar un archivo SWF (`true`) o no (`false`). Si este valor se establece en `true`, aumenta el tamaño del archivo SWF. El valor predeterminado es `false`.

Algunos idiomas sustituyen dinámicamente los pictogramas de caracteres conforme se está escribiendo (por ejemplo, tailandés, árabe, hebreo y griego). Si está diseñando o introduciendo texto en este tipo de idiomas, establezca esta propiedad en `true`.

Ejemplos
Los símbolos de fuente que son compatibles con las API de flash.text aparecen en la biblioteca y el usuario pueden administrarlas directamente. No obstante, los símbolos de fuente que son compatibles con las API de flash.text.engine (FTE) no aparecen en la biblioteca, por lo que se deben administrar manualmente. La siguiente función añade una nueva fuente a la biblioteca que se puede utilizar con las 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 siguiente función muestra todos los símbolos de fuente en el panel Salida.

```
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);
            }
        }
    }
}
```

Consulte también

`fontItem.isDefineFont4Symbol`, `text.embedVariantGlyphs`

fontItem.font

Disponibilidad

Flash CS4 Professional

Uso

```
fontItem.font
```

Descripción

Propiedad; una cadena que especifica el nombre de la fuente de dispositivo asociada al objeto Font. Si introduce una cadena que no se corresponda con ninguna fuente de dispositivo instalada, aparecerá un mensaje de error. Para determinar si una fuente existe en el sistema, utilice `fl.isFontInstalled()`.

Nota: *al establecer este valor, el valor de la propiedad resultante puede no coincidir con la cadena introducida. Consulte el siguiente ejemplo.*

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento Font, el siguiente código muestra el nombre de la fuente de dispositivo asociada actualmente al elemento Font y, después, la cambia a 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

Disponibilidad

Flash CS4 Professional

Uso

```
fontItem.isDefineFont4Symbol
```

Descripción

Propiedad; valor booleano que especifica el formato de la fuente que aparece al publicar un archivo SWF. Si este valor es `true`, Flash devuelve una fuente que se puede utilizar con las API de flash.text.engine (FTE). Si este valor es `false`, la fuente se puede usar con las API de flash.text, incluyendo campos de texto. El valor predeterminado es `false`.

Ejemplo

Consulte `fontItem.embedVariantGlyphs`.

fontItem.italic

Disponibilidad
Flash CS4 Professional

Uso
`fontItem.italic`

Descripción
Propiedad; un valor booleano que especifica si el elemento Font está en cursiva (`true`) o no (`false`).

Ejemplo
Si damos por hecho que el primer elemento de la biblioteca es un elemento Font, el siguiente código muestra `true` en el panel Salida si está en cursiva, y `false` si no lo es (y lo pone en cursiva).

```
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

Disponibilidad
Flash CS4 Professional

Uso
`fontItem.size`

Descripción
Propiedad; un entero que representa el tamaño del elemento Font, en puntos.

Ejemplo
Si damos por hecho que el primer elemento de la biblioteca es un elemento Font, el siguiente código muestra el tamaño en puntos del elemento en el panel Salida y, después, lo establece en 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);
```

Capítulo 21: Objeto Frame

Disponibilidad

Flash MX 2004

Descripción

El objeto Frame representa fotogramas en la capa.

Resumen de métodos

Pueden emplearse los métodos siguientes con el objeto Frame:

Método	Descripción
`frame.getCustomEase()`	Devuelve un conjunto de objetos JavaScript, cada uno de los cuales tiene una propiedad *x* e *y*.
`frame.setCustomEase()`	Especifica una curva cúbica de Bézier que se va a utilizar como curva de suavizado personalizada.

Resumen de propiedades

Pueden emplearse las propiedades siguientes con el objeto Frame:

Propiedad	Descripción
`frame.actionScript`	Una cadena que representa código ActionScript.
`frame.duration`	De sólo lectura; un entero que representa el número de fotogramas en una secuencia de fotogramas.
`frame.elements`	Sólo lectura; conjunto de objetos Element (consulte Objeto Element).
`frame.hasCustomEase`	Un valor booleano que especifica si el fotograma obtiene la información de suavizado de la curva de suavizado personalizada.
`frame.labelType`	Una cadena que especifica el tipo de nombre de fotograma.
`frame.motionTweenOrientToPath`	Un valor booleano que especifica si el elemento interpolado gira el elemento cuando se mueve a lo largo de un trazado para mantener su ángulo con respecto a cada punto del trazado.
`frame.motionTweenRotate`	Una cadena que especifica cómo gira el elemento interpolado.
`frame.motionTweenRotateTimes`	Un entero que especifica el número de veces que el elemento interpolado gira entre el fotograma clave inicial y el siguiente fotograma clave.
`frame.motionTweenScale`	Valor booleano que especifica si el elemento interpolado se escala hasta el tamaño del objeto del siguiente fotograma clave, incrementando su tamaño con cada fotograma de la interpolación (`true`), o si no se escala (`false`).
`frame.motionTweenSnap`	Valor booleano que especifica si el elemento interpolado se ajusta automáticamente al punto más próximo de la capa de guía de movimiento asociada a la capa de este fotograma (`true`) o si no se ajusta (`false`).
`frame.motionTweenSync`	Valor booleano que si se define como `true`, sincroniza la animación del objeto interpolado con la línea de tiempo principal.
`frame.name`	Una cadena que especifica el nombre del fotograma.

Propiedad	Descripción
`frame.shapeTweenBlend`	Una cadena que especifica cómo se mezcla una interpolación de forma entre la forma del fotograma clave al comienzo de la interpolación y la forma del siguiente fotograma clave.
`frame.soundEffect`	Una cadena que especifica efectos para un sonido que está asociado directamente a un fotograma (`frame.soundLibraryItem`).
`frame.soundLibraryItem`	Un elemento de biblioteca (consulte Objeto SoundItem) empleado para crear un sonido.
`frame.soundLoop`	Un valor entero que especifica el número de veces que se reproduce un sonido asociado directamente a un fotograma (`frame.soundLibraryItem`).
`frame.soundLoopMode`	Una cadena que especifica si un sonido asociado directamente a un fotograma (`frame.soundLibraryItem`) debe reproducirse un número específico de veces o realizar un bucle indefinido.
`frame.soundName`	Una cadena que especifica el nombre de un sonido que está asociado directamente a un fotograma (`frame.soundLibraryItem`), tal como está almacenado en la biblioteca.
`frame.soundSync`	Una cadena que especifica el comportamiento de sincronización de un sonido que está asociado directamente a un fotograma (`frame.soundLibraryItem`).
`frame.startFrame`	De sólo lectura; el índice del primer fotograma de una secuencia.
`frame.tweenEasing`	Un entero que especifica la cantidad de suavizado que debe aplicarse al objeto interpolado.
`frame.tweenType`	Una cadena que especifica el tipo de interpolado.
`frame.useSingleEaseCurve`	Un valor booleano que especifica si se utilizará una curva de suavizado personalizada para la información de suavizado de todas las propiedades.

frame.actionScript

Disponibilidad

Flash MX 2004

Uso

```
frame.actionScript
```

Descripción

Propiedad; una cadena que representa código ActionScript. Para insertar una nueva línea de carácter, utilice `"\n"`.

Ejemplo

El ejemplo siguiente asigna `stop()` a la acción primer fotograma de la capa superior:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].actionScript = 'stop();';
```

frame.duration

Disponibilidad
Flash MX 2004

Uso
`frame.duration`

Descripción
Propiedad de sólo lectura; un entero que representa el número de fotogramas en una secuencia de fotogramas.

Ejemplo
El ejemplo siguiente almacena el número de fotogramas de una secuencia que comienza en el primer fotograma de la capa superior en la variable `frameSpan`:

```
var frameSpan = fl.getDocumentDOM().getTimeline().layers[0].frames[0].duration;
```

frame.elements

Disponibilidad
Flash MX 2004

Uso
`frame.elements`

Descripción
Propiedad de sólo lectura; un conjunto de objetos Element (consulte Objeto Element). Los elementos aparecen en el orden en que están almacenados en el archivo FLA. Si hay múltiples formas en el escenario y todas están desagrupadas, Flash las tratará como un elemento. Si todas las formas están agrupadas, de modo que haya múltiples grupos en el escenario, Flash las considerará como elementos independientes. Es decir, Flash trata las formas no agrupadas y en bruto como un solo elemento, independientemente del número de formas independientes haya en el escenario. Si un fotograma se compone de tres formas no agrupadas y en bruto, por ejemplo, `elements.length` de ese fotograma devolverá un valor de 1. Para solucionar este problema, seleccione cada forma individualmente y agrúpela.

Ejemplo
El ejemplo siguiente almacena un conjunto de elementos actuales en la capa superior del primer fotograma en la variable `myElements`:

```
var myElements = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements;
```

frame.getCustomEase()

Disponibilidad
Flash 8

Uso

```
Frame.getCustomEase([property])
```

Parámetros

`property` Una cadena opcional que especifica la propiedad para la que desea devolver el valor de suavizado personalizado. Los valores aceptables son `"all"`, `"position"`, `"rotation"`, `"scale"`, `"color"` y `"filters"`. El valor predeterminado es `"all"`.

Valor devuelto

Devuelve un conjunto de objetos JavaScript, cada uno de los cuales tiene una propiedad *x* e *y*.

Descripción

Método; devuelve un conjunto de objetos que representa los puntos de control de la curva cúbica de Bézier que define la curva de suavizado.

Ejemplo

El ejemplo siguiente devuelve el valor de suavizado personalizado de la propiedad `position` para el primer fotograma de la capa superior:

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
var easeArray = theFrame.getCustomEase("position");
```

Véase también

`frame.hasCustomEase`, `frame.setCustomEase()`, `frame.useSingleEaseCurve`

frame.hasCustomEase

Disponibilidad

Flash 8

Uso

```
frame.hasCustomEase
```

Descripción

Propiedad; un valor booleano. Si es `true`, el fotograma obtiene su información de suavizado de la curva de suavizado personalizada. Si es `false`, el fotograma obtiene la información de suavizado del valor de suavizado.

Ejemplo

El ejemplo siguiente especifica que el primer fotograma de la capa superior debe obtener su información de suavizado del valor de suavizado en lugar de tomarlo de la curva de suavizado personalizado:

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
theFrame.hasCustomEase = false;
```

Véase también

`frame.getCustomEase()`, `frame.setCustomEase()`, `frame.useSingleEaseCurve`

frame.labelType

Disponibilidad
Flash MX 2004

Uso
`frame.labelType`

Descripción
Propiedad; una cadena que especifica el tipo de nombre de fotograma. Los valores aceptables son `"none"`, `"name"`, `"comment"` y `"anchor"`. Si define una etiqueta como `"none"`, se borrará la propiedad `frame.name`.

Ejemplo
El ejemplo siguiente define el nombre del primer fotograma de la capa superior como `"First Frame"` y, a continuación, define su etiqueta como `"comment"`:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].name = 'First Frame';
fl.getDocumentDOM().getTimeline().layers[0].frames[0].labelType = 'comment';
```

frame.motionTweenOrientToPath

Disponibilidad
Flash MX 2004

Uso
`frame.motionTweenOrientToPath`

Descripción
Propiedad; un valor booleano que especifica si el elemento interpolado gira el elemento cuando se mueve a lo largo de un trazado para mantener su ángulo con respecto a cada punto del trazado (`true`) o si no rota (`false`).

Si desea especificar un valor para esta propiedad, deberá definir `frame.motionTweenRotate` como `"none"`.

frame.motionTweenRotate

Disponibilidad
Flash MX 2004

Uso
`frame.motionTweenRotate`

Descripción
Propiedad; una cadena que especifica cómo gira el elemento interpolado. Los valores aceptables son `"none"`, `"auto"`, `"clockwise"` y `"counter-clockwise"`. Un valor de `"auto"` significa que el objeto girará en la dirección que requiera menos movimiento para coincidir con la rotación del objeto en el fotograma clave siguiente.

Si desea especificar un valor para `frame.motionTweenOrientToPath`, defina esta propiedad como `"none"`.

Ejemplo
Consulte `frame.motionTweenRotateTimes`.

frame.motionTweenRotateTimes

Disponibilidad
Flash MX 2004

Uso
```
frame.motionTweenRotateTimes
```

Descripción
Propiedad; un entero que especifica el número de veces que el elemento interpolado gira entre el fotograma clave inicial y el siguiente fotograma clave.

Ejemplo
El ejemplo siguiente gira el elemento de este fotograma en el sentido contrario al de las agujas del reloj tres veces cuando llega al siguiente fotograma clave:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenRotate = "counter-
clockwise";
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenRotateTimes = 3;
```

frame.motionTweenScale

Disponibilidad
Flash MX 2004

Uso
```
frame.motionTweenScale
```

Descripción
Propiedad; un valor booleano que especifica si el elemento interpolado se escala hasta el tamaño del objeto del siguiente fotograma clave, incrementando su tamaño con cada fotograma de la interpolación (`true`) o si no se escala (`false`).

Ejemplo
El siguiente ejemplo especifica que el elemento interpolado se debe escalar al tamaño del objeto del fotograma clave siguiente, aumentando el tamaño con cada fotograma de la interpolación.

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenScale = true;
```

frame.motionTweenSnap

Disponibilidad
Flash MX 2004

Uso
`frame.motionTweenSnap`

Descripción
Propiedad; un valor booleano que especifica si el elemento interpolado se ajusta automáticamente al punto más próximo de la capa de guía de movimiento asociada a la capa de este fotograma (`true`) o si no se ajusta (`false`).

frame.motionTweenSync

Disponibilidad
Flash MX 2004

Uso
`frame.motionTweenSync`

Descripción
Propiedad; un valor booleano que si se define como `true`, sincroniza la animación del objeto interpolado con la línea de tiempo principal.

Ejemplo
El ejemplo siguiente especifica que el objeto interpolado debe sincronizarse con la línea de tiempo:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenSync = true;
```

frame.name

Disponibilidad
Flash MX 2004

Uso
`frame.name`

Descripción
Propiedad; una cadena que especifica el nombre del fotograma.

Ejemplo
El ejemplo siguiente define el nombre del primer fotograma de la capa superior como `"First Frame"` y, a continuación, almacena el valor `name` en la variable `frameLabel`:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].name = 'First Frame';
var frameLabel = fl.getDocumentDOM().getTimeline().layers[0].frames[0].name;
```

frame.setCustomEase()

Disponibilidad

Flash 8

Uso

```
frame.setCustomEase(property, easeCurve)
```

Parámetros

`property` Una cadena que especifica para qué propiedad debe emplearse la curva de suavizado. Los valores aceptables son `"all"`, `"position"`, `"rotation"`, `"scale"`, `"color"` y `"filters"`.

`easeCurve` Un conjunto de objetos que define la curva de suavizado. Cada elemento del conjunto debe ser un objeto JavaScript con propiedades *x* e *y*.

Valor devuelto

Ninguno.

Descripción

Método; especifica un conjunto de coordenadas de puntos de control y puntos finales tangentes que describen una curva cúbica Bézier que se utilizará como curva de suavizado personalizada. Este conjunto se construye mediante la posición horizontal (ordinal: de izquierda a derecha) de los puntos de control y los puntos finales tangentes.

Ejemplo

En el siguiente ejemplo se establece la curva de suavizado para todas las propiedades del primer fotograma en la primera capa en la curva de Bézier especificada mediante el conjunto `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 );
```

Véase también

`frame.getCustomEase()`, `frame.hasCustomEase`, `frame.useSingleEaseCurve`

frame.shapeTweenBlend

Disponibilidad

Flash MX 2004

Uso

```
frame.shapeTweenBlend
```

Descripción

Propiedad; una cadena que especifica cómo se mezcla una interpolación de forma entre la forma del fotograma clave al comienzo de la interpolación y la forma del siguiente fotograma clave. Los valores aceptables son `"distributive"` y `"angular"`.

frame.soundEffect

Disponibilidad

Flash MX 2004

Uso

```
frame.soundEffect
```

Descripción

Propiedad; una cadena que especifica efectos para un sonido que está asociado directamente a un fotograma (`frame.soundLibraryItem`). Los valores aceptables son `"none"`, `"left channel"`, `"right channel"`, `"fade left to right"`, `"fade right to left"`, `"fade in"`, `"fade out"` y `"custom"`.

Ejemplo

El ejemplo siguiente especifica que el sonido asociado al primer fotograma debe desaparecer de forma paulatina:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundEffect = "fade in";
```

frame.soundLibraryItem

Disponibilidad

Flash MX 2004

Uso

```
frame.soundLibraryItem
```

Descripción

Propiedad; un elemento de biblioteca (consulte Objeto SoundItem) empleado para crear un sonido. El sonido se asocia directamente al fotograma.

Ejemplo

El siguiente ejemplo asigna el primer elemento de la biblioteca a la propiedad `soundLibraryItem` del primer fotograma:

```
// 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

Disponibilidad
Flash MX 2004

Uso
`frame.soundLoop`

Descripción
Propiedad; un valor entero que especifica el número de veces que se reproduce un sonido asociado directamente a un fotograma (`frame.soundLibraryItem`). Si desea especificar un valor para esta propiedad, defina `frame.soundLoopMode` como `"repeat"`.

Ejemplo
Consulte `frame.soundLoopMode`.

frame.soundLoopMode

Disponibilidad
Flash MX 2004

Uso
`frame.soundLoopMode`

Descripción
Propiedad; una cadena que especifica si un sonido asociado directamente a un fotograma (`frame.soundLibraryItem`) debe reproducirse un número específico de veces o realizar un bucle indefinido. Los valores aceptables son `"repeat"` y `"loop"`. Para especificar el número de veces que el sonido debe reproducirse, defina un valor para `frame.soundLoop`.

Ejemplo
El ejemplo siguiente especifica que un sonido debe reproducirse dos veces:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLoopMode = "repeat";
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLoop = 2;
```

frame.soundName

Disponibilidad
Flash MX 2004

Uso
`frame.soundName`

Descripción
Propiedad; una cadena que especifica el nombre de un sonido que está asociado directamente a un fotograma (`frame.soundLibraryItem`), tal como está almacenado en la biblioteca.

Ejemplo
El ejemplo siguiente cambia la propiedad `soundName` del primer fotograma a `"song1.mp3"`; song1.mp3 debe estar presente en la biblioteca:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundName = "song1.mp3";
```

frame.soundSync

Disponibilidad
Flash MX 2004

Uso
```
frame.soundSync
```

Descripción
Propiedad; una cadena que especifica el comportamiento de sincronización de un sonido que está asociado directamente a un fotograma (`frame.soundLibraryItem`). Los valores aceptables son `"event"`, `"stop"`, `"start"` y `"stream"`.

Ejemplo
El ejemplo siguiente especifica que un sonido debe reproducirse sin interrupción:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundSync = 'stream';
```

frame.startFrame

Disponibilidad
Flash MX 2004

Uso
```
frame.startFrame
```

Descripción
Propiedad de sólo lectura; el índice del primer fotograma de una secuencia.

Ejemplo
En el ejemplo siguiente, `stFrame` es el índice del primer fotograma de la secuencia de fotogramas. En este ejemplo, una secuencia de fotogramas agrupa los seis fotogramas desde Fotograma 5 hasta Fotograma 10. Por lo tanto, el valor de `stFrame` en cualquier fotograma entre el Fotograma 5 y el Fotograma 10 será 4 (recuerde que los valores de índice no son los valores de número de fotograma).

```
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

Disponibilidad
Flash MX 2004

Uso
`frame.tweenEasing`

Descripción
Propiedad; un entero que especifica la cantidad de suavizado que debe aplicarse al objeto interpolado. Los valores aceptables son de -100 a 100. Para empezar la interpolación de movimiento lentamente y acelerar la interpolación hacia el final de la animación, utilice un valor negativo entre -1 y -100. Para empezar la interpolación de movimiento rápidamente y decelerar la interpolación hacia el final de la animación, utilice un valor positivo entre 1 y -100.

Ejemplo
El ejemplo siguiente especifica que el movimiento del objeto interpolado debe comenzar con bastante rapidez y desacelerarse hacia el final de la animación:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].tweenEasing = 50;
```

frame.tweenType

Disponibilidad
Flash MX 2004

Uso
`frame.tweenType`

Descripción
Propiedad; una cadena que especifica el tipo de interpolación; los valores aceptables son `"motion"`, `"shape"` y `"none"`. El valor `"none"` elimina la interpolación de movimiento. Utilice el método `timeline.createMotionTween()` para crear una interpolación de movimiento.

Si especifica `"motion"`, el objeto del fotograma deberá ser un símbolo, un campo de texto o un objeto agrupado. Se interpolará desde su ubicación en el fotograma clave actual hasta la ubicación del siguiente fotograma clave.

Si especifica `"shape"`, el objeto del fotograma deberá ser un objeto. Se mezclará desde su forma en el fotograma clave actual hasta la forma del siguiente fotograma clave.

Ejemplo
El ejemplo siguiente especifica que el objeto tiene una interpolación de movimiento y, por tanto, debe interpolarse desde su ubicación en el fotograma clave actual hasta la ubicación del siguiente fotograma clave:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].tweenType = "motion";
```

frame.useSingleEaseCurve

Disponibilidad

Flash 8

Uso

```
frame.useSingleEaseCurve
```

Descripción

Propiedad; un valor booleano. Si es `true`, se utilizará una curva de suavizado personalizada para la información de suavizado de todas las propiedades. Si es `false`, cada propiedad tendrá su propia curva.

Esta propiedad se ignora si no se ha aplicado al fotograma el suavizado personalizado.

Ejemplo

El ejemplo siguiente especifica que una curva de suavizado personalizada debe emplearse para todas las propiedades del primer fotograma de la primera capa:

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
theFrame.useSingleEaseCurve = true;
```

Véase también

`frame.getCustomEase()`, `frame.hasCustomEase`, `frame.setCustomEase()`

Capítulo 22: Objeto HalfEdge

Disponibilidad
Flash MX 2004

Descripción
El objeto HalfEdge es el lado dirigido del borde de un Objeto Shape. Un borde consta de dos bordes partidos. Se pueden atravesar los contornos de una forma rodeando estos bordes partidos. Por ejemplo, comenzando desde un borde partido, puede trazar todos los bordes partidos alrededor del contorno de una forma y volver al original.

Los bordes partidos están ordenados. Mientras que un borde partido representa un lado del borde; el otro representa el otro lado.

Resumen de métodos
Los métodos siguientes están disponibles para el objeto HalfEdge:

Método	Descripción
`halfEdge.getEdge()`	Obtiene el Objeto Edge para el objeto HalfEdge.
`halfEdge.getNext()`	Obtiene el siguiente borde partido del contorno actual.
`halfEdge.getOppositeHalfEdge()`	Obtiene el objeto HalfEdge del otro lado del borde.
`halfEdge.getPrev()`	Obtiene el objeto HalfEdge anterior del contorno actual.
`halfEdge.getVertex()`	Obtiene el Objeto Vertex en la cabeza del objeto HalfEdge.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto HalfEdge:

Propiedad	Descripción
`halfEdge.id`	De sólo lectura; un identificador de entero exclusivo para el objeto HalfEdge.
`halfEdge.index`	Un entero con un valor de 0 ó 1 que especifica el índice para este objeto HalfEdge en el borde principal.

halfEdge.getEdge()

Disponibilidad
Flash MX 2004

Uso
```
halfEdge.getEdge()
```

Parámetros
Ninguno.

Valor devuelto
Un Objeto Edge.

Descripción
Método; obtiene el objeto Edge para el objeto HalfEdge. Consulte Objeto Edge.

Ejemplo
El ejemplo siguiente ilustra la obtención de un borde y de un borde partido para la forma especificada:

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge(0);
var edge = hEdge.getEdge();
```

halfEdge.getNext()

Disponibilidad
Flash MX 2004

Uso
`halfEdge.getNext()`

Parámetros
Ninguno.

Valor devuelto
Un objeto HalfEdge.

Descripción
Método; obtiene el siguiente borde partido del contorno actual.

Nota: *los bordes partidos tienen una dirección y un orden de secuencia, a diferencia de los bordes.*

Ejemplo
El ejemplo siguiente almacena el siguiente borde partido del contorno especificado en la variable `nextHalfEdge`:

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge( 0 );
var nextHalfEdge = hEdge.getNext();
```

halfEdge.getOppositeHalfEdge()

Disponibilidad
Flash MX 2004

Uso
`halfEdge.getOppositeHalfEdge()`

Parámetros
Ninguno.

Valor devuelto
Un objeto HalfEdge.

Descripción
Método; obtiene el objeto HalfEdge del otro lado del borde.

Ejemplo
El ejemplo siguiente almacena el borde partido situado frente a `hEdge` en la variable `otherHalfEdge`:

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge(0);
var otherHalfEdge = hEdge.getOppositeHalfEdge();
```

halfEdge.getPrev()

Disponibilidad
Flash MX 2004

Uso
`halfEdge.getPrev()`

Parámetros
Ninguno.

Valor devuelto
Un objeto HalfEdge.

Descripción
Método; obtiene el objeto HalfEdge anterior del contorno actual.

Nota: *los bordes partidos tienen una dirección y un orden de secuencia, a diferencia de los bordes.*

Ejemplo
El ejemplo siguiente almacena el borde partido anterior del contorno especificado en la variable `prevHalfEdge`:

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge( 0 );
var prevHalfEdge = hEdge.getPrev();
```

halfEdge.getVertex()

Disponibilidad
Flash MX 2004

Uso

```
halfEdge.getVertex()
```

Parámetros

Ninguno.

Valor devuelto

Un Objeto Vertex

Descripción

Método; obtiene el objeto Vertex en la cabeza del objeto HalfEdge. Consulte Objeto Vertex.

Ejemplo

El ejemplo siguiente almacena el objeto Vertex en la cabeza de `hEdge` en la variable `vertex`:

```
var shape = fl.getDocumentDOM().selection[0];
var edge = shape.edges[0];
var hEdge = edge.getHalfEdge(0);
var vertex = hEdge.getVertex();
```

halfEdge.id

Disponibilidad

Flash MX 2004

Uso

```
halfEdge.id
```

Descripción

Propiedad de sólo lectura; un identificador de entero exclusivo para el objeto HalfEdge.

Ejemplo

El ejemplo siguiente muestra un identificador exclusivo para el borde partido especificado en el panel Salida:

```
var shape = fl.getDocumentDOM().selection[0];
alert(shape.contours[0].getHalfEdge().id);
```

halfEdge.index

Disponibilidad

Flash MX 2004

Uso

```
halfEdge.index
```

Descripción

Propiedad de sólo lectura; un entero con un valor de 0 ó 1 que especifica el índice para este objeto HalfEdge en el borde principal.

Ejemplo

El ejemplo siguiente muestra el valor de índice para el borde partido especificado en el panel Salida:

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge(0);
var heIndex = hEdge.index;
```

Capítulo 23: Objeto Instance

Herencia Objeto Element > Objeto Instance

Disponibilidad
Flash MX 2004

Descripción
Instance es una subclase del Objeto Element.

Resumen de propiedades
Además de todas las propiedades del objeto Element, el objeto Instance dispone de las propiedades siguientes:

Propiedad	Descripción
`instance.instanceType`	De sólo lectura; una cadena que representa el tipo de instancia.
`instance.libraryItem`	Elemento de biblioteca empleado para crear una instancia de esta instancia.

instance.instanceType

Disponibilidad
Flash MX 2004; valor posible de `"video"` añadido en Flash 8.

Uso
`instance.instanceType`

Descripción
Propiedad de sólo lectura; una cadena que representa el tipo de instancia. Los valores posibles son `"symbol"`, `"bitmap"`, `"embedded video"`, `"linked video"`, `"video"` y `"compiled clip"`.

En Flash MX 2004, el valor de `instance.instanceType` para un elemento añadido a la biblioteca mediante `library.addNewItem("video")` es `"embedded_video"`. En Flash 8 y versiones posteriores, el valor es `"video"`. Consulte `library.addNewItem()`.

Ejemplo
El ejemplo siguiente muestra que el tipo de instancia de un clip de película es `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

Disponibilidad

Flash MX 2004

Uso

```
instance.libraryItem
```

Descripción

Propiedad; un elemento de biblioteca empleado para crear una instancia de esta instancia. Sólo puede cambiar esta propiedad a otro elemento de biblioteca del mismo tipo (es decir, no puede definir una instancia `symbol` para hacer referencia a un mapa de bits). Consulte Objeto library.

Ejemplo

El ejemplo siguiente cambia el símbolo seleccionado para hacer referencia al primer elemento de la biblioteca:

```
fl.getDocumentDOM().selection[0].libraryItem = fl.getDocumentDOM().library.items[0];
```

Capítulo 24: Objeto Item

Disponibilidad

Flash MX 2004

Descripción

El objeto Item es una clase base abstracta. Todos los contenidos de la biblioteca se derivan de Item. Véase también Objeto library.

Resumen de métodos

Los métodos siguientes están disponibles para el objeto Item:

Método	Descripción
`item.addData()`	Añade datos especificados a un elemento de biblioteca.
`item.getData()`	Recupera el valor de los datos especificados.
`item.hasData()`	Determina si el elemento de biblioteca tiene los datos con nombre.
`item.removeData()`	Elimina datos persistentes del elemento de biblioteca.

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto Item:

Propiedad	Descripción
`item.itemType`	De sólo lectura; una cadena que especifica el tipo de elemento.
`item.linkageBaseClass`	Una cadena que especifica la clase de ActionScript 3.0 que se asociará al símbolo.
`item.linkageClassName`	Una cadena que especifica la clase de ActionScript 2.0 que se asociará al símbolo.
`item.linkageExportForAS`	Un valor booleano. Si es `true`, el elemento se exportará para ActionScript.
`item.linkageExportForRS`	Un valor booleano. Si es `true`, el elemento se exportará para compartirlo en tiempo de ejecución.
`item.linkageExportInFirstFrame`	Un valor booleano. Si es `true`, el elemento se exportará en el primer fotograma.
`item.linkageIdentifier`	Una cadena que especifica el nombre que utilizará Flash para identificar el activo cuando cree un vínculo con el archivo SWF de destino.
`item.linkageImportForRS`	Un valor booleano. Si es `true`, el elemento se importará para compartirlo en tiempo de ejecución.
`item.linkageURL`	Una cadena que especifica la URL donde se encuentra el archivo SWF que contiene el activo compartido.
`item.name`	Una cadena que especifica el nombre del elemento de biblioteca, que incluye la estructura de carpetas.

item.addData()

Disponibilidad
Flash MX 2004

Uso
`item.addData(name, type, data)`

Parámetros
`name` Una cadena que especifica el nombre de los datos.

`type` Una cadena que especifica el tipo de los datos. Los tipos válidos son `"integer"`, `"integerArray"`, `"double"`, `"doubleArray"`, `"string"` y `"byteArray"`.

`data` Los datos que se van a añadir al elemento de biblioteca especificado. El tipo de datos depende del valor del parámetro type. Por ejemplo si type es "`integer`", el valor de los datos deberá ser un entero, etc.

Valor devuelto
Ninguno.

Descripción
Método; añade datos especificados a un elemento de biblioteca.

Ejemplo
El ejemplo siguiente añade datos con el nombre `myData` con un valor entero de 12 al primer elemento de la biblioteca:

```
fl.getDocumentDOM().library.items[0].addData("myData", "integer", 12);
```

item.getData()

Disponibilidad
Flash MX 2004

Uso
`item.getData(name)`

Parámetros
`name` Una cadena que especifica el nombre de los datos que se van a recuperar.

Valor devuelto
Los datos especificados por el parámetro *name*. El tipo de datos devueltos depende del tipo de datos almacenados.

Descripción
Método; recupera el valor de los datos especificados.

Ejemplo

El ejemplo siguiente obtiene el valor de los datos con nombre `myData` del primer elemento de la biblioteca y lo almacena en la variable `libData`:

```
var libData = fl.getDocumentDOM().library.items[0].getData("myData");
```

item.hasData()

Disponibilidad

Flash MX 2004

Uso

```
item.hasData(name)
```

Parámetros

`name` Una cadena que especifica el nombre de los datos que se van a comprobar en el elemento de biblioteca.

Valor devuelto

Un valor booleano; `true` si existen los datos especificados; en caso contrario, `false`.

Descripción

Método; determina si el elemento de biblioteca tiene los datos con nombre.

Ejemplo

El ejemplo siguiente muestra un mensaje en el panel Salida si el primer elemento de la biblioteca contiene datos con el nombre `myData`:

```
if (fl.getDocumentDOM().library.items[0].hasData("myData")){
    fl.trace("Yep, it's there!");
}
```

item.itemType

Disponibilidad

Flash MX 2004

Uso

```
item.itemType
```

Descripción

Propiedad de sólo lectura; una cadena que especifica el tipo de elemento. El valor puede ser: `"undefined"`, `"component"`, `"movie clip"`, `"graphic"`, `"button"`, `"folder"`, `"font"`, `"sound"`, `"bitmap"`, `"compiled clip"`, `"screen"` o `"video"`. Si esta propiedad es `"video"`, puede determinar el tipo de vídeo; consulte `videoItem.videoType`.

Ejemplo

El ejemplo siguiente muestra el tipo de la biblioteca especificada en el panel Salida:

```
fl.trace(fl.getDocumentDOM().library.items[0].itemType);
```

item.linkageBaseClass

Disponibilidad

Flash CS3 Professional

Uso

```
item.linkageBaseClass
```

Descripción

Propiedad; una cadena que especifica la clase de ActionScript 3.0 que se asociará al símbolo. El valor especificado aquí aparece en el cuadro de diálogo Vinculación del entorno de edición, así como en otros cuadros de diálogo que incluyen los controles de éste, como el de propiedades de símbolo. (Para especificar este valor para ActionScript 2.0, utilice `item.linkageClassName`.)

Si la clase base es la predeterminada para el tipo de símbolo (por ejemplo, "flash.display.MovieClip" para clips de película, "flash.display.SimpleButton" para botones, etc.), esta propiedad es una cadena vacía (""). De igual forma, para convertir un elemento en la clase base predeterminada, establezca este valor como cadena vacía.

Al establecer este valor, no se realiza ninguna de las comprobaciones realizadas por el cuadro de diálogo Vinculación, y no se emite ningún error si Flash no puede definir la clase base con el valor especificado. Por ejemplo, si se establece este valor en el cuadro de diálogo Vinculación se realizan las comprobaciones para asegurarse de que se puede encontrar la clase base en la ruta de clase del archivo FLA. De este modo, se garantiza que se elija ActionScript 3.0 en la ficha Flash del cuadro de diálogo Configuración de publicación y así sucesivamente. Estas comprobaciones no se realizan cuando se define esta propiedad en un script.

Ejemplo

Las líneas de código siguientes muestran algunas maneras de utilizar esta propiedad:

```
// 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);
```

Véase también

`document.docClass`

item.linkageClassName

Disponibilidad

Flash MX 2004

Uso

```
item.linkageClassName
```

Descripción

Propiedad; una cadena que especifica la clase de ActionScript 2.0 que se asociará al símbolo. (Para especificar este valor para ActionScript 3.0, utilice `item.linkageBaseClass`.)

Para que esta propiedad esté definida, las propiedades `item.linkageExportForAS` y/o `item.linkageExportForRS` se deben establecer como `true`, y la propiedad `item.linkageImportForRS` debe establecerse como `false`.

Ejemplo

El ejemplo siguiente especifica que el nombre de clase de ActionScript 2.0 asociado al primer elemento de la biblioteca es `myClass`:

```
fl.getDocumentDOM().library.items[0].linkageClassName = "myClass";
```

item.linkageExportForAS

Disponibilidad

Flash MX 2004

Uso

```
item.linkageExportForAS
```

Descripción

Propiedad; un valor booleano. Si esta propiedad es `true`, el elemento se exportará para ActionScript. También puede establecer las propiedades `item.linkageExportForRS` y `item.linkageExportInFirstFrame` como `true`.

Si define esta propiedad como true, la propiedad `item.linkageImportForRS` debe definirse como `false`. Asimismo, deberá especificar un identificador (`item.linkageIdentifier`) y una URL (`item.linkageURL`).

Ejemplo

El ejemplo siguiente define esta propiedad para el elemento de biblioteca especificado:

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

item.linkageExportForRS

Disponibilidad

Flash MX 2004

Uso

```
item.linkageExportForRS
```

Descripción

Propiedad; un valor booleano. Si esta propiedad es `true`, el elemento se exportará para compartir en tiempo de ejecución. También puede establecer las propiedades `item.linkageExportForAS` y `item.linkageExportInFirstFrame` como `true`.

Si define esta propiedad como true, la propiedad `item.linkageImportForRS` debe definirse como `false`. Asimismo, deberá especificar un identificador (`item.linkageIdentifier`) y una URL (`item.linkageURL`).

Ejemplo

El ejemplo siguiente define esta propiedad para el elemento de biblioteca especificado:

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

item.linkageExportInFirstFrame

Disponibilidad

Flash MX 2004

Uso

```
item.linkageExportInFirstFrame
```

Descripción

Propiedad; un valor booleano. Si es `true`, el elemento se exportará en el primer fotograma; si es `false`, se exportará en el fotograma de la primera instancia. Si el elemento no aparece en el escenario, no se exportará.

El valor de esta propiedad sólo se puede definir como `true` cuando `item.linkageExportForAS` y/o `item.linkageExportForRS` se definen como `true`.

Ejemplo

El ejemplo siguiente especifica que el elemento de biblioteca especificado se exporta en el primer fotograma:

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

item.linkageIdentifier

Disponibilidad

Flash MX 2004

Uso

```
item.linkageIdentifier
```

Descripción

Propiedad; una cadena que especifica el nombre que utilizará Flash para identificar el activo cuando cree un vínculo con el archivo SWF de destino. Flash no tiene en cuenta esta propiedad si `item.linkageImportForRS`, `item.linkageExportForAS` y `item.linkageExportForRS` se establecen como `false`. Por el contrario, se debe definir esta propiedad cuando cualquiera de dichas propiedades estén definidas como `true`.

Ejemplo

El ejemplo siguiente especifica que la cadena `my_mc` se utilizará para identificar el elemento de biblioteca cuando se vincule al archivo SWF de destino al que se está exportando:

```
fl.getDocumentDOM().library.items[0].linkageIdentifier = "my_mc";
```

Véase también

`item.linkageURL`

item.linkageImportForRS

Disponibilidad

Flash MX 2004

Uso

```
item.linkageImportForRS
```

Descripción

Propiedad; un valor booleano: si es `true`, el elemento se importará para compartirlo en tiempo de ejecución. Si esta propiedad se establece como `true`, tanto `item.linkageExportForAS` como `item.linkageExportForRS` se deben establecer como `false`. Asimismo, deberá especificar un identificador (`item.linkageIdentifier`) y una URL (`item.linkageURL`).

Ejemplo

El ejemplo siguiente define esta propiedad como `true` para el elemento de biblioteca especificado:

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

item.linkageURL

Disponibilidad

Flash MX 2004

Uso

```
item.linkageURL
```

Descripción

Propiedad; una cadena que especifica la URL donde se encuentra el archivo SWF que contiene el activo compartido. Flash no tiene en cuenta esta propiedad si `item.linkageImportForRS`, `item.linkageExportForAS` y `item.linkageExportForRS` se establecen como `false`. Por el contrario, se debe definir esta propiedad cuando cualquiera de dichas propiedades estén definidas como `true`. Puede especificar una URL Web o un nombre de archivo con formato dependiente de la plataforma (es decir, barras diagonales [/] o barras invertidas [\], según la plataforma).

Ejemplo

El ejemplo siguiente especifica una URL de vinculación para el elemento de biblioteca especificado:

```
fl.getDocumentDOM().library.items[0].linkageURL = "theShareSWF.swf";
```

Véase también

item.linkageIdentifier

item.name

Disponibilidad

Flash MX 2004

Uso

```
item.name
```

Descripción

Método; una cadena que especifica el nombre del elemento de biblioteca, que incluye la estructura de carpetas. Por ejemplo, si Symbol_1 se encuentra en una carpeta llamada Folder_1, la propiedad `name` de Symbol_1 será `"Folder_1/Symbol_1"`.

Ejemplo

El ejemplo siguiente muestra el nombre de la biblioteca especificada en el panel Salida:

```
fl.trace(fl.getDocumentDOM().library.items[0].name);
```

item.removeData()

Disponibilidad

Flash MX 2004

Uso

```
item.removeData(name)
```

Parámetros

`name` Especifica el nombre de los datos que se van a eliminar del elemento de biblioteca.

Valor devuelto

Ninguno.

Descripción

Propiedad; elimina datos persistentes del elemento de biblioteca.

Ejemplo

El ejemplo siguiente elimina los datos con el nombre `myData` del primer elemento de la biblioteca:

```
fl.getDocumentDOM().library.items[0].removeData("myData");
```

Capítulo 25: Objeto Layer

Disponibilidad
Flash MX 2004

Descripción
El objeto Layer representa una capa en la línea de tiempo. La propiedad `timeline.layers` contiene un conjunto de objetos Layer a los que puede acceder `fl.getDocumentDOM().getTimeline().layers`.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto Layer:

Propiedad	Descripción
`layer.color`	Una cadena, valor hexadecimal o entero que especifica el color asignado al contorno de la capa.
`layer.frameCount`	De sólo lectura; un entero que especifica el número de fotogramas de la capa.
`layer.frames`	De sólo lectura; un conjunto de objetos Frame.
`layer.height`	Un entero que especifica la altura de la capa en porcentaje; equivale al valor de valor de altura de Capa del cuadro de diálogo Propiedades de Capa.
`layer.layerType`	Una cadena que especifica el uso actual de la capa; equivale a la opción Tipo del cuadro de diálogo Propiedades de Capa.
`layer.locked`	Un valor booleano que especifica el estado bloqueado de la capa.
`layer.name`	Una cadena que especifica el nombre de la capa.
`layer.outline`	Un valor booleano que especifica el estado de los contornos para todos los objetos de la capa.
`layer.parentLayer`	Un objeto Layer que representa la carpeta o capa de guía o enmascaramiento que contiene la capa.
`layer.visible`	Un valor booleano que especifica si se muestran u ocultan los objetos de la capa en el escenario.

layer.color

Disponibilidad
Flash MX 2004

Uso
`layer.color`

Descripción
Propiedad; el color asignado al contorno de la capa, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

  Esta propiedad equivale a la opción Color de contorno del cuadro de diálogo Propiedades de Capa.

Ejemplo

El ejemplo siguiente almacena el valor de la primera capa en la variable `colorValue`:

```
var colorValue = fl.getDocumentDOM().getTimeline().layers[0].color;
```

El ejemplo siguiente muestra tres formas de definir el color de la primera capa como rojo:

```
fl.getDocumentDOM().getTimeline().layers[0].color=16711680;
fl.getDocumentDOM().getTimeline().layers[0].color="#ff0000";
fl.getDocumentDOM().getTimeline().layers[0].color=0xFF0000;
```

layer.frameCount

Disponibilidad

Flash MX 2004

Uso

```
layer.frameCount
```

Descripción

Propiedad de sólo lectura; un entero que especifica el número de fotogramas de la capa.

Ejemplo

El ejemplo siguiente almacena el número de fotogramas de la primera capa en la variable `fcNum`:

```
var fcNum = fl.getDocumentDOM().getTimeline().layers[0].frameCount;
```

layer.frames

Disponibilidad

Flash MX 2004

Uso

```
layer.frames
```

Descripción

Propiedad de sólo lectura; un conjunto de objetos Frame (consulte Objeto Frame).

Ejemplo

El ejemplo siguiente define la variable `frameArray` como el conjunto de objetos Frame para los fotogramas del documento actual:

```
var frameArray = fl.getDocumentDOM().getTimeline().layers[0].frames;
```

Para determinar si un fotograma es un fotograma clave, compruebe si la propiedad `frame.startFrame` coincide con el índice del conjunto, como se muestra en el ejemplo siguiente:

```
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

Disponibilidad
Flash MX 2004

Uso
`layer.height`

Descripción
Propiedad; un entero que especifica la altura de la capa en porcentaje; equivale al valor de valor de altura de Capa del cuadro de diálogo Propiedades de Capa. Los valores válidos representan porcentajes de la altura predeterminada: 100, 200 ó 300.

Ejemplo
El ejemplo siguiente almacena el valor de porcentaje de la altura de la primera capa:

```
var layerHeight = fl.getDocumentDOM().getTimeline().layers[0].height;
```

El ejemplo siguiente define la altura de la primera capa en 300%:

```
fl.getDocumentDOM().getTimeline().layers[0].height = 300;
```

layer.layerType

Disponibilidad
Flash MX 2004

Uso
`layer.layerType`

Descripción
Propiedad; una cadena que especifica el uso actual de la capa; equivale a la opción Tipo del cuadro de diálogo Propiedades de Capa. Los valores aceptables son `"normal"`, `"guide"`, `"guided"`, `"mask"`, `"masked"` y `"folder"`.

Ejemplo
El ejemplo siguiente define la primera capa de la línea de tiempo con el tipo `folder`:

```
fl.getDocumentDOM().getTimeline().layers[0].layerType = "folder";
```

layer.locked

Disponibilidad
Flash MX 2004

Uso
`layer.locked`

Descripción
Propiedad; un valor booleano que especifica el estado bloqueado de la capa. Si se define como `true`, la capa se bloqueará. El valor predeterminado es `false`.

Ejemplo
El ejemplo siguiente almacena el valor booleano del estado de la primera capa en la variable `lockStatus`:

```
var lockStatus = fl.getDocumentDOM().getTimeline().layers[0].locked;
```

El ejemplo siguiente define el estado de la primera capa como desbloqueado:

```
fl.getDocumentDOM().getTimeline().layers[0].locked = false;
```

layer.name

Disponibilidad
Flash MX 2004

Uso
`layer.name`

Descripción
Propiedad; una cadena que especifica el nombre de la capa.

Ejemplo
El ejemplo siguiente define el nombre de la primera capa del documento actual como `foreground`:

```
fl.getDocumentDOM().getTimeline().layers[0].name = "foreground";
```

layer.outline

Disponibilidad
Flash MX 2004

Uso
`layer.outline`

Descripción

Propiedad; un valor booleano que especifica el estado de los contornos para todos los objetos de la capa. Si se define como `true`, todos los objetos de la capa aparecerán sólo con contornos. Si es `false`, los objetos aparecerán tal como se crearon.

Ejemplo

El ejemplo siguiente hace que todos los objetos de la primera capa aparezcan sólo con contornos:

```
fl.getDocumentDOM().getTimeline().layers[0].outline = true;
```

layer.parentLayer

Disponibilidad

Flash MX 2004

Uso

```
layer.parentLayer
```

Descripción

Propiedad; un objeto Layer que representa la carpeta o capa de guía o enmascaramiento que contiene la capa. La capa superior debe ser una carpeta, guía o capa de máscara que preceda a la capa, o la capa `parentLayer` de la capa anterior o siguiente. Al definir la `parentLayer` de la capa no se moverá la posición de la capa en la lista; el intento de definir una `parentLayer` de una capa que requiera movimiento no surtirá ningún efecto. Utiliza `null` para una capa de nivel superior.

Ejemplo

El ejemplo siguiente utiliza dos capas al mismo nivel en la misma línea de tiempo. La primera capa (`layers[0]`) se convierte en una carpeta y, a continuación, se define como carpeta principal de la segunda carpeta (`layers[1]`). Esta acción desplaza la segunda capa dentro de la primera.

```
var parLayer = fl.getDocumentDOM().getTimeline().layers[0];
parLayer.layerType = "folder";
fl.getDocumentDOM().getTimeline().layers[1].parentLayer = parLayer;
```

layer.visible

Disponibilidad

Flash MX 2004

Uso

```
layer.visible
```

Descripción

Propiedad; un valor booleano que especifica si se muestran u ocultan los objetos de la capa en el escenario. Si se define como `true`, todos los objetos de la capa estarán visibles; si es `false`, estarán ocultos. El valor predeterminado es `true`.

Ejemplo

El ejemplo siguiente oculta todos los objetos de la primera capa:

```
fl.getDocumentDOM().getTimeline().layers[0].visible = false;
```

Capítulo 26: Objeto library

Disponibilidad

Flash MX 2004

Descripción

El objeto Library representa el panel Biblioteca. Es una propiedad del objeto Document (consulte `document.library`) y se puede acceder a él a través de `fl.getDocumentDOM().library`.

El objeto library contiene un conjunto de elementos de distintos tipos, como símbolos, mapas de bits, sonidos y vídeo.

Resumen de métodos

Los métodos siguientes están disponibles para el objeto library:

Método	Descripción
`library.addItemToDocument()`	Añade el elemento actual o especificado al escenario en la posición especificada.
`library.addNewItem()`	Crea un elemento nuevo del tipo especificado en el panel Biblioteca y define el nuevo elemento como elemento seleccionado actualmente.
`library.deleteItem()`	Elimina los elementos actuales o un elemento especificado del panel Biblioteca.
`library.duplicateItem()`	Realiza una copia del elemento especificado o seleccionado actualmente.
`library.editItem()`	Abre el elemento especificado o seleccionado actualmente en modo de edición.
`library.expandFolder()`	Expande o contrae la carpeta especificada o seleccionada actualmente en la biblioteca.
`library.findItemIndex()`	Devuelve el valor de índice del elemento de biblioteca (basado en cero).
`library.getItemProperty()`	Obtiene la propiedad del elemento seleccionado.
`library.getItemType()`	Obtiene el tipo de objeto seleccionado actualmente o especificado por una ruta de biblioteca.
`library.getSelectedItems()`	Obtiene el conjunto de todos los elementos seleccionados actualmente en la biblioteca.
`library.importEmbeddedSWF()`	Importa un archivo SWF a la biblioteca como clip compilado.
`library.itemExists()`	Comprueba si el elemento especificado existe en la biblioteca.
`library.moveToFolder()`	Desplaza el elemento de biblioteca seleccionado actualmente o especificado a una carpeta especificada.
`library.newFolder()`	Crea una carpeta con el nombre especificado o un nombre predeterminado en la carpeta seleccionada (`"untitled folder #"`) si no se suministra ningún parámetro `folderName`.
`library.renameItem()`	Cambia el nombre del elemento de biblioteca seleccionado actualmente en el panel Biblioteca.
`library.selectAll()`	Selecciona o anula la selección de todos los elementos de la biblioteca.
`library.selectItem()`	Selecciona un elemento de biblioteca especificado.

Método	Descripción
`library.selectNone()`	Anula la selección de todos los elementos de la biblioteca.
`library.setItemProperty()`	Define la propiedad de todos los elementos de biblioteca seleccionados (ignorando las carpetas).
`library.updateItem()`	Actualiza el elemento especificado.

Resumen de propiedades del objeto library

La propiedad siguiente está disponible para el objeto library.

Propiedad	Descripción
`library.items`	Un conjunto de objetos Item de la biblioteca

library.addItemToDocument()

Disponibilidad

Flash MX 2004

Uso

```
library.addItemToDocument(position [, namePath])
```

Parámetros

`position` Un punto que especifica la posición *x,y* del centro del elemento en el escenario.

`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, puede especificar su nombre y su ruta empleando notación con barras. Si no se especifica *namePath*, se utilizará la selección actual de la biblioteca. Este parámetro es opcional.

Valor devuelto

Un valor booleano: `true` si el elemento se añade correctamente al documento; `false` en caso contrario

Descripción

Método; añade el elemento actual o especificado al escenario en la posición especificada.

Ejemplo

El ejemplo siguiente añade el elemento seleccionado actualmente al escenario en la posición (3, 60):

```
fl.getDocumentDOM().library.addItemToDocument({x:3, y:60});
```

El ejemplo siguiente añade el elemento `Symbol1` situado en la carpeta 1 de la biblioteca al escenario en la posición (550, 485):

```
fl.getDocumentDOM().library.addItemToDocument({x:550.0, y:485.0}, "folder1/Symbol1");
```

library.addNewItem()

Disponibilidad
Flash MX 2004

Uso
`library.addNewItem(type [, namePath])`

Parámetros
`type` Una cadena que especifica el tipo de elemento que se va a crear. Los únicos valores aceptables de *type* son `"video"`, `"movie clip"`, `"button"`, `"graphic"`, `"bitmap"`, `"screen"` y `"folder"` (por ejemplo, no se puede añadir un sonido a la biblioteca con este método). Especificar una ruta de carpeta es lo mismo que utilizar library.newFolder()`library.newFolder()` antes de llamar a este método.

`namePath` Una cadena que especifica el nombre del elemento que se va a añadir. Si el elemento se encuentra en una carpeta, especifique su nombre y su ruta empleando la notación con barras. Este parámetro es opcional.

Valor devuelto
Valor booleano: `true` si el elemento se crea correctamente, y `false` en caso contrario.

Descripción
Método; crea un elemento nuevo del tipo especificado en el panel Biblioteca y define el nuevo elemento como elemento seleccionado actualmente. Para más información sobre la importación de elementos a la biblioteca, incluidos elementos como sonidos, consulte `document.importFile()`.

Ejemplo
El ejemplo siguiente crea un elemento de botón nuevo denominado `start` en una carpeta nueva llamada `folderTwo`:

```
fl.getDocumentDOM().library.addNewItem("button", "folderTwo/start");
```

library.deleteItem()

Disponibilidad
Flash MX 2004

Uso
`library.deleteItem([namePath])`

Parámetros
`namePath` Una cadena que especifica el nombre del elemento que se va a eliminar. Si el elemento se encuentra en una carpeta, puede especificar su nombre y su ruta empleando notación con barras. Si transfiere un nombre de carpeta, se eliminarán la carpeta y todos sus elementos. Si no especifica ningún nombre, Flash eliminará el elemento o elementos seleccionados actualmente. Para eliminar todos los elementos del panel Biblioteca, seleccione todos los elementos antes de utilizar este método. Este parámetro es opcional.

Valor devuelto
Valor booleano: `true` si los elementos se eliminan correctamente, y `false` en caso contrario.

Descripción
Método; elimina los elementos actuales o un elemento especificado del panel Biblioteca. Este método puede afectar a múltiples elementos si hay varios seleccionados.

Ejemplo
El ejemplo siguiente elimina el elemento seleccionado actualmente:

```
fl.getDocumentDOM().library.deleteItem();
```

El ejemplo siguiente elimina el elemento `Symbol_1` de la carpeta de biblioteca `Folder_1`:

```
fl.getDocumentDOM().library.deleteItem("Folder_1/Symbol_1");
```

library.duplicateItem()

Disponibilidad
Flash MX 2004

Uso
```
library.duplicateItem( [ namePath ] )
```

Parámetros
`namePath` Una cadena que especifica el nombre del elemento que se va a duplicar. Si el elemento se encuentra en una carpeta, puede especificar su nombre y su ruta empleando notación con barras. Este parámetro es opcional.

Valor devuelto
Valor booleano: `true` si el elemento se duplica correctamente, y `false` en caso contrario. Si hay varios elementos seleccionados, Flash devolverá `false`.

Descripción
Método; realiza una copia del elemento especificado o seleccionado actualmente. El nuevo elemento tiene un nombre predeterminado (como `item copy`) y se define como el elemento seleccionado actualmente. Si hay varios elementos seleccionados, el comando produce un error.

Ejemplo
El ejemplo siguiente crea una copia del elemento `square` en la carpeta de la biblioteca `test`:

```
fl.getDocumentDOM().library.duplicateItem("test/square");
```

library.editItem()

Disponibilidad
Flash MX 2004

Uso
```
library.editItem([namePath])
```

Parámetros

`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, puede especificar su nombre y su ruta empleando notación con barras. Si no se especifica *namePath*, se abrirá el elemento de biblioteca seleccionado en modo de edición. Si no hay ningún elemento seleccionado en la biblioteca o hay varios, aparecerá la primera escena de la línea de tiempo principal para su edición. Este parámetro es opcional.

Valor devuelto

Un valor booleano: `true` si existe el elemento especificado y se puede editar; `false` en caso contrario.

Descripción

Método; abre el elemento especificado o seleccionado actualmente en modo de edición.

Ejemplo

El ejemplo siguiente abre el elemento `circle` en la carpeta `test`de la biblioteca para su edición:

```
fl.getDocumentDOM().library.editItem("test/circle");
```

library.expandFolder()

Disponibilidad

Flash MX 2004

Uso

```
library.expandFolder(bExpand [, bRecurseNestedParents [, namePath]])
```

Parámetros

`bExpand` Un valor booleano: si es `true`, la carpeta se expande; si es `false` (opción predeterminada), la carpeta se contrae.

`bRecurseNestedParents` Un valor booleano: si es `true`, todas las carpetas de la carpeta especificada se expanden o contraen, según el valor de *bExpand*. El valor predeterminado es `false`. Este parámetro es opcional.

`namePath` Una cadena que especifica el nombre y, opcionalmente, la ruta de la carpeta que se va a expandir o contraer. Si no se especifica este parámetro, el método se aplica a la carpeta seleccionada actualmente. Este parámetro es opcional.

Valor devuelto

Un valor booleano: `true` si el elemento se expande o contrae correctamente; `false` si se produce un error o el elemento especificado no es una carpeta.

Descripción

Método; expande o contrae la carpeta especificada o seleccionada actualmente en la biblioteca.

Ejemplo

El ejemplo siguiente contrae la carpeta de prueba de la biblioteca, así como todas las carpetas que se encuentran en la carpeta de prueba (en su caso):

```
fl.getDocumentDOM().library.expandFolder(false, true, "test");
```

library.findItemIndex()

Disponibilidad
Flash MX 2004

Uso
```
library.findItemIndex(namePath)
```

Parámetros
`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, puede especificar su nombre y su ruta empleando notación con barras.

Valor devuelto
Un valor entero que representa el valor de índice basado en cero del elemento.

Descripción
Método; devuelve el valor de índice del elemento de biblioteca (basado en cero). El índice de biblioteca es plano, por lo que las carpetas se consideran parte del índice principal. Puede utilizar rutas de carpeta para especificar un elemento anidado.

Ejemplo
El ejemplo siguiente almacena el valor de índice basado en cero del elemento de biblioteca square, que se encuentra en una carpeta de prueba, en la variable `sqIndex` y, a continuación, muestra el valor de índice en un cuadro de diálogo:

```
var sqIndex = fl.getDocumentDOM().library.findItemIndex("test/square");
alert(sqIndex);
```

library.getItemProperty()

Disponibilidad
Flash MX 2004

Uso
```
library.getItemProperty(property)
```

Parámetros
`property` Una cadena. Para obtener una lista de valores que puede utilizar como parámetro *property*, consulte la tabla de resumen de propiedades del Objeto Item, junto con el resumen de propiedades para sus subclases.

Valor devuelto
Un valor de cadena para la propiedad.

Descripción
Método; obtiene la propiedad del elemento seleccionado.

Ejemplo
El ejemplo siguiente muestra un cuadro de diálogo que contiene el valor del identificador de vinculación para el símbolo cuando se hace referencia a él empleando ActionScript o para compartir tiempo de ejecución:

```
alert(fl.getDocumentDOM().library.getItemProperty("linkageIdentifier"));
```

library.getItemType()

Disponibilidad
Flash MX 2004

Uso
`library.getItemType([namePath])`

Parámetros
`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, especifique su nombre y su ruta empleando la notación con barras. Si no se especifica *namePath*, Flash proporcionará el tipo de la selección actual. Si hay varios elementos seleccionados actualmente y no se suministra *namePath*, Flash ignorará el comando. Este parámetro es opcional.

Valor devuelto
Un valor de cadena que especifica el tipo de objeto. Para posibles valores devueltos, consulte `item.itemType`.

Descripción
Método; obtiene el tipo de objeto seleccionado actualmente o especificado por una ruta de biblioteca.

Ejemplo
El ejemplo siguiente muestra un cuadro de diálogo que contiene el tipo de elemento de `Symbol_1` situado en la carpeta `Folder_1/Folder_2`:

```
alert(fl.getDocumentDOM().library.getItemType("Folder_1/Folder_2/Symbol_1"));
```

library.getSelectedItems()

Disponibilidad
Flash MX 2004

Parámetros
Ninguno.

Valor devuelto
Un conjunto de valores para todos los elementos seleccionados actualmente en la biblioteca.

Descripción
Método; obtiene el conjunto de todos los elementos seleccionados actualmente en la biblioteca.

Ejemplo

El ejemplo siguiente almacena el conjunto de elementos de biblioteca seleccionados actualmente (en este caso, varios archivos de audio) en la variable `selItems` y, a continuación, cambia la propiedad `sampleRate` del primer archivo de audio del conjunto a `11 kHz`:

```
var selItems = fl.getDocumentDOM().library.getSelectedItems();
selItems[0].sampleRate = "11 kHz";
```

library.importEmbeddedSWF()

Disponibilidad

Flash MX 2004

Uso

```
library.importEmbeddedSWF(linkageName, swfData [, libName])
```

Parámetros

`linkageName` Una cadena que proporciona el nombre de la vinculación SWF del clip de película raíz.

`swfData` Un conjunto de datos SWF binarios que procede de una biblioteca externa o DLL.

`libName` Una cadena que especifica el nombre de biblioteca del elemento creado. Si ya se utiliza el nombre, el método creará un nombre alternativo. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; importa un archivo SWF a la biblioteca como clip compilado. A diferencia de Archivo > Importar > SWF, este método permite incorporar un archivo SWF compilado dentro de la biblioteca. No hay funcionalidad equivalente en la interfaz de usuario, y este método debe emplearse con una biblioteca externa o DLL (consulte "Extensibilidad de nivel C" en la página 536).

El archivo SWF que está importando debe tener un clip de película de nivel superior que incluya todo el contenido. Dicho clip deberá disponer de su propio identificador de vinculación definido con el mismo valor que el parámetro *linkageName* transferido a este método.

Ejemplo

El ejemplo siguiente añade el archivo SWF con el valor *linkageName*`MyMovie` a la biblioteca como clip compilado con el nombre `Intro`:

```
fl.getDocumentDOM().library.importEmbeddedSWF("MyMovie", swfData, "Intro");
```

library.itemExists()

Disponibilidad

Flash MX 2004

Uso

```
library.itemExists(namePath)
```

Parámetros

`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, especifique su nombre y su ruta empleando la notación con barras.

Valor devuelto

Un valor booleano: `true` si existe el elemento especificado en la biblioteca; `false` en caso contrario.

Descripción

Método; comprueba si el elemento especificado existe en la biblioteca.

Ejemplo

El siguiente ejemplo muestra `true` o `false` en un cuadro de diálogo, dependiendo de si el elemento `Symbol_1` se encuentra en la carpeta de biblioteca `Folder_1`:

```
alert(fl.getDocumentDOM().library.itemExists('Folder_1/Symbol_1'));
```

library.items

Disponibilidad

Flash MX 2004

Uso

```
library.items
```

Descripción

Propiedad; un conjunto de objetos de elemento de la biblioteca.

Ejemplo

El ejemplo siguiente almacena el conjunto de todos los elementos de biblioteca en la variable `itemArray`:

```
var itemArray = fl.getDocumentDOM().library.items;
```

library.moveToFolder()

Disponibilidad

Flash MX 2004

Uso

```
library.moveToFolder(folderPath [, itemToMove [, bReplace]])
```

Parámetros

`folderPath` Una cadena que especifica la ruta de la carpeta con la forma `"FolderName"` o `"FolderName/FolderName"`. Para desplazar un elemento al nivel superior, especifique una cadena vacía ("") para *folderPath*.

`itemToMove` Una cadena que especifica el nombre del elemento que se va a mover. Si no especifica *itemToMove*, se desplazarán los elementos seleccionados actualmente. Este parámetro es opcional.

`bReplace` Un valor booleano. Si ya existe un elemento con el mismo nombre y especifica `true` para el parámetro *bReplace*, se sustituirá el elemento existente por el elemento que se está desplazando. Si es `false`, el nombre del elemento colocado cambiará a un nombre exclusivo. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto

Valor booleano: `true` si el elemento se mueve correctamente; `false` en caso contrario.

Descripción

Método; desplaza el elemento de biblioteca seleccionado actualmente o especificado a una carpeta especificada. Si el parámetro *folderPath* está vacío, los elementos se desplazan al nivel superior.

Ejemplo

El ejemplo siguiente desplaza el elemento `Symbol_1` a la carpeta de biblioteca `new` y reemplaza el elemento en esa carpeta con el mismo nombre:

```
fl.getDocumentDOM().library.moveToFolder("new", "Symbol_1", true);
```

library.newFolder()

Disponibilidad

Flash MX 2004

Uso

```
library.newFolder([folderPath])
```

Parámetros

`folderPath` Una cadena que especifica el nombre de la carpeta que se va a crear. Si se especifica como una ruta y la ruta no existe, se creará la ruta. Este parámetro es opcional.

Valor devuelto

Valor booleano: `true` si la carpeta se crea correctamente; `false` en caso contrario.

Descripción

Método; crea una carpeta con el nombre especificado o un nombre predeterminado en la carpeta seleccionada (`"untitled folder #"`) si no se suministra ningún parámetro *folderName*.

Ejemplo

El ejemplo siguiente crea dos nuevas carpetas de biblioteca, la segunda como subcarpeta de la primera:

```
fl.getDocumentDOM().library.newFolder("first/second");
```

library.renameItem()

Disponibilidad
Flash MX 2004

Uso
`library.renameItem(name)`

Parámetros
`name` Una cadena que especifica un nombre nueva para el elemento de biblioteca.

Valor devuelto
Un valor booleano de `true` si el nombre del elemento se cambia correctamente, `false` en caso contrario. Si hay varios elementos seleccionados, los nombres no cambiarán y el valor devuelto será `false` (de acuerdo con el comportamiento de la interfaz de usuario).

Descripción
Método; cambia el nombre del elemento de biblioteca seleccionado actualmente en el panel Biblioteca.

Ejemplo
El ejemplo siguiente cambia el nombre del elemento de biblioteca seleccionado por `"new name"`:

```
fl.getDocumentDOM().library.renameItem("new name");
```

library.selectAll()

Disponibilidad
Flash MX 2004

Uso
`library.selectAll([bSelectAll])`

Parámetros
`bSelectAll` Un valor booleano que especifica si se selecciona o anula la selección de todos los elementos de la biblioteca. Omita este parámetro o utilice el valor predeterminado de `true` para seleccionar todos los elementos de la biblioteca; `false` anula la selección de todos los elementos de biblioteca. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; selecciona o anula la selección de todos los elementos de la biblioteca.

Ejemplo
Los ejemplos siguientes seleccionan todos los elementos de la biblioteca:

```
fl.getDocumentDOM().library.selectAll();
fl.getDocumentDOM().library.selectAll(true);
```

Los ejemplos siguientes anulan la selección de todos los elementos de la biblioteca:

```
fl.getDocumentDOM().library.selectAll(false);
fl.getDocumentDOM().library.selectNone();
```

library.selectItem()

Disponibilidad

Flash MX 2004

Uso

```
library.selectItem(namePath [, bReplaceCurrentSelection [, bSelect]])
```

Parámetros

`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, puede especificar su nombre y su ruta empleando notación con barras.

`bReplaceCurrentSelection` Un valor booleano que especifica si se va a reemplazar la selección actual o si se va a añadir el elemento a la selección actual. El valor predeterminado es `true` (reemplaza la selección actual). Este parámetro es opcional.

`bSelect` Un valor booleano que especifica si se va a seleccionar o anular la selección de un elemento. El valor predeterminado es `true` (seleccionar). Este parámetro es opcional.

Valor devuelto

Un valor booleano; `true` si existe el elemento especificado; en caso contrario, `false`.

Descripción

Método; selecciona un elemento de biblioteca especificado.

Ejemplo

El ejemplo siguiente cambia la selección actual de la biblioteca a `Symbol_1` dentro de `untitled Folder_1`:

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1");
```

El ejemplo siguiente amplía lo que se encuentra seleccionado actualmente en la biblioteca para incluir `Symbol_1` dentro de `untitled Folder_1`:

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1", false);
```

El ejemplo siguiente anula la selección de `Symbol_1` dentro de `untitled Folder_1` y no cambia otros elementos seleccionados:

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1", true, false);
```

library.selectNone()

Disponibilidad
Flash MX 2004

Uso
```
library.selectNone()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; anula la selección de todos los elementos de biblioteca.

Ejemplo
Los ejemplos siguientes anulan la selección de todos los elementos de la biblioteca:

```
fl.getDocumentDOM().library.selectNone();
fl.getDocumentDOM().library.selectAll(false);
```

library.setItemProperty()

Disponibilidad
Flash MX 2004

Uso
```
library.setItemProperty(property, value)
```

Parámetros
`property` Una cadena que es el nombre de la propiedad que se va a definir. Para obtener una lista de propiedades, consulte la tabla de resumen de propiedades del Objeto Item y los resúmenes de propiedades de sus subclases. Para ver qué objetos son subclases del objeto Item, consulte "Resumen de la estructura del DOM" en la página 12.

`value` El valor que se va a asignar a la propiedad especificada.

Valor devuelto
Ninguno.

Descripción
Método; define la propiedad de todos los elementos de biblioteca seleccionados (ignorando las carpetas).

Ejemplo

El ejemplo siguiente asigna el valor botón a la propiedad `symbolType` para el elemento o elementos de biblioteca seleccionados. En este caso, el elemento debe ser un Objeto SymbolItem; `symbolType` es una propiedad válida para objetos SymbolItem.

```
fl.getDocumentDOM().library.setItemProperty("symbolType", "button");
```

library.updateItem()

Disponibilidad

Flash MX 2004

Uso

```
library.updateItem([namePath])
```

Parámetros

`namePath` Una cadena que especifica el nombre del elemento. Si el elemento se encuentra en una carpeta, especifique su nombre y su ruta empleando la notación con barras. En la interfaz de usuario equivale a hacer clic con el botón derecho del ratón en un elemento y seleccionar Actualizar en el menú. Si no se suministra ningún nombre, se actualizará la selección actual. Este parámetro es opcional.

Valor devuelto

Un valor booleano: `true` si Flash actualiza el elemento correctamente; `false` en caso contrario.

Descripción

Método; actualiza el elemento especificado.

Ejemplo

El ejemplo siguiente muestra un cuadro de diálogo que indica si el elemento seleccionado se actualiza (`true`) o no (`false`):

```
alert(fl.getDocumentDOM().library.updateItem());
```

Capítulo 27: Objeto Math

Disponibilidad
Flash MX 2004

Descripción
El objeto Math está disponible como propiedad de sólo lectura del objeto Flash; consulte `fl.Math`. Este objeto proporciona métodos que realizan operaciones matemáticas comunes.

Resumen de métodos
Los métodos siguientes están disponibles para el objeto Math:

Método	Descripción
`Math.concatMatrix()`	Realiza una concatenación de matrices y devuelve el resultado.
`Math.invertMatrix()`	Devuelve la inversa de la matriz especificada.
`Math.pointDistance()`	Calcula la distancia entre dos puntos.

Math.concatMatrix()

Disponibilidad
Flash MX 2004

Uso
```
Math.concatMatrix(mat1, mat2)
```

Parámetros
mat1, mat2 Especifica los objetos Matrix que se van a concatenar (consulte Objeto Matrix). Cada parámetro debe ser un objeto con campos `a`, `b`, `c`, `d`, `tx` y `ty`.

Valor devuelto
Una matriz de objetos concatenados.

Descripción
Método; realiza una concatenación de matrices y devuelve el resultado.

Ejemplo
El ejemplo siguiente almacena el objeto seleccionado actualmente en la variable `elt`, multiplica la matriz de objetos por la matriz de vistas y almacena ese valora en la variable `mat`:

```
var elt = fl.getDocumentDOM().selection[0];
var mat = fl.Math.concatMatrix( elt.matrix , fl.getDocumentDOM().viewMatrix );
```

Math.invertMatrix()

Disponibilidad
Flash MX 2004

Uso
`Math.invertMatrix(mat)`

Parámetros
`mat` Indica el objeto Matrix que se va a invertir (consulte Objeto Matrix). Debe tener los siguientes campos: `a`, `b`, `c`, `d`, `tx` y `ty`.

Valor devuelto
Un objeto Matrix que es la inversa de la matriz original.

Descripción
Método; devuelve la inversa de la matriz especificada.

Ejemplo
El ejemplo siguiente almacena el objeto seleccionado en la variable `elt`, asigna esa matriz a la variable `mat` y almacena la inversa de la matriz en la variable `inv`:

```
var elt = fl.getDocumentDOM().selection[0];
var mat = elt.matrix;
var inv = fl.Math.invertMatrix( mat );
```

Math.pointDistance()

Disponibilidad
Flash MX 2004

Uso
`Math.pointDistance(pt1, pt2)`

Parámetros
pt1, pt2 Especifica los puntos entre los que se mide la distancia.

Valor devuelto
Un valor de coma flotante que representa la distancia entre los puntos.

Descripción
Método; calcula la distancia entre dos puntos.

Ejemplo

El ejemplo siguiente almacena el valor de la distancia entre *pt1* y *pt2* en la variable `dist`:

```
var pt1 = {x:10, y:20}
var pt2 = {x:100, y:200}
var dist = fl.Math.pointDistance(pt1, pt2);
```

Capítulo 28: Objeto Matrix

Disponibilidad
Flash MX 2004

Descripción
El objeto Matrix representa una matriz de transformación.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto Matrix:

Propiedad	Descripción
`matrix.a`	Un valor de coma flotante que especifica el elemento (0,0) de la matriz de transformación.
`matrix.b`	Un valor de coma flotante que especifica el elemento (0,1) de la matriz.
`matrix.c`	Un valor de coma flotante que especifica el elemento (1,0) de la matriz.
`matrix.d`	Un valor de coma flotante que especifica el elemento (1,1) de la matriz.
`matrix.tx`	Un valor de coma flotante que especifica la ubicación en el eje *x* del punto de registro de un símbolo o el centro de una forma.
`matrix.ty`	Un valor de coma flotante que especifica la ubicación en el eje *y* del punto de registro de un símbolo o el centro de una forma.

matrix.a

Disponibilidad
Flash MX 2004

Uso
`matrix.a`

Descripción
Propiedad; un valor de coma flotante que especifica el elemento (0,0) de la matriz de transformación. Este valor representa el factor de escala del eje *x* del objeto.

Ejemplo
Las propiedades `a` y `d` de una matriz representan el escalado. En el ejemplo siguiente, los valore se definen como 2 y 3, respectivamente, para aumentar el objeto seleccionado dos veces su anchura y tres veces su altura:

```
var mat = fl.getDocumentDOM().selection[0].matrix;
mat.a = 2;
mat.d = 3;
fl.getDocumentDOM().selection[0].matrix = mat;
```

Puede girar un objeto definiendo las propiedades de matriz `a`, `b`, `c` y `d` en relación mutua, donde `a = d` y `b = c`. Por ejemplo, los valores de 0,5, 0,8, -0,8 y 0,5 giran el objeto 60º:

```
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;
```

Puede definir `a = d = 1` and `c = b = 0` para devolver la forma original al objeto.

matrix.b

Disponibilidad

Flash MX 2004

Uso

`matrix.b`

Descripción

Propiedad; un valor de coma flotante que especifica el elemento (0,1) de la matriz. Este valor representa el sesgo vertical de una forma; hace que Flash desplace el borde derecho de la forma a lo largo del eje vertical.

Las propiedades `matrix.b` y `matrix.c` de una matriz representan el sesgo (consulte `matrix.c`).

Ejemplo

En el ejemplo siguiente, puede definir `b` y `c` como -1 y 0 respectivamente; estos valores sesgan el objeto en un ángulo vertical de 45º:

```
var mat = fl.getDocumentDOM().selection[0].matrix;
mat.b = -1;
mat.c = 0;
fl.getDocumentDOM().selection[0].matrix = mat;
```

Para sesgar el objeto en su forma original, puede definir `b` y `c` como 0.

Consulte el ejemplo de `matrix.a`.

matrix.c

Disponibilidad

Flash MX 2004

Uso

`matrix.c`

Descripción

Propiedad; un valor de coma flotante que especifica el elemento (1,0) de la matriz. Este valor hace que Flash sesgue el objeto desplazando su borde inferior a lo largo del eje horizontal.

Las propiedades `matrix.b` y `matrix.c` de una matriz representan el sesgo.

Ejemplo
Consulte el ejemplo de `matrix.b`.

matrix.d

Disponibilidad
Flash MX 2004

Uso
`matrix.d`

Descripción
Propiedad; un valor de coma flotante que especifica el elemento (1,1) de la matriz. Este valor representa el factor de escala del eje *y* del objeto.

Ejemplo
Consulte el ejemplo de `matrix.a`.

matrix.tx

Disponibilidad
Flash MX 2004

Uso
`matrix.tx`

Descripción
Propiedad; un valor de coma flotante que especifica la ubicación en el eje *x* del punto de registro de un símbolo (también *punto de origen* o *punto cero*) o el centro de una forma. Define la traducción de *x* de la transformación.

Puede desplazar un objeto definiendo las propiedades `matrix.tx` y `matrix.ty` (consulte `matrix.ty`).

Ejemplo
En el ejemplo siguiente, si define `tx` y `ty` como 0 se desplazará el punto de registro del objeto al punto 0,0 del documento:

```
var mat = fl.getDocumentDOM().selection[0].matrix;
mat.tx = 0;
mat.ty = 0;
fl.getDocumentDOM().selection[0].matrix = mat;
```

matrix.ty

Disponibilidad

Flash MX 2004

Uso

`matrix.ty`

Descripción

Propiedad; un valor de coma flotante que especifica la ubicación en el eje *y* del punto de registro de un símbolo o el centro de una forma. Define la traducción de *y* de la transformación.

Puede desplazar un objeto definiendo las propiedades `matrix.tx` y `matrix.ty`.

Ejemplo

Consulte el ejemplo de `matrix.tx`.

Capítulo 29: Objeto outputPanel

Disponibilidad
Flash MX 2004

Descripción
Este objeto representa el panel Salida, que muestra información de resolución de problemas, como errores de sintaxis. Para acceder a este objeto, utilice `fl.outputPanel` (o `flash.outputPanel`). Consulte `fl.outputPanel`.

Resumen de métodos
El objeto outputPanel utiliza los métodos siguientes:

Método	Descripción
`outputPanel.clear()`	Borra el contenido del panel Salida.
`outputPanel.save()`	Guarda el contenido del panel Salida en un archivo de texto local.
`outputPanel.trace()`	Añade una línea al contenido del panel Salida, terminada con una línea nueva.

outputPanel.clear()

Disponibilidad
Flash MX 2004

Uso
```
outputPanel.clear()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; borra el contenido del panel Salida. Puede utilizar este método en una aplicación de procesamiento por lotes para borrar una lista de errores o para guardarlos incrementalmente empleando este método con `outputPanel.save()`.

Ejemplo
El ejemplo siguiente borra el contenido actual del panel Salida:

```
fl.outputPanel.clear();
```

outputPanel.save()

Disponibilidad

Flash MX 2004; parámetro *bUseSystemEncoding* añadido en Flash 8.

Uso

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

Parámetros

fileURI Una cadena, expresada como URI file:///, que especifica el archivo local que albergará el contenido del panel Salida.

bAppendToFile Un valor booleano opcional. Si es `true`, añade el contenido del panel Salida al archivo de salida, y si es `false`, el método sobrescribe el archivo de salida si ya existe. El valor predeterminado es `false`.

bUseSystemEncoding Un valor booleano opcional. Si es `true`, guarda el texto del panel Salida con la codificación del sistema; si es `false`, guarda el texto del panel de salida con la codificación UTF-8, con los caracteres de marca de orden de byte al inicio del texto. El valor predeterminado es `false`.

Valor devuelto

Ninguno.

Descripción

Método; guarda el contenido del panel Salida en un archivo de texto local, bien sobrescribiendo el archivo o añadiéndolo.

Si *fileURI* no es válido o no se especifica, se produce un error.

Este método resulta útil para procesamiento por lotes. Por ejemplo, puede crear un archivo JSFL que compile varios componentes. Los errores de compilación aparecerán en el panel Salida y podrá utilizar este método para guardar los errores resultantes en un archivo de texto que el sistema de compilación utilizado puede analizar automáticamente.

Ejemplo

El ejemplo siguiente guarda el contenido del panel Salida en un archivo batch.log en la carpeta /tests, sobrescribiendo el archivo batch.log si ya existe:

```
fl.outputPanel.save("file:///c|/tests/batch.log");
```

outputPanel.trace()

Disponibilidad

Flash MX 2004

Uso

```
outputPanel.trace(message)
```

Parámetros

message Una cadena que contiene el texto que se va a añadir al panel Salida.

Valor devuelto

Ninguno.

Descripción

Método; envía una cadena de texto al panel Salida, terminada con una línea nueva y muestra el panel Salida si aún no está visible. Este método es idéntico a `fl.trace()` y funciona igual que la declaración `trace()` en ActionScript.

Para enviar una línea en blanco, utilice `outputPanel.trace("")` o `outputPanel.trace("\n")`. Para utilizar estos comandos en línea, convierta `\n` en parte de la cadena *message*.

Ejemplo

El ejemplo siguiente muestra varias líneas de texto en el panel Salida:

```
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 +"?");
```

Capítulo 30: Objeto Oval

Herencia Objeto Element > Objeto Shape > Objeto Oval

Disponibilidad
Flash CS3 Professional

Descripción
El objeto Oval es una forma que se dibuja con la herramienta Óvalo simple. Para determinar si un elemento es un objeto Rectangle, utilice `shape.isOvalObject`.

Resumen de propiedades
Además de las propiedades del Objeto Shape, puede utilizar las siguientes propiedades con el objeto Oval. Para establecer las propiedades de un objeto Oval, utilice document.setOvalObjectProperty().

Propiedad	Descripción
`OvalObject.closePath`	De sólo lectura; un valor booleano que especifica si está seleccionado el cuadro de diálogo Cerrar trazado en el inspector de propiedades.
`OvalObject.endAngle`	De sólo lectura; un valor flotante que especifica el ángulo final del objeto Oval.
`OvalObject.innerRadius`	De sólo lectura; un valor flotante que especifica el radio interior del objeto Oval con un porcentaje.
`OvalObject.startAngle`	De sólo lectura; un valor flotante que especifica el ángulo inicial del objeto Oval.

OvalObject.closePath

Disponibilidad
Flash CS3 Professional

Uso
`OvalObject.closePath`

Descripción
Propiedad de sólo lectura; un valor booleano que especifica si está seleccionado el cuadro de diálogo Cerrar trazado en el inspector de propiedades. Si los valores de los ángulos inicial y final del objeto son iguales, la configuración de esta propiedad no tendrá ningún efecto hasta que se cambien los valores.

Para establecer este valor, utilice `document.setOvalObjectProperty()`.

Ejemplo
El ejemplo siguiente anula la selección de la propiedad `OvalObject.closePath`:

```
fl.getDocumentDOM().setOvalObjectProperty("closePath",false);
```

Véase también

`document.setOvalObjectProperty()`, `shape.isOvalObject`

OvalObject.endAngle

Disponibilidad

Flash CS3 Professional

Uso

```
OvalObject.endAngle
```

Descripción

Propiedad de sólo lectura; un valor flotante que especifica el ángulo final del objeto Oval. Los valores aceptados van de 0 a 360.

Para establecer este valor, utilice `document.setOvalObjectProperty()`.

Ejemplo

El ejemplo siguiente establece en 270 el ángulo final de los objetos Oval seleccionados.

```
fl.getDocumentDOM().setOvalObjectProperty("endAngle",270);
```

Véase también

`document.setOvalObjectProperty()`, `OvalObject.startAngle`, `shape.isOvalObject`

OvalObject.innerRadius

Disponibilidad

Flash CS3 Professional

Uso

```
OvalObject.innerRadius
```

Descripción

Propiedad de sólo lectura; un valor flotante que especifica el radio interior del objeto Oval con un porcentaje. Los valores aceptados van de 0 a 99.

Para establecer este valor, utilice `document.setOvalObjectProperty()`.

Ejemplo

El ejemplo siguiente establece en 50% el radio interior de los objetos Oval seleccionados:

```
fl.getDocumentDOM().setOvalObjectProperty("innerRadius",50);
```

Véase también

`document.setOvalObjectProperty()`, `shape.isOvalObject`

OvalObject.startAngle

Disponibilidad

Flash CS3 Professional

Uso

`OvalObject.startAngle`

Descripción

Propiedad de sólo lectura; un valor flotante que especifica el ángulo inicial del objeto Oval. Los valores aceptados van de 0 a 360.

Para establecer este valor, utilice `document.setOvalObjectProperty()`.

Ejemplo

El ejemplo siguiente establece en 270 el ángulo inicial de los objetos Oval seleccionados:

```
fl.getDocumentDOM().setOvalObjectProperty("startAngle",270);
```

Véase también

`document.setOvalObjectProperty()`, `OvalObject.endAngle`, `shape.isOvalObject`

Capítulo 31: Objeto Parameter

Disponibilidad

Flash MX 2004

Descripción

El acceso al tipo de objeto Parameter se realiza desde el conjunto `screen.parameters` (que corresponde al inspector de propiedades de la pantalla en la herramienta de edición de Flash) o el conjunto `componentInstance.parameters` (que corresponde al inspector de propiedades del componente en la herramienta de edición).

Resumen de métodos

Los métodos siguientes están disponibles para el objeto Parameter:

Método	Descripción
`parameter.insertItem()`	Inserta un elemento en la lista, objeto o conjunto.
`parameter.removeItem()`	Suprime un elemento del tipo de lista, objeto o conjunto de un parámetro de pantalla o componente.

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto Parameter:

Propiedad	Descripción
`parameter.category`	Una cadena que especifica la propiedad `category` para el parámetro `screen` o el parámetro `componentInstance`.
`parameter.listIndex`	Un entero que especifica el valor del elemento de lista seleccionado.
`parameter.name`	De sólo lectura; una cadena que especifica el nombre del parámetro.
`parameter.value`	Corresponde al campo Valor de la ficha Parámetros del panel Inspector de componentes, la ficha Parámetros del inspector de propiedades o el inspector de propiedades de la pantalla.
`parameter.valueType`	De sólo lectura; una cadena que indica el tipo de parámetro de pantalla o componente.
`parameter.verbose`	Especifica dónde se muestra el parámetro.

parameter.category

Disponibilidad

Flash MX 2004

Uso

```
parameter.category
```

Descripción
Propiedad; una cadena que especifica la propiedad `category` para el parámetro `screen` o el parámetro `componentInstance`. Esta propiedad ofrece una forma alternativa de presentar una lista de parámetros. Esta funcionalidad no está disponible a través de la interfaz de usuario de Flash.

parameter.insertItem()

Disponibilidad
Flash MX 2004

Uso
```
parameter.insertItem(index, name, value, type)
```

Parámetros
`index` Un índice entero basado en cero que indica dónde se insertará el elemento en la lista, objeto o conjunto. Si el índice es 0, el elemento se insertará al principio de la lista. Si el índice es mayor que el tamaño de lista, el nuevo elemento se insertará al final del conjunto.

`name` Una cadena que especifica el nombre del elemento que se va a insertar. Es un parámetro necesario para los parámetros de objeto.

`value` Una cadena que especifica el valor del elemento que se va a insertar.

`type` Una cadena que especifica el tipo de elemento que se va a insertar.

Valor devuelto
Ninguno.

Descripción
Método; inserta un elemento en la lista, objeto o conjunto. Si un parámetro es una lista, objeto o conjunto, la propiedad *value* será un conjunto.

Ejemplo
El ejemplo siguiente inserta el valor de `New Value` en el parámetro `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

Disponibilidad
Flash MX 2004

Uso

```
parameter.listIndex
```

Descripción

Propiedad; el valor del elemento de lista seleccionado. Esta propiedad es válida únicamente si `parameter.valueType` es `"List"`.

Ejemplo

El ejemplo siguiente define el primer parámetro para una diapositiva, que es el parámetro `autoKeyNav`. Para definir el parámetro con uno de sus valores válidos (`true`, `false` o `inherit`) `parameter.listIndex` se define con el índice del elemento de la lista (0 para `true`, 1 para `false`, 2 para `inherit`).

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;
parms[0].listIndex = 1;
```

parameter.name

Disponibilidad

Flash MX 2004

Uso

```
parameter.name
```

Descripción

Propiedad de sólo lectura; una cadena que especifica el nombre del parámetro.

Ejemplo

El ejemplo siguiente muestra el nombre del quinto parámetro para el componente seleccionado:

```
var parms = fl.getDocumentDOM().selection[0].parameters;
fl.trace("name: " + parms[4].name);
```

El ejemplo siguiente muestra el nombre del quinto parámetro para la pantalla especificada:

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters; fl.trace("name: " +
parms[4].name);
```

parameter.removeItem()

Disponibilidad

Flash MX 2004

Uso

```
parameter.removeItem(index)
```

Parámetros

index El índice entero basado en cero del elemento que se va a eliminar de la propiedad de pantalla o componente.

Valor devuelto
Ninguno.

Descripción
Método; suprime un elemento del tipo de lista, objeto o conjunto de un parámetro de pantalla o componente.

Ejemplo
El ejemplo siguiente suprime el elemento en el índice 1 del parámetro `labelPlacement` de un componente:

```
// 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);
}
```

El ejemplo siguiente elimina el elemento en el índice 1 del parámetro `autoKeyNav` de una pantalla:

```
// 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

Disponibilidad
Flash MX 2004

Uso
`parameter.value`

Descripción

Propiedad; corresponde al campo Valor de la ficha Parámetros del panel Inspector de componentes, la ficha Parámetros del inspector de propiedades o el inspector de propiedades de la pantalla. El tipo de la propiedad `value` está determinado por la propiedad `valueType` del parámetro (consulte `parameter.valueType`).

parameter.valueType

Disponibilidad

Flash MX 2004

Uso

```
parameter.valueType
```

Descripción

Propiedad de sólo lectura; una cadena que indica el tipo de parámetro de pantalla o componente. El tipo puede ser uno de los siguientes valores: `"Default"`, `"Array"`, `"Object"`, `"List"`, `"String"`, `"Number"`, `"Boolean"`, `"Font Name"`, `"Color"`, `"Collection"`, `"Web Service URL"` o `"Web Service Operation"`.

Véase también

`parameter.value`

parameter.verbose

Disponibilidad

Flash MX 2004

Uso

```
parameter.verbose
```

Descripción

Propiedad; especifica dónde se muestra el parámetro. Si el valor de esta propiedad es 0 (no detallado), el parámetro sólo se muestra en el inspector de componentes. Si es 1 (detallado), el parámetro se muestra en el inspector de componentes y en la ficha Parámetros del inspector de propiedades.

Capítulo 32: Objeto Path

Disponibilidad
Flash MX 2004

Descripción
El objeto Path define una secuencia de segmentos de línea (recta, curva o ambas) que suele emplearse para crear herramientas ampliables. El ejemplo siguiente muestra una instancia de un objeto Path que devuelve el objeto flash:

```
path = fl.drawingLayer.newPath();
```

Véase también Objeto drawingLayer.

Resumen de métodos
Los métodos siguientes están disponibles para el objeto Path:

Método	Descripción
`path.addCubicCurve()`	Añade un segmento de curva cúbica de Bézier al trazado.
`path.addCurve()`	Añade un segmento cuadrático de Bézier al trazado.
`path.addPoint()`	Añade un punto al trazado.
`path.clear()`	Elimina todos los puntos del trazado.
`path.close()`	Añade un punto en la ubicación del primer punto del trazado y lo extiende a ese punto para cerrarlo.
`path.makeShape()`	Crea una forma en el escenario utilizando la configuración actual de trazo y relleno.
`path.newContour()`	Inicia un nuevo contorno en el trazado.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto Path:

Propiedad	Descripción
`path.nPts`	De sólo lectura; un entero que representa el número de puntos del trazado.

path.addCubicCurve()

Disponibilidad
Flash MX 2004

Uso

```
path.addCubicCurve(xAnchor, yAnchor, x2, y2, x3, y3, x4, y4)
```

Parámetros
`xAnchor` Un número de coma flotante que especifica la posición *x* del primer punto de control.

yAnchor Un número de coma flotante que especifica la posición *y* del primer punto de control.

x2 Un número de coma flotante que especifica la posición *x* del segundo punto de control.

y2 Un número de coma flotante que especifica la posición *y* del segundo punto de control.

x3 Un número de coma flotante que especifica la posición *x* del tercer punto de control.

y3 Un número de coma flotante que especifica la posición *y* del tercer punto de control.

x4 Un número de coma flotante que especifica la posición *x* del cuarto punto de control.

y4 Un número de coma flotante que especifica la posición *y* del cuarto punto de control.

Valor devuelto
Ninguno.

Descripción
Método; añade un segmento de curva cúbica de Bézier al trazado.

Ejemplo
El ejemplo siguiente crea un trazado nuevo, lo almacena en la variable `myPath` y asigna la curva al trazado:

```
var myPath = fl.drawingLayer.newPath();
myPath.addCubicCurve(0, 0, 10, 20, 20, 20, 30, 0);
```

path.addCurve()

Disponibilidad
Flash MX 2004

Uso
```
path.addCurve(xAnchor, yAnchor, x2, y2, x3, y3)
```

Parámetros
xAnchor Un número de coma flotante que especifica la posición *x* del primer punto de control.

yAnchor Un número de coma flotante que especifica la posición *y* del primer punto de control.

x2 Un número de coma flotante que especifica la posición *x* del segundo punto de control.

y2 Un número de coma flotante que especifica la posición *y* del segundo punto de control.

x3 Un número de coma flotante que especifica la posición *x* del tercer punto de control.

y3 Un número de coma flotante que especifica la posición *y* del tercer punto de control.

Valor devuelto
Ninguno.

Descripción
Método; añade un segmento cuadrático de Bézier al trazado.

Ejemplo
El ejemplo siguiente crea un trazado nuevo, lo almacena en la variable `myPath` y asigna la curva al trazado:

```
var myPath = fl.drawingLayer.newPath();
myPath.addCurve(0, 0, 10, 20, 20, 0);
```

path.addPoint()

Disponibilidad
Flash MX 2004

Uso
```
path.addPoint(x, y)
```

Parámetros
x Un número de coma flotante que especifica la posición *x* del punto.

y Un número de coma flotante que especifica la posición *y* del punto.

Valor devuelto
Ninguno.

Descripción
Método; añade un punto al trazado.

Ejemplo
El ejemplo siguiente crea un trazado nuevo, lo almacena en la variable `myPath` y asigna el nuevo punto al trazado:

```
var myPath = fl.drawingLayer.newPath();
myPath.addPoint(10, 100);
```

path.clear()

Disponibilidad
Flash MX 2004

Uso
```
path.clear()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; elimina todos los puntos del trazado.

Ejemplo
El ejemplo siguiente suprime todos los puntos de un trazado almacenado en la variable `myPath`:

```
var myPath = fl.drawingLayer.newPath();
myPath.clear();
```

path.close()

Disponibilidad
Flash MX 2004

Uso
```
path.close()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; añade un punto en la ubicación del primer punto del trazado y lo extiende a ese punto para cerrarlo. Si el trazado no tiene puntos, no se añadirán puntos.

Ejemplo
El ejemplo siguiente crea un trazado cerrado:

```
var myPath = fl.drawingLayer.newPath();
myPath.close();
```

path.makeShape()

Disponibilidad
Flash MX 2004

Uso
```
path.makeShape([bSupressFill [, bSupressStroke]])
```

Parámetros
`bSuppressFill` Un valor booleano que, si se establece como `true`, anula el relleno que se aplicaría a la forma. El valor predeterminado es `false`. Este parámetro es opcional.

bSupressStroke Un valor booleano que, si se establece como `true`, anula el trazo que se aplicaría a la forma. El valor predeterminado es `false`. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; crea una forma en el escenario utilizando la configuración actual de trazo y relleno. El trazado se borra cuando se crea la forma. Este método tiene dos parámetros opcionales para suprimir el relleno y el trazo del objeto de forma resultante. Si omite estos parámetros o los define como `false` se emplearán los valores actuales de relleno y trazo.

Ejemplo
El ejemplo siguiente crea una forma con el relleno actual y sin trazo:

```
var myPath = fl.drawingLayer.newPath();
myPath.makeShape(false, true);
```

path.newContour()

Disponibilidad
Flash MX 2004

Uso
```
path.newContour()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; inicia un nuevo contorno en el trazado.

Ejemplo
El ejemplo siguiente crea un cuadrado vacío:

```
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

Disponibilidad

Flash MX 2004

Uso

`path.nPts`

Descripción

Propiedad de sólo lectura; un entero que representa el número de puntos del trazado. Una trazado nuevo tiene 0 puntos.

Ejemplo

El ejemplo siguiente utiliza el panel Salida para mostrar el número de puntos del trazado al que hace referencia 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
```

Capítulo 33: Objeto presetItem

Disponibilidad
Flash CS4 Professional

Descripción
El objeto presetItem representa un elemento (valor predefinido o carpeta) en el panel Configuración predefinida de movimiento (Ventana > Configuración predefinida de movimiento). El conjunto de objetos presetItem es una propiedad del objeto presetPanel (`presetPanel.items`).

Todas las propiedades del objeto presetItem son de sólo lectura. Para realizar tareas, como eliminación, cambio de nombre o desplazamiento de elementos, utilice los métodos del Objeto presetPanel.

Resumen de propiedades
Puede emplear las siguientes propiedades con el objeto presetItem:

Propiedad	Descripción
presetItem.isDefault	Especifica si el elemento está instalado junto con Flash o es un elemento personalizado que usted u otro usuario ha creado.
presetItem.isFolder	Especifica si el elemento del panel Configuración predefinida de movimiento es una carpeta o un valor predefinido.
presetItem.level	El nivel del elemento en la estructura de carpetas del panel Configuración predefinida de movimiento.
presetItem.name	Nombre del valor predefinido o carpeta, sin información de la ruta.
presetItem.open	Especifica si una carpeta del panel Configuración predefinida de movimiento está expandida actualmente.
presetItem.path	La ruta del elemento en el árbol de carpetas del panel Configuración predefinida de movimiento y el nombre del elemento.

presetItem.isDefault

Disponibilidad
Flash CS4 Professional

Uso
```
presetItem.isDefault
```

Descripción
Propiedad de sólo lectura; un valor booleano que especifica si el elemento está instalado con Flash (`true`) o si se trata de un elemento personalizado creado por otro usuario o por usted mismo (`false`). Si este valor es `true`, puede considerarse un elemento de “sólo lectura”; no es posible moverlo, eliminarlo ni aplicarle operaciones similares.

Ejemplo
El siguiente ejemplo muestra el contenido del panel Configuración predefinida de movimiento e indica si un elemento está instalado con 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

Disponibilidad
Flash CS4 Professional

Uso
`presetItem.isFolder`

Descripción
Propiedad de sólo lectura; un valor booleano que especifica si el elemento del panel Configuración predefinida de movimiento es una carpeta (`true`) o un valor predefinido (`false`).

Ejemplo
El siguiente ejemplo muestra que el primer elemento del panel Configuración predefinida de movimiento es una carpeta y el segundo, un valor predefinido:

```
var presetItemArray=fl.presetPanel.items;
fl.trace(presetItemArray[0].isFolder);
fl.trace(presetItemArray[1].isFolder);
```

presetItem.level

Disponibilidad
Flash CS4 Professional

Uso
`presetItem.level`

Descripción
Propiedad de sólo lectura; un entero que especifica el nivel del elemento en la estructura del panel Configuración predefinida de movimiento. La carpeta predefinida y la personalizada pertenecen al nivel 0.

Ejemplo
El siguiente ejemplo muestra que el primer elemento del panel Configuración predefinida de movimiento es de nivel 0 o de nivel 1:

```
var presetItemArray=fl.presetPanel.items;
fl.trace(presetItemArray[0].level);
fl.trace(presetItemArray[1].level);
```

presetItem.name

Disponibilidad
Flash CS4 Professional

Uso
`presetItem.name`

Descripción
Propiedad de sólo lectura; una cadena que representa el nombre de un valor predefinido o carpeta, sin información de ruta.

Ejemplo
Consulte `presetItem.path`.

presetItem.open

Disponibilidad
Flash CS4 Professional

Uso
`presetItem.open`

Descripción
Propiedad de sólo lectura: especifica si una carpeta del panel Configuración predefinida de movimiento está actualmente expandida (`true`) o no (`false`).

Esta propiedad es `true` si el elemento no es una carpeta. Para determinar si un elemento es una carpeta o un valor predefinido, utilice presetItem.isFolder.

Ejemplo
El siguiente ejemplo muestra información sobre el estado de expansión o contracción de las carpetas del panel Configuración predefinida de movimiento:

```
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

Disponibilidad

Flash CS4 Professional

Uso

`presetItem.path`

Descripción

Propiedad de sólo lectura; una cadena que representa la ruta del elemento en el árbol de carpetas del panel Configuración predefinida de movimiento y el nombre del elemento.

Ejemplo

El siguiente ejemplo ilustra la diferencia entre los valores de `presetItem.name` y `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("");
}
```

Capítulo 34: Objeto presetPanel

Disponibilidad

Flash CS4 Professional

Descripción

El objeto presetPanel representa el panel Configuración predefinida de movimiento (Ventana > Configuración predefinida de movimiento). Se trata de una propiedad del objeto flash (fl.presetPanel).

Resumen de métodos

Puede emplear los métodos siguientes con el objeto presetPanel:

Método	Descripción
`presetPanel.addNewItem()`	Si una sola interpolación de movimiento está seleccionada actualmente en el escenario, añade ese movimiento al panel Configuración predefinida de movimiento.
`presetPanel.applyPreset()`	Aplica el valor predefinido especificado o seleccionado actualmente al elemento seleccionado en el escenario.
`presetPanel.deleteFolder()`	Eliminar la carpeta especificada y todas sus subcarpetas del árbol de carpetas del panel Configuración predefinida de movimiento.
`presetPanel.deleteItem()`	Elimina el valor predefinido especificado del panel Configuración predefinida de movimiento.
`presetPanel.expandFolder()`	Expande o contrae la carpeta o carpetas seleccionadas actualmente en el panel Configuración predefinida de movimiento.
`presetPanel.exportItem()`	Exporta el valor predefinido actualmente seleccionado o especificado a un archivo XML.
`presetPanel.findItemIndex()`	Devuelve un entero que representa la posición del índice de un elemento en el panel Configuración predefinida de movimiento.
`presetPanel.getSelectedItems()`	Devuelve un conjunto de objetos presetItem correspondientes a los elementos seleccionados actualmente en el panel Configuración predefinida de movimiento.
`presetPanel.importItem()`	Añade un valor predefinido al panel Configuración predefinida de movimiento a partir de un archivo XML especificado.
`presetPanel.moveToFolder()`	Mueve el elemento especificado a la carpeta especificada.
`presetPanel.newFolder()`	Crea una carpeta en el árbol de carpetas del panel Configuración predefinida de movimiento.
`presetPanel.renameItem()`	Cambia el nombre del valor predefinido o la carpeta seleccionados por un nombre especificado.
`presetPanel.selectItem()`	Selecciona o anula la selección de un elemento en el panel Configuración predefinida de movimiento.

Resumen de propiedades

Puede emplear la siguiente propiedad con el objeto presetPanel:

Propiedad	Descripción
`presetPanel.items`	Un conjunto de objetos presetItem en el panel Configuración predefinida de movimiento.

presetPanel.addNewItem()

Disponibilidad

Flash CS4 Professional

Uso

```
fl.presetPanel.addNewItem( [namePath] );
```

Parámetros

`namePath` Una cadena que especifica la ruta y el nombre del elemento que se va a añadir al panel Configuración predefinida de movimiento. Este parámetro es opcional.

Valor devuelto

Un valor booleano `true` si se añade correctamente el elemento, y `false` en caso contrario.

Descripción

Método; si hay una sola interpolación de movimiento seleccionada actualmente en el escenario, añade dicho movimiento al panel Configuración predefinida de movimiento en la carpeta especificada y con el nombre indicado. La ruta especificada en *namePath* debe existir en el panel.

Si existe un *namePath* coincidente, el método no surte efecto alguno y devuelve `false`.

Si no se transfiere ningún valor para *namePath*, el elemento se añade a la carpeta Valores predefinidos personalizados con el nombre "Valor predefinido personalizado *n*," donde *n* se incrementa cada vez que se añade un elemento de este modo.

Ejemplo

Si damos por hecho que hay una sola interpolación de movimiento en el escenario, el código siguiente añade un valor predefinido denominado `Bouncing Ball` a la carpeta Valores predefinidos personalizados:

```
fl.presetPanel.addNewItem("Custom Presets/Bouncing Ball");
```

Véase también

presetPanel.newFolder()

presetPanel.applyPreset()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.applyPreset( [presetPath] )
```

Parámetros

`presetPath` Una cadena que especifica la ruta completa y el nombre del valor predefinido que se va a aplicar, tal como aparece en el panel Configuración predefinida de movimiento. Este parámetro es opcional; si no se transfiere ningún valor, se aplica el valor predefinido actualmente seleccionado.

Valor devuelto

Un valor booleano `true` si se aplica correctamente el valor predefinido, y `false` en caso contrario.

Descripción

Aplica el valor predefinido especificado o seleccionado actualmente al elemento seleccionado en el escenario. El elemento debe ser una interpolación de movimiento, un símbolo o un elemento que pueda convertirse en un símbolo. Si el elemento es una interpolación de movimiento, su movimiento actual se sustituye por el valor predefinido seleccionado sin necesidad de confirmación por parte del usuario.

Este método falla en los siguientes casos:

- La ruta especificada como *presetPath* no existe.
- No transfiere ningún valor para *presetPath* y no hay ningún valor predefinido seleccionado.
- No transfiere ningún valor para *presetPath* y hay varios valores predefinidos seleccionados.
- El elemento seleccionado en el escritorio no es un símbolo y no se puede convertir en un símbolo.

Ejemplo

El siguiente ejemplo aplica el valor predefinido `aDribble` al elemento seleccionado actualmente en el escenario:

```
var result = fl.presetPanel.applyPreset("Custom Presets/Bounces/aDribble");
fl.trace(result);
```

presetPanel.deleteFolder()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.deleteFolder( [folderPath])
```

Parámetros

`folderPath` Una cadena que especifica la carpeta que se va a eliminar del panel Configuración predefinida de movimiento. Este parámetro es opcional.

Valor devuelto

Un valor booleano `true` si la carpeta o carpetas se eliminan correctamente; `false` en caso contrario.

Descripción

Elimina la carpeta especificada y todas sus subcarpetas del árbol de carpetas del panel Configuración predefinida de movimiento. Se eliminan también todos los valores predefinidos de las carpetas. No es posible eliminar carpetas de la carpeta Valores predefinidos personalizados.

Si no transfiere un valor para *folderPath*, se eliminarán todas las carpetas seleccionadas actualmente.

Nota: *las carpetas se eliminan sin solicitar confirmación del usuario y la acción no se puede deshacer.*

Ejemplo

El siguiente código elimina una carpeta llamada `Bouncing` situada debajo de la carpeta Valores predefinidos personalizados; todas las subcarpetas de `Bouncing` también se eliminan:

```
fl.presetPanel.deleteFolder("Custom Presets/Bouncing");
```

Véase también

presetPanel.deleteItem()

presetPanel.deleteItem()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.deleteItem( [namePath] )
```

Parámetros

`namePath` Una cadena que especifica la ruta y el nombre del elemento que se va a eliminar del panel Configuración predefinida de movimiento. Este parámetro es opcional.

Valor devuelto

Un valor booleano `true` si el elemento o elementos se eliminan correctamente; `false` en caso contrario.

Descripción

Elimina el valor predefinido especificado del panel Configuración predefinida de movimiento. Si no transfiere un valor para *namePath*, se eliminarán todos los valores predefinidos seleccionados actualmente. No es posible eliminar elementos de la carpeta Valores predefinidos personalizados.

Nota: *los elementos se eliminan sin solicitar confirmación del usuario y la acción no se puede deshacer.*

Ejemplo

El siguiente código elimina un valor predefinido denominado `aDribble` de la carpeta Valores predefinidos personalizados:

```
fl.presetPanel.deleteItem("Custom Presets/aDribble");
```

Véase también

presetPanel.deleteFolder()

presetPanel.expandFolder()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.expandFolder( [bExpand [, bRecurse [, folderPath] ] ] )
```

Parámetros

`bExpand` Un valor booleano que especifica si se expande la carpeta (`true`) o si se contrae (`false`). Este parámetro es opcional; su valor predeterminado es `true`.

`bRecurse` Un valor booleano que especifica si se expanden o se contraen las subcarpetas de la carpeta (`true`) o no `false`). Este parámetro es opcional; su valor predeterminado es `false`.

`folderPath` Una cadena que especifica la ruta de la carpeta que se va a expandir o contraer. Este parámetro es opcional.

Valor devuelto

Un valor booleano `true` si la carpeta o carpetas se expanden o se contraen correctamente; `false` en caso contrario.

Descripción

Expande o contrae la carpeta o carpetas seleccionadas actualmente en el panel Configuración predefinida de movimiento. Para expandir o contraer carpetas distintas de las seleccionadas, transfiera un valor para *folderPath*.

Ejemplo

El siguiente ejemplo expande la carpeta Valores predefinidos personalizados, pero no sus subcarpetas:

```
fl.presetPanel.expandFolder(true, false, "Custom Presets");
```

El siguiente ejemplo expande la carpeta Valores predefinidos personalizados y todas sus subcarpetas:

```
fl.presetPanel.expandFolder(true, true, "Custom Presets");
```

presetPanel.exportItem()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.exportItem(fileURI [, namePath] )
```

Parámetros

`fileURI` Cadena que se expresa en forma de URI file:/// y que especifica la ruta y, opcionalmente, el nombre del archivo exportado. Consulte la descripción a continuación, para obtener más datos.

`namePath` Una cadena que especifica la ruta y el nombre del elemento que se va a seleccionar en el panel Configuración predefinida de movimiento. Este parámetro es opcional.

Valor devuelto

Un valor booleano `true` si se exporta correctamente el valor predefinido, y `false` en caso contrario.

Descripción

Exporta el valor predefinido actualmente seleccionado o especificado a un archivo XML. Sólo es posible exportar valores predefinidos; el método falla al intentar exportar una carpeta. Este método también falla al intentar sobrescribir un archivo en el disco.

Si no especifica un nombre de archivo como parte de *fileURI* (es decir, si el último carácter de *fileURI* es una barra inclinada (/)), el archivo exportado se guarda con el mismo nombre que el valor predefinido. Si no especifica un valor para *namePath*, se exporta el valor predefinido seleccionado actualmente. Consulte el ejemplo siguiente.

Ejemplo

El siguiente ejemplo muestra qué archivos se crean al transferir distintos parámetros a este método. También indica si el archivo especificado se ha creado correctamente. Antes de ejecutar este ejemplo, seleccione el valor predefinido "fly-in-left" en la carpeta Valores predefinidos personalizados y cree la carpeta `Mis valores predefinidos` en el disco.

```
//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");
```

Véase también

`presetPanel.importItem()`

presetPanel.findItemIndex()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.findItemIndex([presetName])
```

Parámetros

`presetName` Una cadena que especifica el nombre del valor predefinido para el que se devuelve el valor de índice. Este parámetro es opcional.

Valor devuelto

Un entero que representa el índice del valor predefinido especificado en el conjunto `presetPanel.items`. Si no transfiere un valor para *presetName*, se devuelve el índice del valor predefinido especificado actualmente. Este método devuelve -1 en los siguientes casos:

- No transfiere ningún valor para *presetName* y no hay ningún valor predefinido seleccionado.
- No transfiere ningún valor para *presetName* y hay varios valores predefinidos seleccionados.
- Ha transferido un valor para *presetName* que no corresponde a ningún elemento del panel.

Descripción

Devuelve un entero que representa la posición del índice de un elemento en el panel Configuración predefinida de movimiento.

Ejemplo

El siguiente código muestra el valor del índice y el nombre de ruta completo del valor predefinido seleccionado actualmente:

```
// 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()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.getSelectedItems()
```

Parámetros

Ninguno.

Valor devuelto

Un conjunto de objetos presetItem.

Descripción

Método; devuelve un conjunto de objetos presetItem correspondientes a los elementos seleccionados actualmente en el panel Configuración predefinida de movimiento (consulte Objeto presetItem). Cada elemento del conjunto representa una carpeta o un valor predefinido.

Ejemplo

El siguiente código muestra los nombres de ruta completos de los elementos seleccionados actualmente en el panel Configuración predefinida de movimiento:

```
var itemArray = fl.presetPanel.getSelectedItems();
var length =  itemArray.length
for (x=0; x<length; x++) {
    fl.trace(itemArray[x].path);
}
```

Véase también

presetPanel.items

presetPanel.importItem()

Disponibilidad
Flash CS4 Professional

Uso
`presetPanel.importItem(fileURI [,namePath ])`

Parámetros
`fileURI` Una cadena, expresada como URI file:///, que especifica el archivo XML que se va a importar como valor predefinido en el panel Configuración predefinida de movimiento.

`namePath` Una cadena que especifica la carpeta en la que se coloca el archivo importado y el nombre que se le asigna. Este parámetro es opcional.

Valor devuelto
Un valor booleano `true` si el archivo se importa correctamente; `false` en caso contrario.

Descripción
Añade un valor predefinido al panel Configuración predefinida de movimiento a partir de un archivo XML especificado. La ruta especificada en *namePath* debe existir en el panel.

Para crear archivos XML importables, utilice `presetPanel.exportItem()`.

Si no transfiere ningún valor para *namePath*, el valor predefinido se coloca en la carpeta Valores predefinidos personalizados y adoptan el mismo nombre que el archivo importado (sin la extensión XML).

Ejemplo
En siguiente ejemplo importa un valor predefinido en la carpeta Valores predefinidos personalizados/Pulso y le asigna el nombre `fastPulse`.

```
fl.presetPanel.importItem("file:///C|/My Presets/thePulsePreset.xml", "Custom
Presets/Pulse/fastPulse");
```

Véase también
`presetPanel.exportItem()`

presetPanel.items

Disponibilidad
Flash CS4 Professional

Uso
`presetPanel.items`

Descripción
Un conjunto de objetos presetItem en el panel Configuración predefinida de movimiento (consulte Objeto presetItem). Cada elemento del conjunto representa una carpeta o un valor predefinido.

Ejemplo

El siguiente código muestra los nombres de ruta completos de los elementos del panel Configuración predefinida de movimiento:

```
var itemArray = fl.presetPanel.items;
var length =  itemArray.length
for (x=0; x<length; x++) {
    fl.trace(itemArray[x].path);
}
```

Véase también

presetPanel.getSelectedItems()

presetPanel.moveToFolder()

Disponibilidad

Flash CS4 Professional

Uso

```
presetPanel.moveToFolder(folderPath [, namePath] )
```

Parámetros

`folderPath` Una cadena que especifica la ruta de la carpeta del panel Configuración predefinida de movimiento a la que se mueven los elementos.

`namePath` Una cadena que especifica el nombre y la ruta del elemento que se va a mover. Este parámetro es opcional.

Valor devuelto

Un valor booleano `true` si los elementos se mueven correctamente; `false` en caso contrario.

Descripción

Mueve el elemento especificado a la carpeta especificada.

Si transfiere una cadena vacía ("") para *folderPath*, los elementos se mueven a la carpeta Valores predefinidos personalizados. Si no transfiere un valor para *namePath*, se mueven los elementos seleccionados actualmente.

No es posible mover elementos en la carpeta Valores predefinidos personalizados.

Ejemplo

En el siguiente ejemplo, los elementos seleccionados se mueven a la carpeta Valores predefinidos personalizados/Rebote y, seguidamente, el valor predefinido de rebote rápido se mueve a la misma carpeta:

```
fl.presetPanel.moveToFolder("Custom Presets/Bouncing");
fl.presetPanel.moveToFolder("Custom Presets/Bouncing" , "Custom Presets/Fast Bounce");
```

presetPanel.newFolder()

Disponibilidad
Flash CS4 Professional

Uso
```
presetPanel.newFolder( [folderPath] )
```

Parámetros
`folderPath` Una cadena que especifica dónde se añade una nueva carpeta en el panel Configuración predefinida de movimiento, y el nombre que se le asigna. Este parámetro es opcional.

Valor devuelto
Un valor booleano `true` si se añade correctamente la carpeta; `false` en caso contrario.

Descripción
Crea una carpeta en el árbol de carpetas del panel Configuración predefinida de movimiento. Sólo es posible crear un nuevo nivel de carpetas con este método. Es decir, si transfiere "Valores predefinidos personalizados/Mi primera carpeta/Mi segunda carpeta" para *folderPath*, "Valores predefinidos personalizados/Mi primera carpeta" debe existir en el árbol de carpetas.

Si no se transfiere ningún valor para *folderPath*, se crea una carpeta llamada"Carpeta sin título *n*" en el primer nivel por debajo de "Valores predefinidos personalizados", donde *n* se incrementa cada vez que se añade una carpeta de este modo.

Nota: *no es posible añadir carpetas a la carpeta Valores predefinidos personalizados.*

Ejemplo
El siguiente ejemplo añade una carpeta denominada `Bouncing` bajo la carpeta Valores predefinidos personalizados:

```
fl.presetPanel.newFolder("Custom Presets/Bouncing");
```

Véase también
`presetPanel.addNewItem()`

presetPanel.renameItem()

Disponibilidad
Flash CS4 Professional

Uso
```
presetPanel.renameItem(newName)
```

Parámetros
`newName` Una cadena que especifica el nuevo nombre del valor predefinido o de la carpeta.

Valor devuelto
Un valor booleano `true` si el valor predefinido o la carpeta se cambian de nombre correctamente; `false` en caso contrario.

Descripción
Cambia el nombre del valor predefinido o la carpeta seleccionados por un nombre especificado. Este método sólo funciona correctamente si se selecciona un solo valor predefinido o una sola carpeta en Valores predefinidos personalizados. Este método falla en los siguientes casos:

- No hay ningún elemento seleccionado.
- Hay varios elementos seleccionados.
- El elemento seleccionado se encuentra en la carpeta Valores predefinidos predeterminados.
- Existe un elemento llamado *newName* en la misma ubicación que el elemento seleccionado.

Ejemplo
El siguiente ejemplo cambia el nombre del valor predefinido seleccionado en la carpeta Valores predefinidos personalizados por `Bounce Faster`.

```
var renamed = fl.presetPanel.renameItem("Bounce Faster");
fl.trace(renamed);
```

presetPanel.selectItem()

Disponibilidad
Flash CS4 Professional

Uso
```
presetPanel.selectItem(namePath [, bReplaceCurrentSelection [, bSelect] ])
```

Parámetros
`namePath` Una cadena que especifica la ruta y el nombre del elemento que se va a seleccionar en el panel Configuración predefinida de movimiento.

`bReplaceCurrentSelection` Un valor booleano que indica si el elemento especificado reemplaza la selección actual (`true`) o si se añade a la selección actual (`false`). Este parámetro es opcional; su valor predeterminado es `true`.

`bSelect` Un valor booleano que especifica si el elemento se selecciona (`true`) o si se anula su selección (`false`). Este parámetro es opcional; su valor predeterminado es `true`. Si transfiere `false` para *bSelect*, el valor de *bReplaceCurrentSelection* se omite.

Valor devuelto
Un valor booleano `true` si el elemento se selecciona y deselecciona correctamente; `false` en caso contrario.

Descripción

Selecciona o anula la selección de un elemento del panel Configuración predefinida de movimiento (opcionalmente, puede reemplazar los elementos seleccionados).

Ejemplo

El siguiente código añade el valor predefinido `fly-in-blur-right` a los valores predefinidos seleccionados (si los hay) en el panel Configuración predefinida de movimiento:

```
fl.presetPanel.selectItem("Default Presets/fly-in-blur-right", false);
```

Capítulo 35: Objeto Rectangle

Herencia Objeto Element > Objeto Shape > Objeto Rectangle

Disponibilidad
Flash CS3 Professional

Descripción
El objeto Rectangle es una forma que se dibuja con la herramienta Rectángulo simple. Para determinar si un elemento es un objeto Rectangle, utilice `shape.isRectangleObject`.

Resumen de propiedades
Además de las propiedades del Objeto Shape puede utilizar las siguientes propiedades con el objeto Rectangle. Para definir las propiedades de un objeto Rectangle, utilice `document.setRectangleObjectProperty()`.

Propiedad	Descripción
`RectangleObject.bottomLeftRadius`	De sólo lectura; un valor flotante que establece el radio de la esquina inferior izquierda del objeto Rectangle.
`RectangleObject.bottomRightRadius`	De sólo lectura; un valor flotante que establece el radio de la esquina inferior derecha del objeto Rectangle.
`RectangleObject.lockFlag`	De sólo lectura; un valor booleano que determina si las distintas esquinas del rectángulo pueden tener diferentes valores de radio.
`RectangleObject.topLeftRadius`	De sólo lectura; un valor flotante que establece el radio de todas las esquinas del rectángulo o que define sólo el radio de la esquina superior izquierda del objeto Rectangle.
`RectangleObject.topRightRadius`	De sólo lectura; un valor flotante que establece el radio de la esquina superior derecha del objeto Rectangle.

RectangleObject.bottomLeftRadius

Disponibilidad
Flash CS3 Professional

Uso
```
RectangleObject.bottomLeftRadius
```

Descripción
Propiedad de sólo lectura; un valor flotante que establece el radio de la esquina inferior izquierda del objeto Rectangle. Si `RectangleObject.lockFlag` es `true`, no tiene ningún efecto la definición de este valor.

Para definir este valor, utilice `document.setRectangleObjectProperty()`.

Véase también

`document.setRectangleObjectProperty()`, `RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`, `RectangleObject.topRightRadius`

RectangleObject.bottomRightRadius

Disponibilidad

Flash CS3 Professional

Uso

```
RectangleObject.bottomRightRadius
```

Descripción

Propiedad de sólo lectura; un valor flotante que establece el radio de la esquina inferior derecha del objeto Rectangle. Si `RectangleObject.lockFlag` es true, no tiene ningún efecto la definición de este valor.

Para definir este valor, utilice `document.setRectangleObjectProperty()`.

Véase también

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`, `RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`, `RectangleObject.topRightRadius`

RectangleObject.lockFlag

Disponibilidad

Flash CS3 Professional

Uso

```
RectangleObject.lockFlag
```

Descripción

Propiedad de sólo lectura; un valor booleano que determina si las distintas esquinas del rectángulo pueden tener diferentes valores de radio. Si este valor es `true`, todas las esquinas tienen el valor asignado como `RectangleObject.topLeftRadius`. Su es `false`, se puede definir cada radio de esquina de forma independiente.

Para definir este valor, utilice `document.setRectangleObjectProperty()`.

Véase también

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`, `RectangleObject.bottomRightRadius`, `RectangleObject.topLeftRadius`, `RectangleObject.topRightRadius`

RectangleObject.topLeftRadius

Disponibilidad

Flash CS3 Professional

Uso

`RectangleObject.topLeftRadius`

Descripción

Propiedad de sólo lectura; un valor flotante que establece el radio de todas las esquinas del rectángulo (si el valor de `RectangleObject.lockFlag` es `true`) o sólo el radio de la esquina superior izquierda (si el valor de `RectangleObject.lockFlag` es `false`).

Para definir este valor, utilice `document.setRectangleObjectProperty()`.

Véase también

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`, `RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`, `RectangleObject.topRightRadius`

RectangleObject.topRightRadius

Disponibilidad

Flash CS3 Professional

Uso

`RectangleObject.topRightRadius`

Descripción

Propiedad de sólo lectura; un valor flotante que establece el radio de la esquina superior derecha del objeto Rectangle. Si `RectangleObject.lockFlag` es `true`, no tiene ningún efecto la definición de este valor.

Para definir este valor, utilice `document.setRectangleObjectProperty()`.

Véase también

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`, `RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`

Capítulo 36: Objeto Screen

Disponibilidad

Flash MX 2004

Descripción

El objeto Screen representa una pantalla única en un documento de diapositivas o formularios. Este objeto contiene propiedades relacionadas con la diapositiva o el formulario. Para acceder al conjunto de todos los objetos Screen del documento, utilice el código siguiente:

```
fl.getDocumentDOM().screenOutline.screens
```

Resumen de propiedades

El objeto Screen tiene las propiedades siguientes:

Propiedades	Descripción
`screen.accName`	Una cadena que equivale al campo Nombre del panel Accesibilidad.
`screen.childScreens`	De sólo lectura; el conjunto de pantallas secundarias de esta pantalla. El conjunto está vacío si no hay pantallas secundarias.
`screen.description`	Una cadena que equivale al campo Descripción del panel Accesibilidad.
`screen.forceSimple`	Un valor booleano que activa o desactiva la accesibilidad para los elementos secundarios del objeto.
`screen.hidden`	Un valor booleano que especifica si la pantalla es visible.
`screen.instanceName`	De sólo lectura; una cadena que representa el nombre de instancia empleado para acceder al objeto desde ActionScript.
`screen.name`	De sólo lectura; una cadena que representa el nombre de la pantalla.
`screen.nextScreen`	De sólo lectura; un objeto que representa la siguiente pantalla del mismo nivel del conjunto `childScreens` principal.
`screen.parameters`	De sólo lectura; un conjunto de propiedades de ActionScript 2.0 accesibles desde el inspector de propiedades de la pantalla.
`screen.parentScreen`	De sólo lectura; un objeto que representa la pantalla principal.
`screen.prevScreen`	De sólo lectura; un objeto que representa la pantalla anterior del mismo nivel del conjunto `childScreens` principal.
`screen.silent`	Un valor booleano que especifica si el objeto es accesible.
`screen.tabIndex`	Equivale al campo Índice de fichas del panel Accesibilidad.
`screen.timeline`	De sólo lectura; el objeto Timeline para la pantalla. Consulte Objeto Timeline.

screen.accName

Disponibilidad

Flash MX 2004

Uso

```
screen.accName
```

Descripción

Propiedad; una cadena que equivale al campo Nombre del panel Accesibilidad. Los lectores de pantalla identifican los objetos mediante la lectura del nombre en voz alta.

Ejemplo

El ejemplo siguiente almacena el valor del nombre del objeto en la variable `theName`:

```
var theName = fl.getDocumentDOM().screenOutline.screens[1].accName;
```

El ejemplo siguiente define el nombre del objeto como `"Home Button"`:

```
fl.getDocumentDOM().screenOutline.screens[1].accName = 'Home Button';
```

screen.childScreens

Disponibilidad

Flash MX 2004

Uso

```
screen.childScreens
```

Descripción

Propiedad de sólo lectura; el conjunto de pantallas secundarias de esta pantalla. El conjunto está vacío si no hay pantallas secundarias.

Ejemplo

El ejemplo siguiente comprueba si el documento actual es una diapositiva o un formulario. Si lo es, almacena el conjunto de pantallas secundarias en la variable `myChildren` y muestra sus nombres en el panel Salida:

```
var myChildren = new Array();
if(fl.getDocumentDOM().allowScreens) {
    var myParent = fl.getDocumentDOM().screenOutline.rootScreen.name
    for (i in fl.getDocumentDOM().screenOutline.rootScreen.childScreens) {
        myChildren.push(" "+fl.getDocumentDOM().screenOutline.rootScreen.childScreens[i].name);
    }
    fl.trace(" The child screens of "+myParent+" are "+myChildren+". ");
}
```

screen.description

Disponibilidad

Flash MX 2004

Uso

```
screen.description
```

Descripción
Propiedad; una cadena que equivale al campo Descripción del panel Accesibilidad. El lector de pantalla lee esta descripción.

Ejemplo
El ejemplo siguiente obtiene la descripción de la pantalla y la almacena en la variable `theDescription`:

```
var theDescription = fl.getDocumentDOM().screenOutline.screens[1].description;
```

En el ejemplo siguiente se define la descripción de la pantalla como `Home Screen`:

```
fl.getDocumentDOM().screenOutline.screens[1].description = "Home Screen";
```

screen.forceSimple

Disponibilidad
Flash MX 2004

Uso
`screen.forceSimple`

Descripción
Propiedad; un valor booleano que activa o desactiva la accesibilidad para los elementos secundarios del objeto. Equivale a la lógica inversa de la opción Hacer que los objetos secundarios sean accesibles del panel Accesibilidad. Es decir, si `forceSimple` es `true`, equivale a la opción desactivada Hacer que los objetos secundarios sean accesibles. Si `forceSimple` es `false`, equivale a la opción activada Hacer que los objetos secundarios sean accesibles.

Ejemplo
El ejemplo siguiente almacena el valor de `forceSimple` en la variable `areChildrenAccessible` (un valor de `false` significa que los elementos secundarios del objeto son accesibles):

```
var areChildrenAccessible = fl.getDocumentDOM().screenOutline.screens[1].forceSimple
```

El ejemplo siguiente hace que los elementos secundarios del objeto sean accesibles:

```
fl.getDocumentDOM().screenOutline.screens[1].forceSimple = false;
```

screen.hidden

Disponibilidad
Flash MX 2004

Uso
`screen.hidden`

Descripción
Propiedad; un valor booleano que especifica si la pantalla es visible. Una pantalla con la propiedad `hidden` definida como `true` no es visible en ninguna otra pantalla.

Ejemplo

El ejemplo siguiente comprueba si la primera pantalla del contorno está oculta y cambia la visibilidad de la pantalla en consonancia. A continuación, un mensaje muestra en el panel Salida cuál era la visibilidad de la pantalla antes del cambio:

```
if (fl.getDocumentDOM().screenOutline.screens[0].hidden) {
    fl.getDocumentDOM().screenOutline.setScreenProperty("hidden", false);
    fl.trace(fl.getDocumentDOM().screenOutline.screens[0].name+" had its 'hidden' property
set to 'false'");
}
else {
    fl.getDocumentDOM().screenOutline.setScreenProperty("hidden", true);
    fl.trace(fl.getDocumentDOM().screenOutline.screens[0].name+" had its 'hidden' property
set to 'true'");
}
```

screen.instanceName

Disponibilidad

Flash MX 2004

Uso

```
screen.instanceName
```

Descripción

Propiedad de sólo lectura; una cadena que representa el nombre de instancia empleado para acceder al objeto desde ActionScript.

Ejemplo

El ejemplo siguiente comprueba si el documento actual admite pantallas (porque es una diapositiva o un formulario). A continuación, asigna el valor `instanceName` de la primera pantalla secundaria del conjunto a la variable `myInstanceName` y abre el panel Salida para mostrar el nombre de instancia de la pantalla:

```
var myChildren = new Array();
if(fl.getDocumentDOM().allowScreens) {
    var myInstanceName =
fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].instanceName;
    fl.trace(" The instanceName is "+myInstanceName+". ");
}
```

screen.name

Disponibilidad

Flash MX 2004

Uso

```
screen.name
```

Descripción

Propiedad de sólo lectura; una cadena que representa el nombre de la pantalla.

Ejemplo

El ejemplo siguiente comprueba si el documento actual admite pantallas (porque es una diapositiva o un documento de formulario). A continuación, asigna el valor `name` de la primera pantalla secundaria del conjunto a la variable `myName` y abre el panel Salida para mostrar el nombre de la pantalla:

```
var myChildren = new Array();
if(fl.getDocumentDOM().allowScreens) {
    var myName = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;
    fl.trace("The name of the screen is "+myName+". ");
}
```

screen.nextScreen

Disponibilidad

Flash MX 2004

Uso

```
screen.nextScreen
```

Descripción

Propiedad de sólo lectura; un objeto que representa la siguiente pantalla del mismo nivel del conjunto `childScreens` principal. Es decir, `screen.NextScreen` se encuentra bajando en un conjunto de pantallas secundarias a la siguiente pantalla del conjunto. Consulte `screen.prevScreen`.

Si no hay una pantalla del mismo nivel, el valor es `null`.

Ejemplo

El ejemplo siguiente comprueba primero si el documento actual es una diapositiva o un formulario y, si lo es, recupera y muestra la secuencia de pantallas en el panel Salida:

```
if(fl.getDocumentDOM().allowScreens) {
    var myCurrent = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;
    var myNext = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].nextScreen.name;
    fl.trace(" The next screen to "+myCurrent+" is "+myNext+". ");
}
```

screen.parameters

Disponibilidad

Flash MX 2004

Uso

```
screen.parameters
```

Descripción
Propiedad de sólo lectura; un conjunto de propiedades de ActionScript 2.0 accesibles desde el inspector de propiedades de la pantalla.

Ejemplo
El ejemplo siguiente almacena los parámetros de la segunda pantalla del contorno en la variable `parms` y, a continuación, asigna el valor `some value` a la primera propiedad:

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;
parms[0].value = "some value";
```

Véase también
Objeto Parameter

screen.parentScreen

Disponibilidad
Flash MX 2004

Uso
```
screen.parentScreen
```

Descripción
Propiedad de sólo lectura; un objeto que representa la pantalla principal. Si `parentScreen` es `null`, la pantalla es una pantalla de nivel superior.

Ejemplo
El ejemplo siguiente almacena los valores de las propiedades `childScreen` y `parentScreen` en variables y, a continuación, muestra esos valores y sus relaciones superior/secundario en el panel Salida:

```
if(fl.getDocumentDOM().allowScreens) {
    var myCurrent = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].name;
    var myParent =
fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].parentScreen.name;
    fl.trace(" The parent screen to "+myCurrent+" is "+myParent+". ");
}
```

screen.prevScreen

Disponibilidad
Flash MX 2004

Uso
```
screen.prevScreen
```

Descripción

Propiedad de sólo lectura; un objeto que representa la pantalla anterior del mismo nivel del conjunto `childScreens` principal. Si no hay una pantalla del mismo nivel, el valor es `null`. Véase también `screen.nextScreen`.

Ejemplo

El ejemplo siguiente comprueba si el documento actual es una diapositiva o un formulario y, si lo es, recupera y muestra la secuencia de pantallas en el panel Salida:

```
if(fl.getDocumentDOM().allowScreens) {
    var myCurrent = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].name;
    var myNext = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].prevScreen.name;
    fl.trace(" The previous screen to "+myCurrent+" is "+myNext+". ");
}
```

screen.silent

Disponibilidad

Flash MX 2004

Uso

```
screen.silent
```

Descripción

Propiedad; un valor booleano que especifica si el objeto es accesible. Equivale a la lógica inversa de la opción Hacer que el objeto sea accesible del panel Accesibilidad. Es decir, si `silent` es `true`, equivale a la opción desactivada Hacer que los objetos secundarios sean accesibles en el panel Accesibilidad. Si `silent` es `false`, equivale a la opción activada Hacer que los objetos secundarios sean accesibles en el panel Accesibilidad.

Ejemplo

El ejemplo siguiente recupera el valor `silent` del objeto (un valor de `false` significa que el objeto es accesible):

```
var isSilent = fl.getDocumentDOM().screenOutline.screens[1].silent;
```

El ejemplo siguiente define el objeto como accesible:

```
fl.getDocumentDOM().screenOutline.screens[1].silent = false;
```

screen.tabIndex

Disponibilidad

Flash MX 2004

Uso

```
screen.tabIndex
```

Descripción

Propiedad; equivale al campo Índice de fichas del panel Accesibilidad. Este valor permite determinar el orden de acceso a los objetos cuando el usuario presiona la tecla Tabulador.

Ejemplo

El ejemplo siguiente obtiene el índice de tabulación del objeto:

```
var theTabIndex = fl.getDocumentDOM().screenOutline.screens[1].tabIndex;
```

El ejemplo siguiente define el índice de tabulación del objeto como 1:

```
fl.getDocumentDOM().screenOutline.screens[1].tabIndex = 1;
```

screen.timeline

Disponibilidad

Flash MX 2004

Uso

```
screen.timeline
```

Descripción

Propiedad de sólo lectura; el Objeto Timeline para la pantalla.

Ejemplo

El ejemplo siguiente obtiene la propiedad `screenOutline` del documento de diapositiva actual, asigna el conjunto de propiedades `timeline` para la primera pantalla a `myArray` y muestra esas propiedades en el panel Salida:

```
myArray = new Array();
if(fl.getDocumentDOM().screenOutline) {
    for(i in fl.getDocumentDOM().screenOutline.screens[0].timeline) {
    myArray.push(" "+i+" : "+fl.getDocumentDOM().screenOutline.screens[0].timeline[i]+" ") ;
    }
    fl.trace("Here are the properties of the screen named "+
fl.getDocumentDOM().screenOutline.screens[0].name+": "+myArray);
}
```

Capítulo 37: Objeto ScreenOutline

Disponibilidad

Flash MX 2004

Descripción

El objeto ScreenOutline representa el grupo de pantallas en un documento de diapositivas o formularios. El acceso al objeto se realiza utilizando `fl.getDocumentDOM().screenOutline`.

El objeto ScreenOutline sólo existe si el documento es una diapositiva o un documento de formulario. Por tanto, antes de acceder a la propiedad, utilice `document.allowScreens()` para verificar que existe un documento de pantallas, como se muestra en el ejemplo siguiente:

```
if(fl.getDocumentDOM().allowScreens) {
var myName = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;
fl.trace("The name of the screen is " + myName + ". ");
}
```

Resumen de métodos

Puede emplear los métodos siguientes con el objeto ScreenOutline:

Método	Descripción
`screenOutline.copyScreenFromFile()`	Inserta todas las pantallas o una pantalla con nombre y sus elementos secundarios, desde un documento especificado en la pantalla seleccionada actualmente.
`screenOutline.deleteScreen()`	Elimina la pantalla o pantallas seleccionadas actualmente, o una pantalla especificada, y los elementos secundarios de dichas pantallas.
`screenOutline.duplicateScreen()`	Duplica la pantalla o pantallas seleccionadas actualmente o una pantalla especificada.
`screenOutline.getSelectedScreens()`	Devuelve un conjunto de objetos Screen que se encuentran seleccionados actualmente en el contorno de pantalla.
`screenOutline.insertNestedScreen()`	Inserta una pantalla anidada de un tipo específico en una determinada ubicación del contorno de pantalla.
`screenOutline.insertScreen()`	Inserta una pantalla nueva vacía de un tipo especificado en el documento en una ubicación determinada.
`screenOutline.moveScreen()`	Mueve la pantalla especificada en relación con el valor del parámetro ***`referenceScreen`***; antes, después, como primer elemento secundario o como último elemento secundario.
`screenOutline.renameScreen()`	Cambia el nombre especificado de la pantalla.
`screenOutline.setCurrentScreen()`	Define la selección actual en el contorno de pantalla en la pantalla especificada.
`screenOutline.setScreenProperty()`	Define la propiedad especificada con el valor especificado para las pantallas seleccionadas.
`screenOutline.setSelectedScreens()`	Selecciona las pantallas especificadas en el panel Contorno de pantalla.

Resumen de propiedades

Puede emplear las propiedades siguientes con el objeto ScreenOutline:

Propiedad	Descripción
`screenOutline.currentScreen`	Un Objeto Screen; la pantalla seleccionada actualmente.
`screenOutline.rootScreen`	De sólo lectura; la primera pantalla del contorno de pantalla.
`screenOutline.screens`	Sólo lectura; el conjunto de los objetos Screen de nivel superior que contiene el documento (consulte Objeto Screen).

screenOutline.copyScreenFromFile()

Disponibilidad

Flash MX 2004

Uso

```
screenOutline.copyScreenFromFile(fileURI [, screenName])
```

Parámetros

fileURI Una cadena, expresada como URI file:///, que especifica el nombre del archivo de edición que contiene las pantallas que se van a copiar en el documento.

screenName El nombre de la pantalla que se va a copiar. Si está presente el parámetro *screenName*, Flash copiará esa pantalla y sus elementos secundarios. Si no se especifica *screenName*, Flash copiará todo el documento. Este parámetro es opcional.

Valor devuelto

Ninguno. Si no se encuentra el archivo, no es un archivo FLA válido o no se encuentra la pantalla especificada, se presenta un error y se cancela el script.

Descripción

Método; inserta todas las pantallas o una pantalla con nombre y sus elementos secundarios, desde un documento especificado en la pantalla seleccionada actualmente. Si hay varias pantallas seleccionadas, se insertarán bajo la última pantalla seleccionada, como elementos del mismo nivel.

Ejemplo

El ejemplo siguiente copia la pantalla `slide1` desde el archivo myTarget.fla del escritorio hasta el documento actual (utilice su nombre de usuario en *userName*):

```
fl.getDocumentDOM().screenOutline.copyScreenFromFile("file:///C|/Documents and
Settings/userName/Desktop/myTarget.fla", "slide1");
```

screenOutline.currentScreen

Disponibilidad

Flash MX 2004

Uso

```
screenOutline.currentScreen
```

Descripción

Propiedad; un objeto Screen, la pantalla seleccionada actualmente (consulte Objeto Screen).

Ejemplo

El ejemplo siguiente almacena el objeto `currentScreen` en la variable `myScreen` y, a continuación, muestra el nombre de esa pantalla en el panel Salida:

```
var myScreen = fl.getDocumentDOM().screenOutline.currentScreen;
fl.trace(myScreen.name);
```

screenOutline.deleteScreen()

Disponibilidad

Flash MX 2004

Uso

```
screenOutline.deleteScreen([screenName])
```

Parámetros

`screenName` Una cadena que especifica el nombre de la pantalla que se va a eliminar. Si no transfiere un valor para *screenName*, se eliminarán la pantalla o pantallas seleccionadas actualmente y sus elementos secundarios. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; elimina la pantalla o pantallas seleccionadas actualmente, o una pantalla especificada, y los elementos secundarios de dichas pantallas.

Ejemplo

El ejemplo siguiente elimina la pantalla llamada `apple` y todos sus elementos secundarios:

```
fl.getDocumentDOM().screenOutline.deleteScreen("apple");
```

screenOutline.duplicateScreen()

Disponibilidad

Flash MX 2004

Uso

```
screenOutline.duplicateScreen([screenName])
```

Parámetros

`screenName` Un valor de cadena que especifica el nombre de la pantalla que se va a duplicar. Si no transfiere un valor para *screenName*, se duplicarán la pantalla o pantallas seleccionadas actualmente. Este parámetro es opcional.

Valor devuelto

Valor booleano: `true` si la pantalla se duplica correctamente, y `false` en caso contrario.

Descripción

Método; duplica la pantalla o pantallas seleccionadas actualmente o una pantalla especificada. Las pantallas duplicadas reciben un nombre predeterminado añadiendo `_copy` al nombre original, como por ejemplo, Screen_copy, Screen_copy2, etc. Si duplica varias pantallas, los duplicados se colocarán directamente bajo la pantalla seleccionada que se encuentre en el nivel más bajo de la jerarquía de contornos de pantallas.

Ejemplo

El ejemplo siguiente duplica una pantalla llamada apple:

```
fl.getDocumentDOM().screenOutline.duplicateScreen("apple");
```

screenOutline.getSelectedScreens()

Disponibilidad

Flash MX 2004

Uso

```
screenOutline.getSelectedScreens()
```

Parámetros

Ninguno.

Valor devuelto

Un conjunto de objetos Screen seleccionados (consulte Objeto Screen).

Descripción

Método; devuelve un conjunto de objetos Screen que se encuentran seleccionados actualmente en el contorno de pantalla.

Ejemplo

El ejemplo siguiente almacena los objetos Screen seleccionados en la variable `myArray` y muestra los nombres de pantalla en el panel Salida:

```
var myArray = fl.getDocumentDOM().screenOutline.getSelectedScreens();
for (var i in myArray) {
    fl.trace(myArray[i].name)
}
```

screenOutline.insertNestedScreen()

Disponibilidad
Flash MX 2004

Uso
```
screenOutline.insertNestedScreen([name [, referenceScreen [, screenTypeName]]])
```

Parámetros
name Una cadena que indica el nombre de la nueva pantalla que se va a insertar. Un nombre vacío insertará una pantalla con un nombre de pantalla predeterminado, como Diapositiva*n* o Formulario*n* (donde *n* es el primer número exclusivo disponible). Este parámetro es opcional.

referenceScreen Una cadena que indica el nombre de la pantalla en la que se insertará la nueva pantalla como elemento secundario. Si se omite este parámetro, se insertará la nueva pantalla como elemento secundario de la pantalla seleccionada actualmente. Este parámetro es opcional.

screenTypeName Una cadena que especifica el tipo de pantalla que se va a asociar a la nueva pantalla anidada. Se definen el tipo de pantalla y el nombre de clase para esta pantalla. Los valores aceptables son `"Form"` y `"Slide"`. Este parámetro es opcional. Si se omite este parámetro, el tipo se heredará de la pantalla principal.

Valor devuelto
Un Objeto Screen.

Descripción
Método; inserta una pantalla anidada de un tipo específico en una determinada ubicación del contorno de pantalla.

Ejemplo
El ejemplo siguiente inserta `slide2` como elemento secundario de `slide1`:

```
fl.getDocumentDOM().screenOutline.insertNestedScreen("slide2", "slide1", "Slide");
```

screenOutline.insertScreen()

Disponibilidad
Flash MX 2004

Uso
```
screenOutline.insertScreen([name [, referenceScreen [, screenTypeName]]])
```

Parámetros
name Una cadena que indica el nombre de la nueva pantalla que se va a insertar. Si se omite este parámetro, el método inserta una pantalla con un nombre de pantalla predeterminado, como Diapositiva*n* o Formulario*n* (donde *n* es el primer número exclusivo disponible). Este parámetro es opcional.

referenceScreen Una cadena que indica el nombre de la pantalla anterior a la nueva pantalla. Si se omite este parámetro, la nueva pantalla se insertará después de la pantalla seleccionada actualmente. Si el parámetro

referenceScreen identifica una pantalla secundaria, la nueva pantalla será un elemento del mismo nivel de la pantalla secundaria y una pantalla secundaria de la misma pantalla principal. Este parámetro es opcional.

screenTypeName Una cadena que especifica el tipo de pantalla que se va a asociar a la nueva pantalla. Se definen el tipo de pantalla y el nombre de clase para esta pantalla. Los valores aceptables son `"Form"` y `"Slide"`. Este parámetro es opcional.

Valor devuelto

Un Objeto Screen.

Descripción

Método; inserta una pantalla nueva vacía de un tipo especificado en el documento en una ubicación determinada.

Ejemplo

El ejemplo siguiente inserta un formulario denominado `slide2` después de la pantalla llamada `slide1`:

```
fl.getDocumentDOM().screenOutline.insertScreen("slide2","slide1","Form");
```

El ejemplo siguiente inserta una diapositiva llamada `slide4` después de la pantalla `slide3`:

```
fl.getDocumentDOM().screenOutline.insertScreen("slide4","slide3","Slide");
```

screenOutline.moveScreen()

Disponibilidad

Flash MX 2004

Uso

```
screenOutline.moveScreen(screenToMove, referenceScreen, position)
```

Parámetros

screenToMove Una cadena que es el nombre de pantalla que se va a desplazar.

referenceScreen Una cadena que especifica la pantalla cerca de la cual se colocará *screenToMove*.

position Una cadena que especifica dónde se desplazará la pantalla en relación con *referenceScreen*. Los valores aceptables son `"before"`, `"after"`, `"firstChild"` y `"lastChild"`.

Valor devuelto

Un valor booleano; `true` si el elemento se mueve correctamente; en caso contrario, `false`.

Descripción

Método; mueve la pantalla especificada en relación con el valor del parámetro *referenceScreen*; antes, después, como primer elemento secundario o como último elemento secundario.

Ejemplo

El ejemplo siguiente desplaza la pantalla `slide1` para que sea el primer elemento secundario de `slide2`:

```
fl.getDocumentDOM().screenOutline.moveScreen("slide1", "slide2", "firstChild");
```

screenOutline.renameScreen()

Disponibilidad
Flash MX 2004

Uso
```
screenOutline.renameScreen(newScreenName [, oldScreenName[, bDisplayError]])
```

Parámetros
newScreenName Una cadena que especifica el nuevo nombre de la pantalla.

oldScreenName Una cadena que especifica el nombre de la pantalla existente que se va a cambiar. Si no se especifica, cambiará el nombre de la pantalla seleccionada actualmente. Este parámetro es opcional.

bDisplayError Un valor booleano que, si se define como `true`, muestra un mensaje si se produce un error, por ejemplo, si ya existe una pantalla con el mismo nombre que el valor transferido a *newScreenName*. El valor predeterminado es `false`.

Valor devuelto
Un valor booleano; `true` si el elemento se cambia de nombre correctamente; en caso contrario, `false`.

Descripción
Método; cambia el nombre especificado de la pantalla.

Ejemplo
El ejemplo siguiente cambia el nombre de `slide1` a `Intro`:

```
fl.getDocumentDOM().screenOutline.renameScreen("Intro", "slide1");
```

screenOutline.rootScreen

Disponibilidad
Flash MX 2004

Uso
```
screenOutline.rootScreen
```

Descripción
Propiedad de sólo lectura; la primera pantalla del contorno de pantalla. Puede utilizar `screenOutline.rootScreen` como método abreviado de `screenOutline.screens[0]`.

Ejemplo
El ejemplo siguiente muestra el nombre del primer elemento secundario de la primera pantalla en el contorno de pantalla:

```
var n = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;
fl.trace(n);
```

screenOutline.screens

Disponibilidad
Flash MX 2004

Uso
`screenOutline.screens`

Descripción
Propiedad de sólo lectura; el conjunto de los objetos Screen de nivel superior que contiene el documento (consulte Objeto Screen).

Ejemplo
El ejemplo siguiente almacena el conjunto de objetos Screen en la variable `myArray` y, a continuación, muestra sus nombres en el panel Salida:

```
var myArray = new Array();
if(fl.getDocumentDOM().allowScreens) {
for(var i in fl.getDocumentDOM().screenOutline.screens) {
    myArray.push(" "+fl.getDocumentDOM().screenOutline.screens[i].name);
}
fl.trace(2"The screens array contains objects whose names are: "+myArray+". ");
}
```

screenOutline.setCurrentScreen()

Disponibilidad
Flash MX 2004

Uso
`screenOutline.setCurrentScreen(name)`

Parámetros
name Una cadena que especifica el nombre de la pantalla que debe convertirse en la pantalla seleccionada actualmente. Si la pantalla es un elemento secundario de otra pantalla, no será necesario que indique una ruta o una jerarquía.

Valor devuelto
Ninguno.

Descripción
Método; define la selección actual en el contorno de pantalla en la pantalla especificada.

Ejemplo
El ejemplo siguiente define la pantalla actual en la pantalla llamada `ChildOfSlide_1`:

```
fl.getDocumentDOM().screenOutline.setCurrentScreen("ChildOfSlide_1");
```

screenOutline.setScreenProperty()

Disponibilidad
Flash MX 2004

Uso
```
screenOutline.setScreenProperty(property, value)
```

Parámetros
property Una cadena que especifica la propiedad que se va a establecer.

value Nuevo valor para la propiedad. El tipo de valor depende de la propiedad que se está definiendo.

Las propiedades disponibles son `screenOutline.currentScreen`, `screenOutline.rootScreen` y `screenOutline.screens`.

Valor devuelto
Ninguno.

Descripción
Método; define la propiedad especificada con el valor especificado para las pantallas seleccionadas.

Ejemplo
El ejemplo siguiente cambia la visibilidad de las pantallas seleccionadas actualmente de oculta a visible:

```
fl.getDocumentDOM().screenOutline.setScreenProperty("hidden", false);
```

screenOutline.setSelectedScreens()

Disponibilidad
Flash MX 2004

Uso
```
screenOutline.setSelectedScreens(selection [, bReplaceCurrentSelection])
```

Parámetros
selection Un conjunto de nombres de pantalla que se seleccionarán en el contorno de pantalla.

bReplaceCurrentSelection Un valor booleano que, si es `true`, permite anular la selección actual. El valor predeterminado es `true`. Si es `false`, Flash extiende la selección actual para incluir las pantallas especificadas. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; selecciona las pantallas especificadas en el contorno de pantalla. Si se especifican múltiples pantallas, se seleccionará en el escenario la pantalla con el último valor de índice del conjunto de selección.

Ejemplo

El ejemplo siguiente anula la selección de las pantallas seleccionadas actualmente y, a continuación, selecciona las pantallas `slide1`, `slide2`, `slide3` y `slide4` en el contorno de pantalla:

```
myArray = new Array("slide1", "slide2", "slide3", "slide4");
fl.getDocumentDOM().screenOutline.setSelectedScreens(myArray, true);
```

Capítulo 38: Objeto Shape

Herencia Objeto Element > Objeto Shape

Disponibilidad

Flash MX 2004

Descripción

El objeto Shape es una subclase del objeto Element. El objeto Shape proporciona un control más preciso que las API de dibujo al manipular o crear geometría en el escenario. Este control es necesario para que los script puedan crear efectos útiles y otros comandos de dibujo (consulte Objeto Element).

Todas las propiedades y métodos Shape que cambian una forma y cualquiera de sus partes subordinadas se deben situar entre las llamadas `shape.beginEdit()` y `shape.endEdit()` para funcionar correctamente.

Resumen de métodos

Además de los métodos del objeto Element, puede emplear los métodos siguientes con el objeto Shape:

Método	Descripción
shape.getCubicSegmentPoints()	Devuelve un conjunto de puntos que definen una curva cúbica.
`shape.getCubicSegmentPoints()`	Define el comienzo de una sesión de edición.
`shape.deleteEdge()`	Elimina el borde especificado.
`shape.endEdit()`	Define el final de una sesión de edición para la forma.

Resumen de propiedades

Además de las propiedades del objeto Element, el objeto Shape dispone de las siguientes:

Propiedad	Descripción
`shape.contours`	Sólo lectura; un conjunto de objetos Contour para la forma (consulte Objeto Contour).
`shape.edges`	Sólo lectura; un conjunto de objetos Edge (consulte Objeto Edge).
`shape.isDrawingObject`	Sólo lectura; si es true, la forma será un objeto de dibujo.
`shape.isGroup`	Sólo lectura; si es true, la forma será un grupo.
`shape.isOvalObject`	Sólo lectura; si es true, la forma es un objeto Oval sencillo (creado con la herramienta Óvalo).
`shape.isRectangleObject`	Sólo lectura; si es true, la forma es un objeto Rectangle sencillo (creado con la herramienta Rectángulo).
`shape.members`	Conjunto de objetos del grupo seleccionado actualmente.
`shape.numCubicSegments`	Sólo lectura; el número de segmentos cúbicos de la forma.
`shape.vertices`	Sólo lectura; un conjunto de objetos Vertex (consulte Objeto Vertex).

shape.beginEdit()

Disponibilidad
Flash MX 2004

Uso
```
shape.beginEdit()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; define el comienzo de una sesión de edición. Deberá utilizar este método antes de ejecutar comandos que cambien el objeto Shape o cualquiera de sus partes subordinadas.

Ejemplo
El ejemplo siguiente toma la forma seleccionada actualmente y le quita el primer borde del conjunto de bordes:

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

shape.contours

Disponibilidad
Flash MX 2004

Uso
```
shape.contours
```

Descripción
Propiedad de sólo lectura; un conjunto de objetos Contour para la forma (consulte Objeto Contour).

Ejemplo
El ejemplo siguiente almacena el primer contorno del conjunto de contornos en la variable *c* y, a continuación, almacena el Objeto HalfEdge de ese contorno en la variable `he`:

```
var c = fl.getDocumentDOM().selection[0].contours[0];
var he = c.getHalfEdge();
```

shape.deleteEdge()

Disponibilidad
Flash MX 2004

Uso
```
shape.deleteEdge(index)
```

Parámetros
index Un índice basado en cero que especifica el borde que se va a eliminar del conjunto shape.edges. Este método cambia la longitud del conjunto `shape.edges`.

Valor devuelto
Ninguno.

Descripción
Método; elimina el borde especificado. Deberá llamar a `shape.beginEdit()` antes de utilizar este método.

Ejemplo
El ejemplo siguiente toma la forma seleccionada actualmente y quita el primer borde del conjunto de bordes:

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

shape.edges

Disponibilidad
Flash MX 2004

Uso
```
shape.edges
```

Descripción
Propiedad de sólo lectura; un conjunto de objetos Edge (consulte Objeto Edge).

shape.endEdit()

Disponibilidad
Flash MX 2004

Uso
```
shape.endEdit()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; define el final de una sesión de edición para la forma. Todos los cambios realizados en el objeto Shape o en cualquiera de sus partes subordinadas se aplicarán a la forma. Deberá utilizar este método después de ejecutar comandos que cambien el objeto Shape o cualquiera de sus partes subordinadas.

Ejemplo
El ejemplo siguiente toma la forma seleccionada actualmente y le quita el primer borde del conjunto de bordes:

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

shape.getCubicSegmentPoints()

Disponibilidad
Flash CS4 Professional

Uso
`shape.getCubicSegmentPoints(cubicSegmentIndex)`

Parámetros
`cubicSegmentIndex` Un entero que especifica qué puntos del segmento cúbico se devuelven.

Valor devuelto
Un conjunto de puntos que definen una curva cúbica para el objeto Edge que corresponde al elemento *cubicSegmentIndex* especificado (consulte `edge.cubicSegmentIndex`).

Descripción
Método; devuelve un conjunto de puntos que definen una curva cúbica.

Ejemplo
El siguiente ejemplo muestra los valores *x* e *y* para cada punto de la curva cúbica del primer borde de la selección:

```
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

Disponibilidad

Flash 8

Uso

```
shape.isDrawingObject
```

Descripción

Propiedad de sólo lectura; si es `true`, la forma será un objeto de dibujo.

Ejemplo

El ejemplo siguiente almacena el primer objeto seleccionado en la variable `sel` y, a continuación, utiliza las propiedades `element.elementType` y `shape.isDrawingObject` para determinar si el elemento seleccionado es un objeto de dibujo:

```
var sel = fl.getDocumentDOM().selection[0];
var shapeDrawingObject = (sel.elementType == "shape") && sel.isDrawingObject;
fl.trace(shapeDrawingObject);
```

Véase también

`document.crop()`, `document.deleteEnvelope()`, `document.intersect()`, `document.punch()`, `document.union()`, `shape.isGroup`

shape.isGroup

Disponibilidad

Flash MX 2004

Uso

```
shape.isGroup
```

Descripción

Propiedad de sólo lectura; si es `true`, la forma será un grupo. Un grupo puede contener distintos tipos de elementos, como elementos de texto o símbolos. Sin embargo, el propio grupo se considera una forma y puede utilizar la propiedad `shape.isGroup` sin importar qué elementos estén contenidos en el grupo.

Ejemplo

El ejemplo siguiente almacena el primer objeto seleccionado en la variable `sel` y, a continuación, utiliza las propiedades `element.elementType` y `shape.isGroup` para determinar si el elemento seleccionado es un grupo:

```
var sel = fl.getDocumentDOM().selection[0];
var shapeGroup = (sel.elementType == "shape") && sel.isGroup;
fl.trace(shapeGroup);
```

Véase también

`shape.isDrawingObject`

shape.isOvalObject

Disponibilidad

Flash CS3 Professional

Uso

```
shape.isOvalObject
```

Descripción

Propiedad de sólo lectura; si es `true`, la forma es un objeto Oval sencillo (creado con la herramienta Óvalo sencillo).

Ejemplo

El ejemplo siguiente muestra `"true"` si el primer elemento seleccionado es un objeto Oval sencillo, y `"false"` si no lo es:

```
var sel = fl.getDocumentDOM().selection[0];
fl.trace(sel.isOvalObject);
```

Véase también

`shape.isRectangleObject`

shape.isRectangleObject

Disponibilidad

Flash CS3 Professional

Uso

```
shape.isRectangleObject
```

Descripción

Propiedad de sólo lectura; si es `true`, la forma es un objeto Rectangle sencillo (creado con la herramienta Rectángulo sencillo).

Ejemplo

El ejemplo siguiente muestra "`true`" si el primer elemento seleccionado es un objeto Rectangle sencillo, y "`false`" si no lo es:

```
var sel = fl.getDocumentDOM().selection[0];
fl.trace(sel.isRectangleObject);
```

Véase también

`shape.isOvalObject`

shape.members

Disponibilidad

Flash CS4 Professional

Uso

```
shape.members
```

Descripción

Propiedad de sólo lectura; un conjunto de objetos del grupo seleccionado actualmente. Esta propiedad sólo está disponible si el valor de la propiedad `shape.isGroup` es `true`. Las formas sin procesar en el grupo no se incluyen en el conjunto `shape.members`.

Por ejemplo, si el grupo contiene tres objetos de dibujo y tres formas sin procesar, el conjunto `shape.members` contiene tres entradas: una para cada objeto de dibujo. Si el grupo sólo contuviese formas sin procesar, el conjunto estaría vacío.

Ejemplo

El siguiente código muestra el número de segmentos cúbicos de cada objeto de dibujo en el grupo seleccionado actualmente:

```
var shapesArray = fl.getDocumentDOM().selection[0].members;
for (i=0; i<shapesArray.length; i++) {
    fl.trace(shapesArray[i].numCubicSegments);
}
```

Véase también

`shape.isGroup`

shape.numCubicSegments

Disponibilidad

Flash CS4 Professional

Uso

```
shape.numCubicSegments
```

Descripción

Propiedad de sólo lectura; el número de segmentos cúbicos de la forma.

Ejemplo

Si damos por hecho que se selecciona una forma cuadrada o rectangular, el siguiente código muestra “4” en el panel Salida:

```
var theShape = fl.getDocumentDOM().selection[0];
fl.trace(theShape.numCubicSegments);
```

shape.vertices

Disponibilidad

Flash MX 2004

Uso

```
shape.vertices
```

Descripción

Propiedad de sólo lectura; un conjunto de objetos Vertex (consulte Objeto Vertex).

Ejemplo

El ejemplo siguiente almacena el primer objeto seleccionado en la variable `someShape` y, a continuación, muestra el número de vértices de ese objeto en el panel Salida:

```
var someShape = fl.getDocumentDOM().selection[0];
fl.trace("The shape has " + someShape.vertices.length + " vertices.");
```

Capítulo 39: Objeto SoundItem

Herencia Objeto Item > Objeto SoundItem

Disponibilidad

Flash MX 2004

Descripción

El objeto SoundItem es una subclase del objeto Item. Representa un elemento de biblioteca utilizado para crear un sonido. Consulte también `frame.soundLibraryItem` y el Objeto Item.

Resumen de métodos

Además de los métodos del objeto Item, el objeto SoundItem dispone del siguiente método:

Propiedad	Descripción
`soundItem.exportToFile()`	Exporta el elemento especificado a un archivo de QuickTime en Macintosh, o a un archivo WAV o QT en Windows.

Resumen de propiedades

Además de las propiedades del objeto Item, el objeto SoundItem dispone de las siguientes:

Propiedad	Descripción
`soundItem.bitRate`	Cadena que especifica la velocidad de transmisión de un sonido de la biblioteca. Disponible únicamente para el tipo de compresión MP3.
`soundItem.bits`	Cadena que especifica el valor en bits de un sonido de la biblioteca con compresión ADPCM.
`soundItem.compressionType`	Cadena que especifica el tipo de compresión de un sonido de la biblioteca.
`soundItem.convertStereoToMono`	Valor booleano disponible sólo para tipos de compresión MP3 y Raw.
`soundItem.fileLastModifiedDate`	Sólo lectura; una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la fecha de modificación del archivo original (en el disco) en el momento en que se importó a la biblioteca.
`soundItem.originalCompressionType`	Sólo lectura; una cadena que especifica si el elemento especificado se ha importado como un archivo MP3.
`soundItem.quality`	Cadena que especifica la calidad de reproducción de un sonido de la biblioteca. Disponible únicamente para el tipo de compresión MP3.
`soundItem.sampleRate`	Cadena que especifica la velocidad de muestreo del clip de audio.
`soundItem.sourceFileExists`	Sólo lectura; un valor booleano que especifica si el archivo importado en la biblioteca sigue existiendo en su ubicación original.

Propiedad	Descripción
soundItem.sourceFileIsCurrent	Sólo lectura; un valor booleano que especifica si la fecha de modificación del archivo en el elemento de la biblioteca coincide con la fecha de modificación en disco del archivo importado.
soundItem.sourceFilePath	Sólo lectura; una cadena, especificada como URI file:/// que especifica la ruta y el nombre del archivo importado en la biblioteca.
soundItem.useImportedMP3Quality	Valor booleano; si es `true`, se omitirán todas las demás propiedades y se utilizará la calidad del MP3 importado.

soundItem.bitRate

Disponibilidad

Flash MX 2004

Uso

```
soundItem.bitRate
```

Descripción

Propiedad; una cadena que especifica la velocidad de transmisión de un sonido de la biblioteca. Esta propiedad sólo está disponible para el tipo de compresión MP3. Los valores aceptables son `"8 kbps"`, `"16 kbps"`, `"20 kbps"`, `"24 kbps"`, `"32 kbps"`, `"48 kbps"`, `"56 kbps"`, `"64 kbps"`, `"80 kbps"`, `"112 kbps"`, `"128 kbps"` y `"160 kbps"`. Los sonidos estéreo exportados a 8 ó 16 Kbps se convierten en mono. La propiedad es `undefined` para otros tipos de compresión.

Si desea especificar un valor para esta propiedad, defina `soundItem.useImportedMP3Quality` como `false`.

Ejemplo

El ejemplo siguiente muestra el valor `bitRate` en el panel Salida si el elemento especificado en la biblioteca tiene compresión MP3:

```
alert(fl.getDocumentDOM().library.items[0].bitRate);
```

Véase también

`soundItem.compressionType`, `soundItem.convertStereoToMono`

soundItem.bits

Disponibilidad

Flash MX 2004

Uso

```
soundItem.bits
```

Descripción

Propiedad; una cadena que especifica el valor en bits de un sonido de la biblioteca con compresión ADPCM. Los valores aceptables son `"2 bit"`, `"3 bit"`, `"4 bit"` y `"5 bit"`.

Si desea especificar un valor para esta propiedad, defina `soundItem.useImportedMP3Quality` como `false`.

Ejemplo

El ejemplo siguiente muestra el valor en bits en el panel Salida si el elemento seleccionado actualmente en la biblioteca tiene compresión ADPCM:

```
alert(fl.getDocumentDOM().library.items[0].bits);
```

Véase también

`soundItem.compressionType`

soundItem.compressionType

Disponibilidad

Flash MX 2004

Uso

```
soundItem.compressionType
```

Descripción

Propiedad; una cadena que especifica el tipo de compresión de un sonido de la biblioteca. Los valores aceptables son `"Default"`, `"ADPCM"`, `"MP3"`, `"Raw"` y `"Speech"`.

Si desea especificar un valor para esta propiedad, defina `soundItem.useImportedMP3Quality` como `false`.

Ejemplo

El ejemplo siguiente cambia un elemento de la biblioteca al tipo de compresión`Raw`:

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

El siguiente ejemplo cambia el tipo de compresión de los elementos de biblioteca seleccionados a `Speech`:

```
fl.getDocumentDOM().library.getSelectedItems().compressionType = "Speech";
```

Véase también

`soundItem.originalCompressionType`

soundItem.convertStereoToMono

Disponibilidad

Flash MX 2004

Uso

```
soundItem.convertStereoToMono
```

Descripción

Propiedad; un valor booleano disponible sólo para tipos de compresión MP3 y Raw. Si se define este valor como `true`, se convertirá un sonido estéreo en mono; `false` lo dejará como estéreo. Para el tipo de compresión MP3, si `soundItem.bitRate` es menor que 20 Kbps, se omitirá esta propiedad y se impondrá como `true` (consulte `soundItem.bitRate`).

Si desea especificar un valor para esta propiedad, defina `soundItem.useImportedMP3Quality` como `false`.

Ejemplo

El ejemplo siguiente convierte un elemento de la biblioteca en mono sólo si el elemento tiene el tipo de compresión MP3 o Raw:

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

Véase también

`soundItem.compressionType`

soundItem.exportToFile()

Disponibilidad

Flash CS4 Professional

Uso

```
soundItem.exportToFile(fileURI)
```

Parámetros

`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta y el nombre del archivo exportado.

Valor devuelto

Un valor booleano de `true` si se exporta correctamente el archivo, y de `false` en caso contrario.

Descripción

Método; exporta el elemento especificado a un archivo de QuickTime en Macintosh, o a un archivo WAV o QT en Windows. Los archivos QuickTime o QT exportados sólo contienen audio; el vídeo no se exporta. La configuración de la exportación se basa en el elemento que se esté exportando.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de sonido, el siguiente código lo exporta como un archivo WAV:

```
var soundFileURL = "file:///C|/out.wav";
var libItem = fl.getDocumentDOM().library.items[0];
libItem.exportToFile(soundFileURL);
```

soundItem.fileLastModifiedDate

Disponibilidad

Flash CS4 Professional

Uso

```
soundItem.fileLastModifiedDate
```

Descripción

Propiedad de sólo lectura; una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la fecha de modificación del archivo original (en el disco) en el momento en que se importó a la biblioteca. Si el archivo ya no existe, el valor es "00000000".

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de sonido, el siguiente código muestra un número hexadecimal, tal como se describe más arriba.

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

Véase también

soundItem.sourceFileExists, soundItem.sourceFileIsCurrent, soundItem.sourceFilePath, FLfile.getModificationDate()

soundItem.originalCompressionType

Disponibilidad

Flash CS4 Professional

Uso

```
soundItem.originalCompressionType
```

Descripción

Propiedad de sólo lectura; una cadena que especifica si el elemento especificado se ha importado como un archivo mp3. Los posibles valores de esta propiedad son “RAW” y “MP3”.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de sonido, el siguiente código muestra "MP3" si el archivo se importó en la biblioteca como archivo mp3, o muestra "RAW" en caso contrario:

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

Véase también

soundItem.compressionType

soundItem.quality

Disponibilidad
Flash MX 2004

Uso
`soundItem.quality`

Descripción
Propiedad; una cadena que especifica la calidad de reproducción de un sonido de la biblioteca. Esta propiedad sólo está disponible para el tipo de compresión MP3. Los valores aceptables son `"Fast"`, `"Medium"` y `"Best"`.

Si desea especificar un valor para esta propiedad, defina `soundItem.useImportedMP3Quality` como `false`.

Ejemplo
El ejemplo siguiente define la calidad de reproducción de un elemento de la biblioteca como `Best` si dicho elemento tiene el tipo de compresión MP3:

```
fl.getDocumentDOM().library.items[0].quality = "Best";
```

Véase también
`soundItem.compressionType`

soundItem.sampleRate

Disponibilidad
Flash MX 2004

Uso
`soundItem.sampleRate`

Descripción
Propiedad; una cadena que especifica la velocidad de muestreo del clip de audio. Esta propiedad sólo está disponible para los tipos de compresión ADPCM, Raw y Speech. Los valores aceptables son `"5 kHz"`, `"11 kHz"`, `"22 kHz"` y `"44 kHz"`.

Si desea especificar un valor para esta propiedad, defina `soundItem.useImportedMP3Quality` como `false`.

Ejemplo
El ejemplo siguiente define la velocidad de muestreo de un elemento de la biblioteca como 5 kHz si el elemento tiene compresión ADPCM, Raw o Speech:

```
fl.getDocumentDOM().library.items[0].sampleRate = "5 kHz";
```

Véase también
`soundItem.compressionType`

soundItem.sourceFileExists

Disponibilidad

Flash CS4 Professional

Uso

```
soundItem.sourceFileExists
```

Descripción

Propiedad de sólo lectura; un valor booleano `true` si el archivo importado en la biblioteca sigue existiendo en su ubicación original; `false` en caso contrario.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de sonido, el siguiente código muestra "true" si el archivo se importó en la biblioteca sigue existiendo.

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

Véase también

`soundItem.sourceFileIsCurrent`, `soundItem.sourceFilePath`

soundItem.sourceFileIsCurrent

Disponibilidad

Flash CS4 Professional

Uso

```
soundItem.sourceFileIsCurrent
```

Descripción

Propiedad de sólo lectura; un valor booleano `true` si la fecha de modificación del archivo en el elemento de la biblioteca coincide con la fecha de modificación en disco del archivo importado; `false` en caso contrario.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de sonido, el siguiente código muestra "true" si el archivo importado no se ha modificado en el disco desde su importación.

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

Véase también

`soundItem.fileLastModifiedDate`, `soundItem.sourceFilePath`

soundItem.sourceFilePath

Disponibilidad

Flash CS4 Professional

Uso

`soundItem.sourceFilePath`

Descripción

Propiedad de sólo lectura; una cadena, especificada como URI file:/// que especifica la ruta y el nombre del archivo importado en la biblioteca.

Ejemplo

El ejemplo siguiente muestra el nombre y la ruta del archivo de origen de todos los elementos de la biblioteca que sean del tipo `"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);
    }
}
```

Véase también

`soundItem.sourceFileExists`

soundItem.useImportedMP3Quality

Disponibilidad

Flash MX 2004

Uso

`soundItem.useImportedMP3Quality`

Descripción

Propiedad; un valor booleano. Si es `true`, se omitirán todas las demás propiedades y se utilizará la calidad del MP3 importado.

Ejemplo

El ejemplo siguiente define un elemento de la biblioteca para utilizar la calidad del MP3 importado:

```
fl.getDocumentDOM().library.items[0].useImportedMP3Quality = true;
```

Véase también

`soundItem.compressionType`

Capítulo 40: Objeto Stroke

Disponibilidad

Flash MX 2004

Descripción

El objeto Stroke contiene toda la configuración de un trazo, incluida la configuración personalizada. Este objeto representa la información que contiene el inspector de propiedades. Utilizando el objeto Stroke con el método `document.setCustomStroke()` puede cambiar la configuración de trazo para el panel Herramientas, el inspector de propiedades y la selección actual. También puede obtener la configuración de trazo del panel Herramientas y del inspector de propiedades, o de la selección actual, empleando el método `document.getCustomStroke()`.

Este objeto siempre tiene las cuatro propiedades siguientes: `style`, `thickness`, `color` y `breakAtCorners`. (En Flash CS3, la propiedad `breakAtCorners` estaba desfasada y en su lugar se emplea `stroke.joinType`.) Se pueden definir otras propiedades, según el valor de la propiedad `stroke.style`.

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto Stroke:

Propiedad	Descripción
`stroke.breakAtCorners`	Valor booleano, equivalente a la opción Esquinas Marcadas del cuadro de diálogo Estilo del Trazo personalizado.
`stroke.capType`	Una cadena que especifica el tipo de extremo del trazo.
`stroke.color`	Una cadena, valor hexadecimal o entero que representa el color de trazo.
`stroke.curve`	Una cadena que especifica el tipo de trama del trazo.
`stroke.dash1`	Un entero que especifica las longitudes de la parte sólida de una línea discontinua.
`stroke.dash2`	Un entero que especifica las longitudes de la parte en blanco de una línea discontinua.
`stroke.density`	Una cadena que especifica la densidad de una línea punteada.
`stroke.dotSize`	Una cadena que especifica el tamaño de punto de una línea punteada.
`stroke.dotSpace`	Un entero que especifica el espaciado entre puntos en una línea de puntos.
`stroke.hatchThickness`	Una cadena que especifica el grosor de una línea de sombreado.
`stroke.jiggle`	Una cadena que especifica la propiedad de vaivén de una línea de sombreado.
`stroke.joinType`	Una cadena que especifica el tipo de unión del trazo.
`stroke.length`	Una cadena que especifica la longitud de una línea de sombreado.
`stroke.miterLimit`	Un valor flotante que especifica el ángulo sobre el cual se truncará la punta del angular en un segmento.
`stroke.pattern`	Una cadena que especifica el patrón de una línea no justificada.
`stroke.rotate`	Una cadena que especifica la rotación de una línea de sombreado.
`stroke.scaleType`	Una cadena que especifica el tipo de escala que se aplicará al trazo.
`stroke.shapeFill`	Un Objeto Fill que representa el valor de relleno del trazo.

Propiedad	Descripción
`stroke.space`	Una cadena que especifica el espaciado de una línea de sombreado.
`stroke.strokeHinting`	Un valor booleano que especifica si se definen sugerencias en el trazo.
`stroke.style`	Una cadena que describe el estilo de trazo.
`stroke.thickness`	Un entero que especifica el tamaño del trazo.
`stroke.variation`	Una cadena que especifica la variación de una línea punteada.
`stroke.waveHeight`	Una cadena que especifica la altura de onda de una línea no justificada.
`stroke.waveLength`	Una cadena que especifica la longitud de onda de una línea no justificada.

stroke.breakAtCorners

Disponibilidad

Flash MX 2004 Desfasada en Flash; en su lugar se emplea `stroke.joinType`.

Uso

```
stroke.breakAtCorners
```

Descripción

Propiedad; un valor booleano. Esta propiedad equivale a la opción Esquinas Marcadas del cuadro de diálogo Estilo del Trazo personalizado.

Ejemplo

El ejemplo siguiente define la propiedad `breakAtCorners` como `true`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.breakAtCorners = true;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.capType

Disponibilidad

Flash 8

Uso

```
stroke.capType
```

Descripción

Propiedad; una cadena que especifica el tipo de extremo del trazo. Los valores aceptables son `"none"`, `"round"` y `"square"`.

Ejemplo

El ejemplo siguiente define el tipo de extremo del trazo como `round`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.capType = "round";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.color

Disponibilidad

Flash MX 2004 En Flash 8 y versiones posteriores, esta propiedad queda desfasada y en su lugar se emplea `stroke.shapeFill.color`.

Uso

`stroke.color`

Descripción

Propiedad; el color del trazo, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

Ejemplo

El ejemplo siguiente define el color del trazo:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.color = "#000000";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

Véase también

`stroke.shapeFill`

stroke.curve

Disponibilidad

Flash MX 2004

Uso

`stroke.curve`

Descripción

Propiedad; una cadena que especifica el tipo de trama del trazo. Esta propiedad sólo se puede definir si la propiedad `stroke.style` es `"hatched"` (consulte `stroke.style`). Los valores aceptables son `"straight"`, `"slight curve"`, `"medium curve"` y `"very curved"`.

Ejemplo

El ejemplo siguiente define la propiedad de la curva, entre otras, para un trazo que tiene el estilo `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

Disponibilidad

Flash MX 2004

Uso

`stroke.dash1`

Descripción

Propiedad; un entero que especifica las longitudes de las partes sólidas de una línea discontinua. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `dashed`(consulte `stroke.style`).

Ejemplo

El ejemplo siguiente define las propiedades `dash1` y `dash2` para un estilo de trazo de `dashed`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "dashed";
myStroke.dash1 = 1;
myStroke.dash2 = 2;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.dash2

Disponibilidad

Flash MX 2004

Uso

`stroke.dash2`

Descripción

Propiedad; un entero que especifica las longitudes de las partes en blanco de una línea discontinua. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `dashed` (consulte `stroke.style`).

Ejemplo

Consulte `stroke.dash1`.

stroke.density

Disponibilidad

Flash MX 2004

Uso

`stroke.density`

Descripción

Propiedad; una cadena que especifica la densidad de una línea punteada. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `stipple` (consulte `stroke.style`). Los valores aceptables son `"very dense"`, `"dense"`, `"sparse"` y `"very sparse"`.

Ejemplo

El ejemplo siguiente define la propiedad de densidad como `sparse` para el estilo de trazo de `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

Disponibilidad

Flash MX 2004

Uso

`stroke.dotSize`

Descripción

Propiedad; una cadena que especifica el tamaño de punto de una línea punteada. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `stipple` (consulte `stroke.style`). Los valores aceptables son `"tiny"`, `"small"`, `"medium"` y `"large"`.

El ejemplo siguiente define la propiedad `dotsize` como `tiny` para el estilo de trazo 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

Disponibilidad

Flash MX 2004

Uso

`stroke.dotSpace`

Descripción

Propiedad; un entero que especifica el espaciado entre puntos en una línea de puntos. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `dotted`. Consulte `stroke.style`.

Ejemplo

El ejemplo siguiente define la propiedad `dotSpace` como 3 para un estilo de trazo de `dotted`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "dotted";
myStroke.dotSpace= 3;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.hatchThickness

Disponibilidad

Flash MX 2004

Uso

`stroke.hatchThickness`

Descripción

Propiedad; una cadena que especifica el grosor de una línea de sombreado. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `hatched` (consulte `stroke.style`). Los valores aceptables son `"hairline"`, `"thin"`, `"medium"` y `"thick"`.

Ejemplo

El ejemplo siguiente define la propiedad `hatchThickness` como `thin` para un estilo de trazo de `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

Disponibilidad

Flash MX 2004

Uso

`stroke.jiggle`

Descripción

Propiedad; una cadena que especifica la propiedad de vaivén de una línea de sombreado. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `hatched` (consulte `stroke.style`). Los valores aceptables son `"none"`, `"bounce"`, `"loose"` y `"wild"`.

Ejemplo

El ejemplo siguiente define la propiedad `jiggle` como `wild` para un estilo de trazo de `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

Disponibilidad

Flash 8

Uso

`stroke.joinType`

Descripción

Propiedad; una cadena que especifica el tipo de unión del trazo. Los valores aceptables son `"miter"`, `"round"` y `"bevel"`.

Véase también

`stroke.capType`

stroke.length

Disponibilidad

Flash MX 2004

Uso

`stroke.length`

Descripción

Propiedad; una cadena que especifica la longitud de una línea de sombreado. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `hatched` (consulte `stroke.style`). Los valores aceptables son `"equal"`, `"slight"`, `"variation"`, `"medium variation"` y `"random"`.

Ejemplo

El ejemplo siguiente define la propiedad `length` como `slight` para un estilo de trazo de `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.miterLimit

Disponibilidad

Flash 8

Uso

`stroke.miterLimit`

Descripción

Propiedad; un valor flotante que especifica el ángulo sobre el cual se truncará la punta del angular en un segmento. Esto significa que el angular sólo se trunca si su ángulo es mayor que el valor de `miterLimit`.

Ejemplo

El ejemplo siguiente cambia el límite del ángulo del trazo a 3. Si el ángulo es superior a 3, el angular se trunca.

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.miterLimit = 3;
var myStroke = fl.getDocumentDOM().setCustomStroke();
```

stroke.pattern

Disponibilidad

Flash MX 2004

Uso

`stroke.pattern`

Descripción

Propiedad; una cadena que especifica el patrón de una línea no justificada. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `ragged` (consulte `stroke.style`). Los valores aceptables son `"solid"`, `"simple"`, `"random"`, `"dotted"`, `"random dotted"`, `"triple dotted"` y `"random triple dotted"`.

Ejemplo

El ejemplo siguiente define la propiedad `pattern` como `random` para un estilo de trazo de `ragged`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
myStroke.pattern = "random";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.rotate

Disponibilidad

Flash MX 2004

Uso

```
stroke.rotate
```

Descripción

Propiedad; una cadena que especifica la rotación de una línea de sombreado. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `hatched` (consulte `stroke.style`). Los valores aceptables son `"none"`, `"slight"`, `"medium"` y `"free"`.

Ejemplo

El ejemplo siguiente define la propiedad `rotate` como `free` para un estilo de trazo de `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

Disponibilidad

Flash 8

Uso

```
stroke.scaleType
```

Descripción

Propiedad; una cadena que especifica el tipo de escala que se aplicará al trazo. Los valores aceptables son `"normal"`, `"horizontal"`, `"vertical"` y `"none"`.

Ejemplo

El ejemplo siguiente define el tipo de escala del trazo como `horizontal`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.scaleType = "horizontal";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.shapeFill

Disponibilidad

Flash 8

Uso

```
stroke.shapeFill
```

Descripción

Propiedad; un Objeto Fill que representa el valor de relleno del trazo.

Ejemplo

El ejemplo siguiente especifica la configuración de relleno y, a continuación, la aplica al trazo:

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.linearGradient = true;
fill.colorArray = [ 00ff00, ff0000, fffff ];
var stroke = fl.getDocumentDOM().getCustomStroke();
stroke.shapeFill = fill;
fl.getDocumentDOM().setCustomStroke(stroke);
```

stroke.space

Disponibilidad

Flash MX 2004

Uso

```
stroke.space
```

Descripción

Propiedad; una cadena que especifica el espaciado de una línea de sombreado. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `hatched` (consulte `stroke.style`). Los valores aceptables son `"very close"`, `"close"`, `"distant"` y `"very distant"`.

Ejemplo

El ejemplo siguiente define la propiedad `space` como `close` para un estilo de trazo de `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

Disponibilidad
Flash 8

Uso
`stroke.strokeHinting`

Descripción
Propiedad; un valor booleano que especifica si se definen sugerencias en el trazo.

Ejemplo
El ejemplo siguiente activa las sugerencias para el trazo:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.strokeHinting = true;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.style

Disponibilidad
Flash MX 2004

Uso
`stroke.style`

Descripción
Propiedad; una cadena que describe el estilo de trazo. Los valores aceptables son `"noStroke"`, `"solid"`, `"dashed"`, `"dotted"`, `"ragged"`, `"stipple"` y `"hatched"`. Algunos de estos valores requieren la definición de propiedades adicionales del objeto stroke, como se describe en la lista siguiente:

- Si el valor es `"solid"` o `"noStroke"`, no hay otras propiedades.
- Si el valor es `"dashed"`, hay dos propiedades adicionales: `dash1` y `dash2`.
- Si el valor es `"dotted"`, hay una propiedad adicional: `dotSpace`.
- Si el valor es `"ragged"`, hay tres propiedades adicionales: `pattern`, `waveHeight` y `waveLength`.
- Si el valor es `"stipple"`, hay tres propiedades adicionales:`dotSize`, `variation` y `density`.

- Si el valor es `"hatched"`, hay seis propiedades adicionales: `hatchThickness`, `space`, `jiggle`, `"rotate"`, `curve` y `length`.

Ejemplo

El ejemplo siguiente define el estilo de trazo como `ragged`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.thickness

Disponibilidad

Flash MX 2004

Uso

```
stroke.thickness
```

Descripción

Propiedad; un entero que especifica el tamaño del trazo.

Ejemplo

El ejemplo siguiente define la propiedad `thickness` del trazo con un valor de `2`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.thickness = 2;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.variation

Disponibilidad

Flash MX 2004

Uso

```
stroke.variation
```

Descripción

Propiedad; una cadena que especifica la variación de una línea punteada. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `stipple` (consulte `stroke.style`). Los valores aceptables son `"one size"`, `"small variation"`, `"varied sizes"` y `"random sizes"`.

Ejemplo

El ejemplo siguiente define la propiedad de variación como `random sizes` para el estilo de trazo de `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

Disponibilidad

Flash MX 2004

Uso

`stroke.waveHeight`

Descripción

Propiedad; una cadena que especifica la altura de onda de una línea no justificada. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `ragged` (consulte `stroke.style`). Los valores aceptables son `"flat"`, `"wavy"`, `"very wavy"` y `"wild"`.

Ejemplo

El ejemplo siguiente define la propiedad `waveHeight` como `flat` para un estilo de trazo de `ragged`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
myStroke.pattern = "random";
myStroke.waveHeight = "flat";
myStroke.waveLength = "short";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.waveLength

Disponibilidad

Flash MX 2004

Uso

`stroke.waveLength`

Descripción

Propiedad; una cadena que especifica la longitud de onda de una línea no justificada. Esta propiedad sólo está disponible si la propiedad `stroke.style` se define como `ragged` (consulte `stroke.style`). Los valores aceptables son `"very short"`, `"short"`, `"medium"` y `"long"`.

Ejemplo

El ejemplo siguiente define la propiedad `waveLength` como `short` para un estilo de trazo de `ragged`:

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
myStroke.pattern = "random";
myStroke.waveHeight = 'flat";
myStroke.waveLength = "short";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

Capítulo 41: Objeto swfPanel

Disponibilidad
Flash CS4 Professional

Descripción
El objeto swfPanel representa un panel SWF de ventana. Los paneles SWF de ventana son archivos SWF que implementan aplicaciones que se pueden ejecutar desde el entorno de edición de Flash; se encuentran disponibles en el menú Ventana> Otros paneles. De forma predeterminada, los paneles SWF de ventana se almacenan en una subcarpeta de la carpeta de configuración (consulte "Almacenamiento de archivos JSFL" en la página 2). Por ejemplo, en Windows, la carpeta se encuentra en *unidad de arranque*\Documents and Settings*usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\WindowSWF. Hay disponible un panel SWF de ventana de ejemplo; consulte "Panel Trazar mapa de bits de muestra" en la página 14. El conjunto de paneles SWF de ventana registrados se almacena en la propiedad `fl.swfPanels`.

Resumen de métodos
Puede emplear el método siguiente con el objeto swfPanel:

Método	Descripción
`swfPanel.call()`	Funciona en combinación con los métodos `ExternalInterface.addCallback()` y `MMExecute()` de ActionScript para comunicarse con el panel SWF del entorno de edición.

Resumen de propiedades
Puede emplear las propiedades siguientes con el objeto swfPanel:

Propiedad	Descripción
swfPanel.name	Sólo lectura; una cadena que representa el nombre del panel SWF de ventana especificado.
swfPanel.path	Sólo lectura; una cadena que representa la ruta del archivo SWF utilizado en el panel SWF de ventana especificado.

swfPanel.call()

Disponibilidad
Flash CS4 Professional

Uso
```
swfPanel.call(request)
```

Parámetros
`request` Parámetros que se transfieren a la función (consulte la descripción y el ejemplo a continuación).

Valor devuelto
`null` o una cadena devuelta por la llamada a la función. El resultado de la función puede ser una cadena vacía.

Descripción

Método; funciona en combinación con los métodos `ExternalInterface.addCallback()` y `MMExecute()` de ActionScript para comunicarse con el panel SWF del entorno de edición.

Ejemplo

En el siguiente ejemplo se muestra cómo utilizar código ActionScript y JavaScript para crear un panel SWF de ventana y comunicarse con el mismo desde el entorno de edición.

1 Cree un archivo FLA de ActionScript 3.0, defina su color como gris medio y establezca su tamaño en 400 píxeles de ancho y 250 píxeles de alto.

2 Coloque un cuadro de texto dinámico en el centro del escenario, establezca su nombre de instancia en `myTextField` y escriba la palabra "Status" en el cuadro de texto.

3 Defina otras propiedades del cuadro de texto similares a las siguientes:

- Centro alineado.
- 355 píxeles de ancho y 46 píxeles de alto.
- Fuente Times New Roman, 28 puntos, rojo.

4 Añada el siguiente código ActionScript:

```
// 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 Guarde el archivo como fileStatus.fla y publique el archivo SWF con la configuración de publicación predeterminada.

6 Cierre Flash.

7 Copie el archivo fileStatus.swf en la carpeta WindowSWF, que es una subcarpeta de la carpeta de configuración (consulte "Almacenamiento de archivos JSFL" en la página 2). Por ejemplo, en Windows, la carpeta se encuentra en *unidad de arranque*\Documents and Settings*usuario*\Local Settings\Application Data\Adobe\Flash CS4*idioma*\Configuration\WindowSWF.

8 Inicie Flash.

9 Cree un archivo JSFL con el siguiente código:

```
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 Guarde el archivo JSFL en el mismo directorio que el archivo SWF, con el nombre fileOp.jsfl.

11 Seleccione Ventana > Otros paneles > fileStatus.

Ahora, conforme cree, abra y cierre archivos FLA, el panel fileStatus muestra un mensaje que indica la operación por la que ha optado.

swfPanel.name

Disponibilidad

Flash CS4 Professional

Uso

`swfPanel.name`

Descripción

Propiedad de sólo lectura; una cadena que representa el nombre del panel SWF de ventana especificado.

Ejemplo

El siguiente código muestra el nombre del primer panel SWF de ventana del panel Salida:

```
fl.trace(fl.swfPanels[0].name);
```

Véase también

`swfPanel.path`, `fl.swfPanels`

swfPanel.path

Disponibilidad

Flash CS4 Professional

Uso

```
swfPanel.path
```

Descripción

Propiedad de sólo lectura; una cadena que representa la ruta del archivo SWF utilizado en el panel SWF de ventana especificado.

Ejemplo

El siguiente código muestra la ruta del archivo SWF utilizado en el primer panel SWF de ventana registrado en el panel Salida:

```
fl.trace(fl.swfPanels[0].path);
```

Véase también

`swfPanel.name`, `fl.swfPanels`

Capítulo 42: Objeto SymbolInstance

Herencia Objeto Element > Objeto Instance > Objeto SymbolInstance

Disponibilidad

Flash MX 2004

Descripción

SymbolInstance es una subclase del objeto Instance y representa un símbolo en un fotograma (consulte Objeto Instance).

Resumen de propiedades

Además de las propiedades del objeto Instance, el objeto SymbolInstance tiene las siguientes:

Propiedad	Descripción
`symbolInstance.accName`	Una cadena que equivale al campo Nombre del panel Accesibilidad.
`symbolInstance.actionScript`	Una cadena que especifica las acciones asignadas al símbolo.
`symbolInstance.blendMode`	Una cadena que especifica el modo de mezcla que se aplica a un símbolo de clip de película.
`symbolInstance.buttonTracking`	Una cadena que define (sólo para símbolos de botón) la misma propiedad que el menú emergente para Seguimiento como botón o Seguimiento como elemento de menú en el Inspector de propiedades.
`symbolInstance.cacheAsBitmap`	Un valor booleano que especifica si está activada la caché de mapa de bits en tiempo de ejecución.
`symbolInstance.colorAlphaAmount`	Un entero que forma parte de la transformación de color de la instancia, especificando la configuración de Efecto avanzado Alfa; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades y ajustar los controles de la derecha del cuadro de diálogo.
`symbolInstance.colorAlphaPercent`	Un entero que especifica parte de la transformación de color para la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la parte izquierda del cuadro de diálogo).
`symbolInstance.colorBlueAmount`	Un entero que forma parte de la transformación de color de la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia.
`symbolInstance.colorBluePercent`	Un entero que forma parte de la transformación de color para la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la parte izquierda del cuadro de diálogo).
`symbolInstance.colorGreenAmount`	Un entero que forma parte de la transformación de color de la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia. Los valores válidos están comprendidos entre -255 y 255.
`symbolInstance.colorGreenPercent`	Parte de la transformación de color para la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la parte izquierda del cuadro de diálogo).
`symbolInstance.colorMode`	Una cadena que especifica el modo de color identificado en el menú emergente Color del inspector de propiedades de símbolo.

Propiedad	Descripción
`symbolInstance.colorRedAmount`	Un entero que forma parte de la transformación de color de la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia.
`symbolInstance.colorRedPercent`	Parte de la transformación de color para la instancia; equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la parte izquierda del cuadro de diálogo).
`symbolInstance.description`	Una cadena que equivale al campo Descripción del panel Accesibilidad.
`symbolInstance.filters`	Conjunto de objetos Filter (consulte Objeto Filter).
`symbolInstance.firstFrame`	Un entero basado en cero que especifica el primer fotograma que aparecerá en la línea de tiempo del gráfico.
`symbolInstance.forceSimple`	Un valor booleano que activa o desactiva la accesibilidad de los elementos secundarios del objeto; equivale a la lógica inversa de la opción Hacer que los objetos secundarios sean accesibles del panel Accesibilidad.
`symbolInstance.loop`	Una cadena que define (para símbolos gráficos) la misma propiedad que el menú emergente Reproducir indefinidamente en el Inspector de propiedades.
`symbolInstance.shortcut`	Una cadena que equivale a la tecla de método abreviado asociada al símbolo; equivale al campo Método abreviado del panel Accesibilidad.
`symbolInstance.silent`	Un valor booleano que activa o desactiva la accesibilidad del objeto; equivale a la lógica inversa de la opción Hacer que el objeto sea accesible del panel Accesibilidad.
`symbolInstance.symbolType`	Una cadena que especifica el tipo de símbolo; equivale al valor de Comportamiento en los cuadros de diálogo Crear nuevo símbolo y Convertir en símbolo.
`symbolInstance.tabIndex`	Un entero que equivale al campo Índice de fichas del panel Accesibilidad.

symbolInstance.accName

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.accName
```

Descripción

Propiedad; una cadena que equivale al campo Nombre del panel Accesibilidad. Los lectores de pantalla identifican los objetos mediante la lectura del nombre en voz alta. Esta propiedad no está disponible para símbolos gráficos.

Ejemplo

El ejemplo siguiente almacena el valor del nombre del panel Accesibilidad del objeto en la variable `theName`:

```
var theName = fl.getDocumentDOM().selection[0].accName;
```

El ejemplo siguiente define el valor del nombre del panel Accesibilidad del objeto como `Home Button`:

```
fl.getDocumentDOM().selection[0].accName = "Home Button";
```

symbolInstance.actionScript

Disponibilidad
Flash MX 2004

Uso
`symbolInstance.actionScript`

Descripción
Propiedad; una cadena que especifica las acciones asignadas al símbolo. Sólo se aplica a instancias de botones y clips de película. Para una instancia de símbolos gráficos, el valor devuelve `undefined`.

Ejemplo
El ejemplo siguiente asigna una acción `onClipEvent` al primer elemento del primer fotograma de la primera capa de la línea de tiempo:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].actionScript
    = "onClipEvent(enterFrame) {trace('movie clip enterFrame');}";
```

symbolInstance.blendMode

Disponibilidad
Flash 8

Uso
`symbolInstance.blendMode`

Descripción
Propiedad; una cadena que especifica el modo de mezcla que se aplica a un símbolo de clip de película. Los valores aceptables son `"normal"`, `"layer"`, `"multiply"`, `"screen"`, `"overlay"`, `"hardlight"`, `"lighten"`, `"darken"`, `"difference"`, `"add"`, `"subtract"`, `"invert"`, `"alpha"` y `"erase"`.

Ejemplo
El ejemplo siguiente establece como `add` el modo de mezcla para el primer símbolo de clip de película en el primer fotograma del primer nivel:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].blendMode = "add";
```

Véase también
`document.setBlendMode()`

symbolInstance.buttonTracking

Disponibilidad
Flash MX 2004

Uso

```
symbolInstance.buttonTracking
```

Descripción

Propiedad; una cadena que define (sólo para símbolos de botón) la misma propiedad que el menú emergente para Seguimiento como botón o Seguimiento como elemento de menú en el Inspector de propiedades. Esta propiedad se ignora con otros tipos de símbolos. Los valores aceptables son `"button"` o `"menu"`.

Ejemplo

El ejemplo siguiente define el primer símbolo del primer fotograma de la primera capa de la línea de tiempo como Seguimiento como elemento de menú, siempre que ese símbolo sea un botón:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].buttonTracking = "menu";
```

symbolInstance.cacheAsBitmap

Disponibilidad

Flash 8

Uso

```
symbolInstance.cacheAsBitmap
```

Descripción

Propiedad; un valor booleano que especifica si está activada la caché de mapa de bits en tiempo de ejecución.

Ejemplo

El ejemplo siguiente activa la caché de mapa de bits en tiempo de ejecución para el primer elemento del primer fotograma de la primera capa:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].cacheAsBitmap = true;
```

symbolInstance.colorAlphaAmount

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.colorAlphaAmount
```

Descripción

Propiedad; un entero que forma parte de la transformación de color de la instancia, especificando la configuración de Efecto avanzado Alfa. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades y ajustar los controles de la derecha del cuadro de diálogo. Este valor reduce o aumenta los valores de tinta y alfa en una cantidad constante. Este valor se añade al valor actual. Esta propiedad es útil sobre todo si se utiliza con `symbolInstance.colorAlphaPercent`. Los valores válidos están comprendidos entre -255 y 255.

Ejemplo

El ejemplo siguiente resta 100 del valor de alfa de la instancia de símbolo seleccionada:

```
fl.getDocumentDOM().selection[0].colorAlphaAmount = -100;
```

symbolInstance.colorAlphaPercent

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.colorAlphaPercent
```

Descripción

Propiedad; un entero que especifica parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la izquierda del cuadro de diálogo). Este valor cambia los valores de tinta y alfa al porcentaje especificado. Los valores válidos están comprendidos entre -100 y 100. Véase también `symbolInstance.colorAlphaAmount`.

Ejemplo

El ejemplo siguiente define el `colorAlphaPercent` de la instancia de símbolo seleccionada como 80:

```
fl.getDocumentDOM().selection[0].colorAlphaPercent = 80;
```

symbolInstance.colorBlueAmount

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.colorBlueAmount
```

Descripción

Propiedad; un entero que forma parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia. Los valores válidos están comprendidos entre -255 y 255.

symbolInstance.colorBluePercent

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.colorBluePercent
```

Descripción

Propiedad; un entero que forma parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la izquierda del cuadro de diálogo). Este valor define los valores de azul en un porcentaje especificado. Los valores válidos están comprendidos entre -100 y 100.

Ejemplo

El ejemplo siguiente define el `colorBluePercent` de la instancia de símbolo seleccionada como 80:

```
fl.getDocumentDOM().selection[0].colorBluePercent = 80;
```

symbolInstance.colorGreenAmount

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.colorGreenAmount
```

Descripción

Propiedad; un entero que forma parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia. Los valores válidos están comprendidos entre -255 y 255.

symbolInstance.colorGreenPercent

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.colorGreenPercent
```

Descripción

Propiedad; forma parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la izquierda del cuadro de diálogo). Este valor define los valores de verde en un porcentaje especificado. Los valores válidos están comprendidos entre -100 y 100.

Ejemplo

El ejemplo siguiente define el `colorGreenPercent` de la instancia de símbolo seleccionada como 70:

```
fl.getDocumentDOM().selection[0].colorGreenPercent = 70;
```

symbolInstance.colorMode

Disponibilidad
Flash MX 2004

Uso
`symbolInstance.colorMode`

Descripción
Propiedad; una cadena que especifica el modo de color identificado en el menú emergente Color del inspector de propiedades de símbolo. Los valores aceptables son `"none"`, `"brightness"`, `"tint"`, `"alpha"` y `"advanced"`.

Ejemplo
El ejemplo siguiente cambia la propiedad `colorMode` del primer elemento del primer fotograma de la primera capa de la línea de tiempo a `alpha`:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].colorMode = "alpha";
```

symbolInstance.colorRedAmount

Disponibilidad
Flash MX 2004

Uso
`symbolInstance.colorRedAmount`

Descripción
Propiedad; un entero que forma parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia. Los valores válidos están comprendidos entre -255 y 255.

Ejemplo
El ejemplo siguiente define el `colorRedAmount` de la instancia de símbolo seleccionada como 255:

```
fl.getDocumentDOM().selection[0].colorRedAmount = 255;
```

symbolInstance.colorRedPercent

Disponibilidad
Flash MX 2004

Uso
`symbolInstance.colorRedPercent`

Descripción

Propiedad; forma parte de la transformación de color de la instancia. Esta propiedad equivale a utilizar la opción Color > Avanzado en el inspector de propiedades de la instancia (los controles de porcentaje de la izquierda del cuadro de diálogo). Este valor define los valores de rojo en un porcentaje especificado. Los valores válidos están comprendidos entre -100 y 100.

Ejemplo

El ejemplo siguiente define el `colorRedPercent` de la instancia de símbolo seleccionada como 10:

```
fl.getDocumentDOM().selection[0].colorRedPercent = 10;
```

symbolInstance.description

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.description
```

Descripción

Propiedad; una cadena que equivale al campo Descripción del panel Accesibilidad. El lector de pantalla lee esta descripción. Esta propiedad no está disponible para símbolos gráficos.

Ejemplo

El ejemplo siguiente almacena el valor de la descripción del panel Accesibilidad del objeto en la variable `theDescription`:

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

El ejemplo siguiente define el valor de la descripción del panel Accesibilidad como `Click the home button to go to home`:

```
fl.getDocumentDOM().selection[0].description= "Click the home button to go to home";
```

symbolInstance.filters

Disponibilidad

Flash 8

Uso

```
symbolInstance.filters
```

Descripción

Propiedad; un conjunto de objetos Filter (consulte Objeto Filter). Para modificar las propiedades de filtro, no se escribe en este conjunto directamente, sino que se debe recuperar el conjunto, definir las propiedades individuales y después definir el conjunto para que refleje las nuevas propiedades.

Ejemplo

El ejemplo siguiente traza el nombre del filtro en el índice 0. Si se trata de un filtro de iluminado, su propiedad `blurX` se establece como 100 y el nuevo valor se escribe en el conjunto de filtros.

```
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

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.firstFrame
```

Descripción

Propiedad; un entero basado en cero que especifica el primer fotograma que aparecerá en la línea de tiempo del gráfico. Esta propiedad sólo se aplica a símbolos gráficos y define la misma propiedad que el campo Primero del inspector de propiedades. Para otros tipos de símbolos, esta propiedad es `undefined`.

Ejemplo

El ejemplo siguiente especifica que el Fotograma 10 debe ser el primer fotograma que aparezca en la línea de tiempo del elemento especificado:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].firstFrame = 10;
```

symbolInstance.forceSimple

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.forceSimple
```

Descripción

Propiedad; un valor booleano que activa o desactiva la accesibilidad para los elementos secundarios del objeto. Esta propiedad equivale a la lógica inversa de la opción Hacer que los objetos secundarios sean accesibles del panel Accesibilidad. Por ejemplo, si `forceSimple` es `true`, equivale a la opción desactivada Hacer que los objetos secundarios sean accesibles. Si `forceSimple` es `false`, equivale a la opción activada Hacer que los objetos secundarios sean accesibles.

Esta propiedad sólo está disponible para objetos MovieClip.

Ejemplo

El ejemplo siguiente comprueba si los elementos secundarios del objeto son accesibles; un valor devuelto de `false` significa que los elementos secundarios son accesibles:

```
var areChildrenAccessible = fl.getDocumentDOM().selection[0].forceSimple;
```

El ejemplo siguiente permite que los elementos secundarios del objeto sean accesibles:

```
fl.getDocumentDOM().selection[0].forceSimple = false;
```

symbolInstance.loop

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.loop
```

Descripción

Propiedad; una cadena que define (para símbolos gráficos) la misma propiedad que el menú emergente Reproducir indefinidamente en el Inspector de propiedades. Para otros tipos de símbolos, esta propiedad es `undefined`. Los valores aceptables son `"loop"`, `"play once"` y `"single frame"` para definir la animación del gráfico en consonancia.

Ejemplo

El ejemplo siguiente define el primer símbolo del primer fotograma de la primera capa de la línea de tiempo como `fotograma único` (muestra un fotograma especificado de la línea de tiempo gráfica), siempre que ese símbolo sea un gráfico:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].loop = 'single frame';
```

symbolInstance.shortcut

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.shortcut
```

Descripción

Propiedad; una cadena que equivale a la tecla de método abreviado asociada al símbolo. Este propiedad equivale al campo Método abreviado del panel Accesibilidad. El lector de pantalla lee esta tecla. Esta propiedad no está disponible para símbolos gráficos.

Ejemplo

El ejemplo siguiente almacena el valor de la tecla de método abreviado del objeto en la variable `theShortcut`:

```
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;
```

El ejemplo siguiente define la tecla de método abreviado del objeto como `Ctrl+i`:

```
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+i";
```

symbolInstance.silent

Disponibilidad
Flash MX 2004

Uso
```
symbolInstance.silent
```

Descripción
Propiedad; un valor booleano que activa o desactiva la accesibilidad del objeto. Esta propiedad equivale a la lógica inversa de la opción Hacer que el objeto sea accesible del panel Accesibilidad. Por ejemplo, si `silent` es `true`, equivale a la opción desactivada Hacer que el objeto sea accesible. Si `silent` es `false`, equivale a la opción activada Hacer que el objeto sea accesible.

Esta propiedad no está disponible para objetos gráficos.

Ejemplo
El ejemplo siguiente comprueba si el objeto es accesible; un valor devuelto de `false` significa que el objeto es accesible:

```
var isSilent = fl.getDocumentDOM().selection[0].silent;
```

El ejemplo siguiente define el objeto como accesible:

```
fl.getDocumentDOM().selection[0].silent = false;
```

symbolInstance.symbolType

Disponibilidad
Flash MX 2004

Uso
```
symbolInstance.symbolType
```

Descripción
Propiedad; una cadena que especifica el tipo de símbolo. Esta propiedad equivale al valor de Comportamiento de los cuadros de diálogo Crear nuevo símbolo y Convertir en símbolo. Los valores aceptables son `"button"`, `"movie clip"` y `"graphic"`.

Ejemplo
El ejemplo siguiente define el primer símbolo del primer fotograma de la primera capa de la línea de tiempo del documento actual para que se comporte como un símbolo gráfico:

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].symbolType = "graphic";
```

symbolInstance.tabIndex

Disponibilidad

Flash MX 2004

Uso

```
symbolInstance.tabIndex
```

Descripción

Propiedad; un entero que equivale al campo Índice de fichas del panel Accesibilidad. Crea un orden de tabulación con el que se accede a los objetos cuando el usuario presiona la tecla Tabulador. Esta propiedad no está disponible para símbolos gráficos.

Ejemplo

El ejemplo siguiente define la propiedad `tabIndex` del objeto `mySymbol` como 3 y muestra ese valor en el panel Salida:

```
var mySymbol = fl.getDocumentDOM().selection[0];
mySymbol.tabIndex = 3;
fl.trace(mySymbol.tabIndex);
```

Capítulo 43: Objeto SymbolItem

Herencia Objeto Item > Objeto SymbolItem

Disponibilidad
Flash MX 2004

Descripción
El objeto SymbolItem es una subclase del Objeto Item.

Resumen de métodos
Además de los métodos del objeto Item, puede emplear los métodos siguientes con el objeto SymbolItem:

Método	Descripción
`symbolItem.convertToCompiledClip()`	Convierte un elemento de símbolo de la biblioteca en un clip de película compilado.
`symbolItem.exportSWC()`	Exporta el elemento de símbolo a un archivo SWC.
`symbolItem.exportSWF()`	Exporta el elemento de símbolo a un archivo SWF.

Resumen de propiedades
Además de las propiedades del objeto Item, el objeto SymbolItem dispone de las siguientes:

Propiedad	Descripción
`symbolItem.scalingGrid`	Un valor booleano que especifica si se activa la escala en 9 divisiones para el elemento.
`symbolItem.scalingGridRect`	Un objeto Rectangle que especifica las ubicaciones de las cuatro guías de 9 divisiones.
`symbolItem.sourceAutoUpdate`	Un valor booleano que especifica si el elemento se actualizará cuando se publique el archivo FLA.
`symbolItem.sourceFilePath`	Una cadena que especifica la ruta del archivo FLA de origen como URI file:///.
`symbolItem.sourceLibraryName`	Una cadena que especifica el nombre del elemento de la biblioteca de archivos de origen.
`symbolItem.symbolType`	Una cadena que especifica el tipo de símbolo.
`symbolItem.timeline`	De sólo lectura; un Objeto Timeline.

symbolItem.convertToCompiledClip()

Disponibilidad
Flash MX 2004

Uso
```
symbolItem.convertToCompiledClip()
```

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; convierte un elemento de símbolo de la biblioteca en un clip de película compilado.

Ejemplo
El ejemplo siguiente convierte un elemento de la biblioteca en un clip de película compilado:

```
fl.getDocumentDOM().library.items[3].convertToCompiledClip();
```

symbolItem.exportSWC()

Disponibilidad
Flash MX 2004

Uso
```
symbolItem.exportSWC(outputURI)
```

Parámetros
`outputURI` Una cadena, expresada como URI file:///, que especifica el archivo SWC al que el método exportará el símbolo. El *outputURI* debe hacer referencia a un archivo local. Flash no crea una carpeta si no existe *outputURI*.

Valor devuelto
Ninguno.

Descripción
Método; exporta el elemento de símbolo a un archivo SWC.

Ejemplo
El ejemplo siguiente exporta un elemento de la biblioteca al archivo SWC llamado `mySymbol.swc` de la carpeta tests:

```
fl.getDocumentDOM.library.selectItem("mySymbol");
var currentSelection = fl.getDocumentDOM().library.getSelectedItems();
currentSelection[0].exportSWC("file:///Macintosh HD/SWCDirectory/mySymbol.swc");
```

symbolItem.exportSWF()

Disponibilidad
Flash MX 2004

Uso
```
symbolItem.exportSWF(outputURI)
```

Parámetros

`outputURI` Una cadena, expresada como URI file:///, que especifica el archivo SWF al que el método exportará el símbolo. El *outputURI* debe hacer referencia a un archivo local. Flash no crea una carpeta si no existe *outputURI*.

Valor devuelto

Ninguno.

Descripción

Método; exporta el elemento de símbolo a un archivo SWF.

Ejemplo

El ejemplo siguiente exporta un elemento de la biblioteca al archivo `my.swf` de la carpeta tests:

```
fl.getDocumentDOM().library.items[0].exportSWF("file:///c|/tests/my.swf");
```

symbolItem.scalingGrid

Disponibilidad

Flash 8

Uso

```
symbolItem.scalingGrid
```

Descripción

Propiedad; un valor booleano que especifica si se activa la escala en 9 divisiones para el elemento.

Ejemplo

El siguiente ejemplo activa la escala en 9 divisiones para un elemento de la biblioteca:

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

Véase también

`symbolItem.scalingGridRect`

symbolItem.scalingGridRect

Disponibilidad

Flash 8

Uso

```
symbolItem.scalingGridRect
```

Descripción

Propiedad; un objeto Rectangle que especifica las ubicaciones de las cuatro guías de 9 divisiones. Para más información sobre el formato del rectángulo, consulte `document.addNewRectangle()`.

Ejemplo

El ejemplo siguiente especifica las ubicaciones de las guías en 9 divisiones.

```
fl.getDocumentDOM().library.items[0].scalingGridRect = {left:338, top:237, right:3859,
bottom:713};
```

Véase también

symbolItem.scalingGrid

symbolItem.sourceAutoUpdate

Disponibilidad

Flash MX 2004

Uso

```
symbolItem.sourceAutoUpdate
```

Descripción

Propiedad; un valor booleano que especifica si el elemento se actualizará cuando se publique el archivo FLA. El valor predeterminado es `false`. Se utiliza para símbolos de bibliotecas compartidas.

Ejemplo

El ejemplo siguiente define la propiedad `sourceAutoUpdate` para un elemento de biblioteca:

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

symbolItem.sourceFilePath

Disponibilidad

Flash MX 2004

Uso

```
symbolItem.sourceFilePath
```

Descripción

Propiedad; una cadena que especifica la ruta del archivo FLA de origen como URI file:///. La ruta debe ser una ruta absoluta, no relativa. Esta propiedad se utiliza para símbolos de bibliotecas compartidas.

Ejemplo

El ejemplo siguiente muestra el valor de la propiedad `sourceFilePath` en el panel Salida:

```
fl.trace(fl.getDocumentDOM().library.items[0].sourceFilePath);
```

symbolItem.sourceLibraryName

Disponibilidad
Flash MX 2004

Uso
`symbolItem.sourceLibraryName`

Descripción
Propiedad; una cadena que especifica el nombre del elemento de la biblioteca de archivos de origen. Se utiliza para símbolos de bibliotecas compartidas.

Ejemplo
El ejemplo siguiente muestra el valor de la propiedad `sourceLibraryName` en el panel Salida:

```
fl.trace(fl.getDocumentDOM().library.items[0].sourceLibraryName);
```

symbolItem.symbolType

Disponibilidad
Flash MX 2004

Uso
`symbolItem.symbolType`

Descripción
Propiedad; una cadena que especifica el tipo de símbolo. Los valores aceptables son `"movie clip"`, `"button"` y `"graphic"`.

Ejemplo
El ejemplo siguiente muestra el valor actual de la propiedad `symbolType`, lo cambia a `button` y vuelve a mostrarlo:

```
alert(fl.getDocumentDOM().library.items[0].symbolType);
fl.getDocumentDOM().library.items[0].symbolType = "button";
alert(fl.getDocumentDOM().library.items[0].symbolType);
```

symbolItem.timeline

Disponibilidad
Flash MX 2004

Uso
`symbolItem.timeline`

Descripción

De sólo lectura; un Objeto Timeline.

Ejemplo

El ejemplo siguiente obtiene y muestra el número de capas que contiene el clip de película seleccionado en la biblioteca:

```
var tl = fl.getDocumentDOM().library.getSelectedItems()[0].timeline;
alert(tl.layerCount);
```

Capítulo 44: Objeto Text

Herencia Objeto Element > Objeto Text

Disponibilidad

Flash MX 2004

Descripción

El objeto Text representa un elemento de texto único en un documento. Todas las propiedades del texto pertenecen al bloque de texto completo.

Para definir las propiedades de una extensión de texto en el campo de texto, consulte Objeto TextAttrs. Para cambiar las propiedades de una selección en un campo de texto, puede utilizar `document.setElementTextAttr()` y especificar un rango de texto o emplear la selección actual.

Para definir las propiedades genéricas del campo de texto seleccionado, utilice `document.setElementProperty()`. El ejemplo siguiente define el valor `x` del punto de registro del campo de texto seleccionado como 50:

```
fl.getDocumentDOM().setElementProperty("x", 50);
```

Resumen de métodos

Además de los métodos del objeto Element, el objeto Text dispone de los siguientes:

Método	Descripción
`text.getTextAttr()`	Recupera el atributo especificado para el texto identificado por los parámetros ***startIndex*** y ***endIndex*** opcionales.
`text.getTextString()`	Recupera el rango de texto especificado.
`text.setTextAttr()`	Define el atributo especificado asociado al texto identificado por ***startIndex*** y ***endIndex***.
`text.setTextString()`	Cambia la cadena de texto en este objeto Text.

Resumen de propiedades

Además de las propiedades del objeto Element, el objeto Text dispone de las siguientes:

Propiedad	Descripción
`text.accName`	Una cadena que equivale al campo Nombre del panel Accesibilidad.
`text.antiAliasSharpness`	Un valor flotante que especifica la nitidez de suavizado del texto.
`text.antiAliasThickness`	Un valor flotante que especifica el grosor de suavizado del texto.
`text.autoExpand`	Un valor booleano que controla la expansión de la anchura de delimitación para campos de texto estático o la anchura y la altura de delimitación para texto dinámico o de entrada.
`text.border`	Un valor booleano que controla si Flash muestra (`true`) u oculta (`false`) un borde alrededor del texto dinámico o de entrada.
`text.description`	Una cadena que equivale al campo Descripción del panel Accesibilidad.
`text.embeddedCharacters`	Una cadena que especifica caracteres para incorporar. Equivale a introducir texto en el cuadro de diálogo Incorporación de caracteres.

Propiedad	Descripción
`text.embedRanges`	Una cadena compuesta por enteros delimitados que corresponde a los elementos que se pueden seleccionar en el cuadro de diálogo Incorporación de caracteres.
`text.embedVariantGlyphs`	Valor booleano que especifica si activar la incorporación de pictogramas variantes.
`text.fontRenderingMode`	Una cadena que especifica el modo de presentación del texto.
`text.length`	De sólo lectura; un entero que representa el número de caracteres del objeto Text.
`text.lineType`	Una cadena que define el tipo de línea como `"single line"`, `"multiline"`, `"multiline no wrap"` o `"password"`.
`text.maxCharacters`	Un entero que especifica los caracteres máximos que el usuario puede introducir en este objeto Text.
`text.orientation`	Una cadena que especifica la orientación del campo de texto.
`text.renderAsHTML`	Un valor booleano que controla si Flash dibuja el texto como HTML e interpreta etiquetas HTML incorporadas.
`text.scrollable`	Un valor booleano que controla si el texto se puede desplazar (`true`) o no (`false`).
`text.selectable`	Un valor booleano que controla si el texto se puede seleccionar (`true`) o no (`false`). El texto de entrada siempre se puede seleccionar.
`text.selectionEnd`	Un entero basado en cero que especifica el desplazamiento del final de una subselección de texto.
`text.selectionStart`	Un entero basado en cero que especifica el desplazamiento del principio de una subselección de texto.
`text.shortcut`	Una cadena que equivale al campo Método abreviado del panel Accesibilidad.
`text.silent`	Un valor booleano que especifica si el objeto es accesible.
`text.tabIndex`	Un entero que equivale al campo Índice de fichas del panel Accesibilidad.
`text.textRuns`	De sólo lectura; un conjunto de objetos TextRun.
`text.textType`	Una cadena que especifica el tipo del campo de texto. Los valores aceptables son `"static"`, `"dynamic"` e `"input"`.
`text.useDeviceFonts`	Valor booleano. Un valor de `true` hace que Flash dibuje el texto utilizando fuentes del dispositivo.
`text.variableName`	Una cadena que alberga el contenido del objeto Text.

text.accName

Disponibilidad

Flash MX 2004

Uso

`text.accName`

Descripción

Propiedad; una cadena que equivale al campo Nombre del panel Accesibilidad. Los lectores de pantalla identifican los objetos mediante la lectura del nombre en voz alta. Esta propiedad no se puede utilizar con texto dinámico.

Ejemplo

El ejemplo siguiente recupera el nombre del objeto:

```
var doc = fl.getDocumentDOM();
var theName = doc.selection[0].accName;
```

El ejemplo siguiente define el nombre del objeto seleccionado actualmente:

```
fl.getDocumentDOM().selection[0].accName = "Home Button";
```

text.antiAliasSharpness

Disponibilidad

Flash 8

Uso

`text.antiAliasSharpness`

Descripción

Propiedad; un valor flotante que especifica la nitidez de suavizado del texto. Esta propiedad controla la nitidez con la que se dibuja el texto; los valores más altos especifican texto de mayor nitidez. Un valor de 0 especifica nitidez normal. Esta propiedad sólo está disponible si `text.fontRenderingMode` se define como `customThicknessSharpness`.

Ejemplo

Consulte `text.fontRenderingMode`.

Véase también

`text.antiAliasThickness`, `text.fontRenderingMode`

text.antiAliasThickness

Disponibilidad

Flash 8

Uso

`text.antiAliasThickness`

Descripción

Propiedad; un valor flotante que especifica el grosor de suavizado del texto. Esta propiedad controla el grosor con el que se dibuja el texto, los valores más altos especifican texto de mayor grosor. Un valor de 0 especifica grosor normal. Esta propiedad sólo está disponible si `text.fontRenderingMode` se define como `customThicknessSharpness`.

Ejemplo

Consulte `text.fontRenderingMode`.

Véase también

`text.antiAliasSharpness`, `text.fontRenderingMode`

text.autoExpand

Disponibilidad

Flash MX 2004

Uso

```
text.autoExpand
```

Descripción

Propiedad; un valor booleano. En campos de texto estático, un valor de `true` hace que la anchura de delimitación se expanda para mostrar todo el texto. En campos de texto dinámico o de entrada, un valor de `true` hace que la anchura y la altura de delimitación se expandan para mostrar todo el texto.

Ejemplo

El ejemplo siguiente define la propiedad `autoExpand` con un valor de `true`:

```
fl.getDocumentDOM().selection[0].autoExpand = true;
```

text.border

Disponibilidad

Flash MX 2004

Uso

```
text.border
```

Descripción

Propiedad; un valor booleano. Un valor de `true` hace que Flash muestre un borde alrededor del texto.

Ejemplo

El ejemplo siguiente define la propiedad `border` con un valor de `true`:

```
fl.getDocumentDOM().selection[0].border = true;
```

text.description

Disponibilidad

Flash MX 2004

Uso

```
text.description
```

Descripción

Propiedad; una cadena que equivale al campo Descripción del panel Accesibilidad. El lector de pantalla lee esta descripción.

Ejemplo

El ejemplo siguiente recupera la descripción del objeto:

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

El ejemplo siguiente establece la descripción del objeto:

```
var doc = fl.getDocumentDOM();
doc.selection[0].description= "Enter your name here";
```

text.embeddedCharacters

Disponibilidad

Flash MX 2004

Uso

`text.embeddedCharacters`

Descripción

Propiedad; una cadena que especifica caracteres para incorporar. Equivale a introducir texto en el cuadro de diálogo Incorporación de caracteres.

Esta propiedad sólo funciona con texto dinámico o de entrada; genera una advertencia si se utiliza con otros tipos de texto.

Ejemplo

El ejemplo siguiente define la propiedad `embeddedCharacters` como `abc`:

```
fl.getDocumentDOM().selection[0].embeddedCharacters = "abc";
```

text.embedRanges

Disponibilidad

Flash MX 2004

Uso

`text.embedRanges`

Descripción

Propiedad; una cadena compuesta por enteros delimitados que corresponde a los elementos que se pueden seleccionar en el cuadro de diálogo Incorporación de caracteres. Esta propiedad sólo funciona con texto dinámico o de entrada; se ignora si se utiliza con texto estático.

Nota: *esta propiedad corresponde al archivo XML de la carpeta Configuration/Font Embedding.*

Ejemplo

El ejemplo siguiente define la propiedad `embedRanges` como `"1|3|7"`:

```
var doc = fl.getDocumentDOM();
doc.selection[0].embedRanges = "1|3|7";
```

El ejemplo siguiente restablece la propiedad:

```
var doc = fl.getDocumentDOM();
doc.selection[0].embedRanges = "";
```

text.embedVariantGlyphs

Disponibilidad

Flash CS4 Professional

Uso

```
text.embedVariantGlyphs
```

Descripción

Propiedad; valor booleano que especifica si activar la incorporación de pictogramas variantes (`true`) o no (`false`). Esta propiedad sólo funciona con texto dinámico o de entrada; se ignora si se utiliza con texto estático. El valor predeterminado es `false`.

Ejemplo

En el siguiente ejemplo se activan los pictogramas variantes que se incorporarán al objeto Text seleccionado:

```
fl.getDocumentDOM().selection[0].embedVariantGlyphs = true;
```

Consulte también

`fontItem.embedVariantGlyphs`

text.fontRenderingMode

Disponibilidad

Flash 8

Uso

```
text.fontRenderingMode
```

Descripción

Propiedad; una cadena que especifica el modo de presentación del texto. Esta propiedad afecta al modo en que se muestra el texto tanto en el escenario como en Flash Player. Los valores aceptables se describen en la siguiente tabla:

Valor de la propiedad	Presentación del texto
`dispositivo`	Presenta el texto con fuentes de dispositivo.
`mapa de bits`	Presenta el texto suavizado como un mapa de bits, o como lo haría una fuente de píxel.
`standard`	Presenta el texto con el método de suavizado estándar de Flash MX 2004. Se trata de la mejor configuración que se puede utilizar para texto animado.
`advanced`	Presenta el texto a través de la tecnología de representación de fuentes con suavizado en Flash 8, que produce un mejor suavizado y aumenta la legibilidad, especialmente la del texto pequeño.
`customThicknessSharpness`	Permite especificar configuraciones personalizadas para la nitidez y el grosor del texto al utilizar la tecnología de representación de fuentes con suavizado implementada en Flash 8.

Ejemplo

El ejemplo siguiente muestra cómo se puede utilizar el valor `customThicknessSharpness` para especificar la nitidez y el grosor del texto:

```
fl.getDocumentDOM().setElementProperty("fontRenderingMode", "customThicknessSharpness");
fl.getDocumentDOM().setElementProperty("antiAliasSharpness", 400);
fl.getDocumentDOM().setElementProperty("antiAliasThickness", -200);
```

Véase también

`text.antiAliasSharpness`, `text.antiAliasThickness`

text.getTextAttr()

Disponibilidad

Flash MX 2004

Uso

```
text.getTextAttr(attrName [, startIndex [, endIndex]])
```

Parámetros

`attrName` Una cadena que especifica el nombre del objeto TextAttrs que se va a devolver. Para ver una lista de los valores posibles de *`attrName`*, consulte el resumen de propiedades del Objeto TextAttrs.

`startIndex` Un entero que es el índice del primer carácter. Este parámetro es opcional.

`endIndex` Un entero que especifica el final del rango de texto, que comienza por *startIndex* y llega hasta *endIndex*, no incluido. Este parámetro es opcional.

Valor devuelto

El valor del atributo especificado en el parámetro *attrName*.

Descripción

Método; recupera el atributo especificado por el parámetro *attrName* para el texto identificado por los parámetros *startIndex* y *endIndex* opcionales. Si el atributo no coincide con el rango especificado, Flash devuelve `undefined`. Si omite los parámetros opcionales *startIndex* y *endIndex*, el método utilizará el rango de texto completo. Si sólo especifica *startIndex*, el rango utilizado será el carácter que se encuentra en esa posición. Si especifica *startIndex* y *endIndex*, el rango comenzará en *startIndex* y llegará hasta *endIndex*, no incluido.

Ejemplo

El ejemplo siguiente obtiene el tamaño de fuente del campo de texto seleccionado actualmente y lo muestra:

```
var TheTextSize = fl.getDocumentDOM().selection[0].getTextAttr("size");
fl.trace(TheTextSize);
```

El ejemplo siguiente obtiene el color de relleno del texto del campo de texto seleccionado:

```
var TheFill = fl.getDocumentDOM().selection[0].getTextAttr("fillColor");
fl.trace(TheFill);
```

El ejemplo siguiente obtiene el tamaño del tercer carácter:

```
var Char3 = fl.getDocumentDOM().selection[0].getTextAttr("size", 2);
fl.trace(Char3);
```

El ejemplo siguiente obtiene el color del campo de texto seleccionado del tercer al octavo carácter:

```
fl.getDocumentDOM().selection[0].getTextAttr("fillColor", 2, 8);
```

text.getTextString()

Disponibilidad

Flash MX 2004

Uso

```
text.getTextString([startIndex [, endIndex]])
```

Parámetros

`startIndex` Un entero que especifica el índice (basado en cero) del primer carácter. Este parámetro es opcional.

`endIndex` Un entero que especifica el final del rango de texto, que comienza por *startIndex* y llega hasta *endIndex*, no incluido. Este parámetro es opcional.

Valor devuelto

Una cadena del texto del rango especificado.

Descripción

Método; recupera el rango de texto especificado. Si omite los parámetros opcionales *startIndex* y *endIndex*, se devolverá la cadena de texto completa. Si sólo especifica *startIndex*, el método devolverá la cadena que comienza en la ubicación de índice y termina al final del campo. Si especifica *startIndex* y *endIndex*, el método devolverá la cadena que comienza en *startIndex* y llega hasta *endIndex*, no incluido.

Ejemplo
El ejemplo siguiente obtiene el carácter o caracteres desde el quinto carácter hasta el final del campo de texto seleccionado:

```
var myText = fl.getDocumentDOM().selection[0].getTextString(4);
fl.trace(myText);
```

El ejemplo siguiente obtiene del cuarto al noveno carácter comenzando en el campo de texto seleccionado:

```
var myText = fl.getDocumentDOM().selection[0].getTextString(3, 9);
fl.trace(myText);
```

text.length

Disponibilidad
Flash MX 2004

Uso
`text.length`

Descripción
Propiedad de sólo lectura; un entero que representa el número de caracteres del objeto Text.

Ejemplo
El ejemplo siguiente devuelve el número de caracteres del texto seleccionado:

```
var textLength = fl.getDocumentDOM().selection[0].length;
```

text.lineType

Disponibilidad
Flash MX 2004

Uso
`text.lineType`

Descripción
Propiedad; una cadena que define el tipo de línea. Los valores aceptables son `"single line"`, `"multiline"`, `"multiline no wrap"` y `"password"`.

Esta propiedad sólo funciona con texto dinámico o de entrada y genera una advertencia si se utiliza con texto estático. El valor de `"password"` sólo funciona con texto de entrada.

Ejemplo
El ejemplo siguiente define la propiedad `lineType` con el valor `multiline no wrap`:

```
fl.getDocumentDOM().selection[0].lineType = "multiline no wrap";
```

text.maxCharacters

Disponibilidad
Flash MX 2004

Uso
`text.maxCharacters`

Descripción
Propiedad; un entero que especifica el número máximo de caracteres que el usuario puede introducir en este objeto Text.

Esta propiedad sólo funciona con texto de entrada; si se utiliza con otros tipos de texto, la propiedad genera una advertencia.

Ejemplo
El ejemplo siguiente define el valor de la propiedad `maxCharacters` como 30:

```
fl.getDocumentDOM().selection[0].maxCharacters = 30;
```

text.orientation

Disponibilidad
Flash MX 2004

Uso
`text.orientation`

Descripción
Propiedad; una cadena que especifica la orientación del campo de texto. Los valores aceptables son `"horizontal"`, `"vertical left to right"` y `"vertical right to left"`.

Esta propiedad sólo funciona con texto estático; genera una advertencia si se utiliza con otros tipos de texto.

Ejemplo
El ejemplo siguiente define la propiedad de orientación como `vertical right to left`:

```
fl.getDocumentDOM().selection[0].orientation = "vertical right to left";
```

text.renderAsHTML

Disponibilidad
Flash MX 2004

Uso
`text.renderAsHTML`

Descripción

Propiedad; un valor booleano. Si el valor es `true`, Flash dibuja el texto como HTML e interpreta etiquetas HTML incorporadas.

Esta propiedad sólo funciona con texto dinámico o de entrada; genera una advertencia si se utiliza con otros tipos de texto.

Ejemplo

El ejemplo siguiente define la propiedad `renderAsHTML` como `true`:

```
fl.getDocumentDOM().selection[0].renderAsHTML = true;
```

text.scrollable

Disponibilidad

Flash MX 2004

Uso

```
text.scrollable
```

Descripción

Propiedad; un valor booleano. Si el valor es `true`, el texto se puede desplazar.

Esta propiedad sólo funciona con texto dinámico o de entrada; genera una advertencia si se utiliza con texto estático.

Ejemplo

El ejemplo siguiente define la propiedad `scrollable` como `false`:

```
fl.getDocumentDOM().selection[0].scrollable = false;
```

text.selectable

Disponibilidad

Flash MX 2004

Uso

```
text.selectable
```

Descripción

Propiedad; un valor booleano. Si el valor es `true`, el texto se puede seleccionar.

El texto de entrada siempre se puede seleccionar. Flash genera una advertencia cuando se define esta propiedad como `false` y se utiliza con texto de entrada.

Ejemplo

El ejemplo siguiente define la propiedad `selectable` como `true`:

```
fl.getDocumentDOM().selection[0].selectable = true;
```

text.selectionEnd

Disponibilidad
Flash MX 2004

Uso
`text.selectionEnd`

Descripción
Propiedad; un entero basado en cero que especifica el final de una subselección de texto. Para más información, consulte `text.selectionStart`.

text.selectionStart

Disponibilidad
Flash MX 2004

Uso
`text.selectionStart`

Descripción
Propiedad; un entero basado en cero que especifica el principio de una subselección de texto. Puede utilizar esta propiedad con `text.selectionEnd` para seleccionar un rango de caracteres. Se seleccionarán los caracteres hasta `text.selectionEnd`, no incluido. Consulte `text.selectionEnd`.

- Si no hay un punto de inserción o una selección, `text.selectionEnd` es igual a `text.selectionStart`.
- Si se define `text.selectionStart` con un valor mayor que `text.selectionEnd`, `text.selectionEnd` se definirá como `text.selectionStart` y no se seleccionará texto.

Ejemplo
El ejemplo siguiente define el comienzo de la subselección de texto en el sexto carácter:

```
fl.getDocumentDOM().selection[0].selectionStart = 5;
```

El ejemplo siguiente selecciona los caracteres `Barbara` de un campo de texto que contiene el texto `My name is Barbara` y les aplica formato de negrita y verde:

```
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()

Disponibilidad
Flash MX 2004

Uso
```
text.setTextAttr(attrName, attrValue [, startIndex [, endIndex]])
```

Parámetros
`attrName` Una cadena que especifica el nombre del objeto TextAttrs que se va a modificar.

`attrValue` El valor de la propiedad del objeto TextAttrs.

Para ver una lista de los valores posibles de *`attrName`* y *`attrValue`*, consulte el resumen de propiedades del Objeto TextAttrs.

`startIndex` Un entero que es el índice (basado en cero) del primer carácter del conjunto. Este parámetro es opcional.

`endIndex` Un entero que especifica el índice del punto final de la cadena de texto seleccionada, que comienza por *startIndex* y llega hasta *endIndex*, no incluido. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; define el atributo especificado por el parámetro *attrName* asociado al texto identificado por *startIndex* y *endIndex* con el valor especificado por *attrValue*. Este método puede emplearse para cambiar atributos de texto que pueden comprender elementos TextRun (consulte Objeto TextRun) o que son partes de elementos TextRun existentes. Su uso puede cambiar la posición y el número de elementos TextRun en el conjunto `text.textRuns` de este objeto (consulte text.textRuns).

Si omite los parámetros opcionales, el método utilizará el rango de caracteres de todo el objeto Text. Si sólo especifica *startIndex*, el rango será el carácter que se encuentra en esa posición. Si especifica *startIndex* y *endIndex*, el rango comenzará en *startIndex* y llegará hasta el carácter situado en *endIndex*, no incluido.

Ejemplo
El ejemplo siguiente define el campo de texto seleccionado como cursiva:

```
fl.getDocumentDOM().selection[0].setTextAttr("italic", true);
```

El ejemplo siguiente define el tamaño del tercer carácter como 10:

```
fl.getDocumentDOM().selection[0].setTextAttr("size", 10, 2);
```

El ejemplo siguiente define el color como rojo para el rango del tercer al octavo carácter del texto seleccionado:

```
fl.getDocumentDOM().selection[0].setTextAttr("fillColor", 0xff0000, 2, 8);
```

text.setTextString()

Disponibilidad
Flash MX 2004

Uso

```
text.setTextString(text [, startIndex [, endIndex]])
```

Parámetros

`text` Una cadena compuesta por los caracteres que se van a insertar en este objeto Text.

`startIndex` Un entero que especifica el índice (basado en cero) del carácter en la cadena donde se insertará el texto. Este parámetro es opcional.

`endIndex` Un entero que especifica el índice el punto final en la cadena de texto seleccionada. El nuevo texto sobrescribe el texto de *startIndex* hasta *endIndex*, no incluido. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Propiedad; cambia la cadena de texto en este objeto Text. Si omite los parámetros opcionales, se reemplazará todo el objeto Text. Si sólo especifica *startIndex*, la cadena especificada se insertará en la posición *startIndex*. Si especifica *startIndex* y *endIndex*, la cadena especificada reemplazará el segmento de texto que comienza en *startIndex* y llega a *endIndex* (no incluido).

Ejemplo

El ejemplo siguiente asigna la cadena `this is a string` al campo de texto seleccionado:

```
fl.getDocumentDOM().selection[0].setTextString("this is a string");
```

El ejemplo siguiente inserta la cadena `abc` que comienza en el quinto carácter del campo de texto seleccionado:

```
fl.getDocumentDOM().selection[0].setTextString("01234567890");
fl.getDocumentDOM().selection[0].setTextString("abc", 4);
// text field is now "0123abc4567890"
```

El ejemplo siguiente reemplaza el texto del rango que va del tercer al octavo carácter de la cadena de texto seleccionada por la cadena `abcdefghij`. Se sobrescribirán los caracteres entre *startIndex* y *endIndex*. Los caracteres que comienzan con *endIndex* siguen a la cadena insertada.

```
fl.getDocumentDOM().selection[0].setTextString("01234567890");
fl.getDocumentDOM().selection[0].setTextString("abcdefghij", 2, 8);
// text field is now "01abcdefghij890"
```

text.shortcut

Disponibilidad

Flash MX 2004

Uso

```
text.shortcut
```

Descripción

Propiedad; una cadena que equivale al campo Método abreviado del panel Accesibilidad. El lector de pantalla lee este método abreviado. Esta propiedad no se puede utilizar con texto dinámico.

Ejemplo

El ejemplo siguiente obtiene la tecla de método abreviado del objeto seleccionado y muestra el valor:

```
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;
fl.trace(theShortcut);
```

El ejemplo siguiente define la tecla de método abreviado del objeto seleccionado:

```
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+i";
```

text.silent

Disponibilidad

Flash MX 2004

Uso

```
text.silent
```

Descripción

Propiedad; un valor booleano que especifica si el objeto es accesible. Equivale a la lógica inversa de la opción Hacer que el objeto sea accesible del panel Accesibilidad. Es decir, si `silent` es `true`, estará desactivada la opción Hacer que el objeto sea accesible. Si es `false`, estará activada la opción Hacer que el objeto sea accesible.

Ejemplo

El ejemplo siguiente determina si el objeto es accesible (un valor de `false` significa que es accesible):

```
var isSilent = fl.getDocumentDOM().selection[0].silent;
```

El ejemplo siguiente define el objeto como accesible:

```
fl.getDocumentDOM().selection[0].silent = false;
```

text.tabIndex

Disponibilidad

Flash MX 2004

Uso

```
text.tabIndex
```

Descripción

Propiedad; un entero que equivale al campo Índice de fichas del panel Accesibilidad. Este valor permite determinar el orden de acceso a los objetos cuando el usuario presiona la tecla Tabulador.

Ejemplo

El ejemplo siguiente obtiene el `tabIndex` del objeto seleccionado actualmente:

```
var theTabIndex = fl.getDocumentDOM().selection[0].tabIndex;
```

El ejemplo siguiente establece el `tabIndex` del objeto seleccionado actualmente:

```
fl.getDocumentDOM().selection[0].tabIndex = 1;
```

text.textRuns

Disponibilidad
Flash MX 2004

Uso
`text.textRuns`

Descripción
Propiedad de sólo lectura; un conjunto de objetos TextRun (consulte Objeto TextRun).

Ejemplo
El ejemplo siguiente almacena el valor de la propiedad `textRuns` en la variable `myTextRuns`:

```
var myTextRuns = fl.getDocumentDOM().selection[0].textRuns;
```

text.textType

Disponibilidad
Flash MX 2004

Uso
`text.textType`

Descripción
Propiedad; una cadena que especifica el tipo del campo de texto. Los valores aceptables son `"static"`, `"dynamic"` e `"input"`.

Ejemplo
El ejemplo siguiente define la propiedad `textType` como `input`:

```
fl.getDocumentDOM().selection[0].textType = "input";
```

text.useDeviceFonts

Disponibilidad
Flash MX 2004

Uso
`text.useDeviceFonts`

Descripción

Propiedad; un valor booleano. Un valor de `true` hace que Flash dibuje el texto utilizando fuentes del dispositivo.

Ejemplo

El ejemplo siguiente hace que Flash utilice fuentes del dispositivo al dibujar texto:

```
fl.getDocumentDOM().selection[0].useDeviceFonts = true;
```

text.variableName

Disponibilidad

Flash MX 2004

Uso

```
text.variableName
```

Descripción

Propiedad; una cadena que contiene el nombre de la variable asociada al objeto Text. Esta propiedad sólo funciona con texto dinámico o de entrada; genera una advertencia si se utiliza con otros tipos de texto.

Esta propiedad se admite únicamente en ActionScript 1.0 y ActionScript 2.0.

Ejemplo

El siguiente ejemplo establece el nombre de la variable del cuadro de texto seleccionado en `firstName`:

```
fl.getDocumentDOM().selection[0].variableName = "firstName";
```

Capítulo 45: Objeto TextAttrs

Disponibilidad

Flash MX 2004

Descripción

El objeto TextAttrs contiene todas las propiedades de texto que se pueden aplicar a una subselección. Este objeto es una propiedad del objeto TextRun (`textRun.textAttrs`).

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto TextAttrs:

Propiedad	Descripción
`textAttrs.aliasText`	Un valor booleano que especifica que Flash debe dibujar el texto empleando un método optimizado para aumentar la legibilidad del texto pequeño.
`textAttrs.alignment`	Una cadena que especifica la justificación del párrafo. Los valores aceptables son `"left"`, `"center"`, `"right"` y `"justify"`.
`textAttrs.autoKern`	Un valor booleano que determina si Flash utiliza (`true`) o ignora (`false`) la información de ajuste entre caracteres en las fuentes para justificar el texto.
`textAttrs.bold`	Valor booleano. Un valor de `true` hace que el texto aparezca con la versión en negrita de la fuente.
`textAttrs.characterPosition`	Una cadena que determina la línea base del texto.
`textAttrs.characterSpacing`	Desfasado; ahora se emplea `textAttrs.letterSpacing`. Un entero que representa el espacio entre caracteres.
`textAttrs.face`	Una cadena que representa el nombre de la fuente, por ejemplo, `"Arial"`.
`textAttrs.fillColor`	Una cadena, valor hexadecimal o entero que representa el color de relleno.
`textAttrs.indent`	Un entero que especifica el sangrado del párrafo.
`textAttrs.italic`	Valor booleano. Un valor de `true` hace que el texto aparezca con la versión en cursiva de la fuente.
`textAttrs.leftMargin`	Un entero que especifica el margen izquierdo del párrafo.
`textAttrs.letterSpacing`	Un entero que representa el espacio entre caracteres.
`textAttrs.lineSpacing`	Un entero que especifica el interlineado (inicial) del párrafo.
`textAttrs.rightMargin`	Un entero que especifica el margen derecho del párrafo.
`textAttrs.rotation`	Valor booleano. Un valor de `true` hace que Flash gire los caracteres del texto 90º. El valor predeterminado es `false`.
`textAttrs.size`	Un entero que especifica el tamaño de la fuente.
`textAttrs.target`	Una cadena que representa la propiedad `target` del campo de texto.
`textAttrs.url`	Una cadena que representa la propiedad `URL` del campo de texto.

textAttrs.aliasText

Disponibilidad
Flash MX 2004

Uso
`textAttrs.aliasText`

Descripción
Propiedad; un valor booleano que especifica que Flash debe dibujar el texto empleando un método optimizado para aumentar la legibilidad del texto pequeño.

Ejemplo
El ejemplo siguiente define la propiedad `aliasText` como `true` para todo el texto del campo de texto seleccionado actualmente:

```
fl.getDocumentDOM().setElementTextAttr('aliasText', true);
```

textAttrs.alignment

Disponibilidad
Flash MX 2004

Uso
`textAttrs.alignment`

Descripción
Propiedad; una cadena que especifica la justificación del párrafo. Los valores aceptables son `"left"`, `"center"`, `"right"` y `"justify"`.

Ejemplo
El ejemplo siguiente define la justificación de los párrafos que contienen caracteres entre el índice 0 y el índice 3, no incluido. Puede afectar a los caracteres fuera del rango especificado si se encuentran en el mismo párrafo.

```
fl.getDocumentDOM().setTextSelection(0, 3);
fl.getDocumentDOM().setElementTextAttr("alignment", "justify");
```

textAttrs.autoKern

Disponibilidad
Flash MX 2004

Uso
`textAttrs.autoKern`

Descripción
Propiedad; un valor booleano que determina si Flash utiliza (`true`) o ignora (`false`) la información de ajuste entre caracteres en las fuentes cuando se justifica el texto.

Ejemplo
El ejemplo siguiente selecciona los caracteres desde el índice 2 hasta el índice 6, no incluido, y define la propiedad `autoKern` como `true`:

```
fl.getDocumentDOM().setTextSelection(3, 6);
fl.getDocumentDOM().setElementTextAttr('autoKern', true);
```

textAttrs.bold

Disponibilidad
Flash MX 2004

Uso
`textAttrs.bold`

Descripción
Propiedad; un valor booleano. Un valor de `true` hace que el texto aparezca con la versión en negrita de la fuente.

Ejemplo
El ejemplo siguiente selecciona el primer carácter del objeto de texto seleccionado y define la propiedad `bold` como `true`:

```
fl.getDocumentDOM().setTextSelection(0, 1);
fl.getDocumentDOM().setElementTextAttr('bold', true);
```

textAttrs.characterPosition

Disponibilidad
Flash MX 2004

Uso
`textAttrs.characterPosition`

Descripción
Propiedad; una cadena que determina la línea base del texto. Los valores aceptables son `"normal"`, `"subscript"` y `"superscript"`. Esta propiedad sólo se aplica a texto estático.

Ejemplo
El ejemplo siguiente selecciona los caracteres desde el índice 2 hasta el índice 6, no incluido, del texto seleccionado y define la propiedad `characterPosition` como `subscript`:

```
fl.getDocumentDOM().setTextSelection(2, 6);
fl.getDocumentDOM().setElementTextAttr("characterPosition", "subscript");
```

textAttrs.characterSpacing

Disponibilidad
Flash MX 2004 Desfasada en Flash 8; ahora se emplea `textAttrs.letterSpacing`.

Uso
```
textAttrs.characterSpacing
```

Descripción
Propiedad; un entero que representa el espacio entre caracteres. Los valores aceptables están comprendidos entre -60 y 60.

Esta propiedad sólo se aplica a texto estático; genera una advertencia si se utiliza con otros tipos de texto.

Ejemplo
El ejemplo siguiente define el espaciado de caracteres del campo de texto seleccionado como 10:

```
fl.getDocumentDOM().setElementTextAttr("characterSpacing", 10);
```

textAttrs.face

Disponibilidad
Flash MX 2004

Uso
```
textAttrs.face
```

Descripción
Propiedad; una cadena que representa el nombre de la fuente, por ejemplo, `"Arial"`.

Ejemplo
El ejemplo siguiente define como `Arial` la fuente del campo de texto seleccionado desde el carácter del índice 2 hasta el carácter del índice 8, no incluido:

```
fl.getDocumentDOM().selection[0].setTextAttr("face", "Arial", 2, 8);
```

textAttrs.fillColor

Disponibilidad
Flash MX 2004

Uso
```
textAttrs.fillColor
```

Descripción
Propiedad; el color del relleno, en uno de los formatos siguientes:

- Una cadena con el formato `"#RRGGBB"` o `"#RRGGBBAA"`
- Un número hexadecimal con el formato `0xRRGGBB`
- Un entero que representa el equivalente decimal del número hexadecimal

Ejemplo
El ejemplo siguiente define como rojo el color del campo de texto seleccionado desde el carácter del índice 2 hasta el carácter del índice 8, no incluido:

```
fl.getDocumentDOM().selection[0].setTextAttr("fillColor", 0xff0000, 2, 8);
```

textAttrs.indent

Disponibilidad
Flash MX 2004

Uso
```
textAttrs.indent
```

Descripción
Propiedad; un entero que especifica el sangrado del párrafo. Los valores aceptables están comprendidos entre -720 y 720.

Ejemplo
El ejemplo siguiente define el sangrado del campo de texto seleccionado desde el carácter del índice 2 hasta el carácter del índice 8 al 100, no incluido: Puede afectar a los caracteres fuera del rango especificado si se encuentran en el mismo párrafo.

```
fl.getDocumentDOM().selection[0].setTextAttr("indent", 100, 2, 8);
```

textAttrs.italic

Disponibilidad
Flash MX 2004

Uso
```
textAttrs.italic
```

Descripción
Propiedad; un valor booleano. Un valor de `true` hace que el texto aparezca con la versión en cursiva de la fuente.

Ejemplo
El ejemplo siguiente define el campo de texto seleccionado como cursiva:

```
fl.getDocumentDOM().selection[0].setTextAttr("italic", true);
```

textAttrs.leftMargin

Disponibilidad
Flash MX 2004

Uso
`textAttrs.leftMargin`

Descripción
Propiedad; un entero que especifica el margen izquierdo del párrafo. Los valores válidos están comprendidos entre 0 y 720.

Ejemplo
El ejemplo siguiente define la propiedad `leftMargin` del campo de texto seleccionado desde el carácter del índice 2 hasta el carácter del índice 8 al 100, no incluido: Puede afectar a los caracteres fuera del rango especificado si se encuentran en el mismo párrafo.

```
fl.getDocumentDOM().selection[0].setTextAttr("leftMargin", 100, 2, 8);
```

textAttrs.letterSpacing

Disponibilidad
Flash 8

Uso
`textAttrs.letterSpacing`

Descripción
Propiedad; un entero que representa el espacio entre caracteres. Los valores válidos están comprendidos entre -60 y 60.

Esta propiedad sólo se aplica a texto estático; genera una advertencia si se utiliza con otros tipos de texto.

Ejemplo
El código siguiente selecciona los caracteres desde el índice 0 hasta el índice 10, éste no incluido, y define el espaciado de caracteres en 60:

```
fl.getDocumentDOM().setTextSelection(0, 10);
fl.getDocumentDOM().setElementTextAttr("letterSpacing", 60);
```

textAttrs.lineSpacing

Disponibilidad
Flash MX 2004

Uso
`textAttrs.lineSpacing`

Descripción
Propiedad; un entero que especifica el interlineado (*inicial*) del párrafo. Los valores válidos están comprendidos entre -360 y 720.

Ejemplo
El ejemplo siguiente define como 100 la propiedad `lineSpacing` del campo de texto seleccionado:

```
fl.getDocumentDOM().selection[0].setTextAttr("lineSpacing", 100);
```

textAttrs.rightMargin

Disponibilidad
Flash MX 2004

Uso
```
textAttrs.rightMargin
```

Descripción
Propiedad; un entero que especifica el margen derecho del párrafo. Los valores válidos están comprendidos entre 0 y 720.

Ejemplo
El ejemplo siguiente define la propiedad `rightMargin` del campo de texto seleccionado desde el carácter del índice 2 hasta el carácter del índice 8 al 100, no incluido: Puede afectar a los caracteres fuera del rango especificado si se encuentran en el mismo párrafo.

```
fl.getDocumentDOM().selection[0].setTextAttr("rightMargin", 100, 2, 8);
```

textAttrs.rotation

Disponibilidad
Flash MX 2004

Uso
```
textAttrs.rotation
```

Descripción
Propiedad; un valor booleano. Un valor de `true` hace que Flash gire los caracteres del texto 90º. El valor predeterminado es `false`. Esta propiedad sólo se aplica a texto estático con orientación vertical; genera una advertencia si se utiliza con otros tipos de texto.

Ejemplo
El ejemplo siguiente define la rotación del campo de texto seleccionado como `true`:

```
fl.getDocumentDOM().setElementTextAttr("rotation", true);
```

textAttrs.size

Disponibilidad
Flash MX 2004

Uso
`textAttrs.size`

Descripción
Propiedad; un entero que especifica el tamaño de la fuente.

Ejemplo
El ejemplo siguiente recupera el tamaño del carácter en el índice 2 y muestra el resultado en el panel Salida:

```
fl.outputPanel.trace(fl.getDocumentDOM().selection[0].getTextAttr("size", 2));
```

textAttrs.target

Disponibilidad
Flash MX 2004

Uso
`textAttrs.target`

Descripción
Propiedad; una cadena que representa la propiedad `target` del campo de texto. Esta propiedad sólo funciona con texto estático.

Ejemplo
El ejemplo siguiente obtiene la propiedad `target` del campo de texto del primer fotograma de la capa superior de la escena actual y la muestra en el panel Salida:

```
fl.outputPanel.trace(fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].getTe
xtAttr("target"));
```

textAttrs.url

Disponibilidad
Flash MX 2004

Uso
`textAttrs.url`

Descripción

Propiedad; una cadena que representa la propiedad `URL` del campo de texto. Esta propiedad sólo funciona con texto estático.

Ejemplo

El ejemplo siguiente define la URL del campo de texto seleccionado como `http://www.adobe.com`:

```
fl.getDocumentDOM().setElementTextAttr("url", "http://www.adobe.com");
```

Capítulo 46: Objeto TextRun

Disponibilidad
Flash MX 2004

Descripción
El objeto TextRun representa una serie de caracteres que tienen atributos que coinciden con todas las propiedades del Objeto TextAttrs. Este objeto es una propiedad del objeto Text (`text.textRuns`).

Resumen de propiedades
Además de las propiedades disponibles para su uso con el objeto Text, el objeto TextRun proporciona las siguientes:

Propiedad	Descripción
`textRun.characters`	Una cadena que representa el texto que contiene el objeto TextRun.
`textRun.textAttrs`	El objeto TextAttrs que contiene los atributos de la extensión de texto.

textRun.textAttrs

Disponibilidad
Flash MX 2004

Uso
`textRun.textAttrs`

Descripción
Propiedad; el Objeto TextAttrs que contiene los atributos de la extensión de texto.

Ejemplo
El ejemplo siguiente muestra las propiedades de la primera extensión de caracteres del campo de texto seleccionado en el panel Salida.

```
var curTextAttrs = fl.getDocumentDOM().selection[0].textRuns[0].textAttrs;
for (var prop in curTextAttrs) {
    fl.trace(prop + " = " + curTextAttrs[prop]);
}
```

textRun.characters

Disponibilidad
Flash MX 2004

Uso
`textRun.characters`

Descripción

Propiedad; el texto que contiene el objeto TextRun.

Ejemplo

El ejemplo siguiente muestra los caracteres que componen la primera extensión de caracteres del campo de texto seleccionado en el panel Salida:

```
fl.trace(fl.getDocumentDOM().selection[0].textRuns[0].characters);
```

Capítulo 47: Objeto Timeline

Disponibilidad

Flash MX 2004

Descripción

El objeto Timeline representa la línea de tiempo de Flash, a la que puede acceder para el documento actual mediante `fl.getDocumentDOM().getTimeline()`. Este método devuelve la línea de tiempo de la escena actual o el símbolo que se está editando.

Cuando se trabaja con escenas, todas las líneas de tiempo de las escenas disponen de un valor de índice y se puede acceder a las mismas mediante `fl.getDocumentDOM().timelines[i]`. (En este ejemplo, `i` es el índice del valor de la línea de tiempo.)

Cuando se trabaja con fotogramas utilizando los métodos y propiedades del objeto Timeline, recuerde que el valor del fotograma es un índice basado en cero (no el número de fotograma real en la secuencia de fotogramas de la línea de tiempo). Es decir, el primer fotograma cuenta con un índice de fotograma de 0.

Resumen de métodos

Los siguientes métodos están disponibles para el objeto Timeline:

Método	Descripción
`timeline.addMotionGuide()`	Añade una capa de guía de movimiento sobre la capa actual y asocia la capa actual a la capa de guía recién añadida.
`timeline.addNewLayer()`	Añade una nueva capa al documento y la convierte en la capa actual.
`timeline.clearFrames()`	Elimina todo el contenido de un fotograma o un intervalo de fotogramas de la capa actual.
`timeline.clearKeyframes()`	Convierte un fotograma clave en un fotograma normal y elimina su contenido en la capa actual.
`timeline.convertToBlankKeyframes()`	Convierte los fotogramas en fotogramas clave en blanco en la capa actual.
`timeline.convertToKeyframes()`	Convierte un intervalo de fotogramas en fotogramas clave (o convierte la selección si no se especifican fotogramas) en la capa actual.
`timeline.copyFrames()`	Copia un intervalo de fotogramas de la capa actual en el portapapeles.
`timeline.copyMotion()`	Copia movimiento en los fotogramas seleccionados, de una interpolación de movimiento o de una animación fotograma por fotograma, por lo que se puede aplicar a otros fotogramas.
`timeline.copyMotionAsAS3()`	Copia el movimiento en los fotogramas seleccionados como código ActionScript 3.0 en el portapapeles, bien desde una interpolación de movimiento o mediante una animación fotograma a fotograma.
`timeline.createMotionTween()`	Define la propiedad `frame.tweenType` como `motion` para cada fotograma clave seleccionado en la capa actual y, si es necesario, convierte el contenido de cada fotograma en una instancia de símbolo único.
`timeline.cutFrames()`	Corta un intervalo de fotogramas de la capa actual en la línea de tiempo y lo guarda en el portapapeles.

Método	Descripción
`timeline.deleteLayer()`	Elimina una capa.
`timeline.expandFolder()`	Expande o contrae la carpeta o carpetas especificadas.
`timeline.findLayerIndex()`	Busca un conjunto de índices para las capas con un determinado nombre.
`timeline.getFrameProperty()`	Recupera el valor de la propiedad especificada para los fotogramas seleccionados.
`timeline.getGuidelines()`	Devuelve una cadena XML que representa las posiciones actuales de las guías horizontal y vertical de una línea de tiempo (Ver > Guías > Mostrar guías).
`timeline.getLayerProperty()`	Recupera el valor de la propiedad especificada para las capas seleccionadas.
`timeline.getSelectedFrames()`	Recupera los fotogramas seleccionados actualmente en un conjunto.
`timeline.getSelectedLayers()`	Recupera los valores de índice basado en cero de las capas seleccionadas actualmente.
`timeline.insertBlankKeyframe()`	Inserta un fotograma clave en blanco en el índice de fotograma especificado; si no se especifica el índice, inserta el fotograma clave en blanco utilizando la cabeza lectora/selección.
`timeline.insertFrames()`	Inserta el número especificado de fotogramas en un número de fotograma concreto.
`timeline.insertKeyframe()`	Inserta un fotograma clave en el fotograma especificado.
`timeline.pasteFrames()`	Pega el intervalo de fotogramas del portapapeles a los fotogramas especificados.
`timeline.pasteMotion()`	Pega el intervalo de fotogramas de movimiento recuperado por `timeline.copyMotion()` en la línea de tiempo.
`timeline.removeFrames()`	Elimina el fotograma.
`timeline.reorderLayer()`	Mueve la primera capa especificada delante o detrás de la segunda capa especificada.
`timeline.reverseFrames()`	Invierte un intervalo de fotogramas.
`timeline.selectAllFrames()`	Selecciona todos los fotogramas de la línea de tiempo actual.
`timeline.setFrameProperty()`	Establece la propiedad del objeto Frame para los fotogramas seleccionados.
`timeline.setGuidelines()`	Sustituye las líneas de guía para la línea de tiempo con la información especificada.
`timeline.setLayerProperty()`	Define con un valor determinado la propiedad especificada en todas las capas seleccionadas.
`timeline.setSelectedFrames()`	Selecciona un intervalo de fotogramas de la capa actual o define los fotogramas seleccionadas con el conjunto de selección transferido a este método.
`timeline.setSelectedLayers()`	Define la capa que se va a seleccionar y, además, convierte la capa especificada en la capa actual.
`timeline.showLayerMasking()`	Muestra el enmascaramiento de capas durante la edición bloqueando la máscara y las capas enmascaradas.

Resumen de propiedades

Las siguientes propiedades están disponibles para el objeto Timeline:

Propiedad	Descripción
`timeline.currentFrame`	Un índice basado en cero para el fotograma en la ubicación actual de la cabeza lectora.
`timeline.currentLayer`	El índice basado en cero para la capa activa actual.
`timeline.frameCount`	Sólo lectura; un entero que representa el número de fotogramas en la capa más larga de esta línea de tiempo.
`timeline.layerCount`	Sólo lectura; un entero que representa el número de capas en la línea de tiempo especificada.
`timeline.layers`	Sólo lectura; conjunto de objetos Layer.
`timeline.name`	Sólo lectura; cadena que representa el nombre de la línea de tiempo actual.

timeline.addMotionGuide()

Disponibilidad

Flash MX 2004

Uso

```
timeline.addMotionGuide()
```

Parámetros

Ninguno.

Valor devuelto

Un entero que representa el índice basado en cero de la capa de guía añadida. Si la capa actual no es de tipo `"Normal"`, Flash devuelve -1.

Descripción

Método; añade una capa de guía de movimiento sobre la capa actual y asocia la capa actual a la capa de guía recién añadida, convirtiendo la capa actual en una capa `"Con guía"`.

Este método sólo funciona en una capa de tipo `"Normal"`. No tiene ningún efecto en capas de tipo `"Carpeta"`, `"Máscara"`, `"Enmascarada"`, `"Guía"` o `"Con guía"`.

Ejemplo

El ejemplo siguiente añade una capa de guía de movimiento sobre la capa actual y convierte la capa actual en `Con guía`:

```
fl.getDocumentDOM().getTimeline().addMotionGuide();
```

timeline.addNewLayer()

Disponibilidad

Flash MX 2004

Uso

```
timeline.addNewLayer([name] [, layerType [, bAddAbove]])
```

Parámetros

`name` Una cadena que especifica el nombre de la nueva capa. Si omite este parámetro, se asignará un nombre de capa nuevo y predeterminado a la nueva capa ("Capa n", donde *n* es el número total de capas). Este parámetro es opcional.

`layerType` Una cadena que especifica el tipo de capa que se va a añadir. Si omite este parámetro, se crea una capa de tipo "Normal". Este parámetro es opcional. Los valores aceptables son `"normal"`, `"guide"`, `"guided"`, `"mask"`, `"masked"` y `"folder"`.

`bAddAbove` Un valor booleano que, si se define como `true` (valor predeterminado), hace que Flash añada la nueva capa sobre la capa actual; `false` hace que Flash añada la capa debajo de la capa actual. Este parámetro es opcional.

Valor devuelto

Un valor entero del índice basado en cero de la capa recién añadida.

Descripción

Método; añade una nueva capa al documento y la convierte en la capa actual.

Ejemplo

El ejemplo siguiente añade una nueva capa a la línea de tiempo con un nombre predeterminado generado por Flash:

```
fl.getDocumentDOM().getTimeline().addNewLayer();
```

El ejemplo siguiente añade una nueva capa de carpeta sobre la capa actual y le asigna el nombre `Folder1`:

```
fl.getDocumentDOM().getTimeline().addNewLayer("Folder1", "folder", true);
```

timeline.clearFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.clearFrames([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que define el comienzo del intervalo de fotogramas que se va a borrar. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que define el final del intervalo de fotogramas que se va a borrar. El intervalo llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; elimina todo el contenido de un fotograma o un intervalo de fotogramas de la capa actual.

Ejemplo
El ejemplo siguiente borra los fotogramas desde el Fotograma 6 hasta el 11, no incluido (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().clearFrames(5, 10);
```

El ejemplo siguiente borra el Fotograma 15:

```
fl.getDocumentDOM().getTimeline().clearFrames(14);
```

timeline.clearKeyframes()

Disponibilidad
Flash MX 2004

Uso

```
timeline.clearKeyframes([startFrameIndex [, endFrameIndex]])
```

Parámetros
`startFrameIndex` Un índice basado en cero que define el comienzo del intervalo de fotogramas que se va a borrar. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que define el final del intervalo de fotogramas que se va a borrar. El intervalo llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; convierte un fotograma clave en un fotograma normal y elimina su contenido en la capa actual.

Ejemplo
El ejemplo siguiente borra los fotogramas clave desde el Fotograma 5 hasta el 10, no incluido (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().clearKeyframes(4, 9);
```

El ejemplo siguiente borra el fotograma clave en el Fotograma 15 y lo convierte en un fotograma normal:

```
fl.getDocumentDOM().getTimeline().clearKeyframes(14);
```

timeline.convertToBlankKeyframes()

Disponibilidad
Flash MX 2004

Uso

```
timeline.convertToBlankKeyframes([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el fotograma inicial para convertir en fotogramas clave. Si omite *startFrameIndex*, el método convertirá los fotogramas seleccionados actualmente. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se detendrá la conversión en fotogramas clave. El intervalo de fotogramas para convertir llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; convierte los fotogramas en fotogramas clave en blanco en la capa actual.

Ejemplo

El ejemplo siguiente convierte desde el Fotograma 2 hasta el Fotograma 10, no incluido, en fotogramas clave en blanco (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().convertToBlankKeyframes(1, 9);
```

El ejemplo siguiente convierte el Fotograma 5 en un fotograma clave en blanco:

```
fl.getDocumentDOM().getTimeline().convertToBlankKeyframes(4);
```

timeline.convertToKeyframes()

Disponibilidad

Flash MX 2004

Uso

```
timeline.convertToKeyframes([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el primer fotograma para convertir en fotogramas clave. Si omite *startFrameIndex*, el método convertirá los fotogramas seleccionados actualmente. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se detendrá la conversión en fotogramas clave. El intervalo de fotogramas para convertir llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; convierte un intervalo de fotogramas en fotogramas clave (o convierte la selección si no se especifican fotogramas) en la capa actual.

Ejemplo

El ejemplo siguiente convierte los fotogramas seleccionados en fotogramas clave:

```
fl.getDocumentDOM().getTimeline().convertToKeyframes();
```

El ejemplo siguiente convierte los fotogramas clave desde el Fotograma 2 hasta el 10, no incluido (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().convertToKeyframes(1, 9);
```

El ejemplo siguiente convierte el Fotograma 5 en un fotograma clave:

```
fl.getDocumentDOM().getTimeline().convertToKeyframes(4);
```

timeline.copyFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.copyFrames([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el comienzo del intervalo de fotogramas que se va a copiar. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se dejará de copiar. El intervalo de fotogramas para copiar llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; copia un intervalo de fotogramas de la capa actual en el portapapeles.

Ejemplo

El ejemplo siguiente copia los fotogramas seleccionados en el portapapeles:

```
fl.getDocumentDOM().getTimeline().copyFrames();
```

El ejemplo siguiente copia desde el Fotograma 2 hasta el Fotograma 10, no incluido, en el portapapeles (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().copyFrames(1, 9);
```

El ejemplo siguiente copia el Fotograma 5 en el portapapeles:

```
fl.getDocumentDOM().getTimeline().copyFrames(4);
```

timeline.copyMotion()

Disponibilidad
Flash CS3 Professional

Uso
`timeline.copyMotion()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción
Método; copia movimiento en los fotogramas seleccionados, de una interpolación de movimiento o de una animación fotograma por fotograma. Luego puede utilizar `timeline.pasteMotion()` para aplicar el movimiento a otros fotogramas.

Para copiar movimiento como texto (código) que se pueda pegar en un script, consulte `timeline.copyMotionAsAS3()`.

Ejemplo
El ejemplo siguiente copia el movimiento del fotograma o fotogramas seleccionados en el portapapeles:

```
fl.getDocumentDOM().getTimeline().copyMotion();
```

Véase también
`timeline.copyMotionAsAS3()`, `timeline.pasteMotion()`

timeline.copyMotionAsAS3()

Disponibilidad
Flash CS3 Professional

Uso
`timeline.copyMotionAsAS3()`

Parámetros
Ninguno.

Valor devuelto
Ninguno.

Descripción

Método; copia el movimiento en los fotogramas seleccionados como código ActionScript 3.0 en el portapapeles, bien desde una interpolación de movimiento o mediante una animación fotograma a fotograma. Luego puede pegar este código en un script.

Para copiar movimiento en un formato aplicable a otros fotogramas, consulte `timeline.copyMotion()`.

Ejemplo

El ejemplo siguiente copia el movimiento del fotograma o fotogramas seleccionados en el portapapeles como código ActionScript 3.0:

```
fl.getDocumentDOM().getTimeline().copyMotionAsAS3();
```

Véase también

`timeline.copyMotion()`

timeline.createMotionTween()

Disponibilidad

Flash MX 2004

Uso

```
timeline.createMotionTween([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el fotograma inicial en el que se creará una interpolación de movimiento. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se detendrá la interpolación de movimiento. El intervalo de fotogramas llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; define la propiedad `frame.tweenType` como `motion` para cada fotograma clave seleccionado en la capa actual y, si es necesario, convierte el contenido de cada fotograma en una instancia de símbolo único. Esta propiedad equivale al elemento de menú Crear interpolación de movimiento de la herramienta de edición de Flash.

Ejemplo

El ejemplo siguiente convierte la forma del primer fotograma hasta el Fotograma 10, no incluido, en una instancia de un símbolo gráfico y define `frame.tweenType` como `motion` (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().createMotionTween(0, 9);
```

timeline.currentFrame

Disponibilidad
Flash MX 2004

Uso
`timeline.currentFrame`

Descripción
Propiedad; un índice basado en cero para el fotograma en la ubicación actual de la cabeza lectora.

Ejemplo
El ejemplo siguiente define la cabeza lectora de la línea de tiempo actual en el Fotograma 10 (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().currentFrame = 9;
```

El ejemplo siguiente almacena el valor de la ubicación de la cabeza lectora actual en la variable `curFrame`:

```
var curFrame = fl.getDocumentDOM().getTimeline().currentFrame;
```

timeline.currentLayer

Disponibilidad
Flash MX 2004

Uso
`timeline.currentLayer`

Descripción
Propiedad; el índice basado en cero para la capa activa actual. Un valor de 0 especifica la capa superior, un valor de 1 especifica la capa que se encuentra por debajo, y así sucesivamente.

Ejemplo
El ejemplo siguiente convierte en activa la capa superior:

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
```

El ejemplo siguiente almacena el índice de la capa activa actualmente en la variable `curLayer`:

```
var curLayer = fl.getDocumentDOM().getTimeline().currentLayer;
```

timeline.cutFrames()

Disponibilidad
Flash MX 2004

Uso

```
timeline.cutFrames([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el comienzo del intervalo de fotogramas que se va a cortar. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se dejará de cortar. El intervalo de fotogramas llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; corta un intervalo de fotogramas de la capa actual en la línea de tiempo y lo guarda en el portapapeles.

Ejemplo

El ejemplo siguiente corta los fotogramas seleccionados en la línea de tiempo y los guarda en el portapapeles:

```
fl.getDocumentDOM().getTimeline().cutFrames();
```

El ejemplo siguiente corta desde el Fotograma 2 hasta el Fotograma 10, no incluido, en la línea de tiempo y los guarda en el portapapeles (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().cutFrames(1, 9);
```

El ejemplo siguiente corta el Fotograma 5 en la línea de tiempo y lo guarda en el portapapeles:

```
fl.getDocumentDOM().getTimeline().cutFrames(4);
```

timeline.deleteLayer()

Disponibilidad

Flash MX 2004

Uso

```
timeline.deleteLayer([index])
```

Parámetros

`index` Un índice basado en cero que especifica la capa que se va a eliminar. Si sólo hay una capa en la línea de tiempo, este método no tiene ningún efecto. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; elimina una capa. Si la capa es una carpeta, se eliminarán todas las capas dentro de la carpeta. Si no especifica el índice de capa, Flash eliminará las capas seleccionadas actualmente.

Ejemplo

El ejemplo siguiente elimina la segunda capa desde la parte superior:

```
fl.getDocumentDOM().getTimeline().deleteLayer(1);
```

El ejemplo siguiente elimina las capas seleccionadas actualmente:

```
fl.getDocumentDOM().getTimeline().deleteLayer();
```

timeline.expandFolder()

Disponibilidad

Flash MX 2004

Uso

```
timeline.expandFolder(bExpand [, bRecurseNestedParents [, index]])
```

Parámetros

`bExpand` Un valor booleano que, si se define como `true`, hace que el método expanda la carpeta; `false` hace que el método contraiga la carpeta.

`bRecurseNestedParents` Un valor booleano que, si se define como `true`, hace que todas las capas de la carpeta especificada se abran o cierren, según el parámetro *bExpand*. Este parámetro es opcional.

`index` Un índice basado en cero de la carpeta que se va a expandir o contraer. Utilice -1 para aplicar a todas las capas (deberá definir también *bRecurseNestedParents* como `true`). Esta propiedad equivale a los elementos de menú Expandir todas/Contraer todas de la herramienta de edición de Flash. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; expande o contrae la carpeta o carpetas especificadas. Si no especifica una carpeta, este método actúa sobre la capa actual.

Ejemplo

Los ejemplos siguientes utilizan esta estructura de carpetas:

```
Folder 1 ***
--layer 7
--Folder 2 ****
----Layer 5
```

El ejemplo siguiente expande sólo la Carpeta 1:

```
fl.getDocumentDOM().getTimeline().currentLayer = 1;
fl.getDocumentDOM().getTimeline().expandFolder(true);
```

El ejemplo siguiente expande sólo la Carpeta 1 (suponiendo que la Carpeta 2 se contrajo al contraerse por última vez la Carpeta 1; en caso contrario, la Carpeta 2 aparece expandida):

```
fl.getDocumentDOM().getTimeline().expandFolder(true, false, 0);
```

El ejemplo siguiente contrae todas las carpetas de la línea de tiempo actual:

```
fl.getDocumentDOM().getTimeline().expandFolder(false, true, -1);
```

timeline.findLayerIndex()

Disponibilidad
Flash MX 2004

Uso
`timeline.findLayerIndex(name)`

Parámetros
`name` Una cadena que especifica el nombre de la carpeta que se va a buscar.

Valor devuelto
Un conjunto de valores de índice para la capa especificada. Si no se encuentra la capa especificada, Flash devuelve `undefined`.

Descripción
Método; busca un conjunto de índices para las capas con un determinado nombre. El índice de capa es plano, por lo que las carpetas se consideran parte del índice principal.

Ejemplo
El ejemplo siguiente muestra los valores de índice de todas las capas denominadas `Capa 7` en el panel Salida:

```
var layerIndex = fl.getDocumentDOM().getTimeline().findLayerIndex("Layer 7");
fl.trace(layerIndex);
```

El ejemplo siguiente ilustra cómo se transfieren los valores devueltos desde este método hasta `timeline.setSelectedLayers()`:

```
var layerIndex = fl.getDocumentDOM().getTimeline().findLayerIndex("Layer 1");
fl.getDocumentDOM().getTimeline().setSelectedLayers(layerIndex[0], true);
```

timeline.frameCount

Disponibilidad
Flash MX 2004

Uso
`timeline.frameCount`

Descripción
Propiedad de sólo lectura; un entero que representa el número de fotogramas en la capa más larga de esta línea de tiempo.

Ejemplo

El ejemplo siguiente utiliza una variable `countNum` para almacenar el número de fotogramas de la capa más larga del documento actual:

```
var countNum = fl.getDocumentDOM().getTimeline().frameCount;
```

timeline.getFrameProperty()

Disponibilidad

Flash MX 2004

Uso

```
timeline.getFrameProperty(property [, startframeIndex [, endFrameIndex]])
```

Parámetros

`property` Una cadena que especifica el nombre de la propiedad para la que se obtiene el valor. Consulte el resumen de propiedades del Objeto Frame para obtener una lista completa de las propiedades.

`startFrameIndex` Un índice basado en cero que especifica el número de fotograma inicial para el que se obtendrá el valor. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el final del intervalo de fotogramas que se va a seleccionar. El intervalo llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Un valor para la propiedad especificada o `undefined` si todos los fotogramas seleccionados no tienen el mismo valor de propiedad.

Descripción

Método; recupera el valor de la propiedad especificada para los fotogramas seleccionados.

Ejemplo

El ejemplo siguiente recupera el nombre del primer fotograma de la capa superior del documento actual y muestra el nombre en el panel Salida:

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
fl.getDocumentDOM().getTimeline().setSelectedFrames(0, 0, true);
var frameName = fl.getDocumentDOM().getTimeline().getFrameProperty("name");
fl.trace(frameName);
```

timeline.getGuidelines()

Disponibilidad

Flash CS4 Professional

Uso

```
timeline.getGuidelines()
```

Parámetros
Ninguno.

Valor devuelto
Una cadena XML.

Descripción
Método: devuelve una cadena XML que representa las posiciones actuales de las guías horizontal y vertical de una línea de tiempo (Ver > Guías > Mostrar guías). Para aplicar estas líneas de guía a una línea de tiempo, utilice `timeline.setGuidelines()`.

Ejemplo
Suponiendo que haya algunas líneas de guía en la primera línea de tiempo, el siguiente ejemplo las muestra como una cadena XML en el panel Salida:

```
var currentTimeline = fl.getDocumentDOM().timelines[0];
fl.trace(currentTimeline.getGuidelines());
```

timeline.getLayerProperty()

Disponibilidad
Flash MX 2004

Uso
`timeline.getLayerProperty(property)`

Parámetros
`property` Una cadena que especifica el nombre de la propiedad cuyo valor desea recuperar. Para ver una lista de las propiedades, consulte la tabla del resumen de propiedades del Objeto Frame.

Valor devuelto
Valor de la propiedad especificada. Flash examina las propiedades de la capa para determinar el tipo. Si no todas las capas especificadas tienen el mismo valor de propiedad, Flash devuelve `undefined`.

Descripción
Método; recupera el valor de la propiedad especificada para las capas seleccionadas.

Ejemplo
El ejemplo siguiente recupera el nombre de la capa superior del documento actual y lo muestra en el panel Salida:

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
var layerName = fl.getDocumentDOM().getTimeline().getLayerProperty("name");
fl.trace(layerName);
```

timeline.getSelectedFrames()

Disponibilidad
Flash MX 2004

Parámetros
Ninguno.

Valor devuelto
Un conjunto que contiene 3*n* enteros, donde *n* es el número de regiones seleccionadas. El primer entero de cada grupo es el índice de capa, el segundo entero es el fotograma inicial del comienzo de la selección y el tercer entero especifica el fotograma final del intervalo de esa selección. El fotograma final no se incluye en la selección.

Descripción
Método; recupera los fotogramas seleccionados actualmente en un conjunto.

Ejemplo
Con la capa superior como capa actual, el ejemplo siguiente muestra `0,5,10,0,20,25` en el panel Salida:

```
var timeline = fl.getDocumentDOM().getTimeline();
timeline.setSelectedFrames(5,10);
timeline.setSelectedFrames(20,25,false);
var theSelectedFrames = timeline.getSelectedFrames();
fl.trace(theSelectedFrames);
```

Véase también
`timeline.setSelectedFrames()`

timeline.getSelectedLayers()

Disponibilidad
Flash MX 2004

Parámetros
Ninguno.

Valor devuelto
Un conjunto de valores de índice basado en cero de las capas seleccionadas.

Descripción
Método; recupera los valores de índice basado en cero de las capas seleccionadas actualmente.

Ejemplo
El ejemplo siguiente muestra `1,0` en el panel Salida:

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(0);
fl.getDocumentDOM().getTimeline().setSelectedLayers(1, false);
var layerArray = fl.getDocumentDOM().getTimeline().getSelectedLayers();
fl.trace(layerArray);
```

Véase también

`timeline.setSelectedLayers()`

timeline.insertBlankKeyframe()

Disponibilidad

Flash MX 2004

Uso

```
timeline.insertBlankKeyframe([frameNumIndex])
```

Parámetros

`frameNumIndex` Un índice basado en cero que especifica el fotograma en el que se insertará el fotograma clave. Si omite *frameNumIndex*, el método utilizará el número de fotograma de la cabeza lectora actual. Este parámetro es opcional.

Si el fotograma especificado o seleccionado es normal, el fotograma clave se insertará en él. Por ejemplo, si tiene un intervalo de 10 fotogramas con números 1-10 y selecciona el Fotograma 5, este método convertirá el Fotograma 5 en un fotograma clave en blanco y la extensión del intervalo de fotogramas seguirá siendo 10 fotogramas. Si selecciona el Fotograma 5 y hay un fotograma clave con un fotograma normal contiguo, este método inserta un fotograma clave vacío en el Fotograma 6. Si el Fotograma 5 es un fotograma clave y el fotograma contiguo no lo es, no se inserta ningún fotograma clave, pero la cabeza lectora avanza hasta el Fotograma 6.

Valor devuelto

Ninguno.

Descripción

Método; inserta un fotograma clave en blanco en el índice de fotograma especificado; si no se especifica el índice, el método inserta el fotograma clave en blanco utilizando la cabeza lectora/selección. Véase también `timeline.insertKeyframe()`.

Ejemplo

El ejemplo siguiente inserta un fotograma clave en blanco en el Fotograma 20 (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().insertBlankKeyframe(19);
```

El ejemplo siguiente inserta un fotograma clave en blanco en el fotograma seleccionado actualmente (o la ubicación de la cabeza lectora si no hay ningún fotograma seleccionado):

```
fl.getDocumentDOM().getTimeline().insertBlankKeyframe();
```

timeline.insertFrames()

Disponibilidad
Flash MX 2004

Uso
```
timeline.insertFrames([numFrames [, bAllLayers [, frameNumIndex]]])
```

Parámetros
`numFrames` Un entero que especifica el número de fotogramas que se van a insertar. Si omite este parámetro, el método insertará fotogramas en la selección actual de la capa actual. Este parámetro es opcional.

`bAllLayers` Un valor booleano que, si se define como `true` (valor predeterminado), hace que el método inserte en todas las capas el número especificado de fotogramas en el parámetro *numFrames*; si se define como `false`, el método insertará fotogramas en la capa actual. Este parámetro es opcional.

`frameNumIndex` Un índice basado en cero que especifica el fotograma en el que se insertará un nuevo fotograma. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; inserta el número especificado de fotogramas en el índice especificado.

Si no se especifican parámetros, este método funciona de este modo:

- Si hay uno o varios fotogramas seleccionados, el método inserta el número de fotogramas seleccionado en la ubicación del primer fotograma seleccionado en la capa actual. Es decir, si se seleccionan los fotogramas 6 a 10 (un total de cinco fotogramas), el método añade cinco fotogramas al Fotograma 6 de la capa que contiene los fotogramas seleccionados.
- Si no hay fotogramas seleccionados, el método inserta un fotograma en el fotograma actual en todas las capas.

  Si se especifican parámetros, el método funciona de este modo:

- Si sólo se especifica *numFrames*, inserta el número especificado de fotogramas en el fotograma actual en la capa actual.
- Si se especifica *numFrames* y *bAllLayers* es `true`, inserta el número especificado de fotogramas en el fotograma actual en todas las capas.
- Si se especifican los tres parámetros, inserta el número especificado de fotogramas en el índice especificado (*frameIndex*); el valor transferido para *bAllLayers* determina si los fotogramas sólo se añaden a la capa actual o a todas las capas.

  Si el fotograma especificado o seleccionado es normal, el fotograma se insertará en él. Por ejemplo, si tiene un intervalo de 10 fotogramas con números 1-10 y selecciona el Fotograma 5 (o transfiere un valor de 4 para *frameIndex*), este método añadirá un fotograma en el Fotograma 5 y la extensión del intervalo de fotogramas será de 11. Si selecciona el Fotograma 5 y es un fotograma clave, este método insertará un fotograma en el Fotograma 6 con independencia de si el fotograma situado junto a él también es un fotograma clave.

Ejemplo
El ejemplo siguiente inserta un fotograma (o fotogramas, según la selección) en la ubicación actual de la capa actual:

```
fl.getDocumentDOM().getTimeline().insertFrames();
```

El ejemplo siguiente inserta cinco fotogramas en el fotograma actual en todas las capas:

```
fl.getDocumentDOM().getTimeline().insertFrames(5);
```

Nota: *si tiene varias capas que contienen fotogramas y selecciona un fotograma en una capa cuando utiliza el comando anterior, Flash sólo insertará los fotogramas en la capa seleccionada. Si tiene varias capas sin fotogramas seleccionados en ellas, Flash insertará los fotogramas en todas las capas.*

El ejemplo siguiente inserta tres fotogramas en la capa actual únicamente:

```
fl.getDocumentDOM().getTimeline().insertFrames(3, false);
```

El ejemplo siguiente inserta cuatro fotogramas en todas las capas, comenzando desde el primer fotograma:

```
fl.getDocumentDOM().getTimeline().insertFrames(4, true, 0);
```

timeline.insertKeyframe()

Disponibilidad

Flash MX 2004

Uso

```
timeline.insertKeyframe([frameNumIndex])
```

Parámetros

`frameNumIndex` Un índice basado en cero que especifica el índice de fotograma en el que se insertará el fotograma clave en la capa actual. Si omite *frameNumIndex*, el método utilizará el número de fotograma de la cabeza lectora actual o el fotograma seleccionado. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; inserta un fotograma clave en el fotograma especificado. Si omite el parámetro, el método insertará un fotograma clave utilizando la ubicación de la cabeza lectora o de la selección.

Este método funciona igual que `timeline.insertBlankKeyframe()`, salvo que el fotograma clave insertado alberga el contenido del fotograma que ha convertido (es decir, no está en blanco).

Ejemplo

El ejemplo siguiente inserta un fotograma clave en la ubicación de la cabeza lectora o la selección:

```
fl.getDocumentDOM().getTimeline().insertKeyframe();
```

El ejemplo siguiente inserta un fotograma clave en el Fotograma 10 de la segunda capa (recuerde que los valores de índice son distintos de los valores de número de fotograma o capa):

```
fl.getDocumentDOM().getTimeline().currentLayer = 1;
fl.getDocumentDOM().getTimeline().insertKeyframe(9);
```

timeline.layerCount

Disponibilidad
Flash MX 2004

Uso
`timeline.layerCount`

Descripción
Propiedad de sólo lectura; un entero que representa el número de capas en la línea de tiempo especificada.

Ejemplo
El ejemplo siguiente utiliza la variable `NumLayer` para almacenar el número de capas de la escena actual:

```
var NumLayer = fl.getDocumentDOM().getTimeline().layerCount;
```

timeline.layers

Disponibilidad
Flash MX 2004

Uso
`timeline.layers`

Descripción
Propiedad de sólo lectura; un conjunto de objetos Layer.

Ejemplo
El ejemplo siguiente utiliza la variable `currentLayers` para almacenar el conjunto de objetos Layer del documento actual:

```
var currentLayers = fl.getDocumentDOM().getTimeline().layers;
```

timeline.name

Disponibilidad
Flash MX 2004

Uso
`timeline.name`

Descripción
Propiedad; una cadena que especifica el nombre de la línea de tiempo actual. Este nombre es el nombre de la escena, pantalla (diapositiva o formulario) o símbolo que se está editando.

Ejemplo

El ejemplo siguiente recupera el nombre de la primera escena:

```
var sceneName = fl.getDocumentDOM().timelines[0].name;
```

El ejemplo siguiente define el nombre de la primera escena como `FirstScene`:

```
fl.getDocumentDOM().timelines[0].name = "FirstScene";
```

timeline.pasteFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.pasteFrames([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el comienzo del intervalo de fotogramas que se va a pegar. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se dejará de pegar fotogramas. El método pega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; pega el intervalo de fotogramas del portapapeles a los fotogramas especificados.

Ejemplo

El ejemplo siguiente pega los fotogramas del portapapeles al fotograma seleccionado o la ubicación de la cabeza lectora:

```
fl.getDocumentDOM().getTimeline().pasteFrames();
```

El ejemplo siguiente pega los fotogramas desde el Fotograma 2 hasta el 10, no incluido (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().pasteFrames(1, 9);
```

El ejemplo siguiente pega los fotogramas del portapapeles comenzando en el Fotograma 5:

```
fl.getDocumentDOM().getTimeline().pasteFrames(4);
```

timeline.pasteMotion()

Disponibilidad

Flash CS3 Professional

Uso

```
timeline.pasteMotion()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; pega el intervalo de fotogramas de movimiento recuperado por `timeline.copyMotion()` en la línea de tiempo. Si es necesario, se desplazan (a la derecha) los fotogramas existentes para dejar espacio a los que se van a pegar.

Ejemplo

El ejemplo siguiente pega el movimiento del portapapeles en el fotograma actualmente seleccionado o en la ubicación de la cabeza lectora, desplazando el fotograma a la derecha de los fotogramas pegados:

```
fl.getDocumentDOM().getTimeline().pasteMotion();
```

Véase también

`timeline.copyMotion()`

timeline.removeFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.removeFrames([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el primer fotograma en el que se iniciará la eliminación de fotogramas. Si omite *startFrameIndex*, el método utiliza la selección actual; si no hay selección, se eliminarán todos los fotogramas en la cabeza lectora actual en todas las capas. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se dejará de eliminar fotogramas; el intervalo de fotogramas llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; elimina el fotograma.

Ejemplo

El ejemplo siguiente convierte desde el Fotograma 5 hasta el Fotograma 10 (no incluido) de la capa superior en la escena actual (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
fl.getDocumentDOM().getTimeline().removeFrames(4, 9);
```

El ejemplo siguiente elimina el Fotograma 8 en la capa superior de la escena actual:

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
fl.getDocumentDOM().getTimeline().removeFrames(7);
```

timeline.reorderLayer()

Disponibilidad

Flash MX 2004

Uso

```
timeline.reorderLayer(layerToMove, layerToPutItBy [, bAddBefore])
```

Parámetros

`layerToMove` Un índice basado en cero que especifica qué capa se va a mover.

`layerToPutItBy` Un índice basado en cero que especifica junto a qué capa desea mover la capa. Por ejemplo, si especifica 1 para *layerToMove* y 0 para *layerToPutItBy*, la segunda capa se situará junto a la primera capa.

`bAddBefore` Especifica si se va a mover la capa delante o detrás de *layerToPutItBy*. Si especifica `false`, la capa se moverá detrás de *layerToPutItBy*. El valor predeterminado es `true`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; mueve la primera capa especificada delante o detrás de la segunda capa especificada.

Ejemplo

El ejemplo siguiente mueve la capa en el índice 2 a la parte superior (sobre la capa en el índice 0):

```
fl.getDocumentDOM().getTimeline().reorderLayer(2, 0);
```

El ejemplo siguiente sitúa la capa en el índice detrás de la capa en el índice 5:

```
fl.getDocumentDOM().getTimeline().reorderLayer(3, 5, false);
```

timeline.reverseFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.reverseFrames([startFrameIndex [, endFrameIndex]])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el primer fotograma en el que se iniciará la inversión de fotogramas. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el fotograma en el que se dejará de invertir fotogramas; el intervalo de fotogramas llega hasta *endFrameIndex*, no incluido. Si sólo especifica *startFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; invierte un intervalo de fotogramas.

Ejemplo

El ejemplo siguiente invierte las posiciones de los fotogramas seleccionados actualmente:

```
fl.getDocumentDOM().getTimeline().reverseFrames();
```

El ejemplo siguiente invierte los fotogramas desde el Fotograma 10 hasta el 15, no incluido (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
fl.getDocumentDOM().getTimeline().reverseFrames(9, 14);
```

timeline.selectAllFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.selectAllFrames()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; selecciona todos los fotogramas de la línea de tiempo actual.

Ejemplo

El ejemplo siguiente selecciona todos los fotogramas de la línea de tiempo actual.

```
fl.getDocumentDOM().getTimeline().selectAllFrames();
```

timeline.setFrameProperty()

Disponibilidad
Flash MX 2004

Uso
```
timeline.setFrameProperty(property, value [, startFrameIndex [, endFrameIndex]])
```

Parámetros
`property` Una cadena que especifica el nombre de la propiedad que se va a modificar. Para ver una lista completa de propiedades y valores, consulte el resumen de propiedades del Objeto Frame.

No es posible utilizar este método para definir valores para propiedades de sólo lectura, como `frame.duration` y `frame.elements`.

`value` Especifica el valor con el que desea definir la propiedad. Para determinar los valores y el tipo adecuados, consulte el resumen de propiedades del Objeto Frame.

`startFrameIndex` Un índice basado en cero que especifica el número del fotograma inicial que desea modificar. Si omite *startFrameIndex*, el método utilizará la selección actual. Este parámetro es opcional.

`endFrameIndex` Un índice basado en cero que especifica el primer fotograma en el que se detendrá. El intervalo de fotogramas llega hasta *endFrameIndex*, no incluido. Si especifica *startFrameIndex* pero omite *endFrameIndex*, *endFrameIndex* utilizará de forma predeterminada el valor de *startFrameIndex*. Este parámetro es opcional.

Valor devuelto
Ninguno.

Descripción
Método; establece la propiedad del objeto Frame para los fotogramas seleccionados.

Ejemplo
El ejemplo siguiente asigna el comando ActionScript `stop()` al primer fotograma de la capa superior del documento actual:

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
fl.getDocumentDOM().getTimeline().setSelectedFrames(0,0,true);
fl.getDocumentDOM().getTimeline().setFrameProperty("actionScript", "stop();");
```

El ejemplo siguiente establece una interpolación de movimiento desde el Fotograma 2 hasta el 5, no incluido (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
var doc = fl.getDocumentDOM();
doc.getTimeline().setFrameProperty("tweenType","motion",1,4);
```

timeline.setGuidelines()

Disponibilidad
Flash CS4 Professional

Uso

```
timeline.setGuidelines(xmlString)
```

Parámetros

`xmlString` Una cadena XML que contiene información sobre las líneas de guía que se deben aplicar.

Valor devuelto

Un valor booleano `true` si se aplican correctamente las líneas de guía, y `false` en caso contrario.

Descripción

Método: sustituye las líneas de guía de la línea de tiempo (Ver > Guías > Mostrar guías) con la información especificada en *xmlString*. Para recuperar una cadena XML que pueda transferirse a este método, utilice `timeline.getGuidelines()`.

Para poder ver las nuevas líneas de guía establecidas, tal vez deba ocultarlas primero.

Ejemplo

El siguiente ejemplo aplica las líneas de guía de un archivo FLA a otro:

```
var doc0 = fl.documents[0];
var guides0 = doc0.timelines[0].getGuidelines();
var doc1 = fl.documents[1];
doc1.timelines[0].setGuidelines(guides0);
```

timeline.setLayerProperty()

Disponibilidad

Flash MX 2004

Uso

```
timeline.setLayerProperty(property, value [, layersToChange])
```

Parámetros

`property` Una cadena que especifica la propiedad que se va a establecer. Para ver una lista de propiedades, consulte "Objeto Layer" en la página 315.

`value` El valor con el que desea definir la propiedad. Utilice el mismo tipo de valor que utilizaría para definir la propiedad en el objeto Layer.

`layersToChange` Una cadena que identifica qué capas deben modificarse. Los valores aceptables son `"selected"`, `"all"` y `"others"`. El valor predeterminado es `"selected"` si omite este parámetro. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; define con un valor determinado la propiedad especificada en todas las capas seleccionadas

Ejemplo

El ejemplo siguiente hace que la capa o capas seleccionadas sean invisibles:

```
fl.getDocumentDOM().getTimeline().setLayerProperty("visible", false);
```

El ejemplo siguiente define el nombre de la capa o capas seleccionadas como `selLayer`:

```
fl.getDocumentDOM().getTimeline().setLayerProperty("name", "selLayer");
```

timeline.setSelectedFrames()

Disponibilidad

Flash MX 2004

Uso

```
timeline.setSelectedFrames(startFrameIndex, endFrameIndex [, bReplaceCurrentSelection])
timeline.setSelectedFrames(selectionList [, bReplaceCurrentSelection])
```

Parámetros

`startFrameIndex` Un índice basado en cero que especifica el fotograma inicial que desea definir.

`endFrameIndex` Un índice basado en cero que especifica el final de la selección; *endFrameIndex* es el fotograma que va detrás del último fotograma del intervalo que se va a seleccionar.

`bReplaceCurrentSelection` Un valor booleano que, si se define como `true`, hace que se anule la selección los fotogramas seleccionados actualmente antes de seleccionar los fotogramas especificados. El valor predeterminado es `true`.

`selectionList` Un conjunto de tres enteros que devuelve `timeline.getSelectedFrames()`.

Valor devuelto

Ninguno.

Descripción

Método; selecciona un intervalo de fotogramas de la capa actual o define los fotogramas seleccionadas con el conjunto de selección transferido a este método.

Ejemplo

El ejemplo siguiente muestra dos formas de seleccionar la capa superior, desde el Fotograma 1 hasta el Fotograma 10 (no incluido), y a continuación, añade a la selección actual desde el Fotograma 12 hasta el Fotograma 15 (no incluido) de la misma capa (recuerde que los valores de índice son distintos de los valores de número de fotograma):

```
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);
```

El ejemplo siguiente almacena en primer lugar el conjunto de fotogramas seleccionados en la variable `savedSelectionList` y, a continuación, utiliza posteriormente el conjunto en el código para volver a seleccionar esos fotogramas cuando un comando o la interacción del usuario ha cambiado la selección:

```
var savedSelectionList = fl.getDocumentDOM().getTimeline().getSelectedFrames();
// Do something that changes the selection.
fl.getDocumentDOM().getTimeline().setSelectedFrames(savedSelectionList);
```

Véase también

`timeline.getSelectedFrames()`

timeline.setSelectedLayers()

Disponibilidad

Flash MX 2004

Uso

```
timeline.setSelectedLayers(index [, bReplaceCurrentSelection])
```

Parámetros

`index` Un índice basado en cero para la capa que desea seleccionar.

`bReplaceCurrentSelection` Un valor booleano que, si se define como `true`, hace que el método reemplace la selección actual; `false` hace que el método amplíe la selección actual. El valor predeterminado es `true`. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; define la capa que se va a seleccionar y, además, convierte la capa especificada en la capa actual. Al seleccionar una capa se seleccionan todos los fotogramas de la capa.

Ejemplo

El ejemplo siguiente selecciona la capa superior:

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(0);
```

El ejemplo siguiente añade la capa siguiente a la selección:

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(1, false);
```

Véase también

`timeline.getSelectedLayers()`

timeline.showLayerMasking()

Disponibilidad

Flash MX 2004

Uso

```
timeline.showLayerMasking([layer])
```

Parámetros

`layer` Un índice basado en cero de una máscara o capa enmascarada para mostrar enmascaramiento durante la edición. Este parámetro es opcional.

Valor devuelto

Ninguno.

Descripción

Método; muestra el enmascaramiento de capas durante la edición bloqueando la máscara y las capas enmascaradas. Este método utiliza la capa actual si no hay ninguna capa especificada. Si utiliza este método en una capa que no es de tipo Máscara o Enmascarada, Flash muestra un error en el panel Salida.

Ejemplo

El ejemplo siguiente especifica que el enmascaramiento de capa de la primera capa debe mostrarse durante la edición.

```
fl.getDocumentDOM().getTimeline().showLayerMasking(0);
```

Capítulo 48: Objeto ToolObj

Disponibilidad

Flash MX 2004

Descripción

Un objeto ToolObj representa una herramienta individual en el panel Herramientas. Para acceder a un objeto ToolObj, utilice las propiedades del Objeto Tools: el conjunto `tools.toolObjs` o `tools.activeTool`.

Resumen de métodos

Los métodos siguientes están disponibles para el objeto ToolObj.

Nota: *los métodos siguientes sólo se utilizan para crear herramientas ampliables.*

Método	Descripción
`toolObj.enablePIControl()`	Activa o desactiva el control especificado en un inspector de propiedades. Sólo se utiliza para crear herramientas ampliables.
`toolObj.setIcon()`	Identifica un archivo PNG para utilizarlo como icono de herramienta en el panel Herramientas de Flash.
`toolObj.setMenuString()`	Define la cadena que aparece en el menú emergente como nombre de la herramienta.
`toolObj.setOptionsFile()`	Asocia un archivo XML con la herramienta.
`toolObj.setPI()`	Define un determinado inspector de propiedades para utilizarlo cuando se active la herramienta.
`toolObj.setToolName()`	Asigna un nombre a la herramienta para la configuración del panel Herramientas.
`toolObj.setToolTip()`	Define la sugerencia que aparece cuando el ratón se mantiene sobre el icono de herramienta.
`toolObj.showPIControl()`	Muestra u oculta un control en el inspector de propiedades.
`toolObj.showTransformHandles()`	Se llama en el método `configureTool()` de un archivo JavaScript de una herramienta ampliable para indicar que los controladores de transformación libre deben aparecer cuando la herramienta está activa.

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto ToolObj:

Propiedad	Descripción
`toolObj.depth`	Un entero que especifica la profundidad de la herramienta en el menú emergente del panel Herramientas.
`toolObj.iconID`	Un entero que especifica el ID de recurso de la herramienta.
`toolObj.position`	De sólo lectura; un entero que especifica la posición de la herramienta en el panel Herramientas.

toolObj.depth

Disponibilidad
Flash MX 2004

Uso
```
toolObj.depth
```

Descripción
Propiedad de sólo lectura; un entero que especifica la profundidad de la herramienta en el menú emergente del panel Herramientas. Esta propiedad sólo se utiliza para crear herramientas ampliables.

Ejemplo
El siguiente ejemplo especifica que la herramienta tiene una profundidad de 1, lo que indica un nivel por debajo de una herramienta del panel Herramientas:

```
fl.tools.activeTool.depth = 1;
```

toolObj.enablePIControl()

Disponibilidad
Flash MX 2004

Uso
```
toolObj.enablePIControl(control, bEnable)
```

Parámetros
`control` Una cadena que especifica el nombre del control que se va a activar o desactivar. Los valores válidos dependen del inspector de propiedades que invoque esta herramienta; consulte toolObj.setPI().

Un inspector de propiedades de forma cuenta con los controles siguientes:

stroke	fill

Un inspector de propiedades de texto cuenta con los controles siguientes:

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

Un inspector de propiedades de película cuenta con los controles siguientes:

size	publish	background
framerate	player	profile

bEnable Un valor booleano que determina si se activa (`true`) o desactiva (`false`) el control.

Valor devuelto
Ninguno.

Descripción
Método; activa o desactiva el control especificado en un inspector de propiedades. Sólo se utiliza para crear herramientas ampliables.

Ejemplo
El comando siguiente en un archivo JavaScript de una herramienta ampliable configura Flash para que no muestre las opciones de trazo en el inspector de propiedades de esa herramienta:

```
theTool.enablePIControl("stroke",false);
```

toolObj.iconID

Disponibilidad
Flash MX 2004

Uso
`toolObj.iconID`

Descripción
Propiedad de sólo lectura; un entero con un valor de -1. Esta propiedad sólo se utiliza cuando se crean herramientas ampliables. Un valor de `iconID` de -1 indica que Flash no intentará buscar un icono para la herramienta. En su lugar, el script de la herramienta deberá especificar el icono que se mostrará en el panel Herramientas; consulte `toolObj.setIcon()`.

Ejemplo
El siguiente ejemplo asigna un valor de -1 (el ID de icono de la herramienta actual) a la variable `toolIconID`:

```
var toolIconID = fl.tools.activeTool.iconID
```

toolObj.position

Disponibilidad
Flash MX 2004

Uso
`toolObj.position`

Descripción
Propiedad de sólo lectura; un entero que especifica la posición de la herramienta en el panel Herramientas. Esta propiedad sólo se utiliza cuando se crean herramientas ampliables.

Ejemplo
Los comandos siguientes del método `mouseDown()` de un archivo JavaScript de una herramienta mostrarán la posición de esa herramienta en el panel Herramientas como un entero en el panel Salida:

```
myToolPos = fl.tools.activeTool.position;
fl.trace(myToolPos);
```

toolObj.setIcon()

Disponibilidad
Flash MX 2004

Uso
`toolObj.setIcon(file)`

Parámetros
`file` Una cadena que especifica el nombre del archivo PNG que se va a utilizar como icono. El archivo PNG se debe colocar en la misma carpeta que el archivo JSFL.

Valor devuelto
Ninguno.

Descripción
Método; identifica un archivo PNG para utilizarlo como icono de herramienta en el panel Herramientas. Este método sólo se utiliza cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente especifica que la imagen del archivo PolyStar.png debe utilizarse como icono para la herramienta llamada `PolyStar`. Este código se toma del archivo de muestra PolyStar.jsfl (consulte "Herramienta PolyStar de muestra" en la página 14):

```
theTool = fl.tools.activeTool;
theTool.setIcon("PolyStar.png");
```

toolObj.setMenuString()

Disponibilidad
Flash MX 2004

Uso
`toolObj.setMenuString(menuStr)`

Parámetros
`menuStr` Una cadena que especifica el nombre que aparece en el menú emergente como nombre de la herramienta.

Valor devuelto
Ninguno.

Descripción
Método; define la cadena que aparece en el menú emergente como nombre de la herramienta. Este método sólo se utiliza cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente especifica que la herramienta llamada `theTool` debe mostrar el nombre "PolyStar Tool" en su menú emergente. Este código se toma del archivo de muestra PolyStar.jsfl (consulte "Herramienta PolyStar de muestra" en la página 14):

```
theTool = fl.tools.activeTool;
theTool.setMenuString("PolyStar Tool");
```

toolObj.setOptionsFile()

Disponibilidad
Flash MX 2004

Uso
`toolObj.setOptionsFile(xmlFile)`

Parámetros
`xmlFile` Una cadena que especifica el nombre del archivo XML que tiene la descripción de las opciones de la herramienta. El archivo XML se debe colocar en la misma carpeta que el archivo JSFL.

Valor devuelto
Ninguno.

Descripción
Método; asocia un archivo XML con la herramienta. El archivo especifica las opciones que van a aparecer en un panel modal que invoca el botón Opciones del inspector de propiedades. Normalmente este método se utilizaría en la función `configureTool()` dentro del archivo JSFL. Consulte `configureTool()`.

Por ejemplo, el archivo PolyStar.xml especifica tres opciones asociadas a la herramienta Polígono:

```
<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>
```

Ejemplo

El ejemplo siguiente especifica que el archivo llamado PolyStar.xml está asociado con la herramienta que se encuentra activa. Este código se toma del archivo de muestra PolyStar.jsfl (consulte "Herramienta PolyStar de muestra" en la página 14):

```
theTool = fl.tools.activeTool;
theTool.setOptionsFile("PolyStar.xml");
```

toolObj.setPI()

Disponibilidad

Flash MX 2004

Uso

```
toolObj.setPI(pi)
```

Parámetros

`pi` Una cadena que especifica el inspector de propiedades que se va a invocar para esta herramienta.

Valor devuelto

Ninguno.

Descripción

Método; especifica qué inspector de propiedades debe utilizarse cuando se active la herramienta. Este método sólo se utiliza cuando se crean herramientas ampliables. Los valores aceptables son `"shape"` (predeterminado), `"text"` y `"movie"`.

Ejemplo

El ejemplo siguiente especifica que debe utilizarse el inspector de propiedades de forma cuando se active la herramienta. Este código se toma del archivo de muestra PolyStar.jsfl (consulte "Herramienta PolyStar de muestra" en la página 14):

```
theTool = fl.tools.activeTool;
theTool.setPI("shape");
```

toolObj.setToolName()

Disponibilidad

Flash MX 2004

Uso

```
toolObj.setToolName(name)
```

Parámetros

`name` Una cadena que especifica el nombre de la herramienta.

Valor devuelto

Ninguno.

Descripción

Método; asigna un nombre a la herramienta para la configuración del panel Herramientas. Este método sólo se utiliza cuando se crean herramientas ampliables. El nombre sólo lo utiliza el archivo de diseño XML que lee Flash para crear el panel Herramientas. El nombre no aparece en la interfaz de usuario de Flash.

Ejemplo

El ejemplo siguiente asigna el nombre `polystar` a la herramienta denominada `theTool`. Este código se toma del archivo de muestra PolyStar.jsfl (consulte "Herramienta PolyStar de muestra" en la página 14):

```
theTool = fl.tools.activeTool;
theTool.setToolName("polystar");
```

toolObj.setToolTip()

Disponibilidad

Flash MX 2004

Uso

```
toolObj.setToolTip(toolTip)
```

Parámetros

`toolTip` Una cadena que especifica la sugerencia que se va a utilizar para la herramienta.

Valor devuelto
Ninguno.

Descripción
Método; define la sugerencia que aparece cuando el ratón se mantiene sobre el icono de herramienta. Este método sólo se utiliza cuando se crean herramientas ampliables.

Ejemplo
El ejemplo siguiente especifica que la sugerencia de la herramienta debe ser `PolyStar Tool`. Este código se toma del archivo de muestra PolyStar.jsfl (consulte "Herramienta PolyStar de muestra" en la página 14):

```
theTool = fl.tools.activeTool;
theTool.setToolTip("PolyStar Tool");
```

toolObj.showPIControl()

Disponibilidad
Flash MX 2004

Uso
```
toolObj.showPIControl(control, bShow)
```

Parámetros
`control` Una cadena que especifica el nombre del control que se va a mostrar u ocultar. Este método sólo se utiliza cuando se crean herramientas ampliables. Los valores válidos dependen del inspector de propiedades que invoque esta herramienta (consulte toolObj.setPI()`toolObj.setPI()`).

Un inspector de propiedades de forma cuenta con los controles siguientes:

stroke	fill

Un inspector de propiedades de texto cuenta con los controles siguientes:

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

El inspector de propiedades de película cuenta con los controles siguientes:

size	publish	background
framerate	player	profile

bShow Un valor booleano que determina si se muestra u oculta el control especificado (`true` muestra el control; `false` oculta el control).

Valor devuelto
Ninguno.

Descripción
Método; muestra u oculta un control en el inspector de propiedades. Este método sólo se utiliza cuando se crean herramientas ampliables.

Ejemplo
El comando siguiente en un archivo JavaScript de una herramienta ampliable configurará Flash para que no muestre las opciones de relleno en el inspector de propiedades de esa herramienta:

```
fl.tools.activeTool.showPIControl("fill", false);
```

toolObj.showTransformHandles()

Disponibilidad
Flash MX 2004

Uso
`toolObj.showTransformHandles(bShow)`

Parámetros
bShow Un valor booleano que determina si se muestran u ocultan los controladores de transformación libre para la herramienta actual (`true` muestra lo controladores; `false` los oculta).

Valor devuelto
Ninguno.

Descripción
Método; se llama en el método `configureTool()` de un archivo JavaScript de una herramienta ampliable para indicar que los controladores de transformación libre deben aparecer cuando la herramienta está activa. Este método sólo se utiliza cuando se crean herramientas ampliables.

Ejemplo
Consulte `configureTool()`.

Capítulo 49: Objeto Tools

Disponibilidad

Flash MX 2004

Descripción

Se puede acceder al objeto Tools desde el objeto Flash (`fl.tools`). La propiedad `tools.toolObjs` contiene un conjunto de objetos ToolObj y la propiedad `tools.activeTool` devuelve el objeto ToolObj para la herramienta activa actualmente. (Véase también Objeto ToolObj y la lista de herramientas ampliables de "Funciones y métodos de nivel superior" en la página 16.)

Nota: *los métodos y las propiedades siguientes sólo se utilizan para crear herramientas ampliables.*

Resumen de métodos

Los métodos siguientes están disponibles para el objeto Tools:

Método	Descripción
`tools.constrainPoint()`	Toma dos puntos y devuelve un nuevo punto ajustado o *restringido*.
`tools.getKeyDown()`	Devuelve la tecla presionada más recientemente.
`tools.setCursor()`	Define el puntero con una apariencia especificada.
`tools.snapPoint()`	Toma un punto como entrada y devuelve un punto nuevo que se puede encajar o *ajustar* al objeto geométrico más próximo.

Resumen de propiedades

Las propiedades siguientes están disponibles para el objeto Tools:

Propiedad	Descripción
`tools.activeTool`	De sólo lectura; devuelve el Objeto ToolObj para la herramienta que se encuentra activa.
`tools.altIsDown`	De sólo lectura; un valor booleano que identifica si se está presionando la tecla Alt.
`tools.ctlIsDown`	De sólo lectura; un valor booleano que identifica si se está presionando la tecla Control.
`tools.mouseIsDown`	De sólo lectura; un valor booleano que identifica si se está presionando el botón izquierdo del ratón.
`tools.penDownLoc`	De sólo lectura; un punto que representa la posición del último evento de pulsación del ratón en el escenario.
`tools.penLoc`	De sólo lectura; un punto que representa la ubicación actual del ratón.
`tools.shiftIsDown`	De sólo lectura; un valor booleano que identifica si se está presionando la tecla Mayús.
`tools.toolObjs`	De sólo lectura; un conjunto de objetos ToolObj.

tools.activeTool

Disponibilidad
Flash MX 2004

Uso
`tools.activeTool`

Descripción
Propiedad de sólo lectura; devuelve el Objeto ToolObj para la herramienta que se encuentra activa.

Ejemplo
El ejemplo siguiente guarda un objeto que representa la herramienta que se encuentra activa en la variable `theTool`:

```
var theTool = fl.tools.activeTool;
```

tools.altIsDown

Disponibilidad
Flash MX 2004

Uso
`tools.altIsDown`

Descripción
Propiedad de sólo lectura; un valor booleano que identifica si se está presionando la tecla Alt. El valor es `true` si la tecla Alt está presionada, y `false` en caso contrario.

Ejemplo
El siguiente ejemplo determina si se está presionando la tecla Alt:

```
var isAltDown = fl.tools.altIsDown;
```

tools.constrainPoint()

Disponibilidad
Flash MX 2004

Uso
`tools.constrainPoint(pt1, pt2)`

Parámetros
`pt1, pt2` Puntos que especifican el punto de inicio del clic y el punto hasta donde se arrastra.

Valor devuelto
Un nuevo punto ajustado o restringido.

Descripción
Método; toma dos puntos y devuelve un nuevo punto ajustado o restringido. Si está presionada la tecla Mayús cuando se ejecuta el comando, el punto devuelto queda restringido a seguir una limitación de 45º (resulta útil para elementos como una línea con flecha) o para restringir un objeto de modo que mantenga su relación de aspecto (como extraer un cuadrado perfecto con la herramienta Rectángulo).

Ejemplo
El ejemplo siguiente devuelve un punto restringido:

```
pt2 = fl.tools.constrainPoint(pt1, tempPt);
```

tools.ctlIsDown

Disponibilidad
Flash MX 2004

Uso
`tools.ctlIsDown`

Descripción
Propiedad de sólo lectura; un valor booleano que es `true` si se presiona la tecla Control, y `false` en caso contrario.

Ejemplo
El ejemplo siguiente determina si se está presionando la tecla Control:

```
var isCtrldown = fl.tools.ctrlIsDown;
```

tools.getKeyDown()

Disponibilidad
Flash MX 2004

Uso
`tools.getKeyDown()`

Parámetros
Ninguno.

Valor devuelto
El valor entero de la tecla.

Descripción
Método; devuelve la tecla presionada más recientemente.

Ejemplo
El ejemplo siguiente muestra el valor de entero de la tecla presionada más recientemente:

```
var theKey = fl.tools.getKeyDown();
fl.trace(theKey);
```

tools.mouseIsDown

Disponibilidad
Flash MX 2004

Uso
`tools.mouseIsDown`

Descripción
Propiedad de sólo lectura; un valor booleano que es `true` si se está presionando el botón izquierdo del ratón y `false` en caso contrario.

Ejemplo
El siguiente ejemplo determina si se presiona el botón izquierdo del ratón.

```
var isMouseDown = fl.tools.mouseIsDown;
```

tools.penDownLoc

Disponibilidad
Flash MX 2004

Uso
`tools.penDownLoc`

Descripción
Propiedad de sólo lectura; un punto que representa la posición del último evento de pulsación del ratón en el escenario. La propiedad `tools.penDownLoc` incluye dos propiedades, *x* e *y*, correspondientes a la posición *x*, *y* del puntero del ratón.

Ejemplo
El ejemplo siguiente determina la posición del último evento de pulsación del ratón en el escenario y muestra los valores *x* e *y* en el panel Salida:

```
var pt1 = fl.tools.penDownLoc;
fl.trace("x,y location of last mouseDown event was " + pt1.x + ", " + pt1.y)
```

Véase también

`tools.penLoc`

tools.penLoc

Disponibilidad

Flash MX 2004

Uso

```
tools.penLoc
```

Descripción

Propiedad de sólo lectura; un punto que representa la ubicación actual del puntero del ratón. La propiedad `tools.penLoc` incluye dos propiedades, *x* e *y*, correspondientes a la posición *x,y* del puntero del ratón.

Ejemplo

El ejemplo siguiente determina la posición actual del ratón:

```
var tempPt = fl.tools.penLoc;
```

Véase también

`tools.penDownLoc`

tools.setCursor()

Disponibilidad

Flash MX 2004

Uso

```
tools.setCursor(cursor)
```

Parámetros

`cursor` Un entero que define la apariencia del puntero, tal como se describe en la lista siguiente:

- 0 = Cursor de signo más (+)
- 1 = Flecha negra
- 2 = Flecha blanca
- 3 = Flecha de cuatro puntas
- 4 = Flecha horizontal de dos puntas
- 5 = Flecha vertical de dos puntas
- 6 = X
- 7 = Cursor de mano

Valor devuelto
Ninguno.

Descripción
Método; define el puntero con una apariencia especificada.

Ejemplo
El ejemplo siguiente define el puntero como una flecha negra.

```
fl.tools.setCursor(1);
```

tools.shiftIsDown

Disponibilidad
Flash MX 2004

Uso
`tools.shiftIsDown`

Descripción
Propiedad de sólo lectura; un valor booleano que es `true` si se presiona la tecla Mayús y `false` en caso contrario.

Ejemplo
El siguiente ejemplo determina si se está presionando la tecla Mayús.

```
var isShiftDown = fl.tools.shiftIsDown;
```

tools.snapPoint()

Disponibilidad
Flash MX 2004

Uso
`tools.snapPoint(pt)`

Parámetros
`pt` Especifica la ubicación del punto para el que desea devolver un punto de ajuste.

Valor devuelto
Un nuevo punto que se puede encajar o ajustar al objeto geométrico más próximo.

Descripción
Método; toma un punto como entrada y devuelve un punto nuevo que se puede encajar o *ajustar* al objeto geométrico más próximo. Si el ajuste está desactivado en el menú Ver de la interfaz de usuario de Flash, el punto devuelto es el punto original.

Ejemplo

El ejemplo siguiente devuelve un nuevo punto que se puede ajustar al objeto geométrico más próximo.

```
var theSnapPoint = fl.tools.snapPoint(pt1);
```

tools.toolObjs

Disponibilidad

Flash MX 2004

Uso

```
tools.toolObjs
```

Descripción

Propiedad de sólo lectura; un conjunto de objetos ToolObj (consulte Objeto ToolObj).

Capítulo 50: Objeto Vertex

Disponibilidad
Flash MX 2004

Descripción
El objeto Vertex forma parte de la estructura de datos de formas que contiene los datos de coordenadas.

Resumen de métodos
Puede emplear los métodos siguientes con el objeto Vertex:

Método	Descripción
`vertex.getHalfEdge()`	Obtiene un Objeto HalfEdge que comparte este vértice.
`vertex.setLocation()`	Define la ubicación del vértice.

Resumen de propiedades
Las propiedades siguientes están disponibles para el objeto Vertex:

Propiedad	Descripción
`vertex.x`	De sólo lectura; la ubicación *x* del vértice en píxeles.
`vertex.y`	De sólo lectura; la ubicación *y* del vértice en píxeles.

vertex.getHalfEdge()

Disponibilidad
Flash MX 2004

Uso
`vertex.getHalfEdge()`

Parámetros
Ninguno.

Valor devuelto
Un Objeto HalfEdge.

Descripción
Método; obtiene un Objeto HalfEdge que comparte este vértice.

Ejemplo
El ejemplo siguiente muestra cómo obtener otros bordes partidos que compartan el mismo vértice:

```
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()

Disponibilidad

Flash MX 2004

Uso

```
vertex.setLocation(x, y)
```

Parámetros

x Un valor de coma flotante que especifica la coordenada *x* de dónde deberá situarse el vértice, en píxeles.

y Un valor de coma flotante que especifica la coordenada *y* de dónde deberá situarse el vértice, en píxeles.

Valor devuelto

Ninguno.

Descripción

Método; define la ubicación del vértice. Deberá llamar a `shape.beginEdit()` antes de utilizar este método.

Ejemplo

El ejemplo siguiente define el vértice en el punto de origen:

```
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

Disponibilidad

Flash MX 2004

Uso

```
vertex.x
```

Descripción

Propiedad de sólo lectura; la ubicación *x* del vértice en píxeles.

Ejemplo

El ejemplo siguiente muestra la ubicación de los valores *x* e *y* del vértice en el panel Salida.

```
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

Disponibilidad

Flash MX 2004

Uso

`vertex.y`

Descripción

Propiedad de sólo lectura; la ubicación *y* del vértice en píxeles.

Ejemplo

Consulte `vertex.x`.

Capítulo 51: Objeto VideoItem

Herencia Objeto Item > Objeto VideoItem

Disponibilidad
Flash MX 2004

Descripción
El objeto VideoItem es una subclase del Objeto Item.

Resumen de métodos
Además de los métodos del objeto Item, el objeto VideoItem cuenta con el siguiente método:

Propiedad	Descripción
`videoItem.exportToFLV()`	Exporta el elemento especificado a un archivo FLV.

Resumen de propiedades
Además de las propiedades del objeto Item, puede utilizar las siguientes propiedades con el objeto VideoItem:

Propiedad	Descripción
`videoItem.fileLastModifiedDate`	Sólo lectura; una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la fecha de modificación del archivo original (en el disco) en el momento en que se importó a la biblioteca.
`videoItem.sourceFileExists`	Sólo lectura; valor booleano que especifica si el archivo importado en la biblioteca sigue existiendo en su ubicación original.
`videoItem.sourceFileIsCurrent`	Sólo lectura; valor booleano que especifica si la fecha de modificación del archivo en el elemento de la biblioteca coincide con la fecha de modificación en disco del archivo importado.
`videoItem.sourceFilePath`	Sólo lectura; cadena que especifica la ruta del elemento de vídeo.
`videoItem.videoType`	Sólo lectura; cadena que especifica el tipo de vídeo que representa el elemento.

videoItem.exportToFLV()

Disponibilidad
Flash CS4 Professional

Uso
```
videoItem.exportToFLV(fileURI)
```

Parámetros
`fileURI` Una cadena, expresada como URI file:///, que especifica la ruta y el nombre del archivo exportado.

Valor devuelto

Un valor booleano de `true` si se exportó correctamente el archivo, y de `false` en caso contrario.

Descripción

Método; exporta el elemento especificado a un archivo FLV.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de vídeo, el siguiente código lo exporta como un archivo FLV:

```
var videoFileURL = "file:///C|/out.flv";
var libItem = fl.getDocumentDOM().library.items[0];
libItem.exportToFLV(videoFileURL);
```

videoItem.fileLastModifiedDate

Disponibilidad

Flash CS4 Professional

Uso

```
videoItem.fileLastModifiedDate
```

Descripción

Propiedad de sólo lectura; una cadena que contiene un número hexadecimal que representa el número de segundos que han transcurrido entre el 1 de enero de 1970 y la fecha de modificación del archivo original (en el disco) en el momento en que se importó a la biblioteca. Si el archivo ya no existe, el valor es "00000000".

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de vídeo, el siguiente código muestra un número hexadecimal, tal como se describe más arriba.

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

Véase también

`videoItem.sourceFileExists`, `videoItem.sourceFileIsCurrent`, `videoItem.sourceFilePath`, `FLfile.getModificationDate()`

videoItem.sourceFileExists

Disponibilidad

Flash CS4 Professional

Uso

```
videoItem.sourceFileExists
```

Descripción

Propiedad de sólo lectura; un valor booleano `true` si el archivo importado en la biblioteca sigue existiendo en su ubicación original; `false` en caso contrario.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de vídeo, el siguiente código muestra "true" si el archivo se importó en la biblioteca sigue existiendo.

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

Véase también

`videoItem.sourceFileIsCurrent`, `videoItem.sourceFilePath`

videoItem.sourceFileIsCurrent

Disponibilidad

Flash CS4 Professional

Uso

```
videoItem.sourceFileIsCurrent
```

Descripción

Propiedad de sólo lectura; un valor booleano `true` si la fecha de modificación del archivo en el elemento de la biblioteca coincide con la fecha de modificación en disco del archivo importado; `false` en caso contrario.

Ejemplo

Si damos por hecho que el primer elemento de la biblioteca es un elemento de vídeo, el siguiente código muestra "true" si el archivo importado no se ha modificado en el disco desde su importación.

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

Véase también

`videoItem.fileLastModifiedDate`, `videoItem.sourceFilePath`

videoItem.sourceFilePath

Disponibilidad

Flash 8

Uso

```
videoItem.sourceFilePath
```

Descripción

Propiedad de sólo lectura; cadena expresada en forma de URI file:/// que especifica la ruta del elemento de vídeo.

Ejemplo

El siguiente ejemplo muestra el nombre y la ruta del archivo de origen de cualquier elemento de la biblioteca que sea de tipo `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);
    }
}
```

Véase también

`videoItem.sourceFileExists`

videoItem.videoType

Disponibilidad

Flash 8

Uso

`videoItem.videoType`

Descripción

Propiedad de sólo lectura; cadena que especifica el tipo de vídeo que representa el elemento. Los valores posibles son `"embedded video"`, `"linked video"` y `"video"`.

Ejemplo

El siguiente ejemplo muestra el nombre y el tipo de cualquier elemento de la biblioteca que sea de tipo `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);
}
}
```

Capítulo 52: Objeto XMLUI

Disponibilidad
Flash MX 2004

Descripción
Flash 8 admite cuadros de diálogo personalizados escritos en un subconjunto del lenguaje de interfaz de usuario XML (XUL). Hay varias funciones de Flash que pueden utilizar un cuadro de diálogo de interfaz de usuario XML (XMLUI) como, por ejemplo, comandos y comportamientos, para suministrar una interfaz de usuario para funciones que se crean utilizando la extensibilidad. El objeto XMLUI permite obtener y definir propiedades de un cuadro de diálogo XMLUI, así como aceptar o cancelar una. Los métodos de XMLUI pueden utilizarse en devoluciones de llamada, por ejemplo, controladores `oncommand` en los botones.

Puede escribir un archivo dialog.xml e invocarlo desde la API de JavaScript empleando el método `document.xmlPanel()`. Para recuperar un objeto que representa el cuadro de diálogo XMLUI actual, utilice `fl.xmlui`.

Resumen de métodos
Los métodos siguientes están disponibles para el objeto XMLUI:

Método	Descripción
`xmlui.accept()`	Cierra el cuadro de diálogo XMLUI actual con un estado "accept".
`xmlui.cancel()`	Cierra el cuadro de diálogo XMLUI actual con un estado "cancel".
`xmlui.get()`	Recupera el valor de la propiedad especificada del cuadro de diálogo XMLUI actual.
`xmlui.getControlItemElement()`	Devuelve el elemento de control actual para el control especificado.
`xmlui.getEnabled()`	Devuelve un valor booleano que especifica si el control está activo o inactivo (atenuado).
`xmlui.getVisible()`	Devuelve un valor booleano que especifica si el control está visible u oculto.
`xmlui.set()`	Modifica el valor de la propiedad especificada del cuadro de diálogo XMLUI actual.
`xmlui.setControlItemElement()`	Establece la etiqueta y el valor del elemento actual.
`xmlui.setControlItemElements()`	Establece los pares de etiqueta y valor del elemento actual.
`xmlui.setEnabled()`	Activa o desactiva (atenúa) un control.
`xmlui.setVisible()`	Muestra u oculta un control.

xmlui.accept()

Disponibilidad
Flash MX 2004

Uso

```
xmlui.accept()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; cierra el cuadro de diálogo XMLUI actual con un estado de aceptar, lo que equivale a que el usuario haga clic en el botón Aceptar.

Véase también

`fl.xmlui`, `document.xmlPanel()`, `xmlui.cancel()`

xmlui.cancel()

Disponibilidad

Flash MX 2004

Uso

```
xmlui.cancel()
```

Parámetros

Ninguno.

Valor devuelto

Ninguno.

Descripción

Método; cierra el cuadro de diálogo XMLUI actual con un estado de cancelar, lo que equivale a que el usuario haga clic en el botón Cancelar.

Véase también

`fl.xmlui`, `document.xmlPanel()`, `xmlui.accept()`

xmlui.get()

Disponibilidad

Flash MX 2004

Uso

```
xmlui.get(controlPropertyName)
```

Parámetros

`controlPropertyName` Una cadena que especifica el nombre de la propiedad XMLUI cuyo valor desea recuperar.

Valor devuelto

Una cadena que representa el valor de la propiedad especificada. En los casos en que se podría esperar un valor booleano de `true` o `false`, devuelve la cadena `"true"` o `"false"`.

Descripción

Método; recupera el valor de la propiedad especificada del cuadro de diálogo XMLUI actual.

Ejemplo

El siguiente ejemplo devuelve el valor de una propiedad denominada `URL`:

```
fl.xmlui.get("URL");
```

Véase también

`fl.xmlui`, `document.xmlPanel()`, `xmlui.getControlItemElement()`, `xmlui.set()`

xmlui.getControlItemElement()

Disponibilidad

Flash 8

Uso

```
xmlui.getControlItemElement(controlPropertyName)
```

Parámetros

`controlPropertyName` Una cadena que especifica la propiedad cuyo elemento de control desea recuperar.

Valor devuelto

Un objeto que representa el elemento de control actual para el control especificado por *controlPropertyName*.

Descripción

Método; devuelve la etiqueta y el valor de la línea seleccionada en un control ListBox o ComboBox para el control especificado por *controlPropertyName*.

Ejemplo

El siguiente ejemplo devuelve la etiqueta y el valor de la línea seleccionada actualmente para el control `myListBox`:

```
var elem = new Object();
elem = fl.xmlui.getControlItemElement("myListBox");
fl.trace("label = " + elem.label + " value = " + elem.value);
```

Véase también

`fl.xmlui`, `document.xmlPanel()`, `xmlui.get()`, `xmlui.setControlItemElement()`, `xmlui.setControlItemElements()`

xmlui.getEnabled()

Disponibilidad
Flash 8

Uso
`xmlui.getEnabled(controlID)`

Parámetros
`controlID` Una cadena que especifica el atributo de identificación del control cuyo estado desea recuperar.

Valor devuelto
Un valor booleano de `true` si el control es correcto, y de `false` en caso contrario.

Descripción
Método; devuelve un valor booleano que especifica si el control está activo o inactivo (atenuado).

Ejemplo
El ejemplo siguiente devuelve un valor que indica si está activado el control con el atributo de identificación `myListBox`:

```
var isEnabled = fl.xmlui.getEnabled("myListBox");
fl.trace(isEnabled);
```

Véase también
`fl.xmlui`, `document.xmlPanel()`, `xmlui.setEnabled()`

xmlui.getVisible()

Disponibilidad
Flash 8

Uso
`xmlui.getVisible(controlID)`

Parámetros
`controlID` Una cadena que especifica el atributo de identificación del control cuyo estado de visibilidad desea recuperar.

Valor devuelto
Un valor booleano de `true` si el control está visible o de `false` si no se ve (está oculto).

Descripción
Método; devuelve un valor booleano que especifica si el control está visible u oculto.

Ejemplo

El ejemplo siguiente devuelve un valor que indica si está visible el control con el atributo de identificación `myListBox`:

```
var isVisible = fl.xmlui.getVisible("myListBox");
fl.trace(isVisible);
```

Véase también

`xmlui.setVisible()`

xmlui.set()

Disponibilidad

Flash MX 2004

Uso

```
xmlui.set(controlPropertyName, value)
```

Parámetros

`controlPropertyName` Una cadena que especifica el nombre de la propiedad XMLUI que se va a modificar.

`value` Una cadena que especifica el valor para el que desea definir la propiedad XMLUI.

Valor devuelto

Ninguno.

Descripción

Método; modifica el valor de la propiedad especificada del cuadro de diálogo XMLUI actual.

Ejemplo

El ejemplo siguiente define como `www.adobe.com` el valor de una propiedad denominada URL:

```
fl.xmlui.set("URL", "www.adobe.com");
```

Véase también

`fl.xmlui`, `document.xmlPanel()`, `xmlui.get()`, `xmlui.setControlItemElement()`, `xmlui.setControlItemElements()`

xmlui.setControlItemElement()

Disponibilidad

Flash 8

Uso

```
xmlui.setControlItemElement(controlPropertyName, elementItem)
```

Parámetros

`controlPropertyName` Una cadena que especifica el elemento de control que se va a definir.

`elementItem` Un objeto JavaScript con una propiedad de cadena llamada `label` y una cadena opcional llamada `value`. Si no existe la propiedad `value`, se creará y se le asignará el mismo valor que `label`.

Valor devuelto

Ninguno.

Descripción

Método; define la etiqueta y el valor de la línea seleccionada actualmente en el control ListBox o ComboBox especificado por *controlPropertyName*.

Ejemplo

El siguiente ejemplo define la etiqueta y el valor para el elemento actual de la propiedad de control denominada `PhoneNumber`:

```
var elem = new Object();
elem.label = "Fax";
elem.value = "707-555-5555";
fl.xmlui.setControlItemElement("PhoneNumber",elem);
```

Véase también

`fl.xmlui`, `document.xmlPanel()`, `xmlui.getControlItemElement()`, `xmlui.set()`, `xmlui.setControlItemElements()`

xmlui.setControlItemElements()

Disponibilidad

Flash 8

Uso

```
xmlui.setControlItemElements(controlID, elementItemArray)
```

Parámetros

`controlID` Una cadena que especifica el atributo de identificación del control que desea definir.

`elementItemArray` Un conjunto de objetos JavaScript, donde cada objeto tiene una propiedad de cadena llamada `label` y una propiedad de cadena opcional llamada `value`. Si no existe la propiedad `value`, se creará y se le asignará el mismo valor que `label`.

Valor devuelto

Ninguno.

Descripción

Método; borra los valores del control ListBox o ComboBox especificado por *controlID* y reemplaza la lista o elementos de menú con los pares `label`,`value` especificados por *elementItemArray*.

Ejemplo

El ejemplo siguiente define la etiqueta y el valor de los elementos del control con el atributo de identificación `myControlID` con los pares `label`,`value` especificados:

```
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);
```

Véase también

`xmlui.getControlItemElement()`, `xmlui.set()`, `xmlui.setControlItemElement()`

xmlui.setEnabled()

Disponibilidad

Flash 8

Uso

```
xmlui.setEnabled(controlID, enable)
```

Parámetros

`controlID` Una cadena que especifica el atributo de identificación del control que desea activar o desactivar.

`enable` Un valor booleano `true` si desea activar el control; `false` si desea desactivarlo (atenuarlo).

Valor devuelto

Ninguno.

Descripción

Método; activa o desactiva (atenúa) un control.

Ejemplo

El siguiente ejemplo atenúa el control con el atributo de identificación `myControl`:

```
fl.xmlui.setEnabled("myControl", false);
```

Véase también

`xmlui.getEnabled()`

xmlui.setVisible()

Disponibilidad

Flash 8

Uso

```
xmlui.setVisible(controlID, visible)
```

Parámetros

`controlID` Una cadena que especifica el atributo de identificación del control que desea mostrar u ocultar.

`visible` Un valor booleano `true` si desea mostrar el control; `false` si desea ocultarlo.

Valor devuelto

Ninguno.

Descripción

Método; muestra u oculta un control.

Ejemplo

El siguiente ejemplo oculta el control con el atributo de identificación `myControl`:

```
fl.xmlui.setVisible("myControl", false);
```

Véase también

`xmlui.getVisible()`

Capítulo 53: Extensibilidad de nivel C

En este capítulo se describe el mecanismo de extensibilidad de nivel C. Este mecanismo le permite implementar los archivos de extensibilidad de Adobe Flash CS4 Professional mediante una combinación de código JavaScript y código C personalizado. No se han realizado modificaciones en el mecanismo con esta nueva versión de Flash.

Extensibilidad

Para implementar la extensibilidad, deberá definir las funciones mediante C, empaquetarlas en una biblioteca de vínculos dinámicos (DLL) o una biblioteca compartida, guardar la biblioteca en el directorio adecuado y, posteriormente, realizar una llamada a las funciones desde JavaScript mediante la API JavaScript de Adobe Flash.

Por ejemplo, es posible que desee definir una función que realice cálculos intensivos de forma más eficiente que el código JavaScript, con la consiguiente mejora del rendimiento, o bien que desee crear herramientas o efectos más avanzados.

El mecanismo de extensibilidad es un subconjunto de la API de Adobe Dreamweaver CS3. Si está familiarizado con dicha API, es posible que reconozca las funciones de la API del mecanismo de extensibilidad de nivel C. No obstante, esta API difiere de la API de Dreamweaver en los siguientes aspectos:

- Esta API no contiene todos los comandos de la API de Dreamweaver.
- Todas las declaraciones de tipo `wchar_t` y `char` de la API de Dreamweaver se implementan como declaraciones `unsigned short` en esta API para ofrecer compatibilidad con Unicode al pasar cadenas.
- La función `JSVal JS_BytesToValue()` de esta API no forma parte de la API de Dreamweaver.
- La ubicación en que deben almacenarse los archivos de DLL o de biblioteca compartida es diferente (consulte "Integración de funciones de C" en la página 536).

Integración de funciones de C

El mecanismo de extensibilidad de nivel C le permite implementar los archivos de extensibilidad de Flash mediante una combinación de código JavaScript y código C. El proceso de implementación de esta capacidad se resume en los siguientes pasos:

1. Defina funciones mediante el lenguaje C o C++.
2. Empaquételas en un archivo DLL (Windows) o una biblioteca compartida (Macintosh).
3. Guarde el archivo DLL o la biblioteca en la ubicación adecuada:
   - Windows Vista:

     unidad de arranque\Users\ *nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS3*idioma*\Configuration\External Libraries
   - Windows XP:

     unidad de arranque\Documents and Settings*nombre de usuario*\Local Settings\Application Data\Adobe\Flash CS3*idioma*\Configuration\External Libraries

- Mac OS X:

  Macintosh HD/Usuarios/*nombre de usuario*/Librería/Application Support/Adobe/Flash CS3/*idioma*/Configuration/External Libraries

4 Cree un archivo JSFL que llame a las funciones.

5 Ejecute el archivo JSFL desde el menú Comandos del entorno de edición de Flash.

Para más información, consulte "Implementación de DLL de muestra" en la página 540.

Extensibilidad de nivel C y el intérprete JavaScript

El código C de la DLL o la biblioteca compartida interactúa con la API JavaScript de Flash en tres momentos diferentes:

- Durante el inicio, para registrar las funciones de la biblioteca
- Cuando se llama a la función C, para desempaquetar los argumentos que se pasan de JavaScript a C
- Antes de que la función C devuelva el resultado, para empaquetar el valor devuelto

Para realizar estas tareas, el intérprete define diversos tipos de datos y expone una API. Las definiciones de los tipos de datos y las funciones enumeradas en esta sección aparecen en el archivo mm_jsapi.h. Para que la biblioteca funcione correctamente, debe incluir el archivo mm_jsapi.h en la parte superior de cada archivo de la biblioteca con la siguiente línea de código:

```
#include "mm_jsapi.h"
```

Al incluir el archivo mm_jsapi.h, se incluye el archivo mm_jsapi_environment.h, que define la estructura MM_Environment.

Para obtener una copia del archivo mm_jsapi.h, extráigalo del archivo ZIP o SIT de muestra (consulte "Implementación de DLL de muestra" en la página 540) o copie el siguiente código en un archivo al que deberá asignar el nombre de 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_ */
```

Implementación de DLL de muestra

En esta sección se describe cómo crear una implementación DLL sencilla. Si desea ver cómo funciona el proceso sin tener que crear usted mismo la DLL, puede instalar los archivos de DLL de muestra que se proporcionan en el archivo Samples.zip; los archivos se encuentran en la carpeta ExtendingFlash/dllSampleComputeSum. (Para obtener información sobre la descarga del archivo Samples.zip, consulte "Implementaciones de muestra" en la página 13.) Extraiga los archivos de muestra del archivo dllSampleComputeSum.dmg o dllSampleComputeSum.zip y proceda del siguiente modo:

- Guarde el archivo Sample.jsfl en el directorio Configuration/Commands (consulte "Almacenamiento de archivos JSFL" en la página 2).
- Guarde el archivo Sample.dll en el directorio Configuration/External Libraries (consulte "Integración de funciones de C" en la página 536).

- En el entorno de edición de Flash, seleccione Comandos > Muestra. La sentencia trace del archivo JSFL envía el resultado de la función definida en Sample.dll al panel Salida.

En el resto de esta sección se aborda el desarrollo del ejemplo. En este caso, la DLL sólo contiene una función que añade dos números. El código C se muestra en el siguiente ejemplo:

```
// 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;
}
```

Tras escribir este código, cree el archivo de la DLL o la biblioteca compartida en el directorio Configuration/External Libraries adecuado (consulte “Integración de funciones de C” en la página 536). A continuación cree un archivo JSFL con el siguiente código y guárdelo en el directorio Configuration/Commands (consulte “Almacenamiento de archivos JSFL” en la página 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 );
```

Para ejecutar la función definida en la DLL, seleccione Comandos > Muestra en el entorno de edición de Flash.

Tipos de datos

El intérprete JavaScript define los tipos de datos descritos en esta sección.

typedef struct JSContext JSContext

Se pasa un puntero a este tipo de datos opaco a la función de nivel C. Algunas funciones de la API aceptan este puntero como uno de sus argumentos.

typedef struct JSObject JSObject

Se pasa un puntero a este tipo de datos opaco a la función de nivel C. Este tipo de datos representa un objeto, que podría ser un objeto conjunto o algún otro tipo de objeto.

typedef struct jsval jsval

Una estructura de datos opaca que puede contener un número entero o un puntero a un elemento flotante, una cadena o un objeto. Algunas funciones de la API pueden leer los valores de los argumentos de la función mediante la lectura del contenido de una estructura `jsval`, mientras que otras pueden utilizarse para escribir el valor devuelto por la función mediante la escritura de una estructura `jsval`.

typedef enum { JS_FALSE = 0, JS_TRUE = 1 } JSBool

Tipo de datos sencillo que almacena un valor booleano.

La API de nivel C

La API de extensibilidad de nivel C está forma por la firma de la función `JSBool (*JSNative)` y por las siguientes funciones:

- `JSBool JS_DefineFunction()`
- `unsigned short *JS_ValueToString()`
- `JSBool JS_ValueToInteger()`
- `JSBool JS_ValueToDouble()`
- `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)

Descripción

Método; describe las implementaciones de nivel C de las funciones JavaScript en las siguientes situaciones:

- El puntero *cx* es un puntero a una estructura `JSContext` opaca que debe pasarse a algunas funciones de la API JavaScript. Esta variable incluye el contexto de ejecución del intérprete.
- El puntero *obj* es un puntero al objeto en cuyo contexto se ejecuta el script. Durante la ejecución del script, la palabra clave `this` es igual a este objeto.
- El entero *argc* es el número de argumentos que se pasan a la función.
- El puntero *argv* es un puntero de un conjunto de estructuras `jsval`. El conjunto tiene una longitud de `argc` elementos.
- El puntero *rval* es un puntero a una única estructura `jsval`. El valor devuelto por la función debe escribirse en `*rval`.

La función devuelve `JS_TRUE` si se ejecuta correctamente; en caso contrario, devuelve `JS_FALSE`. Si la función devuelve `JS_FALSE`, el script actual dejará de ejecutarse y aparecerá un mensaje de error.

JSBool JS_DefineFunction()

Uso

```
JSBool JS_DefineFunction(unsigned short *name, JSNative call, unsigned int nargs)
```

Descripción

Método; registra una función de nivel C con el intérprete JavaScript en Flash. Una vez que la función `JS_DefineFunction()` registre la función de nivel C que usted ha especificado en el argumento *call*, podrá invocarla en un script JavaScript haciendo referencia a ella mediante el nombre que ha especificado en el argumento *name*. El argumento *name* distingue entre mayúsculas y minúsculas.

Típicamente, se llama a esta función desde la función `MM_Init()`, a la que Flash llama durante su inicio.

Argumentos

`unsigned short *`*name*, `JSNative`*call*, `unsigned int` *nargs*

- El argumento *name* es el nombre de la función tal y como se expone a JavaScript.
- El argumento *call* es un puntero a una función de nivel C. La función debe devolver un `JSBool`, que indica si es correcta o incorrecta.
- El argumento *nargs* es el número de argumentos que la función espera recibir.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

unsigned short *JS_ValueToString()

Uso

```
unsigned short *JS_ValueToString(JSContext *cx, jsval v, unsigned int *pLength)
```

Descripción

Método; extrae un argumento de la función de una estructura `jsval`, lo convierte en una cadena si es posible y devuelve el valor convertido al originador de la llamada.

Nota: *no modifique el puntero de búfer devuelto, ya que podría dañar las estructuras de datos del intérprete JavaScript. Para cambiar la cadena, debe copiar los caracteres en otro búfer y crear una nueva cadena JavaScript.*

Argumentos

`JSContext` **cx*, `jsval` *v*, `unsigned int` **pLength*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *v* es la estructura `jsval` de la que se extrae la cadena.
- El argumento *pLength* es un puntero a un entero sin signo. Esta función establece `*plength` con un valor igual a la longitud de la cadena en bytes.

Valor devuelto

Un puntero que señala a una cadena con terminación nula si es correcta o a un valor `null` si es incorrecta. La rutina de llamada no debe liberar esta cadena al finalizar.

JSBool JS_ValueToInteger()

Uso

```
JSBool JS_ValueToInteger(JSContext *cx, jsval v, long *lp);
```

Descripción

Método; extrae un argumento de la función de una estructura `jsval`, lo convierte en un entero (si es posible) y devuelve el valor convertido al originador de la llamada.

Argumentos

`JSContext` **cx*, `jsval` *v*, `long` **lp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *v* es la estructura `jsval` de la que se extrae el entero.
- El argumento *lp* es un puntero a un entero de 4 bytes. Esta función almacena el valor convertido en `*lp`.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_ValueToDouble()

Uso

```
JSBool JS_ValueToDouble(JSContext *cx, jsval v, double *dp);
```

Descripción

Método; extrae un argumento de la función de una estructura `jsval`, lo convierte en el doble (si es posible) y devuelve el valor convertido al originador de la llamada.

Argumentos

`JSContext` **cx*, `jsval` *v*, `double` **dp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *v* es la estructura `jsval` de la que se extrae el doble.
- El argumento *dp* es un puntero a un doble de 8 bytes. Esta función almacena el valor convertido en `*dp`.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_ValueToBoolean()

Uso

```
JSBool JS_ValueToBoolean(JSContext *cx, jsval v, JSBool *bp);
```

Descripción

Método; extrae un argumento de la función de una estructura `jsval`, lo convierte en un valor booleano (si es posible) y devuelve el valor convertido al originador de la llamada.

Argumentos

`JSContext` **cx*, `jsval` *v*, `JSBool` **bp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *v* es la estructura `jsval` de la que se extrae el valor booleano.
- El argumento *bp* es un puntero a un valor booleano `JSBool`. Esta función almacena el valor convertido en `*bp`.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_ValueToObject()

Uso

```
JSBool JS_ValueToObject(JSContext *cx, jsval v, JSObject **op);
```

Descripción

Método; extrae un argumento de la función de una estructura `jsval`, lo convierte en un objeto (si es posible) y devuelve el valor convertido al originador de la llamada. Si el objeto es una conjunto, utilice `JS_GetArrayLength()` y `JS_GetElement()` para leer su contenido.

Argumentos

`JSContext` **cx*, `jsval` *v*, `JSObject` ***op*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.

- El argumento *v* es la estructura `jsval` de la que se extrae el objeto.
- El argumento *op* es un puntero a un puntero `JSObject`. Esta función almacena el valor convertido en `*op`.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_StringToValue()

Uso

```
JSBool JS_StringToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp);
```

Descripción

Método; almacena un valor devuelto de cadena en una estructura `jsval`. Asigna un nuevo objeto de cadena JavaScript.

Argumentos

`JSContext` **cx*, `unsigned short` **bytes*, `size_t`*sz*, `jsval` **vp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *bytes* es la cadena que debe almacenarse en la estructura `jsval`. Los datos de cadena se copian, por lo que el originador de la llamada debe liberar la cadena cuando no sea necesaria. Si no se especifica el tamaño de la cadena (consulte el argumento *sz*), la cadena deberá tener terminación nula.
- El argumento *sz* es el tamaño de la cadena en bytes. Si *sz* es 0, la longitud de la cadena con terminación nula se calculará automáticamente.
- El argumento *vp* es un puntero a la estructura `jsval` en la que se debe copiar el contenido de la cadena.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_DoubleToValue()

Uso

```
JSBool JS_DoubleToValue(JSContext *cx, double dv, jsval *vp);
```

Descripción

Método; almacena un valor devuelto de número de coma flotante en una estructura `jsval`.

Argumentos

`JSContext` **cx*, `double` *dv*, `jsval` **vp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *dv* es un número de coma flotante de 8 bytes.
- El argumento *vp* es un puntero a la estructura `jsval` en la que debe copiarse el contenido del doble.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSVal JS_BooleanToValue()

Uso

```
jsval JS_BooleanToValue(JSBool bv);
```

Descripción

Método; almacena un valor devuelto booleano en una estructura `jsval`.

Argumentos

`JSBool` *bv*

- El argumento *bv* es un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

Valor devuelto

Una estructura `JSVal` que contiene el valor booleano que pasa a la función en forma de argumento.

JSVal JS_BytesToValue()

Uso

```
JSBool JS_BytesToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp);
```

Descripción

Método; convierte los bytes en un valor JavaScript.

Argumentos

`JSContext` **cx*, `unsignedshort`**bytes*, `uint`*sz*, `jsval` **vp*

- El argumento *cx* es el contexto JavaScript.
- El argumento *bytes* es la cadena de bytes que debe convertirse en un objeto JavaScript.
- El argumento *sz* es el número de bytes que deben convertirse.
- El argumento *vp* es el valor JavaScript.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSVal JS_IntegerToValue()

Uso

```
jsval JS_IntegerToValue(long lv);
```

Descripción

Método; convierte un valor de entero largo en estructura `JSVal`.

Argumentos

lv

El argumento *lv* es el valor de entero largo que desea convertir en estructura `jsval`.

Valor devuelto
Una estructura `JSVal` que contiene el entero pasado a la función en forma de argumento.

JSVal JS_ObjectToValue()

Uso
```
jsval JS_ObjectToValue(JSObject *obj);
```

Descripción
Método; almacena un valor devuelto de objeto en `JSVal`. Utilice `JS_NewArrayObject()` para crear un objeto de conjunto; utilice `JS_SetElement()` para definir su contenido.

Argumentos
`JSObject` **obj*

El argumento *obj* es un puntero al objeto `JSObject` que desea convertir en estructura `JSVal`.

Valor devuelto
Una estructura `JSVal` que contiene el objeto que pasó a la función en forma de argumento.

unsigned short *JS_ObjectType()

Uso
```
unsigned short *JS_ObjectType(JSObject *obj);
```

Descripción
Método; dada una referencia de objeto, devuelve el nombre de la clase del objeto. Por ejemplo, si el objeto es un objeto DOM, la función devuelve "`Document`". Si el objeto es un nodo del documento, la función devuelve "`Element`". En el caso de un objeto de conjunto, la función devuelve "`Array`".

Nota: *no modifique el puntero de búfer devuelto, ya que podría dañar las estructuras de datos del intérprete JavaScript.*

Argumentos
`JSObject` **obj*

Típicamente, este argumento se pasa y se convierte mediante la función `JS_ValueToObject()`.

Valor devuelto
Un puntero a una cadena con terminación nula. El originador de la llamada no debe liberar esta cadena al finalizar.

JSObject *JS_NewArrayObject()

Uso
```
JSObject *JS_NewArrayObject(JSContext *cx, unsigned int length [, jsval *v])
```

Descripción
Método; crea un nuevo objeto que contiene un conjunto de `JSVals`.

Argumentos

`JSContext` **cx*, `unsigned int` *length*, `jsval` **v*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *length* es el número de elementos que puede contener el conjunto.
- El argumento *v* es un puntero opcional a la `jsvals` que debe almacenarse en el conjunto. Si el valor devuelto no es `null`, *v* es un conjunto que contiene *length* elementos. Si el valor devuelto es `null`, el contenido inicial del objeto de conjunto es indefinido y puede establecerse mediante la función `JS_SetElement()`.

Valor devuelto

Un puntero a un nuevo objeto de conjunto o el valor `null` en caso de error.

long JS_GetArrayLength()

Uso

```
long JS_GetArrayLength(JSContext *cx, JSObject *obj)
```

Descripción

Método; dado un puntero a un objeto de conjunto, obtiene el número de elementos del conjunto.

Argumentos

`JSContext` **cx*, `JSObject`**obj*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *obj* es un puntero a un objeto de conjunto.

Valor devuelto

El número de elementos del conjunto o -1 en caso de error.

JSBool JS_GetElement()

Uso

```
JSBool JS_GetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp)
```

Descripción

Método; lee un elemento individual de un objeto de conjunto.

Argumentos

`JSContext` **cx*, `JSObject` **obj*, `jsint` *idx*, `jsval` **vp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *obj* es un puntero a un objeto de conjunto.
- El argumento *idx* es un índice de entero del conjunto. El primer elemento es el índice `0` y el último, el índice (`length 1-`).

- El argumento *vp* es un puntero a una `jsval` en la que debe copiarse el contenido de la estructura `jsval` del conjunto.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_SetElement()

Uso

```
JSBool JS_SetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp)
```

Descripción

Método; escribe un elemento individual de un objeto de conjunto.

Argumentos

`JSContext` **cx* , `JSObject` **obj* , `jsint` *idx* , `jsval` **vp*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *obj* es un puntero a un objeto de conjunto.
- El argumento *idx* es un índice de entero del conjunto. El primer elemento es el índice `0` y el último, el índice (`length 1-`).
- El argumento *vp* es un puntero a una estructura `jsval` cuyo contenido debe copiarse en la `jsval` del conjunto.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.

JSBool JS_ExecuteScript()

Uso

```
JS_ExecuteScript (JSContext *cx, JSObject *obj, unsigned short *script, unsigned int sz, jsval
*rval)
```

Descripción

Método; compila y ejecuta una cadena JavaScript. Si el script genera un valor de devolución, lo devuelve en `*rval`.

Argumentos

`JSContext` **cx*, `JSObject` **obj*, `unsigned short` **script*, `unsigned int`*sz*, `jsval` **rval*

- El argumento *cx* es el puntero `JSContext` opaco que pasa a la función JavaScript.
- El argumento *obj* es un puntero al objeto en cuyo contexto se ejecuta el script. Durante la ejecución del script, la palabra clave `this` es igual a este objeto. Normalmente se trata del puntero `JSObject` que pasa a la función JavaScript.
- El argumento *script* es una cadena que contiene código JavaScript. Si no se especifica el tamaño de la cadena (consulte el argumento *sz*), la cadena deberá tener terminación nula.

- El argumento *sz* es el tamaño de la cadena en bytes. Si *sz* es 0, la longitud de la cadena con terminación nula se calculará automáticamente.
- El argumento *rval* es un puntero a una única estructura `jsval`. El valor devuelto por la función se almacena en `*rval`.

Valor devuelto

Un valor booleano: `JS_TRUE` indica que es correcto; `JS_FALSE` indica que es incorrecto.