# ADOBE® AIR®-toepassingen ontwikkelen

## Juridische kennisgeving

Zie http://help.adobe.com/nl_NL/legalnotices/index.html voor de juridische kennisgeving.

**Laatst bijgewerkt 31/5/2017**

# Inhoud

# Hoofdstuk 1: Informatie over Adobe AIR

Adobe® AIR® is een besturingssysteemoverschrijdend runtime-programma met meerdere schermen waarmee u uw bestaande webontwikkelingsvaardigheden kunt optimaliseren om RIA's (Rich Internet Applications) te maken en te implementeren voor het bureaublad en mobiele apparaten. Bureaublad-, televisie- en mobiele AIR-toepassingen kunnen worden gebouwd met ActionScript 3.0 met behulp van Adobe® Flex en Adobe® Flash® (op basis van SWF). AIR-bureaubladtoepassingen kunnen ook worden gebouwd met HTML, JavaScript® en Ajax (op basis van HTML).

Meer informatie over hoe u aan de slag gaat met en werkt met Adobe AIR vindt u op de Adobe AIR Developer Connection (http://www.adobe.com/devnet/air/).

Met AIR werkt u in een bekende programmeeromgeving. U kunt hulpprogramma's gebruiken en benaderingen toepassen die voor u het meest comfortabel zijn. Door ondersteuning voor Flash, Flex, HTML, JavaScript, en Ajax maakt u optimale oplossingen die voldoen aan uw eisen.

U kunt bijvoorbeeld toepassingen ontwikkelen met behulp van een van de volgende technologieën of een combinatie daarvan:

- Flash / Flex / ActionScript
- HTML / JavaScript / CSS / Ajax

Gebruikers werken op dezelfde manier met AIR-toepassingen als met native toepassingen. De runtime wordt éénmaal op de computer of het apparaat van de gebruiker geïnstalleerd, en daarna worden AIR-toepassingen geïnstalleerd en uitgevoerd net als elke andere bureaubladtoepassing. (Op iOS wordt geen afzonderlijke AIR-runtime geïnstalleerd; elke iOS AIR-toepassing is een zelfstandige toepassing.)

De runtime biedt een consistent platform voor meerdere besturingssystemen en een kader voor de implementatie van toepassingen waarbij de functionaliteit op verschillende desktops gegarandeerd constant is, zodat deze niet voor elke browser hoeft te worden getest. U richt zich bij de ontwikkeling op de runtime, niet op een specifiek besturingssysteem. Dit heeft de volgende voordelen:

- Toepassingen die zijn ontwikkeld voor AIR kunnen in meerdere besturingssystemen worden uitgevoerd zonder dat u extra werk hoeft te verrichten. De runtime garandeert een consistente en voorspelbare presentatie en interacties voor alle besturingssystemen die worden ondersteund door AIR.
- Toepassingen maken gaat ook sneller. U maakt immers gebruik van bestaande webtechnologieën en ontwerppatronen. Bovendien kunt u webtoepassingen uitbreiden voor het bureaublad zonder dat u de traditionele technologieën voor bureaubladontwikkeling of de complexe native code moet leren.
- Bij de ontwikkeling van toepassingen hoeft u geen gebruik te maken van lagere niveautalen zoals C of C++. U hoeft niet de complexe laag-niveau-API's te beheren die specifiek zijn voor ieder besturingssysteem.

Bij de ontwikkeling van toepassingen voor AIR kunt u gebruikmaken van een zeer uitgebreide set bestaande frameworks en API's:

- API's die specifiek zijn voor AIR en die worden geleverd door de runtime en het AIR-framework
- ActionScript-API's die worden gebruikt in SWF-bestanden en Flex-frameworks (evenals andere ActionScript-gebaseerde bibliotheken en frameworks)
- HTML, CSS en JavaScript
- De meeste Ajax-frameworks

- Native extensies voor Adobe AIR die ActionScript API's verschaffen waarmee u toegang krijgt tot platformspecifieke functies die in de native code zijn geprogrammeerd. Native extensies kunnen ook toegang verschaffen tot verouderde native code en tot native code die tot betere prestaties leidt.

AIR zorgt voor een enorme verandering in de manier waarop toepassingen kunnen worden gemaakt, geïmplementeerd en ervaren. U krijgt meer creatieve controle en kunt uw Flash-, Flex-, HTML- en Ajax-gebaseerde toepassingen uitbreiden naar het bureaublad, mobiele apparaten en televisies.

Voor informatie over de inhoud van updates van AIR raadpleegt u de Adobe AIR Release-notities (http://www.adobe.com/go/learn_air_relnotes_nl).

# Hoofdstuk 2: Adobe AIR installeren

Met de Adobe® AIR®-runtime kunt u AIR-toepassingen uitvoeren. U kunt de runtime op de volgende manieren installeren:

- U kunt de runtime afzonderlijk installeren (zonder ook een AIR-toepassing te installeren).
- U kunt een AIR-toepassing voor de eerste keer installeren via een installatielabel op een webpagina (u wordt gevraagd om ook de runtime te installeren).
- U kunt een aangepast installatieprogramma maken dat zowel uw toepassing als de runtime installeert. U dient goedkeuring te vragen aan Adobe als u de AIR-runtime op deze manier wilt distribueren. U kunt goedkeuring aanvragen op de Adobe-pagina voor runtimelicenties. Adobe biedt geen tools voor het samenstellen van een dergelijk installatieprogramma. Er zijn echter veel installatietoolkits beschikbaar van andere bedrijven.
- U kunt een AIR-toepassing installeren waarin AIR als een captive runtime is gebundeld. Een captive runtime wordt alleen gebruikt door de toepassing die de bundel maakt. Het wordt niet gebruikt voor het uitvoeren van andere AIR-toepassingen. Het bundelen van de runtime is een optie op Mac en Windows. Op iOS beschikken alle toepassingen over een gebundelde runtime. Vanaf AIR 3.7 beschikken alle Android-toepassingen standaard over een gebundelde runtime (hoewel u de optie hebt om een afzonderlijke runtime te gebruiken).
- U kunt een AIR-ontwikkelomgeving instellen zoals de SDK van AIR, Adobe® Flash® Builder™ of de SDK van Adobe Flex® (deze bevat de AIR-ontwikkelprogramma's voor de opdrachtregel). De runtime die is opgenomen in de SDK wordt alleen gebruikt tijdens foutopsporing bij toepassingen. De runtime wordt niet gebruikt voor het uitvoeren van geïnstalleerde AIR-toepassingen.

Hier worden de systeemvereisten beschreven om AIR en AIR-toepassingen te installeren en uit te voeren: Adobe AIR: System requirements (http://www.adobe.com/nl/products/air/systemreqs/).

Tijdens het installeren, bijwerken of verwijderen van zowel AIR-toepassingen als de runtime van AIR zelf worden logbestanden gemaakt. U kunt deze logboeken raadplegen om de oorzaak van eventuele installatieproblemen te achterhalen. Zie Installatielogboeken.

## Adobe AIR installeren

Voor het installeren of bijwerken van de runtime moet de gebruiker over beheerdersrechten voor de computer beschikken.

**De runtime installeren op een Windows-computer**

1. Download het runtime-installatiebestand van http://get.adobe.com/air.
2. Dubbelklik op het runtime-installatiebestand.
3. Volg in het installatievenster de aanwijzingen om de installatie te voltooien.

**De runtime installeren op een Mac-computer**

1. Download het runtime-installatiebestand van http://get.adobe.com/air.
2. Dubbelklik op het runtime-installatiebestand.
3. Volg in het installatievenster de aanwijzingen om de installatie te voltooien.

4 Als het installatieprogramma een verificatievenster weergeeft, geeft u uw gebruikersnaam en wachtwoord voor Mac OS op.

### De runtime installeren op een Linux-computer

***Opmerking:*** *momenteel worden AIR 2.7 en later niet ondersteund op Linux. Voor AIR-toepassingen die op Linux worden geïmplementeerd, blijft u gebruikmaken van de AIR 2.6-SDK.*

*Het binaire installatieprogramma gebruiken:*

1 Zoek het binaire bestand voor de installatie op http://kb2.adobe.com/nl/cps/853/cpsid_85304.html en download dit.

2 Stel de bestandsmachtigingen in zodat de installatietoepassing kan worden uitgevoerd. Vanaf de opdrachtregel kunt u de bestandsmachtigingen instellen met:

```
chmod +x AdobeAIRInstaller.bin
```

Met bepaalde versies van Linux kunt u de bestandsmachtigingen instellen via het menu Eigenschappen dat u hebt geopend via een contextmenu.

3 Voer het installatieprogramma uit vanaf de opdrachtregel of door te dubbelklikken op het runtime-installatiebestand.

4 Volg in het installatievenster de aanwijzingen om de installatie te voltooien.

Adobe AIR wordt geïnstalleerd als een native pakket. Met andere woorden: als rpm op een rpm-distributie en als deb op een Debian-distributie. AIR biedt momenteel geen ondersteuning voor andere pakketindelingen.

*De pakketinstallatieprogramma's gebruiken:*

1 Zoek het AIR-pakketbestand op http://kb2.adobe.com/nl/cps/853/cpsid_85304.html. Download het rpm- of Debian-pakket, afhankelijk van de pakketindeling die door uw systeem wordt ondersteund.

2 Indien nodig, dubbelklikt u op het AIR-pakketbestand om het pakket te installeren.

U kunt de installatie ook uitvoeren vanaf de opdrachtregel:

**a** Bij een Debian-systeem:

```
sudo dpkg -i <path to the package>/adobeair-2.0.0.xxxxx.deb
```

**b** Bij een rpm-systeem:

```
sudo rpm -i <path to the package>/adobeair-2.0.0-xxxxx.i386.rpm
```

Als u een bestaande versie bijwerkt (AIR 1.5.3 of hoger), typt u het volgende:

```
sudo rpm -U <path to the package>/adobeair-2.0.0-xxxxx.i386.rpm
```

Als u AIR- en AIR 2-toepassingen wilt installeren, moet u beschikken over beheerdersprivileges op uw computer.

Adobe AIR wordt geïnstalleerd op de volgende locatie: /opt/Adobe AIR/Versions/1.0

AIR registreert het mime-type "application/vnd.adobe.air-application-installer-package+zip". Dit betekent dat .air-bestanden van dit mime-type zijn en daarom worden geregistreerd met de AIR-runtime.

### Installeer de runtime op een Android-apparaat

U kunt de laatste versie van de AIR-runtime van de Android Market installeren.

U kunt ontwikkelversies van de AIR-runtime installeren vanaf een koppeling op een webpagina of met behulp van de ADT-opdracht `-installRuntime`. Er kan slechts één versie van de AIR-runtime tegelijk worden geïnstalleerd. U kunt niet zowel een versie als een ontwikkelversie versie geïnstalleerd hebben.

Zie “ADT-opdracht installRuntime” op pagina 185 voor meer informatie.

**Installeer de runtime op een iOS-apparaat**
De vereiste AIR-runtimecode is opgenomen in elke toepassing die wordt gemaakt voor iPhone-, iTouch- en iPad-apparaten. U installeert geen afzonderlijke runtime-component.

### Meer Help-onderwerpen
“AIR for iOS” op pagina 74

# Adobe AIR verwijderen

Wanneer u de runtime hebt geïnstalleerd, kunt u deze als volgt verwijderen.

**De runtime verwijderen op een Windows-computer**
1. Klik in het menu Start van Windows op Instellingen > Configuratiescherm.
2. Open het configuratiescherm Programma's, Programma's en onderdelen, of Programma's installeren of verwijderen (afhankelijk van welke versie van Windows u uitvoert).
3. Kies “Adobe AIR” om de runtime te verwijderen.
4. Klik op de knop Wijzigen/Verwijderen.

**De runtime verwijderen op een Mac-computer**
- Dubbelklik op de “Adobe AIR Uninstaller” in de map /Applications/Utilities.

**De runtime verwijderen op een Linux-computer**
Ga als volgt te werk:

- Selecteer de opdracht om Adobe AIR te verwijderen in het menu Toepassingen.
- Voer het AIR-installatieprogramma binair uit met de optie `-uninstall`
- Verwijder de AIR-pakketten (`adobeair` en `adobecerts`) via het pakketbeheer.

**Verwijder de runtime uit een Android-apparaat**
1. Open de toepassing Settings op het apparaat.
2. Tik op het element Adobe AIR onder Toepassingen > Toepassingen beheren.
3. Tik op de knop Uninstall (Verwijderen)

U kunt ook de ADT-opdracht `-uninstallRuntime` gebruiken. Zie “ADT-opdracht uninstallRuntime” op pagina 186 voor meer informatie.

**Een gebundelde runtime verwijderen**
Als u een gebundelde captive runtime wilt verwijderen, verwijdert u de toepassing waarmee de runtime is geïnstalleerd. Let wel dat captive runtimes alleen worden gebruikt voor het uitvoeren van de installerende toepassing.

# AIR-voorbeeldtoepassingen installeren en uitvoeren

Als een gebruiker een AIR-toepassing wil installeren of bijwerken, moet de gebruiker beschikken over beheerdersprivileges voor de computer.

Er zijn een paar voorbeeldtoepassingen beschikbaar waarin functies van AIR worden gedemonstreerd. U kunt deze als volgt openen en installeren:

1. Download de AIR-voorbeeldtoepassingen en voer ze uit. Zowel de gecompileerde toepassingen als de broncode zijn beschikbaar.
2. Als u een voorbeeldtoepassing wilt downloaden en uitvoeren, klikt u op de knop Install Now van de voorbeeldtoepassing. U wordt gevraagd de toepassing te installeren en uit te voeren.
3. Als u voorbeeldtoepassingen wilt downloaden en later wilt uitvoeren, selecteert u de downloadkoppelingen. U kunt AIR-toepassingen op elk gewenst moment uitvoeren:
   - In Windows dubbelklikt u op het toepassingspictogram op het bureaublad of selecteert u de toepassing in het menu Start van Windows.
   - In Mac OS dubbelklikt u op het toepassingspictogram dat standaard is geïnstalleerd in de map Applications (Programma's) van de gebruikersdirectory (bijvoorbeeld in Macintosh HD/Users/Gebruiker/Applications/).

***Opmerking:*** *updates van de AIR-releasenotes vindt u in http://www.adobe.com/go/learn_air_relnotes_nl.*

# Updates van Adobe AIR

Adobe AIR wordt regelmatig bijgewerkt met nieuwe functies of oplossingen voor kleine problemen. Met de functie voor automatische meldingen en updates kan Adobe gebruikers automatisch laten weten wanneer een bijgewerkte versie van Adobe AIR beschikbaar is.

Updates voor Adobe AIR zorgen ervoor dat Adobe AIR correct werkt en bevatten vaak belangrijke beveiligingswijzigingen. Adobe raadt gebruikers aan om altijd de laatste versie van Adobe AIR te installeren zodra deze beschikbaar is, vooral als er sprake is van een beveiligingsupdate.

Wanneer een AIR-toepassing wordt gestart, controleert de runtime standaard of er een update beschikbaar is. Deze controle wordt uitgevoerd als de vorige controle meer dan twee weken geleden is. Als er een update beschikbaar is, wordt deze op de achtergrond gedownload.

Met de toepassing AIR SettingsManager kunt u deze automatische updates uitschakelen. De toepassing AIR SettingsManager kan worden gedownload op http://airdownload.adobe.com/air/applications/SettingsManager/SettingsManager.air.

Tijdens de normale installatieprocedure van Adobe AIR wordt verbinding gemaakt met http://airinstall.adobe.com om basisgegevens over de installatieomgeving, zoals besturingssysteem en taal, te verzenden. Deze gegevens worden slechts één keer per installatie verzonden en worden door Adobe gebruikt om te bevestigen dat de installatie is gelukt. Er worden geen gegevens verzameld of verzonden waarmee uw identiteit kan worden achterhaald.

### Captive runtimes gebruiken

Als u samen met uw toepassing een captive-runtimebundel distribueert, dient u te weten dat de captive runtime niet automatisch wordt bijgewerkt. Met oog op de beveiliging van de gebruikers, dient u de door Adobe gepubliceerde updates in de gaten te houden en uw toepassing bij te werken met de nieuwe runtimeversie wanneer een relevante beveiligingsupdate wordt gepubliceerd.

# Hoofdstuk 3: Werken met de API's van AIR

Adobe® AIR® bevat functionaliteit die niet beschikbaar is voor SWF-inhoud die in Adobe® Flash® Player wordt uitgevoerd.

**ActionScript 3.0-ontwikkelaars**
De API's van Adobe AIR worden in de volgende twee boeken beschreven:

- ActionScript 3.0-ontwikkelaarsgids
- Naslaggids voor ActionScript 3.0 voor het Adobe Flash-platform

**HTML-ontwikkelaars**
Als u AIR-toepassingen maakt die zijn gebaseerd op HTML, worden de API's die via het bestand AIRAliases.js beschikbaar zijn in JavaScript (zie Accessing AIR API classes from JavaScript), in de volgende twee boeken beschreven:

- HTML Developer's Guide for Adobe AIR
- Adobe AIR API-naslaggids voor HTML-ontwikkelaars

## ActionScript 3.0-klassen die specifiek van toepassing zijn op AIR

De volgende tabel bevat runtime-klassen die specifiek zijn voor Adobe AIR. Deze klassen zijn niet beschikbaar voor SWF-inhoud die wordt uitgevoerd in Adobe® Flash® Player in de browser.

**HTML-ontwikkelaars**
De klassen die via het bestand AIRAliases.js beschikbaar zijn in JavaScript staan in de Adobe AIR API-naslaggids voor HTML-ontwikkelaars .

| Klasse | ActionScript 3.0-pakket | Toegevoegd in AIR-versie |
|---|---|---|
| ARecord | flash.net.dns | 2.0 |
| AAAARecord | flash.net.dns | 2.0 |
| ApplicationUpdater | air.update | 1.5 |
| ApplicationUpdaterUI | air.update | 1.5 |
| AudioPlaybackMode | flash.media | 3.0 |
| AutoCapitalize | flash.text | 3.0 |
| BrowserInvokeEvent | flash.events | 1.0 |
| CameraPosition | flash.media | 3.0 |

| Klasse | ActionScript 3.0-pakket | Toegevoegd in AIR-versie |
|---|---|---|
| CameraRoll | flash.media | 2.0 |
| CameraRollBrowseOptions | flash.media | 3.0 |
| CameraUI | flash.media | 2.5 |
| CertificateStatus | flash.security | 2.0 |
| CompressionAlgorithm | flash.utils | 1.0 |
| DatagramSocket | flash.net | 2.0 |
| DatagramSocketDataEvent | flash.events | 2.0 |
| DNSResolver | flash.net.dns | 2.0 |
| DNSResolverEvent | flash.events | 2.0 |
| DockIcon | flash.desktop | 1.0 |
| DownloadErrorEvent | air.update.events | 1.5 |
| DRMAuthenticateEvent | flash.events | 1.0 |
| DRMDeviceGroup | flash.net.drm | 3.0 |
| DRMDeviceGroupErrorEvent | flash.net.drm | 3.0 |
| DRMDeviceGroupEvent | flash.net.drm | 3.0 |
| DRMManagerError | flash.errors | 1.5 |
| EncryptedLocalStore | flash.data | 1.0 |
| ExtensionContext | flash.external | 2.5 |
| File | flash.filesystem | 1.0 |
| FileListEvent | flash.events | 1.0 |
| FileMode | flash.filesystem | 1.0 |
| FileStream | flash.filesystem | 1.0 |
| FocusDirection | flash.display | 1.0 |
| GameInput | flash.ui | 3.0 |
| GameInputControl | flash.ui | 3.0 |
| GameInputControlType | flash.ui | 3.6 en eerder; verwijderd vanaf 3.7 |
| GameInputDevice | flash.ui | 3.0 |
| GameInputEvent | flash.ui | 3.0 |
| GameInputFinger | flash.ui | 3.6 en eerder; verwijderd vanaf 3.7 |
| GameInputHand | flash.ui | 3.6 en eerder; verwijderd vanaf 3.7 |
| Geolocation | flash.sensors | 2.0 |
| GeolocationEvent | flash.events | 2.0 |
| HTMLHistoryItem | flash.html | 1.0 |

| Klasse | ActionScript 3.0-pakket | Toegevoegd in AIR-versie |
|---|---|---|
| HTMLHost | flash.html | 1.0 |
| HTMLLoader | flash.html | 1.0 |
| HTMLPDFCapability | flash.html | 1.0 |
| HTMLSWFCapabiltiy | flash.html | 2.0 |
| HTMLUncaughtScriptExceptionEvent | flash.events | 1.0 |
| HTMLWindowCreateOptions | flash.html | 1.0 |
| Icon | flash.desktop | 1.0 |
| *IFilePromise* | flash.desktop | 2.0 |
| ImageDecodingPolicy | flash.system | 2.6 |
| InteractiveIcon | flash.desktop | 1.0 |
| InterfaceAddress | flash.net | 2.0 |
| InvokeEvent | flash.events | 1.0 |
| InvokeEventReason | flash.desktop | 1.5.1 |
| IPVersion | flash.net | 2.0 |
| *IURIDereferencer* | flash.security | 1.0 |
| LocationChangeEvent | flash.events | 2.5 |
| MediaEvent | flash.events | 2.5 |
| MediaPromise | flash.media | 2.5 |
| MediaType | flash.media | 2.5 |
| MXRecord | flash.net.dns | 2.0 |
| NativeApplication | flash.desktop | 1.0 |
| NativeDragActions | flash.desktop | 1.0 |
| NativeDragEvent | flash.events | 1.0 |
| NativeDragManager | flash.desktop | 1.0 |
| NativeDragOptions | flash.desktop | 1.0 |
| NativeMenu | flash.display | 1.0 |
| NativeMenuItem | flash.display | 1.0 |
| NativeProcess | flash.desktop | 2.0 |
| NativeProcessExitEvent | flash.events | 2.0 |
| NativeProcessStartupInfo | flash.desktop | 2.0 |
| NativeWindow | flash.display | 1.0 |
| NativeWindowBoundsEvent | flash.events | 1.0 |
| NativeWindowDisplayState | flash.display | 1.0 |
| NativeWindowDisplayStateEvent | flash.events | 1.0 |

| Klasse | ActionScript 3.0-pakket | Toegevoegd in AIR-versie |
|---|---|---|
| NativeWindowInitOptions | flash.display | 1.0 |
| NativeWindowRenderMode | flash.display | 3.0 |
| NativeWindowResize | flash.display | 1.0 |
| NativeWindowSystemChrome | flash.display | 1.0 |
| NativeWindowType | flash.display | 1.0 |
| NetworkInfo | flash.net | 2.0 |
| NetworkInterface | flash.net | 2.0 |
| NotificationType | flash.desktop | 1.0 |
| OutputProgressEvent | flash.events | 1.0 |
| PaperSize | flash.printing | 2.0 |
| PrintMethod | flash.printing | 2.0 |
| PrintUIOptions | flash.printing | 2.0 |
| PTRRecord | flash.net.dns | 2.0 |
| ReferencesValidationSetting | flash.security | 1.0 |
| ResourceRecord | flash.net.dns | 2.0 |
| RevocationCheckSettings | flash.security | 1.0 |
| Screen | flash.display | 1.0 |
| ScreenMouseEvent | flash.events | 1.0 |
| SecureSocket | flash.net | 2.0 |
| SecureSocketMonitor | air.net | 2.0 |
| ServerSocket | flash.net | 2.0 |
| ServerSocketConnectEvent | flash.events | 2.0 |
| ServiceMonitor | air.net | 1.0 |
| SignatureStatus | flash.security | 1.0 |
| SignerTrustSettings | flash.security | 1.0 |
| SocketMonitor | air.net | 1.0 |
| SoftKeyboardType | flash.text | 3.0 |
| SQLCollationType | flash.data | 1.0 |
| SQLColumnNameStyle | flash.data | 1.0 |
| SQLColumnSchema | flash.data | 1.0 |
| SQLConnection | flash.data | 1.0 |
| SQLError | flash.errors | 1.0 |
| SQLErrorEvent | flash.events | 1.0 |
| SQLErrorOperation | flash.errors | 1.0 |

| Klasse | ActionScript 3.0-pakket | Toegevoegd in AIR-versie |
|---|---|---|
| SQLEvent | flash.events | 1.0 |
| SQLIndexSchema | flash.data | 1.0 |
| SQLMode | flash.data | 1.0 |
| SQLResult | flash.data | 1.0 |
| SQLSchema | flash.data | 1.0 |
| SQLSchemaResult | flash.data | 1.0 |
| SQLStatement | flash.data | 1.0 |
| SQLTableSchema | flash.data | 1.0 |
| SQLTransactionLockType | flash.data | 1.0 |
| SQLTriggerSchema | flash.data | 1.0 |
| SQLUpdateEvent | flash.events | 1.0 |
| SQLViewSchema | flash.data | 1.0 |
| SRVRecord | flash.net.dns | 2.0 |
| StageAspectRatio | flash.display | 2.0 |
| StageOrientation | flash.display | 2.0 |
| StageOrientationEvent | flash.events | 2.0 |
| StageText | flash.text | 3.0 |
| StageTextInitOptions | flash.text | 3.0 |
| StageWebView | flash.media | 2.5 |
| StatusFileUpdateErrorEvent | air.update.events | 1.5 |
| StatusFileUpdateEvent | air.update.events | 1.5 |
| StatusUpdateErrorEvent | air.update.events | 1.5 |
| StatusUpdateEvent | air.update.events | 1.5 |
| StorageVolume | flash.filesystem | 2.0 |
| StorageVolumeChangeEvent | flash.events | 2.0 |
| StorageVolumeInfo | flash.filesystem | 2.0 |
| SystemIdleMode | flash.desktop | 2.0 |
| SystemTrayIcon | flash.desktop | 1.0 |
| TouchEventIntent | flash.events | 3.0 |
| UpdateEvent | air.update.events | 1.5 |
| Updater | flash.desktop | 1.0 |
| URLFilePromise | air.desktop | 2.0 |

| Klasse | ActionScript 3.0-pakket | Toegevoegd in AIR-versie |
|---|---|---|
| URLMonitor | air.net | 1.0 |
| URLRequestDefaults | flash.net | 1.0 |
| XMLSignatureValidator | flash.security | 1.0 |

# Flash Player-klassen met specifieke functionaliteit voor AIR

De volgende klassen zijn beschikbaar voor SWF-inhoud die wordt uitgevoerd in de browser, maar AIR beschikt over aanvullende eigenschappen of methoden:

| Pakket | Klasse | Eigenschap, methode of gebeurtenis | Toegevoegd in AIR-versie |
|---|---|---|---|
| flash.desktop | Clipboard | `supportsFilePromise` | 2.0 |
| | ClipboardFormats | BITMAP_FORMAT | 1.0 |
| | | FILE_LIST_FORMAT | 1.0 |
| | | `FILE_PROMISE_LIST_FORMAT` | 2.0 |
| | | `URL_FORMAT` | 1.0 |
| flash.display | LoaderInfo | `childSandboxBridge` | 1.0 |
| | | `parentSandboxBridge` | 1.0 |
| | Stage | assignFocus() | 1.0 |
| | | autoOrients | 2.0 |
| | | deviceOrientation | 2.0 |
| | | `nativeWindow` | 1.0 |
| | | oriëntatie | 2.0 |
| | | gebeurtenis orientationChange | 2.0 |
| | | gebeurtenis orientationChanging | 2.0 |
| | | setAspectRatio | 2.0 |
| | | `setOrientation` | 2.0 |
| | | softKeyboardRect | 2.6 |
| | | supportedOrientations | 2.6 |
| | | supportsOrientationChange | 2.0 |
| | NativeWindow | owner | 2.6 |
| | | listOwnedWindows | 2.6 |
| | NativeWindowInitOptions | owner | 2.6 |

| Pakket | Klasse | Eigenschap, methode of gebeurtenis | Toegevoegd in AIR-versie |
|---|---|---|---|
| flash.events | Event | CLOSING | 1.0 |
| | | DISPLAYING | 1.0 |
| | | PREPARING | 2.6 |
| | | EXITING | 1.0 |
| | | HTML_BOUNDS_CHANGE | 1.0 |
| | | HTML_DOM_INITIALIZE | 1.0 |
| | | HTML_RENDER | 1.0 |
| | | LOCATION_CHANGE | 1.0 |
| | | NETWORK_CHANGE | 1.0 |
| | | STANDARD_ERROR_CLOSE | 2.0 |
| | | STANDARD_INPUT_CLOSE | 2.0 |
| | | STANDARD_OUTPUT_CLOSE | 2.0 |
| | | USER_IDLE | 1.0 |
| | | USER_PRESENT | 1.0 |
| | HTTPStatusEvent | HTTP_RESPONSE_STATUS | 1.0 |
| | | responseHeaders | 1.0 |
| | | responseURL | 1.0 |
| | KeyboardEvent | commandKey | 1.0 |
| | | controlKey | 1.0 |

| Pakket | Klasse | Eigenschap, methode of gebeurtenis | Toegevoegd in AIR-versie |
|---|---|---|---|
| flash.net | FileReference | `extension` | 1.0 |
| | | gebeurtenis `httpResponseStatus` | 1.0 |
| | | `uploadUnencoded()` | 1.0 |
| | NetStream | gebeurtenis `drmAuthenticate` | 1.0 |
| | | gebeurtenis `onDRMContentData` | 1.5 |
| | | `preloadEmbeddedData()` | 1.5 |
| | | `resetDRMVouchers()` | 1.0 |
| | | `setDRMAuthenticationCredentials()` | 1.0 |
| | URLRequest | authenticate | 1.0 |
| | | cacheResponse | 1.0 |
| | | followRedirects | 1.0 |
| | | idleTimeout | 2.0 |
| | | manageCookies | 1.0 |
| | | useCache | 1.0 |
| | | userAgent | 1.0 |
| | URLStream | `httpResponseStatus-gebeurtenis` | 1.0 |

| Pakket | Klasse | Eigenschap, methode of gebeurtenis | Toegevoegd in AIR-versie |
|---|---|---|---|
| flash.printing | PrintJob | `active` | 2.0 |
| | | `copies` | 2.0 |
| | | `firstPage` | 2.0 |
| | | `isColor` | 2.0 |
| | | `jobName` | 2.0 |
| | | `lastPage` | 2.0 |
| | | `maxPixelsPerInch` | 2.0 |
| | | `paperArea` | 2.0 |
| | | `printableArea` | 2.0 |
| | | `printer` | 2.0 |
| | | `printers` | 2.0 |
| | | `selectPaperSize()` | 2.0 |
| | | `showPageSetupDialog()` | 2.0 |
| | | `start2()` | 2.0 |
| | | `supportsPageSetupDialog` | 2.0 |
| | | `terminate()` | 2.0 |
| | PrintJobOptions | `pixelsPerInch` | 2.0 |
| | | `printMethod` | 2.0 |
| flash.system | Capabilities | `languages` | 1.1 |
| | LoaderContext | `allowLoadBytesCodeExecution` | 1.0 |
| | Security | `APPLICATION` | 1.0 |
| flash.ui | KeyLocation | D_PAD | 2.5 |

De meeste van deze nieuwe eigenschappen en methoden zijn alleen beschikbaar voor inhoud in de beveiligingssandbox van de AIR-toepassing. De nieuwe leden van de URLRequest-klassen zijn echter ook beschikbaar voor inhoud die wordt uitgevoerd in andere sandboxen.

De methoden `ByteArray.compress()` en `ByteArray.uncompress()` bevatten elk een nieuwe parameter `algorithm` waarmee u kunt kiezen tussen Deflate- en ZLIB-compressie. Deze parameter is alleen beschikbaar voor inhoud die wordt uitgevoerd in Adobe AIR.

# AIR-specifieke Flex-componenten

De volgende Adobe® Flex™ MX-componenten zijn beschikbaar wanneer u inhoud voor Adobe AIR ontwikkelt:

- FileEvent
- FileSystemComboBox
- FileSystemDataGrid

- FileSystemEnumerationMode
- FileSystemHistoryButton
- FileSystemList
- FileSystemSizeDisplayMode
- FileSystemTree
- FlexNativeMenu
- HTML
- Window
- WindowedApplication
- WindowedSystemManager

Daarnaast bevat Flex 4 de volgende spark AIR-componenten:

- Window
- WindowedApplication

Zie Using the Flex AIR components voor meer informatie over de AIR-componenten van Flex.

# Hoofdstuk 4: Gereedschappen van Adobe Flash-platform voor AIR-ontwikkeling

U kunt AIR-toepassingen ontwikkelen met de volgende programma's van het Adobe Flash-platform.

Voor ActionScript 3.0-ontwikkelaars (Flash en Flex):

- Adobe Flash Professional (zie Publiceren voor Adobe AIR)
- Adobe Flex 3.x en 4.x SDK's (zie "De SDK van Flex instellen" op pagina 19 en "AIR Developer Tool (ADT)" op pagina 173)
- Adobe Flash Builder (zie AIR-toepassingen ontwikkelen met Flash Builder)

Voor HTML- en Ajax-ontwikkelaars:

- SDK van Adobe AIR (zie "De SDK van AIR installeren" op pagina 17 en "AIR Developer Tool (ADT)" op pagina 173)
- Adobe Dreamweaver CS3, CS4, CS5 (zie Extensie AIR voor Dreamweaver)

## De SDK van AIR installeren

De SDK van Adobe AIR bevat de volgende opdrachtregelprogramma's waarmee u toepassingen kunt starten en verpakken:

**ADL (AIR Debug Launcher)** Hiermee kunt u AIR-toepassingen starten zonder deze eerst te installeren. Zie "ADL (AIR Debug Launcher)" op pagina 167.

**AIR Development Tool (ADT)** Hiermee kunt u AIR-toepassingen in distribueerbare pakketten plaatsen. Zie "AIR Developer Tool (ADT)" op pagina 173.

Voor de opdrachtregelprogramma's is vereist dat Java op de computer is geïnstalleerd. U kunt de virtuele Java-machine uit JRE of JDK (versie 1.5 of hoger) gebruiken. Java JRE en Java JDK kunnen worden gedownload via http://java.sun.com/.

Ten minste 2 GB computergeheugen is vereist voor het uitvoeren van het ADT-hulpprogramma.

***Opmerking:*** *eindgebruikers hebben Java niet nodig om AIR-toepassingen uit te voeren.*

Zie "Voor het eerst een AIR-toepassing op HTML-basis maken met de SDK van AIR" op pagina 35 voor een snel overzicht van het maken van AIR-toepassingen met de SDK van AIR.

### De SDK van AIR downloaden en installeren

Volg deze instructies om de SDK van AIR te downloaden en te installeren:

**De SDK van AIR in Windows installeren**

- Download het installatiebestand van de SDK van AIR.
- De SDK van AIR wordt als een standaardarchiefbestand gedistribueerd. Als u AIR wilt installeren, pakt u de inhoud van de SDK uit in een map op de computer (bijvoorbeeld: C:\Program Files\Adobe\AIRSDK of C:\AIRSDK).

- De programma's ADT en ADL staan in de map bin in de SDK van AIR. Voeg het pad naar deze map toe aan de omgevingsvariabele PATH.

**De SDK van AIR in Mac OS installeren**

- Download het installatiebestand van de SDK van AIR.
- De SDK van AIR wordt als een standaardarchiefbestand gedistribueerd. Als u AIR wilt installeren, pakt u de inhoud van de SDK uit in een map op de computer (bijvoorbeeld: /Users/<gebruikersnaam>/Applications/AIRSDK).
- De programma's ADT en ADL staan in de map bin in de SDK van AIR. Voeg het pad naar deze map toe aan de omgevingsvariabele PATH.

**De SDK van AIR in Linux installeren**

- De SDK is beschikbaar in de indeling tbz2.
- Als u de SDK wilt installeren, maakt u een map waarin u de SDK kunt uitpakken. Gebruik vervolgens deze opdracht: tar -jxvf <pad naar SDK van AIR.tbz2>

Zie AIR-toepassingen maken met behulp van opdrachtregelprogramma's voor informatie over het eerste gebruik van de SDK-programma's van AIR.

## Inhoud van de SDK van AIR

In de volgende tabel wordt beschreven wat het doel is van de bestanden in de SDK van AIR:

| **Map in SDK** | **Beschrijving van bestanden/programma's** |
|---|---|
| bin | Met ADL (AIR Debug Launcher) kunt u een AIR-toepassing starten zonder deze eerst te verpakken en te installeren. Zie voor informatie over het gebruik van dit hulpmiddel "ADL (AIR Debug Launcher)" op pagina 167.<br><br>De AIR Developer Tool (ADT) verpakt uw toepassing als een AIR-bestand voor distributie. Zie voor informatie over het gebruik van dit hulpmiddel "AIR Developer Tool (ADT)" op pagina 173. |
| frameworks | De bibliothekenmap bevat codebibliotheken die kunnen worden gebruikt in AIR-toepassingen.<br><br>De projectmap bevat de code voor de gecompileerde SWF- en SWC-bibliotheken. |
| include | De map include bevat het C-taal-headerbestand voor het schrijven van native extensies. |
| install | De installatiemap bevat de Windows USB-stuurprogramma's voor Android-apparaten. (Dit zijn de stuurprogramma's die door Google zijn geleverd in de Android SDK.) |
| lib | Bevat ondersteuningscode voor de AIR SDK-hulpmiddelen. |

| Map in SDK | Beschrijving van bestanden/programma's |
|---|---|
| runtimes | De AIR-runtimes voor het bureaublad en voor mobiele apparaten.<br><br>De bureaubladruntime wordt door ADL gebruikt om AIR-toepassingen te starten voordat deze zijn verpakt of geïnstalleerd.<br><br>De AIR-runtimes voor Android (APK-pakketten) kunnen worden geïnstalleerd op Android-apparaten of emulators voor ontwikkeling en tests. Afzonderlijke APK-pakketten worden gebruikt voor apparaten en emulators. (De openbare AIR-runtime voor Android is beschikbaar in de Android Market.) |
| voorbeelden | Deze map bevat een voorbeeld van een descriptorbestand van een toepassing, een voorbeeld van de functie voor naadloze installatie (badge.swf) en de standaardpictogrammen voor AIR-toepassingen. |
| sjablonen | descriptor-template.xml: een sjabloon van het descriptorbestand van een toepassing, dat verplicht is voor alle AIR-toepassingen. Zie "AIR-toepassingsdescriptorbestanden" op pagina 215 voor een gedetailleerde beschrijving van het descriptorbestand van een toepassing.<br><br>Schemabestanden voor de XML-structuur van de toepassingsdescriptor voor elke releaseversie van AIR die ook in deze map wordt aangetroffen. |

# De SDK van Flex instellen

Als u Adobe® AIR®-toepassingen wilt ontwikkelen met Adobe® Flex™, kunt u kiezen uit de volgende opties:

- U kunt Adobe® Flash® Builder™ downloaden en installeren. Hiermee beschikt u over geïntegreerde programma's om Adobe AIR-projecten te maken en om AIR-toepassingen te testen, fouten erin op te sporen en te verpakken. Zie "Uw eerste Flex-AIR-bureaubladtoepassing in Flash Builder maken" op pagina 21.
- U kunt de SDK van Adobe® Flex™ downloaden en Flex AIR-toepassingen ontwikkelen met uw favoriete teksteditor en opdrachtregelprogramma's.

Zie "Uw eerste AIR-bureaubladtoepassing met de Flex SDK maken" op pagina 39 voor een snel overzicht van het maken van AIR-toepassingen met de SDK van Flex.

## De SDK van Flex installeren

U kunt alleen AIR-toepassingen met de opdrachtregelprogramma's ontwikkelen als Java op de computer is geïnstalleerd. U kunt de virtuele Java-machine uit JRE of JDK (versie 1.5 of hoger) gebruiken. Java JRE en Java JDK kunnen worden gedownload via http://java.sun.com/.

***Opmerking:*** *eindgebruikers hebben Java niet nodig om AIR-toepassingen uit te voeren.*

De SDK van Flex voorziet u van de AIR-API en opdrachtregelprogramma's die u nodig hebt om AIR-toepassingen te verpakken, te compileren en om er fouten in op te sporen.

1. U kunt de SDK van Flex downloaden vanaf http://opensource.adobe.com/wiki/display/flexsdk/Downloads, indien van toepassing.
2. Plaats de inhoud van de SDK in een map (bijvoorbeeld met de naam Flex SDK).
3. Kopieer de inhoud van de SDK van AIR en schrijf daarbij de bestanden van de SDK van Flex over.

***Opmerking:*** *zorg er op Mac-computers voor dat u de afzonderlijke bestanden in ds SDK-mappen kopieert of verplaatst, geen volledige mappen. Standaard worden bij het kopiëren van een map op de Mac naar een map met dezelfde naam de bestaande bestanden in de doelmap verwijderd. De inhoud van de twee mappen wordt niet samengevoegd. U kunt de opdracht* `ditto` *in een terminalvenster gebruiken om de SK van AIR samen te voegen met de SDK van Flex:*`ditto air_sdk_folder flex_sdk_folder`

**4** De opdrachtregelprogramma's van AIR staan in de map bin.

# Externe SDK's instellen

Het ontwikkelen van toepassingen voor Android en iIOS vereist dat u inrichtingsbestanden, SDK's of andere ontwikkelingshulpmiddelen van de product van het platform downloadt.

Zie voor informatie over het downloaden in installeren van de Android SDK Android Developers: Installing the SDK. Per AIR 2.6 bent u niet vereist de SDK van Android te downloaden. De SDK van AIR bevat nu de basiscomponenten die nodig zijn voor het installeren en starten van APK-pakketten. De SDK van Android kan desondanks nuttig zijn voor verschillende ontwikkelingstaken, zoals het maken en uitvoeren van software-emulators en het nemen van schermafbeeldingen van het apparaat.

Een externe SDK is niet nodig voor iOS-ontwikkeling. Er zijn echter speciale certificaten en inrichtingsprofielen nodig. Zie voor meer informatie Bestanden voor ontwikkelaars verkrijgen van Apple.

# Hoofdstuk 5: Uw eerste AIR-toepassing maken

## Uw eerste Flex-AIR-bureaubladtoepassing in Flash Builder maken

Voor een snelle en praktische illustratie van de werking van Adobe® AIR® kunt u met deze instructies een eenvoudig AIR-bestand op SWF-basis maken en in een pakket plaatsen. Voor deze “Hello World”-toepassing gebruikt u Adobe® Flash® Builder.

Download en installeer Flash Builder, indien nodig. Download en installeer ook de meest recente versie van Adobe AIR vanaf www.adobe.com/go/air_nl.

### Een AIR-project maken

Flash Builder bevat hulpprogramma's waarmee u AIR-toepassingen kunt ontwikkelen en in een pakket plaatsen.

Wanneer u een AIR-toepassing maakt in Flash Builder of Flex Builder begint u op dezelfde manier als bij andere Flex-toepassingen: u definieert eerst een nieuw project.

1. Open Flash Builder.
2. Selecteer Bestand > Nieuw > Flex-project
3. Voer de projectnaam AIRHelloWorld in.
4. In Flex worden AIR-toepassingen als een toepassingstype beschouwd. Er zijn twee opties voor het type:
   - een webtoepassing die wordt uitgevoerd in Adobe® Flash® Player
   - een bureaubladtoepassing die wordt uitgevoerd in Adobe AIR

   Selecteer Desktop als toepassingstype.
5. Klik op Voltooien om het project te maken.

AIR-projecten bestaan aanvankelijk uit twee bestanden: het MXML-hoofdbestand en het XML-toepassingsbestand (dit wordt het descriptorbestand van de toepassing genoemd). In dit laatste bestand zijn de eigenschappen van de toepassing opgegeven.

Zie voor meer informatie Developing AIR applications with Flash Builder.

### De AIR-toepassingscode schrijven

Om de toepassingscode voor "Hello World" te schrijven, moet u het MXML-bestand van de toepassing (AIRHelloWorld.mxml) bewerken. Dit bestand is geopend in de editor. (Als dat niet het geval is, opent u het bestand via de projectnavigator.)

Flex-AIR-toepassingen op het bureaublad zijn opgenomen in de MXML WindowedApplication-tag. Met de tag MXML WindowedApplication wordt een eenvoudig venster gemaakt met daarin basisbesturingselementen voor vensters, zoals een titelbalk en een knop Sluiten.

1 Voeg een kenmerk `title` aan de component WindowedApplication toe en wijs hieraan de waarde `"Hello World"` toe:

```
<?xml version="1.0" encoding="utf-8"?>
<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
                       xmlns:s="library://ns.adobe.com/flex/spark"
                       xmlns:mx="library://ns.adobe.com/flex/mx"
                       title="Hello World">
</s:WindowedApplication>
```

2 Voeg een Label-component toe aan de toepassing (plaats deze in de tag WindowedApplication). Stel de eigenschap `text` van de Label-component in op `"Hello AIR"` en pas de lay-out aan zodat de tekst wordt gecentreerd, zoals hier aangegeven:

```
<?xml version="1.0" encoding="utf-8"?>
<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
                       xmlns:s="library://ns.adobe.com/flex/spark"
                       xmlns:mx="library://ns.adobe.com/flex/mx"
                       title="Hello World">

    <s:Label text="Hello AIR" horizontalCenter="0" verticalCenter="0"/>
</s:WindowedApplication>
```

3 Voeg meteen na de openingstag WindowedApplication en vóór de component Label die u zojuist hebt toegevoegd het volgende stijlblok toe:

```
<fx:Style>
    @namespace s "library://ns.adobe.com/flex/spark";
    s|WindowedApplication
    {

skinClass:ClassReference("spark.skins.spark.SparkChromeWindowedApplicationSkin");
        background-color:#999999;
        background-alpha:"0.7";
    }
</fx:Style>
```

Deze stijlinstellingen zijn van toepassing op de gehele toepassing en geven de achtergrond van het venster een enigszins transparante grijze kleur.

De toepassingscode ziet er nu als volgt uit:

```
<?xml version="1.0" encoding="utf-8"?>
<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
                       xmlns:s="library://ns.adobe.com/flex/spark"
                       xmlns:mx="library://ns.adobe.com/flex/mx"
                       title="Hello World">

    <fx:Style>
        @namespace s "library://ns.adobe.com/flex/spark";
        s|WindowedApplication
        {

skinClass:ClassReference("spark.skins.spark.SparkChromeWindowedApplicationSkin");
            background-color:#999999;
            background-alpha:"0.7";
        }
    </fx:Style>

    <s:Label text="Hello AIR" horizontalCenter="0" verticalCenter="0"/>
</s:WindowedApplication>
```

Vervolgens wijzigt u een aantal instellingen in het descriptorbestand van de toepassing, zodat de toepassing transparant wordt:

1. Ga in het navigatievenster van Flex naar het descriptorbestand van de toepassing in de bronmap van het project. Als u het project de naam AIRHelloWorld hebt gegeven, is de naam van dit bestand AIRHelloWorld-app.xml.
2. Dubbelklik op het descriptorbestand van de toepassing om deze in Flash Builder te bewerken.
3. Zoek in de XML-code naar de regels met commentaar voor de eigenschappen `systemChrome` en `transparent` (van de eigenschap `initialWindow`). Verwijder het commentaar. (Verwijder de commentaartekens `"<!--"` en `"-->"` scheidingstekens voor opmerkingen.)
4. Stel de tekstwaarde van de eigenschap `systemChrome` in op `none`, zoals hieronder:

   ```
   <systemChrome>none</systemChrome>
   ```

5. Stel de tekstwaarde van de eigenschap `transparent` in op `true`, zoals hieronder:

   ```
   <transparent>true</transparent>
   ```

6. Sla het bestand op.

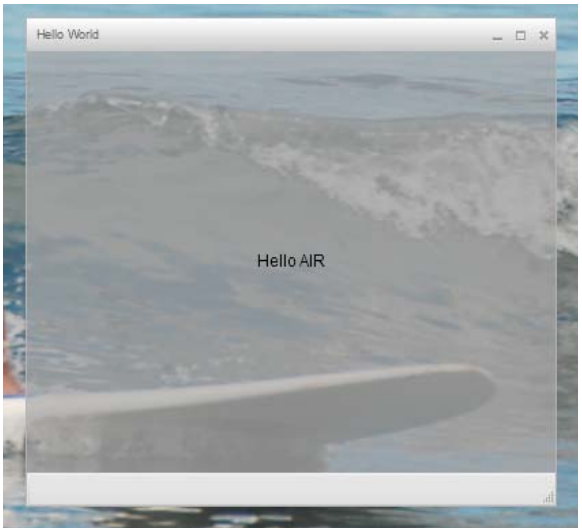
## De AIR-toepassing testen

U test de toepassingscode die u hebt geschreven door deze in de foutopsporingsmodus uit te voeren.

1 Klik op de knop Foutopsporing op de hoofdwerkbalk.

U kunt ook de opdracht Uitvoeren > Foutopsporing > AIRHelloWorld gebruiken.

De resulterende AIR-toepassing zou op het volgende voorbeeld moeten lijken:




2 Met de eigenschappen `horizontalCenter` en `verticalCenter` of het besturingselement Label wordt de tekst midden in het venster geplaatst. U kunt het venster verplaatsen en groter of kleiner maken, net zoals bij andere bureaubladtoepassingen.

***Opmerking:*** *als de toepassing niet gecompileerd kan worden, zoekt u naar eventuele syntaxis- of spelfouten die u per ongeluk in de code hebt gemaakt. Fouten en waarschuwingen worden in de weergave Problemen in Flash Builder weergegeven.*

## De AIR-toepassing in een pakket plaatsen, ondertekenen en uitvoeren

U kunt de toepassing 'Hello World' nu in een AIR-bestand plaatsen dat gedistribueerd kan worden. Een AIR-bestand is een archiefbestand dat de toepassingsbestanden bevat. Deze bestanden staan allemaal in de map bin van het project. In dit eenvoudige voorbeeld zijn dat de SFW- en de XML-toepassingsbestanden. Het AIR-pakket verzendt u aan gebruikers die dit gebruiken om de toepassing te installeren. Een vereiste stap in dit proces is het digitaal ondertekenen van het pakket.

1 Controleer of de toepassing geen compilatiefouten bevat en goed wordt uitgevoerd.

2 Selecteer Project > Exportversie.

3 Controleer of het AIRHelloWorld-project en de AIRHelloWorld.mxml-toepassing voor het project en de toepassing worden vermeld.

4 Selecteer de optie Exporteren als ondertekend AIR-pakket. Klik daarna op Volgende.

5 Als u over een bestaand digitaal certificaat beschikt, klikt u op Bladeren om dit te selecteren.

6 Als u een nieuw, niet-geautoriseerd digitaal certificaat moet maken, klikt u op Maken.

7 Voer de gegevens in en klik op OK.

8 Klik op Voltooien om het AIR-pakket te genereren. Dit pakket krijgt te naam AIRHelloWorld.air.

Dubbelklik op het AIR-bestand om de toepassing te installeren en uit te voeren vanuit de Flash Builder-projectnavigator of vanuit het bestandssysteem.

# Uw eerste AIR-bureaubladtoepassing maken met Flash Professional

Voor een snelle, praktische demonstratie van de werking van Adobe® AIR® volgt u de aanwijzingen in dit onderwerp om een eenvoudige “Hello World” AIR-toepassing te maken en te comprimeren met Adobe® Flash® Professional.

Als u dit nog niet hebt gedaan, downloadt en installeert u Adobe AIR, te vinden op de volgende locatie: www.adobe.com/go/air_nl.

## De Hello World-toepassing in Flash maken

Een Adobe AIR-toepassing in Flash maken, is vergelijkbaar met het maken van iedere andere FLA-toepassing. Met de volgende procedure doorloopt u het proces voor het maken van een eenvoudige Hello World-toepassing met behulp van Flash Professional.

**De Hello World-toepassing aanmaken**

1 Start Flash.

2 In het welkomstscherm klikt u op AIR om een leeg FLA-bestand aan te maken met Adobe AIR-publicatie-instellingen.

3 Selecteer Tekstgereedschap in het venster Gereedschappen en maak een statisch tekstveld (standaard) in het midden van de stage. Maak het breed genoeg zodat het 15 -20 tekens kan bevatten.

4 Typ de tekst “Hello World” in het tekstveld.

5 Sla het bestand op met een naam (bijvoorbeeld helloAIR).

## De toepassing testen

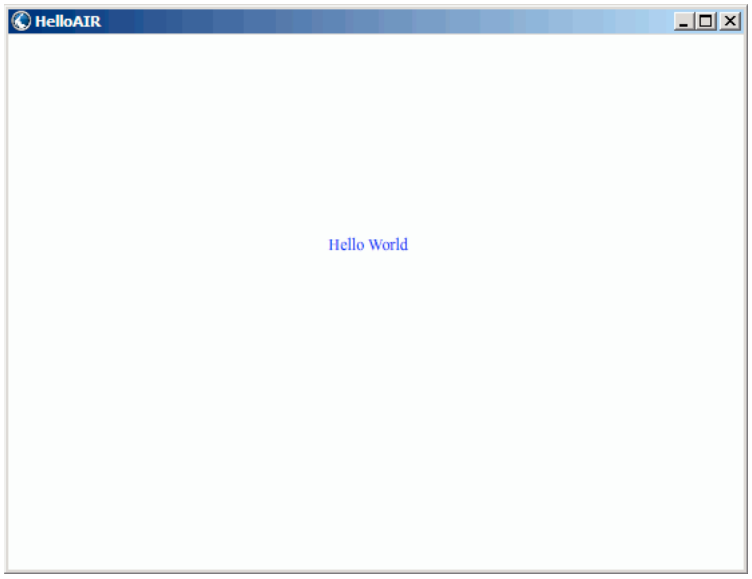
1 Druk op Ctrl + Enter of selecteer Besturing ->Film testen->Testen om de toepassing te testen in Adobe AIR.Test Movie.

2 Om de functie Debug Movie te gebruiken, voegt u eerst de ActionScript-code toe aan de toepassing. U kunt dit snel proberen door als volgt een trace statement toe te voegen:

```
trace("Running AIR application using Debug Movie");
```

3 Druk op Ctrl + Shift + Enter, of selecteer Debuggen->Film debuggen->Debuggen om de toepassing met Debug Movie uit te voeren.

De Hello World-toepassing ziet er als volgt uit:




## De toepassing in een pakket plaatsen

1. Selecteer Bestand > Publiceren.
2. Onderteken het Adobe AIR-pakket met een bestaand digitaal certificaat of maak een zelfondertekend certificaat door deze stappen te volgen:
   - **a** Klik op de knop Nieuw naast het veld Certificaat.
   - **b** Vul de volgende velden in: Naam uitgever, Afdeling organisatie, Naam organisatie, E-mail, Land, Wachtwoord, en Wachtwoord bevestigen.
   - **c** Geef het soort certificaat op. Het soort certificaat houdt verband met het niveau van beveiliging: 1024-RSA gebruikt een 1024-bits sleutel (minder veilig), en 2048-RSA gebruikt een 2048-bits sleutel (veiliger).
   - **d** Sla de gegevens op in een certificaatbestand via Opslaan als, of door te klikken op de knop Bladeren... om naar een maplocatie te gaan. (Bijvoorbeeld, *C:/Temp/mycert.pfx*). Klik als u klaar bent op OK.
   - **e** Flash leidt u terug naar het dialoogvenster Digitale handtekening. Het pad en de bestandsnaam van het zelfondertekende certificaat dat u hebt gemaakt, verschijnt in het tekstvak Certificaat. Indien dit niet het geval is, voert u het pad en de bestandsnaam in of klikt u op de knop Bladeren om het te zoeken en te selecteren.
   - **f** Voer hetzelfde wachtwoord in als u bij stap b in het tekstveld Wachtwoord van het dialoogvenster Digitale handtekening hebt ingevoerd. Zie "AIR-bestanden digitaal ondertekenen" op pagina 199 voor meer informatie over het ondertekenen van Adobe AIR-toepassingen.
3. Druk op de knop Publiceren om de toepassing en het installatiebestand te maken. (In Flash CS4 en CS5 klikt u op de knop OK.) U moet Test Movie of Debug Movie uitvoeren om het SWF-bestand en de .xml-bestanden van de toepassing aan te maken voordat u het AIR-bestand aanmaakt.
4. Om de toepassing te installeren, dubbelklikt u op het AIR-bestand (*toepassing*.air) in dezelfde map waarin u uw toepassing hebt opgeslagen.
5. Klik op de knop Installeren in het dialoogvenster Toepassing installeren.
6. Controleer de installatievoorkeuren en de locatie-instellingen en zorg dat het vakje 'Toepassing opstarten na installatie' is aangevinkt. Klik vervolgens op Doorgaan.

7 Als het bericht Installatie voltooid verschijnt, klikt u op Afsluiten.

# Maak uw eerste AIR-toepassing voor Android in Flash Professional

Om AIR-toepassingen voor Android te kunnen ontwikkelen moet u de uitbreiding Flash Professional CS5 voor Android downloaden van Adobe Labs.

U moet ook de Android SDK downloaden van de Android-website en installeren, zoals beschreven in: Android Developers: Installing the SDK.

### Een project maken

1 Open Flash Professional CS5

2 Maak een nieuw AIR-project voor Android.

Het hoofdscherm van Flash Professional bevat een koppeling voor het maken van een AIR-toepassing voor Android. U kunt ook Bestand > Nieuw selecteren en vervolgens de sjabloon AIR voor Android selecteren.

3 Sla het document op als HelloWorld.fla

### Schrijf de code

Aangezien deze zelfstudie niet echt over het schrijven van code gaat, gebruikt u het tekstgereedschap om te "Hello, World!" in het werkgebied te schrijven.

### Stel de toepassingseigenschappen in

1 Selecteer Bestand > AIR Android-instellingen.

2 Selecteer op het tabblad Algemeen de volgende instellingen:

- Uitvoerbestand: HelloWorld.apk
- Toepassingsnaam: HelloWorld
- Toepassings-id: HelloWorld
- Versienummer: 0.0.1
- Hoogte-breedteverhouding: Staand

3 Selecteer op het tabblad Implementatie de volgende instellingen:

- Certificaat: Wijs naar een geldig certificaat voor ondertekenen van code voor AIR. U kunt op de knop Maken drukken op een nieuw certificaat te maken. (Android-toepassingen die worden geïmplementeerd via de Android Marketplace, moeten certificaten hebben die geldig zijn tot minstens 2033.) Voer het certificaatwachtwoord in het veld Wachtwoord in.
- Implementatietype Android: Debug
- Na Publiceren: Selecteer beide opties
- Voer het pad naar het ADB-hulpprogramma in in de submap voor hulpprogramma's van de Android SDK.

4 Sluit het instellingenvenster van Android door op OK te drukken.

De toepassing heeft in dit stadium van ontwikkeling geen pictogrammen of machtigingen nodig. De meeste AIR-toepassingen voor Android vereisen machtigingen om beschermde functies te kunnen gebruiken. U moet alleen de machtigingen instellen die uw toepassing echt nodig heeft, aangezien gebruikers uw toepassing kunnen weigeren als deze om te veel machtigingen vraagt.

5 Sla het bestand op.

**Verpak in installeer de toepassing op het Android-apparaat**

1 Zorg ervoor dat USB-foutopsporing is ingeschakeld op uw apparaat. U kunt USB-foutopsporing inschakelen in de toepassing Instellingen onder Toepassing > Ontwikkeling.

2 Sluit het apparaat op uw computer aan met een USB-kabel.

3 Installeer de AIR-runtime, als u dat nog niet hebt gedaan, door naar de Android Market te gaan en Adobe AIR te downloaden. (U kunt ook AIR lokaal installeren met behulp van de "ADT-opdracht installRuntime" op pagina 185. Android-pakketten voor gebruik op Android-apparaten en emulators zijn opgenomen in de AIR SDK.)

4 Selecteer Bestand > Publiceren.

Flash Professional maakt het APK-bestand, installeert de toepassing op het aangesloten Android-apparaat en start deze.

# Uw eerste AIR-toepassing voor iOS maken

**AIR 2.6 of hoger, iOS 4.2 of hoger**

U kunt de basisfuncties van een iOS-toepassing coderen, bouwen en testen met niet meer dan Adobe-hulpprogramma's. Als u echter een iOS-toepassing op een apparaat wilt installeren en die toepassing wilt distribueren, moet u zich aansluiten bij het Apple iOS Developer-programma (deze service is gratis). Zodra u zich bij het Apple iOS-ontwikkelprogramma hebt aangemeld, kunt u naar de iOS Provisioning Portal gaan voor de volgende items en bestanden van Apple die worden vereist voor het installeren van een toepassing op een apparaat voor het uitvoeren van testen en vervolgens voor distributie. Deze items en bestanden bevatten onder andere:

- Ontwikkelings- en distributiecertificaten
- Toepassings-id's
- Inrichtingsbestanden voor ontwikkeling en distributie

## Maak de toepassingsinhoud

Maak een SWF-bestand waarin de tekst "Hello world!" wordt weergegeven U kunt deze taak uitvoeren met behulp van Flash Professional, Flash Builder of een andere IDE. In dit voorbeeld worden een teksteditor en de opdrachtregel voor het compileren van SWF opgenomen in de Flex SDK.

1 Maak een map op een handige locatie om uw toepassingsbestanden op te slaan. Maak een bestand met de naam *HelloWorld.as* en bewerk het bestand in uw favoriete code-editor.

2 Voeg de volgende code toe:

```
package{

    import flash.display.Sprite;
    import flash.text.TextField;
    import flash.text.TextFormat;
    import flash.text.TextFieldAutoSize;

    public class HelloWorld extends Sprite
    {
        public function HelloWorld():void
        {
            var textField:TextField = new TextField();
            textField.text = "Hello World!";
            textField.autoSize = TextFieldAutoSize.LEFT;

            var format:TextFormat = new TextFormat();
            format.size = 48;

            textField.setTextFormat ( format );
            this.addChild( textField );
        }
    }
}
```

**3** Compileer de klasse met de compiler amxmlc.

```
amxmlc HelloWorld.as
```

In dezelfde map wordt het SWF-bestand *HelloWorld.swf* gemaakt.

***Opmerking:*** *in dit voorbeeld wordt ervan uitgegaan dat u uw pad voor omgevingsvariabele zo hebt ingesteld dat de map met amxmlc is opgenomen. Zie voor informatie over het instellen van het pad "Omgevingsvariabelen van het pad" op pagina 318. U kunt ook het volledige pad invoeren naar amxmlc en de andere opdrachtregelhulpprogramma's die in dit voorbeeld worden gebruikt.*

## Pictogramafbeeldingen en een afbeelding voor het startscherm van de toepassing maken

Alle iOS-toepassingen hebben pictogrammen die worden weergegeven in de gebruikersinterface van de toepassing iTunes en op het scherm van het apparaat.

**1** Maak een map in de projectmap en geef de map de naam Pictogrammen.

**2** Maak drie PNG-bestanden in de map Pictogrammen. Geef de bestanden de namen Icon_29.png, Icon_57.png en Icon_512.png.

**3** Bewerk de PNG-bestanden om de juiste afbeeldingen voor uw toepassing te maken. De bestanden moeten 29 bij 29 pixels, 57 bij 57 pixels en 512 bij 512 pixels goot zijn. Voor deze tekst kunt u gewoon effen vierkante kleurvlakken als afbeelding gebruiken.

***Opmerking:*** *wanneer u een toepassing naar de Apple App Store verzendt, gebruikt u een JPG-versie (geen PNG-versie) van het bestand van 512 pixels. U gebruikt de PNG-versie tijdens het testen van de ontwikkelversies van een toepassing.*

In alle iPhone-toepassingen wordt een eerste afbeelding weergegeven terwijl de toepassing op de iPhone wordt geladen. U definieert de startinstellingen in een PNG-bestand:

1 Maak in de hoofdontwikkelmap een PNG-bestand met de naam Default.png. (Plaats dit bestand *niet* in de submap Pictogrammen. Geef het bestand de naam Default.png met een hoofdletter D.)

2 Bewerk het bestand zodat het 320 pixels breed is en 480 pixels hoog. Voor dit doeleinde kan de inhoud een normale witte rechthoek zijn. (U wijzigt dit later.)

Zie voor gedetailleerde informatie over deze afbeeldingen "Toepassingspictogrammen" op pagina 92.

## Maak het descriptorbestand van de toepassing

Maak een toepassingsdescriptorbestand waarin de basiseigenschappen voor de toepassing worden opgegeven. U kunt deze taak voltooien met een IDE als Flash Builder of een teksteditor.

1 Maak in de projectmap HelloWorld.as bevat, een XML-bestand met de naam *HelloWorld-app.xml*. Verwerk dit bestand in een XML-editor naar keuze.

2 Voeg de volgende XML toe:

```
<?xml version="1.0" encoding="utf-8" ?>
<application xmlns="http://ns.adobe.com/air/application/2.7" minimumPatchLevel="0">
    <id>change_to_your_id</id>
    <name>Hello World iOS</name>
    <versionNumber>0.0.1</versionNumber>
    <filename>HelloWorld</filename>
    <supportedProfiles>mobileDevice</supportedProfiles>
    <initialWindow>
        <content>HelloWorld.swf</content>
        <title>Hello World!</title>
    </initialWindow>
    <icon>
        <image29x29>icons/AIRApp_29.png</image29x29>
        <image57x57>icons/AIRApp_57.png</image57x57>
        <image512x512>icons/AIRApp_512.png</image512x512>
    </icon>
</application>
```

In dit vereenvoudigde voorbeeld worden slechts een paar van de beschikbare eigenschappen ingesteld.

***Opmerking:*** *als u AIR 2 of eerder gebruikt, moet u het element* `<version>` *gebruiken in plaats van het element* `<versionNumber>`.

3 Wijzig de toepassings-id zo dat deze overeenkomt met de toepassings-id die is opgegeven in de iOS Provisioning Portal. (Neem geen willekeurig gedeelte van de bundlebron op aan het begin van de ID.

4 Test de toepassing met behulp van ADL:

```
adl HelloWorld-app.xml -screensize iPhone
```

ADL moet een venster op uw bureaublad openen met de tekst *Hello World!* Als dit niet gebeurt, controleert u de broncode en de toepassingsdescriptor op fouten.

## Het IPA-bestand compileren

U kunt het IPA-installatieprogramma nu compileren met ADT. U moet uw Apple-ontwikkelaarscertificaat en persoonlijke sleutel in de bestandsindeling P12 en uw Apple-ontwikkelinrichtingsprofiel hebben.

Voer het ADT-hulpmiddel met de volgende opties uit, waarbij u de waarden keystore, storepass en provisioning-profile vervangt door uw eigen:

```
adt -package -target ipa-debug
    -keystore iosPrivateKey.p12 -storetype pkcs12 -storepass qwerty12
    -provisioning-profile ios.mobileprovision
    HelloWorld.ipa
    HelloWorld-app.xml
    HelloWorld.swf icons Default.png
```

(Gebruik een enkele opdrachtregel. De regelafbrekingen in dit voorbeeld zijn toegevoegd om de tekst eenvoudiger leesbaar te maken.)

ADT genereert het iOS-toepassingsinstallatiebestand *HelloWorld.ipa* in de projectmap. Het duurt een paar minuten voordat het IPA-bestand is gecompileerd.

## De toepassing op een apparaat installeren

De iOS-toepassing installeren om te testen:

1. Open de toepassing iTunes.
2. Als u dit nog niet hebt gedaan, voegt u het inrichtingsprofiel voor deze toepassing aan iTunes toe. Selecteer in iTunes Archief > Voeg toe aan bibliotheek. Selecteer vervolgens het inrichtingsprofielbestand (met mobileprovision als bestandstype).

   Gebruik voorlopig het ontwikkelinrichtingsprofiel om de toepassing op het ontwikkelaars-apparaat te testen.

   Gebruik later, wanneer u een toepassing naar de iTunes Store distribueert, het distributieprofiel. Gebruik het ad-hocinrichtingsprofiel als u de toepassing ad hoc (naar meerdere apparaten zonder tussenkomst van de iTunes Store) wilt distribueren.

   Zie “iOS installeren” op pagina 70 voor meer informatie over inrichtingsprofielen.
3. In sommige versies van iTunes wordt de toepassing niet vervangen als dezelfde versie van de toepassing al is geïnstalleerd. Verwijder in dit geval de toepassing van het apparaat en uit de lijst met toepassingen in iTunes.
4. Dubbelklik op het IPA-bestand voor de toepassing. De toepassing moet in uw lijst met iTunes-toepassingen worden weergegeven.
5. Sluit uw apparaat aan op de USB-poort van uw computer.
6. Controleer in iTunes het tabblad Toepassing voor het apparaat en zorg ervoor dat de toepassing is geselecteerd in de lijst met toepassingen die moeten worden geïnstalleerd.
7. Selecteer het apparaat in de lijst links van iTunes. Klik op de knop Synchroniseren. Wanneer het synchronisatieproces is voltooid, verschijnt de Hello World-toepassing op uw iPhone.

Als de nieuwe versie niet is geïnstalleerd, verwijdert u deze van uw apparaat en uit de lijst met toepassingen in iTunes en herhaalt u vervolgens deze procedure. Dit kan zich voordoen als de versie die op dat moment is geïnstalleerd, dezelfde toepassings-id en toepassingsversie gebruikt.

## De afbeelding voor het startscherm bewerken

Voordat u uw toepassing maakte, hebt u het bestand Default.png gemaakt (zie “Pictogramafbeeldingen en een afbeelding voor het startscherm van de toepassing maken” op pagina 29). Dit PNG-bestand dient als de opstartafbeelding die tijdens het laden van de toepassing wordt weergegeven. Toen u de toepassing testte op uw iPhone, hebt u waarschijnlijk dit lege scherm bij het opstarten gezien.

U moet de afbeelding wijzigen, zodat deze overeenkomt met het startscherm van uw toepassing “Hello World!”.:

1 Open de toepassing op het apparaat. Wanneer de eerste tekst “Hello World” wordt weergegeven, drukt u op de thuisknop (onder het scherm) en houdt u deze knop ingedrukt. Druk met de thuisknop ingedrukt op de knop voor de sluimerstand (boven aan de iPhone). Zo maakt u een schermafbeelding en verzendt u de afbeelding naar het Filmrol-album.

2 Breng de afbeelding over naar uw ontwikkelcomputer door deze over te brengen van de iPhone of door een andere toepassing voor het overbrengen van foto's te gebruiken. (Op een Mac-computer kunt u hiervoor de toepassing Image Capture gebruiken.)

   U kunt de afbeelding ook via e-mail naar uw ontwikkelcomputer verzenden:

   - Open de toepassing Foto's.
   - Open het Filmrol-album.
   - Open de schermafbeelding die u hebt vastgelegd.
   - Tik op de foto en tik vervolgens op de pijlknop “vooruit” in de linkerbenedenhoek. Klik vervolgens op de knop voor het verzenden van foto's en verzendt de foto naar uzelf.

3 Vervang het bestand Default.png (in de ontwikkelmap) door een PNG-versie van de schermafbeelding.

4 Compileer de toepassing opnieuw (zie “Het IPA-bestand compileren” op pagina 30) en installeer deze opnieuw op uw apparaat.

In de toepassing wordt nu het nieuwe opstartscherm gebruikt terwijl de toepassing wordt geladen.

***Opmerking:*** *u kunt elke gewenste afbeelding voor het bestand Default.png gebruiken, zo lang de afbeelding de juiste afmetingen heeft (320 bij 480 pixels). Het is echter vaak het beste als de afbeelding Default.png overeenkomt met de beginstatus van uw toepassing.*

# Voor het eerst een AIR-toepassing op HTML-basis maken met Dreamweaver

Voor een snelle en praktische illustratie van de werking van Adobe® AIR® kunt u met deze instructies een eenvoudige AIR-toepassing op HTML-basis maken en in een pakket plaatsen. Voor deze “Hello World”-toepassing gebruikt u de Adobe® AIR®-extensie voor Dreamweaver®.

Als u dit nog niet hebt gedaan, downloadt en installeert u Adobe AIR, te vinden op de volgende locatie: www.adobe.com/go/air_nl.

Zie Extensie AIR voor Dreamweaver installeren voor instructies voor het installeren van de Adobe AIR-extensie voor Dreamweaver.

Zie Extensie AIR voor Dreamweaver voor een overzicht van de extensie, inclusief systeemvereisten.

***Opmerking:*** *AIR-toepassingen op basis van HTML kunnen alleen worden ontwikkeld voor het profielen desktop en extendedDesktop. Het mobiele profiel wordt niet ondersteund.*

## De toepassingsbestanden voorbereiden

De beginpagina en alle gerelateerde pagina's van de Adobe AIR-toepassing moeten in een Dreamweaver-site worden gedefinieerd:

1 Start Dreamweaver en zorg ervoor dat er een site is gedefinieerd.

2 Open een nieuwe HTML-pagina als volgt: selecteer Bestand > Nieuw, selecteer HTML in de kolom Paginatype, selecteer Geen in de kolom Indeling en klik op Maken.

3 Typ **Hello World!** in de nieuwe pagina.

Dit voorbeeld is zeer eenvoudig, maar u kunt desgewenst de tekst opmaken, meer inhoud aan de pagina toevoegen, andere pagina's aan deze startpagina toevoegen, enzovoort.

4 Sla de pagina op (Bestand > Opslaan) als hello_world.html. Let erop dat het bestand in een Dreamweaver-site wordt opgeslagen.

Zie de Help bij Dreamweaver voor meer informatie over Dreamweaver-sites.

## De Adobe AIR-toepassing maken

1 Zorg ervoor dat de pagina hello_world.html is geopend in het documentvenster van Dreamweaver. (Raadpleeg de voorgaande secties voor instructies over het maken van deze pagina.)

2 Selecteer Site > Air-toepassingsinstellingen.

De meeste verplichte instellingen in het dialoogvenster Instellingen voor AIR-toepassing en installatieprogramma worden automatisch ingevuld. U moet echter de eerste inhoud (of startpagina) van de toepassing selecteren.

3 Klik op de knop Bladeren naast de optie Aanvankelijke inhoud, navigeer naar de pagina hello_world.html en selecteer deze pagina.

4 Klik op de knop Instellen naast de optie Digitale handtekening.

Een digitale handtekening vormt een garantie dat de code voor een toepassing niet is gewijzigd of beschadigd is geraakt sinds deze is gemaakt door de auteur van de software. Voor alle Adobe AIR-toepassingen is een digitale handtekening vereist.

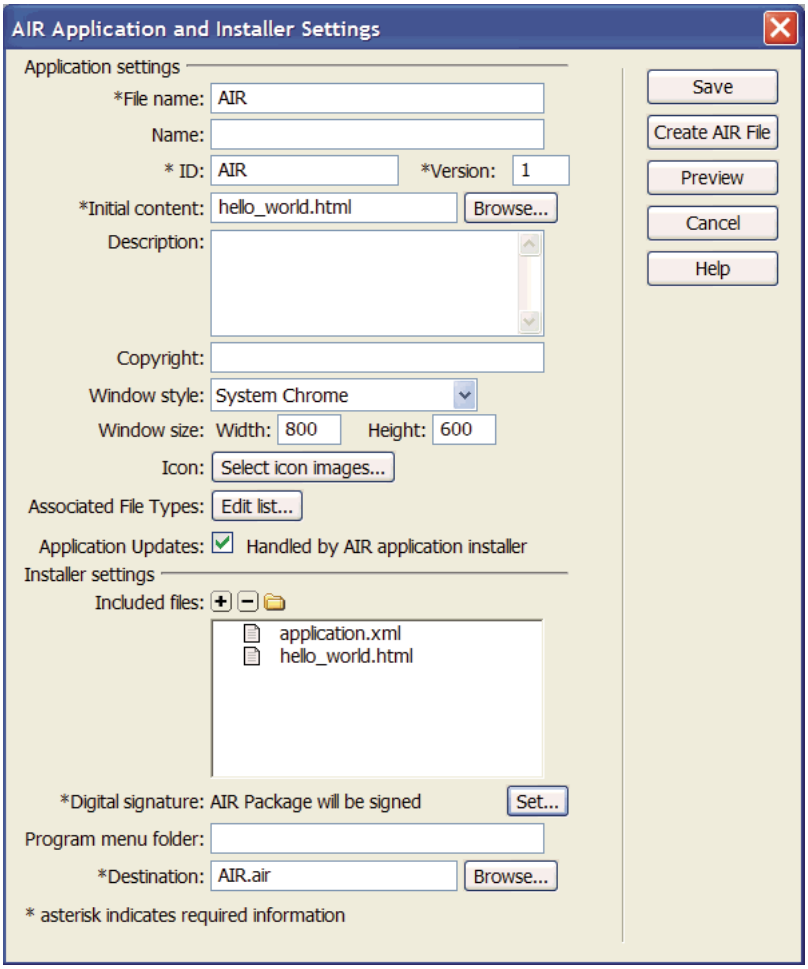
5 Selecteer in het dialoogvenster Digitale handtekening de optie AIR-bestand ondertekenen met digitaal certificaat en klik op de knop Maken. (Als u al toegang hebt tot een digitaal certificaat, kunt u op de knop Bladeren klikken om dit te selecteren.)

6 Vul de velden in het dialoogvenster Niet-geautoriseerd digitaal certificaat in. U moet de volgende gegevens invoeren: uw naam, een wachtwoord (dat u moet bevestigen) en een naam voor het bestand voor het digitale certificaat. Het digitale certificaat wordt in de hoofdmap van de site opgeslagen.

7 Klik op OK om terug te keren naar het dialoogvenster Digitale handtekening.

8 Voer in het dialoogvenster Digitale handtekening het wachtwoord in dat u voor het digitale certificaat hebt opgegeven en klik op OK.

Wanneer het dialoogvenster Adobe AIR - Instellingen voor toepassing en installatieprogramma volledig is ingevuld, ziet het er ongeveer zo uit:




Zie Een AIR-toepassing maken in Dreamweaver als u meer wilt weten over de opties in het dialoogvenster en over hoe u deze opties kunt bewerken.

9 Klik op de knop AIR-bestand maken.

Het Adobe AIR-toepassingsbestand wordt gemaakt en opgeslagen in de hoofdmap van de site. Daarnaast wordt een bestand met de naam application.xml gemaakt en op dezelfde locatie opgeslagen. Dit bestand vormt een manifest, waarin verschillende eigenschappen van de toepassing worden gedefinieerd.

## De toepassing op een bureaublad installeren

Nu het toepassingsbestand is gemaakt, kunt u het op elk gewenst bureaublad installeren.

1 Verplaats het Adobe AIR-toepassingsbestand uit de Dreamweaver-site naar het bureaublad of het bureaublad op een andere computer.

Dfeze stap is optioneel. U kunt de nieuwe toepassing ook rechtstreeks vanuit de Dreamweaver-site op uw computer installeren.

2 Dubbelklik op het uitvoerbare bestand van de toepassing (met extensie .air) om de toepassing te installeren.

## Voorbeeld van de Adobe AIR-toepassing bekijken

U kunt wanneer u maar wilt voorbeelden bekijken van de pagina's die deel uitmaken van AIR-toepassingen. U hoeft de toepassing dus niet in een pakket te plaatsen om te zien hoe de toepassing er na installatie uitziet.

1 Open de pagina hello_world.html in het documentvenster van Dreamweaver.

2 Ga naar de werkbalk Document, klik op de knop Voorvertonen/Fouten opsporen in browser en selecteer Voorbeeld in AIR.

   U kunt ook op Ctrl+Shift+F12 (Windows) of Cmd+Shift+F12 (Macintosh) drukken.

   Wanneer u een voorbeeld van deze pagina bekijkt, ziet u wat een gebruiker te zien krijgt op de beginpagina van de toepassing nadat de toepassing op een computer is geïnstalleerd.

# Voor het eerst een AIR-toepassing op HTML-basis maken met de SDK van AIR

Voor een snelle en praktische illustratie van de werking van Adobe® AIR® kunt u met deze instructies een eenvoudige AIR-toepassing op HTML-basis ("Hello World") maken en in een pakket plaatsen.

Om te beginnen moet de runtime zijn geïnstalleerd en de SDK van AIR zijn ingesteld. In deze zelfstudie worden *ADL* (AIR Debug Launcher) en *ADT* (AIR Developer Tool) gebruikt. ADL en ADT zijn opdrachtregelprogramma's en staan in de map `bin` van de SDK van AIR (zie "De SDK van AIR installeren" op pagina 17). Bij deze zelfstudie wordt ervan uitgegaan dat u weet hoe u programma's vanaf de opdrachtregel uitvoert en hoe u de juiste padomgevingsvariabelen voor het besturingssysteem instelt.

***Opmerking:*** *als u ® Dreamweaver® gebruikt, gaat u naar "Voor het eerst een AIR-toepassing op HTML-basis maken met Dreamweaver" op pagina 32.*

***Opmerking:*** *AIR-toepassingen op basis van HTML kunnen alleen worden ontwikkeld voor het profielen desktop en extendedDesktop. Het mobiele profiel wordt niet ondersteund.*

## De projectbestanden maken

Elk AIR-project op HTML-basis moet de volgende twee bestanden bevatten: een descriptorbestand, waarin de metagegevens van de toepassing zijn opgegeven, en een HTML-pagina voor het hoogste niveau. Naast deze twee vereiste bestanden bevat dit project een bestand met JavaScript-code: `AIRAliases.js`. Hierin worden handige aliasvariabelen voor de AIR API-klassen gedefinieerd.

1 Maak een map met de naam `HelloWorld` waarin de projectbestanden worden geplaatst.

2 Maak een XML-bestand met de naam `HelloWorld-app.xml`.

3 Maak een HTML-bestand met de naam `HelloWorld.html`.

4 Kopieer `AIRAliases.js` uit de map frameworks van de SDK van AIR naar de projectmap.

## Het descriptorbestand van de AIR-toepassing maken

Het bouwen van de AIR-toepassing begint met het maken van een XML-descriptorbestand voor de toepassing met de volgende structuur:

```
<application xmlns="...">
    <id>…</id>
    <versionNumber>…</versionNumber>
    <filename>…</filename>
    <initialWindow>
        <content>…</content>
        <visible>…</visible>
        <width>…</width>
        <height>…</height>
    </initialWindow>
</application>
```

1 Open het bestand `HelloWorld-app.xml` om te bewerken.

2 Voeg het basiselement `<application>` toe, inclusief het naamruimtekenmerk voor AIR:

**<application xmlns="http://ns.adobe.com/air/application/2.7">** Het laatste segment van de naamruimte, '2.7', geeft aan welke versie van de runtime voor de toepassing is vereist.

3 Voeg het element `<id>` toe:

**<id>examples.html.HelloWorld</id>**De toepassings-id vormt tezamen met de uitgevers-id een unieke identificatie van de toepassing (de uitgevers-id wordt door AIR afgeleid van het certificaat waarmee het toepassingspakket word ondertekend). De toepassings-id wordt gebruikt voor installatie, toegang tot de privéopslagmap van de toepassing, toegang tot gecodeerde privéopslag en onderlinge communicatie tussen toepassingen.

4 Voeg het element `<versionNumber>` toe:

**<versionNumber>0.1</versionNumber>** Hiermee kunnen gebruikers bepalen welke versie van de toepassing zijn installeren.

***Opmerking:*** *als u AIR 2 of eerder gebruikt, moet u het element `<version>` gebruiken in plaats van het element `<versionNumber>`.*

5 Voeg het element `<filename>` toe:

**<filename>HelloWorld</filename>** De naam die wordt gebruikt voor het uitvoerbare bestand en de installatiemap van de toepassing, en voor andere verwijzingen naar de toepassing in het besturingssysteem.

6 Voeg het element `<initialWindow>` toe met daarin de volgende onderliggende elementen om de eigenschappen voor het eerste toepassingsvenster te definiëren:

**<content>HelloWorld.html</content>** Hiermee wordt het basisbestand met HTML-inhoud gedefinieerd dat in AIR wordt geladen.

**<visible>true</visible>** Hiermee wordt ingesteld dat het venster meteen zichtbaar is.

**<width>400</width>** Hiermee wordt de breedte van het venster (in pixels) ingesteld.

**<height>200</height>** Hiermee wordt de hoogte van het venster ingesteld.

7 Sla het bestand op. Het voltooide descriptorbestand van de toepassing moet er als volgt uitzien:

```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://ns.adobe.com/air/application/2.7">
    <id>examples.html.HelloWorld</id>
    <versionNumber>0.1</versionNumber>
    <filename>HelloWorld</filename>
    <initialWindow>
        <content>HelloWorld.html</content>
        <visible>true</visible>
        <width>400</width>
        <height>200</height>
    </initialWindow>
</application>
```

In dit voorbeeld wordt slechts een aantal van de mogelijke toepassingseigenschappen ingesteld. Als u een overzicht wilt van de volledige serie toepassingseigenschappen, waarmee u dingen als vensterinterface, vensterformaat, transparantie, standaardmap voor installatie, gekoppelde bestandstypen en toepassingspictogrammen kunt instellen, raadpleegt u "AIR-toepassingsdescriptorbestanden" op pagina 215.

## De HTML-pagina van de toepassing maken

Nu moet u een eenvoudige HTML-pagina maken die het hoofdbestand van de AIR-toepassing vormt.

1 Open het bestand `HelloWorld.html` om te bewerken. Voeg de volgende HTML-code toe:

```
<html>
<head>
    <title>Hello World</title>
</head>
<body onLoad="appLoad()">
    <h1>Hello World</h1>
</body>
</html>
```

2 Importeer in de sectie `<head>` van de HTML het bestand `AIRAliases.js`:

```
<script src="AIRAliases.js" type="text/javascript"></script>
```

AIR definieert een eigenschap met de naam `runtime` voor het HTML-vensterobject. De eigenschap runtime biedt toegang tot de ingebouwde AIR-klassen, waarbij gebruik wordt gemaakt van de volledig gekwalificeerde pakketnaam van de klasse. Als u bijvoorbeeld een object File van AIR wilt maken, kunt u de volgende instructie in JavaScript toevoegen:

```
var textFile = new runtime.flash.filesystem.File("app:/textfile.txt");
```

Het bestand `AIRAliases.js` definieert handige aliassen voor de API's van AIR die het meest worden gebruikt. Met `AIRAliases.js` kunt u de verwijzing naar de klasse File als volgt afkorten:

```
var textFile = new air.File("app:/textfile.txt");
```

3 Voeg onder de scripttag AIRAliases een scripttag toe die een JavaScript-functie bevat voor het verwerken van de gebeurtenis `onLoad`:

```
<script type="text/javascript">
function appLoad(){
    air.trace("Hello World");
}
</script>
```

De functie `appLoad()` roept de functie `air.trace()` aan. Wanneer de toepassing met ADL wordt uitgevoerd, wordt het traceerbericht afgedrukt naar de opdrachtconsole. De instructie trace kan bijzonder handig zijn bij foutopsporing.

**4** Sla het bestand op.

Het bestand `HelloWorld.html` moet er als volgt uitzien:

```
<html>
<head>
    <title>Hello World</title>
    <script type="text/javascript" src="AIRAliases.js"></script>
    <script type="text/javascript">
        function appLoad(){
            air.trace("Hello World");
        }
    </script>
</head>
<body onLoad="appLoad()">
    <h1>Hello World</h1>
</body>
</html>
```

## De toepassing testen

Met het hulpprogramma ADL (AIR Dedug Launcher) kunt u de toepassing vanaf de opdrachtregel uitvoeren en testen. Het uitvoerbare bestand van ADL vindt u in de map `bin` van de SDK van AIR. Als u de SDK voor AIR nog niet hebt geïnstalleerd, gaat u naar "De SDK van AIR installeren" op pagina 17.

**1** Open een opdrachtregelconsole of shell. Ga naar de map die u voor dit project hebt gemaakt.

**2** Voer de volgende opdracht uit:

```
adl HelloWorld-app.xml
```

Er verschijnt een AIR-venster met daarin de toepassing. Daarnaast wordt in het venster het bericht weergegeven dat het resultaat is van het aanroepen van `air.trace()`.

Zie voor meer informatie "AIR-toepassingsdescriptorbestanden" op pagina 215.

## Het AIR-installatiebestand maken

Wanneer de toepassing goed uitgevoerd kan worden, kunt u het met hulpprogramma ADT de toepassing in een AIR-installatiebestand plaatsen. Een AIR-installatiebestand is een archiefbestand met alle toepassingsbestanden dat naar gebruikers gedistribueerd kan worden. Een AIR-bestand kan alleen worden geïnstalleerd als Adobe AIR is geïnstalleerd.

Voor de beveiliging van de toepassing is het noodzakelijk dat alle AIR-installatiebestanden digitaal worden ondertekend. U kunt met ADT of een ander certificaatprogramma een eenvoudig, niet-geautoriseerd certificaat genereren. Het is ook mogelijk om een commercieel met code ondertekend certificaat te kopen bij een geautoriseerde certificaatinstantie, bijvoorbeeld VeriSign of Thawte. Wanneer een AIR-bestand met een niet-geautoriseerd certificaat wordt geïnstalleerd, wordt de uitgever als onbekend weergegeven tijdens het installatieproces. De reden hiervoor is dat een niet-geautoriseerd certificaat alleen de garantie biedt dat het AIR-bestand sinds het maken niet is gewijzigd. Het zou kunnen gebeuren dat iemand een ander AIR-bestand met een niet-geautoriseerd certificaat ondertekent en doet voorkomen alsof dit uw toepassing is. Voor AIR-bestanden die openbaar worden gemaakt wordt een commercieel certificaat sterk aanbevolen. Zie Beveiliging in AIR (voor ActionScript-ontwikkelaars) of AIR security (voor HTML-ontwikkelaars) voor een overzicht van beveiligingsproblemen in AIR.

**Een zelfondertekend certificaat en sleutelpaar genereren**

❖ Voer de volgende opdracht vanaf de opdrachtregel in (het uitvoerbare bestand van ADT staat in de map `bin` van de SDK van AIR):

```
adt -certificate -cn SelfSigned 1024-RSA sampleCert.pfx samplePassword
```

ADT genereert een keystore-bestand met de naam *sampleCert.pfx* met daarin een certificaat en de gerelateerde privésleutel.

In dit voorbeeld wordt het kleinste aantal kenmerken gebruikt dat voor een certificaat kan worden ingesteld. Het sleuteltype moet *1024-RSA* of *2048-RSA* zijn (zie "AIR-toepassingen ondertekenen" op pagina 199).

**Het AIR-installatiebestand maken**

❖ Voer de volgende opdracht (één regel) vanaf de opdrachtregel in:

```
adt -package -storetype pkcs12 -keystore sampleCert.pfx HelloWorld.air
HelloWorld-app.xml HelloWorld.html AIRAliases.js
```

Er wordt gevraagd om een wachtwoord voor het keystore-bestand.

Het argument HelloWorld.air is het AIR-bestand dat door ADT wordt geproduceerd. HelloWorld-app.xml is het descriptorbestand van de toepassing. De volgende argumenten zijn de bestanden die door de toepassing worden gebruikt. In dit voorbeeld worden slechts twee bestanden gebruikt, maar u kunt elk gewenst aantal bestanden en mappen opnemen. ADT controleert of het hoofdinhoudsbestand HelloWorld.html is opgenomen in het pakket, maar als u vergeet AIRAliases.js op te nemen, werkt uw toepassing niet.

Wanneer het AIR-pakket is gemaakt, kan de toepassing worden geïnstalleerd en uitgevoerd door te dubbelklikken op het pakketbestand. Het is ook mogelijk om de naam van het AIR-bestand als opdracht in een shell of opdrachtvenster te typen.

## Volgende stappen

In AIR gedragen HTML-code en JavaScript-code zich meestal net zoals in de meeste browsers. (Het is zelfs zo dat AIR dezelfde WebKit-weergave-engine gebruikt als de Safari-webbrowser.) Er is echter wel een aantal belangrijke verschillen waarmee u rekening moet houden wanneer u HTML-toepassingen in AIR ontwikkelt. Zie Programming HTML and JavaScript voor meer informatie over deze verschillen en andere belangrijke onderwerpen.

# Uw eerste AIR-bureaubladtoepassing met de Flex SDK maken

Voor een snelle en praktische illustratie van de functionaliteit van Adobe® AIR® volgt u deze instructies en maakt u de eenvoudige AIR-toepassing 'Hello World' op basis van SWF met de SDK van Flex. In deze zelfstudie wordt getoond hoe u een AIR-toepassing kunt compileren, testen en verpakken met de opdrachtregelhulpmiddelen van de Flex SDK (de Flex SDK bevat de AIR SDK).

Om te beginnen moet de runtime zijn geïnstalleerd en moet Adobe® Flex™ zijn ingesteld. Bij deze zelfstudie worden de *AMXMLC*- compiler, *ADL (AIR Debug Launcher)* en *ADT (AIR Developer Tool)* gebruikt. Deze programma's staan in de map `bin` van de Flex-SDK (zie "De SDK van Flex instellen" op pagina 19).

## Het descriptorbestand van de AIR-toepassing maken

In deze sectie wordt uitgelegd hoe u de toepassingsdescriptor maakt. Dit XML-bestand heeft de volgende structuur:

```
<application xmlns="...">
    <id>...</id>
    <versionNumber>...</versionNumber>
    <filename>…</filename>
    <initialWindow>
        <content>…</content>
        <visible>…</visible>
        <width>…</width>
        <height>…</height>
    </initialWindow>
</application>
```

1 Maak een XML-bestand genaamd `HelloWorld-app.xml` en sla het bestand op in de projectmap.

2 Voeg het element `<application>` toe, inclusief het naamruimte-attribuut voor AIR:

**<application xmlns="http://ns.adobe.com/air/application/2.7">** Het laatste segment van de naamruimte, '2.7', geeft aan welke versie van de runtime voor de toepassing is vereist.

3 Voeg het element `<id>` toe:

**<id>samples.flex.HelloWorld</id>** De toepassings-id vormt tezamen met de uitgevers-id een unieke identificatie van de toepassing (de uitgevers-id wordt door AIR afgeleid van het certificaat waarmee het toepassingspakket wordt ondertekend). De aanbevolen indeling bestaat uit een omgekeerde DNS-tekenreeks met een punt als scheidingsteken, zoals `"com.company.AppName"`. De toepassings-id wordt gebruikt voor installatie, toegang tot de privéopslagmap van de toepassing, toegang tot gecodeerde privéopslag en onderlinge communicatie tussen toepassingen.

4 Voeg het element `<versionNumber>` toe:

**<versionNumber>1.0</versionNumber>** Hiermee kunnen gebruikers bepalen welke versie van de toepassing ze installeren.

***Opmerking:*** *als u AIR 2 of eerder gebruikt, moet u het element `<version>` gebruiken in plaats van het element `<versionNumber>`.*

5 Voeg het element `<filename>` toe:

**<filename>HelloWorld</filename>** De naam die wordt gebruikt voor het uitvoerbare bestand en de installatiemap van de toepassing, en voor andere verwijzingen in het besturingssysteem.

6 Voeg het element `<initialWindow>` toe met daarin de volgende onderliggende elementen om de eigenschappen voor het eerste toepassingsvenster te definiëren:

**<content>HelloWorld.swf</content>** Hiermee wordt het basisbestand met SWF-inhoud gedefinieerd dat in AIR wordt geladen.

**<visible>true</visible>** Hiermee wordt ingesteld dat het venster meteen zichtbaar is.

**<width>400</width>** Hiermee wordt de breedte van het venster (in pixels) ingesteld.

**<height>200</height>** Hiermee wordt de hoogte van het venster ingesteld.

7 Sla het bestand op. Het volledige descriptorbestand van de toepassing ziet er als volgt uit:

```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://ns.adobe.com/air/application/2.7">
    <id>samples.flex.HelloWorld</id>
    <versionNumber>0.1</versionNumber>
    <filename>HelloWorld</filename>
    <initialWindow>
        <content>HelloWorld.swf</content>
        <visible>true</visible>
        <width>400</width>
        <height>200</height>
    </initialWindow>
</application>
```

In dit voorbeeld wordt slechts een aantal van de mogelijke toepassingseigenschappen ingesteld. Als u een overzicht wilt van de volledige serie toepassingseigenschappen, waarmee u dingen als vensterinterface, vensterformaat, transparantie, standaardmap voor installatie, gekoppelde bestandstypen en toepassingspictogrammen kunt instellen, raadpleegt u "AIR-toepassingsdescriptorbestanden" op pagina 215.

## De toepassingscode schrijven

***Opmerking:*** *AIR-toepassingen op basis van SWF maken gebruik van een hoofdklasse die wordt gedefinieerd met MXML of met Adobe® ActionScript® 3.0. In dit voorbeeld wordt de hoofdklasse gedefinieerd door een MXML-bestand. Het proces voor het maken van een AIR-toepassing met een ActionScript-hoofdklasse is vergelijkbaar. In plaats van het compileren van een MXML-bestand in het SWF-bestand, compileert u het ActionScript-klassenbestand. Wanneer u ActionScript gebruikt, moet de hoofdklasse een uitbreiding vormen voor flash.display.Sprite.*

Net als alle Flex-toepassingen bevatten AIR-toepassingen die zijn gemaakt met het Flex-raamwerk een MXML-hoofdbestand. Bij AIR-bureaubladtoepassingen wordt de *WindowedApplication*-component gebruikt als basiselement (en niet de Application-component). De WindowedApplication-component bevat eigenschappen, methoden en gebeurtenissen waarmee u de toepassing en het startvenster kunt beheren. Blijf op platforms en profielen waarvoor AIR geen meerdere vensters accepteert, de toepassingscomponent gebruiken. In mobiele Flex-toepassingen kunt u ook de onderdelen View of TabbedViewNavigatorApplication gebruiken.

De volgende procedure beschrijft hoe u de toepassing Hello World maakt:

1 Open een tekstbewerker en maak een bestand met de naam `HelloWorld.mxml`. Voeg de volgende MXML-code toe:

```
<?xml version="1.0" encoding="utf-8"?>
<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
                       xmlns:s="library://ns.adobe.com/flex/spark"
                       xmlns:mx="library://ns.adobe.com/flex/mx"
                       title="Hello World">
</s:WindowedApplication>
```

2 Hierna voegt u een Label-component toe aan de toepassing (plaats deze in de WindowedApplication-tag).

3 Stel de eigenschap `text` van de Label-component in op `"Hello AIR"`.

4 Stel de lay-out zodanig in dat de tekst altijd gecentreerd wordt weergegeven.

De code ziet er nu als volgt uit:

```
<?xml version="1.0" encoding="utf-8"?>
<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
                       xmlns:s="library://ns.adobe.com/flex/spark"
                       xmlns:mx="library://ns.adobe.com/flex/mx"
                       title="Hello World">

    <s:Label text="Hello AIR" horizontalCenter="0" verticalCenter="0"/>
</s:WindowedApplication>
```

## De toepassing compileren

Voordat u de toepassing kunt uitvoeren en de foutopsporing starten, moet u de MXML-code naar een SWF-bestand compileren met de amxmlc-compiler. De amxmlc-compiler staat in de map `bin` van de SDK van Flex. Indien gewenst kunt u de map bin van de Flex-SDK opnemen in de padomgeving van uw computer. Als u het pad instelt, kunt u de opdrachtregelprogramma's gemakkelijker uitvoeren.

1 Open een opdrachtshell of een terminal en ga naar de projectmap van uw AIR-toepassing.

2 Voer de volgende opdracht in:

```
amxmlc HelloWorld.mxml
```

Het uitvoeren van `amxmlc` produceert het bestand `HelloWorld.swf` met daarin de gecompileerde code van de toepassing.

***Opmerking:*** *als de toepassing niet compileert, controleert u op fouten in de syntaxis of spelling. Fouten en waarschuwingen worden weergegeven in het consolevenster waarmee de amxmlc-compiler wordt uitgevoerd.*

Zie "MXML- en ActionScript-bronbestanden voor AIR compileren" op pagina 163 voor meer informatie.

## De toepassing testen

Als u de toepassing wilt uitvoeren en testen vanaf de opdrachtregel, gebruikt u ADL (AIR Debug Launcher) om de toepassing te starten met behulp van het descriptorbestand van de toepassing. (ADL staat in de map bin van de SDK van Flex.)

❖ Typ de volgende opdracht bij de opdrachtregelprompt:

```
adl HelloWorld-app.xml
```

De resulterende AIR-toepassing ziet er ongeveer uit zoals in deze afbeelding.


Met de eigenschappen horizontalCenter en verrticalCenter van het besturingselement Label wordt de tekst in het midden van het venster geplaatst. U kunt het venster verplaatsen en groter of kleiner maken, net zoals bij andere bureaubladtoepassingen.

Zie "ADL (AIR Debug Launcher)" op pagina 167 voor meer informatie.

## Het AIR-installatiebestand maken

Wanneer de toepassing goed uitgevoerd kan worden, kunt u het met hulpprogramma ADT de toepassing in een AIR-installatiebestand plaatsen. Een AIR-installatiebestand is een archiefbestand met alle toepassingsbestanden dat naar gebruikers gedistribueerd kan worden. Een AIR-bestand kan alleen worden geïnstalleerd als Adobe AIR is geïnstalleerd.

Voor de beveiliging van de toepassing is het noodzakelijk dat alle AIR-installatiebestanden digitaal worden ondertekend. U kunt met ADT of een ander certificaatprogramma een eenvoudig, niet-geautoriseerd certificaat genereren. U kunt ook een certificaat voor ondertekenen van code kopen van een commerciële certificeringsinstantie. Wanneer een AIR-bestand met een niet-geautoriseerd certificaat wordt geïnstalleerd, wordt de uitgever als onbekend weergegeven tijdens het installatieproces. De reden hiervoor is dat een niet-geautoriseerd certificaat alleen de garantie biedt dat het AIR-bestand sinds het maken niet is gewijzigd. Het zou kunnen gebeuren dat iemand een ander AIR-bestand met een niet-geautoriseerd certificaat ondertekent en doet voorkomen alsof dit uw toepassing is. Voor AIR-bestanden die openbaar worden gemaakt wordt een commercieel certificaat sterk aanbevolen. Zie Beveiliging in AIR (voor ActionScript-ontwikkelaars) of AIR security (voor HTML-ontwikkelaars) voor een overzicht van beveiligingsproblemen in AIR.

### Een zelfondertekend certificaat en sleutelpaar genereren

❖ Typ de volgende opdracht op de opdrachtregel (het uitvoerbare ADT-bestand staat in de map `bin` van de Flex-SDK):

```
adt -certificate -cn SelfSigned 1024-RSA sampleCert.pfx samplePassword
```

In dit voorbeeld wordt het kleinste aantal kenmerken gebruikt dat voor een certificaat kan worden ingesteld. Het sleuteltype moet *1024-RSA* of *2048-RSA* zijn (zie “AIR-toepassingen ondertekenen” op pagina 199).

### Het AIR-pakket maken

❖ Voer de volgende opdracht (één regel) vanaf de opdrachtregel in:

```
adt -package -storetype pkcs12 -keystore sampleCert.pfx HelloWorld.air
HelloWorld-app.xml HelloWorld.swf
```

Er wordt gevraagd om een wachtwoord voor het keystore-bestand. Typ het wachtwoord en druk op Enter. Het wachtwoord wordt niet weergegeven om beveiligingsredenen.

Het argument HelloWorld.air is het AIR-bestand dat door ADT wordt geproduceerd. HelloWorld-app.xml is het descriptorbestand van de toepassing. De volgende argumenten zijn de bestanden die door de toepassing worden gebruikt. In dit voorbeeld worden slechts drie bestanden gebruikt, maar u kunt elk gewenst aantal bestanden en mappen opnemen.

Wanneer het AIR-pakket is gemaakt, kan de toepassing worden geïnstalleerd en uitgevoerd door te dubbelklikken op het pakketbestand. Het is ook mogelijk om de naam van het AIR-bestand als opdracht in een shell of opdrachtvenster te typen.

Zie “Een AIR-bureaubladinstallatiebestand verpakken” op pagina 56 voor meer informatie.

# Voor het eerst een AIR-toepassing voor Android maken met de SDK van Flex

Om te beginnen moet u de SDK's van AIR en Flex hebben geïnstalleerd. In deze zelfstudie worden de *AMXMLC*-compiler uit de SDK en *AIR Debug Launcher* (ADL), en de *AIR Developer Tool* (ADT) van de SDK van AIR gebruikt. Zie "De SDK van Flex instellen" op pagina 19

U moet ook de SDK van Android downloaden van de Android-website en installeren, zoals beschreven in: Android Developers: Installing the SDK.

***Opmerking:*** *zie voor informatie over iPhone-ontwikkelingen Een Hello World-toepassing voor de iPhone maken met Flash Professional CS5.*

## Het descriptorbestand van de AIR-toepassing maken

In deze sectie wordt uitgelegd hoe u de toepassingsdescriptor maakt. Dit XML-bestand heeft de volgende structuur:

```
<application xmlns="...">
    <id>...</id>
    <versionNumber>...</versionNumber>
    <filename>…</filename>
    <initialWindow>
        <content>…</content>
    </initialWindow>
    <supportedProfiles>...</supportedProfiles>
</application>
```

1. Maak een XML-bestand genaamd `HelloWorld-app.xml` en sla het bestand op in de projectmap.
2. Voeg het element `<application>` toe, inclusief het naamruimte-attribuut voor AIR:

   **<application xmlns="http://ns.adobe.com/air/application/2.7">** Het laatste segment van de naamruimte, '2.7', geeft aan welke versie van de runtime voor de toepassing is vereist.
3. Voeg het element `<id>` toe:

   **<id>samples.flex.HelloWorld</id>** De toepassings-id vormt tezamen met de uitgevers-id een unieke identificatie van de toepassing (de uitgevers-id wordt door AIR afgeleid van het certificaat waarmee het toepassingspakket wordt ondertekend). De aanbevolen indeling bestaat uit een omgekeerde DNS-tekenreeks met een punt als scheidingsteken, zoals `"com.company.AppName"`.
4. Voeg het element `<versionNumber>` toe:

   **<versionNumber>0.0.1</versionNumber>** Hiermee kunnen gebruikers bepalen welke versie van de toepassing wordt geïnstalleerd.
5. Voeg het element `<filename>` toe:

   **<filename>HelloWorld</filename>** De naam die wordt gebruikt voor het uitvoerbare bestand en de installatiemap van de toepassing, en voor andere verwijzingen in het besturingssysteem.
6. Voeg het element `<initialWindow>` toe met daarin de volgende onderliggende elementen om de eigenschappen voor het eerste toepassingsvenster te definiëren:

   **<content>HelloWorld.swf</content>** Hiermee wordt het basisbestand met HTML-inhoud gedefinieerd dat in AIR wordt geladen.
7. Voeg het element `<supportedProfiles>` toe.

**<supportedProfiles>mobileDevice</supportedProfiles>** Geeft op dat de toepassing alleen wordt uitgevoerd als in het mobiele profiel.

**8** Sla het bestand op. Het volledige descriptorbestand van de toepassing ziet er als volgt uit:

```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://ns.adobe.com/air/application/2.7">
    <id>samples.android.HelloWorld</id>
    <versionNumber>0.0.1</versionNumber>
    <filename>HelloWorld</filename>
    <initialWindow>
        <content>HelloWorld.swf</content>
    </initialWindow>
    <supportedProfiles>mobileDevice</supportedProfiles>
</application>
```

In dit voorbeeld wordt slechts een aantal van de mogelijke toepassingseigenschappen ingesteld. Er zijn andere instellingen die u kunt gebruiken in het toepassingsdescriptorbestand. U kunt bijvoorbeeld <fullScreen>true</fullScreen> toevoegen aan het element initialWindow om een toepassing voor volledig scherm te maken. Als u foutopsporing op afstand en functies met toegangsbeheer op Android wilt inschakelen, moet u ook Android-machtigingen toevoegen aan de toepassingsdescriptor. Er zijn geen machtigingen nodig voor deze eenvoudige toepassing, dus deze hoeft u nu niet toe te voegen.

Zie “Eigenschappen van mobiele toepassingen instellen” op pagina 75 voor meer informatie.

## De toepassingscode schrijven

Maak een bestand met de naam HelloWorld.as en voeg met behulp van een teksteditor de volgende code toe:

```
package
{
    import flash.display.Sprite;
    import flash.text.TextField;

    public class HelloWorld extends Sprite
    {
        public function HelloWorld()
        {
            var textField:TextField = new TextField();
            textField.text = "Hello, World!";
            stage.addChild( textField );
        }
    }
}
```

## De toepassing compileren

Voordat u de toepassing kunt uitvoeren en de foutopsporing starten, moet u de MXML-code naar een SWF-bestand compileren met de amxmlc-compiler. De amxmlc-compiler staat in de map `bin` van de SDK van Flex. Indien gewenst kunt u de map bin van de Flex-SDK opnemen in de padomgeving van uw computer. Als u het pad instelt, kunt u de opdrachtregelprogramma's gemakkelijker uitvoeren.

**1** Open een opdrachtshell of een terminal en ga naar de projectmap van uw AIR-toepassing.

**2** Voer de volgende opdracht in:

```
amxmlc HelloWorld.as
```

Het uitvoeren van `amxmlc` produceert het bestand `HelloWorld.swf` met daarin de gecompileerde code van de toepassing.

***Opmerking:*** *als de toepassing niet compileert, controleert u op fouten in de syntaxis of spelling. Fouten en waarschuwingen worden weergegeven in het consolevenster waarmee de amxmlc-compiler wordt uitgevoerd.*

Zie "MXML- en ActionScript-bronbestanden voor AIR compileren" op pagina 163 voor meer informatie.

## De toepassing testen

Als u de toepassing wilt uitvoeren en testen vanaf de opdrachtregel, gebruikt u ADL (AIR Debug Launcher) om de toepassing te starten met behulp van het descriptorbestand van de toepassing. (ADL staat in de map bin van de SDK's van AIR en Flex.)

❖ Typ de volgende opdracht bij de opdrachtregelprompt:

```
adl HelloWorld-app.xml
```

Zie voor meer informatie "Apparaatsimulatie met ADL" op pagina 106.

## Het APK-pakketbestand maken

Wanneer de toepassing goed uitgevoerd kan worden, kunt u het met hulpprogramma ADT de toepassing in een AIR-installatiebestand verpakken. Een APK-pakketbestand is de native bestandsindeling van de Android-toepassing die u onder uw gebruikers kunt distribueren.

Alle Android-toepassingen moeten worden ondertekend. In tegenstelling tot AIR-bestanden is het gebruikelijk om Android-toepassingen te ondertekenen met een zelfondertekend certificaat. Het Android-besturingssysteem doet geen poging de identiteit van de ontwikkelaar van de toepassing te achterhalen. U kunt een door ADT gegenereerd certificaat gebruiken om Android-pakketten te ondertekenen. Certificaten voor toepassingen die worden ingediend bij de Android Market, moeten een geldigheidsperiode van ten minste 25 jaar hebben.

### Een zelfondertekend certificaat en sleutelpaar genereren

❖ Typ de volgende opdracht op de opdrachtregel (het uitvoerbare ADT-bestand staat in de map `bin` van de Flex-SDK):

```
adt -certificate -validityPeriod 25 -cn SelfSigned 1024-RSA sampleCert.pfx samplePassword
```

In dit voorbeeld wordt het kleinste aantal kenmerken gebruikt dat voor een certificaat kan worden ingesteld. Het sleuteltype moet ofwel *1024-RSA* of *2048-RSA* zijn (zie "ADT-opdracht voor het certificaat" op pagina 182).

### Het AIR-pakket maken

❖ Voer de volgende opdracht (één regel) vanaf de opdrachtregel in:

```
adt -package -target apk -storetype pkcs12 -keystore sampleCert.p12 HelloWorld.apk
HelloWorld-app.xml HelloWorld.swf
```

Er wordt gevraagd om een wachtwoord voor het keystore-bestand. Typ het wachtwoord en druk op Enter.

Zie "Een mobiele AIR-toepassing verpakken" op pagina 98 voor meer informatie.

### De AIR-runtime installeren

U kunt de laatste versie van de AIR-runtime van de Android Market op uw apparaat installeren. U kunt ook de runtime in uw SDK installeren op een apparaat of een Android-emulator.

❖ Voer de volgende opdracht (één regel) vanaf de opdrachtregel in:

```
adt -installRuntime -platform android -platformsdk
```

Stel de markering `-platformsdk` in op uw Android SDK-map (geef de bovenliggende map van de map voor hulpmiddelen op).

ADT installeert de Runtime.apk die in de SDK is opgenomen.

Zie "De AIR-runtime en -toepassingen installeren voor ontwikkeling" op pagina 115 voor meer informatie.

**De AIR-toepassing installeren**

❖ Voer de volgende opdracht (één regel) vanaf de opdrachtregel in:

```
adt -installApp -platform android -platformsdk path-to-android-sdk -package path-to-app
```

Stel de markering `-platformsdk` in op uw Android SDK-map (geef de bovenliggende map van de map voor hulpmiddelen op).

Zie voor meer informatie "De AIR-runtime en -toepassingen installeren voor ontwikkeling" op pagina 115.

U kunt uw toepassing starten door op het pictogram voor de toepassing te tikken op het scherm van het apparaat of de emulator.

# Hoofdstuk 6: AIR-toepassingen voor het bureaublad ontwikkelen

## Workflow voor het ontwikkelen van een AIR-bureaubladtoepassing

De basisworkflow voor het ontwikkelen van een AIR-toepassing is dezelfde als de meeste traditionele modellen: code, compilatie, test en, tegen het einde van de cyclus, verpakking in een installatiebestand.

U kunt de toepassingscode met Flash, Flex en ActionScript schrijven en compileren met behulp van Flash Professional, Flash Builder of de opdrachtsregelcompilers mxmlc en compc. U kunt de toepassingscode ook schrijven met behulp van HTML en JavaScript en de compilatiestap overslaan.

U kunt AIR-bureaubladtoepassingen testen met het hulpprogramma ADL, dat een toepassing uitvoert zonder dat deze verpakt en geïnstalleerd hoeft te zijn. Flash Professional, Flash Builder, Dreamweaver en de Aptana IDE integreren allemaal met Flash Debugger. U kunt het hulpprogramma voor foutopsporing, FDB, ook handmatig uitvoeren wanneer u ADL vanaf de opdrachtregel gebruikt. ADL geeft zelf fouten en traceringsinstructie-uitvoer weer.

Alle AIR-toepassingen moeten worden verpakt in een installatiebestand. De AIR-bestandsindeling voor verschillende platforms wordt aanbevolen, tenzij:

- U toegang nodig hebt tot platformonafhankelijke API's, zoals de NativeProcess-klasse.
- Uw toepassing native extensies gebruikt.

In zo'n geval kunt u een AIR-toepassing verpakken als een platformspecifiek, eigen installatiebestand.

### Toepassingen op basis van SWF

1. Schrijf de MXML- of ActionScript-code.
2. Maak benodigde assets, zoals bitmapbestanden voor pictogrammen.
3. Maak de toepassingsdescriptor.
4. Stel ActionScript-code samen.
5. Test de toepassing.
6. Maak een pakket en onderteken het als een AIR-bestand met het doel *target*.

### Toepassingen op basis van HTML

1. Schrijf de HTML- en JavaScript-code.
2. Maak benodigde assets, zoals bitmapbestanden voor pictogrammen.
3. Maak de toepassingsdescriptor.
4. Test de toepassing.
5. Maak een pakket en onderteken het als een AIR-bestand met het doel *target*.

## Eigen installatieprogramma's maken voor AIR-toepassingen

1. Schrijf de code (ActionScript of HTML en JavaScript).
2. Maak benodigde assets, zoals bitmapbestanden voor pictogrammen.
3. Maak het descriptorbestand van de toepassing, waarbij u het profiel *extendedDesktop* opgeeft.
4. Stel ActionScript-code samen.
5. Test de toepassing.
6. Maak een toepassingspakket op elk doelplatform met gebruik van het doel *native*.

***Opmerking:*** *het native installatieprogramma voor een doelplatform dient op dat platform te worden gemaakt. Het is bijvoorbeeld niet mogelijk een Windows-installatieprogramma te maken op een Mac. Gebruik een virtuele computer, zoals VMWare, om meerdere platforms uit te voeren op dezelfde computerhardware.*

## AIR-toepassingen maken met een captive-runtimebundel

1. Schrijf de code (ActionScript of HTML en JavaScript).
2. Maak benodigde assets, zoals bitmapbestanden voor pictogrammen.
3. Maak het descriptorbestand van de toepassing, waarbij u het profiel *extendedDesktop* opgeeft.
4. Stel ActionScript-code samen.
5. Test de toepassing.
6. Maak een toepassingspakket op elk doelplatform met gebruik van het doel *bundle*.
7. Maak een installatieprogramma met gebruik van de bundelbestanden. (De SDK van AIR biedt geen gereedschappen voor het maken van zo'n installatieprogramma, maar er zijn vele toolkits van andere bedrijven beschikbaar.)

***Opmerking:*** *de bundel voor een doelplatform dient op dat platform te worden gemaakt. Het is bijvoorbeeld niet mogelijk een Windows-bundel te maken op een Mac. Gebruik een virtuele computer, zoals VMWare, om meerdere platforms uit te voeren op dezelfde computerhardware.*

# Eigenschappen bureaubladtoepassing instellen

Stel de basiseigenschappen voor toepassingen in het toepassingsdescriptorbestand in. In dit deel worden de relevante eigenschappen voor AIR-bureaubladtoepassingen beschreven. De elementen van het toepassingsdescriptorbestand worden volledig beschreven in "AIR-toepassingsdescriptorbestanden" op pagina 215.

## Vereiste AIR-runtimeversie

Geef de versie op van de AIR-runtime die wordt vereist door uw toepassing en gebruik daarbij de naamruimte van het toepassingsdescriptorbestand.

De naamruimte, die wordt toegewezen in het `application`-element, bepaalt voor een groot deel welke functies uw toepassing kan gebruiken. Als uw toepassing bijvoorbeeld de naamruimte AIR 1.5 gebruikt en de gebruiker heeft AIR 3.0 geïnstalleerd, dan ziet uw toepassing de functionaliteit van AIR 1.5 (zelfs als de functionaliteit is gewijzigd in AIR 3.0). Uw toepassing heeft uitsluitend toegang tot de nieuwe functionaliteit en functies als u de naamruimte wijzigt en een update publiceert. Beveiligings- en WebKit-wijzigingen zijn de voornaamste uitzonderingen op dit beleid.

Geef de naamruimte op met behulp van het xmlns-attribuut van het basiselement `application`:

```
<application xmlns="http://ns.adobe.com/air/application/3.0">
```

### Meer Help-onderwerpen

"application" op pagina 220

## Toepassingsidentiteit

Verschillende instellingen moeten uniek zijn voor elke toepassing die u publiceert. Voorbeelden van de unieke instellingen zijn de id, de naam en de bestandsnaam.

```
<id>com.example.MyApplication</id>
<name>My Application</name>
<filename>MyApplication</filename>
```

### Meer Help-onderwerpen

"id" op pagina 237

"filename" op pagina 232

"name" op pagina 245

## Toepassingsversie

Geef in eerdere versies van AIR dan AIR 2.5 de toepassing op in het element `version`. U kunt een willekeurige tekenreeks gebruiken. De AIR-runtime interpreteert de tekenreeks niet. "2.0" wordt niet behandeld als een hogere versie dan "1.0".

```
<!-- AIR 2 or earlier -->
<version>1.23 Beta 7</version>
```

In AIR 2.5 en later geeft u de toepassingsversie op in het element `versionNumber`. Het element `version` mag niet langer worden gebruikt. Wanneer u een waarde opgeeft voor `versionNumber`, moet u een reeks opgeven van maximaal drie cijfers gescheiden door punten, zoals: "0.1.2". Elk segment van het versienummer kan hoogstens drie cijfers hebben. (Met andere woorden: "999.999.999" is het hoogste versienummer dat is toegestaan.) U hoeft niet alle drie de segmenten in het cijfer op te nemen. "1" en "1.0" zijn ook geldige versienummers.

U kunt ook een label voor de versie opgeven met behulp van het element `versionLabel`. Wanneer u een versielabel toevoegt, wordt deze weergegeven in plaats van het versienummer op plaatsen als het dialoogvenster van het AIR-toepassingsinstallatieprogramma.

```
<!-- AIR 2.5 and later -->
<versionNumber>1.23.7<versionNumber>
<versionLabel>1.23 Beta 7</versionLabel>
```

### Meer Help-onderwerpen

"version" op pagina 254

"versionLabel" op pagina 254

"versionNumber" op pagina 254

## Eigenschappen hoofdvenster

Wanneer AIR een toepassing op het bureaublad start, wordt een venster gemaakt en wordt het SWF-hoofdbestand of de HTML-hoofdpagina daarin geladen. AIR gebruikt de onderliggende elementen van het element `initialWindow` om de eerste verschijning en functionaliteit van het eerste toepassingsvenster te beheren.

- **content** — Het SWF-hoofdbestand van de toepassing in het onderliggende element `content` van het element `initalWindow`. Als u apparaten in het bureaubladprofiel als doel instelt, kunt u een SWF- of HTML-bestand gebruikten.

  ```
  <initialWindow>
      <content>MyApplication.swf</content>
  </initialWindow>
  ```

  U moet het bestand in het AIR-pakket opnemen (met behulp van ADT of uw IDE). Door alleen te verwijzen naar de naam in de toepassingsdescriptor wordt het bestand niet automatisch in het pakket opgenomen.

- **depthAndStencil** — Of de diepte- of de stencilbuffer moet worden gebruikt. Deze buffers worden meestal gebruikt tijdens het werken met 3D-inhoud.

  ```
  <depthAndStencil>true</depthAndStencil>
  ```

- **height** — De hoogte van het eerste venster.
- **maximizable** — Of de systeeminterface voor het maximaliseren van het venster wordt weergegeven.
- **maxSize** — De maximale toegestane grootte.
- **minimizable** — Of de systeeminterface voor het minimaliseren van het venster wordt weergegeven.
- **minSize** — De minimale toegestane grootte.
- **renderMode** — in AIR 3 of later kan de rendermodus voor desktoptoepassingen worden ingesteld op *auto*, *cpu*, *direct*, of *gpu*. In eerdere AIR-versies wordt deze instelling op desktopplatforms genegeerd. De renderMode-instelling kan niet worden gewijzigd tijdens de runtime.
  - auto — min of meer gelijk aan de cpu-modus.
  - cpu — weergaveobjecten worden gerenderd en gekopieerd naar het weergavegeheugen in de software. StageVideo is alleen beschikbaar wanneer een venster op volledige schermgrootte wordt weergegeven. Stage3D gebruikt de softwarerenderer.
  - direct — weergaveobjecten worden door de runtimesoftware gerenderd, maar het kopiëren van het gerenderde frame naar het weergavegeheugen (blitting) wordt uitgevoerd aan de hand van hardwareversnelling. StageVideo is beschikbaar. Stage3D gebruikt hardwareversnelling, als dat anderszins mogelijk is. Als venstertransparantie is ingesteld op true, 'valt het venster terug' op softwarerendering en blitting.

    ***Opmerking:*** *voor een optimale GPU-versnelling van Flash-inhoud met AIR for Mobile-platforms raadt Adobe u aan om de instelling renderMode="direct" te gebruiken (dat wil zeggen: Stage3D) in plaats van renderMode="gpu". Adobe biedt officieel ondersteuning voor de volgende Stage3D-frameworks: Starling (2D) en Away3D (3D). Adobe raadt u daarom aan deze frameworks te gebruiken. Voor meer informatie over Stage3D en Starling/Away3D gaat u naar http://gaming.adobe.com/getstarted/.*

  - gpu — hardwareversnelling wordt gebruikt, indien beschikbaar.

- **requestedDisplayResolution** — Of uw toepassing de resolutiemodus *standard* of *high* moet gebruiken op MacBook Pro-computers met hoge-resolutieschermen. Op andere platforms wordt de waarde genegeerd. Als de waarde *standard* is, wordt elke werkgebiedpixel weergegeven als vier pixels op het scherm. Als de waarde *high* is, stemt elke werkgebiedpixel overeen met één fysieke pixel op het scherm. De opgegeven waarde wordt gebruikt voor alle toepassingsvensters. Het gebruik van het `requestedDisplayResolution`-element voor AIR-bureaubladtoepassingen (als een onderliggend element van het `intialWindow`-element) is beschikbaar in AIR 3.6 en later.
- **resizable** — Of de systeeminterface voor het wijzigen van de grootte van het venster wordt weergegeven.
- **systemChrome** — Of de window dressing van het standaardbesturingssysteem wordt gebruikt. De systemChrome-instelling van een venster kan niet bij uitvoering worden gewijzigd.
- **title** — De titel van het venster.
- **transparent** — Of het venster overvloeit in de achtergrond. Het venster kan de systeeminterface niet gebruiken als transparantie is ingeschakeld. De transparante instelling van een venster kan niet worden gewijzigd tijdens runtime.
- **visible** — Of het venster zichtbaar is zodra het is gemaakt. Het venster is standaard niet direct zichtbaar, zodat uw toepassing de inhoud kan tekenen voordat het zichtbaar wordt gemaakt.
- **width** — De breedte van het venster.
- **x** — De horizontale positie van het venster.
- **y** — De verticale positie van het venster.

## Meer Help-onderwerpen

"content" op pagina 226

"depthAndStencil" op pagina 228

"height" op pagina 236

"maximizable" op pagina 244

"maxSize" op pagina 244

"minimizable" op pagina 245

"minimizable" op pagina 245

"minSize" op pagina 245

"renderMode" op pagina 248

"requestedDisplayResolution" op pagina 248

"resizable" op pagina 249

"systemChrome" op pagina 252

"title" op pagina 253

"transparent" op pagina 253

"visible" op pagina 255

"width" op pagina 255

"x" op pagina 256

"y" op pagina 256

## Bureaubladfuncties

De volgende elementen beheren bureaubladinstallatie en updatefuncties.

- customUpdateUI — Hiermee kunt u uw eigen dialoogvensters leveren voor het bijwerken van een toepassing. Wanneer deze optie is ingesteld op `false`, de standaardinstelling, worden de standaard-AIR-dialoogvensters gebruikt.
- fileTypes — Geeft de bestandstypen op waarvoor u de toepassing als standaardtoepassing wilt registreren voor het openen. Als er al een andere toepassing is geïnstalleerd als standaardtoepassing voor het openen van een bestandstype, overschrijft AIR de bestaande registratie niet. Uw toepassing kan echter de registratie bij uitvoering overschrijven met behulp van de methode `setAsDefaultApplication()` van het NativeApplication-object. Het getuigt van goede manieren om toestemming van de gebruiker te vragen voordat u diens bestaande bestandstypekoppelingen overschrijft.

  ***Opmerking:*** *bestandstyperegistratie wordt genegeerd wanneer u een toepassing verpakt als een captive-runtimebundel (met gebruik van het doel `-bundle`). Om een bepaald bestandstype te registreren, dient u een installatieprogramma te maken dat de registratie uitvoert.*
- installFolder — Geeft een pad op dat afhankelijk is van de standaardtoepassingsmap waar de toepassing in is geïnstalleerd. U kunt deze instellingen gebruiken om een andere mapnaam op te geven en om verschillende toepassingen binnen een gemeenschappelijke map te groeperen.
- programMenuFolder — Geeft de menuhiërarchie op voor het Windows-menu Alle programma's. U kunt deze instellingen gebruiken om meerdere toepassingen binnen een gemeenschappelijk menu te groeperen. Als er geen menumap is opgegeven, wordt de toepassingssnelkoppeling direct aan het hoofdmenu toegevoegd.

### Meer Help-onderwerpen

"customUpdateUI" op pagina 227

"fileTypes" op pagina 234

"installFolder" op pagina 241

"programMenuFolder" op pagina 247

## Ondersteunde profielen

Als uw toepassing alleen voor het bureaublad is bedoeld, kunt u voorkomen dan deze in een ander profiel op apparaten wordt geïnstalleerd door dat profiel niet op te nemen in de lijst met ondersteunde profielen. Als uw toepassing gebruikmaakt van de klasse NativeProcess of van native extensies, moet u het profiel `extendedDesktop` ondersteunen.

Als u het element `supportedProfile` uit de toepassingsdescriptor laat, wordt aangenomen dat uw toepassing alle gedefinieerde profielen ondersteunt. Als u uw toepassing wilt beperken tot een bepaalde lijst met profielen, geeft u een lijst van de profielen op, gescheiden door spaties:

```
<supportedProfiles>desktop extendedDesktop</supportedProfiles>
```

Voor een lijst met ActionScript-klassen die worden ondersteund in de profielen `desktop` en `extendedDesktop`: zie "Mogelijkheden van de verschillende profielen" op pagina 258.

### Meer Help-onderwerpen

"supportedProfiles" op pagina 251

## Vereiste native extensies

Toepassingen die het profiel `extendedDesktop` ondersteunen, kunnen native extensies gebruiken.

Declareer alle native extensies die de AIR-toepassing gebruikt in het descriptorbestand van de toepassing. Het volgende voorbeeld illustreert de syntaxis voor het opgeven van twee vereiste native extensies:

```
<extensions>
    <extensionID>com.example.extendedFeature</extensionID>
    <extensionID>com.example.anotherFeature</extensionID>
</extensions>
```

Het `extensionID`-element heeft dezelfde waarde als het element `id` in het descriptorbestand van de extensie. Het descriptorbestand van de extensie is een XML-bestand met de naam "extension.xml". Het wordt verpakt in het ANE-bestand dat u ontvangt van de ontwikkelaar van de native extensie.

## Toepassingspictogrammen

Op het bureaublad worden de in de toepassingsdescriptor opgegeven pictogrammen gebruikt als de pictogrammen voor toepassingsbestand, snelkoppeling en programmenu. Het toepassingspictogram moet worden aangeleverd als set van PNG-afbeeldingen van 16 x 16, 32 x 32, 48 x 48 en 128 x 128 pixels. Geef het pad naar de pictogrambestanden op in het pictogramelement van het descriptorbestand van de toepassing:

```
<icon>
    <image16x16>assets/icon16.png</image16x16>
    <image32x32>assets/icon32.png</image32x32>
    <image48x48>assets/icon48.png</image48x48>
    <image128x128>assets/icon128.png</image128x128>
</icon>
```

Als u geen pictogram van een bepaalde grootte aanlevert, wordt de volgende grootte gebruikt en wordt de afbeelding passend gemaakt. Als u geen pictogrammen aanlevert, wordt een standaardsysteempictogram gebruikt.

### Meer Help-onderwerpen

"icon" op pagina 237

"imageNxN" op pagina 238

## Genegeerde instellingen

Toepassingen op het bureaublad negeren toepassingsinstellingen die van toepassingen zijn op de functies voor het profiel voor mobiele apparaten. De genegeerde instellingen zijn:

- android
- aspectRatio
- autoOrients
- fullScreen
- iPhone
- renderMode (voor AIR 3)

- requestedDisplayResolution
- softKeyboardBehavior

# Fouten opsporen bij AIR-bureaubladtoepassing

Als u uw toepassing ontwikkelt met een IDE als Flash Builder, Flash Professional of Dreamweaver, zijn foutopsporingsprogramma's normaal gesproken ingebouwd. U kunt fouten in uw toepassing eenvoudig opsporen door deze in foutopsporingsmodus te starten. Als u geen IDE gebruikt die direct foutopsporing ondersteunt, kunt u de ADL (AIR Debug Launcher) en Flash Debugger (FDB) gebruiken voor hulp bij het opsporen van fouten in uw toepassing.

### Meer Help-onderwerpen

De Monsters: Monster-foutopsporing

"Fouten opsporen met de AIR HTML Introspector" op pagina 295

## Een toepassing uitvoeren met ADL

U kunt met behulp van ADL een AIR-toepassing uitvoeren zonder dat het hoeft te worden verpakt en geïnstalleerd. Open het toepassingsdescriptorbestand als parameter in ADL, zoals in het volgende voorbeeld (ActionScript-code in de toepassing moet eerst worden gecompileerd):

```
adl myApplication-app.xml
```

ADL drukt traceringsinstructies, runtime-uitzonderingen en HTML-parseerfouten af naar het terminalvenster. Als een FDB-proces op een binnenkomende verbinding wacht, maakt ADL verbinding met het foutopsporingsprogramma.

U kunt ADL ook gebruiken om fouten op te sporen in een AIR-toepassing die native extensies gebruikt. Bijvoorbeeld:

```
adl -extdir extensionDirs myApplication-app.xml
```

### Meer Help-onderwerpen

"ADL (AIR Debug Launcher)" op pagina 167

## Traceerinstructies afdrukken

Als u traceerinstructies wilt afdrukken naar de console waarmee ADL wordt uitgevoerd, voegt u traceerinstructies aan de code toe met behulp van de functie `trace()`.

***Opmerking:*** *als uw `trace()`-instructies niet worden weergegeven op de console, controleert u of u `ErrorReportingEnable` of `TraceOutputFileEnable` niet hebt opgegeven in het mm.cfg-bestand. Raadpleeg Het mm.cfg-bestand bewerken voor meer informatie over de platformspecifieke locatie van dit bestand.*

ActionScript-voorbeeld:

```
//ActionScript
trace("debug message");
```

JavaScript-voorbeeld:

```
//JavaScript
air.trace("debug message");
```

In JavaScript-code kunt u met de functies `alert()` en `confirm()` foutopsporingsberichten uit de toepassing weergeven. Bovendien worden de regelnummers voor syntaxisfouten en alle niet-onderschepte JavaScript-uitzonderingen afgedrukt naar de console.

***Opmerking:*** *als u het voorvoegsel air uit het JavaScript-voorbeeld wilt gebruiken, moet u het bestand AIRAliases.js in de pagina importeren. Dit bestand vindt u in de map frameworks van de SDK van AIR.*

## Verbinding maken met FDB (Flash Debugger)

Als u fouten in AIR-toepassingen wilt opsporen met de Flash Debugger, start u een FDB-sessie en start u de toepassing vervolgens met ADL.

***Opmerking:*** *in AIR-toepassingen op SWF-basis moeten de ActionScript-bronbestanden worden gecompileerd met de opmerking `-debug`. (Schakel in Flash Professional de optie voor het toestaan van foutopsporing in het dialoogvenster met publicatie-instellingen in.)*

1. Start FDB. U kunt het FDB-programma vinden in de map `bin` van de Flex SDK.

   De FDB-prompt wordt weergegeven op de console: `<fdb>`

2. Voer de opdracht `run` uit. `<fdb>run [Enter]`
3. Start een foutopsporingsversie van uw toepassing in een andere opdracht- of shellconsole:

   ```
   adl myApp.xml
   ```

4. Stel de gewenste onderbrekingspunten in met de FDB-opdrachten.
5. Typ: `continue [Enter]`

Als een AIR-toepassing op SWF gebaseerd is, bestuurt het foutopsporingsprogramma alleen de uitvoering van ActionScript-code. Als een AIR-toepassing op HTML gebaseerd is, bestuurt het foutopsporingsprogramma alleen de uitvoering van JavaScript-code.

Als u ADL wilt uitvoeren zonder verbinding te maken met het foutopsporingsprogramma, neemt u de optie `-nodebug` op:

```
adl myApp.xml -nodebug
```

Voor basisinformatie over FDB-opdrachten voert u de opdracht `help` uit:

```
<fdb>help [Enter]
```

Meer informatie over de FDB-opdrachten vindt u in Using the command-line debugger commands in de Flex-documentatie.

# Een AIR-bureaubladinstallatiebestand verpakken

Elke AIR-toepassing moet minimaal een descriptorbestand van de toepassing en een SWF- of HTML-hoofdbestand bevatten. Eventuele overige elementen die bij de toepassing geïnstalleerd moeten worden moeten ook in het AIR-bestand worden verpakt.

In dit artikel wordt het verpakken van een AIR-toepassing met gebruik van de opdrachtsregelhulpmiddelen binnen de SDK besproken. Zie voor informatie over het verpakken van een toepassing met een van de ontwerpgereedschappen van Adobe de volgende pagina's:

- Adobe® Flex® Builder™, zie Packaging AIR applications with Flex Builder.
- Adobe® Flash® Builder™, zie Packaging AIR applications with Flash Builder.

- Adobe® Flash® Professional, zie Publiceren voor Adobe AIR.
- Adobe® Dreamweaver®, zie Een AIR-toepassing maken in Dreamweaver.

Alle AIR-installatiebestanden moeten zijn ondertekend met een digitaal certificaat. Het AIR-installatieprogramma gebruikt de handtekening om te verifiëren of uw toepassingsbestand niet is gewijzigd sinds u het hebt ondertekend. U kunt een certificaat voor ondertekening van programmacode van een certificeringsinstantie of een zelfondertekend certificaat gebruiken.

Wanneer u een certificaat gebruikt dat door een vertrouwde certificeringsinstantie is uitgegeven, hebben gebruikers van uw toepassing enige garantie dat u de uitgever bent. Het installatiedialoogvenster geeft aan dat uw identiteit is gecontroleerd door de certificeringsinstantie:


*Installatiedialoogvenster voor bevestiging voor toepassing die met een vertrouwd certificaat is ondertekend*

Wanneer u een zelfondertekend certificaat gebruikt, kunnen gebruikers uw identiteit als ondertekenaar niet controleren. Een zelfondertekend certificaat vermindert ook de zekerheid dat het pakket ongewijzigd is. (De reden hiervoor is dat een geldig installatieprogramma vervangen zou kunnen zijn door een vervalsing voordat het de gebruiker bereikt.) Het installatiedialoogvenster weerspiegelt het feit dat de identiteit van de uitgever niet kan worden gecontroleerd. Gebruikers nemen een groter beveiligingsrisico wanneer zij uw toepassing installeren:


*Installatiedialoogvenster voor bevestiging voor toepassing die met een zelfondertekend certificaat is ondertekend*

U kunt een AIR-bestand in één stap in een pakket plaatsen en ondertekenen met de ADT-opdracht `-package`. U kunt ook een tussentijds, niet-ondertekend pakket maken met de opdracht `-prepare` en het tussentijdse pakket in een afzonderlijke stap ondertekenen met de opdracht `-sign`.

***Opmerking:*** *bij Java versie 1.5 en later worden hoge ASCII-tekens in wachtwoorden voor de beveiliging van PKCS12-certificatiebestanden niet geaccepteerd. Wanneer u uw code maakt of exporteert en ondertekent met een certificaatbestand, kunt u alleen gewone ASCII-tekens in het wachtwoord gebruiken.*

Bij het ondertekenen van het installatiepakket neemt ADT automatisch contact op met een tijdstempelserver om de tijd te verifiëren. De tijdstempelinformatie wordt opgenomen in het AIR-bestand. Een AIR-bestand dat een geverifieerd tijdstempel bevat, kan op een willekeurig moment in de toekomst worden geïnstalleerd. Als ADT geen verbinding kan maken met de tijdstempelserver, wordt het maken van het pakket geannuleerd. U kunt de tijdstempeloptie uitschakelen, maar zonder tijdstempel kan een AIR-toepassing niet worden geïnstalleerd nadat het certificaat is verlopen waarmee het installatiebestand is ondertekend.

Als u een pakket maakt om een bestaande AIR-toepassing bij te werken, moet het pakket worden ondertekend met hetzelfde certificaat als de oorspronkelijke toepassing. Als het oorspronkelijke certificaat is vernieuwd of minder dan 180 dagen geleden is verlopen, of als u wilt overstappen op een nieuw certificaat, kunt u een migratiehandtekening toepassen. Een migratiehandtekening houdt in dat het AIR-toepassingsbestand wordt ondertekend met zowel het oude als het nieuwe certificaat. Gebruik de opdracht `-migrate` om de migratiehandtekening toe te passen, zoals beschrijven in "ADT-opdracht voor migreren" op pagina 181.

***Belangrijk:*** *een migratiehandtekening kan alleen binnen 180 dagen nadat het oorspronkelijke certificaat verloopt worden toegepast. Zonder migratiehandtekening moeten bestaande gebruikers de bestaande versie verwijderen voordat de nieuwe versie geïnstalleerd kan worden. De periode van 180 dagen geldt alleen voor toepassingen waarbij AIR versie 1.5.3. of hoger is opgegeven in de naamruimte van de het descriptorbestand van de toepassing. Bij oudere versies van de AIR-runtime geldt deze beperking niet.*

Migratiehandtekeningen worden pas vanaf AIR 1.1 ondersteund. U kunt alleen een migratiehandtekening toepassen op toepassingen die met SDK versie 1.1 of hoger zijn verpakt.

Toepassingen die zijn geïmplementeerd met AIR-bestanden zijn toepassingen van het profiel Desktop. ADT kan niet worden gebruikt om een eigen installatieprogramma voor een AIR-toepassing te verpakken als het descriptorbestand van de toepassing het profiel Desktop niet ondersteunt. U kunt dit profiel beperken met het element `supportedProfiles` in het toepassingsdescriptorbestand. Zie "Apparaatprofielen" op pagina 257 en "supportedProfiles" op pagina 251.

***Opmerking:*** *de identiteit en het standaardinstallatiepad van een AIR-toepassing worden bepaald door de instellingen in het toepassingsdescriptorbestand. Zie "AIR-toepassingsdescriptorbestanden" op pagina 215.*

**Uitgevers-id's**

Vanaf AIR 1.5.3 worden uitgevers-id's niet meer gebruikt. Voor nieuwe toepassingen (die aanvankelijk met AIR 1.5.3 of hoger zijn gepubliceerd) hoeft en moet geen uitgevers-id worden opgegeven.

Wanneer u toepassingen bijwerkt die oorspronkelijk met oudere versies van AIR zijn gepubliceerd, moet u de oorspronkelijke uitgevers-id opgeven in het descriptorbestand van de toepassing. Als u dat niet doet, worden de geïnstalleerde versie van de toepassing en de bijgewerkte versie als verschillende toepassingen beschouwd. Als u een andere id gebruikt of de tag voor de uitgevers-id weglaat, moet een gebruiker de oudere versie verwijderen voordat de nieuwe versie geïnstalleerd kan worden.

Als u de oorspronkelijke gebruikers-id wilt bepalen, moet u zoeken naar het bestand `publisherid` in de submap META-INF/AIR waar de oorspronkelijke toepassing is geïnstalleerd. De uitgevers-id wordt aangegeven door de tekenreeks in dit bestand. Als u de uitgevers-id handmatig wilt kunnen opgeven, moet versie 1.5.3 (of hoger) van de AIR-runtime worden opgeven in de naamruimtedeclaratie van het descriptorbestand van de toepassing.

Voor toepassingen die zijn gepubliceerd vóór AIR 1.5.3, of die zijn gepubliceerd met de SDK van AIR 1.5.3 en waarvoor een oudere versie van AIR is opgegeven in de naamruimte van het descriptorbestand van de toepassing, wordt een uitgevers-id berekend op basis van het handtekeningcertificaat. Deze id wordt tezamen met de toepassings-id gebruikt om de identiteit van een toepassing te bepalen. Indien aanwezig wordt de uitgevers-id gebruikt voor de volgende doeleinden:

- Controleren of een AIR-bestand een update is en geen nieuwe toepassing

- Als onderdeel van de coderingssleutel voor de gecodeerde lokale opslagruimte
- Als onderdeel van het pad voor de opslagmap van de toepassing
- Als onderdeel van de verbindingstekenreeks voor lokale verbindingen
- Als onderdeel van de identiteitstekenreeks waarmee een toepassing wordt aangeroepen met de AIR in-browser API
- Als onderdeel van de OSID (gebruikt bij het maken van aangepaste programma's voor het installeren/verwijderen van software)

Vóór AIR 1.5.3 kon de uitgevers-id van een toepassing veranderen als een toepassingsupdate met een nieuw of vernieuwd certificaat werd voorzien van een migratiehandtekening. Wanneer u een uitgevers-id wijzigt, wordt ook het gedrag gewijzigd van de AIR-functies die afhankelijk zijn van de id. Zo zijn de gegevens in de bestaande gecodeerde lokale opslagruimte niet meer toegankelijk en moeten alle Flash- of AIR-instanties die een lokale verbinding maken met de toepassing ook gebruikmaken van de nieuwe id in de verbindingstekenreeks.

In AIR 1.5.3 of hoger is de uitgevers-id niet gebaseerd op het handtekeningcertificaat en wordt deze alleen toegewezen als de tag publisherID in het descriptorbestand van de toepassing is opgenomen. Een toepassing kan alleen worden bijgewerkt als de uitgevers-id die is opgegeven voor het AIR-pakket met de update overeenkomt met de huidige uitgevers-id.

## Verpakken met ADT

U kunt het ADT-opdrachtregelprogramma van AIR gebruiken om een AIR-toepassing in te pakken. Voor het verpakken moeten al uw ActionScript-, MXML- en uitbreidingscode zijn gecompileerd. U moet ook over een certificaat voor het ondertekenen van code beschikken.

Zie voor gedetailleerde informatie over ADT-opdrachten en -opties “AIR Developer Tool (ADT)” op pagina 173.

### Een AIR-pakket maken

Als u een AIR-pakket wilt maken, gebruikt u de ADT-pakketopdracht, waarbij u het doeltype instelt als *air* voor releasebuilds.

```
adt -package -target air -storetype pkcs12 -keystore ../codesign.p12 myApp.air myApp-app.xml
myApp.swf icons
```

In het voorbeeld wordt ervan uitgegaan dat het pad van het ADT-hulpprogramma zich op de pad-definitie van de shell van uw opdrachtregel bevindt. (Zie “Omgevingsvariabelen van het pad” op pagina 318 voor uitleg.)

U moet de opdracht uitvoeren in de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn myApp-app.xml (het descriptorbestand van de toepassing), myApp.swf en een map met pictogrammen.

Wanneer u volgens het voorbeeld de opdracht uitvoert, wordt u door ADT gevraagd het keystore-wachtwoord in te voeren. (De tekens van het wachtwoord die u invoert, worden niet altijd weergegeven. Druk op Enter wanneer u klaar bent met invoeren.)

### Een AIR-pakket van een AIRI-bestand maken

U kunt een AIRI-bestand ondertekenen om een installeerbaar AIR-pakket te maken:

```
adt -sign -storetype pkcs12 -keystore ../codesign.p12 myApp.airi myApp.air
```

# Een eigen bureaubladinstallatieprogramma verpakken

Sinds AIR 2 kunt u ADT gebruiken om eigen toepassingsinstallatieprogramma's te maken voor het distribueren van AIR-toepassingen. U kunt een EXE-installatieprogramma maken voor het distribueren van een AIR-toepassing in Windows. U kunt een DMG-installatieprogramma maken voor het distribueren van een AIR-toepassing in Mac OS. In AIR 2.5 en AIR 2.6 kunt u een DEB- of RPM-installatiebestand maken voor het distribueren van een AIR-toepassing in Linux.

Toepassingen die met een eigen installatieprogramma worden geïnstalleerd, zijn toepassingen met het profiel Uitgebreide desktop. ADT kan niet worden gebruikt om een eigen installatieprogramma voor een AIR-toepassing te verpakken als het descriptorbestand van de toepassing het profiel Uitgebreide desktop niet ondersteunt. U kunt dit profiel beperken met het element `supportedProfiles` in het toepassingsdescriptorbestand. Zie "Apparaatprofielen" op pagina 257 en "supportedProfiles" op pagina 251.

U kunt op twee manieren een versie van een eigen installatieprogramma voor de AIR-toepassing maken:

- U kunt het eigen installatieprogramma baseren op het descriptorbestand van de toepassing en andere bronbestanden. (Andere bronbestanden zijn bijvoorbeeld SWF-bestanden, HTML-bestanden en andere elementen.)
- U kunt het eigen installatieprogramma baseren op een AIR-bestand of een AIRI-bestand.

U moet ADT op hetzelfde besturingssysteem gebruiken als het besturingssysteem van het eigen installatieprogramma dat u wilt genereren. Dus als u een EXE-bestand voor Windows wilt maken, voert u ADT in Windows uit. Als u een DMG-bestand voor Mac OS wilt maken, voert u ADT in Mac OS uit. Als u een DEB- of RPG-bestand voor Linux wilt maken, voert u ADT uit de AIR 2.6-SDK in Linux uit.

Wanneer u een eigen installatieprogramma voor het distribueren van een AIR-toepassing maakt, krijgt deze de volgende mogelijkheden:

- De toepassing kan worden gestart en interactie aangaan met eigen processen wanneer u de klasse NativeProcess gebruikt. Zie een van de volgende bronnen voor meer informatie:
  - Communiceren met eigen processen in AIR (voor ActionScript-ontwikkelaars)
  - Communicating with native processes in AIR (voor HTML-ontwikkelaars)
- De toepassing kan native extensies gebruiken.
- De methode `File.openWithDefaultApplication()` kan worden gebruikt voor het openen van een bestand binnen de standaardsysteemtoepassing die is gedefinieerd voor het openen ervan, onafhankelijk van het bestandstype. (Er zijn beperkingen voor toepassingen die *niet* met een eigen installatieprogramma zijn geïnstalleerd. Zie de informatie over `File.openWithDefaultApplication()` in de naslaggids voor meer informatie.)

Wanneer verpakt als native installatieprogramma gaan echter enkele van de voordelen van de AIR-bestandsindeling verloren. Een enkel bestand kan niet langer worden gedistribueerd naar alle bureaubladcomputers. De ingebouwde updatefunctie (en het updateframework) werkt niet.

Wanneer de gebruiker dubbelklikt op het eigen installatieprogramma, wordt de AIR-toepassing geïnstalleerd. Als de vereiste versie van Adobe AIR nog niet op de computer is geïnstalleerd, wordt dit eerst van het netwerk gedownload en geïnstalleerd. Als er geen netwerkverbinding is voor het ophalen van de juiste versie van Adobe AIR (indien nodig), mislukt de installatie. De installatie mislukt ook als het besturingssysteem niet wordt ondersteund in Adobe AIR 2.

***Opmerking:*** *als u een uitvoerbaar bestand wilt opnemen in de geïnstalleerde toepassing, dient u in het bestandsysteem te controleren of het bestand uitvoerbaar is wanneer u een pakket maakt van de toepassing. (Indien noodzakelijk, kunt u op Mac en Linux chmod gebruiken om de uitvoerbare markering in te stellen.)*

### Een eigen installatieprogramma maken op basis van de bronbestanden van de toepassing

Als u een eigen installatieprogramma wilt maken op basis van de bronbestanden van de toepassing, gebruikt u de opdracht `-package` in combinatie met de volgende syntaxis (op één opdrachtregel):

```
adt -package AIR_SIGNING_OPTIONS
    -target native
    [WINDOWS_INSTALLER_SIGNING_OPTIONS]
    installer_file
    app_xml
    [file_or_dir | -C dir file_or_dir | -e file dir ...] ...
```

Deze syntaxis is vergelijkbaar met de syntaxis voor het verpakken van een AIR-bestand (zonder eigen installatieprogramma). Er zijn echter een paar verschillen:

- De optie `-target native` wordt aan de opdracht toegevoegd. (Als u `-target air` opgeeft, genereert ADT een AIR-bestand in plaats van een eigen installatieprogramma.)
- Het doelbestand (DMG of EXE) wordt opgegeven als `installer_file`.
- In Windows kunt u desgewenst een tweede serie handtekeningopties toevoegen, aangegeven met `[WINDOWS_INSTALLER_SIGNING_OPTIONS]` in het syntaxisvoorbeeld. In Windows kunt u behalve het AIR-bestand ook het Windows-installatieprogramma ondertekenen. Gebruik voor het certificaat en de handtekeningopties dezelfde syntaxis als voor het ondertekenen van het AIR-bestand (zie “ADT-opties voor codeondertekening” op pagina 188). U kunt het AIR-bestand en het installatieprogramma met hetzelfde certificaat ondertekenen, maar u kunt ook verschillende certificaten opgeven. Wanneer een gebruiker een ondertekend Windows-installatieprogramma vanaf internet downloadt, identificeert Windows op basis van het certificaat de bron van het bestand.

Zie voor details over andere ADT-opties dan de optie `-target`“AIR Developer Tool (ADT)” op pagina 173.

Met het volgende voorbeeld wordt een DMG-bestand gemaakt (een eigen installatieprogramma voor Mac OS):

```
adt -package
    -storetype pkcs12
    -keystore myCert.pfx
    -target native
    myApp.dmg
    application.xml
    index.html resources
```

Met het volgende voorbeeld wordt een EXE-bestand gemaakt (een eigen installatieprogramma voor Windows):

```
adt -package
    -storetype pkcs12
    -keystore myCert.pfx
    -target native
    myApp.exe
    application.xml
    index.html resources
```

Met het volgende voorbeeld wordt een EXE-bestand gemaakt en ondertekend:

```
adt -package
    -storetype pkcs12
    -keystore myCert.pfx
    -target native
    -storetype pkcs12
    -keystore myCert.pfx
    myApp.exe
    application.xml
    index.html resources
```

**Een native installatieprogramma maken voor een toepassing die native extensies gebruikt**

U kunt een native installatieprogramma maken van de bronbestanden voor de toepassing en de native extensiepakketten die de toepassing nodig heeft. Gebruik de opdracht `-package` met de volgende syntaxis (op één opdrachtregel):

```
adt -package AIR_SIGNING_OPTIONS
    -migrate MIGRATION_SIGNING_OPTIONS
    -target native
    [WINDOWS_INSTALLER_SIGNING_OPTIONS]
    installer_file
    app_xml
    -extdir extension-directory
    [file_or_dir | -C dir file_or_dir | -e file dir ...] ...
```

Dit is dezelfde syntaxis als de syntaxis die wordt gebruikt voor het verpakken van een native installatieprogramma, maar met twee extra opties. Gebruik de optie `-extdir` *`extension-directory`* om de map op te geven met de ANE-bestanden (native extensies) die de toepassing gebruikt. Gebruik de optionele markering `-migrate` en `MIGRATION_SIGNING_OPTIONS`-parameters om een update van een toepassing te ondertekenen met een migratiehandtekening, wanneer het primaire certificaat voor codeondertekening niet hetzelfde is als het certificaat dat werd gebruikt door de vorige versie. Zie “Een bijgewerkte versie van een AIR-toepassing ondertekenen.” op pagina 209 voor meer informatie.

Zie “AIR Developer Tool (ADT)” op pagina 173 voor informatie over ADT-opties.

Het volgende voorbeeld maakt een DMG-bestand (een native installatiebestand voor Mac OS) voor een toepassing die native extensies gebruikt:

```
adt -package
    -storetype pkcs12
    -keystore myCert.pfx
    -target native
    myApp.dmg
    application.xml
    -extdir extensionsDir
    index.html resources
```

**Een eigen installatieprogramma maken op basis van een AIR-bestand of een AIRI-bestand**

Met ADT kunt u een eigen installatieprogramma genereren op basis van een AIR-bestand of een AIRI-bestand. Als u een eigen installatieprogramma op basis van een AIR-bestand wilt maken, gebruikt u de ADT-opdracht `-package` in combinatie met de volgende syntaxis (op één opdrachtregel):

```
adt -package
    -target native
    [WINDOWS_INSTALLER_SIGNING_OPTIONS]
    installer_file
    air_file
```

Deze syntaxis is vergelijkbaar met de syntaxis voor het maken van een installatieprogramma op basis van de bronbestanden voor de AIR-toepassing. Er zijn echter een paar verschillen:

- Als bron geeft u een AIR-bestand op in plaats van een descriptorbestand van de toepassing en andere bronbestanden voor de AIR-toepassing.
- Geef geen handtekeningopties voor het AIR-bestand op, aangezien het al is ondertekend

Als u een eigen installatieprogramma op basis van een *AIRI*-bestand wilt maken, gebruikt u de ADT-opdracht `-package` in combinatie met de volgende syntaxis (op één opdrachtregel):

```
adt AIR_SIGNING_OPTIONS
    -package
    -target native
    [WINDOWS_INSTALLER_SIGNING_OPTIONS]
    installer_file
    airi_file
```

Deze syntaxis is vergelijkbaar met de syntaxis voor het maken van een eigen installatieprogramma op basis van een AIR-bestand. Er zijn echter een paar verschillen:

- Als bron geeft u een AIRI-bestand op.
- U geeft handtekeningopties voor het doel-AIR-bestand op.

Met het volgende voorbeeld wordt een DMG-bestand (een eigen installatieprogramma voor Mac OS) op basis van een AIR-bestand gemaakt:

```
adt -package -target native myApp.dmg myApp.air
```

Met het volgende voorbeeld wordt een EXE-bestand (een eigen installatieprogramma voor Windows) op basis van een AIR-bestand gemaakt:

```
adt -package -target native myApp.exe myApp.air
```

Met het volgende voorbeeld wordt een EXE-bestand (op basis van een AIR-bestand) gemaakt en ondertekend:

```
adt -package -target native -storetype pkcs12 -keystore myCert.pfx myApp.exe myApp.air
```

Met het volgende voorbeeld wordt een DMG-bestand (een eigen installatieprogramma voor Mac OS) op basis van een AIRI-bestand gemaakt:

```
adt -storetype pkcs12 -keystore myCert.pfx -package -target native myApp.dmg myApp.airi
```

Met het volgende voorbeeld wordt een EXE-bestand (een eigen installatieprogramma voor Windows) op basis van een AIRI-bestand gemaakt:

```
adt -storetype pkcs12 -keystore myCert.pfx -package -target native myApp.exe myApp.airi
```

In het volgende voorbeeld wordt een EXE-bestand gemaakt (gebaseerd op een AIRI-bestand) en wordt zowel een AIR- als een eigen Windows-handtekening gebruikt voor aanmelding:

```
adt -package -storetype pkcs12 -keystore myCert.pfx -target native -storetype pkcs12 -keystore
myCert.pfx myApp.exe myApp.airi
```

# Een captive-runtimebundel verpakken voor desktop-pc's

Een captive-runtimebundel is een pakket dat uw toepassingscode plus een toegewijde versie van de runtime bevat. Een toepassing die op deze manier wordt verpakt, gebruikt de gebundelde runtime in plaats van de elders op de computer van de gebruiker geïnstalleerde gedeelde runtime.

De geproduceerde bundel heeft op Windows de vorm van een zelfstandige map met toepassingsbestanden en op Mac OS de vorm van een .app-bundel. U dient de bundel voor een doelbesturingssysteem te produceren op het desbetreffende besturingssysteem. (Met een virtuele pc, zoals VMWare, kunt u meerdere besturingssystemen uitvoeren op één pc.)

De toepassing kan zonder installatie vanuit die map of bundel worden uitgevoerd.

**Voordelen**

- Produceert een zelfstandige toepassing
- Geen internettoegang vereist voor de installatie
- De toepassing staat los van runtime-updates
- Ondernemingen kunnen de specifieke combinatie van de toepassing en de runtime certificeren
- Ondersteunt het traditionele model voor software-implementatie
- Er is geen afzonderlijke herdistributie van de runtime vereist
- Kan gebruikmaken van de NativeProcess-API
- Kan native extensies gebruiken
- Kan de functie `File.openWithDefaultApplication()` zonder enige beperking gebruiken
- Kan zonder installatie worden uitgevoerd vanaf een USB-schijf of optische schijf

**Nadelen**

- Belangrijke beveiligingscorrecties staan niet automatisch ter beschikking van gebruikers als Adobe een beveiligingspatch publiceert
- De bestandsindeling .air kan niet worden gebruikt
- U dient zelf, indien nodig, een installatieprogramma te maken
- Geen ondersteuning voor update-API en -framework voor AIR
- Geen ondersteuning voor de browser-API voor AIR voor het installeren en starten van AIR-toepassingen vanuit een webpagina
- In Windows dient uw installatieprogramma de bestandsregistratie te verwerken
- Toepassing neemt meer schijfruimte in beslag

## Een captive-runtimebundel maken in Windows

Als u een captive-runtimebundel wilt maken voor Windows, dient u de toepassing te verpakken in Windows. Verpak de toepassing met gebruik van het ADT-doel *bundle*:

```
adt -package
    -keystore ..\cert.p12 -storetype pkcs12
    -target bundle
    myApp
    myApp-app.xml
    myApp.swf icons resources
```

Met deze opdracht maakt u de bundel in een map met de naam myApp. De map bevat de bestanden voor uw toepassing en de runtimebestanden. U kunt het programma rechtstreeks vanuit de map uitvoeren. Als u echter een vermelding in het programmamenu wilt maken of bestandstypen of URI-schemahandlers wilt registreren, dient u een installatieprogramma te maken dat de vereiste registervermeldingen instelt. De SDK van AIR beschikt niet over de tools voor het maken van een dergelijk installatieprogramma, maar er zijn verschillende oplossingen van andere bedrijven beschikbaar, waaronder gratis of tegen betaling beschikbare open-source installatietoolkits.

U kunt het native uitvoerbare bestand in Windows ondertekenen door na de vermelding `-target bundle` op de opdrachtregel een tweede set met ondertekeningsopties op te geven. Deze ondertekeningsopties identificeren de persoonlijke sleutel die en het gekoppelde certificaat dat moet worden gebruikt wanneer u de native Windows-handtekening toepast. (Doorgaans kan een AIR-certificaat voor codeondertekening worden gebruikt.) Alleen het primaire uitvoerbare bestand wordt ondertekend. Eventuele extra uitvoerbare bestanden in hetzelfde pakket als uw toepassing worden niet door dit proces ondertekend.

**Bestandstypekoppeling**
Als u uw toepassing in Windows wilt koppelen aan openbare of aangepaste bestandstypen, dient uw installatieprogramma de desbetreffende registervermeldingen in te stellen. De bestandstypen dienen ook te worden vermeld in het element fileTypes van het descriptorbestand van de toepassing.

Zie MSDN-bibliotheek: bestandstypen en bestandskoppelingen voor meer informatie over bestandstypen in Windows.

**Registratie van URI-handlers**
Uw installatieprogramma dient de vereiste registervermeldingen in te stellen, anders kan uw toepassing het starten van een URL niet afhandelen met gebruik van een bepaald URI-schema.

Zie MSDN-bibliotheek: een toepassing registreren voor een URL-protocol voor meer informatie over het registreren van een toepassing voor het afhandelen van een URI-schema.

## Een captive-runtimebundel maken op Mac OS X

Als u een captive-runtimebundel wilt maken voor Mac OS X, dient u de toepassing te verpakken op het Macintosh-besturingssysteem. Verpak de toepassing met gebruik van het ADT-doel *bundle*:

```
adt -package
    -keystore ../cert.p12 -storetype pkcs12
    -target bundle
    myApp.app
    myApp-app.xml
    myApp.swf icons resources
```

Met deze opdracht maakt u de toepassingsbundel myApp.app. De bundel bevat de bestanden voor uw toepassing en de runtimebestanden. U kunt de toepassing uitvoeren door te dubbelklikken op het pictogram myApp.app en de toepassing installeren door deze naar een geschikte locatie te slepen, zoals bijvoorbeeld de map Toepassingen. Teneinde bestandstypen of URI-schemahandlers te registreren, dient u het bestand met de eigenschappenlijst in het toepassingspakket te bewerken.

U kunt een schijfafbeeldingsbestand (*.dmg) maken voor distributie. De SDK van Adobe AIR beschikt niet over de tools waarmee u een .dmg-bestand voor een captive-runtimebundel kunt maken.

**Bestandstypekoppeling**
Als u uw toepassing op Mac OS X wilt koppelen aan openbare of aangepaste bestandstypen, dient u het bestand info.plist in de bundel te bewerken om de eigenschap CFBundleDocumentTypes in te stellen. Zie Information Property List Key Reference, CFBundleURLTypes in de bibliotheek voor Mac OS X-ontwikkelaars.

**Registratie van URI-handlers**
Als u wilt dat uw toepassing een URL kan starten met gebruik van een bepaald URI-schema, dient u het bestand info.plist in de bundel te bewerken om de eigenschap CFBundleURLTypes in te stellen. Zie Information Property List Key Reference, CFBundleDocumentTypes in de Mac OS X-bibliotheek voor ontwikkelaars.

# AIR-pakketten distribueren voor bureaubladcomputers

AIR-toepassingen kunnen worden gedistribueerd als AIR-pakket, dat de toepassingscode en alle assets bevat. U kunt dit pakket distribueren via de gebruikelijke wegen, zoals download, e-mail of fysieke media als cd-rom. Gebruikers kunnen de toepassing installeren door te dubbelklikken op het AIR-bestand. Met de interne browser-API van AIR (een ActionScript-webbibliotheek) kunt u gebruikers uw AIR-toepassing (en indien noodzakelijk Adobe® AIR®) laten installeren door op een koppeling op een webpagina te klikken.

AIR-toepassingen kunnen ook verpakt en gedistribueerd worden als eigen installatieprogramma's (met andere woorden als EXE-bestanden bij Windows, DMG-bestanden bij Mac en DEB- of RPM-bestanden bij Linux). Eigen installatiepakketten kunnen worden gedistribueerd en geïnstalleerd volgens de relevante platformconventies. Wanneer u uw toepassing als eigen pakket distribueert, beschikt u niet meer over een aantal van de voordelen van de AIR-bestandindeling. Een enkel installatiebestand kan namelijk niet meer worden gebruikt op de meeste platforms, het AIR-updateframework kan niet meer worden gebruikt en de interne browser-API kan niet meer worden gebruikt.

## Een AIR-toepassing op het bureaublad installeren en uitvoeren

U kunt het AIR-bestand eenvoudig naar de geadresseerde verzenden. U kunt het AIR-bestand bijvoorbeeld verzenden als een e-mailbijlage of als een koppeling op een webpagina.

Een gebruiker die de AIR-toepassing downloadt, moet de volgende stappen uitvoeren om de toepassing te installeren:

1 Dubbelklik op het AIR-bestand.

Adobe AIR moet al op de computer zijn geïnstalleerd.

2 Laat de standaardinstellingen geselecteerd in het installatievenster en klik op Doorgaan.

In Windows voert AIR automatisch de volgende bewerkingen uit:

- De toepassing wordt geïnstalleerd in de map Program Files.
- Op het bureaublad wordt een snelkoppeling voor de toepassing gemaakt.
- In het menu Start wordt een snelkoppeling gemaakt.
- De toepassing wordt toegevoegd aan het onderdeel Software in het Configuratiescherm.

In Mac OS wordt de toepassing standaard toegevoegd aan de map Programma's.

Als de toepassing al is geïnstalleerd, kan de gebruiker in het installatieprogramma de bestaande versie van de toepassing openen of de toepassing bijwerken naar de versie in het gedownloade AIR-bestand. In het installatieprogramma wordt de toepassing geïdentificeerd met de toepassings-id en de uitgevers-id in het AIR-bestand.

3 Klik op Voltooien wanneer de installatie is voltooid.

Als een gebruiker in Mac OS een recentere versie van een toepassing wil installeren, moet de gebruiker over voldoende toegangsrechten beschikken om te installeren in de toepassingsmap. In Windows en Linux heeft een gebruiker beheerdersrechten nodig.

Een nieuwe versie van een toepassing kan ook worden geïnstalleerd via ActionScript of JavaScript. Zie "AIR-toepassingen bijwerken" op pagina 270 voor meer informatie.

Nadat de AIR-toepassing is geïnstalleerd, dubbelklikt de gebruiker op het pictogram van de toepassing om de toepassing uit te voeren, net als bij een andere desktoptoepassing.

- Dubbelklik in Windows op het pictogram van de toepassing (die op het bureaublad of in een map is geïnstalleerd) of selecteer de toepassing in het menu Start.
- Dubbelklik in Linux op het pictogram van de toepassing (die op het bureaublad of in een map is geïnstalleerd) of selecteer de toepassing in het menu van de toepassing.
- Dubbelklik in Mac OS op de toepassing in de map waarin deze is geïnstalleerd. De standaard installatiemap is /Applications.

***Opmerking:*** *alleen AIR-toepassingen die zijn ontwikkeld voor AIR 2.6 of eerder kunnen op Linux worden geïnstalleerd.*

Met de AIR-functie voor *naadloze installatie* kan een gebruiker een AIR-toepassing installeren door op een koppeling op een webpagina te klikken. Met de AIR-functie voor *oproepen vanuit de browser* kan een gebruiker een geïnstalleerde AIR-toepassing uitvoeren door op een koppeling op een webpagina te klikken. Deze functies worden beschreven in de volgende sectie.

## AIR-toepassingen vanaf een webpagina installeren en uitvoeren

Met de interne browser-API van AIR kunt u een AIR-toepassing vanaf een webpagina installeren en uitvoeren. De interne browser-API van AIR wordt geleverd in een SWF-bibliotheek, *air.swf*, die wordt gehost door Adobe. De AIR SDK bevat een 'badge'-voorbeeldtoepassing die deze bibliotheek gebruikt om een AIR-toepassing (en indien noodzakelijk de runtime) te installeren, bijwerken of uit te voeren. U kunt de opgegeven voorbeeldbadge bewerken of direct uw eigen badgewebtoepassing maken met behulp van de onlinebibliotheek air.swf.

Elke AIR-toepassing kan worden geïnstalleerd via een webpaginabadge. Echter, alleen toepassingen die het element `<allowBrowserInvocation>true</allowBrowserInvocation>` in het toepassingsdescriptorbestand bevatten, kunnen worden uitgevoerd door een webbadge.

### Meer Help-onderwerpen

"AIR.SWF API interne browser" op pagina 261

## Ondernemingsbrede implementatie op bureaubladcomputers

IT-beheerders kunnen de Adobe AIR-runtime en AIR-toepassingen zonder toezicht installeren met behulp van standaard implementatieprogramma's. IT-beheerders kunnen het volgende doen:

- de Adobe AIR-runtime zonder toezicht installeren met hulpprogramma's zoals Microsoft SMS, IBM Tivoli of een implementatieprogramma waarmee installaties met een bootstrapper zonder toezicht mogelijk zijn;

- de AIR-toepassing zonder toezicht installeren met hetzelfde hulpprogramma waarmee de runtime is geïmplementeerd.

Zie de *Beheerdershandleiding voor Adobe AIR (http://www.adobe.com/go/learn_air_admin_guide_nl) voor meer informatie.*

## Installatielogboeken op bureaubladcomputers

Tijdens de installatie van de AIR-runtime zelf of een AIR-toepassing worden installatielogboeken bijgehouden. Als er installatie- of updateproblemen optreden, kunt u in deze logboeken achterhalen wat hier de oorzaak van is.

De logboeken worden op de volgende locaties gemaakt:

- Mac: het standaardsysteemlogboek (`/private/var/log/system.log`)

  U kunt het systeemlogboek voor Mac bekijken door de Consoletoepassing te openen (normaal gesproken in de map Hulpprogramma's).

- Windows XP: `C:\Documents and Settings\<gebruikersnaam>\Local Settings\Application Data\Adobe\AIR\logs\Install.log`
- Windows Vista, Windows 7: `C:\Gebruikers\<gebruikersnaam>\AppData\Local\Adobe\AIR\logs\Install.log`
- Linux: `/home/<gebruikersnaam>/.appdata/Adobe/AIR/Logs/Install.log`

***Opmerking:** deze logboeken worden pas vanaf AIR 2 gemaakt.*

# Hoofdstuk 7: AIR-toepassingen ontwikkelen voor mobiele apparaten

AIR-toepassingen op mobiele apparaten worden geïmplementeerd als native toepassingen. Voor deze toepassingen wordt de toepassingsindeling gebruikt van het apparaat, niet de bestandsindeling .air. Momenteel ondersteunt Android-APK pakketten en iOS-IPA-pakketten. Zodra u de releaseversie van uw toepassingspakket hebt gemaakt, kunt u uw toepassing distribueren via het mechanisme van het standaardplatform. Voor Android is dit doorgaans de Android Market, voor de iOS de Apple App Store.

U kunt de AIR-SDK en Flash Professional, Flash Builder of een ander ActionScript-ontwikkelingsprogramma gebruiken om AIR-toepassingen voor mobiele apparaten te ontwikkelen. Mobiele AIR-toepassingen die op HTML zijn gebaseerd, worden momenteel niet ondersteund.

***Opmerking:*** *de RIM (Research In Motion) BlackBerry Playbook beschikt over een eigen SDK voor AIR-ontwikkeling. Zie RIM: BlackBerry Tablet OS Development voor informatie over Playbook-ontwikkeling.*

***Opmerking:*** *In dit document wordt beschreven hoe u iOS-toepassingen kunt ontwikkelen met behulp van de AIR 2.6 SDK of later. Toepassingen die zijn gemaakt met AIR 2.6+ kunnen worden geïnstalleerd op iPhone 3Gs-, iPhone 4- en iPad-apparaten met iOS 4 of hoger. Als u AIR-toepassingen wilt ontwikkelen voor eerdere versies van iOS, moet u de AIR 2 Packager for iPhone gebruiken, zoals beschreven in iPhone-toepassingen ontwikkelen.*

Zie Adobe AIR SDK Privacyhandleiding voor meer informatie over privacy.

Zie Adobe AIR-systeemvereisten voor de volledige systeemvereisten voor het uitvoeren van AIR-toepassingen.

## Uw ontwikkelomgeving instellen

Mobiele platforms hebben naast de gangbare installatie van de AIR-, Flex- en Flash-ontwikkelomgeving extra instellingsvereisten. (Zie voor meer informatie over het instellen van de standaard AIR-ontwikkelomgeving "Gereedschappen van Adobe Flash-platform voor AIR-ontwikkeling" op pagina 17.)

### Android installeren

Er is doorgaans geen speciale setup vereist voor Android in AIR 2.6 of later. Het Android ADB-hulpprogramma maakt deel uit van de AIR SDK (in de map lib/android/bin). De AIR-SDK gebruikt het hulpprogramma ADB om toepassingspakketten op het apparaat te installeren, verwijderen en uit te voeren. U kunt ADB ook gebruiken om systeemlogboeken weer te geven. Voor het maken en uitvoeren van een Android-emulator moet u de speciale Android-SDK downloaden.

Als uw toepassing elementen toevoegt aan het `<manifestAdditions>`-element in het descriptorbestand van de toepassing die door uw huidige AIR-versie als ongeldig worden beschouwd, dient u een recentere versie van de Android SDK te installeren. Stel de omgevingsvariabele AIR_ANDROID_SDK_HOME of de opdrachtregelparameter `-platformsdk` in op het bestandspad van de SDK. ADT, het AIR-pakkethulpprogramma, gebruikt deze SDK om de vermeldingen in het `<manifestAdditions>`-element te valideren.

In AIR 2.5 moet u een afzonderlijk exemplaar van de Android-SDK downloaden via Google. U kunt de omgevingsvariabele AIR_ANDROID_SDK_HOME instellen voor het raadplegen van de map Android SDK. Als u deze omgevingsvariabele niet instelt, moet u het pad naar de Android-SDK opgeven in het argument `-platformsdk` in de ADT-opdrachtregel.

### Meer Help-onderwerpen

"ADT-omgevingsvariabelen" op pagina 197

"Omgevingsvariabelen van het pad" op pagina 318

## iOS installeren

Als u een iOS-toepassing op een apparaat wilt installeren en testen en deze toepassing wilt distribueren, moet u zich aanmelden bij het Apple iOS-ontwikkelprogramma (dit is een service waarvoor kosten worden berekend). Zodra u zich bij het Apple iOS-ontwikkelprogramma hebt aangemeld, kunt u naar de iOS Provisioning Portal gaan voor de volgende items en bestanden van Apple die worden vereist voor het installeren van een toepassing op een apparaat voor het uitvoeren van testen en vervolgens voor distributie. Deze items en bestanden bevatten onder andere:

- Ontwikkelings- en distributiecertificaten
- Toepassings-id's
- Inrichtingsbestanden voor ontwikkeling en distributie

# Overwegingen bij het ontwerpen van mobiele toepassingen

De operationele context en fysieke kenmerken van mobiele apparaten behoeven zorgvuldig(e) codering en ontwerp. Zo is het bijvoorbeeld essentieel dat code zodanig wordt gestroomlijnd dat het zo snel mogelijk wordt uitgevoerd. Uiteraard zijn er grenzen aan het optimaliseren van code: met een intelligent ontwerp dat binnen de begrenzingen van het apparaat functioneert, kan worden voorkomen dat uw visuele presentatie het renderingsysteem overbelast.

**Code**

Hoewel het altijd gunstig is om uw code sneller uit te voeren, beloont de langzamere processorsnelheid van de meeste mobiele apparaten de tijd die is besteed aan het schrijven van leancode. Daarnaast werken mobiele apparaten bijna altijd op batterijstroom. Het zelfde resultaat behalen met minder werk kost minder batterijstroom.

**Ontwerpen**

Factoren als een klein scherm, de interactiemodus van het aanraakscherm en zelfs de constant veranderende omgeving van een mobiele gebruiker moeten in overweging worden genomen bij het ontwerpen van de gebruikerservaring van uw toepassing.

**Code en ontwerp samen**

Als in uw toepassing animatie wordt gebruikt, is renderingoptimalisering zeer belangrijk. Optimalisering van code is echter vaak niet voldoende. U moet de visuele aspecten van de toepassing zodanig ontwerpen dat de code deze efficiënt rendert.

Belangrijke optimalisatiemethoden worden besproken in de handleiding De prestaties voor het Flash-platform optimaliseren. De methoden in de handleiding zijn van toepassing op alle Flash- en AIR-inhoud, maar zijn cruciaal voor het ontwerpen van toepassingen die goed werken op mobiele apparaten.

- Paul Trani: Tips and Tricks for Mobile Flash Development
- roguish: GPU Test App AIR for Mobile
- Jonathan Campos: Optimization Techniques for AIR for Android apps
- Charles Schulze: AIR 2.6 Game Development: iOS included

## Gebruiksduur van de toepassing

Wanneer uw toepassing beeldscherpte verliest vanwege een andere toepassing, wordt de framesnelheid door AIR verlaagd naar 4-frames-per-seconde en wordt het renderen van afbeeldingen stopgezet. Onder deze framesnelheid kunnen verbindingen met streamingnetwerken en sockets worden verbroken. Als uw toepassing dergelijke verbindingen niet gebruikt, kunt u de framesnelheid zelfs nog lager zetten.

Indien van toepassing moet u het afspelen van geluid stopzetten en luisteraars naar de geolocatie en acceleratiemetersensoren verplaatsen. Het AIR NativeApplication-object verzendt het activeren en deactiveren van gebeurtenissen. U kunt deze gebeurtenissen gebruiken om de overgang tussen de actieve status en de achtergrondstatus te regelen.

De meeste besturingssystemen beëindigen achtergrondtoepassingen zonder een waarschuwing te geven. Als u de status van uw toepassing regelmatig opslaat, moet de status van de toepassing automatisch worden hersteld in een redelijke status: hetzij vanuit de achtergrondstatus in een actieve status of door op nieuw op te starten.

## Informatiedichtheid

De fysieke grootte van het scherm van mobiele apparaten is kleiner dan op het scherm van een desktopcomputer, hoewel de pixeldichtheid (pixels per inch) daarvan hoger is. Dezelfde tekengrootte levert letters op die fysiek kleiner zijn op het scherm van een mobiel apparaat dan op een scherm van een desktopcomputer. U moet vaak een grotere tekengrootte gebruiken om te zorgen dat de tekst leesbaar is. Over het algemeen is een lettergrootte van 14 punten de kleinste lettergrootte die gemakkelijk kan worden gelezen.

Mobiele apparaten worden vaak onderweg gebruikt en onder slechte lichtomstandigheden. Ga na hoeveel informatie u realistisch gezien leesbaar op het scherm kunt weergeven. Het zou wel eens minder kunnen zijn dan u kunt weergeven op een scherm met dezelfde pixeldichtheid op een desktopcomputer.

Houd er ook rekening mee dat bij het aanraken van een scherm een deel van de weergave wordt geblokkeerd door de vinger en hand van de gebruiker. Plaats interactieve elementen aan de zijkanten en onderkant van het scherm wanneer de gebruiker deze langer interactief moet gebruiken dan gedurende een kortstondige aanraking.

## Tekstinvoer

Bij veel apparaten wordt een virtueel toetsenbord voor tekstinvoer gebruikt. Virtuele toetsenborden verbergen een deel van het scherm en zijn vaak lastig te gebruiken. U moet proberen te vermijden om afhankelijk te zijn van toetsenbordgebeurtenissen (behalve schermtoetsen).

Implementeer eventueel alternatieven voor het gebruik van invoertekstvelden. Als u de gebruiker bijvoorbeeld een numerieke waarde wilt laten invoeren, hebt u geen tekstveld nodig. U kunt twee knoppen maken om de waarde te verhogen of te verlagen.

## Schermtoetsen

Mobiele apparaten beschikken over een verschillend aantal schermtoetsen. Schermtoetsen zijn knoppen die u kunt programmeren voor verschillende functies. Volg de platformconventies voor deze toetsen in uw toepassing.

## Wijzigingen van de schermoriëntatie

Mobiele inhoud kan staand en liggend worden weergegeven. Overweeg hoe uw toepassing omgaat met de wijzigingen van de schermoriëntatie. Zie Oriëntatie werkgebied voor meer informatie.

## Scherm dimmen

Met AIR wordt niet automatisch voorkomen dat het scherm wordt gedimd als er beelden wordt afgespeeld. U kunt de eigenschap `systemIdleMode` van het object van de AIR Native Application gebruiken om te beslissen of het apparaat in een energiebesparingsmodus wordt gezet. (Op bepaalde platforms moet u de relevante machtigingen aanvragen om deze functie te kunnen gebruiken.)

## Binnenkomende telefoongesprekken

Het geluid wordt automatisch door de AIR-runtime gedimd wanneer de gebruiker een telefoongesprek initieert of ontvangt. Bij Android moet u de Android-machtiging READ_PHONE_STATE in de toepassingsdescriptor instellen als door uw toepassing geluid wordt afgespeeld wanneer deze op de achtergrond wordt uitgevoerd. Wanneer u dit niet doet, wordt door Android voorkomen dat telefoongesprekken door de runtime worden gedetecteerd en dat het geluid automatisch wordt gedimd. Zie “Android-machtigingen” op pagina 81.

## Aanraakdoelen

Denk na over de grootte van aanraakdoelen wanneer u knoppen en andere elementen voor de gebruikersinterface ontwerpt waarop de gebruiker tikt. Zorg ervoor dat deze elementen groot genoeg zijn om eenvoudig met een vinger op een aanraakscherm te kunnen worden geactiveerd. Zorg er ook voor dat er voldoende tussenruimte is tussen doelen. Het gebied van de aanraakdoelen moet aan elke kant circa 44 pixels tot 57 pixels zijn voor een standaard telefoonscherm met hoge dpi.

## Installatiegrootte van toepassingspakket

Mobiele apparaten hebben doorgaans veel minder opslagruimte voor het installeren van toepassingen en voor gegevens dan desktopcomputers. U moet de grootte van het pakket beperken door ongebruikte assets en bibliotheken te verwijderen.

Bij Android wordt het toepassingspakket niet uitgepakt en in afzonderlijke bestanden geplaatst wanneer een toepassing wordt geïnstalleerd. In plaats daarvan worden assets op een tijdelijke opslaglocatie gedecomprimeerd wanneer deze worden geopend. Voor het minimaliseren van de opslagruimte van de gerecombineerde asset, moet u de bestandsstream en de URL-stream afsluiten wanneer de assets volledig zijn geladen.

## Toegang tot bestandssysteem

De verschillende mobiele besturingssystemen leggen verschillende beperkingen op de bestandssystemen op. Deze beperkingen zijn vaak niet gelijk aan de door bureaubladbesturingssystemen opgelegde beperkingen. De voor het opslaan van bestanden en gegevens geschikte locatie kan dus per platform verschillen.

Een resultaat van deze verschillende bestandssystemen is dat snelkoppelingen naar gemeenschappelijke mappen die door de klasse AIR File worden verschaft niet altijd beschikbaar zijn. De volgende tabel geeft aan welke snelkoppelingen kunnen worden gebruikt voor Android en iOS:

| | **Android** | **iOS** |
|---|---|---|
| File.applicationDirectory | Alleen-lezen via URL (niet het native pad) | Alleen-lezen |
| File.applicationStorageDirectory | Beschikbaar | Beschikbaar |
| File.cacheDirectory | Beschikbaar | Beschikbaar |
| File.desktopDirectory | Hoofdmap van sd-kaart | Niet beschikbaar |
| File.documentsDirectory | Hoofdmap van sd-kaart | Beschikbaar |
| File.userDirectory | Hoofdmap van sd-kaart | Niet beschikbaar |
| File.createTempDirectory() | Beschikbaar | Beschikbaar |
| File.createTempFile() | Beschikbaar | Beschikbaar |

De richtlijnen van Apple voor iOS-toepassingen bieden specifieke regels waarop opslaglocaties moeten worden gebruikt voor bestanden in diverse situaties. Eén richtlijn houdt bijvoorbeeld in dat alleen bestanden met door gebruikers ingevoerde gegevens of gegevens die niet op een andere manier opnieuw kunnen worden gegenereerd of gedownload, in een map moeten worden opgeslagen die is opgegeven voor externe back-up. Zie Back-ups en caching van bestanden beheren voor informatie over het naleven van de richtlijnen van Apple met betrekking tot back-ups en caching van bestanden.

## UI-componenten

Adobe heeft een versie van het Flex-framework ontwikkeld die is geoptimaliseerd voor mobiele apparaten. Voor meer informatie gaat u naar Developing Mobile Applications with Flex and Flash Builder.

Componenten van communityprojecten die geschikt zijn voor mobiele toepassingen, zijn ook beschikbaar. Deze zijn onder meer:

- Josh Tynjala's Feathers-softwarebesturingselementen voor Starling
- 'skinnable' versie van Minimal Comps van Derrick Grigg
- as3flobile components van Todd Anderson

## Met Stage 3D versnelde rendering van afbeeldingen

Vanaf AIR 3.2 biedt AIR for Mobile ondersteuning voor met Stage 3D versnelde rendering van afbeeldingen. De Stage3D-ActionScript-API's worden gevormd door een set laag-niveau-API's met GPU-versnelling die geavanceerde 2D- en 3D-functies mogelijk maken. Deze laag-niveau-API's geven ontwikkelaars de flexibiliteit om GPU-hardwareversnelling te benutten, hetgeen de prestaties aanzienlijk ten goede komt. U kunt ook de gamingengines gebruiken die de Stage3D-ActionScript-API's ondersteunen.

Zie Gamingengines, 3D en Stage 3D voor meer informatie.

## Vloeiende video's

Met het oog op de prestaties is het vloeiend maken van video's uitgeschakeld in AIR.

### Native functies

**AIR 3.0+**

Veel mobiele platformen bieden functionaliteit die nog niet beschikbaar is via de standaard-API van AIR. Vanaf AIR 3 kunt u AIR uitbreiden met uw eigen bibliotheken met native code. Via deze native extensiebibliotheken hebt u toegang tot beschikbare functies van het besturingssysteem en zelfs tot functies die specifiek bij een bepaald apparaat horen. Native extensies kunnen worden geschreven in C voor iOS, en in Java of C voor Android. Zie Inleiding tot native extensies voor Adobe AIR voor meer informatie over het ontwikkelen van native extensies.

# Workflow voor het maken van AIR-toepassingen voor mobiele apparaten

De workflow voor het maken van een AIR-toepassing voor mobiele apparaten (of andere apparaten) is over het algemeen vrijwel gelijk aan die voor het maken van een desktoptoepassing. De belangrijkste verschillen in workflow doen zich voor bij het verpakken, het uitvoeren van een foutopsporing en het installeren van een toepassing. Bij de AIR for Android-toepassingen wordt bijvoorbeeld de native APK-pakketindeling van Android gebruikt in plaats van de pakketindeling van AIR. Om die reden worden bij deze toepassingen ook de standaardmethoden van Android voor installeren en bijwerken gebruikt.

## AIR for Android

De volgende stappen zijn standaard wanneer u een AIR-toepassing voor Android ontwikkelt:

- Schrijf de ActionScript- of MXML-code.
- Maak een AIR-toepassingsdescriptorbestand (met behulp van de naamruimte 2.5 of later).
- Compileer de toepassing.
- Verpak de toepassing als een Android-pakket (.apk).
- Installeer de AIR-runtime op het apparaat of de Android-emulator (als een externe runtime wordt gebruikt; captive runtime is de standaard in AIR 3.7 en hoger).
- Installeer de toepassing op het apparaat (of Android-emulator).
- Start de toepassing op het apparaat.

U kunt Adobe Flash Builder, Adobe Flash Professional CS5 of de opdrachtregels gebruiken om deze stappen te realiseren.

Zodra uw AIR-toepassing is voltooid en is verpakt als APK-bestand, kunt u het verzenden naar de Android Market of distribueren via andere kanalen.

## AIR for iOS

De volgende stappen zijn standaard wanneer u AIR-toepassingen voor iOS ontwikkelt:

- Installeer iTunes.
- Genereer de vereiste ontwikkelaarsbestanden en id's op de iOS Provisioning Portal van Apple. Deze items zijn onder meer:
  - Certificaat voor ontwikkelaars

- Toepassings-id
- Inrichtingsprofiel

Wanneer u het inrichtingsprofiel maakt, moet u de id's opgeven van alle testapparaten waarop u de toepassing wilt installeren.

- Converteer het ontwikkelingscertificaat en de persoonlijke sleutel naar een P12-sleutelarchief-bestand.
- Schrijf de ActionScript- of MXML-code voor de toepassing.
- Compileer de toepassing met een ActionScript- of MXML-compiler.
- Maak pictogramafbeeldingen en een afbeelding voor het startscherm van de toepassing.
- Maak de toepassingsdescriptor (met behulp van de naamruimte 2.6 of groter).
- Verpak het IPA-bestand met behulp van ADT.
- Gebruik iTunes om uw inrichtingsprofiel op uw testapparaat te plaatsen.
- Installeer en test de toepassing op uw iOS-apparaat. U kunt ofwel iTunes of ADT in combinatie met USB (AIR 3.4 en hoger) gebruiken om het IPA-bestand te installeren.

Zodra uw AIR-toepassing is voltooid, kunt u deze opnieuw verpakken met behulp van een distributiecertificaat en inrichtingsprofiel. U kunt het vervolgens naar de Apple App Store verzenden.

# Eigenschappen van mobiele toepassingen instellen

Net als bij overige AIR-toepassingen stelt u de basiseigenschappen van de toepassing in het toepassingsdescriptorbestand in. Bij mobiele toepassingen worden enkele desktopspecifieke eigenschappen, zoals venstergrootte en transparantie, genegeerd. Bij mobiele toepassingen kunnen ook de eigen platformspecifieke eigenschappen worden gebruikt. U kunt bijvoorbeeld een element `android` opnemen voor Android-toepassingen en een element `iPhone` voor iOS-toepassingen.

## Algemene instellingen

Verscheidene instellingen van de toepassingsdescriptor zijn belangrijk voor alle toepassingen voor mobiele apparaten.

### Vereiste AIR-runtimeversie

Geef de versie op van de AIR-runtime die wordt vereist door uw toepassing en gebruik daarbij de naamruimte van het toepassingsdescriptorbestand.

De naamruimte, die wordt toegewezen in het `application`-element, bepaalt voor een groot deel welke functies uw toepassing kan gebruiken. Als de naamruimte AIR 2.7 bijvoorbeeld door uw toepassing wordt gebruikt en de gebruiker een toekomstige versie heeft geïnstalleerd, wordt door uw toepassing nog steeds de AIR 2.7-functionaliteit waargenomen (zelfs wanneer de functionaliteit in de toekomstige versie is gewijzigd). Uw toepassing heeft uitsluitend toegang tot de nieuwe functionaliteit en functies als u de naamruimte wijzigt en een update publiceert. Beveiligingsfixes zijn een belangrijke uitzondering op deze regel.

Op apparaten waar een runtime afzonderlijk van de toepassing wordt gebruikt, zoals Android op AIR 3.6 en eerder, wordt de gebruiker gevraagd om AIR te installeren of te upgraden als deze niet over de vereiste versie beschikt. Op apparaten waarbij de runtime is inbegrepen, zoals de iPhone, komt deze situatie niet voor (aangezien de vereiste versie bij de toepassing is verpakt).

***Opmerking:*** *(AIR 3.7 en hoger) Standaard verpakt ADT de runtime in Android-toepassingen.*

Geef de naamruimte op met behulp van het xmlns-attribuut van het hoofdelement van de `toepassing`. De volgende naamruimten moeten worden gebruikt voor mobiele toepassingen (afhankelijk vanaf welk mobiele platform u het doel instelt):

```
iOS 4+ and iPhone 3Gs+ or Android:
                            <application xmlns="http://ns.adobe.com/air/application/2.7">
                            iOS only:
                            <application xmlns="http://ns.adobe.com/air/application/2.0">
```

***Opmerking:*** *ondersteuning voor iOS 3-apparaten wordt geleverd door het verpakkingshulpprogramma voor iPhone-SDK, gebaseerd op de AIR 2.0-SDK. Zie iPhone-toepassingen ontwikkelen voor informatie over het ontwikkelen van AIR-toepassingen voor iOS 3. De AIR 2.6 SDK (en hoger) biedt ondersteuning voor iOS 4 en hoger op iPhone 3Gs-, iPhone 4- en iPad-apparaten.*

## Meer Help-onderwerpen

"application" op pagina 220

## Toepassingsidentiteit

Verschillende instellingen moeten uniek zijn voor elke toepassing die u publiceert. Dit zijn onder meer de id, de naam en de bestandsnaam.

### Id's van Android-toepassingen

Bij Android wordt de id geconverteerd naar de Android-verpakkingsnaam met het voorvoegsel "air." in de AIR-id. Dus: als uw AIR-id *com.example.MyApp* is, is de Android-verpakkingsnaam *air.com.example.MyApp*.

```
<id>com.example.MyApp</id>
                                <name>My Application</name>
                                <filename>MyApplication</filename>
```

Bovendien wordt de id, indien deze geen geldige naam voor het Android-besturingssysteem is, geconverteerd naar een geldige naam. Koppeltekens worden gewijzigd in onderstrepingstekens en cijfers voorin een id-component worden voorafgegaan door de hoofdletter "A". Bijvoorbeeld: de id *3-goats.1-boat*wordt omgezet naar de pakketnaam: *air.A3_goats.A1_boat*.

***Opmerking:*** *het voorvoegsel dat aan de toepassings-id wordt toegevoegd, kan worden gebruikt om AIR-toepassingen te identificeren in de Android Market. Als u niet wilt dat uw toepassing vanwege het voorvoegsel wordt geïdentificeerd als een AIR-toepassing, moet u het APK-bestand uitpakken, de toepassings-id wijzigen en deze opnieuw verpakken zoals wordt beschreven in Afzien van analyse van AIR-toepassingen voor Android.*

### Id's van iOS-toepassingen

Stel de id van de AIR-toepassing zodanig in dat deze overeenkomt met de toepassings-id die u hebt gemaakt in de Apple iOS Provisioning Portal.

Id's van iOS-toepassingen bestaan uit een bundlebron-id die wordt gevolgd door een bundle-id. De bundlebron-id is een tekenreeks, zoals 5RM86Z4DJM, die Apple aan de toepassings-id heeft toegewezen. De bundle-id bestaat uit een omgekeerde domein-stijlnaam die u hebt gekozen. Het laatste teken in de bundle-id kan een sterretje (*) zijn dat naar een jokerteken in de toepassings-id verwijst. Als het laatste teken in de bundle-id het jokerteken is, kunt u dit jokerteken wijzigen in een geldige tekenreeks.

Bijvoorbeeld:

- als uw Apple-toepassings-id `5RM86Z4DJM.com.example.helloWorld` is, moet u`com.example.helloWorld` in de toepassingsdescriptor gebruiken

- Als uw Apple-toepassings-id `96LPVWEASL.com.example.*` is (een toepassings-id met jokertekens), kunt u `com.example.helloWorld` gebruiken of `com.example.anotherApp`, of een andere id die begint met `com.example`.
- Ten slotte, als uw Apple-toepassings-id alleen bestaat uit de bundlebron-id en een jokerteken, zoals `38JE93KJL.*`, kunt u elk gewenst toepassings-id in AIR gebruiken.

Wanneer u de toepassings-id opgeeft, moet u het gedeelte met de bundlebron-id van de toepassings-id niet opgeven.

### Meer Help-onderwerpen

"id" op pagina 237

"filename" op pagina 232

"name" op pagina 245

## Toepassingsversie

In AIR 2.5 en later geeft u de toepassingsversie op in het element `versionNumber`. Het element `version` mag niet langer worden gebruikt. Wanneer u een waarde opgeeft voor `versionNumber`, moet u een reeks opgeven van maximaal drie cijfers gescheiden door punten, zoals: "0.1.2". Elk segment van het versienummer kan hoogstens drie cijfers hebben. (Met andere woorden: "999.999.999" is het hoogste versienummer dat is toegestaan.) U hoeft niet alle drie de segmenten in het cijfer op te nemen. "1" en "1.0" zijn ook geldige versienummers.

U kunt ook een label voor de versie opgeven met behulp van het element `versionLabel`. Wanneer u een versielabel toevoegt, wordt het in plaats van het versienummer weergegeven op plaatsen zoals het Android-toepassingsscherm. Er moet een versielabel worden opgegeven voor toepassingen die zijn gedistribueerd via de Android Market. Als u geen waarde voor `versionLabel` opgeeft in de AIR-toepassingsdescriptor, wordt de waarde voor `versionNumber` toegewezen aan het labelveld van de Android-versie.

```
<!-- AIR 2.5 and later -->
                            <versionNumber>1.23.7<versionNumber>
                            <versionLabel>1.23 Beta 7</versionLabel>
```

Op Android wordt het AIR- `versionNumber` omgezet in het gehele Android-getal `versionCode` aan de hand van de formule: `a*1000000 + b*1000 + c`, waarbij a, b en c de componenten van het AIR-versienummer zijn: `a.b.c`.

### Meer Help-onderwerpen

"version" op pagina 254

"versionLabel" op pagina 254

"versionNumber" op pagina 254

## Hoofdtoepassings-SWF

Geef het hoofdtoepassings-SWF-bestand op in het onderliggende element `content` van het element `initialWindow`. Wanneer u in het mobiele profiel een doel instelt voor apparaten, moet u een SWF-bestand gebruiken (op HTML-gebaseerde toepassingen worden niet ondersteund).

```
<initialWindow>
                            <content>MyApplication.swf</content>
                            </initialWindow>
```

U moet het bestand in het AIR-pakket opnemen (met behulp van ADT of uw IDE). Door alleen te verwijzen naar de naam in de toepassingsdescriptor wordt het bestand niet automatisch in het pakket opgenomen.

## Eigenschappen van het hoofdscherm

Verschillende onderliggende elementen van het element initialWindow bepalen de initiële weergave en de functionaliteit van het hoofdscherm van de toepassing.

- **aspectRatio** — Hiermee wordt aangegeven of de toepassing in eerste instantie *staand* (hoogte groter dan breedte) of *liggend* (hoogte kleiner dan breedte) weergegeven, of dat de indeling *elke* wordt toegepast (het werkgebied is automatisch georiënteerd op alle oriëntaties).

  ```
  <aspectRatio>landscape</aspectRatio>
  ```

- **autoOrients** — Hiermee wordt aangegeven of de richting van het werkgebied automatisch moet worden gewijzigd als de gebruiker het apparaat draait of een andere beweging maakt die betrekking heeft op richting, zoals het openen of sluiten van een uitschuifbaar toetsenbord. Bij *false*, wat de standaardinstelling is, wordt de richting van het werkgebied niet gewijzigd als het apparaat wordt gedraaid.

  ```
  <autoOrients>true</autoOrients>
  ```

- **depthAndStencil** — Of de diepte- of de stencilbuffer moet worden gebruikt. Deze buffers worden meestal gebruikt tijdens het werken met 3D-inhoud.

  ```
  <depthAndStencil>true</depthAndStencil>
  ```

- **fullScreen** — Hiermee wordt aangegeven of het volledige scherm van het apparaat voor de toepassing moet wordt gebruikt of dat het scherm moet worden gedeeld met de normale interface van het besturingssysteem, zoals een systeemstatusbalk.

  ```
  <fullScreen>true</fullScreen>
  ```

- **renderMode** — Hiermee wordt aangegeven of de runtime de toepassing moet renderen met de GPU of de CPU. Over het algemeen wordt door GPU-rendering de renderingsnelheid verhoogd, maar enkele functies, zoals bepaalde overvloeimodi en PixelBender-filters zijn niet beschikbaar in de GPU-modus. Daarnaast hebben verschillende apparaten en verschillende stuurprogramma's van apparaten verschillende GPU-capaciteit en beperkingen van de GPU. U moet uw toepassing altijd op zoveel mogelijk verschillende apparaten testen, vooral wanneer u de GPU-modus gebruikt.

  U kunt de renderingsmodus instellen op *gpu*, *cpu*, *direct* of *auto*. De standaardwaarde is *auto*, die momenteel wordt teruggezet in de cpu-modus.

  ***Opmerking:*** *voor een optimale GPU-versnelling van Flash-inhoud met AIR for Mobile-platforms raadt Adobe u aan om de instelling renderMode="direct" te gebruiken (dat wil zeggen: Stage3D) in plaats van renderMode="gpu". Adobe biedt officieel ondersteuning voor de volgende Stage3D-frameworks: Starling (2D) en Away3D (3D). Adobe raadt u daarom aan deze frameworks te gebruiken. Voor meer informatie over Stage3D en Starling/Away3D gaat u naar http://gaming.adobe.com/getstarted/.*

  ```
  <renderMode>direct</renderMode>
  ```

  ***Opmerking:*** *u kunt renderMode="direct" niet gebruiken voor toepassingen die op de achtergrond worden uitgevoerd.*

  De beperkingen van de GPU-modus zijn:

  - Het Flex-framework biedt geen ondersteuning voor de GPU-renderingsmodus.
  - Filters worden niet ondersteund
  - PixelBender-overvloeiing en opvulling wordt niet ondersteund
  - De volgende overvloeimodi worden niet ondersteund: laag, alfa, wissen, bedekking, fel licht, lichter en donkerder
  - Het wordt niet aanbevolen een GPU-renderingsmodus te gebruiken in een toepassing die video afspeelt.

- In de GPU-renderingmodus worden tekstvelden niet altijd naar een zichtbare locatie verplaatst wanneer het virtuele toetsenbord wordt geopend. Gebruik de eigenschap softKeyboardRect van het werkgebied en elektronische-toetsenbordgebeurtenissen om het tekstveld naar het zichtbare gebied te verplaatsen om ervoor te zorgen dat het zichtbaar is als gebruikers tekst invoeren.
- Als een weergaveobject niet door de GPU kan worden gerenderd, wordt het helemaal niet weergegeven. Als een filter bijvoorbeeld op een weergaveobject wordt toegepast, wordt het object niet weergegeven.

***Opmerking:*** *de GPU-implementatie voor iOS in AIR 2.6+ wijkt sterk af van de implementatie die wordt gebruikt in de vorige versie, AIR 2.0. Er zijn verschillende optimalisatieoverwegingen van toepassing.*

### Meer Help-onderwerpen

"aspectRatio" op pagina 224

"autoOrients" op pagina 224

"depthAndStencil" op pagina 228

"fullScreen" op pagina 236

"renderMode" op pagina 248

## Ondersteunde profielen

U kunt het element `supportedProfiles` toevoegen om op te geven welke apparaatprofielen door uw toepassing worden ondersteund. Gebruik het profiel mobileDevice voor mobiele apparaten. Wanneer u uw toepassing uitvoert met de Adobe Debug Launcher (ADL), gebruikt ADL het eerste profiel in de lijst als het actieve profiel. U kunt ook de markering `-profile` bij het uitvoeren van ADL gebruiken om een bepaald profiel in de ondersteunde lijst te selecteren. Als uw toepassing onder alle profielen wordt uitgevoerd, kunt u het element `supportedProfiles` helemaal weglaten. ADL gebruikt in dit geval het profiel desktop als het profiel dat standaard actief is.

Als u wilt opgeven dat uw toepassing zowel de profielen voor mobiele apparaten als desktop ondersteunt, en als u doorgaans de toepassing in het mobiele profiel wilt testen, moet u het volgende element toevoegen:

```
<supportedProfiles>mobileDevice desktop</supportedProfiles>
```

### Meer Help-onderwerpen

"supportedProfiles" op pagina 251

"Apparaatprofielen" op pagina 257

"ADL (AIR Debug Launcher)" op pagina 167

## Vereiste native extensies

Toepassingen die het profiel `mobileDevice` ondersteunen, kunnen native extensies gebruiken.

Declareer alle native extensies die de AIR-toepassing gebruikt in het descriptorbestand van de toepassing. Het volgende voorbeeld illustreert de syntaxis voor het opgeven van twee vereiste native extensies:

```
<extensions>
                            <extensionID>com.example.extendedFeature</extensionID>
                            <extensionID>com.example.anotherFeature</extensionID>
                            </extensions>
```

Het `extensionID`-element heeft dezelfde waarde als het element `id` in het descriptorbestand van de extensie. Het descriptorbestand van de extensie is een XML-bestand met de naam "extension.xml". Het wordt verpakt in het ANE-bestand dat u ontvangt van de ontwikkelaar van de native extensie.

### Functionaliteit van het virtuele toetsenbord

Stel het element `softKeyboardBehavior` in op `none` om de automatische functionaliteit voor pannen en vergroten/verkleinen uit te schakelen die door de runtime wordt gebruikt om te garanderen dat het relevante tekstinvoerveld zichtbaar is nadat het virtuele toetsenbord wordt weergegeven. Als u de automatische functionaliteit uitschakelt, moet door de toepassing worden gezorgd dat het tekstinvoergedeelte of overige relevante inhoud zichtbaar blijft als het toetsenbord wordt weergegeven. U kunt de eigenschap `softKeyboardRect` van het werkgebied in combinatie met de SoftKeyboardEvent gebruiken om te detecteren wanneer het toetsenbord wordt geopend en het gedeelte te bepalen dat door het toetsenbord wordt verborgen.

Voor het inschakelen van de automatische functionaliteit stelt u de waarde van het element in op `pan`:

```
<softKeyboardBehavior>pan</softKeyboardBehavior>
```

Omdat `pan` de standaardwaarde is, wordt door het weglaten van het element `softKeyboardBehavior` eveneens de automatische toetsenbordfunctionaliteit ingeschakeld.

***Opmerking:*** *wanneer u ook GPU-rendering gebruikt, wordt de functionaliteit voor pannen niet ondersteund.*

### Meer Help-onderwerpen

"softKeyboardBehavior" op pagina 250

Stage.softKeyboardRect

SoftKeyboardEvent

## Android-instellingen

Op het Android-platform kunt u het element `android` van de toepassingsdescriptor gebruiken om informatie toe te voegen aan het manifest van de Android-toepassing, dat een toepassingseigenschappenbestand is dat wordt gebruikt door het besturingssysteem Android. Het .xml-bestand van het Android-manifest wordt automatisch door ADT gegenereerd wanneer u het APK-pakket maakt. Door AIR worden enkele eigenschappen ingesteld voor de waarden die worden vereist zodat bepaalde functies werken. Overige eigenschappen die worden ingesteld in het Android-gedeelte van de AIR-toepassingsdescriptor, worden toegevoegd aan het corresponderende gedeelte van het .xml-bestand van het manifest.

***Opmerking:*** *voor de meeste AIR-toepassingen moet u de Android-machtigingen die voor uw toepassing nodig zijn, instellen in het element* `android`*, maar doorgaans hoeft u geen andere eigenschappen in te stellen.*

U kunt alleen attributen instellen met tekenreekswaarden, integere waarden of Boole-waarden. Het instellen van verwijzingen naar bronnen in het toepassingspakket wordt niet ondersteund.

***Opmerking:*** *Voor de runtime wordt minimaal een SDK-versie vereist die gelijk is aan of groter dan 14. Als u een toepassing voor alleen de hogere versies wilt maken, moet u ervoor zorgen dat het volgende is opgenomen in het manifest:* `<uses-sdk android:minSdkVersion=""></uses-sdk>`*, waarbij respectievelijk de juiste versie wordt aangeduid.*

### Gereserveerde instellingen voor het Android-manifest

Door AIR worden verschillende manifestvermeldingen in het gegenereerde Android-manifestdocument ingesteld om te garanderen dat de toepassings- en runtimefuncties correct werken. U kunt de volgende instellingen niet definiëren:

**element manifest**

U kunt de volgende attributen niet instellen voor het element manifest:

- package

- android:versionCode
- android:versionName
- xmlns:android

**activity-element**
U kunt de volgende attributen niet instellen voor het hoofdelement activity:

- android:label
- android:icon

**application-element**
U kunt de volgende attributen niet instellen voor het element application:

- android:theme
- android:name
- android:label
- android:windowSoftInputMode
- android:configChanges
- android:screenOrientation
- android:launchMode

## Android-machtigingen

Het wordt door het Android-beveiligingsmodel vereist dat door elke toepassing een machtiging wordt gevraagd om functies te kunnen gebruiken die gevolgen voor de beveiliging of de privacy hebben. Deze machtigingen moeten worden opgegeven wanneer de toepassing wordt verpakt en kunnen niet worden gewijzigd tijdens het uitvoeren. De gebruikers worden door het Android-besturingssysteem op de hoogte gesteld welke machtiging door een toepassing wordt gevraagd en deze wordt door de gebruiker geïnstalleerd. Als voor een functie de vereiste machtiging niet wordt gevraagd, retourneert het Android-besturingssysteem mogelijk een uitzondering wanneer de functie door uw toepassing wordt geopend, maar een uitzondering is niet gegarandeerd. Uitzonderingen worden door de runtime naar uw toepassing verzonden. In het geval van een fout zonder foutmelding wordt een machtigingsfout toegevoegd aan het Android-systeemlogboek.

In AIR kunt u de Android-machtigingen opgeven in het element `android` van uw toepassingsdescriptor. De volgende indeling wordt gebruikt voor het toevoegen van machtigingen (wanneer PERMISSION_NAME de naam van een Android-machtiging is):

```
<android>
                            <manifestAdditions>
                            <![CDATA[
                            <manifest>
                            <uses-permission
android:name="android.permission.PERMISSION_NAME" />
                            </manifest>
                            ]]>
                            </manifestAdditions>
                            </android>
```

De instructies voor gebruiksmachtigingen in het element `manifest` worden direct toegevoegd aan het document van het Android-manifest.

De volgende machtigingen worden vereist om verschillende AIR-functies te kunnen gebruiken:

**ACCESS_COARSE_LOCATION** Hiermee geeft u de toepassing via de klasse Geolocation toegang tot locatiegevens van het WiFi- en mobiele-telefoonnetwerk.

**ACCESS_FINE_LOCATION** Hiermee krijgt de toepassing via de klasse Geolocation toegang tot GPS-gegevens.

**ACCESS_NETWORK_STATE and ACCESS_WIFI_STATE** Hiermee geeft u de toepassing via de klasse NetworkInfo toegang tot netwerkgegevens.

**CAMERA** Hiermee krijgt de toepassing toegang tot de camera.

***Opmerking:*** *wanneer u toestemming vraagt om de camerafunctie te gebruiken, gaat Android ervan uit dat de camera ook voor uw toepassing wordt vereist. Als de camera een optionele functie van uw toepassing is, moet u een element* `uses-feature` *aan het manifest voor de camera toevoegen, waardoor het vereiste attribuut wordt ingesteld op* `false`*. Zie "Filtering van Android-compatibiliteit" op pagina 84.*

**INTERNET** Hiermee kan de toepassing netwerkverzoeken indienen en wordt foutopsporing op afstand mogelijk gemaakt.

**READ_PHONE_STATE** Hiermee kan de AIR-runtime audio dempen tijdens telefoongesprekken. U moet deze machtiging instellen wanneer uw toepassing geluid afspeelt als deze op de achtergrond wordt uitgevoerd.

**RECORD_AUDIO** Hiermee krijgt de toepassing toegang tot de microfoon.

**WAKE_LOCK en DISABLE_KEYGUARD** Hiermee voorkomt de toepassing via de instellingen voor de klasse SystemIdleMode dat het apparaat inactief wordt.

**WRITE_EXTERNAL_STORAGE** Hiermee kan de toepassing naar de externe geheugenkaart op het apparaat schrijven.

U kunt bijvoorbeeld, als u machtigingen wilt instellen voor een toepassing waarvoor alle machtigingen worden vereist, het volgende toevoegen aan de toepassingsdescriptor:

```
<android>
                            <manifestAdditions>
                            <![CDATA[
                            <manifest>
                            <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
                            <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
                            <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
                            <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
                            <uses-permission android:name="android.permission.CAMERA" />
                            <uses-permission
android:name="android.permission.DISABLE_KEYGUARD" />
                            <uses-permission android:name="android.permission.INTERNET" />
                            <uses-permission
android:name="android.permission.READ_PHONE_STATE" />
                            <uses-permission android:name="android.permission.RECORD_AUDIO"
/>
                            <uses-permission android:name="android.permission.WAKE_LOCK" />
                            <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
                            </manifest>
                            ]]>
                            </manifestAdditions>
                            </android>
```

### Meer Help-onderwerpen

Beveiliging en machtigingen van Android

Android-klasse Manifest.permission

## Aangepaste URI-schema's van Android

U kunt een aangepast URI-schema gebruiken om een AIR-toepassing op te starten op een webpagina of een native Android-toepassing. Ondersteuning van een aangepaste URI is afhankelijk van de intentfilters die worden opgegeven in het Android-manifest. Deze techniek kan dus niet worden gebruikt op overige platforms.

Als u een aangepaste URI wilt gebruiken, moet u een intentfilter toevoegen aan de toepassingsdescriptor in het blok `<android>`. Beide elementen `intent-filter` in het volgende voorbeeld moeten worden opgegeven. De instructie `<data android:scheme="my-customuri"/>` bewerken om de URI-tekenreeks weer te geven voor uw aangepaste schema.

```
<android>
                            <manifestAdditions>
                            <![CDATA[
                            <manifest>
                            <application>
                            <activity>
                            <intent-filter>
                            <action android:name="android.intent.action.MAIN"/>
                            <category android:name="android.intent.category.LAUNCHER"/>
                            </intent-filter>
                            <intent-filter>
                            <action android:name="android.intent.action.VIEW"/>
                            <category android:name="android.intent.category.BROWSABLE"/>
                            <category android:name="android.intent.category.DEFAULT"/>
                            <data android:scheme="my-customuri"/>
                            </intent-filter>
                            </activity>
                            </application>
                            </manifest>
                            ]]>
                            </manifestAdditions>
                            </android>
```

Een intentfilter geeft informatie aan het Android-besturingssysteem over het feit dat uw toepassing beschikbaar is om een bepaalde actie uit te voeren. Bij een aangepaste URI betekent dit dat de gebruiker op een koppeling heeft geklikt die dat URI-schema gebruikt (en die de browser niet kan verwerken).

Wanneer uw toepassing wordt opgeroepen via een custom-URL, wordt door het NativeApplication-object een gebeurtenis `invoke` verzonden. De URL van de koppeling, inclusief queryparameters, wordt geplaatst in de array `arguments` van het object InvokeEvent. U kunt elk gewenst aantal intentfilters gebruiken.

***Opmerking:*** *koppelingen in een StageWebView-instantie kunnen geen URL's openen die een aangepast URI-schema gebruiken.*

### Meer Help-onderwerpen

Intentfilters van Android

Acties en categorieën van Android

## Filtering van Android-compatibiliteit

Het Android-besturingssysteem gebruikt een aantal elementen in het toepassingsmanifest om te bepalen of uw toepassing compatibel is met een bepaald apparaat. Het is optioneel om deze gegevens aan het manifest toe te voegen. Als u deze gegevens niet opneemt, kan uw toepassing op alle Android-apparaten worden geïnstalleerd. De toepassing werkt echter mogelijk niet correct op alle Android-apparaten. Een cameratoepassing is bijvoorbeeld niet erg nuttig op een telefoon zonder camera.

De tags van het Android-manifest die u kunt gebruiken voor filtering zijn onder meer:

- supports-screens
- uses-configuration
- uses-feature
- uses-sdk (in AIR 3+)

### Camera-toepassingen

Als u een cameramachtiging voor uw toepassing vraagt, gaat Android ervan uit dat voor de toepassing alle beschikbare camerafuncties worden vereist, inclusief automatisch scherpstellen en flitsen. Als voor uw toepassing niet alle beschikbare camerafuncties worden vereist, moet u de verschillende elementen `uses-feature` voor de camera instellen om aan te geven dat deze optioneel zijn. Als dit niet wordt gedaan, kunnen gebruikers met apparaten waarop één functie ontbreekt of waarop zich helemaal geen camera bevindt, uw toepassing niet vinden op de Android Market.

In het volgende voorbeeld wordt weergegeven hoe een machtiging voor de camera wordt gevraagd en hoe alle camerafuncties optioneel moeten worden gemaakt:

```
<android>
                                <manifestAdditions>
                                <![CDATA[
                                <manifest>
                                <uses-permission android:name="android.permission.CAMERA" />
                                <uses-feature android:name="android.hardware.camera"
android:required="false"/>
                                <uses-feature
android:name="android.hardware.camera.autofocus" android:required="false"/>
                                <uses-feature android:name="android.hardware.camera.flash"
android:required="false"/>
                                </manifest>
                                ]]>
                                </manifestAdditions>
                                </android>
```

### Toepassingen voor geluidsopnamen

Als u de machtiging vraagt voor het maken van geluidsopnamen, gaat Android er ook vanuit dat voor de toepassing een microfoon wordt gebruikt. Als het maken van geluidsopnamen een optionele functie van uw toepassing is, kunt u een tag 'uses-feature' toevoegen om aan te geven dat de microfoon niet wordt vereist. Als u dit niet doet, kunnen gebruikers met apparaten zonder microfoon uw toepassing niet op de Android Market vinden.

In het volgende voorbeeld wordt weergegeven hoe een machtiging voor het gebruik van de microfoon wordt gevraagd wanneer de microfoonhardware nog steeds optioneel blijft:

```
<android>
                                <manifestAdditions>
                                <![CDATA[
                                <manifest>
                                <uses-permission
android:name="android.permission.RECORD_AUDIO" />
                                <uses-feature android:name="android.hardware.microphone"
android:required="false"/>
                                </manifest>
                                ]]>
                                </manifestAdditions>
                                </android>
```

### Meer Help-onderwerpen

Android-ontwikkelaars: Android-compatibiliteit

Android-ontwikkelaars: constanten Android-functienaam

## Installatielocatie

U kunt toestaan dat uw toepassing op de externe geheugenkaart wordt geïnstalleerd of ernaar wordt verplaatst, door het attribuut `installLocation` van het Android-element `manifest` in te stellen op `auto` of `preferExternal`:

```
<android>
                            <manifestAdditions>
                            <![CDATA[
                            <manifest android:installLocation="preferExternal"/>
                            ]]>
                            </manifestAdditions>
                            </android>
```

Door het Android-besturingssysteem wordt niet gegarandeerd dat uw toepassing in het externe geheugen wordt geïnstalleerd. Een gebruiker kan de toepassing ook van intern naar extern geheugen verplaatsen met behulp van de

Zelfs wanneer deze in een extern geheugen zijn geïnstalleerd, worden de toepassingscache en gebruikersgegevens, zoals de inhoud van een map voor de toepassingsopslag, gedeelde objecten en tijdelijke bestanden, ook nog in het interne geheugen opgeslagen. Als u wilt voorkomen dat er te veel intern geheugen wordt gebruikt, moet u selectief zijn met betrekking tot de gegevens die u opslaat in de map voor de toepassingsopslag. Grote hoeveelheden gegevens moeten op de SD-kaart worden opgeslagen door de locatie `File.userDirectory` of `File.documentsDirectory` te gebruiken (beide locaties leiden bij Android naar de hoofdmap van de SD-kaart).

## Flash Player en andere insteekmodules inschakelen in een StageWebView-object

In Android 3.0+ moet een toepassing de hardwareversnelling inschakelen in het Android application-element om de inhoud van een insteekmodule weer te geven in een StageWebView-object. Als u de weergave van inhoud van een insteekmodule wilt inschakelen, stelt u het `android:hardwareAccelerated`-kenmerk van het `toepassings` element in op `true`:

```
<android>
                            <manifestAdditions>
                            <![CDATA[
                            <manifest>
                            <application android:hardwareAccelerated="true"/>
                            </manifest>
                            ]]>
                            </manifestAdditions>
                            </android>
```

### Kleurdiepte

**AIR 3+**

In AIR 3 en later stelt de runtime de weergave in op het renderen van 32-bits kleuren. In eerdere versies van AIR maakt de runtime gebruik van 16-bits kleur. U kunt de runtime via het element <colorDepth> van het descriptorbestand van de toepassing opdracht geven 16-bits kleur te gebruiken:

```
<android>
                            <colorDepth>16bit</colorDepth>
                            <manifestAdditions>...</manifestAdditions>
                            </android>
```

Het gebruik van de 16-bits kleurdiepte kan de renderprestaties verbeteren, maar dat gaat ten koste van de kleurgetrouwheid.

## iOS-instellingen

Instellingen die alleen van toepassing zijn op iOS-apparaten worden in het `<iPhone>`-element in het descriptorbestand van de toepassing geplaatst. Het `iPhone`-element kan de volgende onderliggende elementen hebben: een `InfoAdditions`-element, een `requestedDisplayResolution`-element, een `Entitlements`-element, een `externalSwfs`-element en een `forceCPURenderModeForDevices`-element.

Met het `InfoAdditions`-element kunt u specifieke sleutel-waardeparen opgeven die worden toegevoegd aan het instellingenbestand Info.plist voor de toepassing. De volgende waarden stellen bijvoorbeeld de stijl van de statusbalk van de toepassing in en geven aan dat voor de toepassing geen permanente Wi-Fi-toegang is vereist.

```
<InfoAdditions>
                            <![CDATA[
                            <key>UIStatusBarStyle</key>
                            <string>UIStatusBarStyleBlackOpaque</string>
                            <key>UIRequiresPersistentWiFi</key>
                            <string>NO</string>
                            ]]>
                            </InfoAdditions>
```

De InfoAdditions-instellingen worden opgenomen in een `CDATA`-tag.

Met het `Entitlements`-element kunt u specifieke sleutelwaardeparen opgeven die worden toegevoegd aan het instellingenbestand Entitlements.plist voor de toepassing. Entitlements.plist-instellingen zorgen ervoor dat de toepassing toegang krijgt tot bepaalde iOS-functies, zoals de functie voor pushberichten.

Zie de documentatie voor ontwikkelaars van Apple voor gedetailleerde informatie over Info.plist- en Entitlements.plist-instellingen.

### Ondersteuning voor achtergrondtaken in iOS

**AIR 3.3**

Adobe AIR 3.3 en hoger biedt ondersteuning voor multitasking in iOS door bepaalde achtergrondfuncties mogelijk te maken:

- Audio
- Locatie-updates
- Netwerken
- Uitvoering van apps op achtergrond uitschakelen

***Opmerking:*** *in SWF-versie 21 en eerder biedt AIR op iOS en Android geen ondersteuning voor achtergronduitvoering als de rendermodus Direct is ingesteld. Daarom kunnen op Stage3D gebaseerde apps geen achtergrondtaken uitvoeren, zoals het afspelen van audio, het bijwerken van de locatie, het uploaden en downloaden naar en van het netwerk, enz. Op iOS is het uitvoeren van OpenGLES-/renderingaanroepen op de achtergrond niet toegestaan. Toepassingen die proberen OpenGL-aanroepen uit te voeren op de achtergrond worden door iOS beëindigd. Android beperkt toepassingen niet in het uitvoeren van OpenGLES-aanroepen op de achtergrond of in het uitvoeren van andere achtergrondtaken, zoals het afspelen van audio. In SWF-versie 22 en hoger kunnen mobiele AIR-toepassingen op de achtergrond worden uitgevoerd wanneer de renderMode Direct is ingesteld. De AIR iOS-runtime resulteert in een ActionScript-fout (3768 - De Stage3D API kan niet worden gebruikt tijdens uitvoering op de achtergrond) als OpenGLES-aanroepen worden uitgevoerd op de achtergrond. Op Android is echter geen sprake van fouten, omdat native toepassingen OpenGLES-aanvragen kunnen uitvoeren op de achtergrond. Voor optimaal gebruik van mobiele resources kunt u beter geen renderingaanroepen uitvoeren als een toepassing op de achtergrond wordt uitgevoerd.*

**Achtergrondaudio**

Neem het volgende sleutelwaardepaar op in het element `InfoAdditions` om het afspelen en opnemen van achtergrondaudio in te schakelen:

```
<InfoAdditions>
                                <![CDATA[
                                <key>UIBackgroundModes</key>
                                <array>
                                <string>audio</string>
                                </array>
                                ]]>
                                </InfoAdditions>
```

**Locatie-updates op de achtergrond**

Neem het volgende sleutelwaardepaar op in het element `InfoAdditions` om locatie-updates op de achtergrond in te schakelen:

```
<InfoAdditions>
                                <![CDATA[
                                <key>UIBackgroundModes</key>
                                <array>
                                <string>location</string>
                                </array>
                                ]]>
                                </InfoAdditions>
```

***Opmerking:*** *gebruik deze functie alleen als het nodig is, aangezien locatie-API's veel van de batterij vergen.*

**Netwerktaken op de achtergrond**

De eigenschap `NativeApplication.nativeApplication.executeInBackground` wordt door de toepassing ingesteld op `true` om korte taken op de achtergrond te kunnen uitvoeren.

Het kan bijvoorbeeld wel eens voorkomen dat uw toepassing begint met het uploaden van een bestand, waarna de gebruiker een andere toepassing op de voorgrond plaatst. Als de toepassing een gebeurtenis ontvangt ten tekenen dat het uploaden is voltooid, kan deze `NativeApplication.nativeApplication.executeInBackground` instellen op `false`.

Het instellen van de eigenschap `NativeApplication.nativeApplication.executeInBackground` op `true` betekent niet dat de toepassing voor onbeperkte tijd wordt uitgevoerd, aangezien iOS een tijdlimiet oplegt aan achtergrondtaken. Als iOS de achtergrondverwerking stopzet, verzendt AIR de gebeurtenis `NativeApplication.suspend`.

**Uitvoering op achtergrond uitschakelen**

Uw toepassing kan er expliciet voor kiezen het uitvoeren op de achtergrond uit te schakelen door het volgende sleutelwaardepaar op te nemen in het element `InfoAdditions`:

```
<InfoAdditions>
                                <![CDATA[
                                <key>UIApplicationExitsOnSuspend</key>
                                <true/>
                                ]]>
                                </InfoAdditions>
```

## Gereserveerde InfoAdditions-instellingen voor iOS

Door AIR worden verschillende vermeldingen in het gegenereerde Info.plist-bestand ingesteld om te garanderen dat de toepassings- en runtimefuncties correct werken. U kunt de volgende instellingen niet definiëren:

| | |
|---|---|
| CFBundleDisplayName | CTInitialWindowTitle |
| CFBundleExecutable | CTInitialWindowVisible |
| CFBundleIconFiles | CTIosSdkVersion |
| CFBundleIdentifier | CTMaxSWFMajorVersion |
| CFBundleInfoDictionaryVersion | DTPlatformName |
| CFBundlePackageType | DTSDKName |
| CFBundleResourceSpecification | MinimumOSVersion (gereserveerd tot 3.2) |
| CFBundleShortVersionString | NSMainNibFile |
| CFBundleSupportedPlatforms | UIInterfaceOrientation |
| CFBundleVersion | UIStatusBarHidden |
| CTAutoOrients | UISupportedInterfaceOrientations |

***Opmerking:*** *U kunt MinimumOSVersion definiëren. De MinimumOSVersion-definitie wordt ondersteund in Air 3.3 en hoger.*

## Verschillende modellen iOS-apparaten ondersteunen

Neem voor iPad-ondersteuning de juiste sleutel-waarde-instellingen voor `UIDeviceFamily` op in het `InfoAdditions`-element. De instelling `UIDeviceFamily` is een array van tekenreeksen. In elke tekenreeks worden ondersteunde apparaten gedefinieerd. De instelling `<string>1</string>` definieert ondersteuning voor de iPhone en de iPod touch. De instelling `<string>2</string>` definieert ondersteuning voor de iPad. De instelling `<string>3</string>` definieert ondersteuning voor de tvOS. Als u slechts één van deze tekenreeksen definieert, wordt alleen de desbetreffende apparaatfamilie ondersteund. Bij de volgende instelling is er bijvoorbeeld alleen ondersteuning voor de iPad:

```
<key>UIDeviceFamily</key>
                            <array>
                            <string>2</string>
                            </array>>
```

En bij deze instelling is er ondersteuning voor beide apparaatfamilies (de iPhone/iPod touch en iPad):

```
<key>UIDeviceFamily</key>
                            <array>
                            <string>1</string>
                            <string>2</string>
                            </array>
```

Bovendien kunt u in AIR 3.7 en hoger de tag `forceCPURenderModeForDevices` gebruiken om de CPU-rendermodus te forceren voor een opgegeven serie apparaten en de GPU-rendermodus in te schakelen voor de resterende iOS-apparaten.

U voegt deze tag toe als een onderliggend element van de `iPhone`-tag en geeft een lijst op waarin de modelnamen van apparaten met een spatie van elkaar zijn gescheiden. Zie "forceCPURenderModeForDevices" op pagina 235 voor een lijst met geldige modelnamen van apparaten.

Geef bijvoorbeeld het volgende op in het descriptorbestand van de toepassing als u de CPU-modus wilt gebruiken in oudere iPods, iPhones en iPads en de GPU-modus voor alle andere apparaten:

```
...
                            <renderMode>GPU</renderMode>
                            ...
                            <iPhone>
                            ...
                               <forceCPURenderModeForDevices>iPad1,1 iPhone1,1 iPhone1,2
iPod1,1
                               </forceCPURenderModeForDevices>
                            </iPhone>
```

## Weergave met hoge resolutie

Het element `requestedDisplayResolution` geeft op of uw toepassing de modus voor *standaard* of *hoge* resolutie moet gebruiken op iOS-apparaten met hoge-resolutieschermen.

```
<requestedDisplayResolution>high</requestedDisplayResolution>
```

In de hoge-resolutiemodus kunt u elke pixel op een scherm met hoge resolutie afzonderlijk verwerken. In de standaardmodus komt het apparaatscherm op uw toepassing over als een scherm met standaardresolutie. Wanneer u één pixel tekent in deze modus, wordt de kleur van vier pixels op het scherm met hoge resolutie ingesteld.

De standaardinstelling is `standard`. Merk op dat u voor het plaatsen van iOS-apparaten het `requestedDisplayResolution`-element gebruikt als een onderliggend element van het `iPhone`-element (niet het `InfoAdditions`-element of `initialWindow`-element).

Als u verschillende instellingen wilt gebruiken op verschillende apparaten, geeft u uw standaardwaarde op als de waarde van het `requestedDisplayResolution`-element. Gebruik het `excludeDevices`-attribuut om apparaten op te geven die de tegenovergestelde waarde moeten gebruiken. Met de volgende code, bijvoorbeeld, wordt de modus voor hoge resolutie gebruikt voor alle apparaten die deze ondersteunen, behalve iPad's van de derde generatie, die de standaardmodus gebruiken:

```
<requestedDisplayResolution excludeDevices="iPad3">high</requestedDisplayResolution>
```

Het `excludeDevices`-attribuut is beschikbaar in AIR 3.6 en later.

### Meer Help-onderwerpen

"requestedDisplayResolution" op pagina 248

Renaun Erickson: Developing for both retina and non-retina iOS screens using AIR 2.6

## Aangepaste URI-schema's voor iOS

U kunt een aangepast URI-schema registreren, zodat uw toepassing kan worden aangeroepen door een koppeling op een webpagina of in een andere, native toepassing op het apparaat. U registreert een URI-schema door een CFBundleURLTypes-sleutel toe te voegen aan het InfoAdditions-element. Het volgende voorbeeld registreert het URI-schema *com.example.app*, zodat een toepassing kan worden aangeroepen door URL's met de notatie: *example://foo*.

```
<key>CFBundleURLTypes</key>
                            <array>
                            <dict>
                            <key>CFBundleURLSchemes</key>
                            <array>
                            <string>example</string>
                            </array>
                            <key>CFBundleURLName</key>
                            <string>com.example.app</string>
                            </dict>
                            </array>
```

Wanneer uw toepassing wordt opgeroepen via een custom-URL, wordt door het NativeApplication-object een gebeurtenis `invoke` verzonden. De URL van de koppeling, inclusief queryparameters, wordt geplaatst in de array `arguments` van het object InvokeEvent. U kunt een willekeurig aantal aangepaste URI-schema's gebruiken.

***Opmerking:*** *koppelingen in een StageWebView-instantie kunnen geen URL's openen die een aangepast URI-schema gebruiken.*

***Opmerking:*** *als een andere toepassing al een schema heeft geregistreerd, kan uw toepassing de eerste toepassing niet vervangen als de voor het desbetreffende URI-schema geregistreerde toepassing.*

## Filteren op iOS-compatibiliteit

Voeg vermeldingen toe aan een UIRequiredDeviceCapabilities-array in het `InfoAdditions`-element als uw toepassing alleen gebruikt mag worden op apparaten met specifieke hardware- of softwarefunctionaliteit. De volgende vermelding geeft bijvoorbeeld aan dat een toepassing een fototoestel en een microfoon nodig heeft:

```
<key>UIRequiredDeviceCapabilities</key>
                            <array>
                            <string>microphone</string>
                            <string>still-camera</string>
                            </array>
```

De toepassing kan niet worden geïnstalleerd als een apparaat daar niet over beschikt. Enkele voor AIR-toepassingen relevante functionaliteitsinstellingen:

| | |
|---|---|
| telefonie | camera met flits |
| Wi-Fi | videocamera |
| sms | versnellingsmeter |
| fototoestel | locatieservices |
| automatisch scherpstellende camera | GPS |
| voorwaarts gerichte camera | microfoon |

AIR 2.6+ voegt *armv7* en *opengles-2* automatisch toe aan de lijst met vereiste functionaliteit.

***Opmerking:*** *u hoeft deze functionaliteit niet op te nemen in het descriptorbestand van de toepassing, uw toepassing kan deze ook benutten als dat niet het geval is. Gebruik de UIRequiredDeviceCapabilities -instellingen alleen om te voorkomen dat gebruikers uw toepassing installeren op apparaten waarop deze niet goed kan functioneren.*

## Afsluiten in plaats van onderbreken

Wanneer een gebruiker bij een AIR-toepassing wegklikt, wordt de toepassing op de achtergrond geplaatst en onderbroken. Als u de toepassing volledig wilt afsluiten in plaats van te onderbreken, stelt u de eigenschap `UIApplicationExitsOnSuspend` in op `YES`:

```
<key>UIApplicationExitsOnSuspend</key>
                            <true/>
```

## Downloadgrootte minimaliseren door externe SWF-bestanden met uitsluitend elementen te laden

**AIR 3.7**

U kunt de downloadgrootte van de aanvankelijke toepassing minimaliseren door een subset van de door uw toepassing gebruikte SWF-bestanden te verpakken en de resterende externe SWF-bestanden (met alleen elementen) tijdens de runtime te laden met behulp van de methode `Loader.load ()`. Als u deze functie wilt gebruiken, dient u de toepassing zodanig te verpakken dat ADT alle ActionScript ByteCode (ABC) uit de extern geladen SWF-bestanden naar de SWF-hoofdtoepassing verplaatst, zodat er een SWF-bestand met uitsluitend elementen overblijft. Zo wordt voldaan aan de regel van de Apple Store die het downloaden van code verbiedt nadat een toepassing is geïnstalleerd.

ADT biedt op de volgende manieren ondersteuning voor extern geladen SWF-bestanden (ook wel gestripte SWF-bestanden genoemd):

- Leest het tekstbestand dat is opgegeven in het subelement `<externalSwfs>` van het element `<iPhone>` voor toegang tot de lijst met via regels van elkaar gescheiden SFW-bestanden die tijdens de uitvoering moeten worden geladen:

  ```
  <iPhone>
                                          ...

      <externalSwfs>FilewithPathsOfSWFsThatAreToNotToBePackaged.txt</externalSwfs>
                                      </iPhone>
  ```

- Brengt de ABC-code van elk extern geladen SWF-bestand over naar het hoofduitvoeringsbestand.
- Sluit extern geladen SWF-bestanden uit van het .ipa-bestand.
- Kopieert gestripte SWF-bestanden naar de map `.remoteStrippedSWFs`. U host deze SWF-bestanden op een webserver en uw toepassing laadt deze bestanden, indien nodig, tijdens de runtime.

U geeft de SWF-bestanden op die tijdens de runtime moeten worden geladen door de desbetreffende bestandsnamen op te geven in een tekstbestand, en wel één naam per regel, zoals in het volgende voorbeeld:

```
assets/Level1/Level1.swf
                            assets/Level2/Level2.swf
                            assets/Level3/Level3.swf
                            assets/Level4/Level4.swf
```

Het opgegeven bestandspad is relatief ten opzichte van het descriptorbestand van de toepassing. Daarnaast moet u deze SWF-bestanden als elementen opgeven in de opdracht `adt`.

***Opmerking:*** *deze functie is alleen van toepassing op standaardpakketten. Voor snel verpakken (bijvoorbeeld met gebruik van interpreter, foutopsporing of simulator) maakt ADT geen gestripte SWF-bestanden.*

Zie Secundaire SWF-bestanden extern hosten voor AIR-apps op iOS, een door Adobe-technicus Abhinav Dhandh geplaatst blog met meer informatie over deze functie, inclusief voorbeeldcode.

### Geolocation-ondersteuning

Voor Geolocation-ondersteuning, voegt u een van de volgende sleutelwaardeparen aan het element `InfoAdditions`:

```
<InfoAdditions>
                            <![CDATA[
                            <key>NSLocationAlwaysUsageDescription</key>
                            <string>Sample description to allow geolocation always</string>
                            <key>NSLocationWhenInUseUsageDescription</key>
                            <string>Sample description to allow geolocation when application
is in foreground</string>
                            ]]>
                            </InfoAdditions>
```

## Toepassingspictogrammen

In de volgende tabel worden de pictogramafmetingen weergegeven die elk mobiel platform worden gebruikt:

| Pictogramafmeting | Platform |
|---|---|
| 29 x 29 | iOS |
| 36 x 36 | Android |
| 40 x 40 | iOS |
| 48 x 48 | Android, iOS |
| 50 x 50 | iOS |
| 57 x 57 | iOS |
| 58 x 58 | iOS |
| 60 x 60 | iOS |
| 72 x 72 | Android, iOS |
| 75x75 | iOS |
| 76 x 76 | iOS |
| 80 x 80 | iOS |
| 87x87 | iOS |
| 96 x 96 | Android |
| 100 x 100 | iOS |
| 114x114 | iOS |
| 120 x 120 | iOS |
| 144x144 | Android, iOS |
| 152 x 152 | iOS |
| 167 x 167 | iOS |
| 180 x 180 | iOS |

| Pictogramafmeting | Platform |
|---|---|
| 192 x 192 | Android |
| 512 x 512 | Android, iOS |
| 1024x1024 | iOS |

Geef het pad naar de pictogrambestanden op in het pictogramelement van het descriptorbestand van de toepassing:

```
<icon>
                        <image36x36>assets/icon36.png</image36x36>
                        <image48x48>assets/icon48.png</image48x48>
                        <image72x72>assets/icon72.png</image72x72>
                        </icon>
```

Als u geen pictogram van een bepaalde grootte aanlevert, wordt de volgende grootte gebruikt en wordt de afbeelding passend gemaakt.

### Pictogrammen bij Android

Bij Android worden de pictogrammen die worden opgegeven in de toepassingsdescriptor, gebruikt als het opstartpictogram. Het opstartpictogram dient te worden geleverd als een aantal PNG-afbeeldingen van 36 x 36, 48 x 48, 72 x 72, 96 x 96, 144 x 144 en 192 x 192 pixels. De pictogramafmetingen worden gebruikt voor schermen met respectievelijk lage dichtheid, normale dichtheid en hoge dichtheid.

Ontwikkelaars moeten het pictogram van 512 x 512 pixels indienen op het moment dat de app bij Google Play Store wordt ingediend.

### Pictogrammen voor iOS

De pictogrammen die in de toepassingsdescriptor worden gedefinieerd, worden op de volgende plaatsen gebruikt voor een iOS-toepassing:

- Een pictogram van 29 bij 29 pixels: Spotlight-zoekpictogram voor iPhones/iPods met lagere resolutie en Settings-pictogram voor iPads met lagere resolutie.
- Een pictogram van 40 bij 40 pixels: Spotlight-zoekpictogram voor iPads met lagere resolutie.
- Een pictogram van 48 bij 48 pixels: AIR voegt een rand toe aan deze afbeelding en gebruikt deze als pictogram van 50 bij 50 voor een Spotlight-zoekopdracht op iPads met lagere resolutie.
- Een pictogram van 50 bij 50 pixels: Spotlight-zoekopdracht voor iPads met lagere resolutie.
- Een pictogram van 57 bij 57 pixels: toepassingspictogram voor iPhones/iPods met lagere resolutie.
- Een pictogram van 58 bij 58 pixels: Spotlight-pictogram voor iPhones/iPods met netvliesweergave en Settings-pictogram voor iPads met netvliesweergave.
- Een pictogram van 60 bij 60 pixels: toepassingspictogram voor iPhones/iPods met lagere resolutie.
- Een pictogram van 72 bij 72 pixels (optioneel): toepassingspictogram voor iPad's met lagere resolutie.
- Een pictogram van 76 bij 76 pixels (optioneel): toepassingspictogram voor iPad's met lagere resolutie.
- Een pictogram van 80 bij 80 pixels: Spotlight-zoekpictogram voor iPhones/iPods/iPads met hogere resolutie.
- Een pictogram van 100 bij 100 pixels: Spotlight-zoekopdracht voor iPads met netvliesweergave.
- Een pictogram van 114 bij 114 pixels: toepassingspictogram voor iPhones/iPods met netvliesweergave.
- Een pictogram van 120 bij 120 pixels: toepassingspictogram voor iPhones/iPods met hogere resolutie.
- Een pictogram van 152 bij 152 pixels: toepassingspictogram voor iPads met hogere resolutie.

- Een pictogram van 167 bij 167 pixels: toepassingspictogram voor iPad Pro met hogere resolutie.
- Een pictogram van 512 bij 512 pixels: toepassingspictogram voor iPhones/iPods/iPads met lagere resolutie. iTunes geeft dit pictogram weer. Het PNG-bestand van 512 bij 512 pixels wordt alleen gebruikt om de ontwikkelversies van uw toepassing te testen. Wanneer u de definitieve versie van uw toepassing indient bij Apple App Store, moet u deze afbeelding afzonderlijk meeleveren als JPG-bestand. De afbeelding wordt niet opgenomen in het IPA-bestand.
- Een pictogram van 1024 bij 1024 pixels: toepassingspictogram voor iPhones/iPods/iPads met netvliesweergave.

Bij iOS wordt aan het pictogram een schittereffect toegevoegd. U hoeft het effect niet toe te passen op de bronafbeelding. Als u dit standaardschittereffect wilt verwijderen, voegt u het volgende toe aan het element `InfoAdditions` in het descriptorbestand van de toepassing:

```
<InfoAdditions>
                                <![CDATA[
                                <key>UIPrerenderedIcon</key>
                                <true/>
                                ]]>
                                </InfoAdditions>
```

***Opmerking:*** *op iOS worden toepassingsmetagegevens ingevoegd als PNG-metagegevens in de toepassingspictogrammen, zodat Adobe het aantal in de Apple iOS App Store beschikbare AIR-toepassingen, kan opvolgen. Als u niet wilt dat uw toepassing vanwege deze pictogrammetagegevens wordt geïdentificeerd als een AIR-toepassing, moet u het IPA-bestand uitpakken, de pictogrammetagegevens verwijderen en de toepassing opnieuw verpakken. Deze procedure wordt beschreven in het artikel Opt-out van AIR-toepassingsanalyse voor iOS.*

### Meer Help-onderwerpen

"icon" op pagina 237

"imageNxN" op pagina 238

Android-ontwikkelaars: richtlijnen voor het ontwerpen van pictogrammen

iOS-richtlijnen voor menselijke interface: richtlijnen voor het maken van aangepaste pictogrammen en afbeeldingen

## iOS-opstartafbeeldingen

Naast toepassingspictogrammen dient u ook minstens één opstartafbeelding met de naam *Default.png* te verschaffen. U kunt desgewenst aparte opstartafbeeldingen opnemen voor verschillende opstartoriëntaties, resoluties (inclusief Retina Display met hoge resolutie en een hoogte-breedteverhouding van 16:9) en apparaten. U kunt ook verschillende opstartafbeeldingen opnemen die moeten worden gebruikt wanneer de toepassing wordt aangeroepen via een URL.

Er wordt niet naar opstartbestanden verwezen in het descriptorbestand van de toepassing. Deze bestanden moeten in de hoofdmap van de toepassing worden geplaatst. (Plaats de bestanden *niet* in een submap.)

#### Schema voor bestandsnaamgeving

Geef de afbeelding een naam op basis van het volgende schema:

```
basename + screen size modifier + urischeme + orientation + scale + device + .png
```

Het deel *basename* van de bestandsnaam is het enige verplichte gedeelte. U geeft of *Default* (met een hoofdletter D) op of de naam met de sleutel `UILaunchImageFile` in het `InfoAdditions`-element in het descriptorbestand van de toepassing.

Het gedeelte *screen size modifier* (schermgrootteaanpassing) duidt de grootte van het scherm aan als deze niet tot een van de standaardschermgrootten behoort. Deze aanpassing is alleen van toepassing op iPhone en iPod Touch-modellen met schermen met een hoogte-breedteverhouding van 16:9 zoals de iPhone 5 en iPod Touch (vijfde generatie). De enige ondersteunde waarde voor deze aanpassing is `-568h`. Omdat deze apparaten ondersteuning bieden voor schermen met hoge resolutie (en Retina Display-functie), wordt deze schermgrootteaanpassing altijd gebruikt met een afbeelding die ook de scale modifier (of schalingsoptie) `@2x` bezit. De volledige naam voor de standaardopstartafbeelding voor deze apparaten is `Default-568h@2x.png`.

Het gedeelte *urischeme* is de tekenreeks waarmee het URI-schema wordt geïdentificeerd. Dit gedeelte is alleen van toepassing als uw toepassing een of meer aangepaste URL-schema's ondersteunt. Als uw toepassing bijvoorbeeld kan worden aangeroepen via een koppeling zoals `example://foo`, gebruikt u `-example` als het schemagedeelte van de bestandsnaam van de opstartafbeelding.

Het gedeelte *orientation* biedt een manier om meerdere opstartafbeeldingen op te geven die kunnen worden gebruikt afhankelijk van de apparaatoriëntatie wanneer de toepassing wordt gestart. Dit gedeelte is alleen van toepassing op afbeeldingen voor iPad-toepassingen. Het kan een van de volgende waarden zijn waarmee de oriëntatie van het apparaat tijdens het starten van de toepassing wordt bepaald:

- -Portrait
- -PortraitUpsideDown
- -Landscape
- -LandscapeLeft
- -LandscapeRight

Het *scale*-gedeelte is `@2x` (voor iPhone 4, iPhone 5 en iPhone 6), of `@3x` (voor iPhone 6 plus) voor de opstartafbeeldingen voor schermen met hoge resolutie (retina). (Laat het scale-gedeelte helemaal weg voor afbeeldingen die zijn bedoeld voor schermen met standaardresolutie.) Voor opstartafbeeldingen voor grotere apparaten zoals iPhone 5 en iPod Touch (vijfde generatie) moet u ook de schermgrootteaanpassing `-528h` opgeven na het gedeelte basename en voor elk ander gedeelte.

Het gedeelte *device* wordt gebruikt om opstartafbeeldingen voor handheldapparaten en telefoons aan te duiden. Dit gedeelte wordt gebruikt wanneer uw toepassing een universele toepassing is die zowel handheldapparaten als tablets met één binaire toepassingsoperator ondersteunt. De mogelijk waarde moet ofwel `~ipad` of `~iphone` zijn (zowel voor iPhone als voor iPod Touch).

Voor iPhone kunt u alleen afbeeldingen met de hoogte-breedteverhouding Staand opnemen. In het geval van iPhone 6 plus kunnen liggende afbeeldingen ook worden toegevoegd. Gebruik afbeeldingen met 320 x 480 pixels voor apparaten met een standaardresolutie, afbeeldingen met 640 x 960 pixels voor apparaten met een hoge resolutie en afbeeldingen met 640 x 1136 pixels voor apparaten met een hoogte-breedteverhouding van 16:9 zoals iPhone 5 en iPod Touch (vijfde generatie).

Voor de iPad kunt u als volgt afbeeldingen opnemen:

- AIR 3.3 of lager — Afbeeldingen die niet het volledige scherm beslaan: zowel met de oriëntatie Landschap (1024 x 748 voor normale resolutie, 2048 x 1496 voor hoge resolutie) als Staand (768 x 1004 voor normale resolutie, 1536 x 2008 voor hoge resolutie).
- AIR 3.4 en hoger — Afbeeldingen die het volledige scherm beslaan: zowel met de oriëntatie Landschap (1024 x 768 voor normale resolutie, 2048 x 1536 voor hoge resolutie) als Staand (768 x 1024 voor normale resolutie, 1536 x 2048 voor hoge resolutie). Opmerking: wanneer u een volledige-schermafbeelding inpakt voor een toepassing die niet op het volledige scherm wordt weergegeven, worden de bovenste 20 pixels (bovenste 40 pixels bij hoge resolutie) overlapt door de statusbalk. Zorg er dus voor dat er in dit gebied geen belangrijke informatie wordt weergegeven.

### Voorbeelden

In de volgende tabel ziet u voorbeelden van opstartafbeeldingen die u kunt opnemen in een hypothetische toepassing met ondersteuning voor het grootst mogelijke bereik apparaten en oriëntaties en die kan worden gestart met URL's die gebruikmaken van het schema `example://`:

| Bestandsnaam | Formaat afbeelding | Gebruik |
|---|---|---|
| Default.png | 320 x 480 | iPhone, standaardresolutie |
| Default@2x.png | 640 x 960 | iPhone, hoge resolutie |
| Default-568h@2x.png | 640 x 1136 | iPhone, hoge resolutie, hoogte-breedteverhouding van 16:9 |
| Default-Portrait.png | 768 x 1004 (AIR 3.3 of eerder)<br>768 x 1024 (AIR 3.4 of hoger) | iPad, oriëntatie Staand |
| Default-Portrait@2x.png | 1536 x 2008 (AIR 3.3 of eerder)<br>1536 x 2048 (AIR 3.4 of hoger) | iPad, hoge resolutie, oriëntatie staand |
| Default-PortraitUpsideDown.png | 768 x 1004 (AIR 3.3 of eerder)768 x 1024 (AIR 3.4 of hoger) | iPad, oriëntatie Staand - omgekeerd |
| Standaard-StaandOndersteboven@2x.png | 1536 x 2008 (AIR 3.3 of eerder) 1536 x 2048 (AIR 3.4 of hoger) | iPad, hoge resolutie, oriëntatie staand ondersteboven |
| Default-Landscape.png | 1024 x 768 | iPad, oriëntatie Liggend - links |
| Standaard-LiggendLinks@2x.png | 2048 x 1536 | iPad, hoge resolutie, oriëntatie links liggend |
| Default-LandscapeRight.png | 1024 x 768 | iPad, oriëntatie Liggend - rechts |
| Standaard-LiggendRechts@2x.png | 2048 x 1536 | iPad, hoge resolutie, oriëntatie rechts liggend |
| Default-example.png | 320 x 480 | example://-URL op standaard iPhone |
| Default-example@2x.png | 640 x 960 | example://-URL op iPhone met hoge resolutie |
| Default-example~ipad.png | 768 x 1004 | example://-URL op iPad in oriëntatie Staand |
| Default-example-Landscape.png | 1024 x 768 | example://-URL op iPad in oriëntatie Liggend |

Dit voorbeeld illustreert slechts één mogelijke aanpak. U kunt bijvoorbeeld de afbeelding `Default.png` gebruiken voor de iPad en met `Default~iphone.png` en `Default@2x~iphone.png` specifieke opstartafbeeldingen opgeven voor de iPhone en de iPod.

### Zie ook

Programmeringsgids voor iOS-toepassingen: opstartafbeeldingen voor toepassingen

### Opstartafbeeldingen die moeten worden ingepakt voor iOS-apparaten

| Apparaten | Resolutie (pixels) | De naam van de opstartafbeelding | Oriëntatie |
|---|---|---|---|
| **iPhones** | | | |
| iPhone 4 (zonder retina) | 640 x 960 | Default~iphone.png | Portrait |
| iPhone 4, 4s | 640 x 960 | Default@2x~iphone.png | Portrait |
| iPhone 5, 5c, 5s | 640 x 1136 | Default-568h@2x~iphone.png | Portrait |
| iPhone6, iPhone 7 | 750 x 1334 | Default-375w-667h@2x~iphone.png | Portrait |
| iPhone6+, iPhone 7+ | 1242 x 2208 | Default-414w-736h@3x~iphone.png | Portrait |
| iPhone6+, iPhone 7+ | 2208 x 1242 | Default-Landscape-414w-736h@3x~iphone.png | Landscape |
| **iPads** | | | |
| iPad 1, 2 | 768 x 1024 | Default-Portrait~ipad.png | Portrait |
| iPad 1, 2 | 768 x 1024 | Default-PortraitUpsideDown~ipad.png | Portrait, ondersteboven |
| iPad 1, 2 | 1024 x 768 | Default-Landscape~ipad.png | Liggend, links |
| iPad 1, 2 | 1024 x 768 | Default-LandscapeRight~ipad.png | Liggend, rechts |
| iPad 3, Air | 1536 x 2048 | Default-Portrait@2x~ipad.png | Portrait |
| iPad 3, Air | 1536 x 2048 | Default-PortraitUpsideDown@2x~ipad.png | Portrait, ondersteboven |
| iPad 3, Air | 2048 x 1536 | Default-LandscapeLeft@2x~ipad.png | Liggend, links |
| iPad 3, Air | 2048 x 1536 | Default-LandscapeRight@2x~ipad.png | Liggend, rechts |
| iPad Pro | 2048 x 2732 | Default-Portrait@2x.png | Portrait |
| iPad Pro | 2732 x 2048 | Default-Landscape@2x.png | Landscape |

### Richtlijnen voor illustraties

U kunt elke gewenste afbeelding als opstartafbeelding gebruiken, zo lang de afbeelding maar de juiste afmetingen heeft. Het is echter vaak het beste als de afbeelding overeenkomt met de beginstatus van uw toepassing. U kunt een opstartafbeelding maken door een screenshot te maken van het opstartscherm van uw toepassing:

1. Open uw toepassing op het iOS-apparaat. Wanneer het eerste scherm van de gebruikersinterface verschijnt, drukt u op de thuisknop en houdt u deze knop ingedrukt (onder het scherm). Terwijl u de thuisknop houdt ingedrukt, drukt u op de knop voor de sluimerstand boven aan het apparaat. Zo maakt u een schermafbeelding en verzendt u de afbeelding naar het Filmrol-album.
2. Breng de afbeelding over naar uw ontwikkelcomputer door deze over te brengen van de iPhone of door een andere toepassing voor het overbrengen van foto's te gebruiken.

Plaats geen tekst in de opstartafbeelding als de toepassing in meerdere talen wordt vertaald. De opstartafbeelding is een statisch bestand, dus de tekst zou dan niet overeenkomen met de andere talen.

**Zie ook**

iOS-richtlijnen voor menselijke interface: opstartafbeeldingen

## Genegeerde instellingen

Toepassingen op mobiele apparaten negeren toepassingsinstellingen die van toepassing zijn op de native functies van het Windows- of desktop-besturingssysteem. De genegeerde instellingen zijn:

- allowBrowserInvocation
- customUpdateUI
- fileTypes
- height
- installFolder
- maximizable
- maxSize
- minimizable
- minSize
- programMenuFolder
- resizable
- systemChrome
- title
- transparent
- visible
- width
- x
- y

# Een mobiele AIR-toepassing verpakken

Gebruik de ADT-opdracht -package om het toepassingspakket te maken voor een AIR-toepassing die bestemd is voor een mobiel apparaat. Met de parameter -target wordt het mobiele platform opgegeven waarvoor het pakket is gemaakt.

**Android-pakketten**
AIR-toepassingen op Android maken gebruik van de Android-toepassingspakketindeling (APK) in plaats van de AIR-pakketindeling.

Pakketten die met ADT worden gemaakt en het doeltype *APK* gebruiken, hebben een indeling die kan worden verzonden naar de Android Market. De Android Market heeft geen vereisten waaraan de verzonden toepassingen moeten voldoen om te worden geaccepteerd. U moet de nieuwste vereisten raadplegen voordat u een definitief pakket maakt. Zie Android-ontwikkelaars: publiceren op de Market.

In tegenstelling tot bij iOS-toepassingen, kunt u een normaal AIR-certificaat voor ondertekening van programmacode gebruiken voor ondertekening van uw Android-toepassing. Om echter een toepassing naar de Android Market te kunnen verzenden, moet het certificaat conform de Market-regels zijn, waarin wordt vereist dat het certificaat geldig moet zijn tot ten minste 2033. U kunt een dergelijk certificaat maken door middel van de ADT-opdracht -certificate.

Als u een toepassing wilt indienen bij een alternatieve market die toepassingen niet toestaat een AIR-download te vereisen van de Google-market, kunt u een alternatieve download-URL opgeven met gebruik van de `-airDownloadURL`-parameter van ADT. Wanneer een gebruiker die niet over de vereiste versie van de AIR-runtime beschikt uw toepassing start, wordt hij of zij omgeleid naar de opgegeven URL. Zie "ADT-opdracht voor verpakken" op pagina 174 voor meer informatie.

Standaard verpakt ADT de Android-toepassing met gedeelde runtime. Om de toepassing uit te voeren moet de gebruiker een afzonderlijke AIR-runtime op het apparaat installeren.

***Opmerking:*** *Als u ADT wilt forceren om een APK te maken die een captive runtime toepast, gebruikt u* `target apk-captive-runtime`.

**iOS-pakketten**
AIR-toepassingen op iOS maken in plaats van de native AIR-indeling gebruik van de iOS-pakketindeling (IPA).

Pakketten die door ADT worden geproduceerd met behulp van het doeltype *ipa-app-store*, het correcte certificaat voor ondertekenen van code en het correcte inrichtingsprofiel hebben een indeling die kan worden verzonden door de Apple App Store. Gebruik het doeltype *ipa-ad-hoc* om een toepassing voor ad-hocdistributie te verpakken.

U moet het correcte door Apple uitgegeven ontwikkelaarscertificaat gebruiken om uw toepassing te ondertekenen. Voor het maken van testbuilds worden andere certificaten gebruikt dan voor het definitieve verpakken voordat de toepassing wordt verzonden.

Zie Piotr Walczyszyn: Packaging AIR application for iOS devices with ADT command and ANT script als u wilt zien hoe u Ant gebruikt om een pakket te maken van een iOS-toepassing.

## Verpakken met ADT

AIR SDK versie 2.6 en hoger biedt ondersteuning voor verpakken met ADT voor zowel iOS als Android. Voor het verpakken moeten al uw ActionScript-, MXML- en uitbreidingscode zijn gecompileerd. U moet ook over een certificaat voor het ondertekenen van code beschikken.

Zie voor gedetailleerde informatie over ADT-opdrachten en -opties "AIR Developer Tool (ADT)" op pagina 173.

### APK-pakketten van Android

**APK-pakketten maken**
Als u een APK-pakket wilt maken, kunt u de ADT-opdracht package maken en stelt u het doeltype in op *apk* voor releasebuilds, *apk-debug* voor builds voor foutopsporing of *apk-emulator* voor builds voor de release-modus als een emulator wordt gebruikt.

```
adt -package
                                -target apk
                                -storetype pkcs12 -keystore ../codesign.p12
                                myApp.apk
                                myApp-app.xml
                                myApp.swf icons
```

Typ de volledige opdracht op één regel. Er zijn uitsluitend regeleinden in het bovenstaande voorbeeld aanwezig om het beter leesbaar te maken. Bovendien wordt er in het voorbeeld van uitgegaan dat het pad naar het ADT-hulpprogramma zich in de paddefinitie van de opdrachtregel-shell bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318 voor uitleg.)

U moet de opdracht uitvoeren in de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn myApp-app.xml (het descriptorbestand van de toepassing), myApp.swf en een map met pictogrammen.

Wanneer u volgens het voorbeeld de opdracht uitvoert, wordt u door ADT gevraagd het keystore-wachtwoord in te voeren. (De tekens waaruit het wachtwoord bestaat, worden niet weergegeven. U kunt op Enter drukken als u het wachtwoord hebt getypt.)

***Opmerking:*** *Standaard hebben alle AIR Android-toepassingen het voorvoegsel* `air.` *in de naam van het pakket. Als u dit standaardgedrag niet wilt toepassen, moet u de omgevingsvariabele* `AIR_NOANDROIDFLAIR` *op uw computer instellen op true.*

**Een APK-pakket maken voor een toepassing die gebruikmaakt van native extensies**

Als u een APK-pakket wilt maken voor een toepassing die native extensies gebruikt, voegt u de naast de gebruikelijke pakketopties ook de optie `-extdir` toe. In het geval van meerdere ANE's die bronnen/bibliotheken delen, kiest de ADT slechts één bron/bibliotheek en worden andere dubbele vermeldingen genegeerd voordat een waarschuwingsbericht wordt verzonden. Met deze optie geeft u de map op met de ANE-bestanden die de toepassing gebruikt. Bijvoorbeeld:

```
adt -package
                                -target apk
                                -storetype pkcs12 -keystore ../codesign.p12
                                myApp.apk
                                myApp-app.xml
                                -extdir extensionsDir
                                myApp.swf icons
```

**Een APK-pakket met een eigen versie van de AIR-runtime maken**

Als u een APK-pakket met zowel de toepassing als een captive versie van de AIR-runtime wilt maken, gebruikt u het doel `apk-captive-runtime`. Met deze optie geeft u de map op met de ANE-bestanden die de toepassing gebruikt. Bijvoorbeeld:

```
adt -package
                                -target apk-captive-runtime
                                -storetype pkcs12 -keystore ../codesign.p12
                                myApp.apk
                                myApp-app.xml
                                myApp.swf icons
```

Mogelijke nadelen van deze techniek bevatten:

- Belangrijke beveiligingscorrecties staan niet automatisch ter beschikking van gebruikers als Adobe een beveiligingspatch publiceert
- Groter RAM-verbruik door toepassing

***Opmerking:*** *wanneer u de runtime bundelt, voegt ADT de machtigingen* `INTERNET` *en* `BROADCAST_STICKY` *toe aan uw toepassing. Deze machtigingen zijn vereist door de AIR-runtime.*

**APK-pakketten voor foutopsporing maken**

Voor het maken van een versie van de toepassing die u in combinatie met een foutopsporingsprogramma kunt gebruiken, kunt u apk-debug gebruiken als het doel en verbindingsopties opgeven:

```
adt -package
                                -target apk-debug
                                -connect 192.168.43.45
                                -storetype pkcs12 -keystore ../codesign.p12
                                myApp.apk
                                myApp-app.xml
                                myApp.swf icons
```

Met de markering -connect wordt aan de AIR-runtime aangegeven met welk apparaat verbinding moet worden gemaakt met een extern foutopsporingsprogramma in het netwerk. Als u fouten via USB wilt opsporen, moet u in plaats daarvan de markering `-listen` opgeven, waarbij de TCP-poort wordt opgegeven voor de verbinding voor foutopsporing:

```
adt -package
                                -target apk-debug
                                -listen 7936
                                -storetype pkcs12 -keystore ../codesign.p12
                                myApp.apk
                                myApp-app.xml
                                myApp.swf icons
```

Voor het functioneren van de meeste foutopsporingsfuncties moet u ook de toepassings-SWF's en -SWC's compileren met de foutopsporingsfunctie ingeschakeld. Zie "Verbindingsopties voor foutopsporing" op pagina 191 voor een volledige beschrijving van de markeringen `-connect` en `-listen`.

***Opmerking:*** *Wanneer u een app verpakt met het doel APK-debug, verpakt ADT standaard een 'captive' exemplaar van de AIR-runtime software bij uw Android-app. Als u ADT wilt foceren om een APK te maken die een externe runtime gebruikt, stelt u de `AIR_ANDROID_SHARED_RUNTIME`-omgevingsvariabele in op `true`.*

Bij Android moet een er voor een toepassing ook toestemming worden gegeven voor toegang tot internet, om verbinding te maken met de computer die de foutopsporing op het netwerk uitvoert. Zie "Android-machtigingen" op pagina 81.

### APK-pakketten maken voor gebruik op een Android-emulator

U kunt een APK-pakket voor foutopsporing gebruiken op een Android-emulator, maar geen pakket in de releasemodus. Als u een APK-pakket in de releasemodus wilt maken voor gebruik op een emulator, moet u de ADT-opdracht package gebruiken, waarmee het doeltype wordt ingesteld op *apk-emulator*:

```
adt -package -target apk-emulator -storetype pkcs12 -keystore ../codesign.p12 myApp.apk myApp-app.xml myApp.swf icons
```

In het voorbeeld wordt ervan uitgegaan dat het pad van het ADT-hulpprogramma zich op de pad-definitie van de shell van uw opdrachtregel bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318 voor uitleg.)

### APK-pakketten maken van een AIR- of AIRI-bestand

U kunt een APK-pakket direct van een bestaand AIR- of AIRI-bestand maken:

```
adt -target apk -storetype pkcs12 -keystore ../codesign.p12 myApp.apk myApp.air
```

Het AIR-bestand moet gebruikmaken van de naamruimte in het toepassingsdescriptorbestand van AIR 2.5 (of later).

### APK-pakketten maken voor het Android x86-platform

Vanaf AIR 14 kan het argument,`-arch` worden gebruikt om een APK voor het Android x86-platform te verpakken. Bijvoorbeeld:

```
    adt     -package
                                -target apk-debug
                                -listen 7936
                                -arch x86
                                -storetype pkcs12 -keystore ../codesign.p12
                                myApp.apk
                                myApp-app.xml
                                myApp.swf icons
```

## iOS-pakketten

Op iOS converteert ADT de bytecode van het SWF-bestand en overige bronbestanden naar een native iOS-toepassing.

1 Maak het SWF-bestand met behulp van Flash Builder, Flash Professional of een opdrachtregelcompiler.

2 Open een opdrachtshell of een terminal en ga naar de projectmap van de iPhone-toepassing.

3 Maak vervolgens met het ADT-hulpprogramma het IPA-bestand met behulp van de volgende syntaxis:

```
adt -package
                                    -target [ipa-test | ipa-debug | ipa-app-store | ipa-ad-
hoc |
                                    ipa-debug-interpreter | ipa-debug-interpreter-simulator
                                    ipa-test-interpreter | ipa-test-interpreter-simulator]
                                    -provisioning-profile PROFILE_PATH
                                    SIGNING_OPTIONS
                                    TARGET_IPA_FILE
                                    APP_DESCRIPTOR
                                    SOURCE_FILES
                                    -extdir extension-directory
                                    -platformsdk path-to-iossdk or path-to-ios-simulator-
sdk
```

Wijzig de verwijzings-`adt` om het volledige pad naar de ADT-toepassing op te nemen. De ADT-toepassing wordt geïnstalleerd in de binsubmap van de AIR-SDK.

Selecteer de optie `-target` die overeenkomt met het type iPhone-toepassing dat u wilt maken:

- `-target ipa test`: kies deze optie om snel een versie van de toepassing voor testdoeleinden te compileren op de ontwikkelaars-iPhone. U kunt ook `ipa-test-interpreter` gebruiken voor nog snellere compilaties, of `ipa-test-interpreter-simulator` voor uitvoer in de iOS-simulator.
- `-target ipa-debug`: kies deze optie om een versie van de toepassing voor foutopsporing te compileren op de ontwikkelaars-iPhone. Met deze optie kunt u een foutopsporingssessie gebruiken om `trace()`-uitvoer te ontvangen van de iPhone-toepassing.

  U kunt een van de volgende `-connect`-opties opnemen (`CONNECT_OPTIONS`) om het IP-adres op te geven van de ontwikkelcomputer waarop het foutopsporingsprogramma wordt uitgevoerd:

  - `-connect` - De toepassing probeert een wifi-verbinding tot stand te brengen met een foutopsporingssessie op de ontwikkelcomputer die wordt gebruikt voor het compileren van de toepassing.
  - `-connect IP_ADDRESS` - De toepassing probeert een wifi-verbinding tot stand te brengen met een foutopsporingssessie op de computer met het opgegeven IP-adres. Bijvoorbeeld:

    ```
    -target ipa-debug -connect 192.0.32.10
    ```

  - `-connect HOST_NAME` - De toepassing probeert een wifi-verbinding tot stand te brengen met een foutopsporingssessie op de computer met de opgegeven hostnaam. Bijvoorbeeld:

    ```
    -target ipa-debug -connect bobroberts-mac.example.com
    ```

  De optie `-connect` is optioneel. Als deze optie niet wordt opgegeven, probeert de resulterende foutopsporingstoepassing geen verbinding tot stand te brengen met een gehost foutopsporingsprogramma. Als alternatief kunt u ook `-listen` opgeven in plaats van `-connect` om USB-foutopsporing in te schakelen, zoals beschreven in “Foutopsporing op afstand met FDB via USB” op pagina 112.

Als een verbindingspoging met een foutopsporingsprogramma mislukt, wordt er een dialoogvenster weergegeven waarin de gebruiker het IP-adres van de hostcomputer met het foutopsporingsprogramma moet opgeven. Een verbindingspoging kan mislukken als het apparaat geen draadloze verbinding heeft. Dit kan ook gebeuren als het apparaat wel een verbinding heeft, maar niet wordt beveiligd door de firewall van de hostcomputer met het foutopsporingsprogramma.

U kunt ook `ipa-debug-interpreter` gebruiken voor nog snellere compilaties, of `ipa-debug-interpreter-simulator` voor uitvoer in de iOS-simulator.

Zie voor meer informatie "Fouten opsporen in een mobiele AIR-toepassing" op pagina 106.

- `-target ipa-ad-hoc`: kies deze optie om een toepassing te maken voor ad-hocimplementatie. Zie het Apple iPhone-ontwikkelaarscentrum
- `-target ipa-app store`: kies deze optie om een definitieve versie van het IPA-bestand te maken voor implementatie op de Apple App Store.

Vervang *`PROFILE_PATH`* door het pad naar het inrichtingsprofielbestand voor uw toepassing. Zie "iOS installeren" op pagina 70 voor meer informatie over inrichtingsprofielen.

Gebruik de optie `-platformsdk` om te wijzen naar de iOS-simulator SDK wanneer u werkt aan een toepassing die u in de iOS-simulator wilt uitvoeren.

Vervang *`SIGNING_OPTIONS`* door verwijzingen naar het iPhone-certificaat en -wachtwoord voor ontwikkelaars. Gebruik de volgende syntaxis:

```
-storetype pkcs12 -keystore P12_FILE_PATH -storepass PASSWORD
```

Vervang *P12_FILE_PATH* door het pad naar het P12-certificaatbestand. Vervang *PASSWORD* door het certificaatwachtwoord. (Zie het onderstaande voorbeeld.) Zie "Een ontwikkelingscertificaat omzetten in een P12-sleutelarchief-bestand" op pagina 207 voor meer informatie over het P12-certificaatbestand.

***Opmerking:*** *u kunt een certificaat gebruiken dat u zelf hebt ondertekend wanneer u een pakket maakt voor de iOS-simulator.*

Vervang *`APP_DESCRIPTOR`* om naar het descriptorbestand van de toepassing te verwijzen.

Vervang *`SOURCE_FILES`* om naar het hoofd-SWF-bestand van uw project te verwijzen, gevolgd door eventuele andere bronnen die moeten worden opgenomen. Neem de paden op naar alle pictogrambestanden die u in het dialoogvenster met de instellingen voor de toepassing in Flash Professional of in een aangepast descriptorbestand van de toepassing hebt gedefinieerd. Voeg ook Default.png, het bestand met de eerste schermillustratie, toe.

Gebruik de optie `-extdir` *`extension-directory`* om de map op te geven met de ANE-bestanden (native extensies) die de toepassing gebruikt. Als de toepassing geen native extensies gebruikt, dient u deze optie niet te gebruiken.

***Belangrijk:*** *maak in uw toepassingsmap geen submap met de naam* `Resources`*. De runtime maakt in overeenkomst met de IPA-pakketstructuur automatisch een map met deze naam. Als u uw eigen Resources-map maakt, ontstaat er een onherstelbaar conflict.*

### iOS-pakketten voor foutopsporing maken

Als u iOS-pakketten wilt maken om op testapparaten te installeren, gebruikt u de ADT-opdracht package waarmee het doeltype wordt ingesteld op *ios-debug*. Voordat u deze opdracht uitvoert, moet u al een ondertekeningscertificaat voor de code en een inrichtingsbestand van Apple hebben gekregen.

```
adt -package
                                -target ipa-debug
                                -storetype pkcs12 -keystore ../AppleDevelopment.p12
                                -provisioning-profile AppleDevelopment.mobileprofile
                                -connect 192.168.0.12 | -listen
                                myApp.ipa
                                myApp-app.xml
                                myApp.swf icons Default.png
```

***Opmerking:*** *u kunt ook* `ipa-debug-interpreter` *gebruiken voor nog snellere compilaties, of* `ipa-debug-interpreter-simulator` *voor uitvoer in de iOS-simulator.*

Typ de volledige opdracht op één regel. Er zijn uitsluitend regeleinden in het bovenstaande voorbeeld aanwezig om het beter leesbaar te maken. Bovendien wordt er in het voorbeeld van uitgegaan dat het pad naar het ADT-hulpprogramma zich in de paddefinitie van de opdrachtregel-shell bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318 voor uitleg.)

U moet de opdracht uitvoeren in de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn: myApp-app.xml (het toepassingsdescriptorbestand), myApp.swf, een map met pictogrammen en het bestand Default.png.

U moet de toepassing ondertekenen met het correcte, door Apple uitgegeven distributiecertificaat. Overige ondertekeningscertificaten voor code kunnen niet worden gebruikt.

Gebruik de optie `-connect` voor wifi-foutopsporing. De toepassing probeert om een foutopsporingssessie te initiëren met de Flash Debugger (FDB), die wordt uitgevoerd op het opgegeven IP-adres of de opgegeven hostnaam. Gebruik de optie `-listen` voor USB-foutopsporing. Als u eerst de toepassing start en vervolgens FDB, wordt een foutopsporingssessie geïnitieerd voor de toepassing die wordt uitgevoerd. Zie voor meer informatie "Verbinding maken met Flash Debugger" op pagina 110.

**iOS-pakketten maken voor het verzenden naar Apple App store.**

Als u een iOS-pakket wilt maken voor verzending naar de Apple App store, gebruikt u de ADT-opdracht package, waarmee het doeltype wordt ingesteld op *ios-app-store*. Voordat u deze opdracht uitvoert, moet u al een ondertekeningscertificaat voor de distributiecode en een inrichtingsbestand van Apple hebben gekregen.

```
adt -package
                                -target ipa-app-store
                                -storetype pkcs12 -keystore ../AppleDistribution.p12
                                -provisioning-profile AppleDistribution.mobileprofile
                                myApp.ipa
                                myApp-app.xml
                                myApp.swf icons Default.png
```

Typ de volledige opdracht op één regel. Er zijn uitsluitend regeleinden in het bovenstaande voorbeeld aanwezig om het beter leesbaar te maken. Bovendien wordt er in het voorbeeld van uitgegaan dat het pad naar het ADT-hulpprogramma zich in de paddefinitie van de opdrachtregel-shell bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318 voor uitleg.)

U moet de opdracht uitvoeren in de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn: myApp-app.xml (het toepassingsdescriptorbestand), myApp.swf, een map met pictogrammen en het bestand Default.png.

U moet de toepassing ondertekenen met het correcte, door Apple uitgegeven distributiecertificaat. Overige ondertekeningscertificaten voor code kunnen niet worden gebruikt.

***Belangrijk:*** *door Apple wordt vereist dat u het Apple-programma Application Loader gebruikt voor het uploaden van toepassingen naar de App Store. Apple publiceert Application Loader uitsluitend voor Mac OS X. Als u dus met een Windows-computer een AIR-toepassing voor de iPhone wilt ontwikkelen, moet u toegang hebben tot een computer waarop OS X wordt uitgevoerd (versie 10.5.3 of later) om de toepassing naar de App Store te kunnen verzenden. Het programma Application Loader is verkrijgbaar bij het Apple iOS Developer Center.*

**iOS-pakketten voor ad-hocdistributie maken.**

Als u een iOS-pakket voor ad-hocdistributie wilt maken, gebruikt u de ADT-opdracht package, waarmee het doeltype wordt ingesteld op *ios-ad-hoc*. Voordat u deze opdracht uitvoert, moet u al een ondertekeningscertificaat voor de adhocdistributiecode en een inrichtingsbestand van Apple hebben gekregen.

```
adt -package
                                -target ipa-ad-hoc
                                -storetype pkcs12 -keystore ../AppleDistribution.p12
                                -provisioning-profile AppleDistribution.mobileprofile
                                myApp.ipa
                                myApp-app.xml
                                myApp.swf icons Default.png
```

Typ de volledige opdracht op één regel. Er zijn uitsluitend regeleinden in het bovenstaande voorbeeld aanwezig om het beter leesbaar te maken. Bovendien wordt er in het voorbeeld van uitgegaan dat het pad naar het ADT-hulpprogramma zich in de paddefinitie van de opdrachtregel-shell bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318 voor uitleg.)

U moet de opdracht uitvoeren in de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn: myApp-app.xml (het toepassingsdescriptorbestand), myApp.swf, een map met pictogrammen en het bestand Default.png.

U moet de toepassing ondertekenen met het correcte, door Apple uitgegeven distributiecertificaat. Overige ondertekeningscertificaten voor code kunnen niet worden gebruikt.

**Een iOS-pakket maken voor een toepassing die gebruikmaakt van native extensies**

Gebruik de ADT-pakketopdracht met de optie `-extdir` om een iOS-pakket te maken voor een toepassing die native extensies gebruikt. De ADT-opdracht is geschikt voor het doel (`ipa-app-store`, `ipa-debug`, `ipa-ad-hoc` en `ipa-test`). Bijvoorbeeld:

```
adt -package
                                -target ipa-ad-hoc
                                -storetype pkcs12 -keystore ../AppleDistribution.p12
                                -provisioning-profile AppleDistribution.mobileprofile
                                myApp.ipa
                                myApp-app.xml
                                -extdir extensionsDir
                                myApp.swf icons Default.png
```

Typ de volledige opdracht op één regel. Er zijn uitsluitend regeleinden in het bovenstaande voorbeeld aanwezig om het beter leesbaar te maken.

Het voorbeeld gaat er wat native extensies betreft van uit dat de map `extensionsDir` in de map staat van waaruit u de opdracht uitvoert. De map `extensionsDir` bevat de ANE-bestanden die de toepassing gebruikt.

# Fouten opsporen in een mobiele AIR-toepassing

U kunt op verschillende manieren fouten opsporen in uw mobiele AIR-toepassing. De eenvoudigste manier om toepassingslogische problemen bloot te leggen, is om op uw ontwikkelingscomputer een foutopsporing uit te voeren met behulp van ADL of de iOS-simulator. U kunt uw toepassing ook op een apparaat installeren en op afstand een foutopsporing uitvoeren op een desktopcomputer.

## Apparaatsimulatie met ADL

De snelste en eenvoudigste manier om de meeste functies van mobiele toepassingen te testen en er een foutopsporing voor uit te voeren, is om uw toepassing op uw computer uit te voeren met het hulpprogramma Adobe Debug Launcher (ADL). ADL maakt gebruik van het element `supportedProfiles` in het descriptorbestand van de toepassing om te bepalen welk profiel moet worden gebruikt. Als er meer dan één profiel wordt weergegeven, wordt het eerste in de lijst door ADL gebruikt. U kunt ook de ADL-parameter `-profile` gebruiken om een van de andere profielen in de lijst `supportedProfiles` te selecteren. (Als u geen element `supportedProfiles` opneemt in de toepassingsdescriptor, kunnen alle profielen worden opgegeven voor het argument `-profile`.) U kunt bijvoorbeeld de volgende opdracht gebruiken om een toepassing te starten voor het simuleren van het profiel van het mobiele apparaat:

```
adl -profile mobileDevice myApp-app.xml
```

Wanneer het mobiele profiel op deze wijze op de desktop wordt gesimuleerd, wordt de toepassing uitgevoerd in een omgeving die zeer veel overeenkomt met een mobiel doelapparaat. ActionScript-API's die geen deel uitmaken van het mobiele profiel, zijn niet beschikbaar. ADL maakt echter geen onderscheid tussen de capaciteit van verschillende mobiele apparaten. U kunt bijvoorbeeld het gesimuleerd op schermoetsen drukken naar uw toepassing verzenden, zelfs wanneer uw werkelijke doelapparaat geen schermtoetsen gebruikt.

ADL ondersteunt simulaties van richtingswijzigingen van apparaten en schermtoetsinvoer via menuopdrachten. Wanneer u ADL in het profiel van het mobiele apparaat uitvoert, geeft ADL een menu weer (in het toepassingsvenster of de desktopmenubalk) waarmee u het draaien van het apparaat of schermtoetsinvoer kunt opgeven.

#### Schermtoetsinvoer

ADL simuleert de schermtoetsen voor Terug, Menu en Zoeken op een mobiel apparaat. U kunt deze toetsen naar het gesimuleerde apparaat verzenden met het menu dat wordt weergegeven wanneer ADL wordt gestart met het mobiele profiel.

#### Het apparaat draaien

Met ADL wordt het draaien van het apparaat via het weergegeven menu gesimuleerd wanneer ADL wordt gestart met het mobiele profiel. U kunt het gesimuleerde apparaat naar rechts of naar links draaien.

De simulatie van het draaien heeft alleen betrekking op een toepassing waarbij het automatisch bepalen van richting is ingeschakeld. U kunt deze functie inschakelen door in de toepassingsdescriptor het element `autoOrients` in te stellen op `true`.

#### Schermgrootte

U kunt uw toepassing in verschillende schermgroottes testen door de ADL-parameter `-screensize` in te stellen. U kunt de code voor een van de vooraf gedefinieerde schermtypen opgeven of een tekenreeks met de vier waarden die de afmetingen van de normale en gemaximaliseerde schermen in pixels weergeven.

Geef altijd de pixeldimensies op voor een staande afdrukstand. Dit houdt in dat de breedte een kleinere waarde heeft dan de hoogte. Met de volgende opdracht kunt u ADL openen om het scherm te simuleren dat wordt gebruikt bij Motorola Droid:

```
adl -screensize 480x816:480x854 myApp-app.xml
```

Zie voor een lijst met de vooraf gedefinieerde schermtypen “Gebruik van ADL” op pagina 167.

**Beperkingen**

Enkele API's die niet op het desktopprofiel worden ondersteund, kunnen niet door ADL worden gesimuleerd. De niet-ondersteunde API's zijn onder meer:

- Versnellingsmeter
- cacheAsBitmapMatrix
- CameraRoll
- CameraUI
- Geolocatie
- Multitouch en handbewegingen bij besturingssystemen die deze functies niet ondersteunen
- SystemIdleMode

Als deze klassen door uw toepassing worden gebruikt, moet u de functies op een echt apparaat of emulator testen.

Er zijn ook API's die functioneren onder ADL op het bureaublad, maar die niet op alle typen mobiele apparatuur functioneren. Deze zijn onder meer:

- Speex- en AAC-audiocodec
- Toegankelijkheid en ondersteuning voor schermlezers
- RTMPE
- SWF-laadbestanden met ActionScript-bytecode
- PixelBender-arceringen

Vergeet niet de toepassingen die gebruikmaken van deze functies te testen op de doelapparaten, aangezien ADL niet de volledige uitvoeromgeving dupliceert.

## Apparaatsimulatie met de iOS-simulator

Met de iOS-simulator (alleen voor de Mac) wordt de uitvoering en foutopsporing van iOS-toepassingen heel eenvoudig. Voor het testen met de iOS-simulator is geen ontwikkelaarscertificaat of inrichtingsprofiel vereist. U moet wel een p12-certificaat maken, alhoewel u die ook zelf kunt ondertekenen.

Standaard start ADT altijd de iPhone-simulator. Ga als volgt te werk om het simulatorapparaat te wijzigen:

- Gebruik de onderstaande opdracht om de beschikbare simulators weer te geven.

  ```
  xcrun simctl list devices
  ```

  De uitvoer lijkt op wat hieronder wordt weergegeven.

```
== Devices ==
-iOS 10.0 -
iPhone 5 (F6378129-A67E-41EA-AAF9-D99810F6BCE8) (Shutdown)
iPhone 5s (5F640166-4110-4F6B-AC18-47BC61A47749) (Shutdown)
iPhone 6 (E2ED9D38-C73E-4FF2-A7DD-70C55A021000) (Shutdown)
iPhone 6 Plus (B4DE58C7-80EB-4454-909A-C38C4106C01B) (Shutdown)
iPhone 6s (9662CB8A-2E88-403E-AE50-01FB49E4662B) (Shutdown)
iPhone 6s Plus (BED503F3-E70C-47E1-BE1C-A2B7F6B7B63E) (Shutdown)
iPhone 7 (71880D88-74C5-4637-AC58-1F9DB43BA471) (Shutdown)
iPhone 7 Plus (2F411EA1-EE8B-486B-B495-EFC421E0A494) (Shutdown)
iPhone SE (DF52B451-ACA2-47FD-84D9-292707F9F0E3) (Shutdown)
iPad Retina (C4EF8741-3982-481F-87D4-700ACD0DA6E1) (Shutdown)
....
```

- U kunt een specifieke simulator kiezen door de omgevingsvariabele `AIR_IOS_SIMULATOR_DEVICE` als volgt in te stellen:

```
export AIR_IOS_SIMULATOR_DEVICE = 'iPad Retina'
```

Start het proces opnieuw nadat u de omgevingsvariabele hebt ingesteld en start de toepassing op het simulatorapparaat van uw keuze.

***Opmerking:*** *wanneer u ADT gebruikt met de iOS-simulator, moet u altijd de optie `-platformsdk` opnemen om het pad naar de iOS-simulator-SDK aan te duiden.*

Een toepassing uitvoeren in de iOS-simulator:

1 Gebruik de ADT-opdracht -package in combinatie met `-target ipa-test-interpreter-simulator` of `-target ipa-debug-interpreter-simulator`, zoals in het volgende voorbeeld:

```
adt -package
                                -target ipa-test-interpreter-simulator
                                -storetype pkcs12 -keystore Certificates.p12
                                -storepass password
                                myApp.ipa
                                myApp-app.xml
                                myApp.swf
                                -platformsdk
/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator5.0.sdk
```

***Opmerking:*** *In het geval van simulators zijn de ondertekeningsopties nu niet meer vereist, zodat elke waarde kan worden geleverd in de `-keystore` opdracht, aangezien deze zal worden genegeerd door ADT.*

2 Gebruik de adt -installApp-opdracht om de toepassing te installeren in de iOS-simulator, zoals in het volgende voorbeeld:

```
adt -installApp
                                -platform ios
                                -platformsdk
/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator5.0.sdk
                                -device ios-simulator
                                -package sample_ipa_name.ipa
```

3 Gebruik de adt -launchApp-opdracht om de toepassing uit te voeren in de iOS-simulator, zoals in het volgende voorbeeld:

***Opmerking:*** *Standaard wordt met de opdracht `adt -launchApp` de toepassing in de iPhone-simulator uitgevoerd. Als u de toepassing wilt uitvoeren in de iPad-simulator, exporteert u de omgevingsvariabele `AIR_IOS_SIMULATOR_DEVICE` = "iPad" en gebruikt u vervolgens de opdracht `adt -launchApp`.*

```
adt -launchApp
                                -platform ios
                                -platformsdk
/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator5.0.sdk
                                -device ios-simulator
                                -appid sample_ipa_name
```

Als u een native extensie wilt testen in de iOS-simulator, moet u de `iPhone-x86`-platformnaam gebruiken in het bestand extension.xml en `library.a` (statische bibliotheek) opgeven in het element `nativeLibrary`, zoals in het volgende voorbeeld met extension.xml:

```
<extension xmlns="http://ns.adobe.com/air/extension/3.1">
                          <id>com.cnative.extensions</id>
                          <versionNumber>1</versionNumber>
                          <platforms>
                            <platform name="iPhone-x86">
                              <applicationDeployment>
                                <nativeLibrary>library.a</nativeLibrary>
                                <initializer>TestNativeExtensionsInitializer </initializer>
                                <finalizer>TestNativeExtensionsFinalizer </finalizer>
                              </applicationDeployment>
                            </platform>
                          </platforms>
                        </extension>
```

***Opmerking:*** *wanneer u een native extensie in de iOS-simulator wilt testen, mag u de statische bibliotheek (`.a`-bestand) dat voor het apparaat is gecompileerd, niet gebruiken. In plaats daarvan gebruikt u de statische bibliotheek die voor de simulator is gecompileerd.*

## Traceringsinstructies

Wanneer u uw mobiele toepassing op de desktopcomputer uitvoert, wordt de traceringsuitvoer afgedrukt voor het foutopsporingsprogramma of het terminalvenster dat wordt gebruikt om ADL op te starten. Wanneer u uw toepassing op een apparaat of emulator uitvoert, kunt u een foutopsporingssessie op afstand instellen om traceringsuitvoer weer te geven. Wanneer dit wordt ondersteund, kunt u ook traceringsuitvoer weergeven met de hulpprogramma's voor softwareontwikkeling die worden geleverd door de fabrikant van het apparaat of het besturingssysteem.

In alle gevallen moeten de SWF-bestanden in de toepassing worden gecompileerd met de functie foutopsporing ingeschakeld, zodat traceringsinstructies door de runtime kunnen worden uitgevoerd.

#### Externe traceringsinstructies bij Android

Bij het uitvoeren op een Android-apparaat of -emulator kunt u in het Android-systeemlogboek uitvoer van traceringsinstructies weergeven met behulp van het hulpprogramma Android Debug Bridge (ADB) dat bij de Android-SDK is inbegrepen. Als u de uitvoer van uw toepassing wilt weergeven, voert u de volgende opdracht uit in een opdrachtprompt- of terminalvenster van uw ontwikkelingscomputer:

```
tools/adb logcat air.MyApp:I *:S
```

waarbij *MyApp* de AIR-toepassings-id is van uw toepassing. Het argument `*:S` onderdrukt uitvoer van alle overige processen. Als u naast de traceringsuitvoer systeeminformatie over uw toepassing wilt weergeven, kunt u de ActivityManager opnemen in de specificatie voor het logcatfilter:

```
tools/adb logcat air.MyApp:I ActivityManager:I *:S
```

Bij deze opdrachtvoorbeelden wordt ervan uitgegaan dat u ADB uitvoert in de map ANDROID SDK of dat u de map `SDK` hebt toegevoegd aan de variabele voor uw padomgeving.

***Opmerking:*** *in AIR 2.6+ is het hulpprogramma ADB inbegrepen bij de AIR-SDK en bevindt deze zich in de binmap lib/android/.*

**Externe traceringsinstructies bij iOS**

Als u de uitvoer wilt weergeven van traceringsinstructies van een toepassing die op een iOS-apparaat wordt uitgevoerd, moet u een foutopsporingssessie op afstand met de Flash Debugger (FDB) tot stand brengen.

### Meer Help-onderwerpen

Android Debug Bridge: Registratie logcat inschakelen

"Omgevingsvariabelen van het pad" op pagina 318

## Verbinding maken met Flash Debugger

Als u een foutopsporing voor een toepassing wilt uitvoeren op een mobiel apparaat, kunt u Flash Debugger uitvoeren op uw ontwikkelingscomputer en er via het netwerk verbinding mee maken. Voor het inschakelen van foutopsporing op afstand doet u het volgende:

- Bij Android kunt u de machtiging android:permission.INTERNET opgeven in de toepassingsdescriptor.
- Compileer de toepassings-SWF's met foutopsporing ingeschakeld.
- Verpak de toepassing met de markering `-target apk-debug voor Android,` of `-target ipa-debug` voor iOS, en gebruik ofwel de markering `-connect` (wifi-foutopsporing) of `-listen` (USB-foutopsporing).

Bij wifi-foutopsporing moet het apparaat via het IP-adres of een volledig correcte domeinnaam toegang hebben tot TPC-poort 7935 van de computer waarop de Flash Debugger wordt uitgevoerd. Bij externe foutopsporing via USB moet het apparaat toegang hebben tot TPC-poort 7936 of tot de poort die is opgegeven in de markering `-listen`.

Bij iOS kunt u ook `-target ipa-debug-interpreter` of `-target ipa-debug-interpreter-simulator` opgeven.

### Foutopsporing op afstand met Flash Professional

Zodra uw toepassing gereed is voor foutopsporing en de machtigingen zijn ingesteld in de toepassingsdescriptor, doet u het volgende:

1. Open het AIR-dialoogvenster voor Android-instellingen.
2. Onder de tab Deployment (Implementatie):
   - Selecteer "Device debugging" (Foutopsporing van apparaat) voor het implementatietype
   - Selecteer "Toepassing installeren op het aangesloten Android-apparaat" voor na publicatie
   - Maak de selectie van "Toepassing installeren op het aangesloten Android-apparaat" ongedaan voor na publicatie
   - Stel het pad in naar de Android-SDK (indien van toepassing).
3. Klik op Publiceren.

   Uw toepassing is geïnstalleerd en opgestart op het apparaat.
4. Sluit het AIR-dialoogvenster voor Android-instellingen.
5. Selecteer Debug (foutopsporing) >Foutopsporingssessie op afstand starten voor > ActionScript 3 in het menu Flash Professional.

   Door Flash Professional wordt in het deelvenster Uitvoer weergegeven: "Wachten op verbinding door speler...".
6. Start de toepassing op het apparaat.

7 Voer het IP-adres of de hostnaam in van de computer waarop het Flash-foutopsporingsprogramma wordt uitgevoerd in het Adobe AIR-dialoogvenster voor verbinding en klik vervolgens op OK.

## Foutopsporing op afstand met FDB via een netwerkverbinding

Als u met behulp van de opdrachtregel Flash Debugger (FDB) fouten wilt opsporen in een toepassing die wordt uitgevoerd op een apparaat, moet u het foutopsporingsprogramma eerst op uw ontwikkelingscomputer uitvoeren en vervolgens de toepassing op het apparaat starten. De procedure hieronder maakt gebruik van de hulpprogramma's AMXMLC, FDB en ADT voor het compileren, verpakken en het uitvoeren van een foutopsporing op het apparaat. In de voorbeelden wordt verondersteld dat u een gecombineerde Flex- en AIR-SDK gebruikt en dat de binmap is opgenomen in de omgevingsvariabele van het pad. (Deze veronderstelling is vooral bedoeld om opdrachtvoorbeelden eenvoudiger te maken.)

1 Open een venster voor terminal- of opdrachtprompts en navigeer naar de map met de broncode voor de toepassing.

2 Compileer de toepassing met amxmlc, waardoor de foutopsporing wordt ingeschakeld:

```
amxmlc -debug DebugExample.as
```

3 Verpak de toepassing met behulp van de doelen `apk-debug` of `ipa-debug`:

```
Android
                                    adt -package -target apk-debug -connect -storetype
pkcs12 -keystore ../../AndroidCert.p12 DebugExample.apk DebugExample-app.xml
DebugExample.swf
                                    iOS
                                    adt -package -target ipa-debug -connect -storetype
pkcs12 -keystore ../../AppleDeveloperCert.p12 -provisioning-profile test.mobileprovision
DebugExample.apk DebugExample-app.xml DebugExample.swf
```

Als u altijd dezelfde hostnaam of hetzelfde IP-adres gebruikt voor foutopsporing, kunt u die waarde na de markering `-connect` plaatsen. De toepassing probeert automatisch verbinding te maken met dat IP-adres of die hostnaam. Als dat niet het geval is, moet u de gegevens over het apparaat elke keer bij het starten van de foutopsporing invoeren.

4 De toepassing installeren.

Bij Android kunt u de opdracht ADT `-installApp` gebruiken:

```
adt -installApp -platform android -package DebugExample.apk
```

Bij iOS kunt u de toepassing installeren met behulp van de ADT-opdracht `-installApp` of met behulp van iTunes.

5 In een tweede terminal- of opdrachtvenster en FDB uitvoeren:

```
fdb
```

6 Typ in het FDB-venster de opdracht `run`:

```
Adobe fdb (Flash Player Debugger) [build 14159]
                                    Copyright (c) 2004-2007 Adobe, Inc. All rights reserved.
                                    (fdb) run
                                    Waiting for Player to connect
```

7 Start de toepassing op het apparaat.

8 Zodra de toepassing op het apparaat of de emulator wordt gestart, wordt het Adobe AIR-dialoogvenster voor verbinding geopend. (Als u een hostnaam of IP-adres hebt opgegeven bij de optie -connect bij het verpakken van de toepassing, probeert de toepassing automatisch verbinding te maken met behulp van dat adres.) Voer het betreffende adres in en tik op OK.

Om in deze modus verbinding te kunnen maken met het foutopsporingsprogramma moet het apparaat het adres of de hostnaam kunnen achterhalen en verbinding kunnen maken met TPC-poort 7935. Een netwerkverbinding wordt vereist.

**9** Wanneer de externe runtime verbinding maakt met het foutopsporingsprogramma, kunt u onderbrekingspunten instellen met de FDB-opdracht `break` en vervolgens de uitvoering starten met de opdracht `continue`:

```
(fdb) run
                                    Waiting for Player to connect
                                    Player connected; session starting.
                                    Set breakpoints and then type 'continue' to resume the
session.
                                    [SWF]
Users:juser:Documents:FlashProjects:DebugExample:DebugExample.swf - 32,235 bytes after
decompression
                                    (fdb) break clickHandler
                                    Breakpoint 1 at 0x5993: file DebugExample.as, line 14
                                    (fdb) continue
```

## Foutopsporing op afstand met FDB via USB

**AIR 2.6 (Android) AIR 3.3 (iOS)**

Als u een foutopsporing wilt uitvoeren via een USB-verbinding, kunt u de toepassing verpakken met de optie `-listen` in plaats van de optie `-connect`. Wanneer u de optie `-listen` opgeeft, voert de runtime bij het starten van de toepassing een detectie uit naar een verbinding van Flash Debugger (FDB) met TPC-poort 7936. Vervolgens voert u FDB uit met de optie `-p`, waarna FDB de verbinding initieert.

**USB-foutopsporingsproces voor Android**

Als u het Flash-foutopsporingsprogramma op de desktopcomputer wilt uitvoeren om verbinding te maken met de AIR-runtime op het apparaat of de emulator, moet u het hulpprogramma Android Debug Bridge (ADB) van de Android-SDK (of iOS Debug Bridge (IDB) van de AIR-SDK) gebruiken om de apparaatpoort naar de bureaubladpoort door te verbinden.

**1** Open een venster voor terminal- of opdrachtprompts en navigeer naar de map met de broncode voor de toepassing.

**2** Compileer de toepassing met amxmlc, waardoor de foutopsporing wordt ingeschakeld:

```
amxmlc -debug DebugExample.as
```

**3** Verpak de toepassing met het correcte foutopsporingsdoel (bijvoorbeeld `apk-debug`) en geef de optie `-listen` op:

```
adt -package -target apk-debug -listen -storetype pkcs12 -keystore ../../AndroidCert.p12
DebugExample.apk DebugExample-app.xml DebugExample.swf
```

**4** Sluit het apparaat met een USB-kabel aan op de computer waarop de foutopsporing wordt uitgevoerd. (U kunt deze methode ook gebruiken om fouten op te sporen bij een toepassing die wordt uitgevoerd op een emulator. In dat geval is een USB-verbinding niet noodzakelijk of mogelijk.)

**5** De toepassing installeren.

U kunt de ADT-opdracht `-installApp` gebruiken:

```
adt -installApp -platform android -package DebugExample.apk
```

**6** Verbind TCP-poort 7936 van het apparaat of de emulator door met de desktopcomputer met behulp van het hulpprogramma Android ADB:

```
adb forward tcp:7936 tcp:7936
```

7 Start de toepassing op het apparaat.

8 Voer FDB uit in een terminal- of opdrachtvenster met behulp van de optie -p:

```
fdb -p 7936
```

9 Typ in het FDB-venster de opdracht `run`:

```
Adobe fdb (Flash Player Debugger) [build 14159]
                                    Copyright (c) 2004-2007 Adobe, Inc. All rights reserved.
                                    (fdb) run
```

10 Het hulpprogramma FDB probeert verbinding met de toepassing te maken.

11 Wanneer de externe verbinding tot stand is gebracht, kunt u onderbrekingspunten instellen met de FDB-opdracht `break` en vervolgens de uitvoering starten met de opdracht `continue`:

```
(fdb) run
                                    Player connected; session starting.
                                    Set breakpoints and then type 'continue' to resume the
session.
                                    [SWF]
Users:juser:Documents:FlashProjects:DebugExample:DebugExample.swf - 32,235 bytes after
decompression
                                    (fdb) break clickHandler
                                    Breakpoint 1 at 0x5993: file DebugExample.as, line 14
                                    (fdb) continue
```

***Opmerking:*** *poort 7936 wordt door de AIR-runtime en FDB gebruikt als de standaardpoort voor USB-foutopsporing. U kunt verschillende poorten opgeven om te gebruiken met de ADT-poortparameter -listen en de FDB-poortparameter -p: In dit geval moet u het hulpprogramma Android Debug Bridge gebruiken voor het doorverbinden van het poortnummer dat in ADT is opgegeven met de poort die is opgegeven in FDB:* `adb forward tcp:adt_listen_port# tcp:fdb_port#`

**USB-foutopsporingsproces voor iOS**

Als u het Flash-foutopsporingsprogramma op de desktopcomputer wilt uitvoeren om verbinding te maken met de AIR-runtime op het apparaat of de emulator, moet u het hulpprogramma iOS Debug Bridge (IDB) van de AIR-SDK gebruiken om de apparaatpoort naar de bureaubladpoort door te verbinden.

1 Open een venster voor terminal- of opdrachtprompts en navigeer naar de map met de broncode voor de toepassing.

2 Compileer de toepassing met amxmlc, waardoor de foutopsporing wordt ingeschakeld:

```
amxmlc -debug DebugExample.as
```

3 Verpak de toepassing met het correcte foutopsporingsdoel (bijvoorbeeld `ipa-debug` of `ipa-debug-interpreter`) en geef de optie `-listen` op:

```
adt -package -target ipa-debug-interpreter -listen 16000
                                    xyz.mobileprovision -storetype pkcs12 -keystore
Certificates.p12
                                    -storepass pass123 OutputFile.ipa InputFile-app.xml
InputFile.swf
```

4 Sluit het apparaat met een USB-kabel aan op de computer waarop de foutopsporing wordt uitgevoerd. (U kunt deze methode ook gebruiken om fouten op te sporen bij een toepassing die wordt uitgevoerd op een emulator. In dat geval is een USB-verbinding niet noodzakelijk of mogelijk.)

5 Installeer en start de toepassing op het iOS-apparaat. In AIR 3.4 en hoger kunt u `adt -installApp` gebruiken om de toepassing via een USB-verbinding te installeren.

6 Bepaal de apparaathandle met behulp van de opdracht `idb -devices` (IDB bevindt zich in *air_sdk_root*`/lib/aot/bin/iOSBin/idb`):

```
./idb -devices
                                    List of attached devices
                                    Handle    UUID
                                        1     91770d8381d12644df91fbcee1c5bbdacb735500
```

***Opmerking:*** *(AIR 3.4 en hoger) U kunt ook `adt -devices` gebruiken in plaats van `idb -devices` om de apparaathandle te bepalen.*

7 U kunt een poort op uw bureaublad doorverbinden naar de poort die is opgegeven in de parameter `adt -listen` (in dit geval 16000; de standaardwaarde is 7936). Gebruik hiervoor het hulpprogramma IDB en de apparaat-id uit de voorgaande stap:

```
idb -forward 7936 16000 1
```

In dit voorbeeld is de bureaubladpoort 7936 en luistert het aangesloten apparaat naar poort 16000. De apparaat-id van het aangesloten apparaat is 1.

8 Voer FDB uit in een terminal- of opdrachtvenster met behulp van de optie -p:

```
fdb -p 7936
```

9 Typ in het FDB-venster de opdracht `run`:

```
Adobe fdb (Flash Player Debugger) [build 23201]
                                    Copyright (c) 2004-2007 Adobe, Inc. All rights reserved.
                                    (fdb) run
```

10 Het hulpprogramma FDB probeert verbinding met de toepassing te maken.

11 Wanneer de externe verbinding tot stand is gebracht, kunt u onderbrekingspunten instellen met de FDB-opdracht `break` en vervolgens de uitvoering starten met de opdracht `continue`:

***Opmerking:*** *poort 7936 wordt door de AIR-runtime en FDB gebruikt als de standaardpoort voor USB-foutopsporing. U kunt verschillende poorten opgeven om te gebruiken met de IDB-poortparameter -listen en de FDB-poortparameter -p.*

# AIR en AIR-toepassingen installeren op mobiele apparaten

Eindgebruikers van uw toepassing kunnen de AIR-runtime en AIR-toepassingen installeren met behulp van het normale toepassings- en distributiemechanisme voor het betreffende apparaat.

Bij Android kunnen gebruikers bijvoorbeeld toepassingen van de Android Market installeren. Of gebruikers kunnen, als zij toestemming hebben gegeven voor de installatie van toepassingen met een onbekende bron, een toepassing installeren door op een koppeling op een webpagina te klikken of door het toepassingspakket naar het betreffende apparaat te kopiëren en te openen. Als een gebruiker probeert om een Android-toepassing te installeren, maar nog geen AIR-runtime heeft geïnstalleerd, wordt deze automatisch naar de Market geleid, waar de gebruiker de runtime kan installeren.

Bij iOs zijn er twee manieren om toepassingen naar eindgebruikers te distribueren. Het primaire distributiekanaal is de Apple App Store. U kunt ook ad-hocdistributie gebruiken om een beperkt aantal gebruikers uw toepassing te laten installeren zonder van de App Store gebruik te hoeven maken.

## De AIR-runtime en -toepassingen installeren voor ontwikkeling

Aangezien AIR-toepassingen als native pakketten op mobiele apparaten zijn geïnstalleerd, kunt u de normale platformfaciliteiten gebruiken voor het installeren van toepassingen voor testdoeleinden. U kunt ADT-opdrachten gebruiken om de AIR-runtime en -toepassingen te installeren wanneer deze worden ondersteund. Momenteel wordt deze benadering ondersteund door Android.

Bij iOS kunt u toepassingen voor testdoeleinden installeren met behulp van iTunes. Testoepassingen moeten worden ondertekend met een ondertekeningscertificaat voor Apple-code dat speciaal wordt uitgegeven voor toepassingsontwikkeling en moeten worden verpakt met een ontwikkelingsinrichtingsprofiel. Een AIR-toepassing is bij iOS een op zichzelf staand pakket. Er wordt geen afzonderlijke runtime gebruikt.

### AIR-toepassingen installeren met behulp van ADT

Wanneer u AIR-toepassingen ontwikkelt, kunt u ADT gebruiken om de runtime en uw toepassingen te installeren en om de installatie ervan ongedaan te maken. (Mogelijk worden deze opdrachten door uw IDE geïntegreerd zodat u ADT niet zelf hoeft uit te voeren.)

U kunt de AIR-runtime op een apparaat of emulator installeren met behulp van het AIR-hulpprogramma ADT. De SDK behorend bij het apparaat moet worden geïnstalleerd. De opdracht `-installRuntime` gebruiken:

```
adt -installRuntime -platform android -device deviceID -package path-to-runtime
```

Wanneer de parameter `-package` niet is opgegeven, wordt het relevante runtimepakket voor het apparaat of de emulator gekozen uit de beschikbare runtimepakketten in de geïnstalleerde AIR-SDK.

Als u een AIR-toepassing wilt installeren op Android of iOS (AIR 3.4 en hoger), gebruikt u de gelijksoortige opdracht `-installApp`:

```
adt -installApp -platform android -device deviceID -package path-to-app
```

De waarde die is ingesteld voor het argument `-platform` moet overeenkomen met het apparaat waarop u de toepassing installeert.

***Opmerking:*** *bestaande versies van de AIR-runtime of de AIR-toepassing moeten worden verwijderd voordat deze opnieuw worden geïnstalleerd.*

### AIR-toepassingen installeren op iOS-apparaten met behulp van iTunes

Een AIR-toepassing installeren op een iOS-apparaat om deze te testen:

1. Open de toepassing iTunes.
2. Als u dit nog niet hebt gedaan, voegt u het inrichtingsprofiel voor deze toepassing aan iTunes toe. Selecteer in iTunes Archief > Voeg toe aan bibliotheek. Selecteer vervolgens het inrichtingsprofielbestand (met mobileprovision als bestandstype).
3. In sommige versies van iTunes wordt de toepassing niet vervangen als dezelfde versie van de toepassing al is geïnstalleerd. Verwijder in dit geval de toepassing van het apparaat en uit de lijst met toepassingen in iTunes.
4. Dubbelklik op het IPA-bestand voor de toepassing. De toepassing moet in uw lijst met iTunes-toepassingen worden weergegeven.
5. Sluit uw apparaat aan op de USB-poort van uw computer.
6. Controleer in iTunes het tabblad Toepassing voor het apparaat en zorg ervoor dat de toepassing is geselecteerd in de lijst met toepassingen die moeten worden geïnstalleerd.
7. Selecteer het apparaat in de lijst links van iTunes. Klik op de knop Synchroniseren. Wanneer het synchronisatieproces is voltooid, verschijnt de Hello World-toepassing op uw iPhone.

Als de nieuwe versie niet is geïnstalleerd, verwijdert u deze van uw apparaat en uit de lijst met toepassingen in iTunes en herhaalt u vervolgens deze procedure. Dit kan zich voordoen als de versie die op dat moment is geïnstalleerd, dezelfde toepassings-id en toepassingsversie gebruikt.

### Meer Help-onderwerpen

"ADT-opdracht installRuntime" op pagina 185

"ADT-opdracht installApp" op pagina 182

## AIR-toepassingen op een apparaat uitvoeren

U kunt geïnstalleerde AIR-toepassingen starten met behulp van de gebruikersinterface van het apparaat. U kunt ook toepassingen op afstand starten met behulp van het AIR-hulpprogramma ADT (wanneer dit wordt ondersteund):

```
adt -launchApp -platform android -device deviceID -appid applicationID
```

De waarde van het argument `-appid` moet de AIR-toepassings-id zijn van de te starten AIR-toepassing. Gebruik de waarde die is opgegeven in de AIR-toepassingsdescriptor (zonder het voorvoegsel *air.* toe te voegen tijdens het verpakken).

Wanneer slechts één apparaat of emulator is aangesloten en wordt uitgevoerd, kunt u de markering `-device` weglaten. De waarde die is ingesteld voor het argument `-platform` moet overeenkomen met het apparaat waarop u de toepassing installeert. Momenteel is de enige ondersteunde waarde *android*.

## AIR-runtime en AIR-toepassingen verwijderen

U kunt voor het verwijderen van toepassingen de gangbare methoden gebruiken die worden geboden door het besturingssysteem van het apparaat. U kunt ook het AIR-hulpprogramma ADT gebruiken om de AIR-runtime en de AIR-toepassingen te verwijderen indien ADT wordt ondersteund. Als u de runtime wilt verwijderen, gebruikt u de opdracht `uninstallRuntime`:

```
adt -uninstallRuntime -platform android -device deviceID
```

Als u een toepassing wilt verwijderen, gebruikt u de opdracht `uninstallApp`:

```
adt -uninstallApp -platform android -device deviceID -appid applicationID
```

Wanneer slechts één apparaat of emulator is aangesloten en wordt uitgevoerd, kunt u de markering `-device` weglaten. De waarde die is ingesteld voor het argument `-platform` moet overeenkomen met het apparaat waarop u de toepassing installeert. Momenteel is de enige ondersteunde waarde *android*.

## Een emulator installeren

Als u uw AIR-toepassing op een apparaatemulator wilt uitvoeren, moet u doorgaans de SDK voor het apparaat gebruiken om een emulatorinstantie op uw ontwikkelingscomputer te maken en uit te voeren. U kunt vervolgens de emulatorversie van de AIR-runtime en uw AIR-toepassing op uw emulator installeren. Houd er rekening mee dat toepassingen op een emulator doorgaans veel langzamer worden uitgevoerd dan op het werkelijke apparaat.

### Een Android-emulator maken

1 De Android-SDK en het hulpprogramma AVD Manager starten:
   - Bij Windows voert u het bestand SDK Setup.exe uit in de hoofdmap van de Android-SDK-directory.
   - Bij Mac OS, moet u de Android-toepassing uitvoeren in de submap Tools van de Android-SDK-directory

2 Selecteer de optie Settings en selecteer de optie "Force https://".

3 Selecteer de optie Available Packages. Er moet een lijst worden weergegeven met beschikbare Android-SDK's.

4 Selecteer een compatibele Android-SDK (Android 2.3 of later) en klik op de knop Install Selected.

5 Selecteer de optie Virtual Devices en klik op de knop New.

6 Maak de volgende instellingen:

- Een naam voor het virtuele apparaat
- De doel-API, zoals Android 2.3, API-niveau 8
- Een grootte voor de SD-kaart (zoals 1024)
- Een skin (zoals Default HVGA)

7 Klik op de knop Create AVD.

Houd er rekening mee dat, afhankelijk van uw systeemconfiguratie, het maken van een virtueel apparaat enige tijd kan duren.

U kunt nu uw nieuwe virtuele apparaat starten.

1 Selecteer Virtual Device in de toepassing AVD Manager. Het virtuele apparaat dat u hierboven hebt gemaakt, moet worden vermeld.

2 Selecteer het virtuele apparaat en klik vervolgens op de starttoets.

3 Klik op de knop Launch op het volgende scherm.

Er moet een emulatorvenster op uw desktop worden geopend. Dit kan enige seconden duren. Het kan ook enige tijd duren voordat het Android-besturingssysteem wordt geïnitialiseerd. U kunt toepassingen op een emulator installeren die met *apk-debug* en *apk-emulator* zijn verpakt. Toepassingen die zijn verpakt met het doel *apk* werken niet op een emulator.

### Meer Help-onderwerpen

http://developer.android.com/guide/developing/tools/othertools.html#android

http://developer.android.com/guide/developing/tools/emulator.html

# Mobiele AIR-toepassingen bijwerken

Mobiele AIR-toepassingen worden als native pakketten gedistribueerd en gebruiken dus de standaardupdateprocedures van andere toepassingen op het platform. Dit betekent doorgaans indiening bij dezelfde market of toepassingenstore als voor distributie van de oorspronkelijke toepassing.

Mobiele AIR-toepassingen kunnen de AIR Updater-klasse of het AIR Updater-framework niet gebruiken.

## AIR-toepassingen bijwerken op Android

Voor toepassingen die zijn gedistribueerd op de Android Market, kunt u een toepassing bijwerken door een nieuwe versie op de Market te plaatsen, voor zover aan de onderstaande voorwaarden wordt voldaan (dit beleid wordt vereist door de Market, niet door AIR)

- Het APK-pakket wordt ondertekend met hetzelfde certificaat.
- De AIR-id is dezelfde.

- De waarde `versionNumber` in de toepassingsdescriptor is hoger. (U moet de waarde `versionLabel` ook verhogen, indien u deze gebruikt.)

Gebruikers die uw toepassing op Android Market hebben gedownload, worden via de apparaatsoftware op de hoogte gesteld dat er een update beschikbaar is.

### Meer Help-onderwerpen

Android-ontwikkelaars: updates publiceren op Android Market

## AIR-toepassingen bijwerken op iOS

Via de iTunes App Store gedistribueerde AIR-toepassingen kunt u bijwerken door de update in te dienen bij de Store, als u zich aan alle volgende regels houdt (deze regels worden opgelegd door Apple, niet door AIR):

- Het ondertekeningscertificaat en de inrichtingsprofielen voor de code hebben dezelfde Apple-id
- Het IPA-pakket gebruikt dezelfde Apple-bundle-id
- De update beperkt het aantal ondersteunde apparaten niet (met andere woorden: als uw oorspronkelijke toepassing ondersteuning bood voor apparaten met iOS 3, mag u geen update maken die geen ondersteuning biedt voor iOS 3).

***Belangrijk:*** *aangezien AIR SDK versie 2.6 en hoger geen ondersteuning biedt voor iOS 3, maar AIR 2 wel, kunt u gepubliceerde iOS-toepassingen die zijn ontwikkeld met AIR 2 niet bijwerken met een update die is ontwikkeld met gebruik van AIR 2.6+.*

# Pushberichten gebruiken

Met pushberichten kunnen externe-berichtenproviders hun berichten sturen naar toepassingen die op een mobiel apparaat worden uitgevoerd. AIR 3.4 biedt ondersteuning voor pushberichten voor iOS-apparaten met APNs (Apple Push Notification service).

***Opmerking:*** *als u pushberichten voor een AIR for Android-toepassing wilt inschakelen, moet u een native extensie gebruiken, zoals as3c2dm, ontwikkeld door Adobe-goeroe Piotr Walczyszyn.*

In het resterende gedeelte van deze sectie wordt beschreven hoe u pushberichten kunt inschakelen in AIR for iOS-toepassingen.

***Opmerking:*** *in de discussie wordt aangenomen dat u beschikt over een Apple-ontwikkelaars-id, bekend bent met de iOS-ontwikkelingsworkflow en al minimaal één toepassing hebt geïmplementeerd op een iOS-apparaat.*

## Overzicht van pushberichten

Met APNs (Apple Push Notification service) kunnen externe-berichtenproviders hun berichten sturen naar toepassingen die op iOS-apparaten worden uitgevoerd. APNs biedt ondersteuning voor de volgende berichttypen:

- Alerts (waarschuwingsberichten)
- Badges
- Geluiden

Voor meer informatie over APNs gaat u naar developer.apple.com.

Er is een aantal aspecten waarmee u te maken krijgt wanneer u pushberichten in uw toepassing gaat gebruiken:

- **Clienttoepassing**: meldt zich aan voor pushberichten, communiceert met de externe-berichtenproviders en ontvangt de pushberichten.
- **iOS**: beheert de interactie tussen de clienttoepassing en APNs.
- **APNs**: levert een tokenID tijdens de clientregistratie en geeft berichten door van de externe-berichtenproviders aan iOS.
- **Externe-berichtenprovider**: slaat informatie op over de tokenId-client-toepassing en levert pushberichten aan APNs.

## Registratieworkflow

De workflow voor de registratie van pushberichten met een serverservice is als volgt:

1. De clienttoepassing dient een verzoek in bij iOS om pushberichten in te schakelen.
2. iOS stuurt het verzoek door naar APNs.
3. De APNs-server retourneert een tokenId naar iOS.
4. iOS retourneert het tokenId naar de clienttoepassing.
5. De clienttoepassing biedt het tokenId (via een toepassingsspecifiek mechanisme) aan bij de externe-berichtenprovider, die het tokenId opslaat voor gebruik met pushberichten.

## Berichtenworkflow

De berichtenworkfllow is als volgt:

1. De externe-berichtenprovider genereert een bericht en geeft de berichtlading door aan APNs, samen met het tokenId.
2. APNs stuurt het bericht door naar iOS op het apparaat.
3. iOS stuurt de berichtlading via pushtechnologie door naar de toepassing.

## Pushbericht-API

In AIR 3.4 wordt een set met API's geïntroduceerd die ondersteuning bieden voor iOS-pushberichten. Deze API's bevinden zich in het `flash.notifications`-pakket. Het pakket bevat onder andere de volgende klassen:

- `NotificationStyle`: definieert constanten voor berichttypen: `ALERT`, `BADGE` en `SOUND`.C
- `RemoteNotifier`: hiermee kunt u zich abonneren op pushberichten, of uw abonnement opzeggen.
- `RemoteNotifierSubscribeOptions`: hiermee kunt u aangeven welke berichttypen u wilt ontvangen. Gebruik de eigenschap `notificationStyles` om een vector met tekenreeksen te definiëren waarmee u zich registreert voor meerdere berichttypen.

AIR 3.4 bevat ook `flash.events.RemoteNotificationEvent`, die wordt verzonden door `RemoteNotifier`, als volgt:

- Wanneer het abonnement van een toepassing correct is ingesteld en een nieuw tokenId wordt ontvangen van APNs.
- Bij het ontvangen van een nieuw extern bericht.

`RemoteNotifier` verzendt bovendien `flash.events.StatusEvent` als er een fout optreedt bij het abonnementsproces.

## Pushberichten in een toepassing beheren

Als u een toepassing wilt registreren voor gebruik van pushberichten, moet u de volgende stappen uitvoeren:

- Creëer code waarmee de toepassing zich abonneert op pushberichten.
- Schakel de optie voor pushberichten in in het XML-bestand van de toepassing.
- Creëer een inrichtingsprofiel en -certificaat waarmee iOS Push Services wordt ingeschakeld.

In de volgende voorbeeldcode ziet u hoe u een toepassing abonneert op pushberichten en hoe de pushberichtgebeurtenissen worden verwerkt:

```
package
                        {
                        import flash.display.Sprite;
                        import flash.display.StageAlign;
                        import flash.display.StageScaleMode;
                        import flash.events.*;
                        import flash.events.Event;
                        import flash.events.IOErrorEvent;
                        import flash.events.MouseEvent;
                        import flash.net.*;
                        import flash.text.TextField;
                        import flash.text.TextFormat;
                        import flash.ui.Multitouch;
                        import flash.ui.MultitouchInputMode;
                        // Required packages for push notifications
                        import flash.notifications.NotificationStyle;
                        import flash.notifications.RemoteNotifier;
                        import flash.notifications.RemoteNotifierSubscribeOptions;
                        import flash.events.RemoteNotificationEvent;
                        import flash.events.StatusEvent;
                        [SWF(width="1280", height="752", frameRate="60")]
                        public class TestPushNotifications extends Sprite
                        {
                        private var notiStyles:Vector.<String> = new Vector.<String>;;
                        private var tt:TextField = new TextField();
                        private var tf:TextFormat = new TextFormat();
                        // Contains the notification styles that your app wants to receive
                        private var preferredStyles:Vector.<String> = new Vector.<String>();
                        private var subscribeOptions:RemoteNotifierSubscribeOptions = new
RemoteNotifierSubscribeOptions();
                        private var remoteNot:RemoteNotifier = new RemoteNotifier();
                        private var subsButton:CustomButton = new CustomButton("Subscribe");
                        private var unSubsButton:CustomButton = new
CustomButton("UnSubscribe");
                        private var clearButton:CustomButton = new CustomButton("clearText");
                        private var urlreq:URLRequest;
                        private var urlLoad:URLLoader = new URLLoader();
                        private var urlString:String;
                        public function TestPushNotifications()
                        {
                        super();
                        Multitouch.inputMode = MultitouchInputMode.TOUCH_POINT;
                        stage.align = StageAlign.TOP_LEFT;
                        stage.scaleMode = StageScaleMode.NO_SCALE;
                        tf.size = 20;
                        tf.bold = true;
```

```
                        tt.x=0;
                        tt.y =150;
                        tt.height = stage.stageHeight;
                        tt.width = stage.stageWidth;
                        tt.border = true;
                        tt.defaultTextFormat = tf;
                        addChild(tt);
                        subsButton.x = 150;
                        subsButton.y=10;
                        subsButton.addEventListener(MouseEvent.CLICK,subsButtonHandler);
                        stage.addChild(subsButton);
                        unSubsButton.x = 300;
                        unSubsButton.y=10;
                        unSubsButton.addEventListener(MouseEvent.CLICK,unSubsButtonHandler);
                        stage.addChild(unSubsButton);
                        clearButton.x = 450;
                        clearButton.y=10;
                        clearButton.addEventListener(MouseEvent.CLICK,clearButtonHandler);
                        stage.addChild(clearButton);
                        //
                        tt.text += "\n SupportedNotification Styles: " +
RemoteNotifier.supportedNotificationStyles.toString() + "\n";
                        tt.text += "\n Before Preferred notificationStyles: " +
subscribeOptions.notificationStyles.toString() + "\n";
                        // Subscribe to all three styles of push notifications:
                        // ALERT, BADGE, and SOUND.
                        preferredStyles.push(NotificationStyle.ALERT
,NotificationStyle.BADGE,NotificationStyle.SOUND );
                        subscribeOptions.notificationStyles= preferredStyles;
                        tt.text += "\n After Preferred notificationStyles:" +
subscribeOptions.notificationStyles.toString() + "\n";

remoteNot.addEventListener(RemoteNotificationEvent.TOKEN,tokenHandler);

remoteNot.addEventListener(RemoteNotificationEvent.NOTIFICATION,notificationHandler);
                        remoteNot.addEventListener(StatusEvent.STATUS,statusHandler);
                        this.stage.addEventListener(Event.ACTIVATE,activateHandler);
                        }
                        // Apple recommends that each time an app activates, it subscribe for
                        // push notifications.
                        public function activateHandler(e:Event):void{
                        // Before subscribing to push notifications, ensure the device supports
it.
                        // supportedNotificationStyles returns the types of notifications
                        // that the OS platform supports
                        if(RemoteNotifier.supportedNotificationStyles.toString() != "")
                        {
                        remoteNot.subscribe(subscribeOptions);
                        }
                        else{
                        tt.appendText("\n Remote Notifications not supported on this Platform
!");
                        }
                        }
                        public function subsButtonHandler(e:MouseEvent):void{
                        remoteNot.subscribe(subscribeOptions);
                        }
```

```
                        // Optionally unsubscribe from push notfications at runtime.
                        public function unSubsButtonHandler(e:MouseEvent):void{
                        remoteNot.unsubscribe();
                        tt.text +="\n UNSUBSCRIBED";
                        }
                        public function clearButtonHandler(e:MouseEvent):void{
                        tt.text = " ";
                        }
                        // Receive notification payload data and use it in your app
                        public function notificationHandler(e:RemoteNotificationEvent):void{
                        tt.appendText("\nRemoteNotificationEvent type: " + e.type +
                        "\nbubbles: "+ e.bubbles + "\ncancelable " +e.cancelable);
                        for (var x:String in e.data) {
                        tt.text += "\n"+ x + ":  " + e.data[x];
                        }
                        }
                        // If the subscribe() request succeeds, a RemoteNotificationEvent of
                        // type TOKEN is received, from which you retrieve e.tokenId,
                        // which you use to register with the server provider (urbanairship, in
                        // this example.
                        public function tokenHandler(e:RemoteNotificationEvent):void
                        {
                        tt.appendText("\nRemoteNotificationEvent type: "+e.type +"\nBubbles:
"+ e.bubbles + "\ncancelable " +e.cancelable +"\ntokenID:\n"+ e.tokenId +"\n");
                        urlString = new
String("https://go.urbanairship.com/api/device_tokens/" +
                        e.tokenId);
                        urlreq = new URLRequest(urlString);
                        urlreq.authenticate = true;
                        urlreq.method = URLRequestMethod.PUT;
                        URLRequestDefaults.setLoginCredentialsForHost
                        ("go.urbanairship.com",
                        "1ssB2iV_RL6_UBLiYMQVfg","t-kZlzXGQ6-yU8T3iHiSyQ");
                        urlLoad.load(urlreq);
                        urlLoad.addEventListener(IOErrorEvent.IO_ERROR,iohandler);
                        urlLoad.addEventListener(Event.COMPLETE,compHandler);
                        urlLoad.addEventListener(HTTPStatusEvent.HTTP_STATUS,httpHandler);
                        }
                        private function iohandler(e:IOErrorEvent):void
                        {
                        tt.appendText("\n In IOError handler" + e.errorID +" " +e.type);
                        }
                        private function compHandler(e:Event):void{
                        tt.appendText("\n In Complete handler,"+"status: " +e.type + "\n");
                        }
                        private function httpHandler(e:HTTPStatusEvent):void{
                        tt.appendText("\n in httpstatus handler,"+ "Status: " + e.status);
                        }
                        // If the subscription request fails, StatusEvent is dispatched with
                        // error level and code.
                        public function statusHandler(e:StatusEvent):void{
                        tt.appendText("\n statusHandler");
                        tt.appendText("event Level" + e.level +"\nevent code " +
                        e.code + "\ne.currentTarget: " + e.currentTarget.toString());
                        }
                        }
                        }
```

## Pushberichten inschakelen in het XML-bestand van de toepassing

Als u pushberichten wilt gebruiken in uw toepassing, moet u het volgende invoeren in de `Entitlements`-tag (onder de `iphone`-tag):

```
<iphone>
                                ...
                                   <Entitlements>
                                      <![CDATA[
                                         <key>aps-environment</key>
                                         <string>development</string>
                                      ]]>
                                   </Entitlements>
                                </iphone>
```

Wanneer u gereed bent om de toepassing via pushtechnologie aan te melden bij de App Store, een `<string>`-element voor ontwikkeling naar productie:

```
<string>production</string>
```

Als uw toepassing ondersteuning biedt voor gelokaliseerde tekenreeksen, moet u de desbetreffende talen opgeven in de `supportedLanguages`-tag, onder de `intialWindow`-tag, zoals in het volgende voorbeeld:

```
<supportedLanguages>en de cs es fr it ja ko nl pl pt</supportedLanguages>
```

## Een inrichtingsprofiel en -certificaat maken voor iOS Push Services

Als u de communicatie tussen de toepassing en APNs wilt inschakelen, moet u de toepassing in een pakket plaatsen met een inrichtingsprofiel en -certificaat voor iOS Push Services, als volgt:

1. Meld u aan bij uw Apple-ontwikkelaarsaccount.
2. Ga naar de portal Provisioning (Inrichting).
3. Klik op de tab App IDs (Toepassings-id's).
4. Klik op de knop New App ID (Nieuwe toepassings-id).
5. Voer een beschrijving en een bundel-id in (gebruik geen asterisk * in de bundel-id).
6. Klik op Submit (Verzenden). De portal Provisioning (Inrichting) genereert vervolgens uw toepassings-id en de pagina App IDs (Toepassings-id's) wordt weergegeven.
7. Klik op Configure (Configureren), rechts van uw toepassings-id. De pagina Configure App ID (Toepassings-id configureren) wordt weergegeven.
8. Schakel het selectievakje Enable for Apple Push Notification service (Inschakelen voor APNs) in. Opmerking: er zijn twee typen Push-SSL-certificaat: een voor onwikkeling/testen en een voor productiedoeleinden.
9. Klik op de knop Configure (Configureren), rechts van het push-SSL-certificaat voor ontwikkeling. De pagina Generate Certificate Signing Request (CSR genereren) wordt weergegeven.
10. Volg de instructies op de pagina om een CSR met het hulpprogramma Keychain Access te maken.
11. Genereer het SSL-certificaat.
12. Download en installeer het SSL-certificaat.
13. (Optioneel) Herhaal stap 9-12 voor het Push-SSL-certificaat voor productiedoeleinden.
14. Klik op Done (Gereed). De pagina Configure App ID (Toepassings-id configureren) wordt weergegeven.
15. Klik op Done (Gereed). De pagina App IDs (Toepassings-id's) wordt weergegeven. Controleer of er een groene cirkel wordt weergegeven naast het pushbericht voor uw toepassings-id.

**16** Sla uw SSL-certificaten op. Deze worden later gebruikt voor de communicatie tussen de toepassing en de provider.

**17** Klik op de tab Provisioning (Inrichting) om de pagina Provisioning Profiles (Inrichtingsprofielen) weer te geven.

**18** Maak een inrichtingsprofiel voor uw nieuwe toepassings-id en download dit profiel.

**19** Klik op de tab Certificates (Certificaten) en download een nieuw certificaat voor het nieuwe inrichtingsprofiel.

### Geluidsopties gebruiken voor pushberichten

Als u geluidsopties wilt inschakelen voor de pushberichten op uw toepassing, moet u de geluidsbestanden bundelen, net als elk ander element, maar wel in dezelfde directory als de SWF- en app-xml-bestanden. Bijvoorbeeld:

```
Build/adt -package -target ipa-app-store -provisioning-profile _-_.mobileprovision -storetype
pkcs12 -keystore _-_.p12 test.ipa test-app.xml test.swf sound.caf sound1.caf
```

Apple biedt ondersteuning voor de volgende geluidsindelingen (in AIFF-, WAV- of CAF-bestanden):

- Linear PCM
- MA4 (IMA/ADPCM)
- uLaw
- aLaw

### Gelokaliseerde waarschuwingsberichten gebruiken

Als u gelokaliseerde waarschuwingsberichten wilt gebruiken in uw toepassing, moet u de gelokaliseerde tekenreeksen bundelen in de vorm van lproj-mappen. Stel bijvoorbeeld dat uw toepassing ondersteuning biedt voor Spaanse waarschuwingsberichten, zoals volgt:

**1** Maak een es.lproj-map in het project op hetzelfde niveau als het app-xml-bestand.

**2** Maak een tekstbestand met de naam Localizable.Strings in de map es.lproj.

**3** Open het bestand Localizable.Strings in een teksteditor en voeg berichtsleutels en de bijbehorende gelokaliseerde tekenreeksen toe. Bijvoorbeeld:

```
"PokeMessageFormat" = "La notificación de alertas en español."
```

**4** Sla het bestand op.

**5** Wanneer de toepassing een waarschuwingsbericht ontvangt met deze sleutelwaarde en het apparaat is ingesteld op Spaans, wordt de vertaalde waarschuwingstekst weergegeven.

## Een externe-berichtenprovider configureren

Als u pushberichten wilt versturen naar uw toepassing, hebt u een externe-berichtenprovider nodig. Deze servertoepassing fungeert als provider: uw pushinvoer wordt geaccepteerd en het bericht en de bijbehorende berichtlading wordt doorgegeven aan APNs, waarna het pushbericht wordt doorgestuurd naar een clienttoepassing.

Voor gedetailleerde informatie over pushberichten van een externe-berichtenprovider gaat u naar Provider Communication with Apple Push Notification Service (Providercommunicatie via APNs) in de Apple-ontwikkelaarsbibliotheek.

## Opties voor de externe-berichtenprovider

U kunt onder andere de volgende opties voor de externe-berichtenprovider instellen:

- Maak uw eigen provider op basis van de APNS-php open-source-server. U kunt een PHP-server instellen met behulp van http://code.google.com/p/apns-php/. Met dit Google Code-project kunt u een interface ontwerpen die voldoet aan uw specifieke vereisten.
- Gebruik een serviceprovider. Zo vindt u bijvoorbeeld een kant-en-klare APNs-provider op http://urbanairship.com/. Nadat u zich bij deze service hebt geregistreerd, kunt u uw apparaat-token aanbieden via code die lijkt op de volgende code:

```
private var urlreq:URLRequest;
                                    private var urlLoad:URLLoader = new URLLoader();
                                    private var urlString:String;
                                    //When subscription is successful then only call the
following code
                                    urlString = new
String("https://go.urbanairship.com/api/device_tokens/" + e.tokenId);
                                    urlreq = new URLRequest(urlString);
                                    urlreq.authenticate = true;
                                    urlreq.method = URLRequestMethod.PUT;

URLRequestDefaults.setLoginCredentialsForHost("go.urbanairship.com",
                                       "Application Key","Application Secret");
                                    urlLoad.load(urlreq);

urlLoad.addEventListener(IOErrorEvent.IO_ERROR,iohandler);
                                    urlLoad.addEventListener(Event.COMPLETE,compHandler);

urlLoad.addEventListener(HTTPStatusEvent.HTTP_STATUS,httpHandler);
                                    private function iohandler(e:IOErrorEvent):void{
                                       trace("\n In IOError handler" + e.errorID +" "
+e.type);
                                    }
                                    private function compHandler(e:Event):void{
                                       trace("\n In Complete handler,"+"status: " +e.type +
"\n");
                                    }
                                    private function httpHandler(e:HTTPStatusEvent):void{
                                       tt.appendText("\n in httpstatus handler,"+ "Status:
" + e.status);
                                    }
```

Vervolgens kunt u testberichten verzenden met de hulpprogramma's van Urban Airship.

## Certificaten voor de externe-berichtenprovider

U moet het SSL-certificaat en de privésleutel (die u eerder hebt gegenereerd) kopiëren naar de juiste locatie op de server van de externe-berichtenprovider. Deze twee bestanden worden normaal gesproken gecombineerd in een enkel `.pem`-bestand. Hiervoor voert u de volgende stappen uit:

1. Open een terminalvenster.
2. Maak een `.pem`-bestand van het SSL-certificaat door de volgende opdracht te typen:

   ```
   openssl x509 -in aps_developer_identity.cer -inform der -out TestPushDev.pem
   ```

3. Maak een .pem-betand van het privésleutelbestand (`.p12`) door de volgende opdracht te typen:

```
openssl pkcs12 -nocerts -out TestPushPrivateKey.pem -in certificates.p12
```

4 Combineer beide .pem-bestanden tot een enkel bestand met de volgende opdracht:

```
cat TestPushDev.pem TestPushPrivateKey.pem > FinalTestPush.pem
```

5 Biedt het gecombineerde `.pem`-bestand aan bij de serverprovider wanneer u de serverpushtoepassing maakt.

Voor meer informatie gaat u naar Installing the SSL Certificate and Key on the Server (SSL-certificaat en -sleutel op de server installeren) in de Apple-programmeergids getiteld 'Apple Local and Push Notification Programming Guide'.

## Pushberichten in een toepassing verwerken

Bij de verwerking van pushberichten in een toepassing wordt de volgende procedure doorlopen:

- Algemene gebruikersconfiguratie en acceptatie van pushberichten
- Gebruikersacceptatie van individuele pushberichten
- Verwerking van pushberichten en berichtladinggegevens

### Configuratie en acceptatie van pushberichten

Wanneer een gebruiker voor het eerst een toepassing start waarbij de functie voor pushberichten is ingeschakeld, wordt het ***appname*-dialoogvenster 'Would Like to Send You Push Notifications' (Wilt u pushberichten verzenden)** weergegeven. Dit dialoogvenster bevat de knoppen Don't Allow (Niet toestaan) en OK. Als de gebruiker op OK klikt, kan de toepassing alle berichttypen ontvangen waarop de toepassing is geabonneerd. Als de gebruiker geen pushberichten toestaat, ontvangt de toepassing geen berichten.

***Opmerking:*** *via de optie Settings > Notifications (Instellingen > Berichten) kan de gebruiker aangeven welke specifieke berichttypen al dan niet zijn toegestaan voor elke toepassing waarbij de pushtechnologie is ingeschakeld.*

Apple raadt aan dat toepassingen zich na elke activering abonneren op pushberichten. Wanneer uw toepassing `RemoteNotifier.subscribe()` aanroept, ontvangt de toepassing een `RemoteNotificationEvent` van het type `token`. Dit token bevat een uniek numeriek tokenID van 32 bits waarmee de desbetreffende toepassing op het specifieke apparaat wordt geïdentificeerd.

Wanneer het apparaat een pushbericht ontvangt, wordt een pop-upvenster weergegeven met de knoppen Close en Launch (respectievelijk Sluiten en Starten). Als de gebruiker Close (Sluiten) aantikt, gebeurt er niets. Als de gebruiker Launch (Starten) aantikt, wordt de toepassing door iOS aangeroepen en ontvangt de toepassing een `flash.events.RemoteNotificationEvent` van het type `notification`, zoals hieronder beschreven.

### Pushberichten en gegevens van berichtladingen verwerken

Wanneer de externe-berichtenprovider een bericht verstuurt naar een apparaat (met behulp van het tokenID), ontvangt uw toepassing een `flash.events.RemoteNotificationEvent` van het type `notification`, onafhankelijk of de toepassing al dan niet wordt uitgevoerd. Op dit punt wordt een toepassingsspecifieke berichtverwerking uitgevoerd door uw toepassing. Als de berichtgegevens door uw toepassing worden verwerkt, hebt u toegang hiertoe via de eigenschap `RemoteNotificationEvent.data` die gebruikmaakt van de JSON-indeling.

# Hoofdstuk 8: AIR-toepassingen voor televisie-apparaten ontwikkelen

## AIR-functionaliteit voor tv's

U kunt Adobe® AIR®-toepassingen ontwikkelen voor tv-apparaten, zoals televisies, digitale videorecorders en Blu-ray-spelers, als het desbetreffende apparaat over Adobe AIR for TV beschikt. AIR for TV is geoptimaliseerd voor tv-apparaten. De hardwareversnellers van een apparaat worden bijvoorbeeld gebruikt om video en grafische afbeeldingen van hoge kwaliteit te genereren.

AIR-toepassingen voor tv-apparaten zijn toepassingen op basis van SWF, niet van HTML. Uw AIR for TV-toepassing kan optimaal gebruikmaken van de hardwareversnelling, net als van de overige AIR-functionaliteit die uiterst geschikt is voor 'woonkameromgevingen'.

### Apparaatprofielen

AIR maakt gebruik van profielen om een groep apparaten met gelijke mogelijkheden als doel te definiëren. Gebruik de volgende profielen voor AIR for TV-toepassingen:

- Het `tv`-profiel. Gebruik dit profiel in AIR-toepassingen die zijn bedoeld voor een AIR for TV-apparaat.
- Het `extendedTV`-profiel. Gebruik dit profiel als uw AIR for TV-toepassing gebruikmaakt van native extensies.

De ActionScript-mogelijkheden die zijn gedefinieerd voor deze profielen, worden besproken in “Apparaatprofielen” op pagina 257. Specifieke ActionScript-verschillen bij AIR for TV-toepassingen zijn opgenomen in Naslaggids voor ActionScript 3.0 voor het Adobe Flash-platform.

Zie “Ondersteunde profielen” op pagina 148 voor informatie over AIR for TV-profielen.

### Hardwareversnelling

Televisieapparaten verschaffen hardwareversnellers die de prestaties van afbeeldingen en video's in uw AIR-toepassing aanzienlijk verbeteren. Zie “Aandachtspunten ontwerp AIR for TV-toepassing” op pagina 129 voor informatie over deze hardwareversnellers.

### Inhoud beschermen

Met AIR for TV kunt u de consument veelzijdige ervaringen bieden op gebied van hoogwaardige video-inhoud, van Hollywood-kaskrakers tot minder bekende films en afleveringen van tv-series. Leveranciers van inhoud kunnen interactieve toepassingen maken met de programma's van Adobe. Ze kunnen Adobe-serverproducten opnemen in de distributie-infrastructuur van hun inhoud of samenwerken met een van de ecosysteempartners van Adobe.

Inhoudbeveiliging is een belangrijke vereiste voor de distributie van kwaliteitsvideo. AIR for TV biedt ondersteuning voor Adobe® Flash® Access™, een oplossing voor bescherming en monetisatie van inhoud die voldoet aan de strenge beveiligingsvereisten van eigenaars van inhoud, zoals de grote filmstudio's.

Flash Access ondersteunt het volgende:

- Streamen en downloaden van video.
- Verschillende bedrijfsmodellen, zoals advertenties, abonnementen, verhuur en elektronische doorverkoop.

- Verschillende technologieën voor het leveren van inhoud, zoals HTTP Dynamic Streaming, streaming via RTMP (Real Time Media Protocol) met gebruik van Flash® Media Server en progressief downloaden met HTTP.

AIR for TV biedt bovendien geïntegreerde ondersteuning voor RTMPE, de gecodeerde versie van RTMP, voor bestaande streamingoplossingen met lagere beveiligingsvereisten. Flash Media Server biedt ondersteuning voor RTMPE en verwante SWF-verificatietechnologieën.

Zie Toegang tot Adobe Flash voor meer informatie.

## Meerkanaalsaudio

Vanaf AIR 3 ondersteunt AIR for TV meerkanaalsaudio voor video's die progressief worden gedownload van een HTTP-server. De volgende codecs worden ondersteund:

- AC-3 (Dolby Digital)
- E-AC-3 (Enhanced Dolby Digital)
- DTS Digital Surround
- DTS Express
- DTS-HD High Resolution Audio
- DTS-HD Master Audio

***Opmerking:*** *Multikanaalsaudio wordt nog niet ondersteund in video's die van een Adobe Flash Media-server worden gestreamd.*

## Game-invoer

Vanaf AIR 3 biedt AIR for TV ondersteuning voor ActionScript-API's die toepassingen in staat stellen te communiceren met aangesloten game-invoerapparaten, zoals joysticks, gamepads en wands. Hoewel dit apparaten voor game-invoer worden genoemd, kunnen niet alleen games, maar alle AIR for TV-toepassingen deze apparaten gebruiken.

Er zijn vele verschillende game-invoerapparaten met verschillende mogelijkheden beschikbaar. De apparaten worden daarom gegeneraliseerd in de API, zodat toepassingen goed kunnen functioneren met verschillende (en wellicht onbekende) typen game-invoerapparaten.

De klasse GameInput vormt het toegangspunt tot de ActionScript-API's voor game-invoer. Zie GameInput voor meer informatie.

## Met Stage 3D versnelde rendering van afbeeldingen

Vanaf AIR 3 biedt AIR for TV ondersteuning voor met Stage 3D versnelde rendering van afbeeldingen. De Stage3D-ActionScript-API's worden gevormd door een set laag-niveau-API's met GPU-versnelling die geavanceerde 2D- en 3D-functies mogelijk maken. Deze laag-niveau-API's geven ontwikkelaars de flexibiliteit om GPU-hardwareversnelling te benutten, hetgeen de prestaties aanzienlijk ten goede komt. U kunt ook de gamingengines gebruiken die de Stage3D-ActionScript-API's ondersteunen.

Zie Gamingengines, 3D en Stage 3D voor meer informatie.

## Native extensies

Wanneer uw toepassing bedoeld is voor het `extendedTV`-profiel, kunt u ANE-pakketten (AIR Native Extension) gebruiken.

De producent van een apparaat levert doorgaans ANE-pakketten voor toegang tot apparaatfuncties die anders niet zouden worden ondersteund door AIR. U kunt bijvoorbeeld met een native extensie kanalen wijzigen op een televisie of de weergave op een videospeler pauzeren.

Wanneer u een AIR for TV-toepassing verpakt die ANE-pakketten gebruikt, verpakt u de toepassing in een AIRN-bestand in plaats van in een AIR-bestand.

Native extensies voor AIR for TV-apparaten zijn altijd *met apparaten gebundelde* native extensies. Met apparaten gebundeld wil zeggen dat de extensiebibliotheken op het AIR for TV-apparaat zijn geïnstalleerd. Het ANE-pakket dat u in uw toepassingspakket opneemt, bevat *nooit* de native bibliotheken van de extensie. Soms bevat dit slechts een exclusieve ActionScript-versie van de native extensie. Deze exclusieve ActionScript-versie is een sectie of simulator van de extensie. De fabrikant van het apparaat installeert de ware extensie, inclusief de native bibliotheken, op het apparaat.

Bedenk het volgende wanneer u native extensies ontwikkelt:

- Raadpleeg altijd de fabrikant van apparaten waarvoor u een native extensie voor AIR for TV maakt.
- Op sommige AIR for TV-apparaten maakt alleen de fabrikant native extensies.
- Voor alle AIR for TV-apparaten geldt dat de apparaatfabrikant bepaalt welke native extensies worden geïnstalleerd.
- De ontwikkelingstools voor het samenstellen van native extensies voor AIR for TV variëren per fabrikant.

Zie "Native extensies gebruiken voor Adobe AIR" op pagina 156 voor meer informatie over het gebruik van native extensies in AIR-toepassingen.

Zie Native extensies ontwikkelen voor Adobe AIR voor informatie over het maken van native extensies.

# Aandachtspunten ontwerp AIR for TV-toepassing

## Aandachtspunten video

### Richtlijnen voor het coderen van video

Adobe raadt de volgende coderingsrichtlijnen aan voor het streamen van video naar een tv-apparaat:

| | |
|---|---|
| Videocodec: | H.264, profiel Main of High, progressief coderen |
| Resolutie: | 720i, 720p, 1080i of 1080p |
| Framesnelheid: | 24 of 30 frames per seconde |
| Audiocodec: | AAC-LC- of AC-3-, 44,1 kHz-, stereo- of de volgende multikaalsaudiocodecs: E-AC-3, DTS, DTS Express, DTS-HD High Resolution Audio en DTS-HD Master Audio |
| Gecombineerde bitsnelheid: | maximaal 8 Mbps, afhankelijk van beschikbare bandbreedte |
| Audiobitsnelheid: | maximaal 192 Kbps |
| Pixelverhouding: | 1 × 1 |

Adobe raadt u aan de H.264-codec te gebruiken voor video die wordt geleverd naar AIR for TV-apparaten.

***Opmerking:*** *AIR for TV ondersteunt ook video die is gecodeerd met Sorenson Spark- of On2 VP6-codecs. Deze codecs worden echter niet gedecodeerd en gepresenteerd door de hardware. In plaats daarvan decodeert en presenteert de runtime deze codecs met gebruik van software, waardoor de video bij een veel lagere framesnelheid wordt afgespeeld. Gebruik daarom, indien mogelijk, H.264.*

**De StageVideo-klasse**

AIR for TV ondersteunt hardwaredecodering en -presentatie van met H.264 gecodeerde video. Gebruik de StageVideo-klasse om deze functie in te schakelen.

Zie De klasse StageVideo gebruiken voor presentatie met hardwareversnelling in de *ActionScript 3.0-ontwikkelaarsgids* voor informatie over:

- de API van de StageVideo-klasse en verwante klassen.
- beperkingen op het gebruik van de StageVideo-klasse.

Teneinde bestaande AIR-toepassingen die het Video-object gebruiken voor met H.264 gecodeerde video het beste te kunnen ondersteunen, maakt AIR for TV *intern* gebruik van een StageVideo-object. Dat betekent dat voor het afspelen van video's hardwaredecodering en -presentatie wordt benut. Voor Video-objecten gelden echter dezelfde beperkingen als voor StageVideo-objecten. Als de toepassing bijvoorbeeld probeert de video te roteren, vindt er geen rotatie plaats, aangezien de video wordt gepresenteerd door de hardware, niet door de runtime.

Als u echter nieuwe toepassingen ontwikkelt, gebruikt u het StageVideo-object voor met H.264 gecodeerde video.

Zie Video en inhoud leveren voor het Flash-platform op tv voor een voorbeeld van het gebruik van de StageVideo-klasse.

**Richtlijnen voor het leveren van video**

Op AIR for TV-apparaten kan de beschikbare bandbreedte van het netwerk variëren tijdens het afspelen van video. Dit kan bijvoorbeeld gebeuren als een andere gebruiker dezelfde internetverbinding gaat gebruiken.

Daarom kan uw systeem voor het leveren van video het beste gebruikmaken van adaptieve bitsnelheidmogelijkheden. Flash Media Server ondersteunt bijvoorbeeld adaptieve bitsnelheidmogelijkheden aan de serverzijde. Aan de clientzijde kunt u het OSMF (Open Source Media Framework) gebruiken.

De volgende protocollen zijn beschikbaar voor het leveren van videoinhoud aan een AIR for TV-toepassing via het netwerk:

- Dynamische HTTP- en HTTPS-streaming (F4F-indeling)
- RTMP-, RTMPE-, RTMFP-, RTMPT- en RTMPTE-streaming
- Progressieve HTTP- en HTTPS-download

Raadpleeg de volgende bronnen voor meer informatie:

- Adobe Flash Media Server Developer's Guide
- Open Source Media Framework

## Aandachtspunten audio

De ActionScript voor het afspelen van geluid is in AIR for TV-toepassingen precies hetzelfde als in andere AIR-toepassingen. Zie Werken met geluid in de *ActionScript 3.0-ontwikkelaarsgids* voor meer informatie.

Bedenk het volgende met betrekking tot ondersteuning voor multikanaalsaudio in AIR for TV:

- AIR for TV biedt ondersteuning voor meerkanaalsaudio voor video's die progressief worden gedownload van een HTTP-server. Multikanaalsaudio wordt nog niet ondersteund in video's die van een Adobe Flash Media-server worden gestreamd.

- Hoewel AIR for TV ondersteuning biedt voor vele audiocodecs, bieden niet alle AIR for TV-*apparaten* ondersteuning voor de volledige set. Controleer met de flash.system.Capabilities-methode `hasMultiChannelAudio()` of een AIR for TV-apparaat ondersteuning biedt voor een bepaalde multikanaalsaudiocodec, zoals AC-3.

  Neem bijvoorbeeld een toepassing die een videobestand op progressieve wijze downloadt van een server. De server heeft verschillende H.264-videobestanden die ondersteuning bieden voor verschillende multikanaalsaudiocodecs. De toepassing kan met `hasMultiChannelAudio()` bepalen welk videobestand moet worden aangevraagd van de server. De toepassing kan de server ook de in `Capabilities.serverString` opgenomen tekenreeks sturen. Deze tekenreeks geeft aan welke multikanaalsaudiocodecs beschikbaar zijn, zodat de server het juiste videobestand kan selecteren.

- Bij gebruik van een van de DTS-audiocodecs bestaan scenario's waarin `true` wordt geretourneerd door `hasMultiChannelAudio()` maar geen DTS-geluid wordt afgespeeld.

  Neem bijvoorbeeld een Blu-ray-speler met een S/PDIF-uitgang, die is aangesloten op een oude versterker. DTS wordt niet ondersteund door de oude versterker, maar S/PDIF heeft geen protocol om de Blu-ray-speler te informeren. Als de Blu-ray-speler de DTS-stream naar de oude versterker verzendt, hoort de gebruiker niets. Daarom kunt u bij het gebruik van DTS het beste een gebruikersinterface verschaffen, zodat de gebruiker het kan melden als er geen geluid wordt afgespeeld. In dat geval kan uw toepassing een andere codec gebruiken.

De volgende tabel geeft aan wanneer de verschillende audiocodecs moeten worden gebruikt in AIR for TV-toepassingen. De tabel geeft ook aan wanneer AIR for TV-apparaten hardwareversnellers gebruiken voor het decoderen van een audiocodec. Hardwaredecodering verbetert de prestaties en ontlast de CPU.

| **Audiocodec** | **Beschikbaar op AIR for TV-apparaat** | **Hardwarede codering** | **Wanneer moet deze audiocodec worden gebruikt?** | **Meer informatie** |
|---|---|---|---|---|
| AAC | Altijd | Altijd | In met H.264 gecodeerde video's.<br>Voor audiostreaming, zoals een internetservice voor muziekstreaming. | Bij gebruik van een AAC-stream met alleen geluid neemt u de audiostream op in een MP4-container. |
| mp3 | Altijd | Nee | Voor geluid in de SWF-bestanden van de toepassing.<br>In video's die met Sorenson Spark of On2 VP6 zijn gecodeerd. | H.264-video's die mp3 gebruiken voor audio worden niet afgespeeld op AIR for TV-apparaten. |

| Audiocodec | Beschikbaar op AIR for TV-apparaat | Hardwarede codering | Wanneer moet deze audiocodec worden gebruikt? | Meer informatie |
|---|---|---|---|---|
| AC-3 (Dolby Digital)<br>E-AC-3 (Enhanced Dolby Digital)<br>DTS Digital Surround<br>DTS Express<br>DTS-HD High Resolution Audio<br>DTS-HD Master Audio | Controleren | Ja | In met H.264 gecodeerde video's. | AIR for TV geeft doorgaans een multikanaalsaudiostream door aan een externe audio/video-ontvanger die de audio decodeert en afspeelt. |
| Speex | Altijd | Nee | Bij ontvangst van een live stemstream. | H.264-video's die Speex gebruiken voor audio worden niet afgespeeld op AIR for TV-apparaten. Gebruik Speex alleen met video's die zijn gecodeerd met Sorenson Spark of On2 VP6. |
| Nellymoser | Altijd | Nee | Bij ontvangst van een live stemstream. | H.264-video's die NellyMoser gebruiken voor audio worden niet afgespeeld op AIR for TV-apparaten. Gebruik NellyMoser alleen met video's die zijn gecodeerd met Sorenson Spark of On2 VP6. |

***Opmerking:*** *sommige videobestanden bevatten twee audiostreams. Een videobestand kan bijvoorbeeld zowel een AAC- als een AC3-stream bevatten. AIR for TV biedt geen ondersteuning voor dergelijke videobestanden. Het gebruik van deze bestanden kan ertoe leiden dat de video zonder geluid wordt afgespeeld.*

## Grafische hardwareversnelling

### Grafische hardwareversnelling gebruiken

AIR for TV-apparaten verschaffen hardwareversnelling voor bewerkingen van 2D-afbeeldingen. De grafische versnellers van de hardware van het apparaat ontlasten de CPU door de volgende bewerkingen uit te voeren:

- Renderen van bitmaps
- Schalen van bitmaps
- Overvloeien van bitmaps
- Rechthoeken vullen met effen kleuren

Dankzij deze grafische hardwareversnelling kunnen vele grafische bewerkingen in een AIR for TV-toepassing snel worden uitgevoerd. Het gaat hierbij bijvoorbeeld om de volgende bewerkingen:

- Schuivende overgangen
- Overgangen met schaal
- In- en uitfaden
- Meerdere afbeeldingen samenstellen met alfa

Gebruik een van de volgende technieken om te profiteren van grafische hardwareversnelling voor dergelijke bewerkingen:

- Stel de eigenschap `cacheAsBitmap` in op `true` voor MovieClip-objecten en andere weergaveobjecten waarvan de inhoud vrijwel ongewijzigd blijft. Voer vervolgens schuivende overgangen, in- of uitfadende overgangen en alfaovervloeiingen uit op deze objecten.
- Gebruik de eigenschap `cacheAsBitmapMatrix` voor weergaveobjecten die u wilt schalen of omzetten (x- en y-repositionering toepassen).

  Door Matrix-klassebewerkingen te gebruiken voor schalen en omzetten, voeren de hardwareversnellers van het apparaat de bewerkingen uit. Aan de andere kant zou u de afmetingen van een object waarvoor de eigenschap `cacheAsBitmap` is ingesteld op `true` kunnen wijzigen. Wanneer de afmetingen worden gewijzigd, tekent de software van de runtime de bitmap opnieuw. Opnieuw tekenen met software levert minder goede resultaten op dan schalen met hardwareversnelling aan de hand van een Matrix-bewerking.

  Neem bijvoorbeeld een toepassing die een afbeelding weergeeft die groter wordt wanneer een gebruiker deze selecteert. Gebruik de Matrix-schaalbewerking dan meerdere malen om de illusie te wekken dat de afbeelding groter wordt. Afhankelijk van het formaat van de originele afbeelding en de uiteindelijke afbeelding, is de kwaliteit van de uiteindelijke afbeelding mogelijk onacceptabel. Herstel daarom de afmetingen van het weergaveobject nadat de vergrootbewerkingen zijn voltooid. Aangezien `cacheAsBitmap` is ingesteld op `true`, tekent de runtimesoftware het weergaveobject opnieuw, maar slechts één keer, en wordt een afbeelding van hoge kwaliteit gerenderd.

  ***Opmerking:*** *AIR for TV-apparaten ondersteunen door hardware versneld roteren en schuintrekken doorgaans niet. Als u dus roteren en schuintrekken opgeeft in de Matrix-klasse, voert AIR for TV alle Matrix-bewerkingen uit in de software. En deze softwarebewerkingen hebben een nadelige invloed op de prestaties.*
- Gebruik de BitmapData-klasse om het in cache plaatsen van bitmaps aan te passen.

Zie het volgende voor meer informatie over bitmaps in cache plaatsen:

- Weergaveobjecten in cache plaatsen
- Bitmaps in cache plaatsen
- Bitmaps handmatig in cache plaatsen

### Het grafisch geheugen beheren

Hardwareversnellers gebruiken een speciaal grafisch geheugen om versnelde grafische bewerkingen uit te voeren. Als uw toepassing het grafische geheugen volledig in beslag neemt, gaat de toepassing trager functioneren, omdat AIR for TV dan software moet gebruiken voor de grafische bewerkingen.

Het gebruik van het grafische geheugen door uw toepassing beheren:

- Als u klaar bent met het gebruik van een afbeelding of andere bitmapgegevens, geeft u het desbetreffende grafische geheugen vrij. Dat doet u door de methode `dispose()` van de eigenschap `bitmapData` van het bitmapobject aan te roepen. Bijvoorbeeld:

  ```
  myBitmap.bitmapData.dispose();
  ```

  ***Opmerking:*** *wanneer u de verwijzing naar het BitmapData-object vrijgeeft, is het grafische geheugen niet meteen beschikbaar. De opschoonfunctie van de runtime geeft het grafische geheugen na verloop van tijd vrij, maar u geeft uw toepassing meer controle door* `dispose()` *aan te roepen.*

- Gebruik PerfMaster Deluxe, een door Adobe verschafte AIR-toepassing, om meer inzicht te krijgen in het de grafische hardwareversnelling op uw doelapparaat. Deze toepassing toont het aantal frames per seconde voor het uitvoeren van verschillende bewerkingen. Met PerfMaster Deluxe kunt u verschillende implementaties van dezelfde bewerking vergelijken. U kunt bijvoorbeeld het verplaatsen van een bitmapafbeelding vergelijken met het verplaatsen van een vectorafbeelding. PerfMaster Deluxe is beschikbaar op Het Flash-platform voor TV.

**Het weergaveoverzicht beheren**

Als u een weergegeven object onzichtbaar wilt maken, moet u de eigenschap `visible` van het object instellen op `false`. Het object blijft op het weergaveoverzicht staan, maar het wordt niet gerenderd of weergegeven door AIR for TV. Deze techniek is handig voor objecten die voortdurend in beeld komen en uit beeld verdwijnen, omdat er slechts weinig verwerking is vereist. Wanneer u de eigenschap `visible` echter instelt op `false`, worden geen van de objectresources vrijgemaakt. Als u een object langdurig (of permanent) niet meer wilt weergeven, kunt u het object daarom het beste verwijderen van het weergaveoverzicht. Bovendien moet u alle referenties naar het object instellen op `null`. Hierdoor kunnen de resources van het object bij het opschonen worden vrijgegeven.

## Het gebruik van PNG- en JPEG-afbeeldingen

PNG en JPEG zijn twee afbeeldingsindelingen die veel worden gebruikt in toepassingen. Overweeg het volgende met betrekking tot deze afbeeldingsindelingen in AIR for TV-toepassingen:

- AIR for TV benut doorgaans hardwareversnelling voor het decoderen van JPEG-bestanden.
- AIR for TV benut doorgaans software voor het decoderen van PNG-bestanden. PNG-bestanden worden snel gedecodeerd door software.
- PNG is de enige bitmapindeling voor verschillende platformen die transparantie (een alfakanaal) ondersteunt.

Gebruik deze afbeeldingsindelingen daarom als volgt in uw toepassingen:

- Gebruik JPEG-bestanden voor foto's, zodat u van door hardware versnelde decodering kunt profiteren.
- Gebruik PNG-afbeeldingsbestanden voor interface-elementen. De elementen van de gebruikersinterface kunnen een alfa-instelling hebben en softwaredecodering is snel genoeg voor deze elementen.

## Het werkgebied in AIR for TV-toepassingen

Wanneer u een AIR for TV-toepassing ontwikkelt, dient u het volgende te overwegen wanneer u met de Stage-klasse werkt:

- Schermresolutie
- Het veilige gebied voor weergave
- De modus voor schaling van het werkgebied
- De uitlijning van het werkgebied
- De weergavestatus van het werkgebied
- Ontwerpen voor meerdere schermformaten
- De kwaliteitsinstelling voor het werkgebied

**Schermresolutie**

De meeste tv's hebben tegenwoordig een van de volgende schermresoluties: 540p, 720p of 1080p. Deze schermresoluties resulteren in de volgende waarden in de ActionScript Capabilities-klasse:

| **Schermresolutie** | `Capabilities.screenResolutionX` | `Capabilities.screenResolutionY` |
|---|---|---|
| 540p | 960 | 540 |
| 720p | 1280 | 720 |
| 1080p | 1920 | 1080 |

Als u een AIR for TV-toepassing op volledig scherm ontwikkelt voor een specifiek apparaat, stelt u de waarden `Stage.stageWidth` en `Stage.stageHeight` in voor de schermresolutie van het apparaat. Als u echter een volledige-schermtoepassing ontwikkelt die kan worden uitgevoerd op meerdere apparaten, stelt u de afmetingen van het werkgebied in met de eigenschappen `Capabilities.screenResolutionX` en `Capabilities.screenResolutionY`.

Bijvoorbeeld:

```
stage.stageWidth = Capabilities.screenResolutionX;
stage.stageHeight = Capabilities.screenResolutionY;
```

### Het veilige gebied voor weergave

Het *veilige gebied voor weergave* op een televisie is het gedeelte van het scherm dat is verschoven ten opzichte van de rand van het scherm. Dit gebied ligt zo ver bij de rand vandaan dat de gebruiker het volledige gebied kan zien, zonder dat de schuine rand van de TV een gedeelte verhult. Aangezien de door het fysieke frame rond het scherm gevormde schuine rand varieert per fabrikant, kan de benodigde inzet ook variëren. Het veilige gebied voor weergave probeert een gebied op het scherm te garanderen dat altijd zichtbaar is. Dit veilige gebied wordt ook wel het *veilige gebied voor titels* genoemd.

*Overscan* verwijst naar het gedeelte van het scherm dat niet zichtbaar is, omdat het zich achter de schuine rand bevindt.

Adobe raadt een inzet van 7,5% aan voor elke rand van het scherm. Bijvoorbeeld:


*Veilig gebied voor weergave voor een schermresolutie van 1920 x 1080*

Denk altijd aan het veilige gebied voor weergave wanneer u een AIR for TV-toepassing voor een volledig scherm ontwikkelt:

- Gebruik het volledige scherm voor achtergronden, zoals achtergrondafbeeldingen of -kleuren.

- Gebruik alleen het veilige gebied voor weergave voor belangrijke toepassingselementen, zoals tekst, afbeeldingen, video en interface-elementen, zoals knoppen.

De volgende tabel toont de afmetingen van het veilige gebied voor weergave voor alle standaardschermresoluties, bij gebruik van een inzet van 7,5%.

| Schermresolutie | Breedte en hoogte van veilig gebied voor weergave | Breedte inzet aan linker- en rechterkant | Hoogte inzet aan boven- en onderkant |
|---|---|---|---|
| 960 x 540 | 816 x 460 | 72 | 40 |
| 1280 x 720 | 1088 x 612 | 96 | 54 |
| 1920 x 1080 | 1632 x 918 | 144 | 81 |

Het is echter altijd het beste het veilige gebied voor weergave op dynamische wijze te berekenen. Bijvoorbeeld:

```
var horizontalInset, verticalInset, safeAreaWidth, safeAreaHeight:int;

horizontalInset = .075 * Capabilities.screenResolutionX;
verticalInset = .075 * Capabilities.screenResolutionY;
safeAreaWidth = Capabilities.screenResolutionX - (2 * horizontalInset);
safeAreaHeight = Capabilities.screenResolutionY - (2 * verticalInset);
```

### Modus voor schalen van het werkgebied

Stel `Stage.scaleMode` in op `StageScaleMode.NO_SCALE` en luister naar gebeurtenissen voor het vergroten of verkleinen van het werkgebied.

```
stage.scaleMode = StageScaleMode.NO_SCALE;
stage.addEventListener(Event.RESIZE, layoutHandler);
```

Met deze instelling worden werkgebiedcoördinaten gelijk aan pixelcoördinaten. Deze instelling zorgt er samen met de weergavestatus `FULL_SCREEN_INTERACTIVE` en de werkgebieduitlijning `TOP_LEFT` voor dat u het veilige gebied voor weergave op effectieve wijze kunt gebruiken.

In het bijzonder betekent deze schaalmodus voor een volledig scherm ook dat de eigenschappen `stageWidth` en `stageHeight` van de Stage-klasse overeenkomen met de eigenschappen `screenResolutionX` en `screenResolutionY` van de Capabilities-klasse.

Bovendien behoudt de inhoud van het werkgebied de gedefinieerde grootte wanneer het toepassingsvenster wordt vergroot of verkleind. De runtime voert geen automatische schaling of lay-out uit. De runtime verzendt bovendien de gebeurtenis `resize` van de Stage-klasse wanneer het vensterformaat verandert. U hebt dus volledige controle over de manier waarop de inhoud van de toepassing wordt aangepast wanneer de toepassing wordt gestart en wanneer het formaat van het toepassingsvenster wordt veranderd.

***Opmerking:*** *het gedrag van `NO_SCALE` is precies hetzelfde als in alle andere AIR-toepassingen. In AIR for TV-toepassingen is het gebruik van deze instelling echter van essentieel belang voor het gebruik van het veilige gebied voor weergave.*

### Uitlijning van het werkgebied

Stel `Stage.align` in op `StageAlign.TOP_LEFT`:

```
stage.align = StageAlign.TOP_LEFT;
```

Deze uitlijning plaatst de coördinaat `0,0` linksboven in het scherm, wat handig is voor het plaatsen van inhoud met ActionScript.

Samen met de schaalmodus `NO_SCALE` en de weergavestatus `FULL_SCREEN_INTERACTIVE` stelt deze instelling u in staat effectief gebruik te maken van het veilige gebied voor weergave.

**Weergavestatus van het werkgebied**

Stel `Stage.displayState` in AIR for TV-toepassingen op volledig scherm in op `StageDisplayState.FULL_SCREEN_INTERACTIVE`:

```
stage.displayState = StageDisplayState.FULL_SCREEN_INTERACTIVE;
```

Met deze waarde wordt de AIR-toepassing zo ingesteld dat het werkgebied op het volledige scherm wordt weergegeven, waarbij gebruikersinvoer is toegestaan.

U wordt aangeraden de instelling `FULL_SCREEN_INTERACTIVE` te gebruiken. Samen met de schaalmodus `NO_SCALE` en de werkgebieduitlijning `TOP_LEFT` stelt deze instelling u in staat effectief gebruik te maken van het veilige gebied voor weergave.

Ga daarom als volgt te werk in een handler voor de `ADDED_TO_STAGE`-gebeurtenis van de hoofddocumentklasse voor toepassingen met een volledig scherm:

```
private function onStage(evt:Event):void
{
    stage.scaleMode = StageScaleMode.NO_SCALE;
    stage.align = StageAlign.TOP_LEFT;
    stage.addEventListener(Event.RESIZE, onResize);
    stage.displayState = StageDisplayState.FULL_SCREEN_INTERACTIVE;
}
```

Vervolgens doet u het volgende in de handler voor de `RESIZE`-gebeurtenis:

- Vergelijk de schermresolutie met de breedte en hoogte van het werkgebied. Als deze gelijk zijn, heeft de gebeurtenis `RESIZE` plaatsgevonden, omdat de weergavestatus van het werkgebied is veranderd in `FULL_SCREEN_INTERACTIVE`.
- Bereken de afmetingen van het veilige gebied voor weergave en de bijbehorende inzet en sla deze op.

```
private function onResize(evt:Event):void
{
    if ((Capabilities.screenResolutionX == stage.stageWidth) &&
        (Capabilities.screenResolutionY == stage.stageHeight))
    {

        // Calculate and save safe viewing area dimensions.
    }
}
```

Wanneer de afmetingen van de stage gelijk zijn aan `Capabilities.screenResolutionX` en `screenResolutionY`, zorgt AIR for TV voor dat uw video en afbeeldingen optimaal worden weergeven door de hardware.

***Opmerking:*** *de weergavekwaliteit van uw afbeeldingen en video-opnamen op de tv is niet altijd gelijk aan de waarden van `Capabilities.screenResolutionX` en `screenResolutionY`. Deze waarden zijn namelijk afhankelijk van het apparaat waarop AIR for TV wordt uitgevoerd. Zo kan de schermresolutie van een set-top box met AIR for TV bijvoorbeeld 1280 x 720 zijn, terwijl de aangesloten tv een schermresolutie van 1920 x 1080 heeft. Desondanks zorgt AIR for TV voor een optimale weergavekwaliteit. In dit voorbeeld wordt daarom een 1080p-video weergegeven met een schermresolutie van 1920 x 1080.*

### Ontwerpen voor meerdere schermformaten

U kunt één AIR for TV-toepassing voor een volledig scherm ontwerpen die naar behoren functioneert en er goed uitziet op verschillende AIR for TV-apparaten. Ga als volgt te werk:

1. Stel de werkgebiedeigenschappen `scaleMode`, `align` en `displayState` respectievelijk in op de aanbevolen waarden `StageScaleMode.NO_SCALE`, `StageAlign.TOP_LEFT` en `StageDisplayState.FULL_SCREEN_INTERACTIVE`.
2. Stel het veilige gebied voor weergave in op basis van `Capabilities.screenResolutionX` en `Capabilities.screenResolutionY`.
3. Pas de grootte en lay-out van uw inhoud aan op basis van de breedte en hoogte van het veilige gebied voor weergave.

   Hoewel de inhoudobjecten groot zijn, vooral in vergelijking met toepassingen voor mobiele apparaten, zijn concepten als dynamische lay-out, relatieve positionering en adaptieve inhoud precies hetzelfde. Zie Mobiele Flash-inhoud ontwerpen voor schermen van verschillend formaat voor meer informatie over de ondersteuning van ActionScript voor deze concepten.

### De kwaliteit van het werkgebied

De eigenschap `Stage.quality` voor een AIR for TV-toepassing is altijd ingesteld op `StageQuality.High`. U kunt deze instelling niet wijzigen.

Deze eigenschap bepaalt de renderkwaliteit van alle werkgebiedobjecten.

## Invoer van een afstandsbediening verwerken

Gebruikers communiceren doorgaans via een afstandsbediening met uw AIR for TV-toepassing. Verwerk toetsinvoer echter op precies dezelfde manier als toetsinvoer via een toetsenbord van een bureaubladtoepassing. Verwerk in het bijzonder de gebeurtenis `KeyboardEvent.KEY_DOWN`. Zie Toetsenbordinvoer vastleggen in de *ActionScript 3.0-ontwikkelaarsgids* voor meer informatie.

De toetsen op de afstandsbediening zijn toegewezen aan ActionScript-constanten. De toetsen op het blok met richtingtoetsen op een afstandsbediening zijn bijvoorbeeld als volgt toegewezen:

| **Toets op richtingtoetsenblok afstandsbediening** | **ActionScript 3.0-constante** |
|---|---|
| Omhoog | `Keyboard.UP` |
| Omlaag | `Keyboard.DOWN` |
| Links | `Keyboard.LEFT` |
| Rechts | `Keyboard.RIGHT` |
| OK of selecteren | `Keyboard.ENTER` |

AIR 2.5 heeft vele andere toetsenbordconstanten toegevoegd ter ondersteuning van invoer via een afstandsbediening. Zie Keyboard-klasse in de *Naslaggids voor ActionScript 3.0 voor het Adobe Flash-platform* voor een volledige lijst.

Houd u aan de volgende aanbevelingen, zodat uw toepassing op zoveel mogelijk verschillende apparaten werkt:

- Gebruik, waar mogelijk, alleen de toetsen op het blok met richtingtoetsen.

  Verschillende afstandsbedieningen hebben verschillende toetsen. Maar de meeste afstandsbedieningen hebben een blok met richtingtoetsen.

  Een afstandsbediening voor een Blu-ray-speler beschikt doorgaans bijvoorbeeld niet over een toets om naar de volgende of vorige zender te gaan. En ook niet elke afstandsbediening heeft een knop voor afspelen, pauzeren en stoppen.

- Gebruik de toetsen Menu en Info als de toepassing naast de richtingtoetsen nog andere toetsen nodig heeft.

  De toetsen Menu en Info zijn op de richtingtoetsen na de meest voorkomende toetsen op afstandsbedieningen.

- Overweeg hoe vaak universele afstandsbedieningen worden gebruikt.

  Zelfs als u een toepassing voor een bepaald apparaat ontwikkelt, dient u erbij stil te staan dat veel gebruikers niet de bij het apparaat geleverde afstandsbediening, maar een universele afstandsbediening gebruiken. Gebruikers programmeren hun universele afstandsbediening bovendien niet altijd zodanig dat er een overeenkomst is voor alle toetsen op de afstandsbediening van het apparaat. Het is daarom beter alleen de meest gebruikelijke toetsen te gebruiken.

- Zorg ervoor dat de gebruiker een situatie altijd kan sluiten met een van de richtingtoetsen.

  Soms kan er een dringende reden zijn waarom uw toepassing een andere toets dan de meest gebruikelijke toetsen op een afstandsbediening moet gebruiken. De mogelijkheid een situatie te kunnen sluiten met een van de richtingtoetsen maakt uw toepassing gebruiksvriendelijker op alle mogelijke apparaten.

- Stel aanwijzerinvoer niet verplicht, tenzij u weet dat het AIR for TV-doelapparaat over functionaliteit voor aanwijzerinvoer beschikt.

  Veel bureaubladtoepassingen verwachten muisinvoer, maar de meeste tv's ondersteunen dergelijke invoer niet. Als u dus bureaubladtoepassingen aanpast voor gebruik op een televisie, dient u muisinvoer uit te schakelen. Deze aanpassingen betreffen onder andere het afhandelen van gebeurtenissen en wijzigingen in instructies voor de gebruiker. Zorg er bijvoorbeeld voor dat het opstartscherm van een toepassing niet de tekst "Klik hier" bevat.

## Focus beheren

Wanneer een interface-element in een bureaubladtoepassing de focus krijgt, is het het doel van gebruikersinvoergebeurtenissen, zoals toetsenbord- en muisgebeurtenissen. Bovendien markeert een toepassing het interface-element dat de focus heeft. Het beheren van de focus in een AIR for TV-toepassing is niet hetzelfde als het beheren van de focus in een bureaubladtoepassing omdat:

- Bureaubladtoepassingen vaak de Tab-toets gebruiken om de focus naar het volgende interface-element te verplaatsen. Het gebruik van de Tab-toets is niet van toepassing op AIR for TV-toepassingen. Afstandsbedieningen hebben doorgaans geen Tab-toets. Het beheren van de focus aan de hand van de eigenschap `tabEnabled` van een DisplayObject zoals op het bureaublad functioneert dus niet.
- Bureaubladtoepassingen vaak verwachten dat de gebruiker de muis gebruikt om de focus op een interface-element te plaatsen.

Ga daarom in uw toepassing als volgt te werk:

- Voeg een gebeurtenislistener toe aan het werkgebied die luistert naar toetsenbordgebeurtenissen zoals `KeyboardEvent.KEY_DOWN`.
- Voorzie in toepassingslogica om te bepalen welk interface-element wordt gemarkeerd voor de eindgebruiker. Zorg ook dat er een interface-element is gemarkeerd wanneer de toepassing wordt gestart.

- Verzend op basis van uw toepassingslogica de Keyboard-gebeurtenis die is ontvangen door het werkgebied naar het desbetreffende interface-elementobject.

  U kunt ook `Stage.focus` of `Stage.assignFocus()` gebruiken om de focus toe te wijzen aan een gebruikersinterface-element. U kunt vervolgens een gebeurtenislistener toevoegen aan dat DisplayObject, zodat dit toetsenbordgebeurtenissen ontvangt.

## Ontwerp gebruikersinterface

Zorg voor een gebruiksvriendelijke gebruikersinterface van AIR for TV-toepassingen op televisies door u te houden aan de volgende aanbevelingen op gebied van:

- het reactievermogen van de toepassing
- de bruikbaarheid van de toepassing
- de persoonlijkheid en verwachtingen van gebruiker

### Reactievermogen

Benut de volgende tips om een AIR for TV-toepassing zo goed mogelijk te laten reageren.

- Maak het aanvankelijke SWF-bestand van de toepassing zo klein mogelijk.

  Laad in het aanvankelijke SWF-bestand alleen de bronnen die nodig zijn om de toepassing te starten. Laad bijvoorbeeld alleen de afbeelding voor het opstartscherm van de toepassing.

  Deze aanbeveling geldt ook voor AIR-bureaubladtoepassingen, maar is van groter belang voor AIR for TV-apparaten. AIR for TV-apparaten beschikken bijvoorbeeld niet over hetzelfde verwerkingsvermogen als bureaubladcomputers. Bovendien slaan deze apparaten de toepassing op in het Flash-geheugen, wat minder snel te benaderen is dan vaste schijven op bureaubladcomputers.

- Zorg dat de toepassing bij een framesnelheid van minstens 20 frames per seconde wordt uitgevoerd.

  Ontwerp uw afbeeldingen met dit idee in uw achterhoofd. De complexiteit van grafische bewerkingen kan het aantal frames per seconde beïnvloeden. Zie De prestaties voor het Adobe Flash-platform optimaliseren voor tips over het verbeteren van de renderprestaties.

  ***Opmerking:*** *de grafische hardware op AIR for TV-apparaten werkt het scherm doorgaans bij met een snelheid van 60 of 120 Hz (60 of 120 keer per seconde). De hardware scant het werkgebied bijvoorbeeld op updates met een snelheid van 30 of 60 frames per seconde voor weergave op een 60 of 120 Hz-scherm. Het hangt echter af van de complexiteit van de grafische bewerkingen van de toepassing of de gebruiker deze hogere framesnelheden ook ervaart.*

- Werk het scherm bij binnen 100 tot 200 milliseconden van de gebruikersinvoer.

  Gebruikers worden ongeduldig als het langer duurt en gaan dan vaak meer toetsen indrukken.

### Bruikbaarheid

Gebruikers van AIR for TV-toepassingen bevinden zich vaak in een woonkamer en zitten meestal op zo'n drie meter afstand van de tv. Soms is het ook nog donker in de kamer. Meestal gebruiken ze een afstandsbediening voor invoer. Het is mogelijk dat meerdere personen de toepassing gebruiken, soms tegelijk, soms na elkaar.

Ontwerp uw gebruikersinterface daarom met oog op de volgende tv-bruikbaarheidsoverwegingen:

- Ontwerp grote elementen voor de gebruikersinterface.

  Bedenk bij het ontwerpen van tekst, knoppen of andere interface-elementen dat de gebruiker aan de andere kant van de kamer zit. Zorg er dus voor dat alles gemakkelijk zicht- en leesbaar is op een afstand van bijvoorbeeld drie meter. Ook al is het scherm groot, zorg dat het niet te vol raakt.

- Gebruik een goed contrast, zodat de inhoud ook vanaf de andere kant van de kamer duidelijk zicht- en leesbaar is.
- Zorg dat het interface-element dat de focus heeft in het oog springt door het een felle kleur te geven.
- Gebruik beweging alleen als het nodig is. Het van het ene scherm naar het andere schuiven om verder te gaan, kan bijvoorbeeld goed werken. Beweging kan echter ook afleiden als het de gebruiker niet helpt bij het navigeren of als het niets bijdraagt aan de toepassing.
- Zorg altijd dat de gebruiker op een gemakkelijke manier terug kan navigeren door de gebruikersinterface.

Zie "Invoer van een afstandsbediening verwerken" op pagina 138 voor meer informatie over het gebruik van de afstandsbediening.

**De persoonlijkheid en verwachtingen van gebruikers**

Bedenk dat gebruikers van AIR for TV-toepassingen doorgaans entertainment van tv-kwaliteit verwachten in een leuke, ontspannen omgeving. Ze zijn lang niet altijd deskundig op gebied van computers of technologie.

Ontwerp daarom AIR for TV-toepassingen met de volgende kenmerken:

- Gebruik geen technische termen.
- Gebruik geen modale dialoogvensters.
- Gebruik vriendelijke, informele instructies die geschikt zijn voor een woonkameromgeving, niet voor een professionele of technische omgeving.
- Gebruik afbeeldingen van de hoge kwaliteit die tv-kijkers verwachten.
- Maak een gebruikersinterface die geschikt is voor een afstandsbediening. Gebruik geen gebruikersinterface- of ontwerpelementen die beter bij toepassingen voor mobiele apparatuur of desktops passen. Zo bevatten gebruikersinterfaces op desktops en mobiele apparatuur vaak knoppen voor aanwijzen en klikken met een muis of vinger.

## Lettertypen en tekst

U kunt de apparaatlettertypen gebruiken of de ingesloten lettertypen van uw AIR for TV-toepassing.

Apparaatlettertypen zijn de op een apparaat geïnstalleerde lettertypen. Alle AIR for TV-apparaten beschikken over de volgende apparaatlettertypen:

| **Lettertypenaam** | **Beschrijving** |
|---|---|
| `_sans` | Het apparaatlettertype `_sans` is een sans-serif-lettertype. Het `_sans`-apparaatlettertype Myriad Pro is geïnstalleerd op alle AIR for TV-apparaten. Vanwege de kijkafstand zien sans-serif-lettertypen er op televisies meestal beter uit dan serif-lettertypen. |
| `_serif` | Het apparaatlettertype `_sans` is een serif-lettertype. Minion Pro is het `_serif` -apparaatlettertype dat is geïnstalleerd op alle AIR for TV-apparaten. |
| `_typewriter` | Het apparaatlettertype `_typewriter` is een lettertype met vaste spatiëring. Courier Std. is het `_typewriter`-apparaatlettertype dat is geïnstalleerd op alle AIR for TV-apparaten. |

Alle AIR for TV-apparaten beschikken ook over de volgende Aziatische apparaatlettertypen:

| Lettertypenaam | Taal | Lettertypecategorie | lokale code |
|---|---|---|---|
| RyoGothicPlusN-Regular | Japans | sans | ja |
| RyoTextPlusN-Regular | Japans | serif | ja |
| AdobeGothicStd-Light | Koreaans | sans | ko |
| AdobeHeitiStd-Regular | Vereenvoudigd Chinees | sans | zh_CN |
| AdobeSongStd-Light | Vereenvoudigd Chinees | serif | zh_CN |
| AdobeMingStd-Light | Traditioneel Chinees | serif | zh_TW en zh_HK |

Deze AIR for TV-apparaatlettertypen:

- Zijn afkomstig uit de Adobe® Type-bibliotheek
- Zien er goed uit op een tv-scherm
- Zijn ontworpen voor videotitels
- Zijn lettertypecontouren, geen bitmaplettertypen

***Opmerking:*** *apparaatproducenten installeren vaak andere apparaatlettertypen op het apparaat. Deze door de producent verschafte apparaatlettertypen worden in aanvulling op de AIR for TV-apparaatlettertypen geïnstalleerd.*

Adobe levert de toepassing FontMaster Deluxe die alle apparaatlettertypen op het apparaat weergeeft. De toepassing is verkrijgbaar op Flash Platform voor tv.

U kunt lettertypen ook insluiten in uw AIR for TV-toepassing. Zie Geavanceerde tekstrendering in de *ActionScript 3.0-ontwikkelaarsgids*.

Adobe raadt het volgende aan met betrekking tot het gebruik van TLF-tekstvelden:

- Gebruik TLF-tekstvelden voor Aziatische tekst om de landinstellingen waarin de toepassing wordt uitgevoerd te benutten. Stel de eigenschap `locale` van het aan het TLFTextField-object gekoppelde TextLayoutFormat-object in. In Landinstellingen kiezen in de *ActionScript 3.0-ontwikkelaarsgids* wordt uitgelegd hoe u de huidige landinstellingen kunt bepalen.
- Geef de naam van het lettertype op in de eigenschap `fontFamily` van het TextLayoutFormat-object als het lettertype niet een van de AIR for TV-apparaatlettertypen is. AIR for TV gebruikt dat lettertype als het beschikbaar is op het apparaat. Als het gewenste lettertype niet op het apparaat aanwezig is, gebruikt AIR for TV in plaats daarvan het overeenkomende AIR for TV-apparaatlettertype op basis van de `locale`-instelling.
- Geef `_sans`, `_serif` of `_typewriter` op voor de eigenschap `fontFamily` en stel de eigenschap `locale` in, zodat AIR for TV het juiste AIR for TV-apparaatlettertype kan kiezen. AIR for TV maakt afhankelijk van de landinstellingen een keuze uit de set Aziatische apparaatlettertypen of uit de set niet-Aziatische apparaatlettertypen. Dankzij deze instellingen wordt op eenvoudige wijze automatisch het juiste lettertype gebruikt voor Engels en de vier belangrijkste Aziatische talen.

***Opmerking:*** *als u klassieke tekstvelden gebruikt voor Aziatische tekst, geeft u de naam op van een AIR for TV-apparaatlettertype om zeker te zijn van correcte rendering. U kunt ook een ander lettertype opgeven als u zeker weet dat dit op het doelapparaat is geïnstalleerd.*

Bedenk het volgende met betrekking tot de prestaties van de toepassing:

- Klassieke tekstvelden leveren snellere prestaties op dan TLF-tekstvelden.

- Een klassiek tekstveld dat gebruikmaakt van bitmaplettertypen resulteert in de snelste prestaties.

  Bitmaplettertypen verschaffen een bitmap voor elk teken, in tegenstelling tot contourlettertypen die slechts de contourgegevens voor elk teken verschaffen. Zowel apparaatlettertypen als ingesloten lettertypen kunnen bitmaplettertypen zijn.

- Als u een apparaatlettertype opgeeft, dient u ervoor te zorgen dat dit lettertype is geïnstalleerd op uw doelapparaat. Als dat niet het geval is, zoekt en gebruikt AIR for TV een ander lettertype dat op het apparaat is geïnstalleerd. Dit vertraagt de toepassing echter.

- Als een TextField-object veelal ongewijzigd blijft, stelt u de eigenschap `cacheAsBitmap` van dit object, net als voor alle andere weergaveobjecten, in op `true`. Zo verbetert u de prestaties van overgangen met in- of uitfaden, schuiven en alfa-overvloeien. Gebruik `cacheAsBitmapMatrix` voor schalen en omzetten. Zie "Grafische hardwareversnelling" op pagina 132 voor meer informatie.

## Beveiliging bestandssysteem

AIR for TV-toepassingen zijn AIR-toepassingen en hebben dus toegang tot het bestandssysteem van het apparaat. Op een 'huiskamerapparaat' is het echter van essentieel belang dat een toepassing geen toegang heeft tot de systeembestanden van het apparaat of tot de bestanden van andere toepassingen. Gebruikers van tv's en verwante apparatuur verwachten en tolereren geen apparaatfouten, ze kijken immers gewoon tv.

Daarom heeft een AIR for TV-toepassing slechts beperkt inzicht in het bestandssysteem van het apparaat. Uw toepassing heeft via ActionScript 3.0 slechts toegang tot specifieke mappen (en de bijbehorende submappen). Bovendien komen de mapnamen die u in ActionScript gebruikt niet overeen met de feitelijke mapnamen op het apparaat. Deze extra laag voorkomt dat AIR for TV-toepassingen per ongeluk of met kwade bedoelingen lokale bestanden openen die niet van hen zijn.

Zie Mapweergave voor AIR for TV-toepassingen voor meer informatie.

## De sandbox van AIR-toepassingen

AIR for TV-toepassingen worden uitgevoerd in de AIR-toepassingssandbox, zoals wordt uitgelegd in De AIR-toepassingssandbox.

Hen enige verschil is dat AIR for TV-toepassingen slechts beperkte toegang hebben tot het bestandssysteem, zoals beschreven in "Beveiliging bestandssysteem" op pagina 143.

## Gebruiksduur van de toepassing

In tegenstelling tot een bureaubladomgeving kan de eindgebruiker het venster waarin uw AIR for TV-toepassing wordt uitgevoerd niet sluiten. Daarom dient u de gebruiker een interfacemechanisme te verschaffen waarmee hij of zij de toepassing kan sluiten.

Op de meeste apparaten kunnen eindgebruikers een toepassing onvoorwaardelijk sluiten met de afsluittoets op de afstandsbediening. AIR for TV verzendt de gebeurtenis `flash.events.Event.EXITING` echter niet naar de toepassing. Sla de toepassingsstatus daarom regelmatig op, zodat automatisch een redelijke status van de toepassing wordt hersteld wanneer deze opnieuw wordt gestart.

## HTTP-cookies

AIR for TV biedt ondersteuning voor permanente HTTP-cookies en sessiecookies. AIR for TV slaat de cookies van elke AIR-toepassing op in een toepassingsspecifieke map:

`/app-storage/<app id>/Local Store`

Het cookiebestand heeft de naam `cookies`.

***Opmerking:*** *op andere apparaten, zoals bureaubladapparatuur, slaat AIR cookies niet afzonderlijk voor elke toepassing op. Toepassingsspecifieke opslag van cookies ondersteunt het toepassings- en systeembeveiligingsmodel van AIR for TV.*

Gebruik de ActionScript-eigenschap `URLRequest.manageCookies` als volgt:

- Stel `manageCookies` in op `true`. Dit is de standaardinstelling, waarbij AIR for TV automatisch cookies toevoegt aan HTTP-aanvragen en de cookies in de HTTP-respons onthoudt.

  ***Opmerking:*** *ook als `manageCookiestrue` is, kan de toepassing met gebruik van `URLRequest.requestHeaders` handmatig een cookie toevoegen aan een HTTP-aanvraag. Als deze cookie dezelfde naam heeft als een cookie die AIR for TV beheert, bevat de aanvraag twee cookies met dezelfde naam. De waarden van de twee cookies kunnen verschillend zijn.*

- Stel `manageCookies` in op `false`. Deze waarde geeft aan dat de toepassing verantwoordelijk is voor het verzenden van cookies in HTTP-aanvragen en voor het onthouden van de cookies in de HTTP-respons.

Zie URLRequest voor meer informatie.

# Workflow voor het ontwikkelen van een AIR for TV-toepassing

U kunt AIR for TV-toepassingen ontwikkelen met de volgende ontwikkeltools van het Adobe Flash-platform:

- Adobe Flash Professional

  Adobe Flash Professional CS5.5 ondersteunt AIR 2.5 for TV, de eerste versie van AIR die AIR for TV-toepassingen ondersteunt.

- Adobe Flash® Builder®

  Flash Builder 4.5 ondersteunt AIR 2.5 for TV.

- De SDK van AIR

  Vanaf AIR 2.5. kunt u de via de opdrachtregel verschafte tools van de SDK van AIR gebruiken voor het ontwikkelen van toepassingen. Zie http://www.adobe.com/nl/products/air/sdk/ voor informatie over het downloaden van de SDK van AIR.

## Flash Professional gebruiken

Het gebruik van Flash Professional voor het ontwikkelen, testen en publiceren van AIR for TV-toepassingen valt te vergelijken met het gebruiken van het programma voor AIR-bureaubladtoepassingen.

Als u ActionScript 3.0-code gebruikt, dient u echter alleen de klassen en methoden te gebruiken die worden ondersteund door de AIR-profielen `tv` en `extendedTV`. Zie “Apparaatprofielen” op pagina 257 voor nadere informatie.

### Projectinstellingen

Ga als volgt te werk om uw project in te stellen voor een AIR for TV-toepassing:

- Stel de Speler in op minimaal AIR 2.5 in het tabblad Flash van het dialoogvenster Publicatie-instellingen.
- In het tabblad Algemeen van het dialoogvenster Adobe AIR - instellingen (Instellingen voor toepassing en installer) stelt u het profiel in op `TV` of `extended` TV.

### Foutopsporing

U kunt uw toepassing uitvoeren met gebruik van de AIR Debug Launcher in Flash Professional. Ga als volgt te werk:

- Als u de toepassing wilt uitvoeren in de foutopsporingsmodus, selecteert u:

  Foutopsporing > Fouten opsporen in film > In AIR Debug Launcher (Bureaublad).

  Als u deze selectie eenmaal hebt aangebracht, kunt u voor volgende foutopsporingen het volgende selecteren:

  Foutopsporing > Fouten opsporen in film > Foutopsporing.

- Als u de toepassing wilt uitvoeren zonder de mogelijkheden van de foutopsporingsmodus, selecteert u:

  Besturing > Film testen > In AIR Debug Launcher (Bureaublad).

  Als u deze selectie eenmaal hebt aangebracht, kunt u Besturing > Film testen > Testen kiezen voor volgende foutopsporingen.

Aangezien u het AIR-profiel hebt ingesteld op TV of extended TV, verschaft de AIR Debug Launcher een menu met de naam Toetsen afstandsbediening. U kunt dit menu gebruiken om het indrukken van toetsen op een afstandsbediening te simuleren.

Zie "Foutopsporing op afstand met Flash Professional" op pagina 153 voor meer informatie.

### Native extensies gebruiken

Als uw toepassing een native extensie gebruikt, volgt u de instructies bij "Takenlijst voor het gebruik van een native extensie" op pagina 158.

Bedenk echter het volgende wanneer een toepassing gebruikmaakt van native extensies:

- U kunt de toepassing niet publiceren met gebruik van Flash Professional. Gebruik ADT om de toepassing te publiceren. Zie "Verpakken met ADT" op pagina 150.
- U kunt geen fouten opsporen in de toepassing met gebruik van Flash Professional en u kunt de toepassing niet uitvoeren met gebruik van Flash Professional. Gebruik ADL om fouten in de toepassing op te sporen vanaf de ontwikkelcomputer. Zie "Apparaatsimulatie met ADL" op pagina 151.

## Flash Builder gebruiken

Vanaf Flash Builder-versie 4.5 biedt Flash Builder ondersteuning voor ontwikkelen in AIR for TV. Het gebruik van Flash Builder voor het ontwikkelen, testen en publiceren van AIR for TV-toepassingen valt te vergelijken met het gebruiken van het programma voor AIR-bureaubladtoepassingen.

### De toepassing instellen

Zorg ervoor dat uw toepassing:

- Het `Application`-element gebruikt als de containerklasse in het MXML-bestand (als u een MXML-bestand gebruikt):

```
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
    xmlns:s="library://ns.adobe.com/flex/spark"
    xmlns:mx="library://ns.adobe.com/flex/mx">

    <!-- Place elements here. -->

</s:Application>.
```

  ***Belangrijk:*** *AIR for TV-toepassingen bieden geen ondersteuning voor het* `WindowedApplication`*-element.*

***Opmerking:*** *het is niet nodig een MXML-bestand te gebruiken, u kunt in plaats daarvan ook een ActionScript 3.0-Project maken.*

- Alleen ActionScript 3.0-klassen en -methoden gebruikt die de AIR-profielen `tv` en `extendedTV` ondersteunen. Zie "Apparaatprofielen" op pagina 257 voor nadere informatie.

Zorg er bovendien in het XML-bestand van uw toepassing voor dat:

- Het `xmlns`-kenmerk van het `application`-element is ingesteld op AIR 2.5:

```
<application xmlns="http://ns.adobe.com/air/application/2.5">
```

- Het `supportedProfiles`-element `tv` of `extendedTV` bevat:

```
<supportedProfiles>tv</supportedProfiles>
```

### Fouten opsporen in de toepassing

U kunt uw toepassing uitvoeren met gebruik van de AIR Debug Launcher in Flash Builder. Ga als volgt te werk:

1. Selecteer Run (Uitvoeren) > Debug Configurations (Fouten opsporen in configuraties).
2. Controleer of het veld Profile (Profiel) is ingesteld op Desktop.
3. Selecteer Run (Uitvoeren) > Debug (Foutopsporing) om de foutopsporingsmodus te activeren of selecteer Run (Uitvoeren) > Run (Uitvoeren) om de toepassing uit te voeren zonder de mogelijkheden van de foutopsporingsmodus.

Aangezien u het `supportedProfiles`-element hebt ingesteld op TV of extended TV, verschaft de AIR Debug Launcher een menu met de naam Remote Control Buttons (Toetsen afstandsbediening). U kunt dit menu gebruiken om het indrukken van toetsen op een afstandsbediening te simuleren.

Zie "Foutopsporing op afstand met Flash Builder" op pagina 154 voor meer informatie.

### Native extensies gebruiken

Als uw toepassing een native extensie gebruikt, volgt u de instructies bij "Takenlijst voor het gebruik van een native extensie" op pagina 158.

Bedenk echter het volgende wanneer een toepassing gebruikmaakt van native extensies:

- U kunt de toepassing niet publiceren met gebruik van Flash Builder. Gebruik ADT om de toepassing te publiceren. Zie "Verpakken met ADT" op pagina 150.
- U kunt geen fouten opsporen in de toepassing met gebruik van Flash Builder en u kunt de toepassing niet uitvoeren met gebruik van Flash Builder. Gebruik ADL om fouten in de toepassing op te sporen vanaf de ontwikkelcomputer. Zie "Apparaatsimulatie met ADL" op pagina 151.

# Eigenschappen van het descriptorbestand van de AIR for TV-toepassing

Net als bij overige AIR-toepassingen stelt u de basiseigenschappen van de toepassing in het toepassingsdescriptorbestand in. Tv-profieltoepassingen negeren een aantal van de bureaubladspecifieke eigenschappen, zoals venstergrootte en transparantie. Toepassingen die zijn gericht op apparaten in het `extendedTV`-profiel kunnen native extensies gebruiken. Deze toepassingen identificeren de native extensies die worden gebruikt in een `extensions`-element.

# Algemene instellingen

Verschillende instellingen voor toepassingsdescriptors zijn belangrijk voor alle tv-profieltoepassingen.

## Vereiste AIR-runtimeversie

Geef de versie op van de AIR-runtime die wordt vereist door uw toepassing en gebruik daarbij de naamruimte van het toepassingsdescriptorbestand.

De naamruimte, die wordt toegewezen in het `application`-element, bepaalt voor een groot deel welke functies uw toepassing kan gebruiken. Neem bijvoorbeeld een toepassing die de AIR 2.5-naamruimte gebruikt, maar waarvan de gebruiker een nieuwere versie heeft geïnstalleerd. In dit geval vertoont de toepassing toch de functionaliteit van AIR 2.5, ook als deze afwijkt van de nieuwere AIR-versie. Uw toepassing heeft uitsluitend toegang tot de nieuwe functionaliteit en functies als u de naamruimte wijzigt en een update publiceert. Beveiligingsfixes zijn een belangrijke uitzondering op deze regel.

Geef de naamruimte op met behulp van het `xmlns`-attribuut van het basiselement `application`:

```
<application xmlns="http://ns.adobe.com/air/application/2.5">
```

AIR 2.5 is de eerste versie van AIR die tv-toepassingen ondersteunt.

## Toepassingsidentiteit

Verschillende instellingen moeten uniek zijn voor elke toepassing die u publiceert. Deze instellingen bevatten de elementen `id`, `name` en `filename`.

```
<id>com.example.MyApp</id>
<name>My Application</name>
<filename>MyApplication</filename>
```

## Toepassingsversie

Geef de toepassingsversie op in het element `versionNumber`. Wanneer u een waarde opgeeft voor `versionNumber`, kunt u een reeks gebruiken van hoogstens drie nummers, gescheiden door punten, zoals "0.1.2". Elk segment van het versienummer kan hoogstens drie cijfers hebben. (Met andere woorden: “999.999.999” is het hoogste versienummer dat is toegestaan.) U hoeft niet alle drie de segmenten in het cijfer op te nemen. “1” en “1.0” zijn ook geldige versienummers.

U kunt ook een label voor de versie opgeven met behulp van het element `versionLabel`. Wanneer u een versielabel toevoegt, wordt dit weergegeven in plaats van het versienummer.

```
<versionNumber>1.23.7<versionNumber>
<versionLabel>1.23 Beta 7</versionLabel>
```

## Hoofdtoepassings-SWF

Geef het hoofdtoepassings-SWF-bestand op in het onderliggende element `versionLabel` van het element `initialWindow`. Wanneer u apparaten in het tv-profiel als doel instelt, moet u een SWF-bestand gebruiken (HTML-toepassingen worden niet ondersteund).

```
<initialWindow>
    <content>MyApplication.swf</content>
</initialWindow>
```

U moet het bestand in het AIR-pakket opnemen (met behulp van ADT of uw IDE). Door alleen te verwijzen naar de naam in de toepassingsdescriptor wordt het bestand niet automatisch in het pakket opgenomen.

### Eigenschappen van het hoofdscherm

Verschillende onderliggende elementen van het element `initialWindow` bepalen de initiële weergave en de functionaliteit van het hoofdscherm van de toepassing. De meeste van deze eigenschappen worden genegeerd op apparaten in de tv-profielen, maar u kunt het element `fullScreen` wel gebruiken:

- **`fullScreen`** — Geeft op of de toepassing de volledige weergave moet gebruiken, of de weergave moet delen met de standaardinterface.

  ```
  <fullScreen>true</fullScreen>
  ```

### Het element visible

Het element `visible` is een onderliggend element van het element `initialWindow`initialWindow AIR for TV negeert het element `visible`, omdat de inhoud van uw toepassing altijd zichtbaar is op AIR for TV-apparaten.

Stel het element `visible` echter in op `true` als uw toepassing ook bestemd is voor bureaubladapparatuur.

Op bureaubladapparatuur is de standaardwaarde van dit element `false`. Als u het element `visible` dus niet opneemt, is de inhoud van de toepassing niet zichtbaar op bureaubladapparatuur. Hoewel u de ActionScript-klasse NativeWindow kunt gebruiken om inhoud zichtbaar te maken op bureaubladapparatuur, ondersteunen TV-apparaatprofielen de klasse NativeWindow niet. Als u probeert de klasse NativeWindow te gebruiken voor een toepassing die wordt uitgevoerd op een AIR for TV-apparaat, kan de toepassing niet worden geladen. Het maakt niet uit of u een methode van de klasse NativeWindow aanroept, toepassingen die deze klasse gebruiken, worden niet geladen op AIR for TV-apparaten.

### Ondersteunde profielen

Als uw toepassing uitsluitend is bedoeld voor televisie, kunt u de installatie ervan op andere typen apparaten voorkomen. Sluit de overige profielen uit in de lijst met ondersteunde profielen in het element `supportedProfiles`:

```
<supportedProfiles>tv extendedTV</supportedProfiles>
```

Als een toepassing een native extensie gebruikt, neemt u alleen het `extendedTV`-profiel op in de lijst met ondersteunde profielen:

```
<supportedProfiles>extendedTV</supportedProfiles>
```

Als u het element `supportedProfiles` weglaat, wordt aangenomen dat de toepassing alle profielen ondersteunt.

Neem niet *alleen* het profiel `tv` op in de lijst `supportedProfiles`. Op sommige tv-apparaten wordt AIR for TV altijd uitgevoerd in een modus die overeenkomt met het profiel `extendedTV`. Op deze manier kan AIR for TV toepassingen uitvoeren die native extensies gebruiken. Als uw element `supportedProfiles` alleen `tv` opgeeft, wordt aangegeven dat uw inhoud niet compatibel is met de AIR for TV-modus voor `extendedTV`. Daarom worden toepassingen die alleen het profiel `tv` opgeven op sommige tv's niet geladen.

Een lijst met ActionScript-klassen die worden ondersteund in de profielen `tv` en `extendedTV` vindt u in "Mogelijkheden van de verschillende profielen" op pagina 258.

## Vereiste native extensies

Toepassingen die het profiel `extendedTV` ondersteunen, kunnen native extensies gebruiken.

Geef alle native extensies op die door de AIR-toepassing worden gebruikt in het descriptorbestand van de toepassing en gebruik hiervoor de elementen `extensions` en `extensionID`. Het volgende voorbeeld illustreert de syntaxis voor het opgeven van twee vereiste native extensies:

```
<extensions>
    <extensionID>com.example.extendedFeature</extensionID>
    <extensionID>com.example.anotherFeature</extensionID>
</extensions>
```

Als een extensie niet wordt vermeld, kan de toepassing deze niet gebruiken.

Het `extensionID`-element heeft dezelfde waarde als het element `id` in het descriptorbestand van de extensie. Het descriptorbestand van de extensie is een XML-bestand met de naam "extension.xml". Het wordt verpakt in het ANE-bestand dat u ontvangt van de apparaatproducent.

Als u een extensie vermeldt in het element `extensions`, maar als deze extensie niet geïnstalleerd is op het AIR for TV-apparaat, kan de toepassing niet worden uitgevoerd. De uitzondering op deze regel betreft gevallen waarin het ANE-bestand dat u verpakt met uw AIR for TV-toepassing over een sectie van de extensie beschikt. In dat geval kan de toepassing worden uitgevoerd en wordt de sectie van de extensie gebruikt. De sectie beschikt over ActionScript-code, maar niet over native code.

## Toepassingspictogrammen

De vereisten voor toepassingspictogrammen op televisieapparaten zijn afhankelijk van het apparaat. De apparaatproducent bepaalt bijvoorbeeld:

- De vereiste pictogrammen en pictogramgrootten.
- De vereiste bestandstypen en naamgevingsconventies.
- Hoe u pictogrammen verschaft aan uw toepassing, bijvoorbeeld of de pictogrammen samen met de toepassing worden verpakt.
- Of u de pictogrammen opgeeft in een `icon`-element in het descriptorbestand van de toepassing.
- Functionaliteit als de toepassing geen pictogrammen verschaft.

Vraag de producent van het apparaat om nadere informatie.

## Genegeerde instellingen

Toepassingen op televisieapparaten negeren toepassingsinstellingen die gelden voor mobiele functies, functies voor native venster of besturingssysteemfuncties voor bureaublad. De genegeerde instellingen zijn:

- allowBrowserInvocation
- aspectRatio
- autoOrients
- customUpdateUI
- fileTypes
- height
- installFolder
- maximizable
- maxSize
- minimizable
- minSize
- programMenuFolder

- renderMode
- resizable
- systemChrome
- title
- transparent
- visible
- width
- x
- y

# Een AIR for TV-toepassing verpakken

## Verpakken met ADT

U kunt het ADT-opdrachtregelprogramma van AIR gebruiken om een AIR for TV-toepassing in te pakken. Vanaf AIR SDK versie 2.5 ondersteunt ADT verpakken voor tv-apparaten. Compileer alle ActionScript- en MXML-code voordat u een pakket gaat maken. U moet ook over een certificaat voor het ondertekenen van code beschikken. U kunt een certificaat maken met behulp van de ADT-opdracht -certificate.

Zie "AIR Developer Tool (ADT)" op pagina 173 voor een gedetailleerde beschrijving van ADT-opdrachten en -opties.

### Een AIR-pakket maken

Als u een AIR-pakket wilt maken, gebruikt u de AIR-pakketopdracht:

```
adt -package -storetype pkcs12 -keystore ../codesign.p12 myApp.air myApp-app.xml myApp.swf icons
```

Het voorbeeld gaat ervan uit dat:

- Het pad van het ADT-hulpprogramma zich op de pad-definitie van de shell van uw opdrachtregel bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318.)
- Het certificaat codesign.p12 zich in de bovenliggende map bevindt van de locatie waar u de ADT-opdracht uitvoert.

Voer de opdracht uit vanuit de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn myApp-app.xml (het descriptorbestand van de toepassing), myApp.swf en een map met pictogrammen.

Wanneer u volgens het voorbeeld de opdracht uitvoert, wordt u door ADT gevraagd het keystore-wachtwoord in te voeren. Niet alle shell-programma's geven de wachtwoordtekens tijdens het typen weer. Druk gewoon op Enter als u het wachtwoord hebt getypt. U kunt ook de `storepass`-parameter gebruiken om het wachtwoord op te nemen in de ADT-opdracht.

### Een AIRN-pakket maken

Als uw AIR for TV-toepassing een native extensie gebruikt, maakt u een AIRN-pakket in plaats van een AIR-pakket. Als u een AIRN-pakket wilt maken, gebruikt u de ADT-pakketopdracht en stelt u het doeltype in op `airn`.

```
adt -package -storetype pkcs12 -keystore ../codesign.p12 -target airn myApp.airn myApp-app.xml
myApp.swf icons -extdir C:\extensions
```

Het voorbeeld gaat ervan uit dat:

- Het pad van het ADT-hulpprogramma zich op de pad-definitie van de shell van uw opdrachtregel bevindt. (Zie "Omgevingsvariabelen van het pad" op pagina 318.)
- Het certificaat codesign.p12 zich in de bovenliggende map bevindt van de locatie waar u de ADT-opdracht uitvoert.
- De parameter `-extdir` benoemt een map met de ANE-bestanden die de toepassing gebruikt.

  Deze ANE-bestanden bevatten een exclusieve ActionScript-sectie of -simulatorversie van de extensie. De versie van de extensie die de native code bevat, is geïnstalleerd op het AIR for TV-apparaat.

Voer de opdracht uit vanuit de map met de toepassingsbestanden. De toepassingsbestanden in het voorbeeld zijn myApp-app.xml (het descriptorbestand van de toepassing), myApp.swf en een map met pictogrammen.

Wanneer u volgens het voorbeeld de opdracht uitvoert, wordt u door ADT gevraagd het keystore-wachtwoord in te voeren. Niet alle shell-programma's geven de wachtwoordtekens tijdens het typen weer. Druk gewoon op Enter als u het wachtwoord hebt getypt. U kunt ook de `storepass`-parameter gebruiken om het wachtwoord op te nemen in de opdracht.

Bovendien kunt u een AIRI-bestand maken voor een AIR for TV-toepassing die native extensies gebruikt. Het AIRI-bestand komt overeen met het AIRN-bestand, het is alleen niet ondertekend. Bijvoorbeeld:

```
adt -prepare myApp.airi myApp.xml myApp.swf icons -extdir C:\extensions
```

Als u klaar bent om de toepassing te ondertekenen, kunt u een AIRN-bestand maken van het AIRI-bestand:

```
adt -package -storetype pkcs12 -keystore ../codesign.p12 -target airn myApp.airn myApp.airi
```

Zie voor meer informatie Native extensies ontwikkelen voor Adobe AIR.

## Verpakken met Flash Builder of Flash Professional

Met Flash Professional en Flash Builder kunt u de AIR-pakketten publiceren of exporteren zonder u ADT zelf uit hoeft te voeren. De procedure voor het maken van een AIR-pakket voor een AIR-toepassing wordt behandeld in de documentatie voor die programma's.

Momenteel kunt u echter alleen met ADT AIRN-pakketten maken, de toepassingspakketten voor AIR for TV-toepassingen die native extensies gebruiken.

Raadpleeg de volgende bronnen voor meer informatie:

- AIR-toepassingen met Flash Builder verpakken
- Publiceren voor Adobe AIR met Flash Professional

# Fouten opsporen bij AIR for TV-toepassingen

## Apparaatsimulatie met ADL

De snelste en eenvoudigste manier om de meeste functies te testen en op fouten te controleren, is om uw toepassing op uw ontwikkelcomputer uit te voeren met behulp van het hulpmiddel Adobe Debug Launcher (ADL).

ADL maakt gebruik van het element `supportedProfiles` in het descriptorbestand van de toepassing om te kiezen welk profiel wordt gebruikt. Specifiek geldt het volgende:

- Als er meer dan één profiel wordt weergegeven, wordt het eerste in de lijst door ADL gebruikt.

- U kunt de ADL-parameter `-profile` gebruiken om een van de andere profielen in de lijst `supportedProfiles` te selecteren.
- Als u geen `supportedProfiles`-element in het descriptorbestand van de toepassing opneemt, kan elk profiel worden opgegeven voor het argument `-profile`.

Gebruik bijvoorbeeld de volgende opdracht om een toepassing uit te voeren om het profiel `tv` te simuleren:

```
adl -profile tv myApp-app.xml
```

Bij simulaties van het profiel `tv` of `extendedTV` op het bureaublad met ADL wordt de toepassing in een omgeving uitgevoerd die meer overeenkomt met een doelapparaat. Bijvoorbeeld:

- ActionScript-API's die geen onderdeel uitmaken van het profiel in het argument `-profile`, zijn niet beschikbaar.
- ADL maakt de invoer via besturingsapparaten, zoals afstandsbedieningen, mogelijk via menuopdrachten.
- Wanneer u `tv` of `extendedTV` in het argument `-profile` opgeeft, kan ADL de klasse StageVideo op het bureaublad simuleren.
- Wanneer u `extendedTV` opgeeft in het argument `-profile`, kan de toepassing native extensiesecties of -simulators gebruiken die zijn verpakt met het AIRN-bestand van de toepassing.

Omdat ADL de toepassing op het bureaublad uitvoert, is het testen van AIR for TV-toepassingen met ADL echter beperkt:

- Het is geen weergave van de prestaties van de toepassing op het apparaat. Prestatietests voert u op het doelapparaat uit.
- Het is een geen simulatie van de beperkingen van het StageVideo-object. Normaal gesproken gebruikt u de StageVideo-klasse, niet de Video-klasse, om een video af te spelen wanneer u AIR for TV-apparaten gebruikt. De klasse StageVideo maakt gebruik van prestatievoordelen van de hardware van het apparaat, maar heeft weergavebeperkingen. ADL speelt de video op het bureaublad af zonder deze beperkingen. Test daarom het afspelen van video op het doelapparaat.
- De native code van een native extensie kan niet worden gesimuleerd. U kunt het profiel `extendedTV`, dat native extensies ondersteunt, echter specificeren in het ADL-argument `-profile`. U kunt met ADL de exclusieve ActionScript-sectie of -simulatorversie van de extensie in het ANE-pakket testen. Meestal bevat de op het apparaat geïnstalleerde, overeenkomende extensie echter ook de native code. Als u het gebruik van de extensie met de native code wilt testen, voert u de toepassing uit op het doelapparaat.

Zie “ADL (AIR Debug Launcher)” op pagina 167 voor meer informatie.

**Native extensies gebruiken**

Als uw toepassing native extensies gebruikt, ziet de ADL-opdracht eruit zoals in het volgende voorbeeld:

```
adl -profile extendedTV -extdir C:\extensionDirs myApp-app.xml
```

Het voorbeeld gaat ervan uit dat:

- Het pad van het ADL-hulpprogramma zich op de pad-definitie van de shell van uw opdrachtregel bevindt. (Zie “Omgevingsvariabelen van het pad” op pagina 318.)
- De huidige map de toepassingsbestanden bevat. Dit zijn de SWF-bestanden en het descriptorbestand van de toepassing, in dit voorbeeld myApp-app.xml.
- De parameter `-extdir` een map benoemt die een map bevat voor elke native extensie die door de toepassing wordt gebruikt. Al deze mappen bevatten het *niet verpakte* ANE-bestand van een native extensie. Bijvoorbeeld:

```
C:\extensionDirs
    extension1.ane
        META-INF
            ANE
                default
                    library.swf
                extension.xml
            signatures.xml
        catalog.xml
        library.swf
        mimetype
    extension2.ane
        META-INF
            ANE
                default
                    library.swf
                extension.xml
            signatures.xml
        catalog.xml
        library.swf
        mimetype
```

Deze niet-verpakte ANE-bestanden bevatten een exclusieve ActionScript-sectie of -simulatorversie van de extensie. De versie van de extensie die de native code bevat, is geïnstalleerd op het AIR for TV-apparaat.

Zie voor meer informatie Native extensies ontwikkelen voor Adobe AIR.

**Beheer invoer**

ADL simuleert de afstandsbedieningsknoppen op een tv-apparaat. U kunt de invoer van deze knoppen naar het gesimuleerde apparaat verzenden met behulp van het menu dat wordt weergegeven wanneer ADL wordt gestart met een van de tv-profielen.

**Schermgrootte**

U kunt uw toepassing in verschillende schermgroottes testen door de ADL-parameter `-screensize` in te stellen. U kunt een tekenreeks opgeven met de vier waarden voor breedte en hoogte van normaal en gemaximaliseerd scherm.

Geef altijd de pixeldimensies op voor een staande afdrukstand. Dit houdt in dat de breedte een kleinere waarde heeft dan de hoogte. Bijvoorbeeld:

```
adl -screensize 728x1024:768x1024 myApp-app.xml
```

## Traceringsinstructies

Bij het uitvoeren van uw tv-toepassing op het bureaublad wordt de traceringsuitvoer afgedrukt naar het foutopsporingsprogramma of naar het terminalvenster dat wordt gebruikt om ADL uit te voeren.

## Foutopsporing op afstand met Flash Professional

U kunt Flash Professional gebruiken om vanaf een afstand fouten in uw AIR for TV-toepassing op te sporen terwijl de toepassing wordt uitgevoerd op het doelapparaat. De stappen voor het instellen van foutopsporing op afstand verschillen echter per apparaat. De Adobe® AIR® for TV MAX 2010 Hardware Development Kit bevat bijvoorbeeld gedetailleerde informatie over dat apparaat.

De volgende stappen dienen echter te worden uitgevoerd ter voorbereiding op foutopsporing op afstand, ongeacht het doelapparaat:

1 Selecteer Foutopsporing toestaan in het tabblad Flash van het dialoogvenster Publicatie-instellingen.

   Flash Professional neemt de foutopsporingsinformatie dan op in alle SWF-bestanden die worden gemaakt op basis van uw FLA-bestand.

2 Selecteer op het tabblad Handtekening van het dialoogvenster Adobe AIR -instellingen (Instellingen voor toepassing en installer) de optie voor het voorbereiden van een AIRI-bestand (AIR Intermediate).

   Wanneer u uw toepassing nog aan het ontwikkelen bent, volstaat het gebruik van een AIRI-bestand waarvoor geen digitale handtekening is vereist.

3 Publiceer uw toepassing, waarbij het AIRI-bestand wordt gemaakt.

De laatste stappen bestaan uit het installeren en uitvoeren van de toepassing op het doelapparaat. Deze stappen zijn echter afhankelijk van het apparaat.

## Foutopsporing op afstand met Flash Builder

U kunt ook Flash Builder gebruiken om vanaf een afstand fouten in uw AIR for TV-toepassing op te sporen terwijl de oplossing wordt uitgevoerd op het doelapparaat. De stappen voor het uitvoeren van foutopsporing op afstand verschillen echter per apparaat.

De volgende stappen dienen echter te worden uitgevoerd ter voorbereiding op foutopsporing op afstand, ongeacht het doelapparaat:

1 Selecteer Project > Exportversie. Selecteer de optie voor het voorbereiden van een AIRI-bestand (AIR Intermediate).

   Wanneer u uw toepassing nog aan het ontwikkelen bent, volstaat het gebruik van een AIRI-bestand waarvoor geen digitale handtekening is vereist.

2 Publiceer uw toepassing, waarbij het AIRI-bestand wordt gemaakt.

3 Wijzig het AIRI-pakket van de toepassing, zodat het de SWF-bestanden met foutopsporingsgegevens bevat.

   De SWF-bestanden met foutopsporingsgegevens bevinden zich in de Flash Builder-projectmap voor de toepassing, en wel in de map "bin-debug". Vervang de SWF-bestanden in het AIRI-pakket door de SWF-bestanden in de map "bin-debug".

Op een Windows-ontwikkelcomputer doet u dat als volgt:

1 Wijzig de naam van het AIRI-pakketbestand, zodat het de bestandsnaamtoevoeging .zip heeft in plaats van .airi.

2 Pak de inhoud van het gezipte bestand uit.

3 Vervang de SWF-bestanden in de uitgepakte mapstructuur door de SWF-bestanden uit de map "bin-debug".

4 Comprimeer de bestanden in de uitgepakte map weer tot een gezipt bestand.

5 Wijzig de naam van het gezipte bestand weer terug in de bestandsnaamtoevoeging .airi.

Als u een Mac-ontwikkelmachine gebruikt, zijn de stappen voor deze vervanging afhankelijk van het individuele apparaat. Vaak dient u echter het volgende te doen:

1 Installeer het AIRI-pakket op het doelapparaat.

2 Vervang de SWF-bestanden in de installatiemap van de toepassing op het doelapparaat door de SWF-bestanden uit de map "bin-debug".

Neem bijvoorbeeld het apparaat dat is meegeleverd met de Adobe AIR for TV MAX 2010 Hardware Development Kit. Installeer het AIRI-pakket, zoals wordt beschreven in de documentatie bij de kit. Gebruik vervolgens telnet op de opdrachtregel van uw Mac-ontwikkelcomputer om het doelapparaat te benaderen. Vervang de SWF-bestanden in de installatiemap van de toepassing op /opt/adobe/stagecraft/apps/<*application name*>/ door de SWF-bestanden uit de map "bin-debug".

De volgende stappen zijn bedoeld voor foutopsporing op afstand met Flash Builder en het apparaat dat is meegeleverd met de Adobe AIR for TV MAX 2010 Hardware Development Kit.

1. Op de computer waarop Flash Builder wordt uitgevoerd, ofwel uw ontwikkelcomputer, voert u de AIR for TV-apparaatconnector uit die bij the MAX 2010 Hardware Development Kit is geleverd. Deze toont het IP-adres van uw ontwikkelcomputer.
2. Start op het apparaat van de hardwarekit de ook bij de ontwikkelkit geleverde toepassing DevMaster.
3. Typ het IP-adres van uw ontwikkelcomputer, zoals dat wordt aangegeven in de AIR for TV-apparaatconnector, in de toepassing DevMaster.
4. Zorg ervoor dat Enable Remote Debugging (Foutopsporing op afstand inschakelen) is ingeschakeld in de DevMaster-toepassing.
5. Sluit de DevMaster-toepassing.
6. Selecteer Starten in de AIR for TV-connector op de ontwikkelcomputer.
7. Start een andere toepassing op het hardwarekitapparaat. Controleer of de traceerinformatie wordt weergegeven in de AIR for TV-apparaatconnector.

   Als deze informatie niet wordt weergegeven, is er geen verbinding tussen de ontwikkelcomputer en het apparaat van de hardwarekit. Controleer of de voor traceerinformatie gebruikte poort op de ontwikkelcomputer beschikbaar is. U kunt een andere poort kiezen in de AIR for TV-apparaatconnector. Zorg er ook voor dat uw firewall toegang tot de gekozen poort toestaat.

Start daarna het foutopsporingsprogramma in Flash Builder. Ga als volgt te werk:

1. Selecteer in Flash Builder Uitvoeren > Debug Configurations (Fouten opsporen in configuraties).
2. Kopieer de naam van het project vanuit de voor lokale foutopsporing bedoelde bestaande foutopsporingsconfiguratie.
3. Selecteer Web Application (Webtoepassing) in het dialoogvenster Debug Configurations (Fouten opsporen in configuraties). Selecteer vervolgens het pictogram New Launch Configuration (Nieuwe opstartconfiguratie).
4. Plak de projectnaam in het veld Project.
5. Schakel Use Default (Standaardwaarde gebruiken) uit in de sectie URL Or Path To Launch (URL of pad voor opstarten). Typ ook **`about:blank`** in het tekstveld.
6. Selecteer Apply (Toepassen) om uw wijzigingen op te slaan.
7. Select Debug (Foutopsporing) om de foutopsporing van Flash Builder te starten.
8. Start uw toepassing op het apparaat van de hardwarekit.

Nu kunt u de foutopsporing van Flash Builder bijvoorbeeld gebruiken om onderbrekingspunten in te stellen en variabelen te onderzoeken.

# Hoofdstuk 9: Native extensies gebruiken voor Adobe AIR

Native extensies voor Adobe AIR verschaffen ActionScript API's waarmee u toegang krijgt tot apparaatspecifieke functies die in de native code zijn geprogrammeerd. Ontwikkelaars van native extensies werken soms samen met apparaatfabrikanten en soms met externe ontwikkelaars.

Raadpleeg Native extensies ontwikkelen voor Adobe AIR als u een native extensie ontwikkelt.

Een native extensie is een combinatie van:

- ActionScript-klassen.
- Native code.

Als ontwikkelaar van AIR-toepassingen die gebruikmaakt van een native extensie, werkt u echter alleen met de ActionScript-klassen.

Native extensies zijn handig in de volgende gevallen:

- De native code-implementatie geeft toegang tot platformspecifieke functies. Deze functies zijn niet beschikbaar in de geïntegreerde ActionScript-klassen en het is niet mogelijk deze te implementeren in toepassingsspecifieke ActionScript-klassen. De native code-implementatie kan deze functionaliteit verschaffen, omdat deze toegang heeft tot apparaatspecifieke hardware en software.
- Soms is een native code-implementatie sneller dan een implementatie die alleen gebruikmaakt van ActionScript.
- De native code-implementatie kan ActionScript toegang geven tot verouderde native code.

Het Adobe Developer Center bevat enkele voorbeelden van native extensies. Een native extensie verleent AIR-toepassingen bijvoorbeeld toegang tot de trilfunctie van Android. Zie Native extensies voor Adobe AIR.

## ANE-bestanden (AIR Native Extension)

Ontwikkelaars van native extensies verpakken een native extensie in een ANE-bestand. Een ANE-bestand is een archiefbestand met de vereiste bibliotheken en bronnen voor de native extensie.

Voor bepaalde apparaten bevat het ANE-bestand de native codebibliotheek waarvan de native extensie gebruikmaakt. Maar voor andere apparaten wordt de native codebibliotheek op het apparaat geïnstalleerd. In sommige gevallen heeft de native extensie helemaal geen native code voor een bepaald apparaat. Deze wordt alleen geïmplementeerd met ActionScript.

Ontwikkelaars van AIR-toepassingen gebruiken het ANE-bestand als volgt:

- Neem het ANE-bestand op in het bibliotheekpad van de toepassing, net zoals u een SWC-bestand opneemt in het bibliotheekpad. Zo stelt u de toepassing in staat te verwijzen naar de ActionScript-klassen van de extensie.

  ***Opmerking:*** *wanneer u de toepassing compileert, moet u ervoor zorgen dat u de dynamic link-optie gebruikt voor het ANE-bestand. Als u Flash Builder gebruikt, moet u de optie Extern opgeven in het ActionScript Builder-deelvenster met padeigenschappen. Als u de opdrachtregel gebruikt, typt u het volgende: -external-library-path.*

- Plaats het ANE-bestand in een pakket met de AIR-toepassing.

# Native extensies vergelijken met de NativeProcess-klasse van ActionScript

ActionScript 3.0 verschaft een NativeProcess-klasse. Met deze klasse kan een AIR-toepassing native processen uitvoeren op het hostbesturingssysteem. Deze functie valt te vergelijken met native extensies die toegang verlenen tot platformspecifieke functies en bibliotheken. Neem het volgende in overweging wanneer u moet beslissen of u de NativeProcess-klasse of een native extensie gaat gebruiken:

- Alleen het AIR-profiel `extendedDesktop` biedt ondersteuning voor de NativeProcess-klasse. Voor toepassingen met de AIR-profielen `mobileDevice` en `extendedMobileDevice` zijn native extensies dus de enige mogelijkheid.
- Ontwikkelaars van native extensies verschaffen vaak native implementaties voor verschillende platforms, maar ze verschaffen doorgaans dezelfde ActionScript-API voor alle platforms. Wanneer de klasse NativeProcess wordt gebruikt, kan de ActionScript-code waarmee het native proces wordt gestart per platform variëren.
- De klasse NativeProcess start een afzonderlijk proces, terwijl een native extensie in hetzelfde proces als de AIR-toepassing wordt uitgevoerd. U kunt dus beter de NativeProcess-klasse gebruiken als u zich zorgen maakt over vastlopende code. De verschillende processen betekenen echter wel dat u wellicht communicatieafhandeling tussen de processen moet implementeren.

# Native extensies vergelijken met ActionScript-klassebibliotheken (SWC-bestanden)

Een SWC-bestand is een ActionScript-klassebibliotheek in een archiefindeling. Het SWC-bestand bevat een SWF-bestand plus andere bronbestanden. SWC-bestanden vormen een handige manier om ActionScript-klassen te delen in plaats van afzonderlijke ActionScript-code en -bronbestanden te delen.

Een ANE-bestand is een native extensiepakket. Net als een SWC-bestand is een ANE-bestand ook een ActionScript-klassebibliotheek met een SWF-bestand en andere bronbestanden in een archiefindeling. Het belangrijkste verschil tussen ANE- en SWC-bestanden is echter dat alleen ANE-bestanden een native codebibliotheek kunnen bevatten.

***Opmerking:*** *wanneer u de toepassing compileert, moet u ervoor zorgen dat u de dynamic link-optie gebruikt voor het ANE-bestand. Als u Flash Builder gebruikt, moet u de optie Extern opgeven in het ActionScript Builder-deelvenster met padeigenschappen. Als u de opdrachtregel gebruikt, typt u het volgende: -external-library-path.*

### Meer Help-onderwerpen

SWC-bestanden

# Ondersteunde apparaten

In AIR 3 of later kunt u native extensies gebruiken in toepassingen voor de volgende apparaten:

- Android-apparaten, vanaf Android 2.2
- iOS-apparaten, vanaf iOS 4.0
- iOS-simulator, vanaf AIR 3.3
- Blackberry Playbook

- Windows-bureaubladapparaten die ondersteuning bieden voor AIR 3.0
- Mac OS X-bureaubladapparaten die ondersteuning bieden voor AIR 3.0

Vaak richten native extensies zich op meerdere platforms. Het ANE-bestand van de extensie bevat ActionScript-bibliotheken en native bibliotheken voor elk ondersteund platform. De ActionScript-bibliotheken hebben doorgaans dezelfde openbare interfaces voor alle platforms. De native bibliotheken zijn vanzelfsprekend anders.

Soms biedt een native extensie ondersteuning voor een standaardplatform. De implementatie van het standaardplatform heeft alleen ActionScript-code, maar geen native code. Als u een toepassing in een pakket plaatst voor een platform waarvoor de extensie niet op expliciete wijze ondersteuning biedt, gebruikt de toepassing de standaardimplementatie tijdens de uitvoering. Neem bijvoorbeeld een extensie die een functie verschaft die alleen van toepassing is op mobiele apparatuur. De extensie kan ook een standaardimplementatie verschaffen die een bureaubladtoepassing kan gebruiken om de functie te simuleren.

# Ondersteunde apparaatprofielen

De volgende AIR-profielen bieden ondersteuning voor native extensies:

- `extendedDesktop`, vanaf AIR 3.0
- `mobileDevice`, vanaf AIR 3.0
- `extendedMobileDevice`, vanaf AIR 3.0

### Meer Help-onderwerpen

Ondersteuning voor AIR-profielen

# Takenlijst voor het gebruik van een native extensie

Voer de volgende taken uit om een native extensie te gebruiken in uw toepassing:

1. Declareer de extensie in het descriptorbestand van de toepassing.
2. Neem het ANE-bestand op in het bibliotheekpad van uw toepassing.
3. Plaats de toepassing in een pakket.

# De extensie in het descriptorbestand van de toepassing declareren

Alle AIR-toepassingen hebben een descriptorbestand van de toepassing. Wanneer een toepassing een native extensie gebruikt, beschikt het descriptorbestand van de toepassing over een `<extensions>`-element. Bijvoorbeeld:

```
<extensions>
    <extensionID>com.example.Extension1</extensionID>
    <extensionID>com.example.Extension2</extensionID>
</extensions>
```

Het `extensionID`-element heeft dezelfde waarde als het element `id` in het descriptorbestand van de extensie. Het descriptorbestand van de extensie is een XML-bestand met de naam "extension.xml". Dit bestand is in een pakket met het ANE-bestand geplaatst. Zoek met een hulpprogramma voor het decomprimeren van archiefbestanden naar het bestand extension.xml.

# Het ANE-bestand opnemen in het bibliotheekpad van uw toepassing

Neem het ANE-bestand op in uw bibliotheekpad als u een toepassing wilt compileren die een native extensie gebruikt.

## Het ANE-bestand gebruiken met Flash Builder

Als uw toepassing een native extensie gebruikt, neemt u het ANE-bestand voor de native extensie op in het bibliotheekpad. Daarna kunt u met Flash Builder uw ActionScript-code compileren.

Voer de volgende stappen uit in Flash Builder 4.5.1:

1. Wijzig de extensie van het ANE-bestand van .ane in .swc. Dit is nodig, omdat Flash Builder het bestand anders niet kan vinden.
2. Selecteer Project > Eigenschappen voor uw Flash Builder-project.
3. Selecteer het Flex-bouwpad in het dialoogvenster Eigenschappen.
4. Selecteer op het tabblad Bibliotheekpad de optie SWC-bestand toevoegen....
5. Blader naar het SWC-bestand en selecteer Openen.
6. Selecteer OK in het dialoogvenster SWC-bestand toevoegen...

   Het ANE-bestand wordt nu weergegeven op het tabblad Bibliotheekpad van het dialoogvenster Eigenschappen.
7. Vouw de vermelding van het SWC-bestand uit. Dubbelklik op Koppelingstype om het dialoogvenster ltempadopties bibliotheek te openen.
8. Wijzig het Koppelingstype in Extern in het dialoogvenster Itempadopties bibliotheek.

Nu kunt u uw toepassing compileren, bijvoorbeeld met gebruik van Project > Build Project (Project samenstellen).

## Het ANE-bestand gebruiken met Flash Professional

Als uw toepassing een native extensie gebruikt, neemt u het ANE-bestand voor de native extensie op in het bibliotheekpad. Daarna kunt u met Flash Professional CS5.5 uw ActionScript-code compileren. Ga als volgt te werk:

1. Wijzig de extensie van het ANE-bestand van .ane in .swc. Dit is nodig, omdat Flash Professional het bestand anders niet kan vinden.
2. Selecteer Bestand > ActionScript-instellingen voor het FLA-bestand.
3. Selecteer het tabblad Bibliotheekpad in het dialoogvenster Geavanceerde ActionScript 3.0-instellingen.
4. Selecteer de knop Bladeren naar SWC-bestand.
5. Blader naar het SWC-bestand en selecteer Openen.

   Het SWC-bestand wordt nu weergegeven in het tabblad Bibliotheekpad van het dialoogvenster Geavanceerde ActionScript 3.0-instellingen.
6. Zorg dat het SWC-bestand is geselecteerd en selecteer de knop Koppelingsopties voor een bibliotheek instellen.

**7** Wijzig het Koppelingstype in Extern in het dialoogvenster Itempadopties bibliotheek.

# Een toepassing die native extensies gebruikt in een pakket plaatsen

Gebruik ADT om een toepassing die native extensies gebruikt in een pakket te plaatsen. Het is niet mogelijk de toepassing met gebruik van Flash Professional CS5.5 of Flash Builder 4.5.1 in een pakket te plaatsen.

Op AIR Developer Tool (ADT) vindt u informatie over het gebruik van ADT.

Met de volgende ADT-opdracht maakt u bijvoorbeeld een DMG-bestand (een native installatiebestand voor Mac OS X) voor een toepassing die native extensies gebruikt:

```
adt -package
    -storetype pkcs12
    -keystore myCert.pfx
    -target native
    myApp.dmg
    application.xml
    index.html resources
    -extdir extensionsDir
```

Met de volgende opdracht maakt u een APK-pakket voor een Android-apparaat:

```
adt -package
    -target apk
    -storetype pkcs12 -keystore ../codesign.p12
    myApp.apk
    myApp-app.xml
    myApp.swf icons
    -extdir extensionsDir
```

En met de volgende opdracht maakt u een iOS-pakket voor een iPhone-toepassing:

```
adt -package
    -target ipa-ad-hoc
    -storetype pkcs12 -keystore ../AppleDistribution.p12
    -provisioning-profile AppleDistribution.mobileprofile
    myApp.ipa
    myApp-app.xml
    myApp.swf icons Default.png
    -extdir extensionsDir
```

Let wel:

- Gebruik een pakkettype met een native installatieprogramma.
- Geef de extensiemap op.
- Zorg ervoor dat het ANE-bestand het doelapparaat van de toepassing ondersteunt.

## Een pakkettype met een native installatieprogramma gebruiken

Het toepassingspakket moet een native installatieprogramma zijn. Het is niet mogelijk een AIR-pakket voor meerdere platforms (een .air-pakket) te maken voor een toepassing die een native extensie gebruikt, omdat native extensies doorgaans native code bevatten. Een native extensie ondersteunt echter doorgaans meerdere native platforms met dezelfde ActionScript-API's. In deze gevallen kunt u hetzelfde ANE-bestand gebruiken in pakketten met verschillende native installatieprogramma's.

In de volgende tabel ziet u een overzicht van de waarde die u moet gebruiken voor de optie `-target` van de ADT-opdracht:

| **Doelplatform van de toepassing** | **-target** |
|---|---|
| Bureaubladapparaten met Mac OS X of Windows | -target native<br>-target bundle |
| Android | -target apk<br>of andere Android-pakketdoelen. |
| iOS | -target ipa-ad-hoc<br>of andere iOS-pakketdoelen. |
| iOS-simulator | -target ipa-test-interpreter-simulator<br>-target ipa-debug-interpreter-simulator |

## De extensiemap opgeven

Deel ADT via de ADT-optie `-extdir` mee in welke map de native extensies (ANE-bestanden) staan.

Zie “Bestands- en padopties” op pagina 190 voor informatie over deze optie.

## Ervoor zorgen dat het ANE-bestand het doelapparaat van de toepassing ondersteunt

Door middel van een ANE-bestand vertelt de ontwikkelaar van de native extensie u welke platforms door de extensie worden ondersteund. U kunt ook met een decomprimeringsprogramma de inhoud van het ANE-bestand bekijken. De gedecomprimeerde bestanden bevatten een map voor elk ondersteund platform.

Het is belangrijk te weten welke platforms door de extensie worden ondersteund wanneer u de toepassing die het ANE-bestand gebruikt in een pakket plaatst. Overweeg de volgende regels:

- Als u een Android-toepassingspakket maakt, dient het ANE-bestand het `Android-ARM`-platform te bevatten. Zo niet, dan dient het ANE-bestand het standaardplatform en minstens één ander platform te bevatten.
- Als u een iOS-toepassingspakket maakt, dient het ANE-bestand het `iPhone-ARM`-platform te bevatten. Zo niet, dan dient het ANE-bestand het standaardplatform en minstens één ander platform te bevatten.
- Als u een toepassingspakket voor een iOS-simulator maakt, moet het ANE-bestand het `iPhone-x86`-platform bevatten.
- Als u een Mac OS X-toepassingspakket maakt, dient het ANE-bestand het `MacOS-x86` -platform te bevatten. Zo niet, dan dient het ANE-bestand het standaardplatform en minstens één ander platform te bevatten.
- Als u een Windows-toepassingspakket maakt, dient het ANE-bestand het `Windows-x86`-platform te bevatten. Zo niet, dan dient het ANE-bestand het standaardplatform en minstens één ander platform te bevatten.

# Hoofdstuk 10: ActionScript-compilers

Voordat ActionScript- en MXML-code kan worden opgenomen in een AIR-toepassing, moet deze worden gecompileerd. Als u een Integrated Development Environment (IDE) gebruikt, zoals Adobe Flash Builder of Adobe Flash Professional, voert de IDE achter de schermen de compilatie uit. U kunt echter ook vanaf de opdrachtregel de ActionScript-compilers inschakelen om uw SWF-bestanden te maken wanneer u geen IDE gebruikt of wanneer u een build-script gebruikt.

## Over de AIR-opdrachtregelprogramma's in de SDK van Flex

Elk opdrachtregelprogramma dat u gebruikt om een Adobe AIR-toepassing te maken roept het bijbehorende programma aan dat wordt gebruikt voor het ontwikkelen van toepassingen:

- amxmlc roept mxmlc aan om toepassingsklassen te compileren
- acompc roept compc aan om bibliotheek- en componentklassen te compileren
- aasdoc roept asdoc aan om documentatiebestanden te genereren op basis van opmerkingen bij de broncode

Het enige verschil tussen de Flex- en AIR-versies van de hulpprogramma's is dat de AIR-versies de configuratieopties uit het bestand air-config.xml laden in plaats van uit het bestand flex-config.xml.

De SDK-programma's van Flex en de bijbehorende opdrachtregelopties worden uitvoerig beschreven in de Flex-documentatie. De SDK-programma's van Flex worden hier kort beschreven, zodat u aan de slag kunt en zodat u weet wat de verschillen zijn tussen het ontwikkelen van Flex-toepassingen en het ontwikkelen van AIR-toepassingen.

### Meer Help-onderwerpen

"Uw eerste AIR-bureaubladtoepassing met de Flex SDK maken" op pagina 39

## Compiler instellen

Gewoonlijk worden compilatieopties zowel op de opdrachtregel als met een of meer configuratiebestanden ingesteld. Het algemene configuratiebestand van SDK bevat standaardwaarden die worden gebruikt wanneer de compilers worden uitgevoerd. U kunt dit bestand aanpassen aan uw eigen ontwikkelomgeving. De map frameworks van de installatie van de SDK van Flex bevat twee algemene configuratiebestanden van Flex. Het bestand air-config.xml wordt gebruikt wanneer u de compiler amxmlc uitvoert. Dit bestand configureert de compiler voor AIR door de AIR-bibliotheken op te nemen. Het bestand flex-config.xml wordt gebruikt wanneer u mxmlc uitvoert.

De waarden van de standaardconfiguratie zijn geschikt om te ontdekken hoe Flex en AIR werken, maar wanneer u aan een echt project begint, moet u de beschikbare opties beter bestuderen. U kunt projectspecifieke waarden voor de compileropties instellen in een lokaal configuratiebestand dat voorrang krijgt boven de algemene waarden voor een bepaald project.

***Opmerking:*** *er zijn geen compilatieopties die specifiek voor AIR-toepassingen worden gebruikt, maar u moet naar de AIR-bibliotheken verwijzen wanneer u een AIR-toepassing compileert. Naar deze bibliotheken wordt meestal verwezen in een configuratiebestand op projectniveau, in een bestand voor een ontwikkelprogramma zoals Ant, of rechtstreeks op de opdrachtregel.*

# MXML- en ActionScript-bronbestanden voor AIR compileren

U kunt de Adobe® ActionScript® 3.0- en MXML-elementen van uw AIR-toepassing compileren met de opdrachtregelcompiler MXML (amxmlc). (Bij toepassingen op HTML-basis is compileren niet nodig. Als u een SWF in Flash Professional wilt compileren, hoeft u alleen de film naar een SWF-bestand te publiceren.)

Het basisopdrachtregelpatroon voor het gebruik van amxmlc is:

```
amxmlc [compiler options] -- MyAIRApp.mxml
```

waarbij *[compiler options]* de opdrachtregelopties bepaalt waarmee uw AIR-toepassing wordt gecompileerd.

De opdracht amxmlc roept de standaardcompiler mxmlc van Flex op met een extra parameter, `+configname=air`. Met deze parameter krijgt de compiler de opdracht om het bestand air-config.xml te gebruiken in plaats van het bestand flex-config.xml. Het gebruik van amxmlc is verder identiek aan het gebruik van mxmlc.

De compiler laadt het configuratiebestand air-config.xml waarin de AIR- en Flex-bibliotheken zijn opgegeven die gewoonlijk zijn vereist om een AIR-toepassing te compileren. U kunt ook een lokaal, projectgebonden configuratiebestand gebruiken om opties in de globale configuratie te overschrijven of daaraan toe te voegen. U kunt een lokaal configuratiebestand vaak het eenvoudigst maken door een kopie van het globale bestand te bewerken. U kunt het lokale bestand laden met de optie `-load-config`:

**-load-config=project-config.xml** Hiermee overschrijft u globale opties.

**-load-config+=project-config.xml** Hiermee voegt u extra waarden toe aan globale opties die meerdere waarden gebruiken, zoals de optie `-library-path`. Globale opties die slechts één waarde gebruiken, worden overschreven.

Als u een speciale naamgevingsconventie gebruikt voor het lokale configuratiebestand, laadt de compiler amxmlc het lokale bestand automatisch. Als het MXML-hoofdbestand bijvoorbeeld `RunningMan.mxml` is, geeft u het lokale configuratiebestand de volgende naam: `RunningMan-config.xml`. U hoeft nu alleen het volgende te typen om de toepassing te compileren:

```
amxmlc RunningMan.mxml
```

`RunningMan-config.xml` wordt automatisch geladen omdat de bestandsnaam overeenkomt met het gecompileerde MXML-bestand.

#### amxmlc, voorbeelden

In de volgende voorbeelden ziet u hoe u de compiler amxmlc gebruikt. (Alleen de ActionScript- en MXML-assets van uw toepassing hoeven te worden gecompileerd.)

Een AIR MXML-bestand compileren:

```
amxmlc myApp.mxml
```

De uitvoernaam compileren en instellen:

```
amxmlc –output anApp.swf -- myApp.mxml
```

Een AIR ActionScript-bestand compileren:

```
amxmlc myApp.as
```

Een configuratiebestand voor de compiler opgeven:

```
amxmlc –load-config config.xml -- myApp.mxml
```

Extra opties toevoegen uit een ander configuratiebestand:

```
amxmlc –load-config+=moreConfig.xml -- myApp.mxml
```

Bibliotheken toevoegen op de opdrachtregel (naast de bibliotheken die al in het configuratiebestand staan):

```
amxmlc –library-path+=/libs/libOne.swc,/libs/libTwo.swc  -- myApp.mxml
```

Een AIR MXML-bestand compileren zonder een configuratiebestand te gebruiken (Windows):

```
mxmlc -library-path [AIR SDK]/frameworks/libs/air/airframework.swc, ^
[AIR SDK]/frameworks/libs/air/airframework.swc, ^
-library-path [Flex SDK]/frameworks/libs/framework.swc ^
-- myApp.mxml
```

Een AIR MXML-bestand compileren zonder een configuratiebestand te gebruiken (Mac OS X of Linux):

```
mxmlc -library-path [AIR SDK]/frameworks/libs/air/airframework.swc, \
[AIR SDK]/frameworks/libs/air/airframework.swc, \
-library-path [Flex 3 SDK]/frameworks/libs/framework.swc \
-- myApp.mxml
```

Een AIR MXML-bestand compileren voor het gebruik van een gedeelde runtimebibliotheek:

```
amxmlc -external-library-path+=../lib/myLib.swc -runtime-shared-libraries=myrsl.swf --
myApp.mxml
```

Een AIR MXML-bestand compileren voor het gebruik van een ANE (gebruik `-external-library-path` voor ANE):

```
amxmlc -external-library-path+=../lib/myANE.ane -output=myAneApp.swf -- myAneApp.mxml
```

Compileren vanuit Java (waarbij `mxmlc.jar` is opgenomen in het klassepad):

```
java flex2.tools.Compiler +flexlib [Flex SDK 3]/frameworks +configname=air [additional
compiler options] -- myApp.mxml
```

Met de optie flexlib bepaalt u de locatie van de map frameworks van de SDK van Flex, zodat de compiler het bestand flex_config.xml kan vinden.

Compileren vanuit Java (waarbij het klassepad niet is ingesteld):

```
java -jar [Flex SDK 2]/lib/mxmlc.jar +flexlib [Flex SDK 3]/frameworks +configname=air
[additional compiler options] -- myApp.mxml
```

De compiler met Apache Ant aanroepen (in het voorbeeld wordt een Java-taak gebruik om mxmlc.jar uit te voeren):

```
<property name="SDK_HOME" value="C:/Flex46SDK"/>
<property name="MAIN_SOURCE_FILE" value="src/myApp.mxml"/>
<property name="DEBUG" value="true"/>
<target name="compile">
    <java jar="${MXMLC.JAR}" fork="true" failonerror="true">
        <arg value="-debug=${DEBUG}"/>
        <arg value="+flexlib=${SDK_HOME}/frameworks"/>
        <arg value="+configname=air"/>
        <arg value="-file-specs=${MAIN_SOURCE_FILE}"/>
    </java>
</target>
```

# AIR-componenten of codebibliotheken compileren (Flex)

Gebruik de componentencompiler acompc om AIR-bibliotheken en onafhankelijke componenten te compileren. De componentencompiler acompc werkt zoals de compiler amxmlc, met de volgende uitzonderingen:

- U moet opgeven welke klassen in de basiscode worden opgenomen in de bibliotheek of component.
- Acompc zoekt niet automatisch naar een lokaal configuratiebestand. Als u een projectgebonden configuratiebestand wilt gebruiken, moet u de optie –load-config gebruiken.

De opdracht acompc roept de standaard componentencompiler compc van Flex op, maar laadt de configuratieopties vanuit het bestand `air-config.xml` in plaats vanuit het bestand `flex-config.xml`.

## Configuratiebestand van componentencompiler

Gebruik een lokaal configuratiebestand om te zorgen dat u het bronpad en de klassennamen niet op de opdrachtregel hoeft te typen (en daarbij typefouten maakt). Voeg de optie -load-config aan de opdrachtregel van acompc toe om het lokale configuratiebestand te laden.

In het volgende voorbeeld ziet u een configuratie voor het bouwen van een bibliotheek met twee klassen, ParticleManager en Particle, die beide aanwezig zijn in het pakket: `com.adobe.samples.particles`. De klassebestanden bevinden zich in de map `source/com/adobe/samples/particles`.

```
<flex-config>
    <compiler>
        <source-path>
            <path-element>source</path-element>
        </source-path>
    </compiler>
    <include-classes>
        <class>com.adobe.samples.particles.ParticleManager</class>
        <class>com.adobe.samples.particles.Particle</class>
    </include-classes>
</flex-config>
```

Als u de bibliotheek wilt compileren met het configuratiebestand met de naam `ParticleLib-config.xml`, typt u:

```
acompc -load-config ParticleLib-config.xml -output ParticleLib.swc
```

Als u dezelfde opdracht volledig vanaf de opdrachtregel wilt uitvoeren, typt u:

```
acompc -source-path source -include-classes com.adobe.samples.particles.Particle
com.adobe.samples.particles.ParticleManager -output ParticleLib.swc
```

(Typ de volledige opdracht op één regel of gebruik het regelvervolgteken voor uw opdrachtshell.)

## Voorbeelden van acompc

In deze voorbeelden wordt aangenomen dat u een configuratiebestand met de naam `myLib-config.xml` gebruikt.

Een AIR-component of -bibliotheek compileren:

```
acompc -load-config myLib-config.xml -output lib/myLib.swc
```

Een gedeelde runtimebibliotheek compileren:

```
acompc -load-config myLib-config.xml -directory -output lib
```

(Let erop dat de map lib bestaat en leeg is voordat u de opdracht uitvoert.)

# Hoofdstuk 11: ADL (AIR Debug Launcher)

Gebruik ADL (AIR Debug Launcher) om zowel SWF- als HTML-toepassingen uit te voeren tijdens het ontwikkelen. Met ADL kunt u een toepassing uitvoeren zonder deze eerst in een pakket te plaatsen en te installeren. ADL maakt standaard gebruik van een runtime die in de SDK is opgenomen, zodat u de runtime niet afzonderlijk hoeft te installeren om ADL te gebruiken.

ADL drukt traceerinstructies en runtimefouten af naar de standaarduitvoer, maar ondersteunt geen onderbrekingspunten of andere foutopsporingsfuncties. U kunt de Flash Debugger (of een geïntegreerde ontwikkelomgeving zoals Flash Builder) gebruiken voor complexe foutopsporingsproblemen.

***Opmerking:*** *als uw* `trace()` *-instructies niet worden weergegeven op de console, controleert u of u* `ErrorReportingEnable` *of* `TraceOutputFileEnable` *niet hebt opgegeven in het mm.cfg-bestand. Raadpleeg Het mm.cfg-bestand bewerken voor meer informatie over de platformspecifieke locatie van dit bestand.*

AIR ondersteunt rechtstreekse foutopsporing, dus een foutopsporingsversie van de runtime is niet nodig (wat wel het geval is bij Adobe® Flash® Player). Als u foutopsporing vanaf de opdrachtregel wilt uitvoeren, gebruikt u de Flash Debugger en ADL (AIR Debug Launcher).

De Flash Debugger staat in de directory van de SDK van Flex. Native versies, zoals fdb.exe in Windows, staan in de subdirectory bin. De Java-versie staat in de subdirectory lib. Het bestand adl.exe (AIR Debug Launcher) staat in de directory bin van de installatiemap van de SDK van Flex. (Er is geen aparte Java-versie).

***Opmerking:*** *het is niet mogelijk om een AIR-toepassing rechtstreeks met fdb te starten, omdat fdb de toepassing in Flash Player probeert te openen. In plaats daarvan moet u de AIR-toepassing verbinding laten maken met een actieve fdb-sessie.*

## Gebruik van ADL

Met het volgende patroon kunt u een toepassing met ADL uitvoeren:

```
adl application.xml
```

Hierbij is *application.xml* het toepassingsdescriptorbestand voor de toepassing.

De volledige syntaxis voor de ADL is:

```
adl [-runtime runtime-directory]
    [-pubid publisher-id]
    [-nodebug]
    [-atlogin]
    [-profile profileName]
    [-screensize value]
    [-extdir extension-directory]
    application.xml
    [root-directory]
    [-- arguments]
```

(Items tussen haakjes, [], zijn optioneel.)

**-runtime *runtime-directory*** Hiermee geeft u de map op die de runtime bevat die u wilt gebruiken. Als u deze niet opgeeft, wordt de runtimemap in dezelfde SDK als het ADL-programma gebruikt. Als u ADL uit de SDK-map verplaatst, moet u de runtimemap opgeven. In Windows en Linux geeft u de map op die de map van `Adobe AIR` bevat. In Mac OS X geeft u de map op die `Adobe AIR.framework` bevat.

***-pubid publisher-id* Hiermee wijst u de opgegeven waarde toe als de uitgevers-id van de AIR-toepassing voor deze uitvoering.** Als u een tijdelijke uitgevers-id opgeeft, kunt u functies van een AIR-toepassing testen, zoals de communicatie via een lokale verbinding, die de uitgevers-id gebruiken om een toepassing uniek te identificeren. Vanaf AIR 1.5.3 kunt u de uitgevers-id opgeven in het descriptorbestand van de toepassing (en hoeft u deze parameter niet meer te gebruiken).

***Opmerking:*** *vanaf AIR 1.5.3 wordt een uitgevers-id niet meer automatisch berekend en toegewezen aan een AIR-toepassing. U kunt een uitgevers-id opgeven wanneer u een update maakt voor een bestaande AIR-toepassing. Voor nieuwe toepassingen is de uitgevers-id echter niet nodig en moet deze dan ook niet worden opgegeven.*

**-nodebug** Hiermee schakelt u ondersteuning voor foutopsporing uit. Als u deze optie gebruikt, kan het toepassingsproces geen verbinding maken met Flash Debugger en worden dialoogvensters voor niet-verwerkte uitzonderingen onderdrukt. (Traceerinstructies worden wel in het consolevenster weergegeven.) Als u foutopsporing uitschakelt, kan uw toepassing iets sneller werken en wordt de uitvoeringsmodus van een geïnstalleerde toepassing beter geëmuleerd.

-**atlogin** Simuleert dat de toepassing bij aanmelding wordt gestart. Met deze vlag kunt u de logica van een toepassing testen die alleen wordt uitgevoerd wanneer de toepassing is geconfigureerd om te worden gestart wanneer de gebruiker zich aanmeldt. Wanneer u `-atlogin` gebruikt, wordt de eigenschap `reason` van het InvokeEvent-object dat naar de toepassing is verzonden ingesteld op `login` in plaats van `standard` (tenzij de toepassing al wordt uitgevoerd).

***-profiel profileName* ADL spoort fouten op bij de toepassing die het opgegeven profiel gebruikt.** De *`profileName`* kan een van de volgende waarden hebben:

- desktop
- extendedDesktop
- mobileDevice

Als de toepassingsdescriptor een element `supportedProfiles` bevat, moet het profiel dat u opgeeft met `-profile`, onderdeel uitmaken van de lijst met ondersteunde elementen. Als de markering `-profile` niet wordt gebruikt, wordt het eerste profiel in de toepassingsdescriptor gebruikt als het actieve profiel. Als de toepassingsdescriptor het element `supportedProfiles` niet bevat en u gebruikt de markering `-profile` niet, dan wordt het profiel *desktop* gebruikt.

Zie “supportedProfiles” op pagina 251 en “Apparaatprofielen” op pagina 257 voor meer informatie.

***waarde* -screensize** De gesimuleerde schermgrootte die wordt gebruikt voor het uitvoeren van toepassingen in het profiel mobileDevice op het bureaublad. Geef de schermgrootte op als vooraf gedefinieerd schermtype of als de pixelafmetingen van de normale breedte en hoogte bij een staande afdrukstand, plus de breedte en hoogte voor een volledige-schermweergave. Als u de waarde als type wilt opgeven, gebruikt u een van de volgende vooraf gedefinieerde schermtypen:

| Schermtype | Normale breedte x hoogte | Breedte x hoogte volledig scherm |
|---|---|---|
| 480 | 720 x 480 | 720 x 480 |
| 720 | 1280 x 720 | 1280 x 720 |
| 1080 | 1920 x 1080 | 1920 x 1080 |
| Droid | 480 x 816 | 480 x 854 |

| Schermtype | Normale breedte x hoogte | Breedte x hoogte volledig scherm |
|---|---|---|
| FWQVGA | 240 x 432 | 240 x 432 |
| FWVGA | 480 x 854 | 480 x 854 |
| HVGA | 320 x 480 | 320 x 480 |
| iPad | 768 x 1004 | 768 x 1024 |
| iPadRetina | 1536 x 2008 | 1536 x 2048 |
| iPhone | 320 x 460 | 320 x 480 |
| iPhoneRetina | 640 x 920 | 640 x 960 |
| iPhone5Retina | 640 x 1096 | 640 x 1136 |
| iPhone6 | 750 x 1294 | 750 x 1334 |
| iPhone6Plus | 1242 x 2148 | 1242 x 2208 |
| iPod | 320 x 460 | 320 x 480 |
| iPodRetina | 640 x 920 | 640 x 960 |
| iPod5Retina | 640 x 1096 | 640 x 1136 |
| NexusOne | 480 x 762 | 480 x 800 |
| QVGA | 240 x 320 | 240 x 320 |
| SamsungGalaxyS | 480 x 762 | 480 x 800 |
| SamsungGalaxyTab | 600 x 986 | 600 x 1024 |
| WQVGA | 240 x 400 | 240 x 400 |
| WVGA | 480 x 800 | 480 x 800 |

Gebruik de volgende indeling om de pixelafmetingen van het scherm direct op te geven:

```
widthXheight:fullscreenWidthXfullscreenHeight
```

Geef altijd de pixeldimensies op voor een staande afdrukstand. Dit houdt in dat de breedte een kleinere waarde heeft dan de hoogte. U kunt bijvoorbeeld het NexusOne-scherm opgeven met:

```
-screensize 480x762:480x800
```

**-extdir *extension-directory*** De map waarin de runtime moet zoeken naar native extensies. De map bevat een submap voor elke native extensie die de toepassing gebruikt. Al deze submappen bevatten het *niet verpakte* ANE-bestand van een extensie. Bijvoorbeeld:

```
C:\extensionDirs\
    extension1.ane\
        META-INF\
            ANE\
                Android-ARM\
                    library.swf
                    extension1.jar
                extension.xml
            signatures.xml
        catalog.xml
        library.swf
        mimetype
    extension2.ane\
        META-INF\
            ANE\
                Android-ARM\
                    library.swf
                    extension2.jar
                extension.xml
            signatures.xml
        catalog.xml
        library.swf
        mimetype
```

Neem het volgende in overweging wanneer u de parameter -extdir gebruikt:

- De ADL-opdracht vereist dat elke opgegeven map de bestandsnaamtoevoeging .ane heeft. Het gedeelte van de bestandsnaam dat aan het achtervoegsel ".ane" voorafgaat, kan echter elke geldige bestandsnaam zijn. Deze naam hoeft *niet* overeen te komen met de waarde van het element `extensionID` van het descriptorbestand van de toepassing.
- U kunt de parameter `-extdir` meerdere malen opgeven.
- Het gebruik van de parameter `-extdir` vanuit de ADT-tool en de ADL-tool is niet hetzelfde. In ADT geeft de parameter een map op die ANE-bestanden bevat.
- U kunt ook de omgevingsvariabele `AIR_EXTENSION_PATH` gebruiken om de extensiemappen op te geven. Zie “ADT-omgevingsvariabelen” op pagina 197.

***application.xml*** Het toepassingsdescriptorbestand. Zie “AIR-toepassingsdescriptorbestanden” op pagina 215. De toepassingsdescriptor is de enige parameter die door ADL wordt vereist en meestal ook de enige benodigde parameter.

***root-directory*** Hiermee geeft u de hoofdmap op van de toepassing die wordt uitgevoerd. Als u deze optie niet opgeeft, wordt de map gebruikt die het descriptorbestand van de toepassing bevat.

-- ***arguments*** Alle tekenreeksen na "--" worden als opdrachtregelargumenten doorgegeven aan de toepassing.

***Opmerking:*** *wanneer u een AIR-toepassing start die al wordt uitgevoerd, wordt geen nieuwe instantie van die toepassing gestart. In plaats daarvan wordt de gebeurtenis* `invoke` *verzonden naar de gestarte instantie.*

# Voorbeelden van ADL

Een toepassing uitvoeren in de huidige map:

```
adl myApp-app.xml
```

Een toepassing uitvoeren in een submap van de huidige map:

```
adl source/myApp-app.xml release
```

Een toepassing uitvoeren en twee opdrachtregelargumenten, "tik" en "tak", doorgeven:

```
adl myApp-app.xml -- tick tock
```

Een toepassing uitvoeren met een bepaalde runtime:

```
adl -runtime /AIRSDK/runtime myApp-app.xml
```

Een oplossing uitvoeren zonder ondersteuning van het foutopsporingsprogramma:

```
adl -nodebug myApp-app.xml
```

Voer een toepassing in het profiel voor mobiele apparaten uit en simuleer de schermgrootte van de Nexus One:

```
adl -profile mobileDevice -screensize NexusOne myMobileApp-app.xml
```

Voer een toepassing uit met Apache Ant om de toepassing uit te voeren (de paden in het voorbeeld zijn voor Windows):

```
<property name="SDK_HOME" value="C:/AIRSDK"/>
<property name="ADL" value="${SDK_HOME}/bin/adl.exe"/>
<property name="APP_ROOT" value="c:/dev/MyApp/bin-debug"/>
<property name="APP_DESCRIPTOR" value="${APP_ROOT}/myApp-app.xml"/>

<target name="test">
    <exec executable="${ADL}">
        <arg value="${APP_DESCRIPTOR}"/>
        <arg value="${APP_ROOT}"/>
    </exec>
</target>
```

# Afsluit- en foutcodes van ADL

In de volgende tabel worden de afsluitcodes beschreven die ADL afdrukt:

| Afsluitcode | Beschrijving |
|---|---|
| 0 | Gestart. ADL wordt afgesloten nadat de AIR-toepassing is afgesloten. |
| 1 | Er is een al gestarte AIR-toepassing opgeroepen. ADL wordt onmiddellijk afgesloten. |
| 2 | Gebruiksfout. Er zijn onjuiste argumenten opgegeven voor ADL. |
| 3 | De runtime is niet gevonden. |
| 4 | De runtime kan niet worden gestart. Dit gebeurt vaak wanneer de versie die is opgegeven in de toepassing niet overeenkomt met de versie van de runtime. |
| 5 | Er is een fout met onbekende oorzaak opgetreden. |
| 6 | Het descriptorbestand van de toepassing is niet gevonden. |
| 7 | De inhoud van de descriptor van de toepassing is niet geldig. Deze fout geeft vaak aan dat de XML-code niet juist is opgebouwd. |
| 8 | Het hoofdbestand met de toepassingsinhoud (opgegeven in het element <content> van het descriptorbestand van de toepassing) is niet gevonden. |

| Afsluitcode | Beschrijving |
|---|---|
| 9 | Het hoofdbestand met de toepassingsinhoud is geen geldig SWF- of HTML-bestand. |
| 10 | De toepassing biedt geen ondersteuning voor het profiel dat is opgegeven met de optie -profile. |
| 11 | Het argument -screensize wordt niet ondersteund door het huidige profiel. |

# Hoofdstuk 12: AIR Developer Tool (ADT)

De AIR Developer Tool (ADT) is een multifunctioneel opdrachtregelprogramma voor het ontwikkelen van AIR-toepassingen. U kunt ADT gebruiken om de volgende taken uit te voeren:

- Een AIR-toepassing verpakken als een .air-installatiebestand
- Een AIR-toepassing verpakken als een native installatieprogramma. Bijvoorbeeld als een .exe-installatiebestand voor Windows, .ipa voor iOS of .apk voor Android
- Een native extensie als een ANE-bestand (AIR Native Extension) verpakken
- Een AIR-toepassing met een digitaal certificaat ondertekenen
- De digitale handtekening wijzigen (migreren) die wordt gebruikt voor toepassingsupdates
- De apparaten bepalen die op een computer zijn aangesloten
- Een zelfondertekend ondertekeningscertificaat voor de digitale code maken
- Een toepassing op afstand op een mobiel apparaat installeren, opstarten en verwijderen
- De AIR-runtime op afstand op een mobiel apparaat installeren en verwijderen

ADT is een Java-programma dat is opgenomen in de AIR-SDK. U moet over Java 1.5 of hoger beschikken om het te kunnen gebruiken. De SDK bevat een scriptbestand voor het oproepen van ADT. Als u dit script wilt gebruiken, moet de locatie van het Java-programma zijn opgenomen in de omgevingsvariabele van het pad. Als de map `bin` ook in de omgevingsvariabele van het pad wordt weergegeven, kunt u in de opdrachtregel `adt` typen, met de correcte argumenten, om ADT op te roepen. (Als u niet weet hoe u de omgevingsvariabele van het pad moet instellen, kunt u de documentatie van uw besturingssysteem raadplegen. Ter ondersteuning worden de procedures voor het instellen van het pad op de meeste computersystemen beschreven in “Omgevingsvariabelen van het pad” op pagina 318.)

Er is minstens 2 GB computergeheugen vereist om ADT te kunnen gebruiken. Als u over minder geheugen beschikt, kan het geheugen van ADT opraken, vooral wanneer toepassingen voor iOS worden verpakt.

Aannemende dat zowel Java als de binmap van de AIR-SDK in de padvariabele zijn opgenomen, kunt u ADT uitvoeren met behulp van de volgende basissyntaxis:

```
adt -command options
```

***Opmerking:*** *de meeste geïntegreerde ontwikkelomgevingen, zoals Adobe Flash Builder en Adobe Flash Professional, kunnen AIR-toepassingen verpakken en ondertekenen. In het algemeen hoeft u voor deze veelvoorkomende taken ADT niet te gebruiken wanneer u al een dergelijke ontwikkelomgeving gebruikt. Het is echter wel mogelijk dat u ADT moet gebruiken voor functies die niet worden ondersteund door de geïntegreerde ontwikkelomgeving. Daarnaast kunt u ADT gebruiken als een opdrachtregelprogramma als onderdeel van een geautomatiseerd buildproces.*

## ATD-opdrachten

In het eerste argument aan ADT wordt een van de volgende opdrachten opgegeven.

- -package — hiermee wordt een AIR-toepassing of een ANE (AIR Native Extension) verpakt.
- -prepare — hiermee wordt een AIR-toepassing als tussentijds bestand verpakt (AIRI) maar niet ondertekend. AIRI-bestanden kunnen niet worden geïnstalleerd.

- -sign — hiermee wordt een AIRI-pakket ondertekend dat is gemaakt met behulp van de opdracht -prepare. Met de opdrachten -prepare en -sign kunnen verpakken en ondertekenen op verschillende tijdstippen worden uitgevoerd. U kunt de opdracht -sign ook gebruiken om een ANE-pakket te ondertekenen of opnieuw te ondertekenen.
- -migrate — hiermee wordt een migratiehandtekening toegepast op een ondertekend AIR-pakket waardoor u een nieuw of vernieuwd certificaat voor ondertekenen van code kunt gebruiken.
- -certificate — hiermee wordt een zelfondertekend certificaat voor ondertekenen van digitale code gemaakt.
- -checkstore — hiermee wordt gecontroleerd of er toegang is tot een digitaal certificaat in een sleutelarchief.
- -installApp — hiermee wordt een AIR-toepassing op een apparaat of op een apparaatemulator geïnstalleerd.
- -launchApp — hiermee wordt een AIR-toepassing op een apparaat of op een apparaatemulator gestart.
- -appVersion — hiermee wordt de versie weergegeven van een AIR-toepassing die momenteel op een apparaat of een apparaatemulator is geïnstalleerd.
- -uninstallApp — hiermee wordt een AIR-toepassing van een apparaat of een apparaatemulator verwijderd.
- -installRuntime — hiermee wordt de AIR-runtime op een apparaat of een apparaatemulator geïnstalleerd.
- -runtimeVersion — hiermee wordt de versie weergegeven van de AIR-runtime die momenteel op een apparaat of een apparaatemulator is geïnstalleerd.
- -uninstallRuntime — hiermee wordt de AIR-runtime verwijderd die momenteel op een apparaat of een apparaatemulator is geïnstalleerd.
- -version — hiermee wordt het ADT-versienummer weergegeven.
- -devices: rapporteert apparaatinformatie voor aangesloten mobiele apparaten of emulators.
- -help — hiermee wordt de lijst met opdrachten en opties weergegeven.

Veel ADT-opdrachten delen gerelateerde sets optiemarkeringen en parameters. Deze sets van opties worden afzonderlijk gedetailleerd beschreven:

- "ADT-opties voor codeondertekening" op pagina 188
- "Bestands- en padopties" op pagina 190
- "Verbindingsopties voor foutopsporing" op pagina 191
- "Opties voor native extensies" op pagina 192

## ADT-opdracht voor verpakken

De opdracht `-package` moet worden uitgevoerd vanuit de hoofdmap van de toepassing. De opdracht gebruikt de volgende syntaxis:

Een AIR-pakket maken van het toepassingsbestand van het onderdeel:

```
adt -package
    AIR_SIGNING_OPTIONS
    -target packageType
    -sampler
    -hideAneLibSymbols
    NATIVE_SIGNING_OPTIONS
    output
    app_descriptor
    FILE_OPTIONS
```

Een native pakket maken van het toepassingsbestand van het onderdeel:

```
adt -package
    AIR_SIGNING_OPTIONS
    -target packageType
    DEBUGGER_CONNECTION_OPTIONS
    -airDownloadURL URL
    NATIVE_SIGNING_OPTIONS
    output
    app_descriptor
    -platformsdk path
    FILE_OPTIONS
```

Maak een native pakket dat een native extensie van de toepassingsbestanden van het onderdeel bevat:

```
adt -package
    AIR_SIGNING_OPTIONS
    -migrate MIGRATION_SIGNING_OPTIONS
    -target packageType
    DEBUGGER_CONNECTION_OPTIONS
    -airDownloadURL URL
    NATIVE_SIGNING_OPTIONS
    output
    app_descriptor
    -platformsdk path
    FILE_OPTIONS
```

Een native pakket maken van een AIR- of AIRI-bestand:

```
adt -package
    -target packageType
    NATIVE_SIGNING_OPTIONS
    output
    input_package
```

Een native extensiepakket maken op basis van de native extensiebestanden van de component:

```
adt -package
    AIR_SIGNING_OPTIONS
    -target ane
    output
    ANE_OPTIONS
```

***Opmerking:*** *het is niet nodig een ANE-bestand te ondertekenen, de* `AIR_SIGNING_OPTIONS`*-parameters in dit voorbeeld zijn dus optioneel.*

**AIR_SIGNING_OPTIONS** In de AIR-ondertekeningsopties wordt het certificaat weergegeven waarmee een AIR-installatie kan worden ondertekend. De ondertekeningsopties worden volledig beschreven in "ADT-opties voor codeondertekening" op pagina 188.

**-migrate** Deze markering geeft op dat de toepassing is ondertekend met een migratiecertificaat als aanvulling op het certificaat dat is opgegeven in de `AIR_SIGNING_OPTIONS`-parameters. Deze markering is alleen geldig als u een bureaubladtoepassing als native installatieprogramma in een pakket plaatst en als de toepassing een native extensie gebruikt. In andere gevallen treedt er een fout op. De ondertekeningsopties voor het migratiecertificaat worden opgegeven als de **MIGRATION_SIGNING_OPTIONS**-parameters. Deze ondertekeningsopties worden volledig beschreven in "ADT-opties voor codeondertekening" op pagina 188. Door de markering `-migrate` te gebruiken, kunt u een update maken voor een bureaubladtoepassing als native installatieprogramma die een native extensie gebruikt en het certificaat voor codeondertekening voor de toepassing wijzigen, zoals wanneer het oorspronkelijke certificaat verloopt. Zie "Een bijgewerkte versie van een AIR-toepassing ondertekenen." op pagina 209 voor meer informatie.

De markering `-migrate` van de opdracht `-package` is beschikbaar in AIR 3.6 en later.

**-target** Het type pakket dat wordt gemaakt. De ondersteunde pakkettypen zijn:

- air — een AIR-pakket. "air" is de standaardwaarde en de markering -target hoeft niet te worden opgegeven bij het maken van AIR- of AIRI-bestanden.
- airn — een native toepassingspakket voor apparaten in het uitgebreide televisieprofiel.
- ane —een native AIR-extensiepakket
- Doelstellingen van het Android-pakket:
  - apk — een Android-pakket. Een pakket dat is gemaakt met dit doel, kan alleen worden geïnstalleerd op een Android-apparaat, niet op een emulator.
  - apk-captive-runtime — een Android-pakket met zowel de toepassing als een captive versie van de AIR-runtime. Een pakket dat is gemaakt met dit doel, kan alleen worden geïnstalleerd op een Android-apparaat, niet op een emulator.
  - apk-debug — een Android-pakket met extra informatie over foutopsporing. (De SWF-bestanden in de toepassing moeten ook worden gecompileerd met ondersteuning voor foutopsporing.)
  - apk-emulator — een Android-pakket voor gebruik op een emulator zonder ondersteuning voor foutopsporing. (Gebruik het doel apk-debug om foutopsporing toe te staan op emulators en apparaten.)
  - apk-profile — een Android-pakket dat de prestatie van de toepassing en het instellen van profielen voor geheugens ondersteunt.
- Doelstellingen van het iOS-pakket:
  - ipa-ad-hoc — een iOS-pakket voor ad-hocdistributie.
  - ipa-app-store — een iOS-pakket voor distributie van Apple App store.
  - ipa-debug — een iOS-pakket met extra informatie over foutopsporing. (De SWF-bestanden in de toepassing moeten ook worden gecompileerd met ondersteuning voor foutopsporing.)
  - ipa-test — een iOS-pakket gecompileerd zonder informatie over optimalisatie of foutopsporing.
  - ipa-debug-interpreter: in functionaliteit equivalent aan een foutopsporingspakket, maar met snellere compilatiemogelijkheden. De ActionScript-bytecode wordt echter geïnterpreteerd en niet vertaald naar computercode. Als gevolg hiervan wordt de code in een interpreter-pakket langzamer uitgevoerd.
  - ipa-debug-interpreter-simulator: in functionaliteit equivalent aan ipa-debug-interpreter, maar ingepakt voor de iOS-simulator. Alleen Macintosh. Als u deze optie gebruikt, moet u ook de optie -platformsdk opnemen om het pad naar de iOS-simulator-SDK aan te duiden.
  - ipa-test-interpreter: in functionaliteit equivalent aan een testpakket, maar met snellere compilatiemogelijkheden. De ActionScript-bytecode wordt echter geïnterpreteerd en niet vertaald naar computercode. Als gevolg hiervan wordt de code in een interpreter-pakket langzamer uitgevoerd.
  - ipa-test-interpreter-simulator: in functionaliteit equivalent aan ipa-test-interpreter, maar ingepakt voor de iOS-simulator. Alleen Macintosh. Als u deze optie gebruikt, moet u ook de optie -platformsdk opnemen om het pad naar de iOS-simulator-SDK aan te duiden.
- native — een native desktopinstallatieprogramma Het gemaakte bestandstype is de native installatie-indeling van het besturingssysteem waarop de opdracht is uitgevoerd:
  - EXE — Windows
  - DMG — Mac
  - DEB — Ubuntu Linux (AIR 2.6 of eerder)
  - RPM — Fedora of OpenSuse Linux (AIR 2.6 of eerder)

Zie "Een eigen bureaubladinstallatieprogramma verpakken" op pagina 60 voor meer informatie.

**-sampler** (alleen iOS, AIR 3.4 of hoger) Hiermee wordt de ActionScript-sampler op telemetriebasis ingeschakeld in iOS-toepassingen. Met deze markering kunt u een toepassingsprofiel instellen met Adobe Scout. Alhoewel u met Scout een profiel kunt instellen voor elke inhoud op een Flash-platform, krijgt u door de gedetailleerde telemetrie in te schakelen een beter inzicht in de timing van ActionScript-functies, DisplayList, Stage3D-rendering en andere aspecten. Opmerking: door deze markering in te schakelen worden de prestaties in geringe mate beïnvloed. Gebruik deze optie dus niet voor productietoepassingen.

**-hideAneLibSymbols** (alleen iOS, AIR 3.4 of hoger) Ontwikkelaars van toepassingen kunnen meerdere native extensies van meerdere bronnen gebruiken. Als de ANE's een algemene symboolnaam delen, genereert ADT de fout 'Duplicate symbol in object file' (Dubbel symbool in objectbestand). In sommige gevallen veroorzaakt deze fout een crash tijdens runtime. Met de optie `hideAneLibSymbols` kunt u aangegeven of de symbolen van de ANE-bibliotheek alleen zichtbaar zijn voor de bronnen van de desbetreffende bibliotheek (yes) of dat ze algemeen zichtbaar zijn (no):

- **yes** — Verbergt ANE-symbolen en lost daardoor eventuele onbedoelde symboolconflicten op.
- **no** — (Standaard) ANE-symbolen worden niet verborgen. Dit is het standaardgedrag bij AIR-versies die voorafgaan aan AIR 3.4.

**-embedBitcode** (alleen iOS, AIR 25 of hoger) App-ontwikkelaars kunnen met de optie `embedBitcode` aangeven of er al dan niet bitcode moet worden ingesloten in hun iOS-toepassing door 'ja' of 'nee' op te geven. De standaardwaarde van deze schakeloptie is 'no' als deze niet is opgegeven.

**DEBUGGER_CONNECTION_OPTIONS** De verbindingsopties voor foutopsporing bepalen of een foutopsporingspakket moet proberen te verbinden met een extern foutopsporingsprogramma dat wordt uitgevoerd op een andere computer of moet luisteren naar een verbinding met een extern foutopsporingsprogramma. Deze opties worden alleen ondersteund voor mobiele foutopsporingspakketten (gericht op apk-debug en ipa-debug). Deze opties worden beschreven in "Verbindingsopties voor foutopsporing" op pagina 191.

**-airDownloadURL** Geeft een alternatieve URL op voor het downloaden en installeren van de AIR-runtime op Android-apparaten. Als hier niets wordt opgegeven, leidt een AIR-toepassing de gebruiker om naar de AIR-runtime op Android Market als de runtime nog niet is geïnstalleerd.

Als uw toepassing via een andere markt (dan de door Google beheerde Android Market) wordt gedistribueerd, dient u wellicht de URL op te geven voor het downloaden van de AIR-runtime van de desbetreffende markt. Bepaalde alternatieve markten staan toepassingen niet toe een download van een externe markt te vereisen. Deze optie wordt alleen ondersteund voor Android-pakketten.

**NATIVE_SIGNING_OPTIONS** In de native ondertekeningsopties wordt het certificaat weergegeven dat wordt gebruikt om een native pakketbestand te ondertekenen. Deze ondertekeningsopties worden gebruikt om een handtekening toe te passen die wordt gebruikt door het native besturingssysteem, niet door de AIR-runtime. De opties zijn overigens identiek aan de AIR_SIGNING_OPTIONS en worden volledig beschreven in "ADT-opties voor codeondertekening" op pagina 188.

Native ondertekening wordt ondersteund door Windows en Android. Bij Windows dienen de AIR-ondertekeningsopties en de native ondertekeningsopties te worden opgegeven. Bij Android kunnen alleen de native ondertekeningsopties worden opgegeven.

In veel gevallen kunt u hetzelfde certificaat voor het ondertekenen van code gebruiken voor een native ondertekening en een AIR-ondertekening. Dit geldt echter niet in alle gevallen. In het Google-beleid voor toepassingen die zijn ingediend bij de Android Market wordt voorgeschreven dat alle toepassingen moeten worden ondertekend met een certificaat dat ten minste geldig is tot het jaar 2033. Dit betekent dat een certificaat dat door een bekende certificeringsinstantie wordt uitgegeven, wat wordt aanbevolen bij het toepassen van een AIR-ondertekening, niet mag worden gebruikt voor ondertekening van een Android-toepassing. (Er bestaan geen certificeringsinstanties die een certificaat voor het ondertekenen van code uitgeven met een geldigheidsperiode die zo lang duurt.)

**output** De naam van het te maken pakketbestand. Het opgeven van de bestandsextensie is optioneel. Als de extensie niet wordt opgegeven, wordt een extensie toegevoegd die geschikt is voor de waarde -target en het huidige besturingssysteem.

**app_descriptor** Het pad naar het descriptorbestand van de toepassing. Dit kan een relatief pad ten opzichte van de huidige map of een absoluut pad zijn. (De naam van het descriptorbestand van de toepassing wordt in het AIR-bestand gewijzigd in *application.xml*.)

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat:

- Android: de AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)
- iOS: de AIR-SDK wordt geleverd met een captive iOS-SDK. Met de optie -platformsdk kunt u toepassingen inpakken met een extern SDK, zodat u niet beperkt bent tot het gebruik van de captive iOS-SDK. Als u bijvoorbeeld een uitbreiding hebt gemaakt met de nieuwste iOS-SDK, kunt u die SDK opgeven bij het inpakken van uw toepassing. Bovendien geldt dat wanneer u ADT gebruikt met de iOS-simulator, u altijd de optie -platformsdk moet opnemen om het pad naar de iOS-simulator-SDK aan te duiden.

**-arch**Ontwikkelaars van apps kunnen dit argument gebruiken om APK voor x86-platforms te maken. Er zijn de volgende waarden:

- armv7- ADT verpakt APK voor het Android armv7-platform.
- x86 - ADT verpakt APK voor het Android x86-platform.

armv7 is de standaardwaarde wanneer geen waarde is opgegeven

**FILE_OPTIONS** Hiermee worden de toepassingsbestanden weergegeven die in het pakket moeten worden opgenomen. De bestandsopties worden volledig beschreven in “Bestands- en padopties” op pagina 190. U moet geen bestandsopties opgeven wanneer u een native pakket van een AIR- of AIRI-bestand maakt.

**input_airi** Opgeven als een native pakket van een AIRI-bestand wordt gemaakt. De AIR_SIGNING_OPTIONS worden vereist als het doel *air* is (of als geen doel is opgegeven).

**input_air** Opgeven als een native pakket van een AIR-bestand wordt gemaakt. AIR_SIGNING_OPTIONS niet opgeven.

**ANE_OPTIONS** Hiermee worden de opties en bestanden geïdentificeerd waarmee een native extensiepakket wordt gemaakt. De extensiepakketopties worden volledig beschreven in “Opties voor native extensies” op pagina 192.

## Voorbeelden van de ADT-opdracht -package

Specifieke toepassingsbestanden in de huidige directory verpakken voor een AIR-toepassing op SWF-basis:

```
adt –package -storetype pkcs12 -keystore cert.p12 myApp.air myApp.xml myApp.swf components.swc
```

Specifieke toepassingsbestanden in de huidige directory verpakken voor een AIR-toepassing op HTML-basis:

```
adt –package -storetype pkcs12 -keystore cert.p12 myApp.air myApp.xml myApp.html AIRAliases.js
image.gif
```

Alle bestanden en submappen in de huidige werkmap in een pakket plaatsen:

```
adt –package -storetype pkcs12 -keystore ../cert.p12 myApp.air myApp.xml .
```

***Opmerking:*** *het sleutelarchiefbestand bevat de persoonlijke sleutel waarmee uw toepassing wordt ondertekend. Neem het handtekeningcertificaat nooit op in het AIR-pakket! Als u jokertekens gebruikt in de ADT-opdracht, plaatst u het sleutelarchiefbestand op een andere locatie, zodat het niet wordt opgenomen in het pakket. In dit voorbeeld bevindt het sleutelarchiefbestand, cert.p12, zich in de bovenliggende map.*

Alleen de hoofdbestanden en een submap met afbeeldingen in een pakket plaatsen:

```
adt –package -storetype pkcs12 -keystore cert.p12 myApp.air myApp.xml myApp.swf images
```

Een HTML-toepassing en alle bestanden in de submappen met HTML, scripts en afbeeldingen in een pakket plaatsen:

```
adt –package -storetype pkcs12 -keystore cert.p12 myApp.air myApp.xml index.html AIRALiases.js
html scripts images
```

Het bestand application.xml en het SWF-hoofdbestand dat in een werkmap (release/bin) staat in een pakket plaatsen:

```
adt –package -storetype pkcs12 -keystore cert.p12 myApp.air release/bin/myApp.xml –C
release/bin myApp.swf
```

Assets vanuit meerdere plaatsen in uw gebouwde bestandssysteem in een pakket plaatsen. In dit voorbeeld bevinden de assets van de toepassing zich in de volgende mappen voordat ze in een pakket worden geplaatst:

```
/devRoot
    /myApp
        /release
            /bin
                myApp-app.xml
                myApp.swf or myApp.html
    /artwork
        /myApp
            /images
                image-1.png
                ...
                image-n.png
    /libraries
        /release
            /libs
                lib-1.swf
                lib-2.swf
                lib-a.js
                AIRAliases.js
```

De volgende ADT-opdracht uitvoeren vanuit de map /devRoot/myApp:

```
adt –package -storetype pkcs12 -keystore cert.p12 myApp.air release/bin/myApp-app.xml
    –C release/bin myApp.swf (or myApp.html)
    –C ../artwork/myApp images
    –C ../libraries/release libs
```

Resultaten in de volgende pakketstructuur:

```
/myAppRoot
    /META-INF
        /AIR
            application.xml
            hash
    myApp.swf or myApp.html
    mimetype
    /images
        image-1.png
        ...
        image-n.png
    /libs
        lib-1.swf
        lib-2.swf
        lib-a.js
        AIRAliases.js
```

ADT uitvoeren als een Java-programma voor een eenvoudige toepassing op SWF-basis (waarbij het klassepad niet wordt ingesteld):

```
java –jar {AIRSDK}/lib/ADT.jar –package -storetype pkcs12 -keystore cert.p12 myApp.air
myApp.xml myApp.swf
```

ADT uitvoeren als een Java-programma voor een eenvoudige toepassing op HTML-basis (waarbij het klassepad niet wordt ingesteld):

```
java –jar {AIRSDK}/lib/ADT.jar –package -storetype pkcs12 -keystore cert.p12 myApp.air
myApp.xml myApp.html AIRAliases.js
```

ADT uitvoeren als een Java-programma (waarbij het pakket ADT.jar is opgenomen in het Java-klassepad):

```
java -com.adobe.air.ADT –package -storetype pkcs12 -keystore cert.p12 myApp.air myApp.xml
myApp.swf
```

ADT uitvoeren als een Java-taak in Apache Ant (alhoewel u de ADT-opdracht beter rechtstreeks kunt toepassen in Ant-scripts). De paden in het voorbeeld gelden voor Windows:

```
<property name="SDK_HOME" value="C:/AIRSDK"/>
<property name="ADT.JAR" value="${SDK_HOME}/lib/adt.jar"/>

target name="package">
    <java jar="${ADT.JAR}" fork="true" failonerror="true">
        <arg value="-package"/>
        <arg value="-storetype"/>
        <arg value="pkcs12"/>
        <arg value="-keystore"/>
        <arg value="../../ExampleCert.p12"/>
        <arg value="myApp.air"/>
        <arg value="myApp-app.xml"/>
        <arg value="myApp.swf"/>
        <arg value="icons/*.png"/>
    </java>
</target>
```

***Opmerking:*** *bij bepaalde computersystemen kunnen double-bytetekens in het bestandssysteem onjuist worden geïnterpreteerd. Wanneer dit zich voordoet, kunt u de JRE die is gebruikt voor het uitvoeren van ADT, instellen om de tekenset UTF-8 te gebruiken. Dit kan standaard worden ingesteld in het script dat wordt gebruikt om ADT op Mac en Linux op te starten. In het adt.bat-bestand voor Windows (of wanneer u ADT rechtstreeks vanuit Java uitvoert), moet u de optie* `-Dfile.encoding=UTF-8` *opgeven op de Java-opdrachtregel.*

## ADT-opdracht voor voorbereiden

Met de ADT-opdracht -prepare wordt een niet-ondertekend AIRI-pakket gemaakt. Een AIRI-pakket mag niet alleen worden gebruikt. Gebruik de opdracht -sign om een AIRI-bestand om te zetten naar een ondertekend AIR-pakket of gebruik de opdracht -package om het AIRI-bestand om te zetten naar een native pakket.

Voor de opdracht -prepare wordt de volgende syntaxis gebruikt:

```
adt -prepare output app_descriptor FILE_OPTIONS
```

**uitvoer** De naam van het gemaakte AIRI-bestand.

**app_descriptor** Het pad naar het descriptorbestand van de toepassing. Dit kan een relatief pad ten opzichte van de huidige map of een absoluut pad zijn. (De naam van het descriptorbestand van de toepassing wordt in het AIR-bestand gewijzigd in *application.xml*.)

**FILE_OPTIONS** Hiermee worden de toepassingsbestanden weergegeven die in het pakket moeten worden opgenomen. De bestandsopties worden volledig beschreven in "Bestands- en padopties" op pagina 190.

## ADT-opdracht voor ondertekenen

Met de opdracht -sign worden AIRI- en ANE-bestanden ondertekend.

Voor de opdracht -sign wordt de volgende syntaxis gebruikt:

```
adt -sign AIR_SIGNING_OPTIONS input output
```

**AIR_SIGNING_OPTIONS** In de AIR-ondertekeningsopties wordt het certificaat weergegeven waarmee een pakketbestand kan worden ondertekend. De ondertekeningsopties worden volledig beschreven in "ADT-opties voor codeondertekening" op pagina 188.

**invoer** De naam van het AIRI- of ANE-bestand dat ondertekend moet worden.

**uitvoer** De naam van het te maken ondertekende pakket.

Als een ANE-bestand al is ondertekend, wordt de oude ondertekening verwijderd. (AIR-bestanden kunnen niet opnieuw worden ondertekend. Als u een nieuwe ondertekening wilt gebruiken voor een toepassingsupdate, gebruikt u de opdracht `-migrate`.)

## ADT-opdracht voor migreren

Met de opdracht -migrate wordt een migratiehandtekening toegepast op een AIR-bestand. Een migratiehandtekening moet worden gebruikt wanneer u uw digitale certificaat vernieuwt of wijzigt en wanneer u uw toepassingen moet bijwerken die zijn ondertekend met het oude certificaat.

Zie "Een bijgewerkte versie van een AIR-toepassing ondertekenen." op pagina 209 voor meer informatie over het in een pakket plaatsen van AIR-toepassingen met een migratiehandtekening.

***Opmerking:*** *het migratiecertificaat moet binnen 365 dagen worden toegepast vanaf het verlopen van het certificaat. Zodra deze respijtperiode is verlopen, kunnen uw toepassingsupdates niet langer worden ondertekend met een migratiehandtekening. Gebruikers kunnen eerst bijwerken naar een versie van uw toepassing die is ondertekend met een migratiehandtekening en vervolgens de nieuwste update installeren, of zij kunnen de oorspronkelijke toepassing verwijderen en het nieuwe AIR-pakket installeren.*

Als u een migratiehandtekening wilt gebruiken, ondertekent u eerst uw AIR-toepassing met het nieuwe of vernieuwde certificaat (met behulp van de opdrachten -package of -sign) en past u vervolgens de migratiehandtekening toe met behulp van het oude certificaat en de opdracht -migrate.

Voor de opdracht `-migrate` wordt de volgende syntaxis gebruikt:

```
adt -migrate AIR_SIGNING_OPTIONS input output
```

**AIR_SIGNING_OPTIONS** In de AIR-ondertekeningsopties wordt het originele certificaat weergegeven dat is gebruikt om bestaande versies van de AIR-toepassing te ondertekenen. De ondertekeningsopties worden volledig beschreven in "ADT-opties voor codeondertekening" op pagina 188.

**invoer** Het AIR-bestand dat al is ondertekend met het NIEUWE toepassingscertificaat.

**uitvoer** De naam van het definitieve pakket met de ondertekeningen van het oude en nieuwe certificaat.

De bestandsnamen die worden gebruikt voor de invoer- en uitvoerbestanden van AIR moeten verschillend zijn.

***Opmerking:*** *de ADT-opdracht voor migreren kan niet worden gebruikt met AIR-bureaubladtoepassingen die native extensies bevatten, omdat deze toepassingen in een pakket zijn opgenomen als native installatieprogramma's, en niet als AIR-bestanden. Als u certificaten wilt wijzigen voor een AIR-bureaubladtoepassing die een native extensie bevat, neemt u de toepassing op in een pakket met behulp van de "ADT-opdracht voor verpakken" op pagina 174 met de markering -migrate.*

## ADT-opdracht voor checkstore

Met de opdracht -checkstore kunt u de geldigheid van een sleutelarchief controleren. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -checkstore SIGNING_OPTIONS
```

**SIGNING_OPTIONS** In de ondertekeningsopties wordt het te controleren sleutelarchief weergegeven. De ondertekeningsopties worden volledig beschreven in "ADT-opties voor codeondertekening" op pagina 188.

## ADT-opdracht voor het certificaat

Met de opdracht -certificate kunt u een zelfondertekend ondertekeningscertificaat voor de digitale code maken. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -certificate -cn name -ou orgUnit -o orgName -c country -validityPeriod years key-type
output password
```

**-cn** De tekenreeks die is toegewezen als de algemene naam (CN) van het nieuwe certificaat.

**-ou** Een tekenreeks die is toegewezen als de organisatie die het certificaat uitgeeft. (Optioneel.)

**`-o`** Een tekenreeks die is toegewezen als de organisatie die het certificaat uitgeeft. (Optioneel.)

**`-c`** Een ISO-3166-landcode van twee letters. Er wordt geen certificaat gegenereerd als een ongeldige code is opgegeven. (Optioneel.)

**`-validityPeriod`** Het aantal jaren dat het certificaat geldig is. Als de geldigheidsduur niet is opgegeven, wordt een geldigheidsduur van vijf jaar toegewezen. (Optioneel.)

**`key_type`** Het type sleutel dat voor het certificaat wordt gebruikt is *2048-RSA*.

**uitvoer** Het pad en de bestandsnaam voor het te genereren certificaatbestand.

**password** Het wachtwoord voor toegang tot het nieuwe certificaat. Het wachtwoord is vereist voor het ondertekenen van AIR-bestanden met dit certificaat.

## ADT-opdracht installApp

Met de opdracht -installApp installeert u een toepassing op een apparaat of emulator.

U moet een bestaande toepassing verwijderen voordat u opnieuw een toepassing installeert met dit bestand.

Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -installApp -platform platformName -platformsdk path-to-sdk -device deviceID -package
fileName
```

**-platform** De naam van het platform van het apparaat. Geef *ios* of *android* op.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat (optioneel):

- Android: de AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)
- iOS: de AIR-SDK wordt geleverd met een captive iOS-SDK. Met de optie -platformsdk kunt u toepassingen inpakken met een extern SDK, zodat u niet beperkt bent tot het gebruik van de captive iOS-SDK. Als u bijvoorbeeld een uitbreiding hebt gemaakt met de nieuwste iOS-SDK, kunt u die SDK opgeven bij het inpakken van uw toepassing. Bovendien geldt dat wanneer u ADT gebruikt met de iOS-simulator, u altijd de optie -platformsdk moet opnemen om het pad naar de iOS-simulator-SDK aan te duiden.

**-device** Geef de *ios_simulator*, het serienummer (Android) of de handle (iOS) van het aangesloten apparaat op. Voor iOS is deze parameter vereist. Voor Android moet deze parameter alleen worden opgegeven als er meer dan één Android-apparaat of emulator is aangesloten op uw computer en wordt uitgevoerd. Als het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout (Android) of Ongeldig apparaat opgegeven (iOS). Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

***Opmerking:** u kunt een IPA-bestand rechtstreeks installeren op een iOS-apparaat in AIR 3.4 en hoger. Hiervoor is iTunes 10.5.0 of hoger vereist.*

Gebruik de ADT-opdracht `-devices` (beschikbaar in AIR 3.4 en hoger) om de handle of het serienummer van de aangesloten apparaten te bepalen. Opgelet: bij iOS gebruikt u de handle, niet de apparaat-UUID. Zie "ADT-opdracht devices" op pagina 187 voor meer informatie.

Bij Android kunt u bovendien het Android ADB-hulpprogramma gebruiken om de serienummers van de aangesloten apparaten en uitgevoerde emulators weer te geven:

```
adb devices
```

**-package** De bestandsnaam van het te installeren pakket. Bij iOS moet dit een IPA-bestand zijn. Bij Android moet dit een APK-pakket zijn. Als het opgegeven pakket al is geïnstalleerd, retourneert ADT foutcode 14: Apparaatfout.

## ADT-opdracht appVersion

Met de opdracht -appVersion wordt de geïnstalleerde versie van een toepassing op een apparaat of emulator weergegeven. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -appVersion -platform platformName -platformsdk path_to_sdk -device deviceID -appid
applicationID
```

**-platform** De naam van het platform van het apparaat. Geef *ios* of *android* op.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat:

- Android: de AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)
- iOS: de AIR-SDK wordt geleverd met een captive iOS-SDK. Met de optie -platformsdk kunt u toepassingen inpakken met een extern SDK, zodat u niet beperkt bent tot het gebruik van de captive iOS-SDK. Als u bijvoorbeeld een uitbreiding hebt gemaakt met de nieuwste iOS-SDK, kunt u die SDK opgeven bij het inpakken van uw toepassing. Bovendien geldt dat wanneer u ADT gebruikt met de iOS-simulator, u altijd de optie -platformsdk moet opnemen om het pad naar de iOS-simulator-SDK aan te duiden.

**-device** Geef *ios_simulator* of het serienummer van het apparaat op. Het apparaat hoeft alleen te worden opgegeven wanneer meer dan één Android-apparaat of emulator met uw computer is verbonden en actief is. Als het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout. Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

Bij Android kunt u het Android-hulpprogramma ADB gebruiken om de serienummers weer te geven van aangesloten apparaten en actieve emulators:

```
adb devices
```

**-appid** De AIR-toepassings-id van de geïnstalleerde toepassing. Als geen toepassing voor de opgegeven id op het apparaat is geïnstalleerd, retourneert ADT afsluitcode 14: Apparaatfout.

## ADT-opdracht launchApp

Met de opdracht -launchApp wordt een geïnstalleerde toepassing op een apparaat of emulator uitgevoerd. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -launchApp -platform platformName -platformsdk path_to_sdk -device deviceID -appid
applicationID
```

**-platform** De naam van het platform van het apparaat. Geef *ios* of *android* op.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat:

- Android: de AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)
- iOS: de AIR-SDK wordt geleverd met een captive iOS-SDK. Met de optie -platformsdk kunt u toepassingen inpakken met een extern SDK, zodat u niet beperkt bent tot het gebruik van de captive iOS-SDK. Als u bijvoorbeeld een uitbreiding hebt gemaakt met de nieuwste iOS-SDK, kunt u die SDK opgeven bij het inpakken van uw toepassing. Bovendien geldt dat wanneer u ADT gebruikt met de iOS-simulator, u altijd de optie -platformsdk moet opnemen om het pad naar de iOS-simulator-SDK aan te duiden.

**-device** Geef *ios_simulator* of het serienummer van het apparaat op. Het apparaat hoeft alleen te worden opgegeven wanneer meer dan één Android-apparaat of emulator met uw computer is verbonden en actief is. Als het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout. Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

Bij Android kunt u het Android-hulpprogramma ADB gebruiken om de serienummers weer te geven van aangesloten apparaten en actieve emulators:

```
adb devices
```

**-appid** De AIR-toepassings-id van de geïnstalleerde toepassing. Als geen toepassing voor de opgegeven id op het apparaat is geïnstalleerd, retourneert ADT afsluitcode 14: Apparaatfout.

## ADT-opdracht uninstallApp

Met de opdracht -uninstallApp wordt een geïnstalleerde toepassing op een extern apparaat of externe emulator volledig verwijderd. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -uninstallApp -platform platformName -platformsdk path_to_sdk -device deviceID -appid
applicationID
```

**-platform** De naam van het platform van het apparaat. Geef *ios* of *android* op.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat:

- Android: de AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)
- iOS: de AIR-SDK wordt geleverd met een captive iOS-SDK. Met de optie -platformsdk kunt u toepassingen inpakken met een extern SDK, zodat u niet beperkt bent tot het gebruik van de captive iOS-SDK. Als u bijvoorbeeld een uitbreiding hebt gemaakt met de nieuwste iOS-SDK, kunt u die SDK opgeven bij het inpakken van uw toepassing. Bovendien geldt dat wanneer u ADT gebruikt met de iOS-simulator, u altijd de optie -platformsdk moet opnemen om het pad naar de iOS-simulator-SDK aan te duiden.

**-device** Geef *ios_simulator* of het serienummer van het apparaat op. Het apparaat hoeft alleen te worden opgegeven wanneer meer dan één Android-apparaat of emulator met uw computer is verbonden en actief is. Als het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout. Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

Bij Android kunt u het Android-hulpprogramma ADB gebruiken om de serienummers weer te geven van aangesloten apparaten en actieve emulators:

```
adb devices
```

**-appid** De AIR-toepassings-id van de geïnstalleerde toepassing. Als geen toepassing voor de opgegeven id op het apparaat is geïnstalleerd, retourneert ADT afsluitcode 14: Apparaatfout.

## ADT-opdracht installRuntime

Met de opdracht -installRuntime wordt de AIR-runtime op een apparaat geïnstalleerd.

U moet een bestaande versie van de AIR-runtime verwijderen voordat u opnieuw gaat installeren met behulp van deze opdracht.

Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -installRuntime -platform platformName -platformsdk path_to_sdk -device deviceID -package
fileName
```

**-platform** De naam van het platform van het apparaat. Momenteel wordt deze opdracht alleen ondersteund door het Android-platform. Gebruik de naam *android*.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat. Momenteel is Android de enige ondersteunde platform-SDK. De AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)

**-device** Het serienummer van het apparaat. Het apparaat hoeft alleen te worden opgegeven wanneer meer dan één apparaat of emulator met uw computer is verbonden en actief is. Als het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout. Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

Bij Android kunt u het Android-hulpprogramma ADB gebruiken om de serienummers weer te geven van aangesloten apparaten en actieve emulators:

```
adb devices
```

**-package** De bestandsnaam van de te installeren runtime. Bij Android moet dit een APK-pakket zijn. Als geen pakket is opgegeven, wordt de geschikte runtime voor het apparaat of de emulator gekozen uit de beschikbare runtimes in de AIR-SDK. Als de runtime al is geïnstalleerd, retourneert ADT foutcode 14: Apparaatfout.

## ADT-opdracht runtimeVersion

Met de opdracht -runtimeVersion wordt de geïnstalleerde versie van een AIR-runtime op een apparaat of emulator weergegeven. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -runtimeVersion -platform platformName -platformsdk path_to_sdk -device deviceID
```

**-platform** De naam van het platform van het apparaat. Momenteel wordt deze opdracht alleen ondersteund door het Android-platform. Gebruik de naam *android*.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat. Momenteel is Android de enige ondersteunde platform-SDK. De AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)

**-device** Het serienummer van het apparaat. Het apparaat hoeft alleen te worden opgegeven wanneer meer dan één apparaat of emulator met uw computer is verbonden en actief is. Als de runtime niet is geïnstalleerd of het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout. Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

Bij Android kunt u het Android-hulpprogramma ADB gebruiken om de serienummers weer te geven van aangesloten apparaten en actieve emulators:

```
adb devices
```

## ADT-opdracht uninstallRuntime

Met de opdracht -uninstallRuntime wordt de AIR-runtime volledig van een apparaat of een emulator verwijderd. Voor de opdracht wordt de volgende syntaxis gebruikt:

```
adt -uninstallRuntime -platform platformName -platformsdk path_to_sdk -device deviceID
```

**-platform** De naam van het platform van het apparaat. Momenteel wordt deze opdracht alleen ondersteund door het Android-platform. Gebruik de naam *android*.

**-platformsdk** Het pad naar de platform-SDK voor het doelapparaat. Momenteel is Android de enige ondersteunde platform-SDK. De AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Bovendien hoeft het pad naar de platform-SDK niet in de opdrachtregel te worden opgegeven als de omgevingsvariabele AIR_ANDROID_SDK_HOME al is ingesteld. (Als beide mogelijkheden zijn ingesteld, wordt de in het pad opgegeven opdrachtregel gebruikt.)

**-device** Het serienummer van het apparaat. Het apparaat hoeft alleen te worden opgegeven wanneer meer dan één apparaat of emulator met uw computer is verbonden en actief is. Als het opgegeven apparaat niet is aangesloten, retourneert ADT afsluitcode 14: Apparaatfout. Als meer dan één apparaat of emulator is aangesloten en een apparaat niet is opgegeven, retourneert ADT afsluitcode 2: Gebruiksfout.

Bij Android kunt u het Android-hulpprogramma ADB gebruiken om de serienummers weer te geven van aangesloten apparaten en actieve emulators:

```
adb devices
```

## ADT-opdracht devices

Met de ADT-opdracht -help geeft u de apparaat-id's weer van de momenteel aangesloten mobiele apparaten en emulators:

```
adt -devices -platform iOS|android
```

**-platform** De naam van het platform dat moet worden gecontroleerd. Geef `android` of `iOS` op.

***Opmerking:*** *op iOS is voor deze opdracht iTunes 10.5.0 of hoger vereist.*

## ADT-opdracht help

Met de ADT-opdracht -help wordt een beknopte geheugensteun weergegeven van de opties voor opdrachtregels:

```
adt -help
```

Bij de helpuitvoer worden de volgende standaardsymbolen gebruikt:

- <> — onderwerpen tussen punthaken geven aan welke gegevens u moet opgeven.
- () — onderwerpen tussen haakjes geven opties aan die in de uitvoer van de helpopdracht als groep worden behandeld.
- ALL_CAPS — onderwerpen in hoofdletters geven een aantal opties aan die afzonderlijk wordt beschreven.
- | — OF. Bijvoorbeeld: ( A | B ) betekent onderwerp A of onderwerp B.
- ? — 0 of 1. Een vraagteken na een onderwerp geeft aan dat een onderwerp optioneel is en dat slechts één instantie ervan kan voorkomen, indien er gebruik van wordt gemaakt.
- * — 0 of meer. Een asterisk na een onderwerp geeft aan dat een onderwerp optioneel is en dat een willekeurig aantal instanties ervan kan voorkomen.
- + — 1 of meer. Een plusteken na een onderwerp geeft aan dat een onderwerp vereist is en dat meerdere instanties ervan voorkomen.
- geen symbool — Als er geen symbool achter een onderwerp wordt weergegeven, is dat onderwerp vereist en kan slechts één instantie ervan voorkomen.

# ADT-opties

Enkele ADT-opdrachten delen dezelfde opties.

## ADT-opties voor codeondertekening

ADT gebruikt JCA (Java Cryptography Architecture) om toegang te krijgen tot persoonlijke sleutels en certificaten voor het ondertekenen van AIR-toepassingen. Met de ondertekeningsopties worden het sleutelarchief en de persoonlijke sleutel en het certificaat in dat sleutelarchief opgegeven.

Het sleutelarchief moet zowel de persoonlijke sleutel als de bijbehorende certificaatketen bevatten. Als het handtekeningcertificaat via een certificaatketen is gekoppeld aan een vertrouwd certificaat op een computer, wordt de inhoud van het veld algemene naam van het certificaat als de uitgeversnaam weergegeven in het dialoogvenster van de AIR-installatie.

Voor ADT moet het certificaat aan de x509v3-standaard voldoen (RFC3280) en de EKU-uitbreiding (Extended Key Usage) met de juiste waarden voor de ondertekening van code bevatten. Beperkingen in het certificaat worden gerespecteerd, waardoor bepaalde certificaten niet kunnen worden gebruikt voor het ondertekenen van AIR-toepassingen.

***Opmerking:*** *ADT gebruikt de proxy-instellingen van de Java-runtimeomgeving (indien van toepassing) om verbinding te maken met internetbronnen voor het controleren van certificaatintrekkingslijsten en het verkrijgen van tijdstempels. Als u bij het gebruik van ADT problemen ondervindt wanneer verbinding wordt gemaakt met deze internetbronnen en er voor uw netwerk specifieke proxy-instellingen zijn vereist, kan het zijn dat u de JRE-proxy-instellingen moet configureren.*

#### Syntaxis van AIR-ondertekeningsopties

De ondertekeningsopties maken gebruik van de volgende syntaxis (op één opdrachtregel):

```
-alias aliasName
-storetype type
-keystore path
-storepass password1
-keypass password2
-providerName className
-tsa url
```

**-alias** De alias van een sleutel in het sleutelarchief. U hoeft geen alias op te geven wanneer een sleutelarchief slechts één certificaat bevat. Als er geen alias is opgegeven, wordt de eerste sleutel in het sleutelarchief gebruikt.

Niet in alle toepassingen voor sleutelarchiefbeheer mag een alias worden toegewezen aan certificaten. Als u het sleutelarchief van Windows gebruikt, wordt bijvoorbeeld de DN-naam van het certificaat als alias gebruikt. Met het Java-hulpprogramma Keytool kunt u de beschikbare certificaten weergeven zodat u de alias kunt bepalen. Als u bijvoorbeeld deze opdracht uitvoert:

```
keytool -list -storetype Windows-MY
```

wordt soortgelijke uitvoer als hieronder weergegeven voor een certificaat:

```
CN=TestingCert,OU=QE,O=Adobe,C=US, PrivateKeyEntry,
Certificate fingerprint (MD5): 73:D5:21:E9:8A:28:0A:AB:FD:1D:11:EA:BB:A7:55:88
```

Als u in de ADT-opdracht op de opdrachtregel naar dit certificaat wilt verwijzen, stelt u de alias in op:

```
CN=TestingCert,OU=QE,O=Adobe,C=US
```

In Mac OS X is de alias van een certificaat in de Keychain de naam die wordt weergegeven in de toepassing Keychain Access.

**-storetype** Het type sleutelarchief. Dit wordt bepaald door de implementatie van het sleutelarchief. De standaard sleutelarchiefimplementatie die in de meeste installaties van Java is opgenomen, ondersteunt de typen `JKS` en `PKCS12`. Java 5.0 biedt ondersteuning voor het type `PKCS11`, voor toegang tot sleutelarchieven op hardwaretokens, en het type `Keychain`, voor toegang tot de KeyChain van Mac OS X. Java 6.0 biedt ondersteuning voor het type `MSCAPI` (in Windows). Als er andere JCA-providers zijn geïnstalleerd en geconfigureerd, kunnen er meer sleutelarchieftypen beschikbaar zijn. Als er geen sleutelarchieftype is opgegeven, wordt het standaardtype voor de standaard JCA-provider gebruikt.

| Archieftype | Indeling van sleutelarchief | Minimale Java-versie |
|---|---|---|
| JKS | Java-sleutelarchiefbestand (.keystore) | 1.2 |
| PKCS12 | PKCS12-bestand (.p12 of .pfx) | 1.4 |
| PKCS11 | Hardwaretoken | 1.5 |
| KeyChain-archief | KeyChain van Mac OS X | 1.5 |
| Windows-MY of Windows-ROOT | MSCAPI | 1.6 |

**-keystore** Het pad naar het sleutelarchiefbestand voor archieftypen die op bestanden zijn gebaseerd.

**-storepass** Het wachtwoord voor toegang tot het sleutelarchief. Als u dit niet opgeeft, vraagt ADT om het wachtwoord.

**-keypass** Het wachtwoord voor toegang tot de persoonlijke sleutel waarmee de AIR-toepassing wordt ondertekend. Als u dit niet opgeeft, vraagt ADT om het wachtwoord.

***Opmerking:*** *als u een wachtwoord opgeeft als onderdeel van de ADT-opdracht, worden de wachtwoordtekens opgeslagen in de geschiedenis van de opdrachtregel. Om die reden wordt het gebruik van de opties -keypass of -storepass niet aanbevolen wanneer de beveiliging van het certificaat belangrijk is. Wanneer u de wachtwoordopties weglaat, worden de tekens die u typt niet weergegeven (om dezelfde beveiligingsredenen). U kunt eenvoudig het wachtwoord typen en op Enter drukken.*

**-providerName** De JCA-provider voor het opgegeven sleutelarchieftype. Als u geen provider opgeeft, wordt de standaardprovider voor dat type sleutelarchief gebruikt.

**-tsa** De URL van een RFC3161-compatibele tijdstempelserver om een tijdstempel voor de digitale handtekening te verkrijgen. Als u geen URL opgeeft, wordt een standaard tijdstempelserver van Geotrust gebruikt. Wanneer de handtekening van een AIR-toepassing een tijdstempel heeft, kan de toepassing nog steeds worden geïnstalleerd nadat het handtekeningcertificaat is verlopen, omdat het tijdstempel aangeeft dat het certificaat geldig was op het moment van ondertekening.

Als ADT geen verbinding kan maken met de tijdstempelserver, wordt het ondertekenen geannuleerd en wordt geen pakket gemaakt. Geef `-tsa none` op om het ophalen van een tijdstempel uit te schakelen. Een AIR-toepassing in een pakket zonder tijdstempel kan echter niet worden geïnstalleerd nadat het handtekeningcertificaat is verlopen.

***Opmerking:*** *veel ondertekeningsopties zijn gelijk aan dezelfde optie in het Java-hulpprogramma Keytool. Met het hulpprogramma Keytool kunt u sleutelarchieven in Windows controleren en beheren. In Mac OS X kunt u hiervoor het beveiligingsprogramma van Apple® gebruiken.*

**-provisioning-profile** Het Apple iOS-inrichtingsbestand. (Alleen vereist voor het verpakken van iOS-toepassingen.)

**Voorbeelden van ondertekeningsopties**
Ondertekenen met een .p12-bestand:

```
-storetype pkcs12 -keystore cert.p12
```

Ondertekenen met het standaard sleutelarchief van Java:

```
-alias AIRcert -storetype jks
```

Ondertekenen met een specifiek sleutelarchief van Java:

```
-alias AIRcert -storetype jks -keystore certStore.keystore
```

Ondertekenen met de KeyChain van Mac OS X:

```
-alias AIRcert -storetype KeychainStore -providerName Apple
```

Ondertekenen met het sleutelarchief van Windows:

```
-alias cn=AIRCert -storeype Windows-MY
```

Ondertekenen met een hardwaretoken (raadpleeg de instructies van de fabrikant van het token om Java te configureren voor het gebruik van het token en voor de juiste waarde van `providerName`):

```
-alias AIRCert -storetype pkcs11 -providerName tokenProviderName
```

Ondertekenen zonder een tijdstempel:

```
-storetype pkcs12 -keystore cert.p12 -tsa none
```

## Bestands- en padopties

Met de bestands- en padopties worden alle bestanden aangegeven die zijn opgenomen in het pakket. Voor de bestands- en padopties wordt de volgende syntaxis gebruikt:

```
files_and_dirs -C dir files_and_dirs -e file_or_dir dir -extdir dir
```

**files_and_dirs** De bestanden en mappen in het AIR-pakket. U kunt een willekeurig aantal bestanden en mappen opgeven, gescheiden door spaties. Als u een map opgeeft, worden alle bestanden en mappen in die map, behalve verborgen bestanden, aan het pakket toegevoegd. (Als het descriptorbestand van de toepassing wordt opgegeven, rechtstreeks of door het omzetten van jokertekens of het uitbreiden van mappen, wordt het genegeerd en niet opnieuw aan het pakket toegevoegd.) De opgegeven bestanden en mappen moeten zich in de huidige map of een submap van de huidige map bevinden. Gebruik de optie `-C` om een andere map dan de huidige te selecteren.

***Belangrijk:*** *in de argumenten `file_or_dir` die na de optie `–C` volgen, kunnen geen jokertekens worden gebruikt. (Opdrachtshells zetten de jokertekens om voordat de argumenten worden doorgegeven aan ADT, waardoor ADT de bestanden op onjuiste locaties zoekt.) U kunt echter wel een punt (".") gebruiken om de huidige map aan te geven. Bijvoorbeeld: `-C assets .` hiermee wordt alles in de map assets, inclusief alle submappen, gekopieerd naar het hoofdniveau van het toepassingspakket.*

**`-C dir files_and_dirs`** Hiermee wijzigt u de werkmap in de waarde van *dir* voordat de volgende bestanden en mappen worden toegevoegd aan het toepassingspakket (opgegeven in **`files_and_dirs`**). De bestanden en mappen worden toegevoegd aan de hoofdmap van het toepassingspakket. U kunt de optie `–C` een willekeurig aantal keren gebruiken om bestanden op te nemen vanuit meerdere plaatsen in het bestandssysteem. Als u voor `dir` een relatief pad opgeeft, wordt het pad altijd gebaseerd op de oorspronkelijke werkmap.

Wanneer ADT de bestanden en mappen omzet die in het pakket aanwezig zijn, worden de relatieve paden tussen de huidige map en de doelbestanden opgeslagen. Deze paden worden naar de mapstructuur van de toepassing omgezet wanneer het pakket wordt geïnstalleerd. Als u `-C release/bin lib/feature.swf` opgeeft, wordt het bestand `release/bin/lib/feature.swf` daarom in de submap `lib` van de hoofdmap van de toepassing geplaatst.

**`-e file_or_dir dir`** Hiermee wordt het bestand of de map in de opgegeven pakketmap geplaatst. U kunt deze optie niet gebruiken wanneer u een ANE-bestand in een pakket plaatst.

***Opmerking:*** *in het element `<content>` van het descriptorbestand van de toepassing moet de uiteindelijke locatie zijn opgegeven van het hoofdbestand van de toepassing in de mapstructuur van het toepassingspakket.*

**-extdir *dir*** De waarde van *`dir`* is de naam van een map waarin naar native extensies (ANE-bestanden) moet worden gezocht. Geef een absoluut pad op, of een pad in relatie tot de huidige map. U kunt de optie `-extdir` meerdere malen opgeven.

De opgegeven map bevat ANE-bestanden voor native extensies die de toepassing gebruikt. Elk ANE-bestand in deze map heeft de bestandsnaamtoevoeging .ane. De bestandsnaam vóór de toevoeging .ane hoeft echter *niet* overeen te komen met de waarde van het `extensionID`-element van het descriptorbestand van de toepassing.

Als u bijvoorbeeld `-extdir ./extensions` gebruikt, kan de map `extensions` er als volgt uitzien:

```
extensions/
    extension1.ane
    extension2.ane
```

***Opmerking:*** *het gebruik van de optie -extdir vanuit de ADT-tool en de ADL-tool is niet hetzelfde. In ADL verwijst de optie naar een map die submappen bevat die elk op hun beurt een niet-verpakt ANE-bestand bevatten. In ADT geeft de optie een map op die ANE-bestanden bevat.*

## Verbindingsopties voor foutopsporing

Wanneer het doel van het pakket apk-debug, ipa-debug of ipa-debug-interpreter is, kunt u aan de hand van de verbindingsopties opgeven of de toepassing zal proberen te verbinden met een extern foutopsporingsprogramma (typisch gebruikt bij wifi-foutopsporing) of zal luisteren naar een inkomende verbinding van een extern foutopsporingsprogramma (typisch gebruikt bij USB-foutopsporing). Gebruik de optie `-connect` om te verbinden met een foutopsporingsprogramma. Gebruik de optie `-listen` om een USB- verbinding met een foutopsporingsprogramma te accepteren. Deze opties kunnen niet met elkaar worden gecombineerd.

De optie -connect gebruikt de volgende syntaxis:

```
-connect hostString
```

**-connect** Indien beschikbaar, probeert de toepassing verbinding te maken met een extern foutopsporingsprogramma.

**hostString** Een reeks waarmee de computer wordt aangeven waarop het Flash hulpprogramma voor foutopsporing (FDB) wordt uitgevoerd. Indien het niet wordt aangegeven, probeert de toepassing verbinding te maken met een foutopsporingsprogramma dat wordt uitgevoerd op de computer waarop het pakket is gemaakt. De hoststring kan een correcte domeinnaam van een computer zijn: *machinename.subgroup.example.com* of een IP-adres: *192.168.4.122*. Als de opgegeven computer (of standaardcomputer) niet wordt gevonden, wordt door de runtime een dialoogvenster weergegeven waarin om een geldige hostnaam wordt gevraagd.

De optie `-listen` gebruikt de volgende syntaxis:

```
-listen port
```

**-listen** Indien aanwezig, wacht de runtime op een verbinding met een extern foutopsporingsprogramma.

**port** (Optioneel) De poort die wordt gebruikt om te luisteren. Standaard gebruikt de runtime poort 7936 om te luisteren. Meer informatie over het gebruik van de optie `-listen` vindt u in "Foutopsporing op afstand met FDB via USB" op pagina 112.

## Opties voor het instellen van profielen van de Android-toepassing

Wanneer het doel van het pakket apk-profile is, kunnen de opties voor het instellen van een profiel worden gebruikt om op te geven welk vooraf geladen SWF-bestand moet worden gebruikt voor het instellen van profielen voor prestatie en geheugen. Voor de opties voor het instellen van een profiel wordt de volgende syntaxis gebruikt:

```
-preloadSWFPath directory
```

**-preloadSWFPath** Indien beschikbaar probeert de toepassing het vooraf geladen SWF-bestand te zoeken in de opgegeven map. Indien het niet is opgegeven, neemt ADT het vooraf geladen SWF-bestand van de AIR SDK op.

**directory** De map met het vooraf geladen SWF-bestand voor het instellen van profielen.

## Opties voor native extensies

De opties voor native extensies geven de opties en bestanden op voor het verpakken van een ANE-bestand voor een native extensie. Gebruik deze opties met een ADT-pakketopdracht waarin de `-target`-optie `ane` is.

```
extension-descriptor -swc swcPath
   -platform platformName
  -platformoptions path/platform.xml
   FILE_OPTIONS
```

**extension-descriptor** Het descriptorbestand voor de native extensie.

**-swc** Het SWC-bestand met de ActionScript-code en -bronnen voor de native extensie.

**-platform** De naam van het platform dat door dit ANE-bestand wordt ondersteund. U kunt meerdere `-platform`-opties gebruiken, elk met hun eigen `FILE_OPTIONS`.

**-platformoptions** Het pad naar een 'platform options'-bestand (platform.xml). Met dit bestand kunt u niet-standaard Linker-opties, gedeelde bibliotheken en statische bibliotheken van derden die door de extensie worden gebruikt, instellen. Zie Native iOS-bibliotheken voor meer informatie en voorbeelden.

**FILE_OPTIONS** Identificeert de native platformbestanden die in het pakket worden opgenomen, zoals statische bibliotheken die in het native extensiepakket worden opgenomen. De bestandsopties worden volledig beschreven in “Bestands- en padopties” op pagina 190. (De optie -e kan niet worden gebruikt bij het verpakken van een ANE-bestand.)

### Meer Help-onderwerpen

Een native extensie in een pakket plaatsen

# ADT-foutmeldingen

In de onderstaande tabel worden de mogelijke fouten aangeven die kunnen worden gerapporteerd door het ADT-programma en de mogelijke oorzaken daarvan:

**Fouten bij de validatie van toepassingsbeschrijvingen**

| Foutcode | Beschrijving | Opmerkingen |
|---|---|---|
| 100 | Toepassingsbeschrijving kan niet worden geparseerd | Controleer of er fouten voorkomen in de XML-syntaxis van het toepassingsdescriptorbestand, zoals tags zonder eindtag. |
| 101 | Naamruimte ontbreekt | Voeg de ontbrekende naamruimte toe. |
| 102 | Ongeldige naamruimte | Controleer de spelling van de naamruimte. |
| 103 | Onverwacht element of attribuut | Verwijder de desbetreffende elementen en attributen. Aangepaste waarden zijn niet toegestaan in het descriptorbestand.<br>Controleer de spelling van de namen van elementen en attributen.<br>Controleer of elementen zijn geplaatst binnen de correcte bovenliggende elementen en of attributen worden gebruikt met de correcte elementen. |
| 104 | Ontbrekend element of attribuut | Voeg het vereiste element of attribuut toe. |
| 105 | Element of attribuut bevat een ongeldige waarde | Corrigeer de desbetreffende waarde. |
| 106 | Ongeldige combinatie van vensterattributen | Bepaalde vensterinstellingen zoals `transparency = true` en `systemChrome = standard`, kunnen niet gezamenlijk worden gebruikt. Wijzig een van de incompatibele instellingen. |
| 107 | Minimumgrootte van venster is groter dan maximumgrootte | Verander de instelling voor de minimum- of maximumgrootte. |
| 108 | Attribuut wordt al gebruikt in vorig element | |
| 109 | Dubbel element. | Verwijder het dubbele element. |
| 110 | Ten minste één element van het opgegeven type wordt vereist. | Voeg het ontbrekende element toe. |
| 111 | De profielen die in de toepassingsdescriptor zijn opgenomen, ondersteunen geen native extensies. | Voeg een profiel toe aan de lijst supportedProfies waarmee de native extensies worden ondersteund. |
| 112 | Het AIR-doel ondersteunt geen native extensies. | Kies een doel dat native extensies ondersteunt. |
| 113 | <nativeLibrary> en <initializer> moeten samen worden opgegeven. | Er moet voor elke native bibliotheek een initialisatiefunctie worden opgegeven in de native extensie. |
| 114 | <finalizer> gevonden zonder <nativeLibrary>. | Geef geen initialisatiefunctie op tenzij een native bibliotheek door het platform wordt gebruikt. |

| Foutcode | Beschrijving | Opmerkingen |
|---|---|---|
| 115 | Het standaardplatform mag geen native implementatie bevatten. | Geef geen native bibliotheek op in het standaardplatformelement. |
| 116 | Het is met dit doel niet mogelijk een browser aan te roepen. | Het element `<allowBrowserInvocation>` kan niet `true` zijn voor het opgegeven pakketdoel. |
| 117 | Dit doel vereist minstens naamruimte n om native extensies in een pakket te kunnen plaatsen. | Wijzig de AIR-naamruimte in het descriptorbestand van de toepassing in een ondersteunde waarde. |

Zie “AIR-toepassingsdescriptorbestanden” op pagina 215 voor meer informatie over de naamruimtes, elementen en attributen en hun geldige waarden.

### Fouten bij toepassingspictogrammen

| Foutcode | Beschrijving | Opmerkingen |
|---|---|---|
| 200 | Kan pictogrambestand niet openen | Controleer of het bestand aanwezig is op het opgegeven pad.<br>Gebruik een andere toepassing om er zeker van te zijn dat het bestand kan worden geopend. |
| 201 | Pictogram heeft het verkeerde formaat | Het formaat van het pictogram (in pixels) moet overeenkomen met de XML-tag. Kijk bijvoorbeeld naar het volgende toepassingsbeschrijvingselement:<br>`<image32x32>icon.png</image32x32>`<br>De afbeelding in `icon.png` moet precies 32 x 32 pixels bevatten. |
| 202 | Pictogrambestand bevat een afbeeldingsindeling die niet wordt ondersteund | Alleen de PNG-indeling wordt ondersteund. Converteer afbeeldingen in andere indelingen voordat u de toepassing inpakt. |

### Fouten in toepassingsbestanden

| Foutcode | Beschrijving | Opmerkingen |
|---|---|---|
| 300 | Bestand ontbreekt of kan niet worden geopend | Een bestand dat wordt gespecificeerd op de opdrachtregel, kan niet worden gevonden of geopend. |
| 301 | Toepassingsdescriptorbestand ontbreekt of kan niet worden geopend | Het toepassingsdescriptorbestand kan niet worden gevonden op het gespecificeerde pad of kan niet worden geopend. |
| 302 | Hoofdinhoudsbestand ontbreekt uit pakket | Het SWF- of HTML-bestand waarnaar wordt verwezen in het element `<content>` van het toepassingsdescriptorbestand, moet worden toegevoegd aan het pakket door het op te nemen in de bestanden die worden opgesomd op de ADT-opdrachtregel. |

| Foutcode | Beschrijving | Opmerkingen |
|---|---|---|
| 303 | Pictogrambestand ontbreekt uit pakket | De pictogrambestanden die worden opgegeven in de toepassingsbeschrijving, moeten worden toegevoegd aan het pakket door ze op te nemen in de bestanden die worden opgesomd op de ADT-opdrachtregel. Pictogrambestanden worden niet automatisch toegevoegd. |
| 304 | Initiële vensterinhoud is ongeldig | Het bestand waarnaar wordt verwezen in het element `<content>` van de toepassingsbeschrijving, wordt niet herkend als geldig HTML- of SWF-bestand. |
| 305 | SWF-versie van initiële vensterinhoud is hoger dan naamruimteversie | De SWF-versie van het bestand waarnaar wordt verwezen in het element `<content>` van de toepassingsbeschrijving, wordt niet ondersteund door de versie van AIR die wordt gespecificeerd in de descriptor-naamruimte. Deze fout wordt bijvoorbeeld gegenereerd als u probeert een SWF10 (Flash Player 10)-bestand te verpakken als de initiële inhoud van een AIR 1.1-toepassing. |
| 306 | Profiel wordt niet ondersteund. | Het profiel dat u hebt opgegeven in het descriptorbestand van de toepassing wordt niet ondersteund. Zie "supportedProfiles" op pagina 251. |
| 307 | Naamruimte moet ten minste *nnn* zijn. | Gebruik de juiste naamruimte voor de functies die in de toepassing worden gebruikt (zoals naamruimte 2.0). |

## Afsluitcodes voor andere fouten

| Afsluitcode | Beschrijving | Opmerkingen |
|---|---|---|
| 2 | Gebruiksfout | Controleer de opdrachtregelargumenten op fouten. |
| 5 | Onbekende fout | Deze fout geeft een situatie aan die niet kan worden verklaard door normale foutcondities. Mogelijke basisoorzaken omvatten incompatibiliteit tussen ADT en de Java-runtimeomgeving, beschadigde ADT- of JRE-installaties of programmeerfouten in ADT. |
| 6 | Kan niet schrijven naar uitvoermap | Controleer of de gespecificeerde (of geïmpliceerde) uitvoermap toegankelijk is en of het station waarop deze map zich bevindt, voldoende schijfruimte heeft. |
| 7 | Kan certificaat niet openen | Controleer of het pad naar de sleutelopslag correct is gespecificeerd.<br><br>Controleer of het certificaat in de sleutelopslag toegankelijk is. U kunt het hulpprogramma Java 1.6 Keytool gebruiken om problemen met de toegang tot certificaten op te lossen. |

| Afsluitcode | Beschrijving | Opmerkingen |
|---|---|---|
| 8 | Ongeldig certificaat | Het certificaat is verkeerd samengesteld, gewijzigd, verlopen of ingetrokken. |
| 9 | Kan het AIR-bestand niet ondertekenen | Verifieer de ondertekeningsopties die worden doorgegeven aan ADT. |
| 10 | Kan geen tijdstempel maken | ADT kan geen verbinding met de tijdstempelserver tot stand brengen. Als u via een proxyserver verbinding maakt met internet, moet u mogelijk de JRE-proxyinstellingen configureren. |
| 11 | Fout bij maken certificaat | Verifieer de opdrachtregelargumenten die worden gebruikt voor het maken van handtekeningen. |
| 12 | Ongeldige invoer | Verifieer de bestandspaden en andere argumenten die via de opdrachtregel worden doorgegeven naar ADT. |
| 13 | Ontbrekende SDK van apparaat | Controleer de configuratie van apparaat met de SDK. ADT kan de SDK van het apparaat niet vinden die nodig is om de opgegeven opdracht uit te voeren. |
| 14 | Apparaatfout | ADT kan de opdracht niet uitvoeren omdat er een beperking voor het apparaat geldt of omdat er problemen met het apparaat zijn. Deze afsluitcode wordt bijvoorbeeld gegeven wanneer wordt geprobeerd om een toepassing te verwijderen die in werkelijkheid niet is geïnstalleerd. |
| 15 | Geen apparaten | Controleer of er een apparaat is aangesloten en ingeschakeld of een emulator actief is. |
| 16 | Ontbrekende GPL-componenten | De huidige AIR SDK bevat niet alle componenten die vereist worden om de gevraagde actie uit te voeren. |
| 17 | Hulpprogramma voor in pakket plaatsen van apparaat mislukt. | Kan het pakket niet maken, omdat verwachte besturingssysteemonderdelen ontbreken. |

### Android-fouten

| Afsluitcode | Beschrijving | Opmerkingen |
|---|---|---|
| 400 | Attribuut wordt niet ondersteund door de huidige Android-SDK-versie. | Controleer of de naam van het attribuut correct is gespeld en of het attribuut geldig is voor het element waarin het wordt weergegeven. U moet mogelijk de markering -platformsdk in de ADT-opdracht instellen als het attribuut is geïntroduceerd na Android 2.2. |
| 401 | De attribuutwaarde wordt niet ondersteund door de huidige Android-SDK-versie | Controleer of de naam van de attribuutwaarde correct is gespeld en of het een geldige waarde is voor het attribuut. U moet mogelijk de markering -platformsdk in de ADT-opdracht instellen als het attribuut is geïntroduceerd na Android 2.2. |
| 402 | De XML-tag wordt niet ondersteund door de huidige Android-SDK-versie | Controleer of de naam van de XML-tag correct is gespeld en een geldig documentelement van het Android-manifest is. U moet mogelijk de markering -platformsdk in de ADT-opdracht instellen als het element is geïntroduceerd na Android 2.2. |
| 403 | Android-tag mag niet worden overschreven | De toepassing probeert een Android-manifestelement te overschrijven dat is gereserveerd voor gebruik door AIR. Zie "Android-instellingen" op pagina 80. |
| 404 | Android-attribuut mag niet worden overschreven | De toepassing probeert een Android-manifestattribuut te overschrijven dat is gereserveerd voor gebruik door AIR. Zie "Android-instellingen" op pagina 80. |
| 405 | Android-tag %1 dient het eerste element in manifestAdditions-tag te zijn | Verplaats de opgegeven tag naar de vereiste locatie. |
| 406 | Het kenmerk %1 van Android-tag %2 heeft de ongeldige waarde %3. | Geef een geldige waarde op voor het kenmerk. |

# ADT-omgevingsvariabelen

ADT leest de waarden van de volgende omgevingsvariabelen (als deze zijn ingesteld):

**AIR_ANDROID_SDK_HOME** geeft het pad aan naar de hoofdmap van de Android-SDK. (De map met de map met hulpprogramma's). De AIR 2.6+ SDK bevat de hulpprogramma's van de Android-SDK die nodig zijn om de relevante ADT-opdrachten te implementeren. U moet deze waarde alleen instellen als u een andere versie van de Android-SDK wilt gebruiken. Als deze variabele is ingesteld, hoeft de optie -platformsdk niet worden opgegeven wanneer de ADT-opdrachten worden uitgevoerd waarvoor dit vereist is. Als deze variabele en de opdrachtregeloptie zijn ingesteld, wordt het pad gebruikt dat in de opdrachtregel wordt opgegeven.

**AIR_EXTENSION_PATH** geeft een mappenlijst op die moet worden doorzocht op door een toepassing vereiste native extensies. De mappenlijst wordt in de juiste volgorde doorzocht nadat native extensiemappen zijn opgegeven op de ADT-opdrachtregel. De ADL-opdracht gebruikt deze omgevingsvariabele ook.

***Opmerking:*** *Bij bepaalde computersystemen kunnen double-bytetekens in de bestandssysteempaden die zijn opgeslagen in deze omgevingsvariabelen, onjuist worden geïnterpreteerd. Wanneer dit zich voordoet, kunt u de JRE die is gebruikt voor het uitvoeren van ADT, instellen om de tekenset UTF-8 te gebruiken. Dit kan standaard worden ingesteld in het script dat wordt gebruikt om ADT op Mac en Linux op te starten. In het Windows-bestand adt.bat file of wanneer u ADT direct in Java uitvoert, geeft u de optie* `-Dfile.encoding=UTF-8` *op in de Java-opdrachtregel.*

# Hoofdstuk 13: AIR-toepassingen ondertekenen

## AIR-bestanden digitaal ondertekenen

Als u uw AIR-installatiebestanden digitaal ondertekent met een certificaat dat is uitgegeven door een erkende certificeringsinstantie (CA), hebben uw gebruikers een goede garantie dat de toepassing die zij installeren niet per ongeluk of met boze opzet is gewijzigd en weten zij dat u de ondertekenaar (uitgever) bent. De uitgeversnaam wordt tijdens de installatie weergegeven wanneer de AIR-toepassing is ondertekend met een certificaat dat wordt vertrouwd of dat via een *certificaatketen* is gekoppeld aan een certificaat dat wordt vertrouwd op de installatiecomputer:


*Installatiedialoogvenster voor bevestiging voor toepassing die met een vertrouwd certificaat is ondertekend*

Als u een toepassing ondertekent met een zelfondertekend certificaat (of een certificaat dat niet aan een vertrouwd certificaat is gekoppeld), moet de gebruiker bij het installeren van de toepassing een groter beveiligingsrisico accepteren. In het dialoogvenster wordt dit extra risico aangegeven:


*Installatiedialoogvenster voor bevestiging voor toepassing die met een zelfondertekend certificaat is ondertekend*

***Belangrijk:** een kwaadwillende entiteit kan uw identiteit in een AIR-bestand vervalsen als deze op een of andere manier uw sleutelarchiefbestand voor ondertekening in handen krijgt of uw persoonlijke sleutel ontdekt.*

## Certificaten voor de ondertekening van code

De beveiligingsgaranties, beperkingen en wettelijke verplichtingen die betrekking hebben op het gebruik van certificaten voor ondertekening van programmacode worden beschreven in de CPS (Certificate Practice Statements) en gebruiksrechtovereenkomsten die zijn uitgegeven door de certificeringsinstantie. Voor meer informatie over de overeenkomsten voor de certificeringsinstanties die momenteel certificaten voor de ondertekening van AIR-code uitgeven, raadpleegt u:

ChosenSecurity (http://www.chosensecurity.com/products/tc_publisher_id_adobe_air.htm)

ChosenSecurity CPS (http://www.chosensecurity.com/resource_center/repository.htm)

GlobalSign (http://www.globalsign.com/code-signing/index.html)

GlobalSign CPS (http://www.globalsign.com/repository/index.htm)

Thawte CPS (http://www.thawte.com/cps/index.html)

VeriSign CPS (http://www.verisign.com/repository/CPS/)

VeriSign Subscriber's Agreement (https://www.verisign.com/repository/subscriber/SUBAGR.html)

## Informatie over ondertekening van AIR-programmacode

Wanneer een AIR-bestand wordt ondertekend, wordt een digitale handtekening opgenomen in het installatiebestand. De handtekening bevat een samenvatting van het pakket waarmee wordt geverifieerd of het AIR-bestand niet is gewijzigd sinds het werd ondertekend, en bevat informatie over het handtekeningcertificaat waarmee de identiteit van de uitgever wordt geverifieerd.

AIR maakt gebruik van PKI (Public Key Infrastructure, openbare-sleutelinfrastructuur) dat door het certificaatarchief van het besturingssysteem wordt ondersteund, om te bepalen of een certificaat kan worden vertrouwd. Om de gegevens van de uitgever te kunnen verifiëren moet de computer waarop een AIR-toepassing wordt geïnstalleerd het certificaat waarmee de AIR-toepassing wordt ondertekend rechtstreeks vertrouwen of een certificaatketen vertrouwen waarmee het certificaat aan een vertrouwde certificeringsinstantie wordt gekoppeld.

Als een AIR-bestand is ondertekend met een certificaat dat niet via een certificaatketen is gekoppeld aan een van de vertrouwde basiscertificaten (en dit betreft alle zelfondertekende certificaten), kunnen de gegevens van de uitgever niet worden geverifieerd. Hoewel AIR kan bepalen of het AIR-pakket niet is gewijzigd sinds het werd ondertekend, is niet bekend wie het bestand heeft gemaakt en ondertekend.

***Opmerking:*** *als een gebruiker een zelfondertekend certificaat vertrouwt, geven alle AIR-toepassingen die met het certificaat zijn ondertekend de waarde van de algemene naam (CN) in het certificaat weer als de uitgeversnaam. AIR beschikt niet over mogelijkheden waarmee een gebruiker een certificaat als vertrouwd kan aanmerken. Het certificaat (exclusief de persoonlijke sleutel) moet afzonderlijk aan de gebruiker worden gegeven en de gebruiker moet een van de mechanismen van het besturingssysteem of een geschikt hulpprogramma gebruiken om het certificaat te importeren op de juiste locatie in het certificaatarchief van het systeem.*

## Informatie over uitgevers-id's voor AIR

***Belangrijk:*** *vanaf AIR 1.5.3 is de uitgevers-id afgekeurd en wordt deze niet meer berekend op basis van het certificaat voor ondertekenen van code. Een uitgevers-id is niet meer nodig bij nieuwe toepassingen en moet niet meer worden gebruikt. Wanneer u bestaande toepassingen bijwerkt, moet u de oorspronkelijke uitgevers-id opgeven in het descriptorbestand van de toepassing.*

Voorafgaand aan versie AIR 1.5.3 werd door het installatieprogramma van de AIR-toepassing een uitgevers-id gegenereerd bij de installatie van een AIR-bestand. Dit was een unieke id voor het certificaat waarmee het AIR-bestand wordt ondertekend. Als u hetzelfde certificaat had gebruikt voor meerdere AIR-toepassingen, werd dezelfde uitgevers-id toegepast. De uitgevers-id werd gewijzigd doordat de update van een toepassing met een ander certificaat en soms zelfs met een vernieuwde instantie van het oorspronkelijke certificaat werd ondertekend.

Bij AIR 1.5.3 en hoger wordt er geen uitgevers-id toegewezen door AIR. Een toepassing die is gepubliceerd met AIR 1.5.3 kan een tekenreeks voor de uitgevers-id opgeven in de toepassingsdescriptor. U moet alleen een uitgevers-id opgeven wanneer u updates voor toepassingen publiceert die oorspronkelijk zijn gepubliceerd voor versies van AIR die voorafgaan aan versie 1.5.3. Als u de oorspronkelijke id niet opgeeft in de toepassingsdescriptor, wordt het nieuwe AIR-pakket niet behandeld als een update van de bestaande toepassing.

Als u de oorspronkelijke gebruikers-id wilt bepalen, moet u zoeken naar het bestand `publisherid` in de submap META-INF/AIR waar de oorspronkelijke toepassing is geïnstalleerd. De uitgevers-id wordt aangegeven door de tekenreeks in dit bestand. Als u de uitgevers-id handmatig wilt kunnen opgeven, moet versie 1.5.3 (of hoger) van de AIR-runtime worden opgeven in de naamruimtedeclaratie van het descriptorbestand van de toepassing.

Indien aanwezig wordt de uitgevers-id gebruikt voor de volgende doeleinden:

- Als onderdeel van de coderingssleutel voor de gecodeerde lokale opslagruimte
- Als onderdeel van het pad voor de opslagmap van de toepassing
- Als onderdeel van de verbindingstekenreeks voor lokale verbindingen
- Als onderdeel van de identiteitstekenreeks waarmee een toepassing wordt aangeroepen met de AIR in-browser API
- Als onderdeel van de OSID (gebruikt bij het maken van aangepaste programma's voor het installeren/verwijderen van software)

Wanneer u een uitgevers-id wijzigt, wordt ook het gedrag gewijzigd van de AIR-functies die afhankelijk zijn van de id. Zo zijn de gegevens in de bestaande gecodeerde lokale opslagruimte niet meer toegankelijk en moeten alle Flash- of AIR-instanties die een lokale verbinding maken met de toepassing ook gebruikmaken van de nieuwe id in de verbindingstekenreeks. De uitgever-id voor een geïnstalleerde toepassing kan niet worden gewijzigd in AIR 1.5.3 of later. Als u een andere uitgevers-id gebruikt wanneer u een AIR-pakket publiceert, wordt het nieuwe pakket door het installatieprogramma behandeld als een andere toepassing, en niet als een update.

## Informatie over certificaatindelingen

De ondertekeningsprogramma's van AIR accepteren alle sleutelarchieven die toegankelijk zijn via JCA (Java Cryptography Architecture). Hiertoe behoren op bestanden gebaseerde sleutelarchieven, zoals PKCS12-bestanden (die gewoonlijk de extensie .pfx of .p12 hebben), sleutelarchiefbestanden van Java, PKCS11-hardwaresleutelarchieven en de sleutelarchieven van het systeem. Welke sleutelarchiefindelingen ADT kan openen, is afhankelijk van de versie en de configuratie van de Java-runtime waarmee ADT wordt uitgevoerd. Voor sommige typen sleutelarchieven, zoals PKCS11-hardwaretokens, moeten mogelijk extra stuurprogramma's en JCA-invoegtoepassingen worden geïnstalleerd en geconfigureerd.

Als u AIR-bestanden wilt ondertekenen, kunt u de meeste bestaande certificaten voor ondertekening van code gebruiken, of u kunt een nieuwe verkrijgen die speciaal voor de ondertekening van AIR-toepassingen is uitgegeven. U kunt bijvoorbeeld alle volgende typen certificaten gebruiken van VeriSign, Thawte, GlobalSign of ChosenSecurity:

- ChosenSecurity
  - TC Publisher ID for Adobe AIR
- GlobalSign
  - ObjectSign Code Signing Certificate

- Thawte:
  - AIR-certificaat voor ontwikkelaars
  - Apple-certificaat voor ontwikkelaars
  - JavaSoft-certificaat voor ontwikkelaars
  - Microsoft Authenticode-certificaat
- VeriSign:
  - Adobe AIR Digital ID
  - Digitale Microsoft Authenticode-id
  - Digitale handtekening-id van Sun Java

***Opmerking:*** *het certificaat moet zijn gemaakt voor het ondertekenen van code. U kunt geen SSL of ander type certificaat gebruiken om AIR-bestanden te ondertekenen.*

## Tijdstempels

Wanneer u een AIR-bestand ondertekent, wordt bij de server van een tijdstempelinstantie gevraagd om een datum en tijd van ondertekening die onafhankelijk te verifiëren zijn. Het tijdstempel wordt ingesloten in het AIR-bestand. Wanneer het handtekeningcertificaat geldig is op het moment van ondertekenen, kan het AIR-bestand worden geïnstalleerd, zelfs nadat het certificaat is verlopen. Als er geen tijdstempel wordt opgehaald, kan het AIR-bestand niet meer worden geïnstalleerd wanneer het certificaat is verlopen of is ingetrokken.

Standaard wordt een tijdstempel opgehaald wanneer een AIR-pakket wordt gemaakt. U kunt het ophalen van een tijdstempel echter uitschakelen, zodat u toepassingspakketten kunt maken wanneer de tijdstempelservice niet beschikbaar is. Adobe raadt u aan in alle openbaar gedistribueerde AIR-bestanden een tijdstempel op te nemen.

De standaardinstantie voor tijdstempels die voor AIR-pakketten wordt gebruikt, is Geotrust.

## Certificaten ophalen

Als u een certificaat nodig hebt, gaat u gewoonlijk naar de website van een certificeringsinstantie en voltooit u het aankoopproces van dat bedrijf. Welke hulpprogramma's nodig zijn om het sleutelarchiefbestand te maken, is afhankelijk van het type van het gekochte certificaat, hoe het certificaat op de ontvangende computer wordt opgeslagen en, in sommige gevallen, de browser waarmee het certificaat is opgehaald. Als u bijvoorbeeld een Adobe Developer-certificaat wilt verkrijgen en exporteren van Thawte, moet u Mozilla Firefox gebruiken. Het certificaat kan vervolgens rechtstreeks vanuit de gebruikersinterface van Internet Explorer worden geëxporteerd als een .p12- of .pfx-bestand.

***Opmerking:*** *bij Java versie 1.5 en later worden hoge ASCII-tekens in wachtwoorden voor de beveiliging van PKCS12-certificatiebestanden niet geaccepteerd. Java wordt gebruikt door de AIR-ontwikkelingsprogramma's bij het maken van ondertekende AIR-pakketten. Wanneer u het certificaat exporteert als een .p12 of .pfx-bestand, moet u alleen normale ASCII-tekens gebruiken in het wachtwoord.*

U kunt een zelfondertekend certificaat genereren met het hulpprogramma ADT (Air Development Tool) waarmee AIR-toepassingsbestanden in een pakket worden geplaatst. U kunt ook sommige hulpprogramma's van derden gebruiken.

Zie “AIR Developer Tool (ADT)” op pagina 173 voor instructies voor het genereren van een zelfondertekend certificaat en het ondertekenen van een AIR-bestand. U kunt AIR-bestanden ook exporteren en ondertekenen met Flash Builder, Dreamweaver en de AIR-update voor Flash.

In het volgende voorbeeld ziet u hoe u een AIR-certificaat voor ontwikkelaars ophaalt bij de certificeringsinstantie Thawte en dit voorbereidt voor gebruik met ADT.

## Voorbeeld: Een AIR-certificaat voor ontwikkelaars ophalen bij Thawte

***Opmerking:*** *dit voorbeeld bevat slechts een van de vele manieren waarop u een certificaat voor ondertekening van programmacode kunt ophalen en voorbereiden. Elke certificeringsinstantie heeft zijn eigen beleid en procedures.*

Als u een AIR-certificaat voor ontwikkelaars wilt aanschaffen, moet u de website van Thawte bezoeken met de browser Mozilla Firefox. De persoonlijke sleutel voor het certificaat wordt opgeslagen in het sleutelarchief van de browser. Zorg ervoor dat het Firefox-sleutelarchief is beveiligd met een hoofdwachtwoord en dat de computer zelf fysiek is beveiligd. (U kunt het certificaat en de persoonlijke sleutel uit het sleutelarchief van de browser exporteren en verwijderen nadat het aankoopproces is voltooid.)

Tijdens het certificaatinschrijvingsproces wordt een combinatie van een persoonlijke en een openbare sleutel gegenereerd. De persoonlijke sleutel wordt automatisch opgeslagen in het sleutelarchief van Firefox. Voor het aanvragen en het ophalen van het certificaat op de website van Thawte moet u dezelfde computer en browser gebruiken.

1. Ga op de website van Thawte naar de productpagina voor certificaten voor ondertekening van programmacode.
2. Selecteer Adobe AIR Developer Certificate in de lijst met certificaten voor ondertekening van programmacode.
3. Voltooi de drie stappen van het inschrijvingsproces. U moet gegevens van uw organisatie en een contactpersoon opgeven. Thawte verifieert vervolgens uw identiteit en kan om aanvullende informatie vragen. Nadat de verificatie is voltooid, stuurt Thawte u een e-mailbericht met instructies voor het ophalen van het certificaat.

   ***Opmerking:*** *Meer informatie over het type documentatie dat nodig is, kunt u hier vinden: https://www.thawte.com/ssl-digital-certificates/free-guides-whitepapers/pdf/enroll_codesign_eng.pdf.*
4. Haal het uitgegeven certificaat op vanaf de site van Thawte. Het certificaat wordt automatisch opgeslagen in het sleutelarchief van Firefox.
5. Exporteer op de volgende manier een sleutelarchiefbestand dat de persoonlijke sleutel en het certificaat uit het sleutelarchief van Firefox bevat:

   ***Opmerking:*** *Wanneer u de persoonlijke sleutel en het certificaat vanuit Firefox exporteert, worden deze geëxporteerd naar een P12-indeling (.pfx) die in ADT, Flex, Flash en Dreamweaver kan worden gebruikt.*

   a. Open in Firefox het dialoogvenster *Certificatenbeheerder*:
   b. In Windows: Open Extra -> Opties -> Geavanceerd -> Codering -> Certificaten weergeven
   c. In Mac OS: Open Firefox-> Voorkeuren-> Geavanceerd -> Codering -> Certificaten weergeven
   d. In Linux: Open Bewerken-> Voorkeuren-> Geavanceerd -> Codering -> Certificaten weergeven
   e. Selecteer het Adobe AIR-certificaat voor ondertekening van programmacode in de lijst met certificaten en klik op de knop **Reservekopie maken**.
   f. Voer een bestandsnaam en een locatie in voor het te exporteren sleutelarchiefbestand en klik op **Opslaan**.
   g. Als u het hoofdwachtwoord van Firefox gebruikt, moet u uw wachtwoord voor het softwarebeveiligingsapparaat invoeren om het bestand te exporteren. (Dit wachtwoord wordt alleen door Firefox gebruikt.)
   h. Maak in het dialoogvenster *Wachtwoord voor reservekopie van certificaat kiezen* een wachtwoord voor het sleutelarchiefbestand.

      ***Belangrijk:*** *dit wachtwoord beveiligt het sleutelarchiefbestand en is vereist wanneer het bestand wordt gebruikt voor het ondertekenen van AIR-toepassingen. Kies een veilig wachtwoord.*

   i. Klik op OK. U ontvangt een bericht wanneer de reservekopie is gemaakt. Het sleutelarchiefbestand dat de persoonlijke sleutel en het certificaat bevat, wordt opgeslagen met de extensie .p12 (in de PKCS12-indeling).
6. Gebruik het geëxporteerde sleutelarchiefbestand in ADT, Flash Builder, Flash Professional of Dreamweaver. Het wachtwoord dat voor het bestand is gemaakt, is vereist wanneer een AIR-toepassing wordt ondertekend.

***Belangrijk:*** *de persoonlijke sleutel en het certificaat blijven opgeslagen in het sleutelarchief van Firefox. Hoewel u daardoor een extra kopie van het certificaatbestand kunt exporteren, is dit ook een toegangspunt dat u moet beveiligen om de veiligheid van uw certificaat en de persoonlijke sleutel te garanderen.*

## Certificaten wijzigen

In sommige gevallen moet u het certificaat wijzigen waarmee u de updates voor uw AIR-toepassing ondertekent. Bijvoorbeeld:

- Bij het vernieuwen van het oorspronkelijke certificaat voor ondertekenen.
- U wilt een zelfondertekend certificaat wijzigen in een certificaat dat door een ondertekeningsinstantie is uitgegeven.
- U wilt een zelfondertekend certificaat dat bijna is verlopen, wijzigen in een ander certificaat.
- U wilt een commercieel certificaat wijzigen in een ander commercieel certificaat, bijvoorbeeld wanneer de identiteit van uw bedrijf is gewijzigd.

Als u wilt dat een AIR-bestand door AIR wordt herkend als een update, moet u zowel de oorspronkelijke als de bijgewerkte versie van de AIR-bestanden met hetzelfde certificaat ondertekenen, of u moet een certificaat met migratiehandtekening toepassen op de update. Een migratiehandtekening is een tweede handtekening die wordt toegepast op het AIR-updatepakket met behulp van het oorspronkelijke certificaat. De migratiehandtekening gebruikt het oorspronkelijke certificaat, waardoor wordt vastgelegd dat de ondertekenaar de oorspronkelijke uitgever van de toepassing is.

Nadat een AIR-bestand met een migratiehandtekening is geïnstalleerd, wordt het nieuwe certificaat het primaire certificaat. Voor alle volgende updates is er geen migratiehandtekening vereist. U kunt migratiehandtekeningen het beste zo lang mogelijk toepassen om tegemoet te komen aan gebruikers die sommige updates overslaan.

***Belangrijk:*** *u moet voordat het verloopt het certificaat wijzigen en een migratiehandtekening toepassen op de update van het oorspronkelijke certificaat. Als dit niet plaatsvindt, moeten gebruikers hun bestaande versie van uw toepassing verwijderen voordat zij een nieuwe versie installeren. Voor AIR 1.5.3 of later kunt u binnen een respijtperiode van 365 dagen na het verlopen een migratiehandtekening toepassen met een verlopen certificaat. Met het verlopen certificaat kunt u echter de handtekening van de hoofdtoepassing niet toepassen.*

Certificaten wijzigen:

1. Maak een update voor uw toepassing.
2. Plaats het AIR-updatebestand in een pakket en onderteken het met het **nieuwe** certificaat.
3. Onderteken het AIR-bestand opnieuw met het **oorspronkelijke** certificaat (met de ADT-opdracht `-migrate`).

Een AIR-bestand met een migratiehandtekening is verder hetzelfde als een normaal AIR-bestand. Als de toepassing wordt geïnstalleerd op een systeem zonder de oorspronkelijke versie, wordt de nieuwe versie op de gebruikelijke manier geïnstalleerd.

***Opmerking:*** *voorafgaand aan AIR 1.5.3 was er voor het ondertekenen van een AIR-toepassing met een vernieuwd certificaat niet altijd een migratiehandtekening vereist. In AIR 1.5.3 en hogere versies is er altijd een migratiehandtekening vereist voor vernieuwde certificaten.*

Als u een migratiehandtekening wilt toepassen, gebruikt u de "ADT-opdracht voor migreren" op pagina 181, zoals beschreven in "Een bijgewerkte versie van een AIR-toepassing ondertekenen." op pagina 209.

***Opmerking:*** *de ADT-opdracht voor migreren kan niet worden gebruikt met AIR-bureaubladtoepassingen die native extensies bevatten, omdat deze toepassingen in een pakket zijn opgenomen als native installatieprogramma's, en niet als AIR-bestanden. Als u certificaten wilt wijzigen voor een AIR-bureaubladtoepassing die een native extensie bevat, neemt u de toepassing op in een pakket met behulp van de "ADT-opdracht voor verpakken" op pagina 174 met de markering -migrate.*

**Gewijzigde toepassingsidentiteit**

Voorafgaand aan AIR 1.5.3 werd de identiteit van de AIR-toepassing gewijzigd bij het installeren van een update die was ondertekend met een migratiehandtekening. Het wijzigen van de identiteit van een toepassing heeft verschillende gevolgen, zoals:

- De nieuwe versie van de toepassing heeft geen toegang tot gegevens in het bestaande gecodeerde lokale archief.
- De locatie van de opslagmap van de toepassing wordt gewijzigd. Gegevens op de oude locatie worden niet naar de nieuwe map gekopieerd. (Maar de nieuwe toepassing kan de oorspronkelijke locatie wel vinden op basis van de oude uitgevers-id.)
- De toepassing kan geen lokale verbindingen meer openen met de oude uitgevers-id.
- De identiteitstekenreeks waarmee een toepassing toegankelijk wordt vanaf een webpagina verandert.
- De OSID van de toepassing verandert. (De OSID wordt gebruikt bij het schrijven van aangepaste programma's voor het installeren en verwijderen van software.)

Bij het publiceren van een update met AIR 1.5.3 of later kan de identiteit van de toepassing niet worden gewijzigd. De oorspronkelijke toepassings-id en uitgevers-id moeten worden opgegeven in de toepassingsdescriptor van het AIR-bestand voor de update. Als dat niet het geval is, wordt het nieuwe pakket niet herkend als een update.

***Opmerking:*** *bij het publiceren van een nieuwe AIR-toepassing met AIR 1.5.3 of hoger, moet u geen uitgevers-id opgeven.*

## Terminologie

Deze sectie bevat een woordenlijst met een aantal belangrijke termen die u moet begrijpen wanneer u beslissingen neemt over de manier waarop u uw toepassing ondertekent voor openbare distributie.

| Term | Beschrijving |
|---|---|
| Certificeringsinstantie (CA) | Een entiteit in een netwerk met een openbare-sleutelinfrastructuur die optreedt als vertrouwde derde partij en de identiteit van de eigenaar van een openbare sleutel certificeert. Een CA geeft gewoonlijk digitale certificaten uit, die met een eigen persoonlijke sleutel zijn ondertekend, om te verklaren dat de identiteit van de certificaathouder is geverifieerd. |
| CPS (Certificate Practice Statement) | Bevat de procedures en het beleid van de certificeringsinstantie voor het uitgeven en verifiëren van certificaten. De CPS is opgenomen in het contract tussen de CA en haar abonnees en afhankelijke partijen. Hierin wordt ook het beleid voor identiteitsverificatie beschreven en het garantieniveau van de uitgegeven certificaten. |
| CRL (Certificate Revocation List) | Een lijst met uitgegeven certificaten die zijn ingetrokken en niet meer mogen worden vertrouwd. AIR raadpleegt de CRL op het moment dat een AIR-toepassing wordt ondertekend, en als er geen tijdstempel aanwezig is, opnieuw wanneer de toepassing wordt geïnstalleerd. |
| Certificaatketen | Een certificaatketen is een reeks certificaten waarbij elk certificaat is ondertekend door het volgende certificaat. |
| Digitaal certificaat | Een digitaal document dat informatie bevat over de identiteit van de eigenaar, de openbare sleutel van de eigenaar en de identiteit van het certificaat zelf. Een certificaat dat door een certificeringsinstantie is uitgegeven, is zelf ondertekend door een certificaat dat eigendom is van de uitgevende certificeringsinstantie. |

| Term | Beschrijving |
|---|---|
| Digitale handtekening | Een gecodeerd bericht of gecodeerde samenvatting die alleen kan worden gedecodeerd met de openbare sleutel van een sleutelpaar met een openbaar en een persoonlijk gedeelte. In een PKI bevat een digitale handtekening een of meer digitale certificaten die uiteindelijk te traceren zijn tot de certificeringsinstantie. Een digitale handtekening kan worden gebruikt om te valideren of een bericht (of computerbestand) niet is veranderd sinds het werd ondertekend (binnen de zekerheidsbeperkingen van de gebruikte cryptografiealgoritme) en om de identiteit van de ondertekenaar te valideren (aangenomen dat de uitgevende certificeringsinstantie wordt vertrouwd). |
| Sleutelarchief | Een database die digitale certificaten bevat en soms ook de bijbehorende persoonlijke sleutels. |
| JCA (Java Cryptography Architecture) | Een uitbreidbare architectuur voor het beheer van en de toegang tot sleutelarchieven. Zie de Java Cryptography Architecture Reference Guide voor meer informatie. |
| PKCS #11 | De interfacestandaard van RSA Laboratories voor cryptografietokens. Een sleutelarchief met hardwaretokens. |
| PKCS #12 | De standaardsyntaxis van RSA Laboratories voor uitwisseling van persoonlijke gegevens. Een sleutelarchief met bestanden die gewoonlijk een persoonlijke sleutel en het bijbehorende digitale certificaat bevatten. |
| Persoonlijke sleutel | De persoonlijke helft van een asymmetrisch cryptografiesysteem dat uit een openbare en een persoonlijke sleutel bestaat. De persoonlijke sleutel moet geheim blijven en mag nooit via een netwerk worden verzonden. Digitaal ondertekende berichten worden door de ondertekenaar gecodeerd met de persoonlijke sleutel. |
| Openbare sleutel | De openbare helft van een asymmetrisch cryptografiesysteem dat uit een openbare en een persoonlijke sleutel bestaat. De openbare sleutel is algemeen beschikbaar en wordt gebruikt voor het decoderen van berichten die met de persoonlijke sleutel zijn gecodeerd. |
| PKI (Public Key Infrastructure) | Een systeem van vertrouwen waarin certificeringsinstanties de identiteit van de eigenaren van openbare sleutels bevestigen. Clients van het netwerk vertrouwen erop dat de digitale certificaten die door een vertrouwde CA zijn uitgegeven de identiteit van de ondertekenaar van een digitaal bericht (of bestand) garanderen. |
| Tijdstempel | Digitaal ondertekende informatie die de datum en tijd bevat waarop een gebeurtenis is opgetreden. ADT kan een tijdstempel van een RFC 3161-compatibele tijdserver opnemen in een AIR-pakket. Wanneer een tijdstempel aanwezig is, wordt dit gebruikt om de geldigheid van een certificaat vast te stellen op het moment van ondertekenen. Hierdoor kan een AIR-toepassing worden geïnstalleerd nadat het handtekeningcertificaat is verlopen. |
| Tijdstempelinstantie | Een instantie die tijdstempels uitgeeft. Om door AIR te worden herkend, moet een tijdstempel voldoen aan RFC 3161 en moet de handtekening van het tijdstempel via een certificaatketen kunnen worden gekoppeld aan een vertrouwd basiscertificaat op de installatiecomputer. |

## iOS-certificaten

De certificaten voor het ondertekenen van de code die door Apple worden uitgegeven, worden gebruikt voor het ondertekenen van iOS-toepassingen, inclusief toepassingen die zijn ontwikkeld met Adobe AIR. Wanneer u een toepassing op testapparatuur wilt installeren, wordt het toepassen van een handtekening met een Apple-ontwikkelingscertificaat vereist. Bij het distribueren van de voltooide toepassing wordt het toepassen van een handtekening met een distributiecertificaat vereist.

Wanneer u een toepassing wilt ondertekenen, wordt door ATD toegang vereist tot het certificaat voor het ondertekenen van de code en de bijbehorende persoonlijke sleutel. In het certificaatbestand zelf is geen persoonlijke sleutel opgenomen. U moet een sleutelarchief maken in de vorm van een uitwisselingsbestand van persoonlijke gegevens (P12-bestand of .pfx) waarin het certificaat en de persoonlijke sleutel zijn opgenomen. Zie “Een ontwikkelingscertificaat omzetten in een P12-sleutelarchief-bestand” op pagina 207.

## Een certificaataanvraag genereren

U verkrijgt een ontwikkelingscertificaat door een certificaataanvraag te genereren en in te dienen op de Apple iOS Provisioning Portal.

Door het proces voor certificaataanvraag wordt een openbare-persoonlijke sleutelcombinatie gegenereerd. De persoonlijke sleutel blijft op uw computer. U verzendt de ondertekeningsaanvraag met de openbare sleutel en uw identificatiegegevens naar Apple, die fungeert als certificeringsinstantie. Apple ondertekent uw certificaat met het World Wide Developer Relations-certificaat van Apple.

### Een certificaataanvraag genereren op Mac OS

In Mac OS genereert u een certificaataanvraag met de toepassing Sleutelhangertoegang. De toepassing Sleutelhangertoegang bevindt zich in de submap Hulpprogramma's van de map Toepassingen. Instructies voor het genereren van de certificaataanvraag zijn beschikbaar op de Apple iOS Provisioning Portal.

### Een certificaataanvraag genereren in Windows

Voor Windows-ontwikkelaars is het wellicht het eenvoudigst om het iPhone-certificaat voor ontwikkelaars voor een Mac-computer te verkrijgen. Het is echter ook mogelijk om een certificaat te verkrijgen voor een Windows-computer. U maakt eerst een CSR-bestand (Certificate Signing Request) met OpenSSL:

1. Installeer OpenSSL op de Windows-computer. (Ga naar http://www.openssl.org/related/binaries.html.)

   U moet wellicht ook de Visual C++ 2008 Redistributable-bestanden op de Open SSL-downloadpagina installeren. (Visual C++ hoeft *niet* op uw computer te staan.)

2. Open een Windows-opdrachtregelsessie en ga naar de bin-map van OpenSSL (bijvoorbeeld c:\OpenSSL\bin\).

3. Maak de persoonlijke sleutel door het volgende op de opdrachtregel op te geven:

   ```
   openssl genrsa -out mykey.key 2048
   ```

   Sla dit bestand met de persoonlijke sleutel op. U hebt het later nodig.

   Negeer foutberichten niet wanneer u OpenSSL gebruikt. Ook als OpenSSL een foutbericht genereert, voert het wellicht nog bestanden uit. Deze bestanden zijn dan echter waarschijnlijk niet bruikbaar. Als u fouten ziet, controleert u de syntaxis en voert u opdracht nogmaals uit.

4. Maak het CSR-bestand door het volgende op de opdrachtregel op te geven:

   ```
   openssl req -new -key mykey.key -out CertificateSigningRequest.certSigningRequest  -subj
   "/emailAddress=yourAddress@example.com, CN=John Doe, C=US"
   ```

   Vervang de waarden voor het e-mailadres, CN (certificaatnaam) en C (land) door uw eigen waarden.

5. Upload het CSR-bestand naar Apple op de website voor iPhone-ontwikkelaars. (Zie "Een iPhone-certificaat voor ontwikkelaars aanvragen en een inrichtingsprofiel maken".)

## Een ontwikkelingscertificaat omzetten in een P12-sleutelarchief-bestand

Wanneer u een P12-sleutelarchief wilt maken, moet u in één bestand uw Apple ontwikkelingscertificaat combineren met de bijbehorende persoonlijke sleutel. Het proces voor het maken van een sleutelarchiefbestand is afhankelijk van de methode die u hebt gebruikt om de oorspronkelijke certificaataanvraag te genereren en van de locatie waar de persoonlijke sleutel is opgeslagen.

**Het iPhone-certificaat voor ontwikkelaars omzetten naar een P12-bestand in Mac OS**

Als u het Apple iPhone-certificaat bij Apple hebt gedownload, exporteert u het certificaat naar de P12-sleutelarchiefindeling. Ga in Mac® OS als volgt te werk:

1. Open de toepassing Sleutelhangertoegang (in de map Toepassingen/Hulpprogramma's).
2. Als u het certificaat nog niet aan Keychain hebt toegevoegd, selecteert u Bestand > Importeren. Navigeer naar het certificaatbestand (*.cer) dat u van Apple hebt ontvangen.
3. Selecteer de sleutelcategorie in Sleutelhangertoegang.
4. Selecteer de persoonlijke sleutel die aan het ontwikkelingscertificaat van uw iPhone is gekoppeld.

   De persoonlijke sleutel wordt door de iPhone-ontwikkelaar geïdentificeerd: <Voornaam> <Achternaam> openbaar certificaat dat aan een de sleutel is gekoppeld.
5. Houd Command ingedrukt en klik op het iPhone-certificaat voor ontwikkelaars en selecteer *"iPhone Developer: Name..."(iPhone-ontwikkelaar: naam...) exporteren*.
6. Sla het sleutelarchief op in de P12-bestandsindeling (Personal Information Exchange).
7. U wordt gevraagd een wachtwoord te maken dat wordt gebruikt wanneer u het sleutelarchief gebruikt om toepassingen te ondertekenen of voor het overbrengen van de sleutel en certificaat van dit sleutelarchief naar een ander sleutelarchief.

**Een Apple-certificaat voor ontwikkelaars omzetten in een P12-bestand voor Windows**

Voor het ontwikkelen van AIR voor iOS-toepassingen moet u een P12-certificaatbestand gebruiken. U genereert dit certificaat op basis van het Apple iPhone-certificaatbestand voor ontwikkelaars dat u van Apple ontvangt.

1. Zet het certificaatbestand voor ontwikkelaars dat u van Apple ontvangt, om in een PEM-certificaatbestand. Voer de volgende opdrachtregelinstructie uit vanuit de OpenSSL-bin-map:

   ```
   openssl x509 -in developer_identity.cer -inform DER -out developer_identity.pem -outform PEM
   ```
2. Als u de persoonlijke sleutel van de sleutelhanger op een Mac-computer gebruikt, zet u de sleutel om in een PEM-sleutel:

   ```
   openssl pkcs12 -nocerts -in mykey.p12 -out mykey.pem
   ```
3. U kunt nu een geldig P12-bestand genereren op basis van de sleutel en de PEM-versie van het iPhone-certificaat voor ontwikkelaars:

   ```
   openssl pkcs12 -export -inkey mykey.key -in developer_identity.pem -out iphone_dev.p12
   ```

   Als u een sleutel gebruikt van een Mac OS-sleutelhanger, gebruikt u de PEM-versie die u in de vorige stap hebt gegenereerd. Gebruik in het andere geval de OpenSSL-sleutel die u eerder hebt gegenereerd (op de Windows-computer).

# Niet-ondertekende tussentijdse AIR-bestanden maken met ADT

Gebruik de opdracht `-prepare` om een niet-ondertekend tussentijds AIR-bestand te maken. Een tussentijds AIR-bestand moet met de ADT-opdracht `-sign` zijn ondertekend om een geldig AIR-installatiebestand te produceren.

De opdracht `-prepare` heeft dezelfde vlaggen en parameters als de opdracht `-package` (met uitzondering van de ondertekeningsopties). Het enige verschil is dat het uitvoerbestand niet wordt ondertekend. Het tussentijdse bestand wordt gegenereerd met de bestandsextensie `airi`.

Gebruik de ADT-opdracht `-sign` om een tussentijds AIR-bestand te ondertekenen. (Zie "ADT-opdracht voor voorbereiden" op pagina 181.)

**ADT - voorbeeld van de opdracht -prepare**

```
adt -prepare unsignedMyApp.airi myApp.xml myApp.swf components.swc
```

# Tussentijdse AIR-bestanden ondertekenen met ADT

Gebruik de opdracht `-sign` om een tussentijds AIR-bestand te ondertekenen met ADT. De ondertekeningsopdracht werkt alleen met tussentijdse AIR-bestanden (extensie `airi`). Een AIR-bestand kan maar één keer worden ondertekend.

Gebruik de ADT-opdracht `-prepare` om een tussentijds AIR-bestand te maken. (Zie "ADT-opdracht voor voorbereiden" op pagina 181.)

**AIRI-bestanden ondertekenen**

❖ Gebruik de ADT-opdracht `-sign` met de volgende syntaxis:

```
adt -sign SIGNING_OPTIONS airi_file air_file
```

**signING_OPTIONS** Met de ondertekeningsopties bepaalt u de persoonlijke sleutel en het certificaat waarmee het AIR-bestand wordt ondertekend. Deze opties worden beschreven in "ADT-opties voor codeondertekening" op pagina 188.

**airi_file** Het pad naar het niet-ondertekende tussentijdse AIR-bestand dat moet worden ondertekend.

**air_file** De naam van het AIR-bestand dat moet worden gemaakt.

**ADT - voorbeeld van de opdracht -sign**

```
adt -sign -storetype pkcs12 -keystore cert.p12 unsignedMyApp.airi myApp.air
```

Zie voor meer informatie de "ADT-opdracht voor ondertekenen" op pagina 181.

# Een bijgewerkte versie van een AIR-toepassing ondertekenen.

Telkens wanneer u een bijgewerkte versie van een bestaande AIR-toepassing maakt, ondertekent u de bijgewerkte toepassing. In het beste geval kunt u hetzelfde certificaat gebruiken om de bijgewerkte versie te ondertekenen als het certificaat waarmee u de vorige versie hebt ondertekend. In dit geval verloopt het ondertekenen op dezelfde manier als toen u de toepassing voor het eerst ondertekende.

Als het certificaat dat u hebt gebruikt om de vorige versie te ondertekenen, is verlopen en werd vernieuwd of vervangen, kunt u het vernieuwde of het nieuwe (vervangende) certificaat gebruiken om de bijgewerkte versie te ondertekenen. Om dit te doen, ondertekent u de toepassing met het nieuwe certificaat *en* past u een migratiehandtekening toe met behulp van het oorspronkelijke certificaat. Door de migratiehandtekening wordt bevestigd dat de eigenaar van het oorspronkelijke certificaat de update heeft gepubliceerd.

Voordat u een migratiehandtekening toepast, moet u de volgende punten in overweging nemen:

- Als u een migratiehandtekening wilt toepassen, moet het oorspronkelijke certificaat nog geldig zijn of niet meer dan 365 dagen zijn verlopen. Deze periode wordt 'respijtperiode' genoemd en de duur ervan kan in de toekomst worden gewijzigd.

  ***Opmerking:*** *tot aan AIR 2.6 duurde de respijtperiode 180 dagen.*

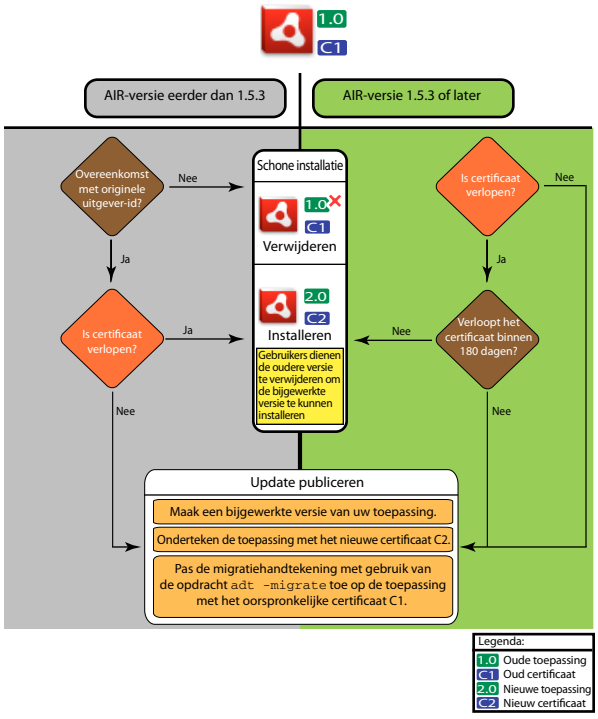
- U mag geen migratiehandtekening toepassen nadat het certificaat is verlopen en de respijtperiode van 365 dagen is verstreken. In dit geval moeten gebruikers de bestaande versie verwijderen voordat ze de bijgewerkte versie installeren.
- De periode van 365 dagen geldt alleen voor toepassingen waarbij AIR versie 1.5.3. of hoger is opgegeven in de naamruimte van de toepassingsdescriptor.

***Belangrijk:*** *updates ondertekenen met migratiehandtekeningen van verlopen certificaten is een tijdelijke oplossing. Voor een uitgebreide oplossing maakt u een gestandaardiseerde ondertekeningsworkflow voor het beheer van de implementatie van toepassingsupdates. U kunt bijvoorbeeld elke update ondertekenen met het meest recente certificaat en een migratiecertificaat toepassen met behulp van het certificaat dat wordt gebruikt om de vorige update te ondertekenen (indien van toepassing). Upload elke update naar zijn eigen URL vanwaar gebruikers de toepassing kunnen downloaden. Zie "Workflow voor toepassingsupdates ondertekenen" op pagina 272 voor meer informatie.*

De volgende tabel en afbeelding vat de workflow voor migratiehandtekeningen samen:

| **Scenario** | **Status van oorspronkelijk certificaat** | **Actie van ontwikkelaar** | **Actie van gebruiker** |
|---|---|---|---|
| Toepassing gebaseerd op runtimeversie 1.5.3 of hoger van Adobe AIR | Geldig | De nieuwste versie van de AIR-toepassing publiceren | Geen actie vereist<br>Toepassing voert automatisch een upgrade uit |
| | Verlopen, maar binnen de respijtperiode van 365 dagen | Onderteken de toepassing met het nieuwe certificaat. Pas een migratiehandtekening toe met behulp van het verlopen certificaat. | Geen actie vereist<br>Toepassing voert automatisch een upgrade uit |
| | Verlopen en valt niet binnen de respijtperiode | U kunt de migratiehandtekening niet toepassen op de update van de AIR-toepassing.<br>U moet in plaats daarvan een andere versie van de AIR-toepassing publiceren met behulp van een nieuw certificaat. Gebruikers kunnen de nieuwe versie installeren nadat zij de bestaande versie van de AIR-toepassing hebben verwijderd. | De huidige versie van de AIR-toepassing verwijderen en de nieuwste versie installeren |

| Scenario | Status van oorspronkelijk certificaat | Actie van ontwikkelaar | Actie van gebruiker |
|---|---|---|---|
| • Toepassing is gebaseerd op runtimeversie 1.5.2 of lager van Adobe AIR<br>• Uitgever-id in de toepassingsdescriptor van de update stemt overeen met de uitgever-id van de vorige versie | Geldig | De nieuwste versie van de AIR-toepassing publiceren | Geen actie vereist<br>Toepassing voert automatisch een upgrade uit |
| | Verlopen en valt niet binnen de respijtperiode | U kunt de migratiehandtekening niet toepassen op de update van de AIR-toepassing.<br>U moet in plaats daarvan een andere versie van de AIR-toepassing publiceren met behulp van een nieuw certificaat. Gebruikers kunnen de nieuwe versie installeren nadat zij de bestaande versie van de AIR-toepassing hebben verwijderd. | De huidige versie van de AIR-toepassing verwijderen en de nieuwste versie installeren |
| • Toepassing is gebaseerd op runtimeversie 1.5.2 of lager van Adobe AIR<br>• Uitgever-id in de toepassingsdescriptor van de update **stemt niet overeen** met de uitgever-id van de vorige versie | Alle | De AIR-toepassing ondertekenen met behulp van een geldig certificaat en de nieuwste versie van de AIR-toepassing publiceren | De huidige versie van de AIR-toepassing verwijderen en de nieuwste versie installeren |




*Ondertekeningsworkflow voor updates*

## AIR-toepassingen migreren voor het gebruik van een nieuw certificaat

U kunt als volgt een AIR-toepassing migreren naar een nieuw certificaat terwijl de toepassing wordt bijgewerkt:

1. Maak een update voor uw toepassing.
2. Plaats het AIR-updatebestand in een pakket en onderteken het met het **nieuwe** certificaat.
3. Onderteken het AIR-bestand opnieuw met het **oorspronkelijke** certificaat en de opdracht `-migrate`.

Een AIR-bestand dat wordt ondertekend met de opdracht `-migrate`, kan ook worden gebruikt om een nieuwe versie van de toepassing te installeren en kan bovendien ook worden gebruikt om elke vorige versie die werd ondertekend met het oude certificaat, bijwerken.

***Opmerking:*** *wanneer u een toepassing bijwerkt die is gepubliceerd voor een eerdere versie dan AIR 1.5.3, moet u de oorspronkelijke uitgevers-id opgeven in de toepassingsdescriptor. Als dit niet gebeurt, moeten gebruikers van uw toepassing de eerdere versie verwijderen voordat ze de update kunnen installeren.*

Gebruik de ADT-opdracht `-migrate` met de volgende syntaxis:

```
adt -migrate SIGNING_OPTIONS air_file_in air_file_out
```

- **SIGNING_OPTIONS** Met de ondertekeningsopties bepaalt u de persoonlijke sleutel en het certificaat waarmee het AIR-bestand wordt ondertekend. Met deze opties moet het **oorspronkelijke** handtekeningcertificaat worden geïdentificeerd. De opties worden beschreven in "ADT-opties voor codeondertekening" op pagina 188.
- **air_file_in** Het AIR-bestand voor de update, ondertekend met het **nieuwe** certificaat.
- **air_file_out** Het AIR-bestand dat wordt gemaakt.

***Opmerking:*** *de bestandsnamen die worden gebruikt voor de invoer- en uitvoerbestanden van AIR moeten verschillend zijn.*

Het volgende voorbeeld toont hoe ADT wordt aangeroepen met de markering `-migrate` om een migratiehandtekening toe te passen op een bijgewerkte versie van een AIR-toepassing:

```
adt -migrate -storetype pkcs12 -keystore cert.p12 myAppIn.air myApp.air
```

***Opmerking:*** *de opdracht* `-migrate` *is aan ADT toegevoegd in AIR 1.1.*

## Een AIR-toepassing als een native installatieprogramma migreren om een nieuw certificaat te gebruiken

Een AIR-toepassing die is gepubliceerd als een native installatieprogramma (bijvoorbeeld een toepassing die de native extensie API gebruikt), kan niet worden ondertekend met de ADT-opdracht `-migrate`, omdat deze een platformspecifieke native toepassing is en niet een AIR-bestand. In plaats daarvan kunt u als volgt een AIR-toepassing die is gepubliceerd als een native extensie, migreren naar een nieuw certificaat:

1. Maak een update voor uw toepassing.
2. Zorg ervoor dat het label `<supportedProfiles>` in uw descriptorbestand van de toepassing zowel het bureaubladprofiel als het extendedDesktop-profiel bevat (of verwijder het label `<supportedProfiles>` van de toepassingsdescriptor).
3. Plaats de updatetoepassing in een pakket en onderteken deze **als een AIR-bestand** met de ADT-opdracht `-package` met het **nieuwe** certificaat.
4. Pas het migratiecertificaat toe op het AIR-bestand met de ADT-opdracht `-migrate` met het **oorspronkelijke** certificaat (zoals eerder beschreven in "AIR-toepassingen migreren voor het gebruik van een nieuw certificaat" op pagina 212).

5 Plaats het AIR-bestand in een pakket in een native installatieprogramma met de ADT-opdracht `-package` met de markering `-target native`. Omdat de toepassing al ondertekend is, moet u geen handtekeningcertificaat opgeven als deel van deze stap.

Het volgende voorbeeld toont stappen 3-5 van dit proces. De code roept ADT aan met de opdracht `-package`, roept ADT aan met de opdracht `-migrate` en roept vervolgens ADT opnieuw aan met de opdracht `-package` om een bijgewerkte versie van een AIR-toepassing als een native installatieprogramma in een pakket te plaatsen:

```
adt -package -storetype pkcs12 -keystore new_cert.p12 myAppUpdated.air myApp.xml myApp.swf
adt -migrate -storetype pkcs12 -keystore original_cert.p12 myAppUpdated.air myAppMigrate.air
adt -package -target native myApp.exe myAppMigrate.air
```

## Een AIR-toepassing migreren die een native extensie gebruikt, migreren voor het gebruik van een nieuw certificaat

Een AIR-toepassing die een native extensie gebruikt, kan niet worden ondertekend met de ADT-opdracht `-migrate`. Deze kan ook niet worden gemigreerd met behulp van deze procedure voor het migreren van een AIR-toepassing als een native installatieprogramma omdat deze niet kan worden gepubliceerd als een tussentijds AIR-bestand. In plaats daarvan kunt u als volgt een AIR-toepassing die een native extensie gebruikt, migreren naar een nieuw certificaat:

1 Maak een update voor uw toepassing.

2 Plaats het native update-installatieprogramma in een pakket en onderteken het met de ADT-opdracht `-package`. Plaats de toepassing met het **nieuwe** certificaat in een pakket en neem de markering `-migrate` op die het **oorspronkelijke** certificaat opgeeft.

Gebruik de volgende syntaxis om de ADT-opdracht `-package` met de markering `-migrate` aan te roepen:

```
adt -package AIR_SIGNING_OPTIONS -migrate MIGRATION_SIGNING_OPTIONS -target package_type
NATIVE_SIGNING_OPTIONS output app_descriptor FILE_OPTIONS
```

- **AIR_SIGNING_OPTIONS** Met de ondertekeningsopties bepaalt u de persoonlijke sleutel en het certificaat waarmee het AIR-bestand wordt ondertekend. Met deze opties wordt het **nieuwe** handtekeningcertificaat geïdentificeerd. De opties worden beschreven in "ADT-opties voor codeondertekening" op pagina 188.
- **MIGRATION_SIGNING_OPTIONS** Met de ondertekeningsopties bepaalt u de persoonlijke sleutel en het certificaat waarmee het AIR-bestand wordt ondertekend. Met deze opties wordt het **oorspronkelijke** handtekeningcertificaat geïdentificeerd. De opties worden beschreven in "ADT-opties voor codeondertekening" op pagina 188.
- De overige opties zijn dezelfde opties die worden gebruikt om een AIR-toepassing als een native installatieprogramma in een pakket te plaatsen en worden beschreven in de "ADT-opdracht voor verpakken" op pagina 174.

Het volgende voorbeeld toont het aanroepen van ADT met de opdracht `-package` en de markering `-migrate` om een bijgewerkte versie van een AIR-toepassing die een native extensie gebruikt, in een pakket te plaatsen en een migratiehandtekening op de update toe te passen:

```
adt -package -storetype pkcs12 -keystore new_cert.p12 -migrate -storetype pkcs12 -keystore
original_cert.p12 -target native myApp.exe myApp.xml myApp.swf
```

***Opmerking:*** *de markering `-migrate` van de opdracht `-package` is beschikbaar in ADT in AIR 3.6 en later.*

# Zelfondertekende certificaten maken met ADT

Met niet-geautoriseerde certificaten kunt u een geldig AIR-installatiebestand maken. Niet-geautoriseerde certificaten bieden uw gebruikers echter beperkte beveiligingsgaranties. De authenticiteit van niet-geautoriseerde certificaten kan niet worden geverifieerd. Wanneer een zelfondertekend AIR-bestand wordt geïnstalleerd, wordt bij de gebruiker weergegeven dat de uitgever onbekend is. Een certificaat dat door ADT is gegenereerd, is vijf jaar geldig.

Als u een update maakt voor een AIR-toepassing die is ondertekend met een zelfondertekend certificaat, moet u het oorspronkelijke en het bijgewerkte AIR-bestand met hetzelfde certificaat ondertekenen. De certificaten die ADT produceert, zijn altijd uniek, zelfs als dezelfde parameters worden gebruikt. Als u updates zelf wilt ondertekenen met een certificaat dat door ADT is gegenereerd, moet u het oorspronkelijke certificaat daarom op een veilige locatie bewaren. Bovendien kunt u geen bijgewerkt AIR-bestand produceren nadat het oorspronkelijke, door ADT gegenereerde certificaat is verlopen. (U kunt nieuwe toepassingen met een ander certificaat publiceren, maar geen nieuwe versies van dezelfde toepassing.)

***Belangrijk:*** *vanwege de beperkingen van zelf-ondertekende certificaten, adviseert Adobe u ten zeerste om voor het ondertekenen van openbaar te maken AIR-toepassingen gebruik te maken van een commercieel certificaat dat is uitgegeven door een erkende certificeringsinstantie.*

Het certificaat en de bijbehorende persoonlijke sleutel die door ADT zijn gegenereerd, worden opgeslagen in een sleutelarchiefbestand van het type PKCS12. Het opgegeven wachtwoord wordt op de sleutel zelf ingesteld, niet op het sleutelarchief.

**Voorbeelden van het genereren van certificaten**

```
adt -certificate -cn SelfSign -ou QE -o "Example, Co" -c US 2048-RSA newcert.p12 39#wnetx3tl
adt -certificate -cn ADigitalID 1024-RSA SigningCert.p12 39#wnetx3tl
```

Als u deze certificaten wilt gebruiken om AIR-bestanden te ondertekenen, gebruikt u de volgende ondertekeningsopties met de ADT-opdracht `-package` of `-prepare`:

```
-storetype pkcs12 -keystore newcert.p12 -storepass 39#wnetx3tl
-storetype pkcs12 -keystore SigningCert.p12 -storepass 39#wnetx3tl
```

***Opmerking:*** *bij Java versie 1.5 en later worden hoge ASCII-tekens in wachtwoorden voor de beveiliging van PKCS12-certificatiebestanden niet geaccepteerd. Gebruik alleen normale ASCII-tekens in het wachtwoord.*

# Hoofdstuk 14: AIR-toepassingsdescriptorbestanden

Elke AIR-toepassing vereist een toepassingsdescriptorbestand. Het toepassingsdescriptorbestand is een XML-document waarin de basiseigenschappen van de toepassing worden gedefinieerd.

Veel ontwikkelomgevingen die ondersteuning bieden voor AIR genereren automatisch een toepassingsdescriptor wanneer u een project maakt. Als dat niet het geval is, moet u zelf een descriptorbestand maken. Een voorbeeld van een descriptorbestand met de naam `descriptor-sample.xml` staat in de map `samples` van de SDK's van AIR en Flex.

U kunt een willekeurige naam voor het descriptorbestand van de toepassing gebruiken. Wanneer u de toepassing in een pakket opneemt, wordt de naam van het descriptorbestand van de toepassing gewijzigd in `application.xml` en wordt het bestand in een speciale map in het AIR-pakket geplaatst.

**Voorbeeld toepassingsdescriptor**

In het volgende toepassingsdescriptordocument worden de basiseigenschappen ingesteld die worden gebruikt door de meeste AIR-toepassingen:

```
<?xml version="1.0" encoding="utf-8" ?>
<application xmlns="http://ns.adobe.com/air/application/3.0">
    <id>example.HelloWorld</id>
    <versionNumber>1.0.1</versionNumber>
    <filename>Hello World</filename>
    <name>Example Co. AIR Hello World</name>
     <description>
        <text xml:lang="en">This is an example.</text>
        <text xml:lang="fr">C'est un exemple.</text>
        <text xml:lang="es">Esto es un ejemplo.</text>
    </description>
    <copyright>Copyright (c) 2010 Example Co.</copyright>
    <initialWindow>
        <title>Hello World</title>
        <content>
            HelloWorld.swf
        </content>
    </initialWindow>
    <icon>
        <image16x16>icons/smallIcon.png</image16x16>
        <image32x32>icons/mediumIcon.png</image32x32>
        <image48x48>icons/bigIcon.png</image48x48>
        <image128x128>icons/biggerIcon.png</image128x128>
    </icon>
</application>
```

Als de toepassing een HTML-bestand als hoofdinhoud gebruikt in plaats van een SWF-bestand, is alleen het element `<content>` anders:

```
<content>
    HelloWorld.html
</content>
```

# Wijzigingen toepassingsdescriptor

De AIR-toepassingsdescriptor is gewijzigd in de volgende AIR-versies.

## Wijzigingen descriptor AIR 1.1

Toepassingselementen `name` en `description` mochten worden gelokaliseerd met behulp van het element `tekst`.

## Wijzigingen descriptor AIR 1.5

`contentType` werd vereist onderliggend element of `fileType`.

## Wijzigingen descriptor AIR 1.5.3

Het element `publisherID` `toegevoegd opdat toepassingen een uitgever-id-waarde kunnen opgeven.`

## Wijzigingen descriptor AIR 2.0

Toegevoegd:

- `aspectRatio`
- `autoOrients`
- `fullScreen`
- `image29x29` (zie imageNxN)
- `image57x57`
- `image72x72`
- `image512x512`
- `iPhone`
- `renderMode`
- `supportedProfiles`

## Wijzigingen descriptor AIR 2.5

Verwijderd: `version`

Toegevoegd:

- `android`
- `extensionID`
- `extensions`
- `image36x36` (zie imageNxN)
- `manifestAdditions`
- `versionLabel`
- `versionNumber`

## Wijzigingen descriptor AIR 2.6

Toegevoegd:

- `image114x114` (zie `imageNxN`)
- `requestedDisplayResolution`
- `softKeyboardBehavior`

## Wijzigingen descriptor AIR 3.0

Toegevoegd:

- `colorDepth`
- *direct* als een geldige waarde voor `renderMode`
- renderMode wordt niet meer genegeerd bij bureaubladplatformen
- Het Android-element `<uses-sdk>` kan worden opgegeven. (Dit was voorheen niet toegestaan.)

## Wijzigingen in AIR 3.1-descriptorbestand

Toegevoegd:

- "Entitlements" op pagina 230

## Wijzigingen descriptor AIR 3.2

Toegevoegd:

- `depthAndStencil`
- `supportedLanguages`

## Wijzigingen in AIR 3.3-descriptorbestand

Toegevoegd:

- `aspectRatio` bevat nu de optie `ANY`.

## Wijzigingen in AIR 3.4-descriptorbestand

Toegevoegd:

- `image50 x 50` (zie "`imageNxN`" op pagina 238)
- `image58x58` (zie "`imageNxN`" op pagina 238)
- `image100 x 100` (zie "`imageNxN`" op pagina 238)
- `image1024x1024` (zie "`imageNxN`" op pagina 238)

## Wijzigingen in AIR 3.6-descriptorbestand

Toegevoegd:

- "requestedDisplayResolution" op pagina 248 in element "iPhone" op pagina 242 bevat nu een `excludeDevices`-attribuut waardoor u kunt opgeven welke iOS-doelapparaten hoge of standaardresolutie gebruiken

- nieuw element “requestedDisplayResolution” op pagina 248 in “initialWindow” op pagina 240 geeft op of de hoge of standaardresolutie wordt gebruikt op bureaubladplatforms zoals Mac's met hoge-resolutieschermen

### Wijzigingen in AIR 3.7-descriptorbestand

Toegevoegd:

- Het element “iPhone” op pagina 242 verschaft nu een element “externalSwfs” op pagina 232 waarmee u een lijst met SWF-bestanden kunt opgeven die tijdens de runtime moeten worden geladen.
- Het element “iPhone” op pagina 242 verschaft nu een element “forceCPURenderModeForDevices” op pagina 235 waarmee u de CPU-rendermodus kunt forceren voor een opgegeven aantal apparaten.

## De structuur van het descriptorbestand van de toepassing

Het toepassingsdescriptorbestand is een XML-document met de volgende structuur:

```
<application xmlns="http://ns.adobe.com/air/application/3.0">
    <allowBrowserInvocation>...<allowBrowserInvocation>
    <android>
        <colorDepth>...</colorDepth>
        <manifestAdditions
                <manifest>...</manifest>
            ]]>
        </manifestAdditions
    </android>
    <copyright>...</copyright>
    customUpdateUI>...</
    <beschrijving>
        <tekst xml:lang="...">...</tekst>
    </beschrijving>
    <extensions>
        <extensionID>...</extensionID>
    </extensions>
    <filename>...</filename>
    <fileTypes>
        <fileType>
            <contentType>...</contentType>
            <beschrijving>...</beschrijving>
            <extension>...</extension>
            <icon>
                <imageNxN>...</imageNxN>
            </icon>
            <name>...</name>
        </fileType>
    </fileTypes>
    <icon>
        <imageNxN>...</imageNxN>
    </icon>
    <id>...</id>
    <initialWindow>
        <aspectRatio>...</aspectRatio>
        <autoOrients>...</autoOrients>
```

```
        <content>...</content>
        <depthAndStencil>...</depthAndStencil>
        <fullScreen>...</fullScreen>
        <height>...</height>
        <maximizable>...</maximizable>
        <maxSize>...</maxSize>
        <minimizable>...</minimizable>
        <minSize>...</minSize>
        <renderMode>...</renderMode>
        <requestedDisplayResolution>...</requestedDisplayResolution>
        <resizable>...</resizable>
        <softKeyboardBehavior>...</softKeyboardBehavior>
        <systemChrome>...</systemChrome>
        <title>...</title>
        <transparent>...</transparent>
        <visible>...</visible>
        <width>...</width>
        <x>...</x>
        <y>...</y>
    </initialWindow>
    <installFolder>...</installFolder>
    <iPhone>
        <Entitlements>...</Entitlements>
        <InfoAdditions>...</InfoAdditions>
        <requestedDisplayResolution>...</requestedDisplayResolution>
        <forceCPURenderModeForDevices>...</forceCPURenderModeForDevices>
        <externalSwfs>...</externalSwfs>
    </iPhone>
    <name>
        <tekst xml:lang="...">...</tekst>
    </name>
    <programMenuFolder>...</programMenuFolder>
    <publisherID>...</publisherID>
    <"supportedLanguages" op pagina 250>...</"supportedLanguages" op pagina 250>
    <supportedProfiles>...</supportedProfiles>
    <versionNumber>...</versionNumber>
    <versionLabel>...</versionLabel>
</application>
```

# Descriptorelementen AIR-toepassing

In de volgende woordenlijst met elementen wordt elk juridisch element van een AIR-toepassingsdescriptorbestand beschreven.

## allowBrowserInvocation

**Adobe AIR 1.0 of hoger — Optioneel**

Hiermee kan de interne browser-API van AIR de toepassing herkennen en starten.

Als u deze waarde instelt op `true`, moet u rekening houden met gevolgen voor de beveiliging. Deze worden beschreven in AIR-toepassingen aanroepen vanuit de browser (voor ActionScript-ontwikkelaars) en Invoking an AIR application from the browser (voor HTML-ontwikkelaars).

Zie "Geïnstalleerde AIR-toepassingen starten vanuit de browser" op pagina 268 voor meer informatie.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

`true` of `false` (standaard)

**Voorbeeld**

```
<allowBrowserInvocation>true</allowBrowserInvocation>
```

## android

**Adobe AIR 2.5 of hoger — Optioneel**

Hiermee kunt u elementen toevoegen aan het Android-manifestbestand AIR maakt het bestand AndroidManifest.xml file voor elk APK-pakket. U kunt het android-element in de AIR-toepassingsdescriptor gebruiken om er aanvullende items aan toe te voegen. Genegeerd op alle platforms behalve Android.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:**

- "colorDepth" op pagina 225
- "manifestAdditions" op pagina 243

**Inhoud**

Elementen die voor Android specifieke eigenschappen definiëren om toe te voegen aan het Android-toepassingsmanifest.

**Voorbeeld**

```
<android>
    <manifestAdditions>
        ...
    </manifestAdditions>
</android>
```

### Meer Help-onderwerpen

"Android-instellingen" op pagina 80

Het bestand AndroidManifest.xml

## application

**Adobe AIR 1.0 of hoger — Vereist**

Het hoofdelement van een AIR-toepassingsdescriptordocument.

**Bovenliggend element:** geen

**Onderliggende elementen:**

- "allowBrowserInvocation" op pagina 219

- “android” op pagina 220
- “copyright” op pagina 227
- “customUpdateUI” op pagina 227
- “beschrijving” op pagina 228
- “extensions” op pagina 231
- “filename” op pagina 232
- “fileTypes” op pagina 234
- “icon” op pagina 237
- “id” op pagina 237
- “initialWindow” op pagina 240
- “installFolder” op pagina 241
- “iPhone” op pagina 242
- “name” op pagina 245
- “programMenuFolder” op pagina 247
- “publisherID” op pagina 247
- “supportedLanguages” op pagina 250
- “supportedProfiles” op pagina 251
- “version” op pagina 254
- “versionLabel” op pagina 254
- “versionNumber” op pagina 254

**Attributen**

minimumPatchLevel — Het minimale patchniveau AIR runtime dat wordt vereist door de ze toepassing.

xmlns — het XML-naamruimteattribuut bepaalt de vereiste AIR-runtimeversie van de toepassing.

De naamruimte verandert bij elke grote release van AIR (maar niet bij kleine patches). Het laatste segment van de naamruimte, bijvoorbeeld '3.0', geeft aan welke versie van de runtime voor de toepassing is vereist.

De xmlns-waarden voor de belangrijkste AIR-versies zijn:

```
xmlns="http://ns.adobe.com/air/application/1.0"
xmlns="http://ns.adobe.com/air/application/1.1"
xmlns="http://ns.adobe.com/air/application/1.5"
xmlns="http://ns.adobe.com/air/application/1.5.2"
xmlns="http://ns.adobe.com/air/application/1.5.3"
xmlns="http://ns.adobe.com/air/application/2.0"
xmlns="http://ns.adobe.com/air/application/2.5"
xmlns="http://ns.adobe.com/air/application/2.6"
xmlns="http://ns.adobe.com/air/application/2.7"
xmlns="http://ns.adobe.com/air/application/3.0"
xmlns="http://ns.adobe.com/air/application/3.1"
xmlns="http://ns.adobe.com/air/application/3.2"
xmlns="http://ns.adobe.com/air/application/3,3"
xmlns="http://ns.adobe.com/air/application/3.4"
xmlns="http://ns.adobe.com/air/application/3.5"
xmlns="http://ns.adobe.com/air/application/3.6"
xmlns="http://ns.adobe.com/air/application/3.7"
```

Voor op SWF gebaseerde toepassingen, bepaalt de in de toepassingsdescriptor opgegeven AIR-runtimeversie de maximale SWF-versie die kan worden geladen als de eerste inhoud van de toepassing. Toepassingen waarin AIR 1.0 of AIR 1.1 wordt opgegeven, kunnen alleen SWF9-bestanden (Flash Player 9) gebruiken als eerste inhoud, zelfs als de AIR 2-runtime wordt gebruikt. Toepassingen waarin AIR 1.5 (of hoger) wordt opgegeven kunnen SWF9- of SWF10-bestanden (Flash Player 10) als eerste inhoud gebruiken.

De SWF-versie bepaalt welke versie van de AIR- en Flash Player-API's beschikbaar zijn. Als een SWF9-bestand wordt gebruikt als de eerste inhoud van een AIR 1.5-toepassing, heeft die toepassing alleen toegang tot de AIR 1.1- en Flash Player 9-API's. Bovendien hebben gedragswijzigingen die zijn aangebracht in bestaande API's in AIR 2.0 of Flash Player 10.1 geen effect. (Belangrijke wijzigingen in API's met betrekking tot veiligheid zijn een uitzondering op dit principe en kunnen retro-actief worden toegepast op huidige en toekomstige patches van de runtime.)

Bij toepassingen op HTML-basis bepaalt de in de toepassingsdescriptor opgegeven runtimeversie welke versie van de AIR- en Flash Player-API's beschikbaar zijn voor de toepassing. Het HTML-, CSS- en JavaScript-gedrag wordt altijd bepaald door de versie van Webkit die wordt gebruikt in de geïnstalleerde AIR-runtime, en niet door de toepassingsdescriptor.

Als een AIR-toepassing SWF-inhoud laadt, is de versie van de AIR- en Flash Player-API's die beschikbaar zijn voor die inhoud afhankelijk van hoe de inhoud wordt geladen. Soms wordt de effectieve versie bepaald door de naamruimte van de toepassingsdescriptor, soms door de versie van de inhoud die wordt geladen, en soms door de versie van de inhoud die is geladen. In de volgende tabel wordt aangegeven hoe de API-versie wordt bepaald op basis van de laadmethode:

| **Hoe de inhoud wordt geladen** | **Hoe de API-versie wordt bepaald** |
|---|---|
| Eerste inhoud, op SWF gebaseerde toepassing | SWF-versie van het **geladen** bestand |
| Eerste inhoud, op HTML gebaseerde toepassing | Naamruimte van de toepassingsdescriptor |
| SWF geladen door SWF-inhoud | Versie van de **ladende** inhoud |
| SWF-bibliotheek geladen door HTML-inhoud met <script>-tag | Naamruimte van de toepassingsdescriptor |
| SWF geladen door HTML-inhoud door AIR- of Flash Player-API's (zoals flash.display.Loader) | Naamruimte van de toepassingsdescriptor |
| SWF geladen door HTML-inhoud met <object>- of <embed>-tags (of de equivalente JavaScript-API's) | SWF-versie van het **geladen** bestand |

Bij het laden van een SWF-bestand van een andere versie dan de ladende inhoud, kunt u op twee problemen stuiten:

- Als een SWF-bestand van een nieuwere versie wordt geladen door een SWF-bestand van een oudere versie, worden de referenties naar API's die in de nieuwere versies van AIR en Flash Player in de geladen inhoud zijn toegevoegd, niet opgelost.
- Als een SWF-bestand van een oudere versie wordt geladen door een SWF-bestand van een nieuwere SWF-versie, gedragen de API's die zijn gewijzigd in de nieuwere versies van AIR en Flash Player zich op een manier die niet wordt verwacht door de geladen inhoud.

### Inhoud

Het toepassingselement bevat onderliggende elementen die de eigenschappen van een AIR-toepassing definiëren.

### Voorbeeld

```
<?xml version="1.0" encoding="utf-8" ?>
<application xmlns="http://ns.adobe.com/air/application/3.0">
    <id>HelloWorld</id>
    <version>2.0</version>
    <filename>Hello World</filename>
    <name>Example Co. AIR Hello World</name>
     <description>
        <text xml:lang="en">This is an example.</text>
        <text xml:lang="fr">C'est un exemple.</text>
        <text xml:lang="es">Esto es un ejemplo.</text>
    </description>
    <copyright>Copyright (c) 2010 Example Co.</copyright>
    <initialWindow>
        <title>Hello World</title>
        <content>
            HelloWorld.swf
        </content>
        <systemChrome>none</systemChrome>
        <transparent>true</transparent>
        <visible>true</visible>
        <minSize>320 240</minSize>
    </initialWindow>
    <installFolder>Example Co/Hello World</installFolder>
    <programMenuFolder>Example Co</programMenuFolder>
    <icon>
        <image16x16>icons/smallIcon.png</image16x16>
        <image32x32>icons/mediumIcon.png</image32x32>
```

```
        <image48x48>icons/bigIcon.png</image48x48>
        <image128x128>icons/biggestIcon.png</image128x128>
    </icon>
    <customUpdateUI>true</customUpdateUI>
    <allowBrowserInvocation>false</allowBrowserInvocation>
    <fileTypes>
        <fileType>
            <name>adobe.VideoFile</name>
            <extension>avf</extension>
            <description>Adobe Video File</description>
            <contentType>application/vnd.adobe.video-file</contentType>
            <icon>
                <image16x16>icons/avfIcon_16.png</image16x16>
                <image32x32>icons/avfIcon_32.png</image32x32>
                <image48x48>icons/avfIcon_48.png</image48x48>
                <image128x128>icons/avfIcon_128.png</image128x128>
            </icon>
        </fileType>
    </fileTypes>
</application>
```

## aspectRatio

**Adobe AIR 2.0 of hoger, iOS en Android — Optioneel**

Hiermee wordt de hoogte-breedteverhouding van de toepassing opgegeven.

Als deze niet wordt opgegeven, wordt de toepassing geopend met de standaardverhouding en -oriëntatie van het apparaat. De standaardoriëntatie varieert per apparaat. Op apparaten met kleine schermen, zoals telefoons, is het vaak de verhouding Staand. Op sommige apparaten, zoals de iPad-tablet, wordt de toepassing in de huidige oriëntatie geopend. Bij AIR 3.3 en hoger heeft dit betrekking op de gehele toepassing en niet alleen op de oorspronkelijke weergave.

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

### Inhoud

`portrait, landscape` of `any` (respectievelijk een staande, liggende of willekeurige oriëntatie)

### Voorbeeld

```
<aspectRatio>landscape</aspectRatio>
```

## autoOrients

**Adobe AIR 2.0 of hoger, iOS en Android — Optioneel**

Geeft op of de oriëntatie van inhoud in de toepassing automatisch wordt gewijzigd als het apparaat overgaat op een andere fysieke oriëntatie. Zie Oriëntatie werkgebied voor meer informatie.

Wanneer de automatische oriëntatie wordt gebruikt, kunt u de eigenschappen `align` en `scaleMode` van het werkgebied als volgt instellen:

```
stage.align = StageAlign.TOP_LEFT;
stage.scaleMode = StageScaleMode.NO_SCALE;
```

Met deze instellingen kan de toepassing rond de linkerbovenhoek roteren en wordt voorkomen dat de toepassingsinhoud automatisch wordt geschaald. De andere schaalmodi passen uw inhoud aan de afmetingen van het geroteerde werkgebied aan, maar knippen uw inhoud daarbij af, vervormen deze of krimpen deze sterk in. U krijgt bijna altijd een beter resultaat wanneer u de inhoud zelf opnieuw tekent of de lay-out ervan wijzigt.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

`true` of `false` (standaard)

**Voorbeeld**

```
<autoOrients>true</autoOrients>
```

## colorDepth

**Adobe AIR 3 of hoger — Optioneel**

Bepaalt of 16-bits of 32-bits kleur moet worden gebruikt.

Het gebruik van 16-bits kleur kan de renderprestaties verbeteren, maar dat gaat ten koste van de kleurgetrouwheid. Vóór AIR 3 wordt altijd 16-bits kleur gebruikt op Android. In AIR 3 wordt standaard 32-bits kleur gebruikt.

***Opmerking:*** *als uw toepassing de StageVideo-klasse gebruikt, dient u 32-bits kleur te gebruiken.*

**Bovenliggend element:**"android" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

Een van de volgende waarden:

- 16-bits
- 32-bits

**Voorbeeld**

```
<android>
    <colorDepth>16bit</colorDepth>
    <manifestAdditions>...</manifestAdditions>
</android>
```

## containsVideo

Geeft aan of de toepassing al dan niet video-inhoud bevat.

**Bovenliggend element:**"android" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

Een van de volgende waarden:

- true

- false

**Voorbeeld**

```
<android>
    <containsVideo>true</containsVideo>
    <manifestAdditions>...</manifestAdditions>
</android>
```

## content

**Adobe AIR 1.0 of hoger — Vereist**

De waarde die wordt opgegeven voor het element `content`, is de URL voor het hoofdinhoudsbestand van de toepassing. Dit kan een SWF- of een HTML-bestand zijn. De URL wordt opgegeven op basis van de hoofdmap van de installatiemap van de toepassing. (Als een AIR-toepassing met ADL wordt gebruikt, is de URL gebaseerd op de map waarin zich het descriptorbestand van de toepassing bevindt. U kunt de parameter `root-dir` van ADL gebruiken om een andere hoofdmap op te geven.)

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

Een URL op basis van de toepassingsmap. Aangezien de waarde van het element content wordt beschouwd als een URL, moeten de tekens in de naam van het inhoudsbestand URL-gecodeerd zijn overeenkomstig de regels in RFC 1738. Spaties moeten bijvoorbeeld als `%20` worden gecodeerd.

**Voorbeeld**

```
<content>TravelPlanner.swf</content>
```

## contentType

**Adobe AIR 1.0 tot 1.1 — Optioneel; AIR 1.5 of hoger — Vereist**

`contentType` is vereist per AIR 1.5 (dit was optioneel in AIR 1.0 en 1.1). Met de eigenschap kunnen bepaalde besturingssystemen de beste toepassing lokaliseren om een bestand te openen. De waarde moet het MIME-type van de bestandsinhoud zijn. De waarde wordt genegeerd in Linux als het bestandstype al is geregistreerd en een toegewezen MIME-type heeft.

**Bovenliggend element:**"fileType" op pagina 233

**Onderliggende elementen:** geen

**Inhoud**

Het MIME-type en subtype. Zie RFC2045 voor meer informatie over MIME-typen.

**Voorbeeld**

```
<contentType>text/plain</contentType>
```

## copyright

**Adobe AIR 1.0 of hoger — Optioneel**

De copyrightinformatie voor de AIR-toepassing. In Mac OS verschijnt de copyrighttekst in het venster Over voor de geïnstalleerde toepassing. In Mac OS wordt de copyrightinformatie ook gebruikt in het veld NSHumanReadableCopyright in het bestand Info.plist voor de toepassing.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

### Inhoud

Een tekenreeks die de copyrightinformatie voor de toepassings bevat.

### Voorbeeld

```
<copyright>© 2010, Examples, Inc.All rights reserved.</copyright>
```

## customUpdateUI

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft aan of een toepassing eigen updatedialoogvensters levert. Indien `false` presenteert AIR standaardupdatedialoogvensters voor de gebruiker. Alleen toepassingen die zijn gedistribueerd als AIR-bestanden, kunnen het ingebouwde AIR-updatesysteem gebruiken.

Wanneer het element `customUpdateUI` in de geïnstalleerde versie van uw toepassing is ingesteld op `true` en de gebruiker dubbelklikt op het AIR-bestand voor een nieuwe versie of hij installeert een update van de toepassing met de functie voor naadloze installatie, opent het runtimeprogramma de geïnstalleerde versie van de toepassing. Het standaardinstallatieprogramma van de AIR-toepassing wordt niet geopend. De logica van uw toepassing kan vervolgens bepalen hoe de update verder wordt afgehandeld. (De toepassings- en uitgever-id in het AIR-bestand moeten overeenkomen met de waarden in de geïnstalleerde toepassing, anders kan de upgrade niet verdergaan.)

***Opmerking:*** *het mechanisme* `customUpdateUI` *wordt alleen gebruikt als de toepassing al is geïnstalleerd en de gebruiker op het AIR-installatiebestand dubbelklikt dat een update bevat of een update van de toepassing installeert met behulp van de functie voor naadloze installatie. U kunt een update downloaden en starten via de logica van uw eigen toepassing en desgewenst uw eigen gebruikersinterface weergeven, ongeacht of* `customUpdateUI` *op* `true` *is ingesteld.*

Zie "AIR-toepassingen bijwerken" op pagina 270 voor meer informatie.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

### Inhoud

`true` of `false` (standaard)

### Voorbeeld

```
<customUpdateUI>true</customUpdateUI>
```

## depthAndStencil

**Adobe AIR 3.2 of hoger — Optioneel**

Hiermee wordt aangegeven dat de toepassing gebruik van de diepte- of stencilbuffer vereist. Deze buffers worden meestal gebruikt tijdens het werken met 3D-inhoud. De standaardwaarde van dit element is `false`, zodat de diepte- en stencilbuffers worden uitgeschakeld. Dit element is vereist omdat de buffers moeten worden toegewezen bij het starten van de toepassing, nog voordat inhoud wordt geladen.

De instelling van dit element moet overeenkomen met de waarde die voor het argument `enableDepthAndStencil` is doorgegeven aan de methode `Context3D.configureBackBuffer()`. AIR geeft een foutmelding weer als deze waarden niet overeenkomen.

Dit element is alleen van toepassing in geval van de `renderModedirect`. Als `renderMode` niet gelijk is aan `direct`, genereert ADT fout 118:

```
<depthAndStencil> element unexpected for render mode cpu.  It requires "direct" render mode.
```

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

### Inhoud

`true` of `false` (standaard)

### Voorbeeld

```
<depthAndStencil>true</depthAndStencil>
```

## beschrijving

**Adobe AIR 1.0 of hoger — Optioneel**

De beschrijving van de toepassing, weergegeven in het AIR-toepassingsinstallatieprogramma.

Als u één enkel tekstknooppunt opgeeft (in de plaats van meerdere text-elementen), gebruikt het installatieprogramma van de AIR-toepassing deze beschrijving, ongeacht de systeemtaal. Anders gebruikt het AIR-toepassingsinstallatieprogramma de beschrijving die het meest overeenkomt met de taal van de gebruikersinterface van het besturingssysteem van de gebruiker. Voorbeeld: een installatie waarbij het element `description` van het descriptorbestand van de toepassing een waarde bevat voor de landinstelling 'en' (Engels). Het AIR-toepassingsinstallatieprogramma gebruikt de en-beschrijving als het systeem van de gebruiker en (Engels) als de taal van de gebruikersinterface herkent. De en-beschrijving wordt ook gebruikt als de taal van de gebruikersinterface van het systeem en-US (Engels VS). Als de gebruikersinterfacetaal van het systeem echter 'en-US' is en het descriptorbestand van de toepassing zowel de naam 'en-US' als de naam 'en-GB' definieert, gebruikt het installatieprogramma van de AIR-toepassing de waarde 'en-US'. Als de toepassing geen beschrijving definieert die overeenkomt met de taal van de gebruikersinterface van het systeem, gebruikt het AIR-toepassingsinstallatieprogramma de eerste `description`-waarde die is gedefinieerd in het toepassingsdescriptorbestand.

Zie “AIR-toepassingen lokaliseren” op pagina 306 voor meer informatie over het ontwikkelen van meertalige toepassingen.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:**“tekst” op pagina 252

**Inhoud**

In het descriptorschema van de AIR 1.0-toepassing kan slechts één enkelvoudig tekstknooppunt voor de naam worden gedefinieerd (niet meerdere `tekst`-elementen).

In AIR 1.1 (of hoger) kunt u meerdere talen opgeven in het element `description`. Het attribuut `xml:lang` voor elk tekstelement geeft een taalcode op die is gedefinieerd is in RFC4646 (http://www.ietf.org/rfc/rfc4646.txt).

**Voorbeeld**

Beschrijving met eenvoudige tekstnode:

```
<description>This is a sample AIR application.</description>
```

Beschrijving met gelokaliseerde tekstelementen voor Engels, Frans en Spaans (geldig in AIR 1.1 en hoger):

```
<description>
    <text xml:lang="en">This is an example.</text>
    <text xml:lang="fr">C'est un exemple.</text>
    <text xml:lang="es">Esto es un ejemplo.</text>
</description>
```

## beschrijving

**Adobe AIR 1.0 of hoger — Vereist**

De bestandstypebeschrijving wordt aan de gebruiker weergegeven door het besturingssysteem. De bestandstypebeschrijving is niet lokaliseerbaar.

Zie ook: “beschrijving” op pagina 228 als onderliggend element van het toepassingselement

**Bovenliggend element:**“fileType” op pagina 233

**Onderliggende elementen:** geen

**Inhoud**

Een tekenreeks waarin de bestandsinhoud wordt beschreven.

**Voorbeeld**

```
<description>PNG image</description>
```

## embedFonts

Hiermee kunt u eigen lettertypen op StageText gebruiken in de AIR-toepassing. Dit element is optioneel.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:**“font” op pagina 234

**Inhoud**

Het element embedFonts kan een willekeurig aantal font-elementen bevatten.

**Voorbeeld**

```
<embedFonts>
   <font>
      <fontPath>ttf/space age.ttf</fontPath>
      <fontName>space age</fontName>
   </font>
   <font>
      <fontPath>ttf/xminus.ttf</fontPath>
      <fontName>xminus</fontName>
   </font>
</embedFonts>
```

## Entitlements

**Adobe AIR 3.1 of hoger — Alleen iOS, optioneel**

iOS gebruikt eigenschappen die entitlements worden genoemd. Hiermee krijgt de toepassing toegang tot extra bronnen en functionaliteit. Met het element Entitlements kunt u deze informatie opgeven in een iOS-toepassing voor mobiele apparaten.

**Bovenliggend element:**"iPhone" op pagina 242

**Onderliggende elementen:** iOS Entitlements.plist-elementen

**Inhoud**

Bevat onderliggende elementen waarin sleutelwaardeparen worden opgegeven die moeten worden gebruikt als Entitlements.plist-instellingen voor de toepassing. Inhoud van het element Entitlements moet zijn opgenomen in een CDATA-blok. Voor meer informatie gaat u naar het naslagwerk Entitlement Key Reference in de iOS-ontwikkelaarsbibliotheek.

**Voorbeeld**

```
<iphone>
...
   <Entitlements>
      <![CDATA[
         <key>aps-environment</key>
         <string>development</string>
      ]]>
   </Entitlements>
</iphone>
```

## extension

**Adobe AIR 1.0 of hoger — Vereist**

De uitbreidingstekenreeks van een bestandstype.

**Bovenliggend element:**"fileType" op pagina 233

**Onderliggende elementen:** geen

**Inhoud**

Een tekenreeks die de tekens van de bestandsuitbreiding definieert (zonder de punt, ".").

**Voorbeeld**

```
<extension>png</extension>
```

## extensionID

**Adobe AIR 2.5 of hoger**

Geeft de id op van een ActionScript-extensie die wordt gebruikt door de toepassing. De ID wordt gedefinieerd in het uitbreidingsdescriptordocument.

**Bovenliggend element:**"extensions" op pagina 231

**Onderliggende elementen:** geen

**Inhoud**

Een tekenreeks waarin de Actionscript-extensie-id wordt geïdentificeerd.

**Voorbeeld**

```
<extensionID>com.example.extendedFeature</extensionID>
```

## extensions

**Adobe AIR 2.5 of hoger — Optioneel**

Identificeert de ActionScript-extensies die worden gebruikt door een toepassing.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:**"extensionID" op pagina 231

**Inhoud**

Onderliggende `extensionID`-elementen die de ActionScript-extensie-id's van het descriptorbestand van de extensie bevatten.

**Voorbeeld**

```
<extensions>
    <extensionID>extension.first</extensionID>
    <extensionID>extension.next</extensionID>
    <extensionID>extension.last</extensionID>
</extensions>
```

## externalSwfs

**Adobe AIR 3.7 of hoger, alleen iOS — Optioneel**

Hier geeft u de naam op van een tekstbestand dat een lijst bevat met SWF-bestanden die door ADT moeten worden geconfigureerd voor externe hosting. U kunt de downloadgrootte van de aanvankelijke toepassing minimaliseren door een subset van de door uw toepassing gebruikte SWF-bestanden te verpakken en de resterende externe SWF-bestanden (met alleen elementen) tijdens de runtime te laden met behulp van de methode `Loader.load ()`. Als u deze functie wilt gebruiken, dient u de toepassing zodanig te verpakken dat ADT alle ActionScript ByteCode (ABC) uit de extern geladen SWF-bestanden naar de SWF-hoofdtoepassing verplaatst, zodat er een SWF-bestand met uitsluitend elementen overblijft. Zo wordt voldaan aan de regel van de Apple Store die het downloaden van code verbiedt nadat een toepassing is geïnstalleerd.

Zie "Downloadgrootte minimaliseren door externe SWF-bestanden met uitsluitend elementen te laden" op pagina 91 voor meer informatie.

**Bovenliggend element:**"iPhone" op pagina 242, "initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Naam van een tekstbestand dat een lijst bevat met via een regel afgebakende SWF-bestanden die extern worden gehost.

**Attributen:**
Geen.

**Voorbeelden**
iOS:

```
<iPhone>
    <externalSwfs>FileContainingListofSWFs.txt</externalSwfs>
</iPhone>
```

## filename

**Adobe AIR 1.0 of hoger — Vereist**

De tekenreeks die als bestandsnaam van de toepassing wordt gebruikt (zonder extensie) wanneer de toepassing wordt geïnstalleerd. Het toepassingsbestand start de AIR-toepassing in de runtime. Als de waarde `name` niet is opgegeven, wordt `filename` ook als de naam van de installatiemap gebruikt.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**
De eigenschap `filename` mag een willekeurig Unicode-teken (UTF-8) bevatten, behalve de volgende tekens, omdat deze tekens in verschillende bestandssystemen niet in een bestandsnaam mogen worden gebruikt:

| Teken | Hexadecimale code |
|---|---|
| *diverse* | 0x00 - x1F |
| * | x2A |
| " | x22 |
| : | x3A |
| > | x3C |
| < | x3E |
| ? | x3F |
| \ | x5C |
| \| | x7C |

De waarde `filename` mag niet met een punt eindigen.

### Voorbeeld

```
<filename>MyApplication</filename>
```

## fileType

**Adobe AIR 1.0 of hoger — Optioneel**

Beschrijft een enkel bestandstype waar de toepassing voor kan worden geregistreerd.

**Bovenliggend element:**“fileTypes” op pagina 234

**Onderliggende elementen:**

- “contentType” op pagina 226
- “beschrijving” op pagina 229
- “extension” op pagina 230
- “icon” op pagina 237
- “name” op pagina 246

### Inhoud

Elementen waarin een bestandstype wordt beschreven.

### Voorbeeld

```
<fileType>
    <name>foo.example</name>
    <extension>foo</extension>
    <description>Example file type</description>
    <contentType>text/plain</contentType>
    <icon>
        <image16x16>icons/fooIcon16.png</image16x16>
        <image48x48>icons/fooIcon48.png</imge48x48>
    <icon>
</fileType>
```

## fileTypes

**Adobe AIR 1.0 of hoger — Optioneel**

Met het element `fileTypes` kunt u de bestandstypen definiëren waaraan een AIR-toepassing kan worden gekoppeld.

Als een AIR-toepassing is geïnstalleerd, worden alle gedefinieerde bestandstypen in het besturingssysteem geregistreerd. Als deze bestandstypen nog niet aan een andere toepassing zijn gekoppeld, worden ze aan de AIR-toepassing gekoppeld. Als u een bestaande koppeling tussen een bestandstype en een andere toepassing wilt overschrijven, gebruikt u de methode `NativeApplication.setAsDefaultApplication()` tijdens de runtime (bij voorkeur met toestemming van de gebruiker).

***Opmerking:*** *de runtimemethoden kunnen alleen koppelingen beheren voor de bestandstypen die zijn opgegeven in het descriptorbestand van de toepassing.*

Het element `fileTypes` is optioneel.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:**“fileType” op pagina 233

### Inhoud

Het element `fileTypes` kan een willekeurig aantal `fileType`-elementen bevatten.

### Voorbeeld

```
<fileTypes>
    <fileType>
        <name>adobe.VideoFile</name>
        <extension>avf</extension>
        <description>Adobe Video File</description>
        <contentType>application/vnd.adobe.video-file</contentType>
        <icon>
            <image16x16>icons/AIRApp_16.png</image16x16>
            <image32x32>icons/AIRApp_32.png</image32x32>
            <image48x48>icons/AIRApp_48.png</image48x48>
            <image128x128>icons/AIRApp_128.png</image128x128>
        </icon>
    </fileType>
</fileTypes>
```

## font

Beschrijft één eigen lettertype dat in de AIR-toepassing kan worden gebruikt.

**Bovenliggend element:**“embedFonts” op pagina 229

**Onderliggende elementen:**“fontName” op pagina 235, “fontPath” op pagina 235

### Inhoud

Elementen waarmee de naam van het eigen lettertype en het bijbehorende pad worden opgegeven.

**Voorbeeld**

```
<font>
   <fontPath>ttf/space age.ttf</fontPath>
   <fontName>space age</fontName>
</font>
```

## fontName

Hiermee wordt de naam van het eigen lettertype opgegeven.

**Bovenliggend element:**"font" op pagina 234

**Onderliggende elementen:** geen

**Inhoud**

Naam van het eigen lettertype dat in StageText.fontFamily moet worden opgegeven

**Voorbeeld**

```
<fontName>space age</fontName>
```

## fontPath

Hiermee wordt de locatie van het bestand met de aangepaste lettertypen opgegeven.

**Bovenliggend element:**"font" op pagina 234

**Onderliggende elementen:** geen

**Inhoud**

Pad van het bestand met het eigen lettertype (ten opzichte van de bron).

**Voorbeeld**

```
<fontPath>ttf/space age.ttf</fontPath>
```

## forceCPURenderModeForDevices

**Adobe AIR 3.7 of hoger, alleen iOS — Optioneel**

Forceer de CPU-rendermodus voor een opgegeven aantal apparaten. Met deze functie kunt u in feite de GPU-rendermodus selectief inschakelen voor de resterende iOS-apparaten.

U voegt deze tag toe als een onderliggend element van de `iPhone`-tag en geeft een lijst op waarin de modelnamen van apparaten met een spatie van elkaar zijn gescheiden. Tot de geldige modelnamen van apparaten behoren onder meer:

| | | |
|---|---|---|
| iPad1,1 | iPhone1,1 | iPod1,1 |
| iPad2,1 | iPhone1,2 | iPod2,1 |
| iPad2,2 | iPhone2,1 | iPod3,3 |
| iPad2,3 | iPhone3.1 | iPod4,1 |
| iPad2,4 | iPhone3,2 | iPod5,1 |
| iPad2,5 | iPhone4,1 | |

| | | |
|---|---|---|
| iPad3,1 | iPhone5,1 | |
| iPad3,2 | | |
| iPad3,3 | | |
| iPad3,4 | | |

**Bovenliggend element:**“iPhone” op pagina 242, “initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Lijst met via een spatie van elkaar gescheiden modelnamen van apparaten.

**Attributen:**
Geen.

**Voorbeelden**
iOS:

```
...
<renderMode>GPU</renderMode>
...
<iPhone>
...
   <forceCPURenderModeForDevices>iPad1,1 iPhone1,1 iPhone1,2 iPod1,1
   </forceCPURenderModeForDevices>
</iPhone>
```

## fullScreen

**Adobe AIR 2.0 of hoger, iOS en Android — Optioneel**

Geeft op of de toepassing in volledigschermmodus wordt gestart.

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
`true` of `false` (standaard)

**Voorbeeld**

```
<fullscreen>true</fullscreen>
```

## height

**Adobe AIR 1.0 of hoger — Optioneel**

De aanvankelijke hoogte van het hoofdvenster van de toepassing.

Als u geen hoogte instelt, wordt deze bepaald door de instellingen in het SWF-hoofdbestand, of, in het geval van een AIR-toepassing op basis van HTML, door het besturingssysteem.

De maximale hoogte van een venster is in AIR 2 gewijzigd van 2048 pixels in 4096 pixels.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Een positief geheel getal met een maximale waarde van 4095.

**Voorbeeld**
```
<height>4095</height>
```

## icon

**Adobe AIR 1.0 of hoger — Optioneel**

De eigenschap `icon` geeft een of meer pictogrambestanden op die voor de toepassing moeten worden gebruikt. Een pictogram is optioneel. Als u de eigenschap `icon` niet opgeeft, gebruikt het besturingssysteem een standaardpictogram.

Het pad wordt opgegeven op basis van de hoofdmap van de toepassing. Pictogrambestanden moeten de PNG-indeling hebben. U kunt de volgende pictogramgroottes opgeven:

Als een element voor een bepaalde grootte aanwezig is, moet de afbeelding in het bestand exact de opgegeven grootte hebben. Als niet alle groottes aanwezig zijn, wordt de grootte die daar het dichtst bij aansluit, door het besturingssysteem aangepast voor een bepaald gebruik van het pictogram.

***Opmerking:*** *de opgegeven pictogrammen worden niet automatisch toegevoegd aan het AIR-pakket. De pictogrambestanden moeten in hun correcte relatieve locaties worden opgenomen wanneer de toepassing wordt verpakt.*

Voor de beste resultaten geeft u voor elke beschikbare grootte een afbeelding op. Bovendien moeten de pictogrammen er goed uitzien in zowel 16- als 32-bit kleuren.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:**"imageNxN" op pagina 238

**Inhoud**
Een imageNxN-element voor elk gewenst pictogramformaat.

**Voorbeeld**
```
<icon>
    <image16x16>icons/smallIcon.png</image16x16>
    <image32x32>icons/mediumIcon.png</image32x32>
    <image48x48>icons/bigIcon.png</image48x48>
    <image128x128>icons/biggestIcon.png</image128x128>
</icon>
```

## id

**Adobe AIR 1.0 of hoger — Vereist**

Een identificatiereeks die uniek is voor de toepassing; wordt ook toepassings-id genoemd. Een omgekeerde DNS-stijlidentificatie wordt vaak gebruikt, maar deze stijl is niet vereist.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:** geen

### Inhoud

De ID-waarde is beperkt tot de volgende tekens:

- 0-9
- a-z
- A-Z
- . (punt)
- - (koppelteken)

De waarde moet uit 1 tot 212 tekens bestaan. Dit element is vereist.

### Voorbeeld

```
<id>org.example.application</id>
```

## imageNxN

**Adobe AIR 1.0 of hoger — Optioneel**

Definieert het pad naar een pictogram in verhouding tot de toepassingsmap.

De volgende pictogramafbeeldingen kunnen worden gebruikt. Elke geeft een ander pictogramformaat op:

- image16x16
- image29x29 (AIR 2+)
- image32x32
- image36x36 (AIR 2.5+)
- image48x48
- image50x50 (AIR 3.4+)
- image57x57 (AIR 2+)
- image58x58 (AIR 3.4+)
- image72x72 (AIR 2+)
- image100x100 (AIR 3.4+)
- image114x114 (AIR 2.6+)
- image128x128
- image144x144 (AIR 3.4+)
- image512x512 (AIR 2+)
- image1024x1024 (AIR 3.4+)

Het pictogram moet een PNG-afbeelding zijn die exact even groot is als is aangeven door het afbeeldingselement. Pictogrambestanden moeten worden opgenomen in het toepassingspakket; pictogrammen waarnaar wordt verwezen in de toepassingsdescriptor, worden niet automatisch opgenomen.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

Het bestandspad naar het pictogram kan alle Unicode-tekens (UTF-8) bevatten behalve de volgende, die niet mogen worden gebruikt als bestandsnamen op verschillende bestandssystemen:

| Teken | Hexadecimale code |
|---|---|
| *diverse* | 0x00 - x1F |
| * | x2A |
| " | x22 |
| : | x3A |
| > | x3C |
| < | x3E |
| ? | x3F |
| \ | x5C |
| \| | x7C |

**Voorbeeld**

```
<image32x32>icons/icon32.png</image32x32>
```

## InfoAdditions

**Adobe AIR 1.0 of hoger — Optioneel**

Hiermee kunt u aanvullende eigenschappen van een iOS-toepassing opgeven.

**Bovenliggend element:**“iPhone” op pagina 242

**Onderliggende elementen:**iOS Info.plist-elementen

**Inhoud**

Bevat onderliggende elementen waarin sleutelwaardeparen worden opgegeven die moeten worden gebruikt als instellingen Info.plist voor de toepassing. Inhoud van het element InfoAdditions moet zijn opgenomen in een CDATA-blok.

Zie Information Property List Key Reference in de Apple iPhone Reference Library voor informatie over de sleutelwaardeparen en hoe u deze kunt uitdrukken in XML.

**Voorbeeld**

```
<InfoAdditions>
    <![CDATA[
        <key>UIStatusBarStyle</key>
        <string>UIStatusBarStyleBlackOpaque</string>
        <key>UIRequiresPersistentWiFi</key>
        <string>NO</string>
    ]]>
</InfoAdditions>
```

### Meer Help-onderwerpen

"iOS-instellingen" op pagina 86

## initialWindow

**Adobe AIR 1.0 of hoger — Vereist**

Definieert het hoofdinhoudsbestand en de aanvankelijke opmaak van de toepassing.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** Alle volgende elementen kunnen voorkomen als onderliggende elementen van het element initialWindow. Sommige elementen worden echter genegeerd, afhankelijk van of AIR vensters op een platform ondersteunt:

| Element | Desktop | Mobile |
|---|---|---|
| "aspectRatio" op pagina 224 | genegeerd | gebruikt |
| "autoOrients" op pagina 224 | genegeerd | gebruikt |
| "content" op pagina 226 | gebruikt | gebruikt |
| "depthAndStencil" op pagina 228 | gebruikt | gebruikt |
| "fullScreen" op pagina 236 | genegeerd | gebruikt |
| "height" op pagina 236 | gebruikt | genegeerd |
| "maximizable" op pagina 244 | gebruikt | genegeerd |
| "maxSize" op pagina 244 | gebruikt | genegeerd |
| "minimizable" op pagina 245 | gebruikt | genegeerd |
| "minSize" op pagina 245 | gebruikt | genegeerd |
| "renderMode" op pagina 248 | gebruikt (AIR 3.0 of hoger) | gebruikt |
| "requestedDisplayResolution" op pagina 248 | gebruikt (AIR 3.6 of hoger) | genegeerd |
| "resizable" op pagina 249 | gebruikt | genegeerd |
| "softKeyboardBehavior" op pagina 250 | genegeerd | gebruikt |
| "systemChrome" op pagina 252 | gebruikt | genegeerd |
| "title" op pagina 253 | gebruikt | genegeerd |
| "transparent" op pagina 253 | gebruikt | genegeerd |
| "visible" op pagina 255 | gebruikt | genegeerd |
| "width" op pagina 255 | gebruikt | genegeerd |
| "x" op pagina 256 | gebruikt | genegeerd |
| "y" op pagina 256 | gebruikt | genegeerd |

**Inhoud**
Onderliggende elementen die de opmaak en functionaliteit van de toepassing definiëren.

**Voorbeeld**
```
<initialWindow>
    <title>Hello World</title>
    <content>
        HelloWorld.swf
    </content>
    <depthAndStencil>true</depthAndStencil>
    <systemChrome>none</systemChrome>
    <transparent>true</transparent>
    <visible>true</visible>
    <maxSize>1024 800</maxSize>
    <minSize>320 240</minSize>
    <maximizable>false</maximizable>
    <minimizable>false</minimizable>
    <resizable>true</resizable>
    <x>20</x>
    <y>20</y>
    <height>600</height>
    <width>800</width>
    <aspectRatio>landscape</aspectRatio>
    <autoOrients>true</autoOrients>
    <fullScreen>false</fullScreen>
    <renderMode>direct</renderMode>
</initialWindow>
```

## installFolder

**Adobe AIR 1.0 of hoger — Optioneel**

Identificeert de submap van de hoofdinstallatiemap.

In Windows is de map Program Files ingesteld als standaardinstallatiemap. In Mac OS is dit de map /Applications. Bij Linux is dit /opt/. Voorbeeld: als de eigenschap `installFolder` is ingesteld op `"Acme"` en een toepassing de naam `"ExampleApp"` heeft, wordt de toepassing in Windows in C:\Program Files\Acme\ExampleApp, in MacOS in /Applications/Acme/Example.app en in Linux op /opt/Acme/ExampleApp geïnstalleerd.

De eigenschap `installFolder` is optioneel. Als u de eigenschap `installFolder` niet opgeeft, wordt de toepassing geïnstalleerd in een submap van de standaard installatiemap op basis van de eigenschap `name`.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:** geen

**Inhoud**
De eigenschap `installFolder` mag een willekeurig Unicode-teken (UTF-8) bevatten, behalve de tekens die in verschillende bestandssystemen niet in een mapnaam mogen worden gebruikt (zie de eigenschap `filename` hierboven voor de lijst van uitzonderingen).

Gebruik de schuine streep (/) als mapscheidingsteken als u een geneste submap wilt opgeven.

**Voorbeeld**
```
<installFolder>utilities/toolA</installFolder>
```

## iPhone

**Adobe AIR 2.0, alleen iOS — Optioneel**

Definieert iOS-specifieke toepassingseigenschappen..

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:**

- "Entitlements" op pagina 230
- "externalSwfs" op pagina 232
- "forceCPURenderModeForDevices" op pagina 235
- "InfoAdditions" op pagina 239
- "requestedDisplayResolution" op pagina 248

### Meer Help-onderwerpen

"iOS-instellingen" op pagina 86

## manifest

**Adobe AIR 2.5 of hoger, alleen Android — Optioneel**

Geeft informatie op om toe te voegen aan het Android-manifestbestand voor de toepassing.

**Bovenliggend element:**"manifestAdditions" op pagina 243

**Onderliggende elementen:** Gedefinieerd door de Android SDK.

### Inhoud

Het manifest-element is technisch gezien geen onderdeel van het AIR-toepassingsdescriptorschema. Dit is het hoofdelement van de het XML-document AndroidManifest. Alle inhoud die u in het manifest-element invoert, moet overeenkomen met het AndroidManifest.xml-schema. Wanneer u een APK-bestand maakt met de AIR-hulpmiddelen, wordt informatie in het manifestelement gekopieerd naar het overeenkomstige onderdeel van de gegenereerde AndroidManifest.xml van de toepassing.

Als u Android manifest-waarden opgeeft die alleen beschikbaar zijn in een meer recente SDK-versie dan de versie die rechtstreeks wordt ondersteund door AIR, moet u de markering `-platformsdk` instellen op ADT wanneer u de toepassing in het pakket bundelt. Stel de markering voor het bestandssysteempad in op een versie van de Android SDK die ondersteuning biedt voor de waarden die u toevoegt.

Het manifestelement zelf moet zijn opgenomen in een CDATA-blok binnen de AIR-toepassingsdescriptor.

### Voorbeeld

```
<![CDATA[
    <manifest android:sharedUserID="1001">
        <uses-permission android:name="android.permission.CAMERA"/>
        <uses-feature android:required="false" android:name="android.hardware.camera"/>
        <application android:allowClearUserData="true"
                    android:enabled="true"
                    android:persistent="true"/>
    </manifest>
]]>
```

### Meer Help-onderwerpen

"Android-instellingen" op pagina 80

Het bestand AndroidManifest.xml

## manifestAdditions

**Adobe AIR 2.5 en hoger, alleen Android**

Geeft informatie op die wordt toegevoegd aan het Android-manifestbestand.

Elke Android-toepassing bevat een manifestbestand dat basistoepassingseigenschappen definieert. Het Android-manifest is qua concept gelijk aan de AIR-toepassingsdescriptor. Een AIR-voor-Android-toepassing heeft zowel een toepassingsdescriptor als een automatisch gegenereerd Android-manifestbestand. Wanneer AIR-for-Android-toepassing is verpakt, wordt de informatie in dit element `manifestAdditions` toegevoegd aan de overeenkomstige delen van het Android-manifestdocument.

**Bovenliggend element:**"android" op pagina 220

**Onderliggende elementen:**"manifest" op pagina 242

### Inhoud

Informatie in het element `manifestAdditions` wordt toegevoegd aan het document AndroidManifest.xml.

Door AIR worden verschillende manifestvermeldingen in het gegenereerde Android-manifestdocument ingesteld om te garanderen dat de toepassings- en runtimefuncties correct werken. U kunt de volgende instellingen niet overschrijven:

U kunt de volgende attributen niet instellen voor het element manifest:

- package
- android:versionCode
- android:versionName

U kunt de volgende attributen niet instellen voor het hoofdelement activity:

- android:label
- android:icon

U kunt de volgende attributen niet instellen voor het element application:

- android:theme
- android:name
- android:label
- android:windowSoftInputMode
- android:configChanges
- android:screenOrientation
- android:launchMode

**Voorbeeld**

```
<manifestAdditions>
    <![CDATA[
        <manifest android:installLocation="preferExternal">
            <uses-permission android:name="android.permission.INTERNET"/>
            <application android:allowClearUserData="true"
                         android:enabled="true"
                         android:persistent="true"/>
        </manifest>
    ]]>
</manifestAdditions>
```

### Meer Help-onderwerpen

"Android-instellingen" op pagina 80

Het bestand AndroidManifest.xml

## maximizable

**Adobe AIR 1.0 of hoger — Optioneel**

Hiermee wordt aangegeven of het venster kan worden gemaximaliseerd.

***Opmerking:*** *in besturingssystemen waar het maximaliseren van een venster betekent dat de grootte van het venster wordt aangepast (bijvoorbeeld Max OS X), moet zowel maximizable als resizable op* `false` *worden ingesteld om te voorkomen dat op het venster wordt ingezoomd of dat de grootte van het venster wordt aangepast.*

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

`true` (standaard) of `false`

**Voorbeeld**

```
<maximizable>false</maximizable>
```

## maxSize

**Adobe AIR 1.0 of hoger — Optioneel**

De maximumgrootte van het venster. Als u geen maximumgrootte instelt, wordt deze bepaald door het besturingssysteem.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

Twee gehele getallen die de maximale breedte en hoogte bepalen, gescheiden door een spatie.

***Opmerking:*** *de maximale door AIR ondersteunde venstergrootte is in AIR 2 van 2048 x 2048 pixels omhoog gegaan naar 4096 x 4096 pixels. (Omdat de schermcoördinaten op 0 zijn gebaseerd, is de maximale waarde die u voor hoogte kunt gebruiken, 4095.)*

**Voorbeeld**

```
<maxSize>1024 360</maxSize>
```

## minimizable

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft op of het venster kan worden geminimaliseerd.

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

`true` (standaard) of `false`

**Voorbeeld**

```
<minimizable>false</minimizable>
```

## minSize

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft de minimale toegestane grootte voor het venster op.

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

Twee gehele getallen die de minimale breedte en hoogte bepalen, gescheiden door een spatie. Let op: de minimumgrootte die door het besturingssysteem wordt bepaald, heeft voorrang ten opzichte van de waarde in de toepassingsdescriptor.

**Voorbeeld**

```
<minSize>120 60</minSize>
```

## name

**Adobe AIR 1.0 of hoger — Optioneel**

De toepassingstitel die wordt weergegeven door het AIR-toepassingsinstallatieprogramma.

Als geen `name`-element is opgegeven, geeft het installatieprogramma van de AIR-toepassing de waarde van `filename` weer als de toepassingsnaam.

**Bovenliggend element:**“application” op pagina 220

**Onderliggende elementen:**“tekst” op pagina 252

**Inhoud**

Als u een enkele tekstnode opgeeft (in plaats van meerdere `<text>`-elementen), gebruikt de AIR-toepassing deze naam, onafhankelijk van de systeemtaal.

In het descriptorschema van de AIR 1.0-toepassing kan slechts één enkelvoudig tekstknooppunt voor de naam worden gedefinieerd (niet meerdere `text`-elementen). In AIR 1.1 (of hoger) kunt u meerdere talen opgeven in het element `name`.

Het attribuut `xml:lang` voor elk tekstelement geeft een taalcode op die gedefinieerd is in RFC4646 (http://www.ietf.org/rfc/rfc4646.txt).

Het installatieprogramma van de AIR-toepassing gebruikt de naam die het best overeenkomt met de gebruikersinterfacetaal van het besturingssysteem van de gebruiker. Voorbeeld: een installatie waarbij het element `name` van het descriptorbestand van de toepassing een waarde bevat voor de landinstelling 'en' (Engels). Het installatieprogramma van de AIR-toepassing gebruikt de naam 'en' als het besturingssysteem 'en' (Engels) identificeert als de gebruikersinterfacetaal. Het installatieprogramma gebruikt ook de naam 'en' als de gebruikersinterfacetaal van het systeem 'en-US' (Amerikaans Engels) is. Als de gebruikersinterfacetaal echter 'en-US' is en het descriptorbestand van de toepassing zowel de naam 'en-US' als de naam 'en-GB' definieert, gebruikt het installatieprogramma van de AIR-toepassing de waarde 'en-US'. Als de toepassing geen naam definieert die overeenkomt met de gebruikersinterfacetalen van het systeem, gebruikt het installatieprogramma van de AIR-toepassing de eerste `name`-waarde die in het descriptorbestand van de toepassing is gedefinieerd.

Het element `name` definieert alleen de toepassingstitel die in het installatieprogramma van de AIR-toepassing wordt gebruikt. Het AIR-toepassingsinstallatieprogramma ondersteunt meerdere talen: Traditioneel Chinees, Vereenvoudigd Chinees, Tsjechisch, Nederlands, Engels, Frans, Duits, Italiaans, Japans, Koreaans, Pools, Braziliaans Portugees, Russisch, Spaans, Zweeds en Turks. Het installatieprogramma van de AIR-toepassing kiest de schermtaal (voor andere tekst dan de toepassingstitel en -beschrijving) op basis van de gebruikersinterfacetaal van het systeem. Deze taalkeuze is onafhankelijk van de instellingen in het descriptorbestand van de toepassing.

Het element `name` definieert *niet* de landinstellingen die beschikbaar zijn voor de actieve, geïnstalleerde toepassing. Zie "AIR-toepassingen lokaliseren" op pagina 306 voor meer informatie over het ontwikkelen van meertalige toepassingen.

**Voorbeeld**
In het volgende voorbeeld wordt een naam gedefinieerd met een eenvoudige tekstnode:

```
<name>Test Application</name>
```

In het volgende voorbeeld, geldig in AIR 1.1 en latere versies, wordt de naam in drie talen opgegeven (Engels, Frans en Spaans) met behulp van <text>-elementnodes:

```
<name>
    <text xml:lang="en">Hello AIR</text>
    <text xml:lang="fr">Bonjour AIR</text>
    <text xml:lang="es">Hola AIR</text>
</name>
```

## name

**Adobe AIR 1.0 of hoger — Vereist**

Identificeer de naam van een bestandstype.

**Bovenliggend element:**"fileType" op pagina 233

**Onderliggende elementen:** geen

**Inhoud**
Een tekenreeks die de naam van het bestandstype vertegenwoordigt.

**Voorbeeld**

```
<name>adobe.VideoFile</name>
```

## programMenuFolder

**Adobe AIR 1.0 of hoger — Optioneel**

Identificeert de locatie waar de snelkoppelingen naar de toepassing moeten worden geplaatst in het menu Alle programma's van het Windows-besturingssysteem of het Toepassingen-menu in Linux. (Deze instelling wordt momenteel genegeerd in andere besturingssystemen.)

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

De tekenreeks die wordt gebruikt voor `programMenuFolder`, mag een willekeurig Unicode-teken (UTF-8) bevatten, behalve de tekens die in verschillende bestandssystemen niet in een mapnaam mogen worden gebruikt (zie het element `filename` hierboven voor de lijst van uitzonderingen). Gebruik *geen* schuine streep (/) als laatste teken van deze waarde.

**Voorbeeld**

```
<programMenuFolder>Example Company/Sample Application</programMenuFolder>
```

## publisherID

**Adobe AIR 1.5.3 of hoger — Optioneel**

Identificeert de uitgevers-ID voor het bijwerken van een AIR-toepassing die oorspronkelijk is gemaakt met AIR versie 1.5.2 of eerder.

Geef alleen een uitgevers-ID op bij het maken van een toepassingsupdate. De waarde van het element `publisherID` moet overeenkomen met de uitgevers-ID die wordt gegenereerd door AIR voor de vroegere versie van de toepassing. Voor een geïnstalleerde toepassing kunt u de uitgevers-ID vinden in de map waarin een toepassing is geïnstalleerd, in het bestand `META-INF/AIR/publisherid`.

Nieuwe toepassingen die zijn gemaakt met AIR 1.5.3 of hoger, moeten geen uitgevers-ID opgeven.

Zie "Informatie over uitgevers-id's voor AIR" op pagina 200 voor meer informatie.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

Een uitgevers-ID-tekenreeks.

**Voorbeeld**

```
<publisherID>B146A943FBD637B68C334022D304CEA226D129B4.1</publisherID>
```

## renderMode

**Adobe AIR 2.0 of hoger — Optioneel**

Specificeert of versnelling van GPU (Graphics Processing Unit: grafische verwerkingseenheid) wordt gebruikt, indien ondersteund op het huidige apparaat.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

Een van de volgende waarden:

- `auto` (standaard) — op dit moment wordt CPU-modus gebruikt.
- `cpu` — hardwareversnelling wordt niet gebruikt.
- `direct` — rendercompositie vindt plaats in de CPU; blitting maakt gebruik van de GPU. Beschikbaar in AIR 3 en hoger.

  ***Opmerking:*** *voor een optimale GPU-versnelling van Flash-inhoud met AIR for Mobile-platforms raadt Adobe u aan om de instelling renderMode="direct" te gebruiken (dat wil zeggen: Stage3D) in plaats van renderMode="gpu". Adobe biedt officieel ondersteuning voor de volgende Stage3D-frameworks: Starling (2D) en Away3D (3D). Adobe raadt u daarom aan deze frameworks te gebruiken. Voor meer informatie over Stage3D en Starling/Away3D gaat u naar http://gaming.adobe.com/getstarted/.*

- `gpu` — hardwareversnelling wordt gebruikt, indien beschikbaar.

  ***Belangrijk:*** *gebruik de GPU-renderingsmodus niet voor Flex-toepassingen.*

**Voorbeeld**

```
<renderMode>direct</renderMode>
```

## requestedDisplayResolution

**Adobe AIR 2.6 en later, alleen iOS; Adobe AIR 3.6 en later, OS X — Optioneel**

Geeft op of de standaardresolutie of hoge resolutie moet worden gebruikt voor een apparaat of computermonitor met een hoge-resolutiescherm. Bij instelling op *standard*, de standaard, wordt het scherm als scherm met standaardresolutie voor de toepassing weergegeven. Bij instelling op *high* kan de toepassing elke hoge-resolutiepixel aanspreken.

Op een 640 x 960 iPhone-scherm met hoge resolutie zijn bij de instelling *standard* de afmetingen van het werkgebied bij volledig scherm 320 x 480 en wordt elke toepassingspixel weergegeven met vier schermpixels. Als de instelling *high* is, zijn de afmetingen van het werkgebied bij volledig scherm 640 x 960.

Op apparaten met een scherm met standaardresolutie komen de afmetingen van het werkgebied overeen met de schermafmetingen, onafhankelijk van de instelling.

Als het `requestedDisplayResolution`-element genest is binnen het `iPhone`-element, is het van toepassing op iOS-apparaten. In dat geval kan het `excludeDevices`-attribuut worden gebruikt om apparaten op te geven waarvoor de instelling niet wordt toegepast.

Als het `requestedDisplayResolution`-element genest is binnen het `initialWindow`-element, is het van toepassing op AIR-bureaubladtoepassingen op MacBook Pro-computers die hoge-resolutieschermen ondersteunen. De opgegeven waarde is van toepassing op alle native vensters die in de toepassing worden gebruikt. Het nesten van het `requestedDisplayResolution`-element in het `initialWindow`-element wordt ondersteund in AIR 3.6 en later.

**Bovenliggend element:**“iPhone” op pagina 242, “initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Ofwel *standard*, de standaard, of *high*.

**Attribuut:**
excludeDevices — een door spaties gescheiden lijst met iOS-modelnamen of voorvoegsels van modelnamen. Hierdoor kan de ontwikkelaar bepaalde apparaten met hoge resolutie gebruiken en andere met standaardresolutie. Dit attribuut is alleen beschikbaar op iOS (het `requestedDisplayResolution`-element is genest in het `iPhone`-element). Het `excludeDevices`-attribuut is beschikbaar in AIR 3.6 en later.

Voor elk apparaat waarvan de modelnaam is opgegeven in dit attribuut, is de `requestedDisplayResolution`-waarde het tegenovergestelde van de opgegeven waarde. Met andere woorden, als de `requestedDisplayResolution`-waarde *high* is, gebruiken de uitgesloten apparaten de standaardresolutie. Als de `requestedDisplayResolution`-waarde *standard* is, gebruiken de uitgesloten apparaten de hoge resolutie.

De waarden zijn modelnamen of voorvoegsels van modelnamen van iOS-apparaten. De waarde iPad3,1, bijvoorbeeld, verwijst specifiek naar een iPad van de derde generatie met Wi-Fi (en niet naar GSM- of CDMA-versie van iPad's van de derde generatie). De waarde iPad3 verwijst naar elke iPad van de derde generatie. Een niet-officiële lijst met iOS-modelnamen vindt u op de iPhone Wiki Models-pagina.

**Voorbeelden**
Desktop:

```
<initialWindow>
    <requestedDisplayResolution>high</requestedDisplayResolution>
</initialWindow>
```

iOS:

```
<iPhone>
    <requestedDisplayResolution excludeDevices="iPad3
iPad4">high</requestedDisplayResolution>
</iPhone>
```

## resizable

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft op of de grootte van het venster kan worden gewijzigd.

***Opmerking:*** *in besturingssystemen waar het maximaliseren van een venster betekent dat de grootte van het venster wordt aangepast (bijvoorbeeld Max OS X), moet zowel maximizable als resizable op* `false` *worden ingesteld om te voorkomen dat op het venster wordt ingezoomd of dat de grootte van het venster wordt aangepast.*

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:**

**Inhoud**
`true` (standaard) of `false`

**Voorbeeld**
```
<resizable>false</resizable>
```

## softKeyboardBehavior

**Adobe AIR 2.6 of hoger, mobiel profiel — Optioneel**

Geeft de standaardfunctionaliteit van de toepassing op wanneer een virtueel toetsenbord wordt weergegeven. De standaardfunctionaliteit is dat de toepassing naar boven wordt gepand. De runtime houdt het tekstveld of het interactieve object dat de focus heeft, op het scherm. Gebruik de optie *pan* als uw toepassing geen eigen logica voor toetsenbordafhandeling heeft.

U kunt de automatische functionaliteit ook uitschakelen door het element `softKeyboardBehavior` in te stellen op *none*. In dit geval versturen tekstvelden en interactieve objecten een SoftKeyboardEvent wanneer het softwaretoetsenbord wordt weergegeven, maar de runtime de toepassing niet pant of het formaat van de toepassing niet aanpast. Het tekstinvoergebied wordt door uw toepassing in beeld gehouden.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Ofwel *none* of *pan*. De standaardwaarde is *pan*.

**Voorbeeld**
```
<softKeyboardBehavior>none</softKeyboardBehavior>
```

### Meer Help-onderwerpen

SoftKeyboardEvent

## supportedLanguages

**Adobe AIR 3.2 of hoger — Optioneel**

Geeft aan welke talen worden ondersteund door de toepassing. Dit element wordt alleen gebruikt door iOS-, Mac captive runtime- en Android-toepassingen. Dit element wordt genegeerd door alle overige toepassingstypen.

Als u dit element niet opgeeft, worden standaard de volgende acties uitgevoerd door de pakketsoftware, afhankelijk van het toepassingstype:

- iOS — Alle talen die door de AIR-runtime worden ondersteund, zijn opgenomen in de iOS App Store als talen die door de toepassing worden ondersteund.
- Mac captive runtime — De toepassing die is opgenomen in het pakket met de captive-bundel bevat geen lokalisatiegegevens.
- Android — De toepassingsbundel heeft bronnen voor alle talen die worden ondersteund door de AIR-runtime.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

Een door spaties gescheiden lijst met ondersteunde talen. De volgende ISO 639-1-taalwaarden zijn geldig en representeren de talen die door de AIR-runtime worden ondersteund: `en`, `de`, `es`, `fr`, `it`, `ja`, `ko`, `pt`, `ru`, `cs`, `nl`, `pl`, `sv`, `tr`, `zh`, `da`, `nb`, `iw`.

De pakketsoftware genereert een fout als de waarde van het element `<supportedLanguages>` leeg is.

***Opmerking:*** *gelokaliseerde tags (zoals de naamtag) negeren de waarde van een taal als u de tag `<supportedLanguages>` gebruikt en deze de desbetreffende taal niet bevat. Als een native extensie bronnen heeft voor een taal die niet wordt aangeduid door de tag `<supportedLangauges>`, wordt er een waarschuwing weergegeven en worden de bronnen voor die taal genegeerd.*

**Voorbeeld**

```
<supportedLanguages>en ja fr es</supportedLanguages>
```

## supportedProfiles

**Adobe AIR 2.0 of hoger — Optioneel**

Hiermee worden de profielen geïdentificeerd die voor de toepassing worden ondersteund.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**

U kunt al deze waarden opnemen in het element `supportedProfiles`:

- `desktop`—Het bureaubladprofiel is voor AIR-toepassingen die zijn geïnstalleerd op een computer en gebruikmaken van een AIR-bestand. Deze toepassingen hebben geen toegang tot de klasse NativeProcess (die zorgt voor communicatie met native toepassingen).
- `extendedDesktop`—Het uitgebreide bureaubladprofiel is voor AIR-toepassingen die zijn geïnstalleerd op een bureaubladcomputer met behulp van een native toepassingsinstallatieprogramma. Deze toepassingen hebben toegang tot de klasse NativeProcess (die zorgt voor communicatie met native toepassingen).
- `mobileDevice`—Het profiel voor uitgebreide mobiele apparaten is voor mobiele toepassingen.
- `extendedMobileDevice`—Het profiel voor uitgebreide mobiele apparaten wordt momenteel niet gebruikt.

De eigenschap `supportedProfiles` is optioneel. Als u dit element niet opneemt in het descriptorbestand voor de toepassing, kan de toepassing voor elk profiel worden gecompileerd en geïmplementeerd.

Als u meerdere profielen wilt opgeven, gebruikt u een spatie als scheidingsteken. Met de volgende instelling wordt bijvoorbeeld aangegeven dat de toepassing alleen beschikbaar is in de profielen desktop en extended:

```
<supportedProfiles>desktop extendedDesktop</supportedProfiles>
```

***Opmerking:*** *wanneer u een toepassing uitvoert met ADL en geen waarde opgeeft voor de optie `-profile` wordt het eerste profiel in de toepassingsdescriptor gebruikt. (Als er ook geen profielen zijn opgegeven in de toepassingsdescriptor, wordt het bureaubladprofiel gebruikt.)*

**Voorbeeld**

```
<supportedProfiles>desktop mobileDevice</supportedProfiles>
```

### Meer Help-onderwerpen

"Apparaatprofielen" op pagina 257

"Ondersteunde profielen" op pagina 79

## systemChrome

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft op of het eerste toepassingsvenster is gemaakt met de standaardtitelbalk, -randen en -besturingselementen die worden geleverd door het besturingssysteem.

De systeeminterface-instelling van het venster kan niet bij uitvoering worden gewijzigd.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

### Inhoud

Een van de volgende waarden:

- `none` — Er wordt geen systeeminterface geleverd. De toepassing (of een toepassingsframework als Flex) is verantwoordelijk voor het weergeven van een vensteromlijsting.
- `standard` (standaard) — Systeeminterface wordt opgegeven door het besturingsysteem.

### Voorbeeld

```
<systemChrome>standard</systemChrome>
```

## tekst

**Adobe AIR 1.1 of hoger — Optioneel**

Geeft een gelokaliseerde tekenreeks op.

Het attribuut `xml:lang` van een tekstelement geeft een taalcode op, zoals gedefinieerd in RFC4646 (http://www.ietf.org/rfc/rfc4646.txt).

Het AIR-toepassingsinstallatieprogramma gebruikt het element `text` met de attribuutwaarde `xml:lang` die het meest overeenkomt met de interfacetaal van het besturingssysteem van de gebruiker.

Overweeg bijvoorbeeld een installatie waarin een element `text` een waarde bevat voor de landinstelling en (Engels). Het installatieprogramma van de AIR-toepassing gebruikt de naam 'en' als het besturingssysteem 'en' (Engels) identificeert als de gebruikersinterfacetaal. Het installatieprogramma gebruikt ook de naam 'en' als de gebruikersinterfacetaal van het systeem 'en-US' (Amerikaans Engels) is. Als de gebruikersinterfacetaal echter 'en-US' is en het descriptorbestand van de toepassing zowel de naam 'en-US' als de naam 'en-GB' definieert, gebruikt het installatieprogramma van de AIR-toepassing de waarde 'en-US'.

Als in de toepassing geen element `text` wordt gedefinieerd dat overeenkomt met de interfacetaal van het besturingssysteem, gebruikt het AIR-toepassingsinstallatieprogramma de eerste `name`-waarde die wordt gedefinieerd in het toepassingsdescriptorbestand.

**Bovenliggende elementen:**

- "name" op pagina 245

- "beschrijving" op pagina 228

**Onderliggende elementen:** geen

**Inhoud**
Een attribuut `xml:lang` dat een landinstelling en een tekenreeks van gelokaliseerde tekst opgeeft.

**Voorbeeld**
```
<text xml:lang="fr">Bonjour AIR</text>
```

## title

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft de titel op die wordt weergegeven in de titelbalk van het eerste toepassingsvenster.

Een titel wordt alleen weergegeven als het element `systemChrome` is ingesteld op `standard`.

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Een tekenreeks die de venstertitel bevat.

**Voorbeeld**
```
<title>Example Window Title</title>
```

## transparent

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft op of het eerste toepassingsvenster overvloeit met het bureaublad.

Een venster met ingeschakelde transparantie kan langzamer laden en meer geheugen gebruiken. De transparantie-instelling kan niet worden gewijzigd tijdens de runtime.

***Belangrijk:*** *U kunt* `transparent` *alleen op* `true` *instellen als* `systemChrome` *op* `none` *is ingesteld.*

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
`true` of `false` (standaard)

**Voorbeeld**
```
<transparent>true</transparent>
```

## version

**Adobe AIR 1.0 tot 2.0 — Vereist, Niet toegestaan in AIR 2.5 en hoger**

Geeft de versie-informatie voor de toepassing op.

De tekenreeks voor de versie is een benaming die door de toepassing wordt gedefinieerd. AIR interpreteert de tekenreeks voor de versie helemaal niet. Met andere woorden, versie '3.0' wordt niet als recenter dan versie '2.0' beschouwd. Voorbeelden: `"1.0"`, `".4"`, `"0.5"`, `"4.9"`, `"1.3.4a"`.

In AIR 2.5 en hoger hebben de elementen `versionNumber` en `versionLabel` voorrang op het element `version`.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**
Een tekenreeks die de toepassingsversie bevat.

**Voorbeeld**

```
<version>0.1 Alpha</version>
```

## versionLabel

**Adobe AIR 2.5 of hoger — Optioneel**

Geeft een leesbare versietekenreeks op.

In installatiedialoogvensters wordt de waarde van de versielabel weergegeven in plaats van de waarde van het element `versionNumber`. Als `versionLabel` niet wordt gebruikt, wordt het `versionNumber` voor beide gebruikt.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**
Een tekenreeks met de openlijk weergegeven versietekst.

**Voorbeeld**

```
<versionLabel>0.9 Beta</versionlabel>
```

## versionNumber

**Adobe AIR 2.5 en hoger — Vereist**

Het toepassingsversienummer.

**Bovenliggend element:**"application" op pagina 220

**Onderliggende elementen:** geen

**Inhoud**
Het versienummer kan een reeks bevatten van hoogstens drie gehele getallen, gescheiden door punten. Elk gehele getal moet een cijfer van 0 tot en met 999 zijn.

**Voorbeelden**

```
<versionNumber>1.0.657</versionNumber>

<versionNumber>10</versionNumber>

<versionNumber>0.01</versionNumber>
```

## visible

**Adobe AIR 1.0 of hoger — Optioneel**

Geeft op of het eerste toepassingsvenster zichtbaar is zodra het is gemaakt.

AIR-vensters, waaronder het eerste venster, worden standaard gemaakt in onzichtbare status. U kunt een venster weergeven door de methode `activate()` van het NativeWindow-object te gebruiken of door de eigenschap `visible` in te stellen op `true`. U wordt aangeraden het hoofdvenster in eerste instantie verborgen te houden zodat wijzigingen in de vensterpositie, venstergrootte en inhoudopmaak niet zichtbaar zijn.

Het venster wordt automatisch door de Flex-component `mx:WindowedApplication` weergegeven en geactiveerd voordat de gebeurtenis `applicationComplete` wordt verzonden, tenzij het attribuut `visible` is ingesteld op `false` in de MXML-definitie.

Op apparaten in het mobiele profiel, dat geen vensters ondersteunt, wordt de zichtbare instelling genegeerd.

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

`true` of `false` (standaard)

**Voorbeeld**

```
<visible>true</visible>
```

## width

**Adobe AIR 1.0 of hoger — Optioneel**

De eerst breedte van het hoofdvenster van de toepassing.

Als u geen breedte instelt, wordt deze bepaald door de instellingen in de SWF-hoofdbestand of. in het geval van een AIR-toepassing op basis van HTML, door het besturingssysteem.

De maximale breedte van een venster is in AIR 2 gewijzigd van 2048 pixels in 4096 pixels.

**Bovenliggend element:**“initialWindow” op pagina 240

**Onderliggende elementen:** geen

**Inhoud**

Een positief geheel getal met een maximale waarde van 4095.

**Voorbeeld**

```
<width>1024</width>
```

## x

**Adobe AIR 1.0 of hoger — Optioneel**

De horizontale positie van het eerste toepassingsvenster.

In de meeste gevallen is het beter om het besturingssysteem de eerste positie van het venster te laten bepalen in plaats van om een bepaalde waarde toe te kennen.

De bron van het schermcoördinaatsysteem (0,0) is de linkerbovenhoek van het hoofdscherm (zoals bepaald door het besturingssysteem).

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Een geheel getal.

**Voorbeeld**

```
<x>120</x>
```

## y

**Adobe AIR 1.0 of hoger — Optioneel**

De verticale positie van het eerste toepassingsvenster.

In de meeste gevallen is het beter om het besturingssysteem de eerste positie van het venster te laten bepalen in plaats van om een bepaalde waarde toe te kennen.

De bron van het schermcoördinaatsysteem (0,0) is de linkerbovenhoek van het hoofdscherm (zoals bepaald door het besturingssysteem).

**Bovenliggend element:**"initialWindow" op pagina 240

**Onderliggende elementen:** geen

**Inhoud**
Een geheel getal.

**Voorbeeld**

```
<y>250</y>
```

# Hoofdstuk 15: Apparaatprofielen

**Adobe AIR 2 of hoger**

Profielen zijn een mechanisme voor het definiëren van klassen computerapparaten waarop uw toepassing werkt. Een profiel definieert een set API's en mogelijkheden die normaal gesproken op een bepaalde apparaatklasse worden ondersteund. Voorbeelden van de beschikbare profielen:

- desktop
- extendedDesktop
- mobileDevice
- extendedMobileDevice

U kunt de profielen voor uw toepassing in de toepassingsdescriptor definiëren. Gebruikers van computers en apparaten in de opgenomen profielen kunnen uw toepassing installeren. Gebruikers van andere computers kunnen dat niet. Als u bijvoorbeeld alleen het bureaubladprofiel uw toepassingsdescriptor opneemt, kunnen gebruikers uw toepassing alleen op bureaubladcomputers installeren en uitvoeren.

Als u een profiel opneemt die uw toepassing niet ondersteunt, kunnen de ervaringen van de gebruikers in zulke omgevingen gering zijn. Als u geen profielen in de toepassingsdescriptor opgeeft, beperkt AIR uw toepassing niet. U kunt de toepassing in elk van de ondersteunde indelingen verpakken en gebruikers met apparaten van elk profiel kunnen deze installeren. De toepassing werkt echter bij runtime mogelijk niet naar behoren.

Waar mogelijk worden profielrestricties afgedwongen wanneer u uw toepassing verpakt. Als u bijvoorbeeld slechts het profiel extendedDesktop opneemt, kunt u uw toepassing niet als AIR-bestand opslaan, alleen als native installatieprogramma. Als u het profiel mobileDevice niet opneemt, kunt u uw toepassing niet als APK van Android inpakken.

Een enkele computer kan meer dan één profiel ondersteunen. AIR ondersteunt bijvoorbeeld op bureaubladcomputers toepassingen van zowel het profiel desktop als het profiel extendedDesktop. Een toepassing met profiel uitgebreid bureaublad kan echter communiceren met native processen en MOET zijn verpakt als native installatieprogramma (exe, dmg, deb of rpm). Een toepassing met bureaublad aan de andere kant kan niet communiceren met een native proces. Een toepassing met bureaubladprofiel kan worden verpakt als AIR-bestand of als native installatieprogramma.

Het opnemen van een functie in een profiel duidt erop dat ondersteuning van die functie algemeen is in de apparaatklasse waarvoor dat profiel is gedefinieerd. Het betekent echter niet dat elk apparaat in een profiel elke functie ondersteunt. De meeste, maar niet alle mobiele telefoons kunnen bijvoorbeeld een accelerometer ondersteunen. Klassen en functies die geen algemene ondersteuning genieten, hebben normaal gesproken een booleaanse eigenschap die u kunt controleren voordat u de functie gebruikt. In het geval van een accelerometer bijvoorbeeld kunt u de statische eigenschap `Accelerometer.isSupported` testen om te bepalen of het huidige apparaat een ondersteunde accelerometer bevat.

De volgende profielen kunnen worden toegewezen aan uw AIR-toepassing met behulp van het element supportedProfiles in de toepassingsdescriptor:

**Desktop** Het profiel Desktop definieert een serie functionaliteiten voor AIR-toepassingen die als AIR-bestanden op een desktopcomputer worden geïnstalleerd. Deze toepassingen worden geïnstalleerd en uitgevoerd op ondersteunde desktopplatforms (Mac OS, Windows en Linux). AIR-toepassingen die zijn ontwikkeld in versies van AIR ouder dan AIR 2 kunnen worden beschouwd als toepassingen met het profiel Desktop. Sommige API's werken niet in dit profiel. Zo kunnen toepassingen met het profiel Desktop niet met native processen communiceren.

**Uitgebreide desktop** Het uitgebreide bureaubladprofiel definieert een reeks mogelijkheden voor AIR-toepassingen die zijn verpakt en geïnstalleerd met een native installatieprogramma. Deze native installatieprogramma's zijn EXE-bestanden bij Windows, DMG-bestanden bij Mac OS en BIN-, DEB- of RPM-bestanden bij Linux. Toepassingen met het profiel Uitgebreide desktop hebben extra functies die niet beschikbaar zijn in toepassingen met het profiel Desktop. Zie voor meer informatie "Een eigen bureaubladinstallatieprogramma verpakken" op pagina 60.

**Mobiel apparaat** Het mobiele-apparaatprofiel definieert een reeks mogelijkheden voor toepassingen die zijn geïnstalleerd op mobiele apparaten zoals mobiele telefoons en tablets. Deze toepassingen kunnen worden installeerd en uitgevoerd op ondersteunde mobiele platforms, zoals Android, Blackberry Tablet OS, en iOS.

**Uitgebreid mobiel apparaat** Het profiel voor uitgebreide mobiele apparaten definieert een uitgebreide reeks mogelijkheden voor toepassingen die zijn geïnstalleerd op mobiele apparaten. Er zijn momenteel geen apparaten die dit profiel ondersteunen.

# Doelprofielen beperken in het descriptorbestand van de toepassing

**Adobe AIR 2 of hoger**

Vanaf AIR 2 bevat het toepassingsdescriptorbestand een element `supportedProfiles`, waarmee u doelprofielen kunt beperken. Met de volgende instelling wordt bijvoorbeeld aangegeven dat de toepassing alleen beschikbaar is in het profiel Desktop:

```
<supportedProfiles>desktop</supportedProfiles>
```

Wanneer dit element is ingesteld, kan de toepassing alleen worden verpakt in de profielen die u aangeeft. Gebruik de volgende waarden:

- `desktop`: het profiel Desktop
- `extendedDesktop`: het profiel Uitgebreide desktop
- `mobileDevice`: het profiel Mobiel apparaat

Het element `supportedProfiles` is optioneel. Als u dit element niet opneemt in het descriptorbestand van de toepassing, kan de toepassing voor elk profiel worden verpakt en geïmplementeerd.

Als u meerdere profielen in het element `supportedProfiles` wilt opgeven, plaatst u tussen de profielen een spatie als scheidingsteken, zoals hieronder:

```
<supportedProfiles>desktop extendedDesktop</supportedProfiles>
```

# Mogelijkheden van de verschillende profielen

**Adobe AIR 2 of hoger**

In de volgende tabel zijn de klassen en functies opgenomen die niet in alle profielen worden ondersteund.

| Klasse of functie | desktop | extendedDesktop | mobileDevice |
|---|---|---|---|
| Accelerometer (Accelerometer.isSupported) | Nee | Nee | Controleren |
| Accessibility (Capabilities.hasAccessibility) | Ja | Ja | Nee |
| Acoustische echo-opheffing (Microphone.getEnhancedMicrophone()) | Ja | Ja | Nee |
| ActionScript 2 | Ja | Ja | Nee |
| Matrix CacheAsBitmap | Nee | Nee | Ja |
| Camera (Camera.isSupported) | Ja | Ja | Ja |
| CameraRoll | Nee | Nee | Ja |
| CameraUI (CameraUI.isSupported) | Nee | Nee | Ja |
| Captive-runtimebundels | Ja | Ja | Ja |
| ContextMenu (ContextMenu.isSupported) | Ja | Ja | Nee |
| DatagramSocket (DatagramSocket.isSupported) | Ja | Ja | Ja |
| DockIcon (NativeApplication.supportsDockIcon) | Controleren | Controleren | Nee |
| Drag-and-drop (NativeDragManager.isSupported) | Ja | Ja | Controleren |
| EncyptedLocalStore (EncyptedLocalStore.isSupported) | Ja | Ja | Ja |
| Flash Access (DRMManager.isSupported) | Ja | Ja | Nee |
| GameInput (GameInput.isSupported) | Nee | Nee | Nee |
| Geolocation (Geolocation.isSupported) | Nee | Nee | Controleren |
| HTMLLoader (HTMLLoader.isSupported) | Ja | Ja | Nee |
| IME (IME.isSupported) | Ja | Ja | Controleren |
| LocalConnection (LocalConnection.isSupported) | Ja | Ja | Nee |
| Microphone (Microphone.isSupported) | Ja | Ja | Controleren |
| Meerkanaalsaudio (Capabilities.hasMultiChannelAudio()) | Nee | Nee | Nee |
| Native extensies | Nee | Ja | Ja |
| NativeMenu (NativeMenu.isSupported) | Ja | Ja | Nee |
| NativeProcess (NativeProcess.isSupported) | Nee | Ja | Nee |
| NativeWindow (NativeWindow.isSupported) | Ja | Ja | Nee |
| NetworkInfo (NetworkInfo.isSupported) | Ja | Ja | Controleren |
| Bestanden openen met standaardtoepassing | Beperkt | Ja | Nee |
| PrintJob (PrintJob.isSupported | Ja | Ja | Nee |

| Klasse of functie | desktop | extendedDesktop | mobileDevice |
|---|---|---|---|
| SecureSocket (SecureSocket.isSupported) | Ja | Ja | Controleren |
| ServerSocket (ServerSocket.isSupported) | Ja | Ja | Ja |
| Arcering | Ja | Ja | Beperkt |
| Stage3D (Stage.stage3Ds.length) | Ja | Ja | Ja |
| Oriëntatie werkgebied (Stage.supportsOrientationChange) | Nee | Nee | Ja |
| StageVideo | Nee | Nee | Controleren |
| StageWebView (StageWebView.isSupported) | Ja | Ja | Ja |
| Toepassing opstarten bij aanmelding (NativeApplication.supportsStartAtLogin) | Ja | Ja | Nee |
| StorageVolumeInfo (StorageVolumeInfo.isSupported) | Ja | Ja | Nee |
| Modus systeem inactief | Nee | Nee | Ja |
| SystemTrayIcon (NativeApplication.supportsSystemTrayIcon) | Controleren | Controleren | Nee |
| Invoer framework tekstlay-out | Ja | Ja | Nee |
| Updater (Updater.isSupported) | Ja | Nee | Nee |
| XMLSignatureValidator (XMLSignatureValidator.isSupported) | Ja | Ja | Nee |

De elementen in het tabel hebben de volgende betekenis:

- *Controleren* — De functie wordt ondersteund op sommige, maar niet alle apparaten in het profiel. U moet tijdens uitvoering controleren of de functie wordt ondersteund voordat u deze gebruikt.
- *Beperkt* — De functie wordt ondersteund, maar is aanzienlijk beperkt. Raadpleeg de relevante documentatie voor meer informatie.
- *Nee* — De functie wordt niet ondersteund in het profiel.
- *Ja* — De functie wordt ondersteund in het profiel. Let op: het kan zijn dat afzonderlijke computerapparaten de hardware die nodig is voor een functie, niet bevatten. Niet alle telefoons hebben bijvoorbeeld een camera.

## Profielen opgeven tijdens foutopsporing met ADL

**Adobe AIR 2 of hoger**

ADL controleert of er ondersteunde profielen zijn opgegeven in het element `supportedProfiles` van het descriptorbestand van de toepassing. Als dat het geval is, gebruikt ADL standaard het eerste ondersteunde profiel als profiel tijdens de foutopsporing.

U kunt een profiel voor ADL-foutopsporing opgeven met het opdrachtregelargument `-profile`. Zie "ADL (AIR Debug Launcher)" op pagina 167. U kunt dit argument altijd gebruiken, of u het element `supportedProfiles` nu wel of niet in het descriptorbestand van de toepassing hebt opgegeven. Maar als u wel een element `supportedProfiles` opgeeft, moet dit het profiel bevatten dat u in de opdrachtregel opgeeft, anders genereert ADL een fout.

# Hoofdstuk 16: AIR.SWF API interne browser

## Het bestand badge.swf voor naadloze installatie aanpassen

Naast het bestand badge.swf in de SDK kunt u ook zelf een SWF-bestand voor gebruik in een browserpagina maken. Uw aangepaste SWF-bestand kan op de volgende manieren interageren met de runtime:

- Een AIR-toepassing installeren. Zie "AIR-toepassingen installeren vanuit de browser" op pagina 267.
- Controleren of een bepaalde AIR-toepassing is geïnstalleerd. Zie "Vanuit een webpagina controleren of een AIR-toepassing is geïnstalleerd" op pagina 266.
- Controleren of de runtime is geïnstalleerd. Zie "Controleren of de runtime is geïnstalleerd" op pagina 265.
- Een geïnstalleerde AIR-toepassing op het systeem van de gebruiker starten. Zie "Geïnstalleerde AIR-toepassingen starten vanuit de browser" op pagina 268.

Deze functies worden mogelijk gemaakt door het oproepen van API's in een SWF-bestand op adobe.com: air.swf. U kunt het bestand badge.swf aanpassen, zodat u de air.swf-API's kunt oproepen vanuit uw eigen SWF-bestand.

Een SWF-bestand dat in de browser wordt uitgevoerd, kan ook met een gestarte AIR-toepassing communiceren met behulp van de klasse LocalConnection. Zie Verbinding maken met andere instanties van Flash Player en AIR (voor ActionScript-ontwikkelaars) of Verbinding maken met andere instanties van Flash Player en AIR (voor HTML-ontwikkelaars) voor meer informatie.

***Belangrijk:*** *de functies die in dit deel worden beschreven (en de API's in het bestand air.swf), vereisen dat de eindgebruiker Adobe® Flash® Player 9 update 3 of een latere versie heeft geïnstalleerd in de webbrowser op Windows of Mac OS. Op Linux vereist de naadloze installatiefunctie Flash Player 10 (versie 10,0,12,36 of hoger). U kunt code schrijven om de geïnstalleerde versie van Flash Player te controleren en de gebruiker een alternatieve interface bieden als de vereiste versie van Flash Player niet is geïnstalleerd. Als er bijvoorbeeld een oudere versie van Flash Player is geïnstalleerd, kunt u een koppeling naar de downloadbare versie van het AIR-bestand opnemen (in plaats van het bestand badge.swf of de air.swf-API te gebruiken voor de installatie van de toepassing).*

## AIR-toepassingen installeren met het bestand badge.swf

In de SDK van AIR en de SDK van Flex is het bestand badge.swf opgenomen waarmee u de functie voor naadloze installatie eenvoudig kunt gebruiken. Met badge.swf kunt u de runtime en een AIR-toepassing installeren via een koppeling op een webpagina. Het bestand badge.swf en de broncode worden geleverd voor distributie op uw website.

**Het bestand badge.swf insluiten in een webpagina**

1 Zoek de volgende bestanden in de map samples/badge van de SDK van AIR of de SDK van Flex en voeg deze toe aan uw webserver.

   - badge.swf

- default_badge.html
- AC_RunActiveContent.js

2 Open de pagina default_badge.html in een teksteditor.

3 Pas in de pagina default_badge.html in de JavaScript-functie `AC_FL_RunContent()` de `FlashVars`-parameters als volgt aan:

| Parameter | Beschrijving |
|---|---|
| `appname` | De naam van de toepassing. Deze wordt door het SWF-bestand weergegeven wanneer de runtime niet is geïnstalleerd. |
| `appurl` | (Vereist.) De URL van het AIR-bestand dat moet worden gedownload. U moet een absolute en geen relatieve URL opgeven. |
| `airversion` | (Vereist.) Voor versie 1.0 van de runtime stelt u deze in op 1.0. |
| `imageurl` | De URL van de afbeelding (optioneel) die wordt weergegeven in de badge. |
| `buttoncolor` | De kleur van de downloadknop (opgegeven als een hexadecimale waarde, zoals `FFCC00`). |
| `messagecolor` | De kleur van het tekstbericht dat onder de knop wordt weergegeven als de runtime niet is geïnstalleerd (opgegeven als een hexadecimale waarde, zoals `FFCC00`). |

4 De minimale grootte van het bestand badge.swf file is 217 pixels breed en 180 pixels hoog. Pas de waarden van de parameters `width` en `height` van de functie `AC_FL_RunContent()` aan uw wensen aan.

5 Wijzig de naam van het bestand default_badge.html en pas de code aan uw wensen aan (of neem deze op in een andere HTML-pagina).

***Opmerking:*** *stel het kenmerk `wmode` niet in voor de HTML-tag `embed` waarmee het bestand badge.swf wordt geladen, maar laat de standaardinstelling (`"window"`) ongewijzigd. Met andere `wmode`-instellingen wordt installtie op bepaalde systemen voorkomen. Wanneer u andere `wmode`-instellingen gebruikt, leidt dit tot een fout: "Error #2044: Unhandled ErrorEvent:. text=Error #2074: The stage is too small to fit the download ui."*

U kunt het bestand badge.swf ook bewerken en opnieuw compileren. Zie "Het bestand badge.swf wijzigen" op pagina 263 voor meer informatie.

## AIR-toepassingen installeren via een koppeling voor naadloze installatie op een webpagina

Wanneer u de koppeling voor naadloze installatie aan een pagina hebt toegevoegd, kan de gebruiker de AIR-toepassing installeren door op de koppeling in het SWF-bestand te klikken.

1 Navigeer naar de HTML-pagina in een webbrowser waarin Flash Player (versie 9 update 3 of hoger op Windows en Mac OS, of versie 10 op Linux) is geïnstalleerd.

2 Klik op de webpagina op de koppeling in het bestand badge.swf.

- Als u de runtime hebt geïnstalleerd, gaat u verder met de volgende stap.
- Als u de runtime niet hebt geïnstalleerd, wordt een dialoogvenster geopend met de vraag of u deze wilt installeren. Installeer de runtime (zie "Adobe AIR installeren" op pagina 3) en ga verder met de volgende stap.

3 Laat de standaardinstellingen geselecteerd in het installatievenster en klik op Doorgaan.

Op een Windows-computer voert AIR automatisch de volgende bewerkingen uit:

- De toepassing wordt geïnstalleerd in C:\Program Files\.
- Op het bureaublad wordt een snelkoppeling voor de toepassing gemaakt.

- In het menu Start wordt een snelkoppeling gemaakt.
- De toepassing wordt toegevoegd aan het onderdeel Software in het Configuratiescherm.

In Mac OS wordt de toepassing toegevoegd aan de programmamap (bijvoorbeeld in de map /Applications in Mac OS).

Op een Linux-computer voert AIR automatisch de volgende bewerkingen uit:

- Installeert de toepassing in /opt.
- Op het bureaublad wordt een snelkoppeling voor de toepassing gemaakt.
- In het menu Start wordt een snelkoppeling gemaakt.
- Voegt een item voor de toepassing toe in het systeempakketbeheer

4 Selecteer de gewenste opties en klik op de knop Installeren.

5 Klik op Voltooien wanneer de installatie is voltooid.

## Het bestand badge.swf wijzigen

De Flex SDK en AIR SDK bieden de bronbestanden voor het bestand badge.swf. Deze bestanden zijn te vinden in de map samples/badge van de SDK:

| Bronbestanden | Beschrijving |
|---|---|
| badge.fla | Het Flash -bronbestand om het bestand badge.swf te compileren. Het bestand badge.fla wordt gecompileerd naar een SWF 9-bestand (dat kan worden geladen in Flash Player). |
| AIRBadge.as | Een ActionScript 3.0-klasse waarin de klasse base is gedefinieerd die in het bestand badge.fla wordt gebruikt. |

U kunt de visuele interface van het bestand badge.fla opnieuw ontwerpen met Flash Professional.

Met de constructorfunctie `AIRBadge()`, die in de klasse AIRBadge is gedefinieerd, laadt u het bestand air.swf dat zich bevindt op http://airdownload.adobe.com/air/browserapi/air.swf. Het bestand air.swf bevat code voor het gebruik van de functie voor naadloze installatie.

De methode `onInit()` (in de klasse AIRBadge) wordt opgeroepen nadat het bestand air.swf file is geladen:

```
private function onInit(e:Event):void {
    _air = e.target.content;
    switch (_air.getStatus()) {
        case "installed" :
            root.statusMessage.text = "";
            break;
        case "available" :
            if (_appName && _appName.length > 0) {
                root.statusMessage.htmlText = "<p align='center'><font color='#"
                        + _messageColor + "'>In order to run " + _appName +
                        ", this installer will also set up Adobe® AIR®.</font></p>";
            } else {
                root.statusMessage.htmlText = "<p align='center'><font color='#"
                        + _messageColor + "'>In order to run this application, "
                        + "this installer will also set up Adobe® AIR®.</font></p>";
            }
            break;
        case "unavailable" :
            root.statusMessage.htmlText = "<p align='center'><font color='#"
                        + _messageColor
                        + "'>Adobe® AIR® is not available for your system.</font></p>";
            root.buttonBg_mc.enabled = false;
            break;
    }
}
```

Met de code wordt de globale variabele `_air` ingesteld op de hoofdklasse van het geladen bestand air.swf. In deze klasse zijn de volgende openbare methoden opgenomen, die in het bestand badge.swf worden gebruikt om de functionaliteit voor naadloze installatie op te roepen:

| Methode | Beschrijving |
|---|---|
| `getStatus()` | Hiermee wordt bepaald of de runtime is geïnstalleerd (of kan worden geïnstalleerd) op de computer. Zie "Controleren of de runtime is geïnstalleerd" op pagina 265 voor meer informatie.<br>• `runtimeVersion`: een tekenreeks die de versie van de runtime bevat (bijvoorbeeld `"1.0.M6"`) die is vereist voor de toepassing die wordt geïnstalleerd. |
| `installApplication()` | Hiermee wordt de opgegeven toepassing geïnstalleerd op de computer van de gebruiker. Zie "AIR-toepassingen installeren vanuit de browser" op pagina 267 voor meer informatie.<br>• `url`: een tekenreeks waarmee de URL wordt gedefinieerd. U moet een absoluut en geen relatief URL-pad opgeven.<br>• `runtimeVersion`—Een tekenreeks die de runimteversie aangeeft (zoals `"2.5"`) die wordt vereist door de te installeren toepassing.<br>• `arguments`: argumenten die aan de toepassing worden doorgegeven als deze na de installatie wordt gestart. De toepassing wordt na de installatie gestart als het element `allowBrowserInvocation` op `true` is ingesteld in het descriptorbestand van de toepassing. (Zie "AIR-toepassingsdescriptorbestanden" op pagina 215 voor meer informatie over het descriptorbestand van de toepassing.) Als de toepassing wordt gestart als gevolg van een naadloze installatie vanuit de browser (en de gebruiker de toepassing na de installatie start), verzendt het NativeApplication-object van de toepassing alleen een BrowserInvokeEvent-object als er argumenten zijn doorgegeven. Houd rekening met de gevolgen voor de beveiliging van gegevens die u aan de toepassing doorgeeft. Zie "Geïnstalleerde AIR-toepassingen starten vanuit de browser" op pagina 268 voor meer informatie. |

De instellingen voor `url` en `runtimeVersion` worden aan het SWF-bestand doorgegeven via de FlashVars-instellingen in de HTML-containerpagina.

Als de toepassing na de installatie automatisch wordt gestart, kunt u via communicatie met LocalConnection de geïnstalleerde toepassing bij het oproepen contact laten opnemen met het bestand badge.swf. Zie Verbinding maken met andere instanties van Flash Player en AIR (voor ActionScript-ontwikkelaars) of Verbinding maken met andere instanties van Flash Player en AIR (voor HTML-ontwikkelaars) voor meer informatie.

U kunt ook de methode `getApplicationVersion()` van het bestand air.swf oproepen om te controleren of een toepassing is geïnstalleerd. U kunt deze methode oproepen voordat de toepassing wordt geïnstalleerd of nadat de installatie is gestart. Zie "Vanuit een webpagina controleren of een AIR-toepassing is geïnstalleerd" op pagina 266 voor meer informatie.

# Het bestand air.swf laden

U kunt zelf een SWF-bestand maken waarin u de API's van het bestand air.swf gebruikt voor de interactie met de runtime en AIR-toepassingen vanuit een webpagina in een browser. Het bestand air.swf bevindt zich op http://airdownload.adobe.com/air/browserapi/air.swf. Als u vanuit uw SWF-bestand wilt verwijzen naar de API's van air.swf, laadt u het bestand air.swf in hetzelfde toepassingsdomein als uw SWF-bestand. In het volgende voorbeeld ziet u code voor het laden van het bestand air.swf in het toepassingsdomein van het SWF-bestand:

```
 var airSWF:Object; // This is the reference to the main class of air.swf
var airSWFLoader:Loader = new Loader(); // Used to load the SWF
var loaderContext:LoaderContext = new LoaderContext();
                                  // Used to set the application domain

loaderContext.applicationDomain = ApplicationDomain.currentDomain;

airSWFLoader.contentLoaderInfo.addEventListener(Event.INIT, onInit);
airSWFLoader.load(new URLRequest("http://airdownload.adobe.com/air/browserapi/air.swf"),
                    loaderContext);

function onInit(e:Event):void
{
    airSWF = e.target.content;
}
```

Nadat het bestand air.swf is geladen (wanneer het `contentLoaderInfo`-object van het Loader-object de `init`-gebeurtenis verzendt), kunt u elke API van air.swf oproepen, zoals hieronder wordt beschreven.

***Opmerking:*** *met het bestand badge.swf, dat aanwezig is in de SDK van AIR en de SDK van Flex, wordt het bestand air.swf automatisch geladen. Zie "AIR-toepassingen installeren met het bestand badge.swf" op pagina 261. De instructies in deze sectie zijn van toepassing wanneer u zelf een SWF-bestand maakt waarmee het bestand air.swf wordt geladen.*

# Controleren of de runtime is geïnstalleerd

Een SWF-bestand kan controleren of de runtime is geïnstalleerd door de methode `getStatus()` op te roepen in het bestand air.swf dat is geladen vanaf http://airdownload.adobe.com/air/browserapi/air.swf. Zie "Het bestand air.swf laden" op pagina 265 voor meer informatie.

Nadat het bestand air.swf is geladen, kunt u vanuit het SWF-bestand de methode `getStatus()` van het bestand air.swf als volgt oproepen:

```
 var status:String = airSWF.getStatus();
```

De methode `getStatus()` retourneert een van de volgende tekenreekswaarden, afhankelijk van de status van de runtime op de computer:

| Tekenreekswaarde | Beschrijving |
|---|---|
| `"available"` | De runtime kan op deze computer worden geïnstalleerd, maar is momenteel niet geïnstalleerd. |
| `"unavailable"` | De runtime kan niet op deze computer worden geïnstalleerd. |
| `"installed"` | De runtime is op deze computer geïnstalleerd. |

De methode `getStatus()` meldt een fout als de vereiste versie van Flash Player (versie 9 upgrade 3 op Windows en Mac OS, of versie 10 op Linux) niet in de browser is geïnstalleerd.

# Vanuit een webpagina controleren of een AIR-toepassing is geïnstalleerd

Een SWF-bestand kan controleren of een AIR-toepassing (met een overeenkomstige toepassings-id en uitgevers-id) is geïnstalleerd door de methode `getApplicationVersion()` op te roepen in het bestand air.swf dat is geladen vanaf http://airdownload.adobe.com/air/browserapi/air.swf. Zie "Het bestand air.swf laden" op pagina 265 voor meer informatie.

Nadat het bestand air.swf is geladen, kunt u vanuit het SWF-bestand de methode `getApplicationVersion()` van het bestand air.swf als volgt oproepen:

```
 var appID:String = "com.example.air.myTestApplication";
var pubID:String = "02D88EEED35F84C264A183921344EEA353A629FD.1";
airSWF.getApplicationVersion(appID, pubID, versionDetectCallback);

function versionDetectCallback(version:String):void
{
    if (version == null)
    {
        trace("Not installed.");
        // Take appropriate actions. For instance, present the user with
        // an option to install the application.
    }
    else
    {
        trace("Version", version, "installed.");
        // Take appropriate actions. For instance, enable the
        // user interface to launch the application.
    }
}
```

De methode `getApplicationVersion()` heeft de volgende parameters:

| Parameters | Beschrijving |
|---|---|
| `appID` | De toepassings-id voor de toepassing. Zie voor details "id" op pagina 237. |
| `pubID` | De uitgevers-id voor de toepassing. Zie "publisherID" op pagina 247 voor meer informatie. Als de desbetreffende toepassing geen uitgevers-id heeft, stelt u de parameter `pubID` in als een lege tekenreeks (""). |
| `callback` | Een callbackfunctie die optreedt als de handlerfunctie. De methode getApplicationVersion() werkt asynchroon en wanneer de geïnstalleerde versie is aangetroffen (of juist ontbreekt), wordt deze callbackmethode opgeroepen. In de definitie van de callbackmethode moet één parameter zijn opgenomen: een tekenreeks die de versie van de geïnstalleerde toepassing bevat. Als de toepassing niet is geïnstalleerd, wordt de waarde null doorgegeven aan de functie, zoals in het vorige codevoorbeeld. |

De methode `getApplicationVersion()` meldt een fout als de vereiste versie van Flash Player (versie 9 upgrade 3 op Windows en Mac OS, of versie 10 op Linux) niet in de browser is geïnstalleerd.

***Opmerking:*** *vanaf AIR 1.5.3 is de uitgevers-id afgekeurd. Uitgevers-id's worden niet meer automatisch toegewezen aan een toepassing. Voor compatibiliteit met oudere versies kunnen toepassingen alsnog een uitgevers-id opgeven.*

# AIR-toepassingen installeren vanuit de browser

Een SWF-bestand kan een AIR-toepassing installeren door de methode `installApplication()` op te roepen in het bestand air.swf dat is geladen vanaf `http://airdownload.adobe.com/air/browserapi/air.swf`. Zie "Het bestand air.swf laden" op pagina 265 voor meer informatie.

Nadat het bestand air.swf is geladen, kunt u vanuit het SWF-bestand de methode `installApplication()` van het bestand air.swf als volgt oproepen:

```
var url:String = "http://www.example.com/myApplication.air";
var runtimeVersion:String = "1.0";
var arguments:Array = ["launchFromBrowser"]; // Optional
airSWF.installApplication(url, runtimeVersion, arguments);
```

Met de methode `installApplication()` wordt de opgegeven toepassing geïnstalleerd op de computer van de gebruiker. Deze methode heeft de volgende parameters:

| Parameter | Beschrijving |
|---|---|
| `url` | Een tekenreeks die de URL bevat van het AIR-bestand dat wordt geïnstalleerd. U moet een absoluut en geen relatief URL-pad opgeven. |
| `runtimeVersion` | Een tekenreeks die de versie van de runtime bevat (bijvoorbeeld "1.0") die is vereist voor de toepassing die wordt geïnstalleerd. |
| `arguments` | Een array met argumenten die aan de toepassing worden doorgegeven als deze na de installatie wordt gestart. Alle alfanumerieke tekens worden in de argumenten herkend. Als u andere waarden moet doorgeven, kunt u het gebruik van een coderingsschema overwegen.<br><br>De toepassing wordt na de installatie gestart als het element `allowBrowserInvocation` op `true` is ingesteld in het descriptorbestand van de toepassing. (Zie "AIR-toepassingsdescriptorbestanden" op pagina 215 voor meer informatie over het descriptorbestand van de toepassing.) Als de toepassing wordt gestart als gevolg van een naadloze installatie vanuit de browser (en de gebruiker de toepassing na de installatie start), verzendt het NativeApplication-object van de toepassing alleen een BrowserInvokeEvent-object als er argumenten zijn doorgegeven. Zie "Geïnstalleerde AIR-toepassingen starten vanuit de browser" op pagina 268 voor meer informatie. |

De methode `installApplication()` werkt alleen wanneer deze wordt opgeroepen in de gebeurtenishandler voor een gebruikersgebeurtenis, zoals een muisklik.

De methode `installApplication()` meldt een fout als de vereiste versie van Flash Player (versie 9 upgrade 3 op Windows en Mac OS, of versie 10 op Linux) niet in de browser is geïnstalleerd.

Als een gebruiker in Mac OS een recentere versie van een toepassing wil installeren, moet de gebruiker over voldoende toegangsrechten beschikken om te installeren in de toepassingsmap (en over beheerdersrechten als de toepassing de runtime bijwerkt). In Windows moet de gebruiker over beheerdersrechten beschikken.

U kunt ook de methode `getApplicationVersion()` van het bestand air.swf oproepen om te controleren of een toepassing al is geïnstalleerd. U kunt deze methode oproepen voordat het installatieproces van de toepassing begint of nadat de installatie is gestart. Zie "Vanuit een webpagina controleren of een AIR-toepassing is geïnstalleerd" op pagina 266 voor meer informatie. Wanneer de toepassing is gestart, kan deze met de SWF-inhoud in de browser communiceren via de klasse LocalConnection. Zie Verbinding maken met andere instanties van Flash Player en AIR (voor ActionScript-ontwikkelaars) of Communicating with other Flash Player and AIR instances (voor HTML-ontwikkelaars) voor meer informatie.

# Geïnstalleerde AIR-toepassingen starten vanuit de browser

Als u de browseroproepfunctie wilt gebruiken (zodat een toepassing kan worden gestart vanuit de browser), moet in het descriptorbestand van de doeltoepassing de volgende instelling zijn opgenomen:

```
<allowBrowserInvocation>true</allowBrowserInvocation>
```

(Zie "AIR-toepassingsdescriptorbestanden" op pagina 215 voor meer informatie over het descriptorbestand van de toepassing.)

Een SWF-bestand in de browser kan een AIR-toepassing starten door de methode `launchApplication()` op te roepen in het bestand air.swf dat is geladen vanaf http://airdownload.adobe.com/air/browserapi/air.swf. Zie "Het bestand air.swf laden" op pagina 265 voor meer informatie.

Nadat het bestand air.swf is geladen, kunt u vanuit het SWF-bestand de methode `launchApplication()` van het bestand air.swf als volgt oproepen:

```
var appID:String = "com.example.air.myTestApplication";
var pubID:String = "02D88EEED35F84C264A183921344EEA353A629FD.1";
var arguments:Array = ["launchFromBrowser"]; // Optional
airSWF.launchApplication(appID, pubID, arguments);
```

De methode `launchApplication()` is gedefinieerd op het hoogste niveau van het bestand air.swf (dat is geladen in het toepassingsdomein van het SWF-bestand van de gebruikersinterface). Als u deze methode oproept, wordt de opgegeven toepassing gestart (als deze is geïnstalleerd en oproepen via de browser is toegestaan via de instelling van `allowBrowserInvocation` in het descriptorbestand van de toepassing). De methode heeft de volgende parameters:

| Parameter | Beschrijving |
|---|---|
| `appID` | De toepassings-id voor de toepassing die moet worden gestart. Zie voor details "id" op pagina 237. |
| `pubID` | De uitgevers-id voor de toepassing die moet worden gestart. Zie voor details "publisherID" op pagina 247. Als de desbetreffende toepassing geen uitgevers-id heeft, stelt u de parameter `pubID` in als een lege tekenreeks ("") |
| `arguments` | Een array met argumenten die worden doorgegeven aan de toepassing. Het NativeApplication-object van de toepassing verzendt een BrowserInvokeEvent-gebeurtenis waarvoor de eigenschap arguments is ingesteld op deze array. Alle alfanumerieke tekens worden in de argumenten herkend. Als u andere waarden moet doorgeven, kunt u het gebruik van een coderingsschema overwegen. |

De methode `launchApplication()` werkt alleen wanneer deze wordt opgeroepen in de gebeurtenishandler voor een gebruikersgebeurtenis, zoals een muisklik.

De methode `launchApplication()` meldt een fout als de vereiste versie van Flash Player (versie 9 upgrade 3 op Windows en Mac OS, of versie 10 op Linux) niet in de browser is geïnstalleerd.

Als het element `allowBrowserInvocation` op `false` is ingesteld in het descriptorbestand van de toepassing, heeft het oproepen van de methode `launchApplication()` geen effect.

Voordat u de gebruikersinterface weergeeft om de toepassing te starten, kunt u het best de methode `getApplicationVersion()` in het bestand air.swf oproepen. Zie "Vanuit een webpagina controleren of een AIR-toepassing is geïnstalleerd" op pagina 266 voor meer informatie.

Wanneer de toepassing wordt opgeroepen via de browseroproepfunctie, verzendt het NativeApplication-object van de toepassing een BrowserInvokeEvent-object. Zie AIR-toepassingen aanroepen vanuit de browser (voor ActionScript-ontwikkelaars) of Invoking an AIR application from the browser (voor HTML-ontwikkelaars) voor meer informatie.

Als u de browseroproepfunctie gebruikt, moet u rekening houden met de gevolgen voor de beveiliging. Deze gevolgen worden beschreven in AIR-toepassingen aanroepen vanuit de browser (voor ActionScript-ontwikkelaars) en Invoking an AIR application from the browser (voor HTML-ontwikkelaars).

Wanneer de toepassing is gestart, kan deze met de SWF-inhoud in de browser communiceren via de klasse LocalConnection. Zie Verbinding maken met andere instanties van Flash Player en AIR (voor ActionScript-ontwikkelaars) of Communicating with other Flash Player and AIR instances (voor HTML-ontwikkelaars) voor meer informatie.

***Opmerking:*** *vanaf AIR 1.5.3 is de uitgevers-id afgekeurd. Uitgevers-id's worden niet meer automatisch toegewezen aan een toepassing. Voor compatibiliteit met oudere versies kunnen toepassingen alsnog een uitgevers-id opgeven.*

# Hoofdstuk 17: AIR-toepassingen bijwerken

Gebruikers kunnen een AIR-toepassing installeren of bijwerken door te dubbelklikken op een AIR-bestand op hun computer of vanuit de browser (met behulp van de functie voor naadloze installatie). Het installatieprogramma van Adobe® AIR® beheert de installatie en waarschuwt de gebruiker als deze een bestaande toepassing gaat bijwerken.

Met behulp van de klasse Updater kunt u er echter ook voor zorgen dat een geïnstalleerde toepassing zichzelf bijwerkt naar een nieuwe versie. (Een geïnstalleerde toepassing kan mogelijk detecteren dat er een nieuwe versie beschikbaar is die kan worden gedownload en geïnstalleerd.) De klasse Updater bevat de methode `update()`, waarmee u naar een AIR-bestand op de computer van de gebruiker kunt verwijzen en een update naar die versie kunt uitvoeren. Uw toepassing moet zijn verpakt als AIR-bestand voordat u de klasse Updater kunt gebruiken. Toepassingen die zijn verpakt als native uitvoerbaar bestand of als pakket, moeten de updatefaciliteiten gebruiken die door het native platform worden geleverd.

Zowel de toepassings- als de uitgevers-id van een AIR-updatebestand moeten overeenkomen met die van de toepassing die moet worden bijgewerkt. De uitgevers-id is afgeleid van het handtekeningcertificaat. Zowel de update als de toepassing die moet worden bijgewerkt moeten met hetzelfde certificaat zijn ondertekend.

Voor AIR 1.5.3 of hoger bevat het descriptorbestand van de toepassing een element `<publisherID>`. U moet dit element gebruiken als er versies van de toepassing bestaan die zijn ontwikkeld met AIR 1.5.2 of ouder. Zie voor meer informatie “publisherID” op pagina 247.

Vanaf AIR 1.1 en hoger kunt u overstappen op het gebruik van een nieuw digitaal ondertekend certificaat voor een toepassing. Dit wordt certificaatmigratie genoemd. Als u wilt overstappen op het gebruik van een nieuwe handtekening voor een toepassing, moet het AIR-updatebestand worden ondertekend met zowel het nieuwe als het oorspronkelijke certificaat. Certificaatmigratie is een eenrichtingsproces. Na de migratie worden alleen AIR-bestanden die zijn ondertekend met het nieuwe certificaat (of met beide certificaten), herkend als updates voor een bestaande installatie.

Het beheer van updates van toepassingen kan vrij gecompliceerd zijn. AIR 1.5 is voorzien van het nieuwe *updateframework voor AdobeAIR-toepassingen*. Dit framework biedt API's waarmee ontwikkelaars goede updatemogelijkheden kunnen inbouwen in AIR-toepassingen.

U kunt certificaatmigratie gebruiken om over te stappen van een zelfondertekend certificaat op een commercieel digitaal ondertekend certificaat of om over te stappen van het ene zelfondertekende of commerciële certificaat op een ander. Als u niet voor migratie kiest, moeten bestaande gebruikers de huidige versie van uw toepassing van hun computer verwijderen voordat ze de nieuwe versie kunnen installeren. Zie “Certificaten wijzigen” op pagina 204 voor meer informatie.

Het is verstandig om altijd een updatemechanisme in de toepassing op te nemen. Als u een nieuwe versie van de toepassing maakt, zorgt het updatemechanisme ervoor dat de gebruiker wordt gevraagd om de nieuwe versie te installeren.

Tijdens het installeren, bijwerken of verwijderen van AIR worden logbestanden gemaakt. U kunt deze logboeken raadplegen om de oorzaak van eventuele installatieproblemen te achterhalen. Zie Installatielogboeken.

***Opmerking:*** *nieuwe versies van de runtime voor Adobe AIR kunnen bijgewerkte versies van WebKit bevatten. Een bijgewerkte versie van WebKit kan mogelijk leiden tot onverwachte wijzigingen in HTML-inhoud in een geïmplementeerde AIR-toepassing. Het kan zijn dat u door deze wijzigingen gedwongen wordt de toepassing bij te werken. Een updatemechanisme kan de gebruiker op de hoogte stellen van de nieuwe versie van de toepassing. Zie Informatie over de HTML-omgeving (voor ActionScript-ontwikkelaars) of About the HTML environment (voor HTML-ontwikkelaars) voor meer informatie.*

# Informatie over het bijwerken van toepassingen

De Updater-klasse (in het flash.desktop-pakket) bevat één methode, `update()`, waarmee u de huidige versie van de toepassing die wordt uitgevoerd, kunt bijwerken naar een andere versie. Als bijvoorbeeld een versie van het AIR-bestand (Sample_App_v2.air) op het bureaublad van de gebruiker staat, wordt de toepassing met behulp van de volgende code bijgewerkt.

ActionScript-voorbeeld:

```
var updater:Updater = new Updater();
var airFile:File = File.desktopDirectory.resolvePath("Sample_App_v2.air");
var version:String = "2.01";
updater.update(airFile, version);
```

JavaScript-voorbeeld:

```
var updater = new air.Updater();
var airFile = air.File.desktopDirectory.resolvePath("Sample_App_v2.air");
var version = "2.01";
updater.update(airFile, version);
```

Voordat de klasse Updater door een toepassing wordt gebruikt, moet de gebruiker of toepassing de bijgewerkte versie van het AIR-bestand downloaden naar de computer. Zie "AIR-bestanden naar de computer van de gebruiker downloaden" op pagina 273 voor meer informatie.

## Resultaten van het oproepen van de methode Updater.update()

Wanneer een toepassing in de runtime de methode `update()` oproept, sluit de runtime de toepassing af en probeert de runtime de nieuwe versie van de toepassing vanuit het AIR-bestand te installeren. De runtime controleert of de toepassings- en uitgevers-id in het AIR-bestand overeenkomen met de toepassings- en uitgevers-id voor de toepassing die de methode `update()` oproept. (Zie "AIR-toepassingsdescriptorbestanden" op pagina 215 voor informatie over de toepassings- en uitgevers-id.) De runtime controleert ook of de tekenreeks version overeenkomt met de tekenreeks `version` die aan de methode `update()` is doorgegeven. Als de installatie zonder problemen wordt voltooid, opent de runtime de nieuwe versie van de toepassing. Als de installatie niet kan worden voltooid, wordt opnieuw de bestaande (eerder geïnstalleerde) versie van de toepassing geopend.

Als een gebruiker in Mac OS een recentere versie van een toepassing wil installeren, moet de gebruiker over voldoende toegangsrechten beschikken om te installeren in de toepassingsmap. In Windows en Linux moet de gebruiker over beheerdersrechten beschikken.

Als de bijgewerkte versie van de toepassing een bijgewerkte versie van de runtime vereist, wordt de nieuwe runtimeversie geïnstalleerd. Voor het bijwerken van de runtime moet de gebruiker over beheerdersrechten voor de computer beschikken.

Wanneer een toepassing wordt getest met ADL, resulteert het oproepen van de methode `update()` in een runtime-uitzondering.

## Informatie over de tekenreeks version

De tekenreeks die is opgegeven voor de parameter `version` van de methode `update()`, moet overeenkomen met de tekenreeks in het element `version` of `versionNumber` van het toepassingsdescriptorbestand voor het AIR-bestand dat moet worden geïnstalleerd. Uit veiligheidsoverwegingen is het verplicht de parameter `version` op te geven. Doordat u eist dat het versienummer in het AIR-bestand wordt geverifieerd door de toepassing, wordt er niet per ongeluk een oudere versie geïnstalleerd. (Een oudere versie van de toepassing bevat misschien een zwakte in de beveiliging die is gecorrigeerd in de huidige geïnstalleerde toepassing.) De toepassing moet ook de tekenreeks version in het AIR-bestand vergelijken met de tekenreeks version in de geïnstalleerde toepassing om te voorkomen dat de computer van de gebruiker wordt aangevallen en er ten onrechte een downgrade wordt uitgevoerd.

In oudere versies van AIR dan 2.5 kan de versiereeks elke indeling hebben. De tekenreeks kan bijvoorbeeld bestaan uit cijfers, zoals '2.01', maar ook uit cijfers en letters, zoals 'versie 2'. In AIR 2.5 of hogere versie moet de versiereeks een sequentie zijn van hoogstens drie getallen van drie cijfers, gescheiden door putnen. ".0", "1.0" en "67.89.999" zijn bijvoorbeeld alle drie geldige versienummers. U moet de updateversiereeks verifiëren voordat u de toepassing bijwerkt.

Als een Adobe AIR-toepassing een AIR-bestand via het web downloadt, is het een goede gewoonte om een mechanisme te gebruiken waarmee de webservice de Adobe AIR-toepassing kan informeren over de versie die wordt gedownload. De toepassing kan deze tekenreeks vervolgens gebruiken voor de parameter `version` van de methode `update()`. Als het AIR-bestand op een andere manier wordt verkregen, waarbij de versie van het AIR-bestand onbekend blijft, kan de AIR-toepassing het AIR-bestand onderzoeken om de versiegegevens vast te stellen. (Een AIR-bestand is een met ZIP gecomprimeerd archief en het descriptorbestand van de toepassing is de tweede record in het archief.)

Zie “AIR-toepassingsdescriptorbestanden” op pagina 215 voor meer informatie over het descriptorbestand van een toepassing.

## Workflow voor toepassingsupdates ondertekenen

Wanneer u de updates ad hoc publiceert, wordt het beheren van verschillende versies van een toepassing ingewikkelder en wordt het volgen van verloopdatums van certificaten lastiger. Certificaten kunnen verlopen voordat u een update publiceert.

Adobe AIR-runtime behandelt een toepassingsupdate die wordt gepubliceerd zonder migratie-ondertekening, als een nieuwe toepassing. Gebruikers moeten hun huidige AIR-toepassing verwijderen voordat zij de toepassingsupdate kunnen installeren.

Als u dit probleem wilt oplossen, uploadt u elke bijgewerkte toepassing met het nieuwste certificaat naar een afzonderlijke implementatie-URL. Neem een herinneringsmechanisme op voor het toepassen van migratie-ondertekeningen wanneer uw certificaat zich in de respijtperiode van 180 dagen bevindt. Zie “Een bijgewerkte versie van een AIR-toepassing ondertekenen.” op pagina 209 voor meer informatie.

Zie “ATD-opdrachten” op pagina 173 voor informatie over het toepassen van ondertekeningen.

Voer de volgende taken uit om het toepassen van de migratie-ondertekeningen te stroomlijnen:

- Upload elke bijgewerkte toepassing naar een afzonderlijke implementatie-URL.
- Upload de XML-bestand van de upgradedescriptor en het nieuwste certificaat voor de update naar dezelfde URL.
- Onderteken de bijgewerkte toepassing met het nieuwste certificaat.
- Pas een migratie-ondertekening toe op de bijgewerkte toepassing met het certificaat dat is gebruikt om de vorige versie, die zich op een andere URL bevindt, te ondertekenen.

# Aangepaste gebruikersinterfaces voor toepassingsupdates presenteren

AIR bevat een standaardinterface voor updates:


Deze interface wordt altijd gebruikt wanneer een gebruiker voor het eerst een versie van een toepassing op een computer installeert. U kunt echter uw eigen interface definiëren die moet worden gebruikt voor daarop volgende instanties. Als in uw toepassing een aangepaste interface voor updates wordt gedefinieerd, specificeert u een `customUpdateUI`-element in het descriptorbestand voor de nu geïnstalleerde toepassing:

```
<customUpdateUI>true</customUpdateUI>
```

Wanneer de toepassing is geïnstalleerd en de gebruiker een AIR-bestand opent met een toepassings- en een uitgevers-id die overeenkomen met die van de geïnstalleerde toepassing, opent de runtime de toepassing in plaats van het standaard installatieprogramma van AIR. Zie voor meer informatie “customUpdateUI” op pagina 227.

Wanneer de toepassing wordt uitgevoerd (wanneer het object `NativeApplication.nativeApplication` de gebeurtenis `load` verzendt), kan de toepassing beslissen of de toepassing moet worden bijgewerkt (met behulp van de klasse Updater). Als de toepassing besluit een update uit te voeren, kan deze een eigen installatie-interface presenteren aan de gebruiker (die verschilt van de standaardinterface die wordt weergegeven als de toepassing wordt uitgevoerd).

# AIR-bestanden naar de computer van de gebruiker downloaden

Met de klasse Updater moet de gebruiker of de toepassing eerst een AIR-bestand lokaal op de computer van de gebruiker opslaan.

***Opmerking:*** *AIR 1.5 is voorzien van een updateframework waarmee ontwikkelaars goede updatemogelijkheden kunnen inbouwen in AIR-toepassingen. In veel gevallen is het handiger om gebruik te maken van dit framework dan om de methode* `update()` *van de klasse Update rechtstreeks te gebruiken. Raadpleeg “Het updateframework gebruiken” op pagina 277 voor meer informatie.*

Met de volgende code wordt bijvoorbeeld een AIR-bestand gelezen van een URL (http://example.com/air/updates/Sample_App_v2.air) en wordt het AIR-bestand in de opslagmap van de toepassing opgeslagen.

ActionScript-voorbeeld:

```
var urlString:String = "http://example.com/air/updates/Sample_App_v2.air";
var urlReq:URLRequest = new URLRequest(urlString);
var urlStream:URLStream = new URLStream();
var fileData:ByteArray = new ByteArray();
urlStream.addEventListener(Event.COMPLETE, loaded);
urlStream.load(urlReq);

function loaded(event:Event):void {
    urlStream.readBytes(fileData, 0, urlStream.bytesAvailable);
    writeAirFile();
}

function writeAirFile():void {
    var file:File = File.applicationStorageDirectory.resolvePath("My App v2.air");
    var fileStream:FileStream = new FileStream();
    fileStream.open(file, FileMode.WRITE);
    fileStream.writeBytes(fileData, 0, fileData.length);
    fileStream.close();
    trace("The AIR file is written.");
}
```

JavaScript-voorbeeld:

```
var urlString = "http://example.com/air/updates/Sample_App_v2.air";
var urlReq = new air.URLRequest(urlString);
var urlStream = new air.URLStream();
var fileData = new air.ByteArray();
urlStream.addEventListener(air.Event.COMPLETE, loaded);
urlStream.load(urlReq);

function loaded(event) {
    urlStream.readBytes(fileData, 0, urlStream.bytesAvailable);
    writeAirFile();
}

function writeAirFile() {
    var file = air.File.desktopDirectory.resolvePath("My App v2.air");
    var fileStream = new air.FileStream();
    fileStream.open(file, air.FileMode.WRITE);
    fileStream.writeBytes(fileData, 0, fileData.length);
    fileStream.close();
    trace("The AIR file is written.");
}
```

Raadpleeg de volgende bronnen voor meer informatie:

- Workflow voor het lezen van en schrijven naar bestanden (voor ActionScript-ontwikkelaars)
- Workflow for reading and writing files (voor HTML-ontwikkelaars)

# Controleren of een toepassing voor de eerste keer wordt uitgevoerd

Wanneer u eenmaal een toepassing hebt bijgewerkt, kunt u desgewenst een informatief bericht of een welkomstbericht voor de gebruiker weergeven. Wanneer de toepassing wordt gestart, controleert de toepassing of deze voor het eerst wordt uitgevoerd, om te bepalen of het bericht moet worden weergegeven.

***Opmerking:*** *AIR 1.5 is voorzien van een updateframework waarmee ontwikkelaars goede updatemogelijkheden kunnen inbouwen in AIR-toepassingen. Dit framework biedt gemakkelijke methoden om te controleren of een versie van een toepassing voor de eerste keer wordt uitgevoerd. Raadpleeg "Het updateframework gebruiken" op pagina 277 voor meer informatie.*

Deze controle kan onder andere worden uitgevoerd als u een bestand in de opslagmap van de toepassing opslaat bij het initialiseren van de toepassing. Telkens wanneer de toepassing wordt opgestart, controleert de toepassing of dat bestand bestaat. Als het bestand niet bestaat, wordt de toepassing voor de eerste keer uitgevoerd voor de huidige gebruiker. Als het bestand bestaat, is de toepassing al ten minste één keer uitgevoerd. Als het bestand bestaat maar een versienummer bevat dat ouder is dan het huidige versienummer, weet u dat de gebruiker de nieuwe versie voor de eerste keer uitvoert.

In het volgende Flex-voorbeeld ziet u het concept:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:WindowedApplication xmlns:mx="http://www.adobe.com/2006/mxml"
    layout="vertical"
    title="Sample Version Checker Application"
    applicationComplete="system extension()">
    <mx:Script>
        <![CDATA[
            import flash.filesystem.*;
            public var file:File;
            public var currentVersion:String = "1.2";
            public function system extension():void {
                file = File.applicationStorageDirectory;
                file = file.resolvePath("Preferences/version.txt");
                trace(file.nativePath);
                if(file.exists) {
                    checkVersion();
                } else {
                    firstRun();
                }
            }
            private function checkVersion():void {
                var stream:FileStream = new FileStream();
                stream.open(file, FileMode.READ);
                var reversion:String = stream.readUTFBytes(stream.bytesAvailable);
                stream.close();
                if (reversion != currentVersion) {
                    log.text = "You have updated to version " + currentVersion + ".\n";
```

```
                } else {
                    saveFile();
                }
                log.text += "Welcome to the application.";
            }
            private function firstRun():void {
                log.text = "Thank you for installing the application. \n"
                    + "This is the first time you have run it.";
                saveFile();
            }
            private function saveFile():void {
                var stream:FileStream = new FileStream();
                stream.open(file, FileMode.WRITE);
                stream.writeUTFBytes(currentVersion);
                stream.close();
            }
        ]]>
    </mx:Script>
    <mx:TextArea ID="log" width="100%" height="100%" />
</mx:WindowedApplication>
```

In het volgende voorbeeld wordt het concept gedemonstreerd in JavaScript:

```
<html>
    <head>
        <script src="AIRAliases.js" />
        <script>
            var file;
            var currentVersion = "1.2";
            function system extension() {
                file = air.File.appStorageDirectory.resolvePath("Preferences/version.txt");
                air.trace(file.nativePath);
                if(file.exists) {
                    checkVersion();
                } else {
                    firstRun();
                }
            }
            function checkVersion() {
                var stream = new air.FileStream();
                stream.open(file, air.FileMode.READ);
                var reversion = stream.readUTFBytes(stream.bytesAvailable);
                stream.close();
                if (reversion != currentVersion) {
                    window.document.getElementById("log").innerHTML
                            = "You have updated to version " + currentVersion + ".\n";
                } else {
                    saveFile();
                }
                window.document.getElementById("log").innerHTML
                                += "Welcome to the application.";
```

```
            }
            function firstRun() {
                window.document.getElementById("log").innerHTML
                            = "Thank you for installing the application. \n"
                            + "This is the first time you have run it.";
                saveFile();
            }
            function saveFile() {
                var stream = new air.FileStream();
                stream.open(file, air.FileMode.WRITE);
                stream.writeUTFBytes(currentVersion);
                stream.close();
            }
        </script>
    </head>
    <body onLoad="system extension()">
        <textarea ID="log" rows="100%" cols="100%" />
    </body>
</html>
```

Als uw toepassing gegevens lokaal opslaat (bijvoorbeeld in de opslagmap van de toepassing), wordt u aangeraden te controleren of er eerder opgeslagen gegevens (van vorige versies) aanwezig zijn wanneer de toepassing voor het eerst wordt uitgevoerd.

# Het updateframework gebruiken

Het kan lastig zijn om updates voor toepassingen te beheren. Het *updateframework voor AdobeAIR-toepassingen* levert API's waarmee ontwikkelaars krachtige updatemogelijkheden hebben voor AIR-toepassingen. Het AIR-updateframework voert de volgende taken voor ontwikkelaars uit:

- Controleer van tijd tot tijd op updates op basis van intervallen of wanneer de gebruiker erom vraagt
- Download AIR-bestanden (updates) van een bron op internet
- De gebruiker op de hoogte stellen bij de eerste keer dat de nieuw geïnstalleerde versie wordt uitgevoerd
- Bevestig dat de gebruiker op updates wil controleren
- Informatie over de nieuwe updateversie aan de gebruiker weergeven
- Downloadvoortgang en foutgegevens aan de gebruiker weergeven

Het AIR-updateframework biedt een testgebruikersinterface voor uw toepassing. Deze biedt de gebruiker basisinformatie en configuratie-opties voor updates van de toepassing. Uw toepassing kan ook een aangepaste klantinterface definiëren voor gebruik met het updateframework.

Met het AIR-updateframework kunt u informatie over de updateversie van een AIR-toepassing opslaan in eenvoudige XML-configuratiebestanden. Voor de meeste toepassingen is het instellen van deze configuratiebestanden met basiscode een goede updatefunctionaliteit voor de eindgebruiker.

Zelfs als u niet gebruikmaakt van het updateframework, omvat Adobe AIR een klasse Updater die AIR-toepassingen kunnen gebruiken om te upgraden naar een nieuwe versie. Met de klasse Updater kan een toepassing een upgrade uitvoeren naar een versie die aanwezig is in een AIR-bestand op de computer van de gebruiker. Bij upgrademanagement komt echter vaak meer bij kijken dan alleen de update in een lokaal opgeslagen AIR-bestand.

## Bestanden van het updateframework van AIR

Het updateframework van AIR staat in de directory frameworks/libs/air directory van de SDK van AIR 2. Deze bevat de volgende bestanden:

- applicationupdater.swc: definieert de basisfunctionaliteit van de updatebibliotheek voor gebruik in ActionScript. Deze versie bevat geen gebruikersinterface.
- applicationupdater.swf: definieert de basisfunctionaliteit van de updatebibliotheek voor gebruik in JavaScript. Deze versie bevat geen gebruikersinterface.
- applicationupdater_ui.swc: definieert een Flex 4-versie van de basisfunctionaliteit van de updatebibliotheek, inclusief een gebruikersinterface die uw toepassing kan gebruiken om de updateopties weer te geven.
- applicationupdater_ui.swc: definieert een JavaScript-versie van de basisfunctionaliteit van de updatebibliotheek, inclusief een gebruikersinterface die uw toepassing kan gebruiken om de updateopties weer te geven.

Raadpleeg de volgende secties voor meer informatie:

- "De Flex-ontwikkelomgeving instellen" op pagina 278
- "Frameworkbestanden opnemen in een HTML-gebaseerde AIR-toepassing" op pagina 278
- "Eenvoudig voorbeelden: De ApplicationUpdaterUI-versie gebruiken" op pagina 279

## De Flex-ontwikkelomgeving instellen

De SWC-bestanden in de directory frameworks/libs/air van de SDK van AIR 2 definiëren klassen die u kunt gebruiken bij het ontwikkelen van toepassingen in Flex en Flash.

Als u het updateframework wilt gebruiken bij het compileren met de Flex SDK, neemt u het bestand ApplicationUpdater.swc of ApplicationUpdater_UI.swc op in de aanroep naar de amxmlc-compiler. In het volgende voorbeeld laadt de compiler het bestand ApplicationUpdater.swc in de submap lib van de Flex SDK-map:

```
amxmlc -library-path+=lib/ApplicationUpdater.swc  -- myApp.mxml
```

In het volgende voorbeeld laadt de compiler het bestand ApplicationUpdater_UI.swc in de submap lib van de Flex SDK-map:

```
amxmlc -library-path+=lib/ApplicationUpdater_UI.swc  -- myApp.mxml
```

Als u toepassingen ontwikkelt met Flash Builder, voegt u het SWC-bestand toe op het tabblad Library Path van de Flex Build Path-instellingen in het dialoogvenster Properties.

Let op dat u de SWC-bestanden kopieert naar de map waarnaar u verwijst in de amxmlc-compiler (wanneer u gebruikmaakt van de SDK van Flex) of Flash Builder.

## Frameworkbestanden opnemen in een HTML-gebaseerde AIR-toepassing

De map frameworks/html van het updateframework bevat de volgende SWF-bestanden:

- applicationupdater.swf—Definieert de basisfunctionaliteit van de updatebibliotheek, zonder gebruikersinterface
- ApplicationUpdater_UI.swf—Definieert de basisfunctionaliteit van de updatebibliotheek en omvat een gebruikersinterface die uw toepassing kan gebruiken om de update-opties weer te geven

JavaScript-code in AIR-toepassingen kan gebruikmaken van klassen die zijn gedefinieerd in SWF-bestanden.

Als u het updateframework wilt gebruiken, neemt u het bestand applicationupdater.swf of applicationupdater_ui.swf op in uw toepassingsmap (of een submap). Voeg vervolgens in het HTML-bestand dat het framework zal gebruiken (in JavaScript-code) een `script`-tag op waardoor het bestand wordt geladen:

```
<script src="applicationUpdater.swf" type="application/x-shockwave-flash"/>
```

Of gebruik deze `script`-tag om het bestand applicationupdater_ui.swf te laden:

```
<script src="applicationupdater_ui.swf" type="application/x-shockwave-flash"/>
```

De API die wordt gedefinieerd in deze twee bestanden, wordt beschreven in het resterende deel van dit document.

## Eenvoudig voorbeelden: De ApplicationUpdaterUI-versie gebruiken

De ApplicationUpdaterUI-versie van het updateframework biedt een basisinterface die u gemakkelijk kunt gebruiken in uw toepassing. Hier volgt een eenvoudig voorbeeld.

Maak eerst een AIR-toepassing die het updateframework aanroept:

1 Als het een AIR-toepassing is op basis van HTML, laadt u het bestand applicationupdaterui.js:

```
<script src="ApplicationUpdater_UI.swf" type="application/x-shockwave-flash"/>
```

2 Instantieer in de AIR-toepassingsprogrammalogica een object ApplicationUpdaterUI.

Gebruik in ActionScript de volgende code:

```
var appUpdater:ApplicationUpdaterUI = new ApplicationUpdaterUI();
```

Gebruik in JavaScript de volgende code:

```
var appUpdater = new runtime.air.update.ApplicationUpdaterUI();
```

U kunt desgewenst deze code toevoegen aan een initialisatiefunctie die wordt uitgevoerd wanneer de toepassing is geladen.

3 Maak een tekstbestand met de naam updateConfig.xml en voeg daar het volgende aan toe:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration xmlns="http://ns.adobe.com/air/framework/update/configuration/1.0">
    <url>http://example.com/updates/update.xml</url>
    <delay>1</delay>
</configuration>
```

Bewerk het `URL`-element van het bestand updateConfig.xml zodat dit overeenkomt met de uiteindelijke locatie van het update-descriptorbestand op uw webserver (zie de volgende procedure).

`delay` is het aantal dagen dat de toepassing wacht voordat wordt gecontroleerd of er een update beschikbaar is.

4 Voeg het bestand updateConfig.xml toe aan de projectmap van uw AIR-toepassing.

5 Zorg dat het updater-object verwijst naar het bestand updateConfig.xml en roep de methode `initialize()` van het object aan.

Gebruik in ActionScript de volgende code:

```
appUpdater.configurationFile = new File("app:/updateConfig.xml");
appUpdater.initialize();
```

Gebruik in JavaScript de volgende code:

```
appUpdater.configurationFile = new air.File("app:/updateConfig.xml");
appUpdater.initialize();
```

6 Maak een tweede versie van de AIR-toepassing, die verschilt van de eerste versie van deze toepassing. (De versie wordt gespecificeerd in het descriptorbestand van de toepassing, in het element `version`.)

Voeg vervolgens de geüpdate versie van de AIR-toepassing toe aan uw webserver:

1 Plaats de geüpdate versie van het AIR-bestand op uw webserver.

2 Maak een tekstbestand genaamd updateDescriptor.2.5.xml en voeg er de volgende inhoud aan toe:

```
<?xml version="1.0" encoding="utf-8"?>
     <update xmlns="http://ns.adobe.com/air/framework/update/description/2.5">
       <versionNumber>1.1</versionNumber>
       <url>http://example.com/updates/sample_1.1.air</url>
       <description>This is the latest version of the Sample application.</description>
    </update>
```

Bewerk `versionNumber`, `URL` en `description` van het bestand updatedescriptor.xml zo dat deze overeenkomen met uw AIR-bestand. Deze updatedescriptorindeling wordt gebruikt door toepassingen die het updateframework gebruiken van de SDK van AIR 2.5 (en hogere versies).

3 Maak een tekstbestand genaamd updateDescriptor.1.0.xml en voeg er de volgende inhoud aan toe:

```
<?xml version="1.0" encoding="utf-8"?>
     <update xmlns="http://ns.adobe.com/air/framework/update/description/1.0">
       <version>1.1</version>
       <url>http://example.com/updates/sample_1.1.air</url>
       <description>This is the latest version of the Sample application.</description>
    </update>
```

Bewerk de `version`, `URL` en `description` van het bestand updateDescriptor.xml zodat deze overeenkomen met uw geüpdate AIR-bestand. Deze updatedescriptorindeling wordt gebruikt door toepassingen die het updateframework gebruiken van de SDK van AIR 2 (en eerdere versies).

***Opmerking:*** *het maken van dit tweede updatedescriptorbestand is alleen nodig wanneer u updates voor toepassingen ondersteunt die gemaakt zijn vóór AIR 2.5.*

4 Voeg het bestand updateDescriptor.2.5.xml toe en werk het bestand updateDescriptor.1.0.xml bij met dezelfde webservermap die het AIR-updatebestand bevat.

Dit is een eenvoudig voorbeeld, maar de updatefunctionaliteit die hier beschreven wordt is voldoende voor veel toepassingen. In de rest van dit document wordt beschreven hoe u het updateframework zo kunt gebruiken dat het optimaal voldoet aan uw vereisten.

Een ander voorbeeld van het gebruik van het updateframework vindt u in de volgende voorbeeldtoepassing in het Adobe AIR Developer Center:

- Update-raamwerk in een Flash-toepassing (http://www.adobe.com/go/learn_air_qs_update_framework_flash_nl)

## Bijwerken naar AIR 2.5

Omdat de regels voor het toewijzen van versienummers aan toepassingen is gewijzigd in AIR 2.5, kan het AIR 2-updateframework de versiegegevens in een AIR 2.5-toepassingsdescriptor niet parsen. Deze incompatibiliteit betekent dat u uw toepassing zo moet bijwerken dat deze het nieuwe updateframework gebruikt VOORDAT u uw toepassing zo bijwerkt dat deze de SDK van AIR 2.5 gebruikt. Het bijwerken van uw toepassing naar AIR 2.5 of hoger van elke eerdere AIR-versie dan 2.5 vereist dus TWEE updates. De eerste update moet de naamruimte AIR 2 gebruiken en de AIR 2.5-updateframeworkbibliotheek bevatten (u kunt het toepassingspakket nog steeds maken met de SDK van AIR 2.5). Bij de tweede update kunt u de naamruimte AIR 2.5 gebruiken en de nieuwe functies van uw toepassing opnemen.

U kunt ook de tussenupdate niets laten uitvoeren behalve het direct bijwerken van uw AIR 2.5-toepassing met de AIR-Updater-klasse.

In het volgende voorbeeld wordt beschreven hoe u een toepassing kunt bijwerken van versie 1.0 naar 2.0. Versie 1.0 gebruikte oude naamruimte 2.0. Versie 2.0 gebruikt de naamruimte 2.5 en bevat nieuwe functies die zijn geïmplementeerd met AIR 2.5-API's.

1 Maak een tussenversie van de toepassing, versie 1.0.1, op basis van versie 1.0 van de toepassing.

   a Gebruik het AIR 2.5-toepassingsupdaterframework bij het maken van de toepassing.

   ***Opmerking:*** *gebruik* `applicationupdater.swc` *of* `applicationupdater_ui.swc` *voor AIR-toepassingen die zijn gebaseerd op Flash-technologie en* `applicationupdater.swf` *of* `applicationupdater_ui.swf` *voor AIR-toepassingen op basis van HTML.*

   b Maak een updatedescriptorbestand voor versie 1.0.1 door de oude naamruimte en de versie hieronder te gebruiken:

```
<?xml version="1.0" encoding="utf-8"?>
    <update xmlns="http://ns.adobe.com/air/framework/update/description/2.0">
        <version>1.0.1</version>
        <url>http://example.com/updates/sample_1.0.1.air</url>
        <description>This is the intermediate version.</description>
    </update>
```

2 Maak de versie 2.0 van de toepassing die AIR 2.5-API's en naamruimte 2.5 gebruikt.

3 Maak een updatedescriptor om de toepassing bij te werken van versie 1.0.1 naar 2.0.

```
<?xml version="1.0" encoding="utf-8"?>
    <update xmlns="http://ns.adobe.com/air/framework/update/description/2.5">
        <version>2.0</version>
        <url>http://example.com/updates/sample_2.0.air</url>
        <description>This is the intermediate version.</description>
    </update>
```

## De updatedescriptorbestanden definiëren en het AIR-bestand aan uw webserver toevoegen

Wanneer u het AIR-updateframework gebruikt, definieert u de basisgegevens over de beschikbare update in updatedescriptorbestanden die worden opgeslagen op uw webserver. Een updatedescriptorbestand is een eenvoudig XML-bestand. Het updateframework dat is opgenomen in de toepassing, controleert dit bestand om na te gaan of een nieuwe versie is geüpload.

De indeling van het updatedescriptorbestand is gewijzigd voor AIR 2.5. De nieuwe indeling maakt gebruik van een andere naamruimte: De oorspronkelijke naamruimte is “http://ns.adobe.com/air/framework/update/description/1.0”. De naamruimte van AIR 2.5 is "http://ns.adobe.com/air/framework/update/description/2.5".

AIR-toepassingen die vóór AIR 2.5 zijn gemaakt, kunnen alleen de updatedescriptor van versie 1.0 lezen. AIR-toepassingen die zijn gemaakt met het updaterframework van AIR 2.5 of hoger, kunnen alleen de updatedescriptor van versie 2.5 lezen. Vanwege deze oncompatibiliteit in versies moet u vaak twee updatedescriptorbestanden maken. De updatelogica in de AIR 2.5-versies van uw toepassing moet een updatedescriptor downloaden die gebruikmaakt van de nieuwe indeling. Eerdere versies van uw AIR-toepassing moeten de oorspronkelijke indeling blijven gebruiken. Beide bestanden moeten worden bewerkt voor elke update die u uitbrengt (totdat u stopt met het ondersteunen van versies die zijn gemaakt vóór AIR 2.5).

Het update-descriptorbestand bevat de volgende gegevens:

- `versionNumber`—De nieuwe versie van de AIR-toepassing. Gebruik het element `versionNumber` in updatedescriptors die worden gebruikt om AIR 2.5-toepassingen bij te werken. De waarde moet dezelfde tekenreeks zijn die wordt gebruikt in het element `versionNumber` van de nieuwe AIR-toepassingdescriptorbestand. Als het versienummer in het update-descriptorbestand niet overeenkomt met het versienummer van het geüpdate AIR-bestand, produceert het framework een uitzonderingsfout.
- `version`—De nieuwe versie van de AIR-toepassing. Gebruik de `version`-elementen in updatedescriptors die worden gebruikt om toepassingen bij te werken die vóór AIR 2.5 zijn gemaakt. De waarde moet dezelfde reeks zijn die wordt gebruikt in het element `version` van het nieuwe AIR-toepassingsdescriptorbestand. Als de versie in het update-descriptorbestand niet overeenkomt met de versie van het geüpdate AIR-bestand, produceert het framework een uitzonderingsfout.
- `versionLabel`—De leesbare versietekenreeks, bedoeld om te worden weergegeven aan gebruikers. Het `versionLabel` is optioneel, maar kan alleen worden opgegeven in updatedescriptorbestanden versie 2.5. Gebruik dit als u een `versionLabel` in de toepassingsdescriptor gebruikt en stel in op dezelfde waarde.
- `url`—De locatie van het geüpdate AIR-bestand. Dit is het bestand dat de geüpdate versie van de AIR-toepassing bevat.
- `description`—Meer informatie over de nieuwe versie. Deze informatie kan tijdens het bijwerken worden weergegeven aan de gebruiker.

De elementen `version` en `url` zijn verplicht. Het element `description` is optioneel.

Dit is een voorbeeld van een update-descriptorbestand versie 2.5:

```
<?xml version="1.0" encoding="utf-8"?>
     <update xmlns="http://ns.adobe.com/air/framework/update/description/2.5">
       <versionNumber>1.1.1</versionNumber>
       <url>http://example.com/updates/sample_1.1.1.air</url>
       <description>This is the latest version of the Sample application.</description>
    </update>
```

En hier volgt een voorbeeld van een update-descriptorbestand versie 1.0:

```
<?xml version="1.0" encoding="utf-8"?>
     <update xmlns="http://ns.adobe.com/air/framework/update/description/1.0">
       <version>1.1.1</version>
       <url>http://example.com/updates/sample_1.1.1.air</url>
       <description>This is the latest version of the Sample application.</description>
    </update>
```

Als u de tag `description` wilt definiëren voor meerdere talen, gebruikt u meerdere `text`-elementen die allemaal een `lang`-attribuut definiëren:

```
<?xml version="1.0" encoding="utf-8"?>
     <update xmlns="http://ns.adobe.com/air/framework/update/description/2.5">
       <versionNumber>1.1.1</versionNumber>
       <url>http://example.com/updates/sample_1.1.1.air</url>
       <description>
           <text xml:lang="en">English description</text>
           <text xml:lang="fr">French description</text>
           <text xml:lang="ro">Romanian description</text>
       </description>
    </update>
```

Plaats het update-descriptorbestand samen met het geüpdate AIR-bestand op uw webserver.

De map templates die wordt opgenomen bij de update-descriptor, bevat voorbeelden van update-descriptorbestanden. Hierbij zijn eentalige en meertalige versies.

## Een updater-object instantiëren

Nadat u het AIR-updateframework hebt geladen in uw code (zie "De Flex-ontwikkelomgeving instellen" op pagina 278 en "Frameworkbestanden opnemen in een HTML-gebaseerde AIR-toepassing" op pagina 278), moet u een updater-object instantiëren, zoals in het onderstaande voorbeeld.

ActionScript-voorbeeld:

```
var appUpdater:ApplicationUpdater = new ApplicationUpdater();
```

JavaScript-voorbeeld:

```
var appUpdater = new runtime.air.update.ApplicationUpdater();
```

De vorige code maakt gebruik van de klasse ApplicationUpdater (die geen gebruikersinterface biedt). Als u de klasse ApplicationUpdaterUI wilt gebruiken (die wel een gebruikersinterface biedt), gebruikt u het volgende.

ActionScript-voorbeeld:

```
var appUpdater:ApplicationUpdaterUI = new ApplicationUpdaterUI();
```

JavaScript-voorbeeld:

```
var appUpdater = new runtime.air.update.ApplicationUpdaterUI();
```

In de overige codevoorbeelden in dit document wordt ervan uitgegaan dat u een updater-object met de naam `appUpdater` hebt geïnstantieerd.

## De update-instellingen configureren

Zowel ApplicationUpdater als ApplicationUpdaterUI kunnen worden geconfigureerd via een configuratiebestand dat wordt geleverd met de toepassing, of via ActionScript of JavaScript in de toepassing.

### Update-instellingen definiëren in een XML-configuratiebestand

Het update-configuratiebestand is een XML-bestand. Dit kan de volgende elementen bevatten:

- `updateURL`— Een tekenreeks. Deze geeft de locatie van de update-descriptor op de externe server aan. Iedere geldige URLRequest-locatie is toegestaan. U moet de eigenschap `updateURL` definiëren via het configuratiebestand of via een script (zie "De updatedescriptorbestanden definiëren en het AIR-bestand aan uw webserver toevoegen" op pagina 281). U moet deze eigenschap definiëren voordat u de updater gebruikt (voordat u de methode `initialize()` van het updater-object aanroept, beschreven in "Het updateframework initialiseren" op pagina 286).
- `delay`—Een getal. Dit geeft het tijdsinterval in dagen aan (waarden zoals `0,25` zijn toegestaan) waarna op updates moet worden gecontroleerd. De standaardwaarde 0 specificeert dat de updater geen automatische periodieke controles uitvoert.

Het configuratiebestand voor de ApplicationUpdaterUI kan naast de elementen `updateURL` en `delay` het volgende bevatten:

- `defaultUI`: Een lijst van `dialog`-elementen. Ieder `dialog`-element heeft een attribuut `name` dat correspondeert met een dialoogvenster in de gebruikersinterface. Ieder `dialog`-element heeft een attribuut `visible` dat aangeeft of het dialoogvenster zichtbaar is. De standaardwaarde is `true`. Het attribuut `name` heeft de volgende mogelijke waarden:
  - `"checkForUpdate"`—Correspondeert met de dialoogvensters Check for Update, No Update en Update Error
  - `"downloadUpdate"`—Correspondeert met het dialoogvenster Download Update
  - `"downloadProgress"`—Correspondeert met de dialoogvensters Download Progress en Download Error
  - `"installUpdate"`—Correspondeert met het dialoogvenster Install Update
  - `"fileUpdate"`—Correspondeert met de dialoogvensters File Update, File No Update en File Error
- `"unexpectedError"`—Correspondeert met het dialoogvenster Unexpected Error

  Wanneer de waarde wordt ingesteld op `false`, wordt het corresponderende dialoogvenster niet weergegeven als onderdeel van de updateprocedure.

Hier ziet u een voorbeeld van het configuratiebestand voor het ApplicationUpdater-framework:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration xmlns="http://ns.adobe.com/air/framework/update/configuration/1.0">
      <url>http://example.com/updates/update.xml</url>
      <delay>1</delay>
</configuration>
```

Hier ziet u een voorbeeld van het configuratiebestand voor het ApplicationUpdaterUI-framework, dat een definitie van het `defaultUI`-element bevat:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration xmlns="http://ns.adobe.com/air/framework/update/configuration/1.0">
      <url>http://example.com/updates/update.xml</url>
      <delay>1</delay>
      <defaultUI>
         <dialog name="checkForUpdate" visible="false" />
         <dialog name="downloadUpdate" visible="false" />
         <dialog name="downloadProgress" visible="false" />
      </defaultUI>
</configuration>
```

Leid de eigenschap `configurationFile` naar de locatie van het bestand:

ActionScript-voorbeeld:

```
appUpdater.configurationFile = new File("app:/cfg/updateConfig.xml");
```

JavaScript-voorbeeld:

```
appUpdater.configurationFile = new air.File("app:/cfg/updateConfig.xml");
```

De map templates van het updateframework bevat een voorbeeldconfiguratiebestand, config-template.xml.

## Update-instellingen definiëren met behulp van ActionScript- of JavaScript-code

U kunt deze configuratieparameters ook instellen door middel van code in de toepassing, zoals in het volgende voorbeeld:

```
appUpdater.updateURL = " http://example.com/updates/update.xml";
appUpdater.delay = 1;
```

De eigenschappen van het updater-object zijn `updateURL` en `delay`. Deze eigenschappen definiëren dezelfde instellingen als de elementen `updateURL` en `delay` in het configuratiebestand: de URL van het update-descriptorbestand en het interval waarna op updates moet worden gecontroleerd. Als u een configuratiebestand *en* instellingen in code definieert, hebben de eigenschappen die zijn ingesteld via code voorrang op de corresponderende instellingen in het configuratiebestand.

U moet de eigenschap `updateURL` definiëren via het configuratiebestand of via een script (zie "De updatedescriptorbestanden definiëren en het AIR-bestand aan uw webserver toevoegen" op pagina 281) vóórdat u de updater gebruikt (voordat u de methode `initialize()` van het updater-object aanroept zoals beschreven in "Het updateframework initialiseren" op pagina 286).

Het ApplicationUpdaterUI-framework definieert de volgende aanvullende eigenschappen van het updater-object:

- `isCheckForUpdateVisible`—Correspondeert met de dialoogvensters Check for Update, No Update en Update Error
- `isDownloadUpdateVisible`—Correspondeert met het dialoogvenster Download Update
- `isDownloadProgressVisible`—Correspondeert met de dialoogvensters Download Progress en Download Error
- `isInstallUpdateVisible`—Correspondeert met het dialoogvenster Install Update
- `isFileUpdateVisible`—Correspondeert met de dialoogvensters File Update, File No Update en File Error
- `isUnexpectedErrorVisible`—Correspondeert met het dialoogvenster Unexpected Error

Iedere eigenschap correspondeert met een of meer dialoogvensters in de ApplicationUpdaterUI-gebruikersinterface. Iedere eigenschap is een Booleaanse waarde met een standaardwaarde van `true` (waar). Wanneer de waarde wordt ingesteld op `false`, worden de corresponderende dialoogvensters niet weergegeven als onderdeel van de updateprocedure.

Deze dialoogvenstereigenschappen hebben voorrang op de instellingen in het update-configuratiebestand.

## Het updateproces

Het AIR-updateframework voert het updateproces uit in de volgende stappen:

1. De updater-initialisatieprocedure controleert of er een updatecontrole is uitgevoerd binnen het gedefinieerde delay-interval (zie "De update-instellingen configureren" op pagina 283). Als er een updatecontrole moet worden uitgevoerd, wordt het updateproces voortgezet.
2. De updater downloadt en interpreteert het update-descriptorbestand.
3. De updater downloadt het geüpdate AIR-bestand.
4. De updater installeert de geüpdate versie van de toepassing.

Bij iedere stap die wordt voltooid, verzendt het updater-object gebeurtenissen. In de ApplicationUpdater-versie kunt u de gebeurtenissen annuleren die de geslaagde voltooiing van een stap in het proces aangeven. Als u een van deze gebeurtenissen annuleert, wordt de volgende stap in het proces geannuleerd. In de ApplicationUpdaterUI-versie geeft de updater een dialoogvenster weer waarmee de gebruiker het proces bij iedere stap kan annuleren of voortzetten.

Als u de gebeurtenis annuleert, kunt u methoden van het updater-object aanroepen om het proces te hervatten.

Terwijl de ApplicationUpdater-versie van de updater het updateproces doorloopt, wordt de huidige status ervan geregistreerd in de eigenschap `currentState`. Deze eigenschap wordt ingesteld op een tekenreeks die de volgende waarden kan hebben:

- `"UNINITIALIZED"`—De updater is niet geïnitialiseerd.

- `"INITIALIZING"`—De updater is bezig met initialiseren.
- `"READY"`—De updater is geïnitialiseerd.
- `"BEFORE_CHECKING"`—De updater heeft nog niet gecontroleerd op het update-descriptorbestand.
- `"CHECKING"`—De updater controleert op de aanwezigheid van een update-descriptorbestand.
- `"AVAILABLE"`—Het update-descriptorbestand is beschikbaar.
- `"DOWNLOADING"`—De updater downloadt het AIR-bestand.
- `"DOWNLOADED"`—De updater heeft het AIR-bestand gedownload.
- `"INSTALLING"`—De updater is het AIR-bestand aan het installeren.
- `"PENDING_INSTALLING"`—De updater is geïnitialiseerd en er zijn updates in behandeling.

Bepaalde methoden van het updater-object worden alleen uitgevoerd als de updater een bepaalde status heeft.

## Het updateframework initialiseren

Nadat u de configuratie-eigenschappen hebt ingesteld (zie "Eenvoudig voorbeelden: De ApplicationUpdaterUI-versie gebruiken" op pagina 279), roept u de methode `initialize()` aan om de update te initialiseren:

```
appUpdater.initialize();
```

Deze methode doet het volgende:

- De methode initialiseert het updateframework, waarbij eventuele updates in behandeling synchroon zonder toezicht worden geïnstalleerd. Het is verplicht deze methode aan te roepen tijdens het opstarten van de toepassing, omdat de toepassing tijdens het aanroepen mogelijk opnieuw wordt opgestart.
- Als er een uitgestelde update aanwezig is, wordt deze geïnstalleerd.
- Als er tijdens het updateproces een fout optreedt, worden het updatebestand en de versie-informatie gewist uit het opslaggebied van de toepassing.
- Als de opgegeven vertragingsperiode (delay) is verlopen, wordt het updateproces gestart. Als dat niet het geval is, wordt de timer opnieuw gestart.

Het aanroepen van deze methode kan als resultaat hebben dat het updater-object de volgende gebeurtenissen verzendt:

- `UpdateEvent.INITIALIZED`—Wordt verzonden wanneer de initialisatie is voltooid.
- `ErrorEvent.ERROR`—Wordt verzonden wanneer er tijdens de initialisatie een fout optreedt.

Na het verzenden van de gebeurtenis `UpdateEvent.INITIALIZED` is het updateproces voltooid.

Wanneer u de methode `initialize()` aanroept, start de updater het updateproces en worden alle stappen voltooid op basis van de ingestelde waarde voor delay. U kunt het updateproces echter altijd opstarten door de methode `checkNow()` van het updater-object aan te roepen:

```
appUpdater.checkNow();
```

Deze methode doet niets als het updateproces al actief is. Zo niet, dan wordt het updateproces gestart.

Het updater-object kan als resultaat van het aanroepen van de methode `checkNow()` de volgende gebeurtenis verzenden:

- gebeurtenis `UpdateEvent.CHECK_FOR_UPDATE` vlak voordat de poging het update-descriptorbestand te downloaden, wordt uitgevoerd.

Als u de gebeurtenis `checkForUpdate` annuleert, kunt u de methode `checkForUpdate()` van het updater-object aanroepen. (Zie de volgende sectie.) Als u de gebeurtenis niet annuleert, controleert het updateproces op het update-descriptorbestand.

## Het updateproces beheren in de ApplicationUpdaterUI-versie

In de ApplicationUpdaterUI-versie kan de gebruiker het proces annuleren door middel van de knop Cancel (Annuleren) in de dialoogvensters van de gebruikersinterface. U kunt het updateproces ook via programmacode annuleren door de methode `cancelUpdate()` van het object ApplicationUpdaterUI aan te roepen.

U kunt eigenschappen van het ApplicationUpdaterUI-object instellen of u kunt elementen in het updateconfiguratiebestand definiëren om te bepalen welke bevestigingen in dialoogvensters worden weergegeven door de updater. Raadpleeg "De update-instellingen configureren" op pagina 283 voor meer informatie.

## Het updateproces beheren in de ApplicationUpdater-versie

U kunt de methode `preventDefault()` van gebeurtenisobjecten aanroepen die zijn verzonden door het object ApplicationUpdater om stappen van het updateproces te annuleren (zie "Het updateproces" op pagina 285). Door het standaardgedrag te annuleren heeft uw toepassing de mogelijkheid om een bericht weer te geven waarin de gebruiker wordt gevraagd of deze wil doorgaan.

In de volgende secties wordt beschreven hoe u het updateproces kunt voortzetten wanneer een stap van het proces is geannuleerd.

### Het update-descriptorbestand downloaden en interpreteren

Het object ApplicationUpdater verzendt de gebeurtenis `checkForUpdate` voordat het updateproces begint, vlak voordat de updater probeert het update-descriptorbestand te downloaden. Als u het standaardgedrag van de gebeurtenis `checkForUpdate` annuleert, downloadt de updater het update-descriptorbestand niet. U kunt de methode `checkForUpdate()` gebruiken om het updateproces te hervatten:

```
appUpdater.checkForUpdate();
```

Door de methode `checkForUpdate()` aan te roepen, wordt het update-descriptorbestand asynchroon gedownload en geïnterpreteerd door de updater. Als resultaat van het aanroepen van de methode `checkForUpdate()` kan de updater de volgende gebeurtenissen verzenden:

- `StatusUpdateEvent.UPDATE_STATUS`—De updater heeft het update-descriptorbestand gedownload en geïnterpreteerd. Deze gebeurtenis heeft de volgende eigenschappen:
  - `available`—Een Booleaanse waarde. Wordt ingesteld op `true` als er een andere versie dan die van de huidige toepassing beschikbaar is; als de versie hetzelfde is, wordt deze ingesteld op `false`.
  - `version`—Een tekenreeks. De versie van het toepassingsdescriptorbestand van het updatebestand
  - `details`—Een array. Als er geen gelokaliseerde versies van de beschrijving aanwezig zijn, retourneert deze array een lege tekenreeks (`""`) als het eerste element en de beschrijving als het tweede element.

    Als er meerdere versies van de beschrijving aanwezig zijn (in het update-descriptorbestand), bevat deze array meerdere sub-array's. Iedere array heeft twee elementen: het eerste is een taalcode (bijvoorbeeld `"en"`), het tweede is de corresponderende beschrijving (een tekenreeks) voor die taal. Zie "De updatedescriptorbestanden definiëren en het AIR-bestand aan uw webserver toevoegen" op pagina 281.
- `StatusUpdateErrorEvent.UPDATE_ERROR`—Er is een fout opgetreden en de updater kon het update-descriptorbestand niet downloaden of interpreteren.

### Het update-AIR-bestand downloaden

Het object ApplicationUpdater verzendt de gebeurtenis `updateStatus` nadat de updater het update-descriptorbestand heeft gedownload en geïnterpreteerd. Het standaardgedrag is dat wordt gestart met het downloaden van het updatebestand als dit beschikbaar is. Als u het standaardgedrag annuleert, kunt u de methode `downloadUpdate()` aanroepen om het updateproces te hervatten:

```
appUpdater.downloadUpdate();
```

Door het aanroepen van deze methode downloadt de updater asynchroon de updateversie van het AIR-bestand.

De methode `downloadUpdate()` kan de volgende gebeurtenissen verzenden:

- `UpdateEvent.DOWNLOAD_START`—De verbinding met de server is tot stand gebracht. Wanneer de ApplicationUpdaterUI-bibliotheek wordt gebruikt, wordt door deze gebeurtenis een dialoogvenster weergegeven met een voortgangsbalk die de voortgang van het downloaden aangeeft.
- `ProgressEvent.PROGRESS`—Wordt periodiek verzonden terwijl het bestand wordt gedownload.
- `DownloadErrorEvent.DOWNLOAD_ERROR`—Wordt verzonden als er een fout optreedt bij de verbinding of het downloaden van het updatebestand. Wordt ook verzonden bij een ongeldige HTTP-status (bijvoorbeeld "404 - Bestand niet gevonden"). Deze gebeurtenis heeft een eigenschap `errorID`, een geheel getal waarmee aanvullende foutgegevens worden gedefinieerd. Een aanvullende eigenschap `subErrorID` kan meer foutgegevens bevatten.
- `UpdateEvent.DOWNLOAD_COMPLETE`—De updater heeft het update-descriptorbestand gedownload en geïnterpreteerd. Als u deze gebeurtenis niet annuleert, gaat de ApplicationUpdater-versie door met de installatie van de updateversie. In de ApplicationUpdaterUI-versie krijgt de gebruiker een dialoogvenster te zien waarin deze de mogelijkheid krijgt om door te gaan.

### De toepassing bijwerken

Het object ApplicationUpdater verzendt de gebeurtenis `downloadComplete` wanneer het downloaden van het updatebestand is voltooid. Als u het standaardgedrag annuleert, kunt u de methode `installUpdate()` aanroepen om het updateproces te hervatten:

```
appUpdater.installUpdate(file);
```

Door het aanroepen van deze methode installeert de updater een updateversie van het AIR-bestand. Deze methode omvat een parameter, `file`, die bestaat uit een object File dat verwijst naar het AIR-bestand dat als update moet worden gebruikt.

Het object ApplicationUpdater kan de gebeurtenis `beforeInstall` verzenden als resultaat van het aanroepen van de methode `installUpdate()`:

- `UpdateEvent.BEFORE_INSTALL`—Wordt verzonden vlak voordat de update wordt geïnstalleerd. In bepaalde gevallen is het aan te raden te voorkomen dat de update op dit moment wordt geïnstalleerd, zodat de gebruiker het huidige werk kan afmaken voordat de update wordt voortgezet. Als de methode `preventDefault()` van het object Event wordt aangeroepen, wordt de installatie uitgesteld totdat het systeem opnieuw wordt opgestart; er kan ook geen ander updateproces worden gestart. (Het gaat hierbij onder andere om updates die worden veroorzaakt door het aanroepen van de methode `checkNow()` of door de periodieke controle.)

## Installatie via een willekeurig AIR-bestand

U kunt de methode `installFromAIRFile()` aanroepen om de updateversie te installeren via een AIR-bestand op de computer van de gebruiker.

```
appUpdater.installFromAIRFile();
```

Door deze methode installeert de updater een updateversie van de toepassing vanuit het AIR-bestand.

De methode `installFromAIRFile()` kan de volgende gebeurtenissen verzenden:

- `StatusFileUpdateEvent.FILE_UPDATE_STATUS`—Wordt verzonden nadat de ApplicationUpdater het bestand heeft gevalideerd dat is verzonden via de methode `installFromAIRFile()`. Deze gebeurtenis heeft de volgende eigenschappen:
  - `available`—Wordt ingesteld op `true` als er een andere versie dan die van de huidige toepassing beschikbaar is; als de versie hetzelfde is, wordt deze waarde ingesteld op `false`.
  - `version` —De tekenreeks die de nieuwe beschikbare versie vertegenwoordigt.
  - `path`—Vertegenwoordigt het oorspronkelijke pad van het updatebestand.

  U kunt deze gebeurtenis annuleren als de eigenschap available van het object StatusFileUpdateEvent is ingesteld op `true`. Als u deze gebeurtenis annuleert, wordt de updater niet verder uitgevoerd. Roep de methode `installUpdate()` aan om de geannuleerde update verder uit te voeren.
- `StatusFileUpdateErrorEvent.FILE_UPDATE_ERROR`—Er is een fout opgetreden en de updater kon de AIR-toepassing niet installeren.

## Het updateproces annuleren

U kunt de methode `cancelUpdate()` gebruiken om het updateproces te annuleren:

```
appUpdater.cancelUpdate();
```

Met deze methode worden alle downloads die in behandeling zijn, geannuleerd. Verder worden alle onvolledig gedownloade bestanden verwijderd en wordt de periodieke controle op updates opnieuw opgestart.

Deze methode doet niets als het updater-object bezig is met initialiseren.

## De interface ApplicationUpdaterUI lokaliseren

De klasse ApplicationUpdaterUI biedt een standaardgebruikersinterface voor het updateproces. Deze interface is voorzien van dialoogvensters waarmee de gebruiker het proces kan starten, annuleren en andere handelingen kan uitvoeren.

Via het element `description` van het update-descriptorbestand kunt u de beschrijving van de toepassing definiëren in meerdere talen. Gebruik meerdere `text`-elementen die `lang`-attributen definiëren, zoals in het volgende voorbeeld:

```
<?xml version="1.0" encoding="utf-8"?>
     <update xmlns="http://ns.adobe.com/air/framework/update/description/1.0">
       <version>1.1a1</version>
       <url>http://example.com/updates/sample_1.1a1.air</url>
       <description>
           <text xml:lang="en">English description</text>
           <text xml:lang="fr">French description</text>
           <text xml:lang="ro">Romanian description</text>
       </description>
    </update>
```

Het updateframework gebruikte beschrijving die het best past bij de lokalisatieketen van de eindgebruiker. Zie Het update-descriptorbestand definiëren en het AIR-bestand toevoegen aan uw webserver voor meer informatie.

Flex-ontwikkelaars kunnen rechtstreeks een nieuwe taal toevoegen aan de bundel `"ApplicationUpdaterDialogs"`.

JavaScript-ontwikkelaars kunnen de methode `addResources()` van het updater-object aanroepen. Deze methode voegt dynamisch een nieuwe resourcebundel toe voor een taal. De resourcebundel definieert gelokaliseerde tekenreeksen voor een taal. Deze tekenreeksen worden gebruikt in de tekstvelden van de verschillende dialoogvensters.

JavaScript-ontwikkelaars kunnen de eigenschappen `localeChain` van de klasse ApplicationUpdaterUI gebruiken om de keten van landinstellingen die wordt gebruikt door de gebruikersinterface, te definiëren. In het algemeen maken alleen JavaScript (HTML)-ontwikkelaars gebruik van deze eigenschap. Flex-ontwikkelaars kunnen de ResourceManager gebruiken om de keten van landinstellingen te beheren.

De volgende JavaScript-code definieert bijvoorbeeld resourcebundels voor Roemeens en Hongaars:

```
appUpdater.addResources("ro_RO",
                 {titleCheck: "Titlu", msgCheck: "Mesaj", btnCheck: "Buton"});
appUpdater.addResources("hu", {titleCheck: "Cím", msgCheck: "Üzenet"});
var languages = ["ro", "hu"];
languages = languages.concat(air.Capabilities.languages);
var sortedLanguages = air.Localizer.sortLanguagesByPreference(languages,
                     air.Capabilities.language,
                     "en-US");
sortedLanguages.push("en-US");
appUpdater.localeChain = sortedLanguages;
```

Zie de beschrijving van de methode `addResources()` van de klasse ApplicationUpdaterUI in de taalreferentie voor meer informatie.

# Hoofdstuk 18: Broncode bekijken

Net zoals gebruikers de broncode van een HTML-pagina in een webbrowser kunnen bekijken, is het ook mogelijk om de broncode van een AIR-toepassing op HTML-basis te bekijken. De SDK van Adobe® AIR® bevat een JavaScript-bestand met de naam AIRSourceViewer.js waarmee de broncode in de toepassing bekeken kan worden.

## De Source Viewer laden, configureren en openen

De code voor de Source Viewer staat in een JavaScript-bestand met de naam AIRSourceViewer.js. Dit bestand staat in de map frameworks van de SDK van AIR. Als u de Source Viewer in een toepassing wilt gebruiken, kopieert u het bestand AIRSourceViewer.js naar de projectmap van de toepassing en laadt u het bestand via een scripttag in het hoofd-HTML-bestand in de toepassing:

```
<script type="text/javascript" src="AIRSourceViewer.js"></script>
```

Het bestand AIRSourceViewer.js definieert een klasse met de naam SourceViewer. Om vanuit JavaScript toegang tot deze klasse te krijgen roept u `air.SourceViewer` aan.

De klasse SourceViewer definieert drie methoden: `getDefault()`, `setup()` en `viewSource()`.

| Methode | Beschrijving |
|---|---|
| `getDefault()` | Een statische methode. Hiermee wordt een instantie SourceViewer geretourneerd, waarmee u de andere methoden kunt aanroepen. |
| `setup()` | Hiermee worden configuratie-instellingen op de Source Viewer toegepast. Zie "De Source Viewer configureren" op pagina 291 voor meer informatie. |
| `viewSource()` | Hiermee wordt een nieuw venster geopend waarin de gebruiker naar bronbestanden van de hosttoepassing kan bladeren en deze kan openen. |

***Opmerking:*** *code die gebruik maakt van de Source Viewer moet in de beveiligingssandbox van de toepassing staan (dit is een bestand in de toepassingsmap).*

Met de volgende JavaScript-code wordt bijvoorbeeld een Source Viewer-object ingesteld en wordt het venster Source Viewer geopend met daarin een lijst van alle bronbestanden:

```
var viewer = air.SourceViewer.getDefault();
viewer.viewSource();
```

### De Source Viewer configureren

De methode `config()` past gegeven instellingen toe op de Source Viewer. Deze methode heeft één parameter nodig: `configObject`. Het object `configObject` heeft eigenschappen die configuratie-instellingen voor de Source Viewer definiëren. Deze eigenschappen zijn: `default`, `exclude`, `initialPosition`, `modal`, `typesToRemove` en `typesToAdd`.

**default**
Een tekenreeks die het relatieve pad aangeeft ten opzichte van het eerste bestand dat in de Source Viewer wordt weergegeven.

Met de volgende JavaScript-code wordt bijvoorbeeld het venster van de Source Viewer geopend met het bestand index.html als eerste weergegeven bestand:

```
var viewer = air.SourceViewer.getDefault();
var configObj = {};
configObj.default = "index.html";
viewer.viewSource(configObj);
```

**exclude**

Een matrix met tekenreeksen waarmee bestanden of mappen worden aangegeven die worden weggelaten uit de lijst in de Source Viewer. De paden zijn relatief ten opzichte van de toepassingsmap. Jokertekens worden niet ondersteund.

Met de volgende JavaScript-code wordt bijvoorbeeld het Source Viewer-venster geopend met daarin een lijst van alle bronbestanden met uitzondering van het bestand AIRSourceViewer.js en bestanden in de subcategorieën Images en Sounds:

```
var viewer = air.SourceViewer.getDefault();
var configObj = {};
configObj.exclude = ["AIRSourceViewer.js", "Images" "Sounds"];
viewer.viewSource(configObj);
```

**initialPosition**

Een matrix die twee getallen bevat waarmee de eerste x- en y-coördinaten van het Source Viewer-venster worden aangegeven.

Met de volgende JavaScript-code wordt het Source Viewer-venster bijvoorbeeld geopend op de schermcoördinaten [40, 60] (X = 40, Y = 60):

```
var viewer = air.SourceViewer.getDefault();
var configObj = {};
configObj.initialPosition = [40, 60];
viewer.viewSource(configObj);
```

**modal**

Een Booleaanse waarde die aangeeft of de Source Viewer een modaal (true) of niet-modaal (false) venster is. Het Source Viewer-venster is standaard modaal.

Met de volgende JavaScript-code wordt het Source Viewer-venster bijvoorbeeld zodanig geopend dat de gebruiker kan communiceren met zowel het Source Viewer-venster als met andere toepassingsvensters:

```
var viewer = air.SourceViewer.getDefault();
var configObj = {};
configObj.modal = false;
viewer.viewSource(configObj);
```

**typesToAdd**

Een matrix met tekenreeksen waarmee wordt aangegeven welke bestandstypen in de lijst in de Source Viewer worden opgenomen, naast de standaardbestandstypen.

Standaard worden in de Source Viewer de volgende bestandstypen weergegeven:

- Tekstbestanden: TXT, XML, MXML, HTM, HTML, JS, AS, CSS, INI, BAT, PROPERTIES, CONFIG
- Afbeeldingsbestanden: JPG, JPEG, PNG, GIF

  Als er geen waarde is opgegeven, worden alle standaardtypen opgenomen (met uitzondering van de bestandstypen die in de eigenschap `typesToExclude` zijn opgegeven).

  Met de volgende JavaScript-code worden in het Source Viewer-venster bijvoorbeeld VCF- en VCARD-bestanden weergegeven:

```
var viewer = air.SourceViewer.getDefault();
var configObj = {};
configObj.typesToAdd = ["text.vcf", "text.vcard"];
viewer.viewSource(configObj);
```

Voor elk bestandstype moet u opgeven of het een tekstbestandstype ("text") of afbeeldingsbestandstype ("image") betreft.

**typesToExclude**

Een matrix met tekenreeksen waarmee wordt aangeven welke bestandstypen niet in de Source Viewer worden weergegeven.

Standaard worden in de Source Viewer de volgende bestandstypen weergegeven:

- Tekstbestanden: TXT, XML, MXML, HTM, HTML, JS, AS, CSS, INI, BAT, PROPERTIES, CONFIG
- Afbeeldingsbestanden: JPG, JPEG, PNG, GIF

  Met de volgende JavaScript-code worden in het Source Viewer-venster bijvoorbeeld geen GIF- en XML-bestanden weergegeven:

```
var viewer = air.SourceViewer.getDefault();
var configObj = {};
configObj.typesToExclude = ["image.gif", "text.xml"];
viewer.viewSource(configObj);
```

  Voor elk bestandstype moet u opgeven of het een tekstbestandstype (`"text"`) of een afbeeldingsbestandstype (`"image"`) betreft.

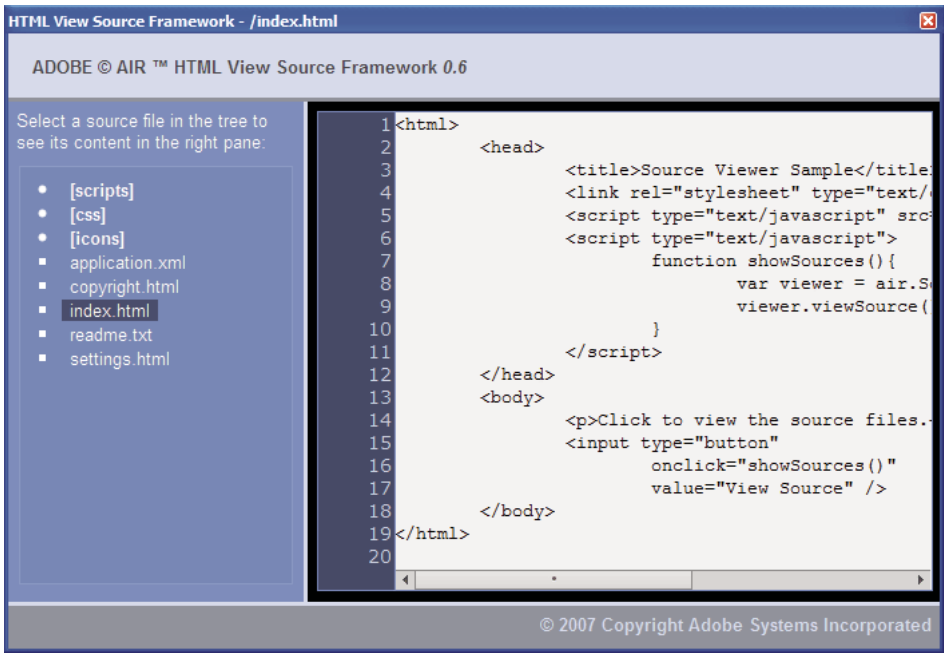
## De Source Viewer openen

Neem een gebruikersinterface-element op, bijvoorbeeld een koppeling, knop of menuopdracht, waarmee de Source Viewer-code kan worden aangeroepen. In de volgende eenvoudige toepassing wordt de Source Viewer bijvoorbeeld geopend wanneer de gebruiker op een koppeling klikt:

```
<html>
    <head>
        <title>Source Viewer Sample</title>
        <script type="text/javascript" src="AIRSourceViewer.js"></script>
        <script type="text/javascript">
            function showSources(){
                var viewer = air.SourceViewer.getDefault();
                viewer.viewSource()
            }
        </script>
    </head>
    <body>
        <p>Click to view the source files.</p>
        <input type="button"
            onclick="showSources()"
            value="View Source" />
    </body>
</html>
```

# Gebruikersinterface van de Source Viewer

Wanneer de toepassing de methode `viewSource()` van een object SourceViewer aanroept, wordt een Source Viewer-venster door AIR geopend. Dit venster bevat een lijst met bronbestanden en -mappen (links) en een weergavegebied met de broncode voor het geselecteerde bestand (rechts):




Mappen staan tussen vierkante haken. De gebruiker kan hierop klikken om een map uit te vouwen of samen te vouwen.

In de Source Viewer kan de bron worden weergegeven voor tekstbestanden met bekende extensies (zoals HTML, JS, TXT, XML, enzovoort) of voor afbeeldingsbestanden met bekende afbeeldingsbestandsextensies (JPG, JPEG, PNG en GIF). Als de gebruiker een bestand selecteert dat geen bekende bestandsextensie heeft, wordt een foutbericht weergegeven met de melding dat de tekstinhoud niet kan worden opgehaald.

Bronbestanden die zijn uitgesloten door middel van de methode `setup()` worden niet weergegeven (zie "De Source Viewer laden, configureren en openen" op pagina 291).

# Hoofdstuk 19: Fouten opsporen met de AIR HTML Introspector

De SDK van Adobe® AIR® bevat een JavaScript-bestand met de naam AIRIntrospector.js dat u kunt opnemen in een toepassing voor het opsporen van fouten in HTML-toepassingen.

## Informatie over de AIR Introspector

De AIR HTML/JavaScript Introspector biedt handige functies voor het ontwikkelen van HTML-toepassingen en het opsporen van fouten daarin:

- Een introspectorgereedschap waarmee u een element in de gebruikersinterface van de toepassing kunt aanwijzen zodat de opmaakcode en DOM-eigenschappen van dat element worden weergegeven.
- Een console voor het verzenden van objectverwijzingen voor introspectie en waarin u eigenschapwaarden kunt aanpassen en JavaScript-code kunt uitvoeren. Daarnaast kunt u objecten naar de console serialiseren, zodat de gegevens niet bewerkt kunnen worden. U kunt in de console ook tekst kopiëren en opslaan.
- Een boomstructuurweergave voor DOM-eigenschappen en -functies.
- Functies voor het bewerken van kenmerken en tekstknooppunten voor DOM-elementen.
- Lijsten met koppelingen, CSS-stijlen, afbeeldingen en JavaScript-bestanden die in de toepassing zijn geladen.
- Functies voor het weergeven van de eerste HTML-code en de huidige opmaakcode voor de gebruikersinterface.
- Toegang tot bestanden in de toepassingsmap. (Deze functie is alleen beschikbaar voor de AIR HTML Introspector-console die voor de toepassingssandbox is geopend. De functie is dus niet beschikbaar voor consoles die zijn geopend voor sandboxinhoud die niet gerelateerd is aan de toepassing.)
- Een viewer voor XMLHttpRequest-objecten en de bijbehorende eigenschappen, inclusief de eigenschappen `responseText` en `responseXML` (indien beschikbaar).
- Een zoekfunctie voor tekst in de broncode en bestanden.

## De AIR Introspector-code laden

De code voor de AIR Introspector staat in een JavaScript-bestand met de naam AIRIntrospector.js. Dit bestand staat in de map frameworks van de SDK van AIR. Als u de AIR Introspector in een toepassing wilt gebruiken, kopieert u het bestand AIRIntrospector.js naar de projectmap van de toepassing en laadt u het bestand via een scripttag in het hoofd-HTML-bestand in de toepassing:

```
<script type="text/javascript" src="AIRIntrospector.js"></script>
```

U moet het bestand ook opnemen in alle HTML-bestanden die overeenkomen met de verschillende vensters in de toepassing

***Belangrijk:*** *neem het bestand AIRIntrospector.js alleen op wanneer u de toepassing ontwikkelt en bij het opsporen van fouten in de toepassing. Verwijder het bestand uit het pakket met de AIR-toepassing dat u gaat distribueren.*

Het bestand AIRIntrospector.js definieert een klasse met de naam Console. Om vanuit JavaScript-code toegang te krijgen tot deze klasse, roept u `air.Introspector.Console` aan.

***Opmerking:*** *code die gebruik maakt van de AIR Introspector moet in de beveiligingssandbox van de toepassing staan (dit is een bestand in de toepassingsmap).*

# Een object op het tabblad Console inspecteren

De klasse Console definieert vijf methoden: `log()`, `warn()`, `info()`, `error()` en `dump()`.

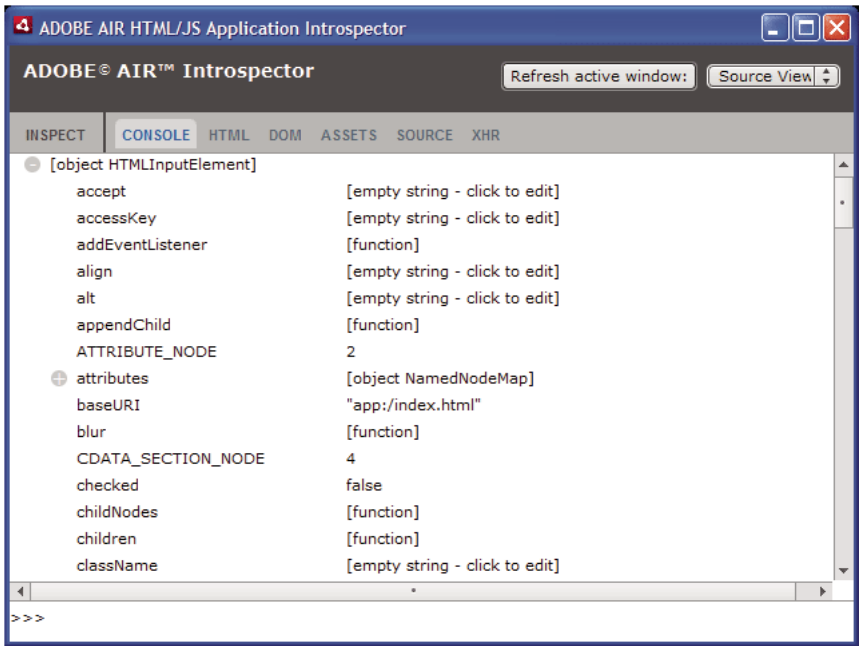
Met de methoden `log()`, `warn()`, `info()` en `error()` kunt u een object naar het tabblad Console verzenden. De eenvoudigste van deze methode is de methode `log()`. Met de volgende code wordt een eenvoudig object, aangegeven met de variabele `test`, naar het tabblad Console verzonden:

```
var test = "hello";
air.Introspector.Console.log(test);
```

Het is echter handiger om een complex object naar het tabblad Console te verzenden. Op bijvoorbeeld de volgende HTML-pagina staat een knop (`btn1`) waarmee een functie wordt aangeroepen die het knopobject zelf naar het tabblad Console verzendt:

```
<html>
    <head>
        <title>Source Viewer Sample</title>
        <script type="text/javascript" src="scripts/AIRIntrospector.js"></script>
        <script type="text/javascript">
            function logBtn()
            {
                var button1 = document.getElementById("btn1");
                air.Introspector.Console.log(button1);
            }
        </script>
    </head>
    <body>
        <p>Click to view the button object in the Console.</p>
        <input type="button" id="btn1"
            onclick="logBtn()"
            value="Log" />
    </body>
</html>
```

Wanneer op de knop wordt geklikt, wordt het object btn1 op het tabblad Console weergegeven en kunt u de boomstructuurweergave van het object uitvouwen om de eigenschappen te inspecteren:




Als u een eigenschap van het object wilt bewerken, klikt u op het item rechts naast de naam van de eigenschap zodat u de tekst kunt bewerken.

De methoden `info()`, `error()` en `warn()` werken net zoals de methode `log()`. Maar als u een van deze methoden aanroept, wordt in de Console een pictogram aan het begin van de regel weergegeven:

| Methode | Pictogram |
|---|---|
| `info()` | |
| `error()` | |
| `warn()` | |

De methoden `log()`, `warn()`, `info()` en `error()` verzenden alleen een verwijzing naar een werkelijk object. Dit betekent dat alleen de eigenschappen die aanwezig zijn op het moment van weergave beschikbaar zijn. U kunt het werkelijke object serialiseren met de methode `dump()`. Deze methode heeft twee parameters:

| Parameter | Beschrijving |
|---|---|
| `dumpObject` | Het object dat moet worden geserialiseerd. |
| `levels` | Het maximumaantal niveaus dat (naast het basisniveau) in de objectstructuur moet worden gecontroleerd. De standaardwaarde is 1. Dit houdt in dat één niveau boven het basisniveau van de boomstructuur wordt weergegeven. Deze parameter is optioneel. |

Als de methode `dump()` wordt aangeroepen, wordt een object geserialiseerd voordat het naar het tabblad Console wordt verzonden, zodat de objecteigenschappen niet bewerkt kunnen worden. Neem bijvoorbeeld de volgende code:

```
var testObject = new Object();
testObject.foo = "foo";
testObject.bar = 234;
air.Introspector.Console.dump(testObject);
```

Wanneer u deze code uitvoert, worden in de Console het object `testObject` en de bijbehorende eigenschappen weergegeven, maar de eigenschapwaarden kunnen niet worden bewerkt in de Console.

# De AIR Introspector configureren

Als u de console wilt configureren, stelt u de eigenschappen van de globale variabele `AIRIntrospectorConfig` in. Met bijvoorbeeld de volgende JavaScript-code stelt u in de AIR Introspector tekstterugloop in op 100 tekens:

```
var AIRIntrospectorConfig = new Object();
AIRIntrospectorConfig.wrapColumns = 100;
```

U moet de eigenschappen van de variabele `AIRIntrospectorConfig` instellen voordat het bestand AIRIntrospector.js wordt geladen (door middel van een `script` tag).

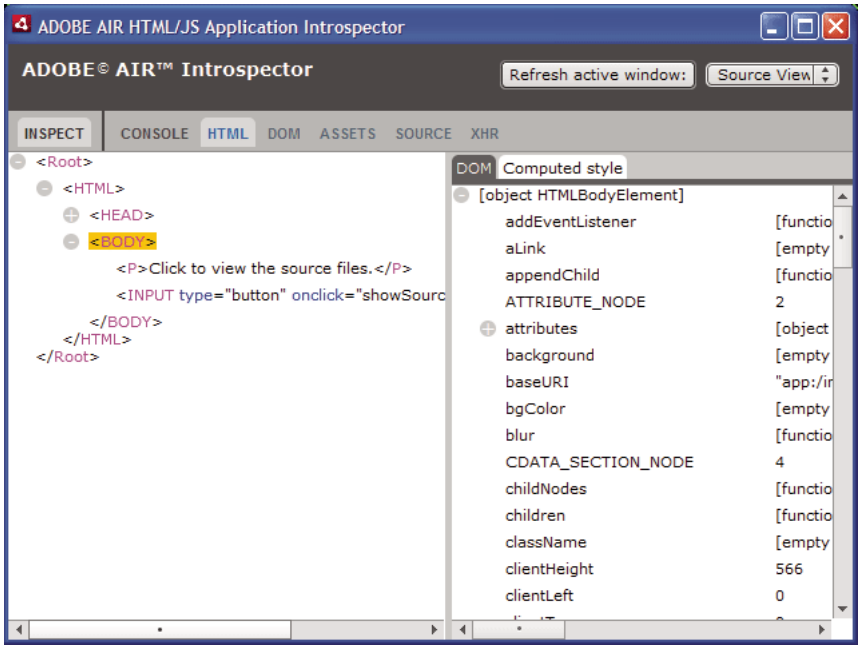
De variabele `AIRIntrospectorConfig` heeft acht eigenschappen:

| Eigenschap | Standaardwaarde | Beschrijving |
|---|---|---|
| `closeIntrospectorOnExit` | `true` | Hiermee wordt ingesteld dat het Inspector-venster wordt gesloten wanneer alle andere vensters van de toepassing worden gesloten. |
| `debuggerKey` | 123 (de toets F12) | De toetscode voor de sneltoets om het venster van de AIR Introspector weer te geven en te verbergen. |
| `debugRuntimeObjects` | `true` | Hiermee wordt ingesteld dat de Introspector naast objecten die in JavaScript zijn gedefinieerd ook runtime-objecten uitvouwt. |
| `flashTabLabels` | `true` | Hiermee wordt ingesteld dat op de tabbladen Console en XMLHttpRequest wordt aangegeven wanneer er sprake is van een wijziging (als er bijvoorbeeld tekst op deze tabbladen wordt gelogd). |
| `introspectorKey` | 122 (de toets F11) | De toetscode voor de snelkoppeling waarmee het controledeelvenster wordt geopend. |
| `showTimestamp` | `true` | Hiermee wordt ingesteld dat op het tabblad Console aan het begin van elke regel een tijdstempel wordt weergegeven. |
| `showSender` | `true` | Hiermee wordt ingesteld dat op het tabblad Console aan het begin van elke regel informatie wordt weergegeven over het object dat het bericht verzendt. |
| `wrapColumns` | 2000 | Het aantal kolommen voor terugloop van bronbestanden. |

# AIR Introspector-interface

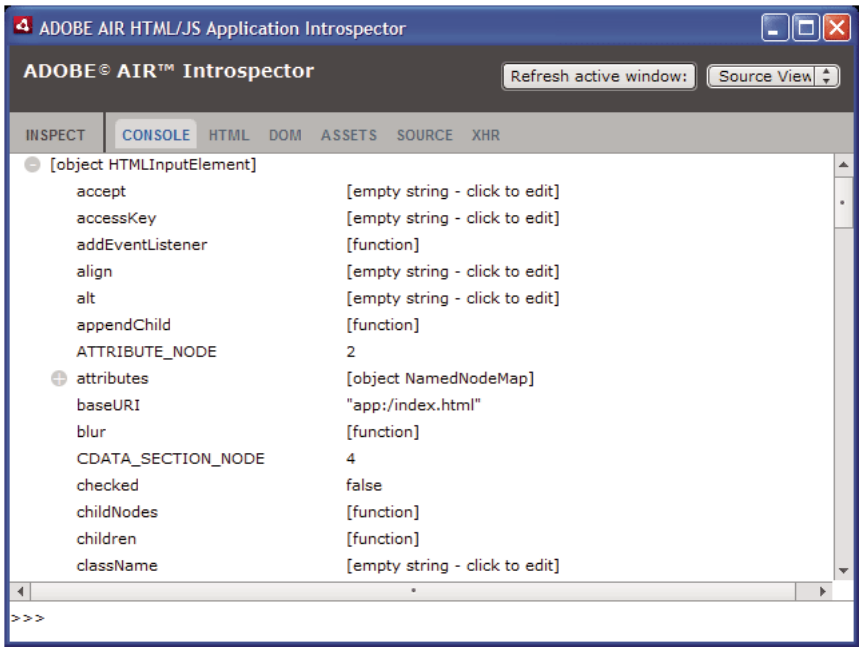
Als u het venster van AIR Introspector wilt openen tijdens het opsporen van fouten in de toepassing, drukt u op F12 of roept u een van de methoden van de klasse Console aan (zie “Een object op het tabblad Console inspecteren” op pagina 296). U kunt een andere sneltoets configureren in plaats van F12. Zie hiervoor “De AIR Introspector configureren” op pagina 298.

Het venster van de AIR Introspector heeft zes tabbladen: Console, HTML, DOM, Middelen, Bron en XHR, zoals wordt weergegeven in de volgende illustratie:




### Het tabblad Console

Op het tabblad Console worden de waarden weergegeven van eigenschappen die als parameters worden doorgegeven aan een van de methoden van de klasse air.Introspector.Console. Zie “Een object op het tabblad Console inspecteren” op pagina 296 voor meer informatie.

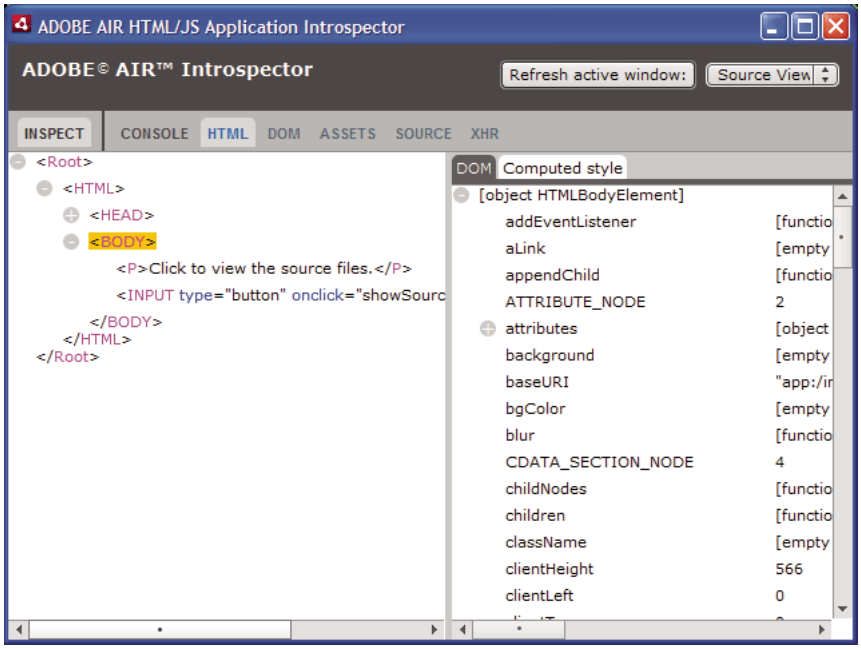



- Als u de console wilt leegmaken, klikt u met de rechtermuisknop op de tekst en selecteert u Console wissen.

- Als u de tekst op het tabblad Console wilt opslaan in een bestand, klikt u met de rechtermuisknop op het tabblad Console en selecteert u Console opslaan in bestand.
- Als u de tekst op het tabblad Console naar het klembord wilt kopiëren, klikt u met de rechtermuisknop op het tabblad Console en selecteert u Console opslaan op klembord. Als u alleen geselecteerde tekst naar het klembord wilt kopiëren, klikt u met de rechtermuisknop op de tekst en selecteert u Kopiëren.
- Als u de tekst in de klasse Console wilt opslaan in een bestand, klikt u met de rechtermuisknop op het tabblad Console en selecteert u Console opslaan in bestand.
- Als u wilt zoeken in de tekst die op het tabblad wordt weergegeven, klikt u op CTRL+F in Windows of Command+F in Mac OS. (Er wordt niet gezocht in knooppunten die niet zichtbaar zijn.)

### Het tabblad HTML

Op het tabblad HTML kunt u de gehele HTML DOM in een boomstructuur bekijken. Als u op een element klikt, worden de bijbehorende eigenschappen aan de rechterkant van het tabblad weergegeven. Als u op het plusteken (+) of minteken (-) klikt, wordt een knooppunt in de boomstructuur uitgevouwen of samengevouwen.




U kunt elk kenmerk of tekstelement op het tabblad HTML bewerken. Deze wijzigingen worden meteen in de toepassing weerspiegeld.

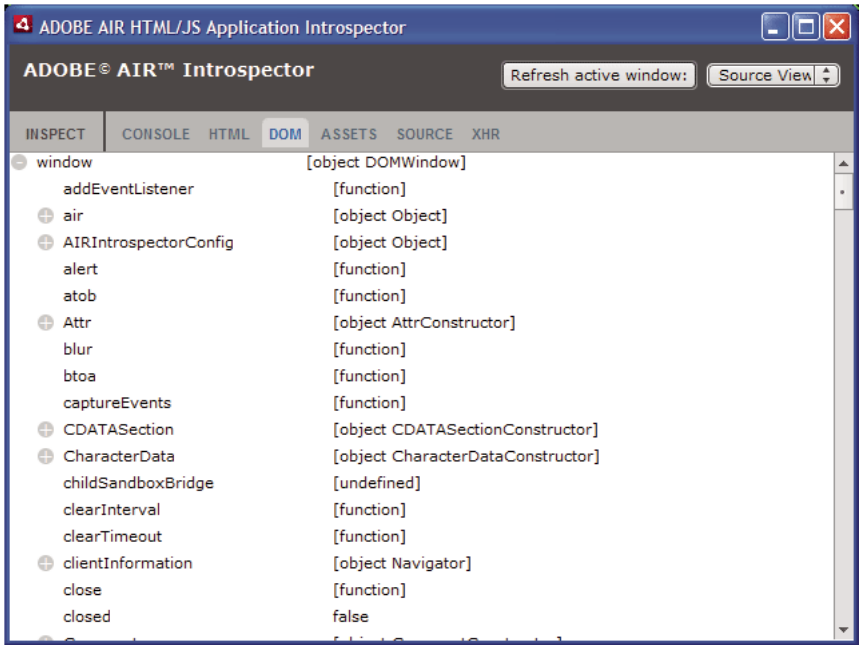
Klik op de knop Inspecteren (links naast de lijst met tabbladen in het venster van de AIR Introspector). Als u op een willekeurig element op de HTML-pagina van het hoofdvenster klikt, wordt het bijbehorende DOM-object op het tabblad HTML weergegeven. Wanneer het hoofdvenster de focus heeft, kunt u de knop Inspecteren ook met de snelkoppeling in- en uitschakelen. De snelkoppeling is standaard F11. U kunt een andere sneltoets configureren in plaats van F11. Zie hiervoor "De AIR Introspector configureren" op pagina 298.

Klik op de knop Actieve venster vernieuwen (boven aan het venster van de AIR Introspector) om de gegevens die op het tabblad HTML worden weergegeven te vernieuwen.

Klik op CTRL+F in Windows of Command+F in Mac OS om te zoeken in de tekst die op het tabblad wordt weergegeven. (Er wordt niet gezocht in knooppunten die niet zichtbaar zijn.)

### Het tabblad DOM

Op het tabblad DOM wordt het vensterobject in een boomstructuur weergegeven. U kunt elke tekenreeks en alle numerieke eigenschappen bewerken. Deze wijzigingen worden meteen in de toepassing weerspiegeld.

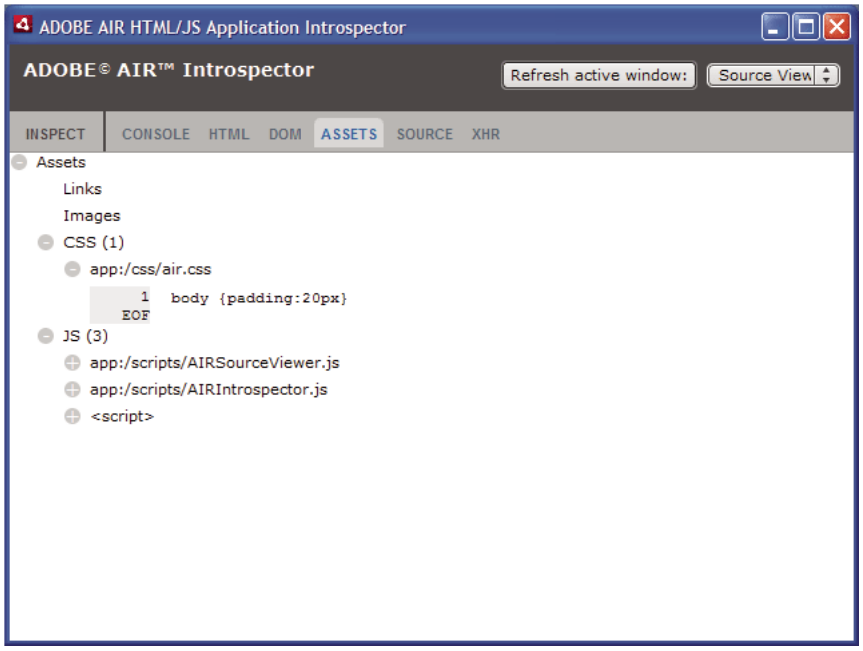



Klik op de knop Actieve venster vernieuwen (boven aan het venster van de AIR Introspector) om de gegevens die op het tabblad DOM worden weergegeven te vernieuwen.

Klik op CTRL+F in Windows of Command+F in Mac OS om te zoeken in de tekst die op het tabblad wordt weergegeven. (Er wordt niet gezocht in knooppunten die niet zichtbaar zijn.)

### Het tabblad Middelen

Op het tabblad Middelen kunt u de koppelingen, afbeeldingen en CSS- en JavaScript-bestanden controleren die in het native venster zijn geladen. Als u een van deze knooppunten uitvouwt, wordt de inhoud van het bestand of de gebruikte afbeelding weergegeven.




Klik op de knop Actieve venster vernieuwen (boven aan het venster van de AIR Introspector) om de gegevens die op het tabblad Middelen worden weergegeven te vernieuwen.

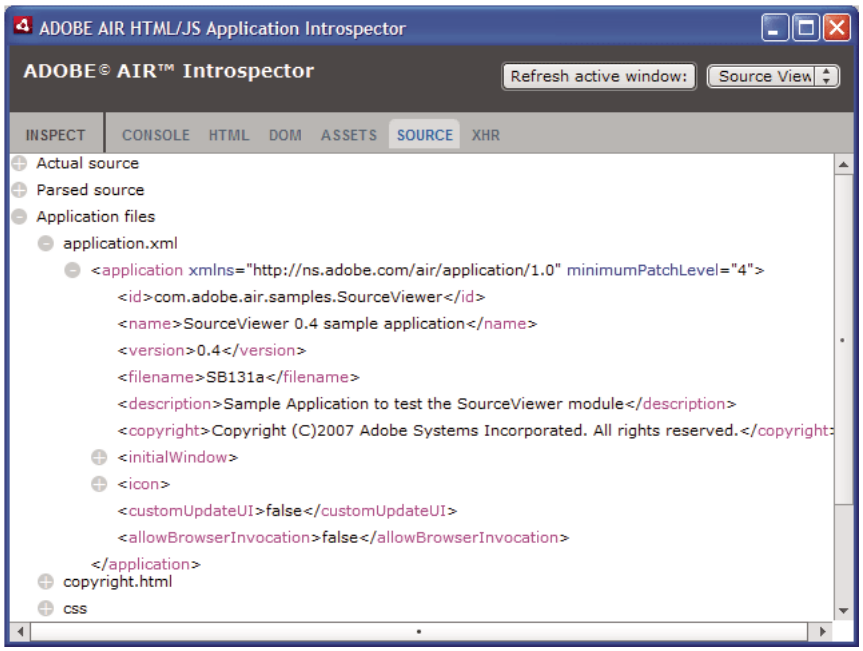
Klik op CTRL+F in Windows of Command+F in Mac OS om te zoeken in de tekst die op het tabblad wordt weergegeven. (Er wordt niet gezocht in knooppunten die niet zichtbaar zijn.)

### Het tabblad Bron

Het tabblad Bron bevat drie onderdelen:

- Werkelijke bron: hier wordt de HTML-broncode weergegeven van de pagina die bij het starten van de toepassing wordt geladen als basisinhoud.
- Geparseerde bron: hier wordt de huidige opmaakcode weergegeven die de interface van de toepassing vormt. Deze kan verschillen van de werkelijke bron, omdat de toepassingscode met behulp van Ajax de opmaakcode ter plekke genereert.

- Toepassingsbestanden: een lijst van de bestanden in de toepassingsmap. Deze lijst is alleen beschikbaar voor de AIR Introspector wanneer deze wordt gestart vanuit inhoud in de beveiligingssandbox van de toepassing. In dit onderdeel kunt u de inhoud van tekst of afbeeldingen bekijken.

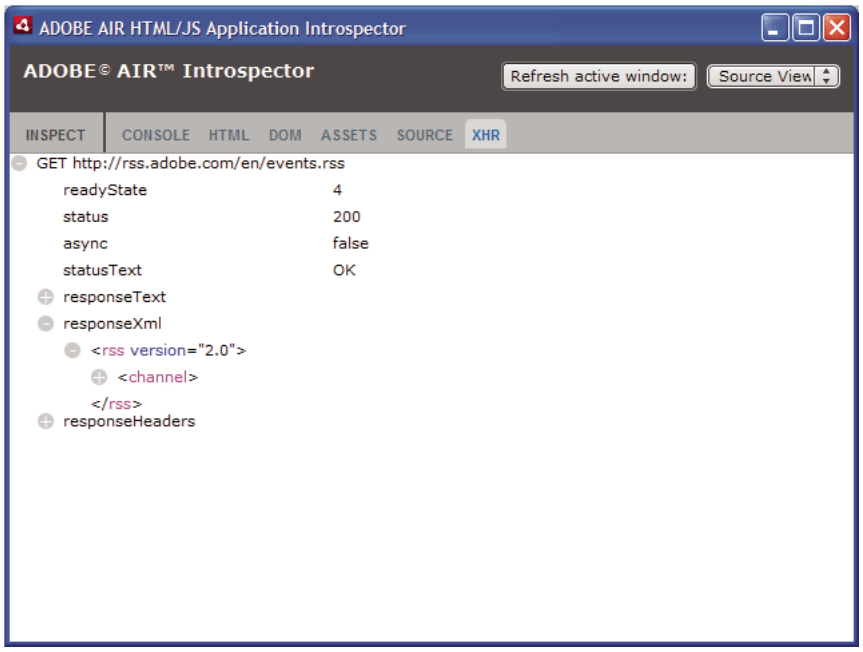



Klik op de knop Actieve venster vernieuwen (boven aan het venster van de AIR Introspector) om de gegevens die op het tabblad Bron worden weergegeven te vernieuwen.

Klik op CTRL+F in Windows of Command+F in Mac OS om te zoeken in de tekst die op het tabblad wordt weergegeven. (Er wordt niet gezocht in knooppunten die niet zichtbaar zijn.)

### Het tabblad XHR

Het tabblad XHR onderschept alle XMLHttpRequest-communicatie in de toepassing en slaat deze gegevens op in een logboek. Hierdoor kunt u de XMLHttpRequest-eigenschappen, zoals `responseText` en `responseXML` (indien beschikbaar) in een boomstructuurweergave bekijken.




Klik op CTRL+F in Windows of Command+F in Mac OS om te zoeken in de tekst die op het tabblad wordt weergegeven. (Er wordt niet gezocht in knooppunten die niet zichtbaar zijn.)

# De AIR Introspector gebruiken met inhoud in een niet-toepassingssandbox

U kunt inhoud vanuit de toepassingsmap laden naar een iframe of frame dat is toegewezen aan een niet-toepassingssandbox. (Zie HTML-beveiliging in Adobe AIR voor ActionScript-ontwikkelaars of HTML security in Adobe AIR voor ActionScript-ontwikkelaars). U kunt de AIR Introspector met dergelijke inhoud gebruiken, maar houd u daarbij aan de volgende regels:

- Het bestand AIRIntrospector.js moet worden opgenomen in de inhoud van de toepassingssandbox en in de inhoud van de niet-toepassingssandbox (het iframe).
- De eigenschap `parentSandboxBridge` mag niet worden overschreven omdat de AIR Introspector-code gebruik maakt van deze eigenschap. Voeg naar wens eigenschappen toe. Dus u schrijft niet het volgende:

  ```
  parentSandboxBridge = mytrace: function(str) {runtime.trace(str)}} ;
  ```

  In plaats daarvan gebruikt u de volgende syntaxis:

  ```
  parentSandboxBridge.mytrace = function(str) {runtime.trace(str)};
  ```

- Vanuit de niet-toepassingssandbox kunt u de AIR Introspector niet openen met F12 of door een van de methoden in de klasse air.Introspector.Console aan te roepen. U kunt het Introspector-venster alleen openen door te klikken op de knop Introspector openen. De knop staat standaard in de rechterbovenhoek van het iframe of frame. (Vanwege beveiligingsbeperkingen met betrekking tot inhoud in een niet-toepassingssandbox kan een nieuw venster alleen als gevolg van een actie van de gebruiker worden geopend.)
- U kunt verschillende AIR Introspector-vensters openen voor de toepassingssandbox en de niet-toepassingssandbox. Met de titels van de AIR Introspector-vensters kunt u het verschil tussen de twee vensters aangeven.
- Op het tabblad Bron worden geen toepassingsbestanden weergegeven wanneer de AIR Introspector wordt uitgevoerd vanuit een niet-toepassingssandbox.
- De AIR Introspector kan alleen kijken naar code in de sandbox van waaruit deze is geopend.

# Hoofdstuk 20: AIR-toepassingen lokaliseren

**Adobe AIR 1.1 en hoger**

Adobe® AIR® biedt ondersteuning voor meerdere talen.

Zie "Toepassingen lokaliseren" in de ActionScript 3.0-ontwikkelaarsgids voor een overzicht van het lokaliseren van inhoud in ActionScript 3.0 en het Flex-framework.

**Talen die in AIR worden ondersteund**

Vanaf AIR 1.1 wordt lokalisatie voor AIR-toepassingen ondersteund voor de volgende talen:

- Vereenvoudigd Chinees
- Traditioneel Chinees
- Frans
- Duits
- Italiaans
- Japans
- Koreaans
- Portugees (Braziliaans)
- Russisch
- Spaans

In AIR 1.5 zijn de volgende talen toegevoegd:

- Tsjechisch
- Nederlands
- Pools
- Zweeds
- Turks

## Meer Help-onderwerpen

Building multilingual Flex applications on Adobe AIR (Meertalige Flex-toepassingen op Adobe AIR maken)

Building a multilingual HTML-based application (Een meertalige toepassing op HTML-basis maken)

# De toepassingsnaam en -beschrijving lokaliseren in het installatieprogramma van de AIR-toepassing

**Adobe AIR 1.1 en hoger**

U kunt meerdere talen opgeven voor de elementen `name` en `description` in het descriptorbestand van de toepassing. Hieronder wordt bijvoorbeeld de toepassingsnaam in drie talen opgegeven (Engels, Frans en Duits):

```
<name>
    <text xml:lang="en">Sample 1.0</text>
    <text xml:lang="fr">Échantillon 1.0</text>
    <text xml:lang="de">Stichprobe 1.0</text>
</name>
```

Het attribuut `xml:lang` voor elk tekstelement geeft een taalcode aan, zoals gedefinieerd in RFC4646 (http://www.ietf.org/rfc/rfc4646.txt).

Het element name definieert de toepassingsnaam die wordt weergegeven in het installatieprogramma van de AIR-toepassing. Het installatieprogramma van de AIR-toepassing maakt gebruik van de gelokaliseerde waarde die het best overeenkomt met de gebruikersinterfacetalen die zijn gedefinieerd in de instellingen van het besturingssysteem.

Op dezelfde manier kunt u meerdere taalversies opgeven voor het element `description` in het descriptorbestand van de toepassing. Dit element definieert beschrijvende tekst die wordt weergegeven in het installatieprogramma van de AIR-toepassing.

Deze instellingen zijn alleen van toepassing op de talen die beschikbaar zijn in het installatieprogramma van de AIR-toepassing. Ze definiëren niet de landinstellingen die beschikbaar zijn voor de actieve geïnstalleerde toepassing. AIR-toepassingen kunnen gebruikersinterfaces bieden die meerdere talen ondersteunen, inclusief en als aanvulling op de talen die al beschikbaar zijn in het AIR-installatieprogramma.

Zie voor meer informatie “Descriptorelementen AIR-toepassing” op pagina 219.

### Meer Help-onderwerpen

Building multilingual Flex applications on Adobe AIR (Meertalige Flex-toepassingen op Adobe AIR maken)

Building a multilingual HTML-based application (Een meertalige toepassing op HTML-basis maken)

# HTML-inhoud lokaliseren met het AIR HTML-lokalisatieframework

**Adobe AIR 1.1 en hoger**

De AIR 1.1 SDK bevat een HTML-lokalisatieframework. Dit framework wordt gedefinieerd in het JavaScript-bestand AIRLocalizer.js. De map frameworks van de AIR SDK bevat het bestand AIRLocalizer.js. Dit bestand bevat de klasse air.Localizer, die functies bevat die helpen bij het maken van toepassingen die meerdere gelokaliseerde versies ondersteunen.

## De AIR HTML-lokalisatieframeworkcode laden

Als u het lokalisatieframework wilt gebruiken, kopieert u het bestand AIRLocalizer.js naar uw project. Neem het vervolgens met behulp van een scripttag op in het HTML-hoofdbestand van de toepassing.

```
<script src="AIRLocalizer.js" type="text/javascript" charset="utf-8"></script>
```

JavaScript kan dan in het vervolg het `air.Localizer.localizer`-object oproepen:

```
<script>
    var localizer = air.Localizer.localizer;
</script>
```

Het `air.Localizer.localizer`-object is een singletonobject waarin methoden en eigenschappen worden gedefinieerd voor het gebruik en het beheer van gelokaliseerde resources. De klasse Localizer bevat de volgende methoden:

| Methode | Beschrijving |
|---|---|
| `getFile()` | Haalt de tekst van een opgegeven resourcebundel op voor een opgegeven landinstelling. Zie "Resources ophalen voor een specifieke landinstelling" op pagina 314. |
| `getLocaleChain()` | Retourneert de talen in de keten van landinstellingen. Zie "De keten van landinstellingen definiëren" op pagina 314. |
| `getResourceBundle()` | Retourneert de bundelsleutels en de corresponderende waarden als een object. Zie "Resources ophalen voor een specifieke landinstelling" op pagina 314. |
| `getString()` | Haalt de tekenreeks op die is gedefinieerd voor een resource. Zie "Resources ophalen voor een specifieke landinstelling" op pagina 314. |
| `setBundlesDirectory()` | Stelt de locatie van de bundelmap in. Zie "AIR HTML Localizer-instellingen aanpassen" op pagina 313. |
| `setLocalAttributePrefix()` | Stelt het voorvoegsel in dat wordt gebruikt door localizer-kenmerken die worden gebruikt in HTML DOM-elementen. Zie "AIR HTML Localizer-instellingen aanpassen" op pagina 313 |
| `setLocaleChain()` | Stelt de volgorde in van de talen in de keten van landinstellingen. Zie "De keten van landinstellingen definiëren" op pagina 314. |
| `sortLanguagesByPreference()` | Sorteert de landinstellingen in de keten op basis van de volgorde van de landinstellingen in het besturingssysteem. Zie "De keten van landinstellingen definiëren" op pagina 314. |
| `update()` | Werkt het HTML DOM (of een DOM-element) bij met gelokaliseerde tekenreeksen uit de huidige keten van landinstellingen. Zie "Landinstellingen beheren" op pagina 310 voor meer informatie over ketens van landinstellingen. Meer informatie over de methode `update()` vindt u in "DOM-elementen bijwerken voor het gebruik van de huidige landinstelling" op pagina 311. |

De klasse Localizer bevat de volgende statische eigenschappen:

| Eigenschap | Beschrijving |
|---|---|
| `localizer` | Retourneert een verwijzing naar het Localizer-singletonobject voor de toepassing. |
| `ultimateFallbackLocale` | De landinstelling die moet worden gebruikt wanneer de toepassing geen gebruikersvoorkeur ondersteunt. Zie "De keten van landinstellingen definiëren" op pagina 314. |

## Ondersteunde talen opgeven

Met het element `<supportedLanguages>` in het descriptorbestand van de toepassing kunt u aangeven welke talen worden ondersteund door de toepassing. Dit element wordt alleen gebruikt door iOS-, Mac captive runtime- en Android-toepassingen. Het element wordt genegeerd door alle overige toepassingstypen.

Als u het element `<supportedLanguages>` niet opgeeft, worden standaard de volgende acties uitgevoerd door de pakketsoftware, afhankelijk van het toepassingstype:

- iOS — Alle talen die door de AIR-runtime worden ondersteund, zijn opgenomen in de iOS App Store als talen die door de toepassing worden ondersteund.
- Mac captive runtime — De toepassing die is opgenomen in het pakket met de captive-bundel bevat geen lokalisatiegegevens.
- Android — De toepassingsbundel heeft bronnen voor alle talen die worden ondersteund door de AIR-runtime.

Zie “supportedLanguages” op pagina 250 voor meer informatie.

## Resourcebundels definiëren

Het HTML-lokalisatieframework leest gelokaliseerde versies van tekenreeksen uit *lokalisatie* bestanden. Een lokalisatiebestand is een verzameling op sleutels gebaseerde waarden die serieel worden opgeslagen in een tekstbestand. Een lokalisatiebestand wordt ook wel een *bundel* genoemd.

Maak in de toepassingsprojectmap een submap met de naam locale. (U kunt ook een andere naam gebruiken. Zie “AIR HTML Localizer-instellingen aanpassen” op pagina 313.) Deze map zal de lokalisatiebestanden bevatten. Deze map wordt de *bundelmap* genoemd.

Voor iedere landinstelling die uw toepassing ondersteunt, maakt u een submap van de bundelmap. Geef iedere submap een naam die overeenkomt met de landinstellingscode. De map voor Frankrijk noemt u bijvoorbeeld “fr” en de map voor Engeland “en”. U kunt een onderstrepingsteken (_) gebruiken om een landinstelling te definiëren die een taal- en een landcode heeft. De map voor Amerikaans Engels noemt u bijvoorbeeld “en_us”. (U kunt ook een koppelteken gebruiken in plaats van een onderstrepingsteken, bijvoorbeeld “en-us”. Beide worden ondersteund door het HTML-lokalisatieframework.)

U kunt elk willekeurig aantal resourcebestanden toevoegen aan een submap locale. In het algemeen maakt u een lokalisatiebestand voor iedere taal (en zet u dat bestand in de map voor die taal). Het HTML-lokalisatieframework biedt de methode `getFile()`, waarmee u de inhoud van een bestand kunt lezen (zie “Resources ophalen voor een specifieke landinstelling” op pagina 314.

Bestanden met de extensie .properties staan bekend als lokalisatie-eigenschappenbestanden. U kunt deze gebruiken om sleutelwaardeparen voor een landinstelling te definiëren. In een eigenschappenbestand wordt op iedere regel één tekenreekswaarde gedefinieerd. Het volgende definieert bijvoorbeeld de tekenreekswaarde `"Hello in English."` voor de sleute `greeting`:

```
greeting=Hello in English.
```

Een eigenschappenbestand met de volgende tekst definieert zes sleutelwaardeparen:

```
title=Sample Application
greeting=Hello in English.
exitMessage=Thank you for using the application.
color1=Red
color2=Green
color3=Blue
```

Dit voorbeeld toont een Engelse versie van het eigenschappenbestand, die moet worden opgeslagen in de map en.

Een Franse versie van dit eigenschappenbestand wordt opgeslagen in de map fr:

```
title=Application Example
greeting=Bonjour en français.
exitMessage=Merci d'avoir utilisé cette application.
color1=Rouge
color2=Vert
color3=Bleu
```

U kunt meerdere resourcebestanden definiëren voor verschillende soorten informatie. Zo zou het bestand legal.properties juridische standaardteksten met bijvoorbeeld copyrightgegevens kunnen bevatten. U kunt deze bron in meerdere toepassingen hergebruiken. Ook kunt u eventueel afzonderlijke bestanden definiëren die gelokaliseerde inhoud definiëren voor verschillende onderdelen van de gebruikersinterface.

Gebruik UTF-8-codering voor deze bestanden voor de ondersteuning van meerdere talen.

## Landinstellingen beheren

Wanneer uw toepassing het bestand AIRLocalizer.js laadt, onderzoekt het bestand de landinstellingen die in de toepassing zijn gedefinieerd. Deze landinstellingen corresponderen met de submappen van de bundelmap (zie “Resourcebundels definiëren” op pagina 309). Dit is de *keten van landinstellingen*. Het bestand AIRLocalizer.js sorteert automatisch de keten van landinstellingen op basis van de voorkeursvolgorde die is gedefinieerd in het besturingssysteem. (De eigenschap `Capabilities.languages` geeft een lijst weer van de gebruikersinterfacetalen van het besturingssysteem, op volgorde van voorkeur.)

Als dus een toepassing resources definieert voor de landinstellingen "en", "en_US" en "en_UK", sorteert het AIR HTML Localizer-framework de keten van landinstellingen op de correcte manier. Wanneer een toepassing wordt gestart op een systeem waarop "en" wordt gerapporteerd als de primaire landcode, wordt de keten van landinstellingen gesorteerd als `["en", "en_US", "en_UK"]`. In dat geval zoekt de toepassing eerst naar resources in de bundel "en", en vervolgens in de bundel "en_US".

Als het systeem echter "en-US" rapporteert als de primaire landinstelling, wordt bij de sortering gebruikgemaakt van `["en_US", "en", en_UK"]`. In dat geval zoekt de toepassing eerst naar resources in de bundel "en_US", en vervolgens in de bundel "en".

Standaard definieert de toepassing de eerste landinstelling in de keten als de standaard landinstelling die moet worden gebruikt. U kunt de gebruiker vragen een landinstelling te selecteren wanneer deze de toepassing voor het eerst uitvoert. Vervolgens kunt u ervoor kiezen de selectie op te slaan in een voorkeurenbestand en die landinstelling te gebruiken wanneer de toepassing in het vervolg wordt opgestart.

Uw toepassing kan resourcetekenreeksen gebruiken in een willekeurige landinstelling van de keten. Als een bepaalde landinstelling geen resourcetekenreeks definieert, gebruikt de toepassing de volgende overeenkomende resourcetekenreeks voor andere landinstellingen die is gedefinieerd in de keten.

U kunt de keten van landinstellingen aanpassen door de methode `setLocaleChain()` van het Localizer-object op te roepen. Zie “De keten van landinstellingen definiëren” op pagina 314.

## DOM-elementen bijwerken met gelokaliseerde inhoud

Een element in de toepassing kan verwijzen naar een sleutelwaarde in een lokalisatie-eigenschappenbestand. Zo geeft het element `title` in het volgende voorbeeld het kenmerk `local_innerHTML` op. Het lokalisatieframework gebruikt dit kenmerk om een gelokaliseerde waarde op te zoeken. Standaard zoekt het framework naar kenmerknamen die beginnen met `"local_"`. Het framework werkt de kenmerken bij die een naam hebben die overeenkomt met de tekst na `"local_"`. In dit geval stelt het framework het kenmerk `innerHTML` van het element `title` in. Het kenmerk `innerHTML` gebruikt de waarde die is gedefinieerd voor de sleutel `mainWindowTitle` in het standaard eigenschappenbestand (default.properties):

```
<title local_innerHTML="default.mainWindowTitle"/>
```

Als de huidige landinstelling geen overeenkomende waarde definieert, doorzoekt het Localizer-framework de rest van de keten van landinstellingen. De volgende landinstelling in de keten waarvoor een waarde is gedefinieerd, wordt dan gebruikt.

In het volgende voorbeeld gebruikt de tekst (kenmerk `innerHTML`) van het element `p` de waarde van de sleutel `greeting` die is gedefinieerd in het standaard eigenschappenbestand:

```
<p local_innerHTML="default.greeting" />
```

In het volgende voorbeeld gebruikt het kenmerk value (en weergegeven tekst) van het element `input` de waarde van de sleutel `btnBlue` die is gedefinieerd in het standaard eigenschappenbestand:

```
<input type="button" local_value="default.btnBlue" />
```

Om het HTML DOM zodanig bij te werken dat de tekenreeksen worden gebruikt die zijn gedefinieerd in de huidige keten van landinstellingen, roept u de methode `update()` van het Localizer-object op. Het oproepen van de methode `update()` zorgt ervoor dat het Localizer-object het DOM parseert en manipulaties toepast waar het lokalisatiekenmerken (`"local_..."`) aantreft:

```
air.Localizer.localizer.update();
```

U kunt waarden definiëren voor zowel een kenmerk (zoals "innerHTML") als het corresponderende lokalisatiekenmerk (zoals "local_innerHTML"). In dat geval overschrijft het Localizer-framework alleen de waarde van het kenmerk als er een corresponderende waarde wordt aangetroffen in de lokalisatieketen. Het volgende element definieert bijvoorbeeld zowel kenmerken van het type `value` als van het type `local_value`:

```
<input type="text" value="Blue" local_value="default.btnBlue"/>
```

U kunt ook alleen een specifiek DOM-element bijwerken. Zie de volgende sectie, "DOM-elementen bijwerken voor het gebruik van de huidige landinstelling" op pagina 311.

Standaard gebruikt de AIR HTML Localizer `"local_"` als voorvoegsel voor kenmerken die lokalisatie-instellingen voor een element definiëren. Standaard definieert bijvoorbeeld het kenmerk `local_innerHTML` de bundel- en resourcenaam die worden gebruikt voor de waarde `innerHTML` van een element. Verder definieert het kenmerk `local_value` standaard de bundel- en resourcenaam die worden gebruikt voor het kenmerk `value` van een element. U kunt de Localizer zodanig configureren dat een ander kenmerkvoorvoegsel wordt gebruikt dan `"local_"`. Zie "AIR HTML Localizer-instellingen aanpassen" op pagina 313.

## DOM-elementen bijwerken voor het gebruik van de huidige landinstelling

Wanneer het Localizer-object het HTML DOM bijwerkt, maken gemarkeerde elementen gebruik van kenmerkwaarden op basis van tekenreeksen die zijn gedefinieerd in de huidige keten van landinstellingen. Als de HTML Localizer het HTML DOM moet bijwerken, roept u de methode `update()` van het `Localizer`-object op:

```
air.Localizer.localizer.update();
```

Als u alleen een specifiek DOM-element wilt bijwerken, geeft u dat element als parameter door aan de methode `update()`. De methode `update()` heeft maar één parameter, `parentNode`, die optioneel is. Wanneer de parameter `parentNode` wordt opgegeven, definieert deze het DOM-element dat moet worden gelokaliseerd. Als de methode `update()` wordt opgeroepen en de parameter `parentNode` wordt opgegeven, worden gelokaliseerde waarden ingesteld voor alle onderliggende elementen die lokalisatiekenmerken opgeven.

Kijk bijvoorbeeld naar het volgende `div`-element:

```
<div id="colorsDiv">
    <h1 local_innerHTML="default.lblColors" ></h1>
    <p><input type="button" local_value="default.btnBlue" /></p>
    <p><input type="button" local_value="default.btnRed" /></p>
    <p><input type="button" local_value="default.btnGreen" /></p>
</div>
```

Als u dit element wilt bijwerken zodat gelokaliseerde tekenreeksen worden gebruikt, gebruikt u de volgende JavaScript-code:

```
var divElement = window.document.getElementById("colorsDiv");
air.Localizer.localizer.update(divElement);
```

Als een sleutelwaarde niet wordt gevonden in de keten van landinstellingen, stelt het Localizer-framework de kenmerkwaarde in op de waarde van het kenmerk `"local_"`. Stel dat in het vorige voorbeeld het Localizer-framework in geen enkel standaard eigenschappenbestand in de keten van tekenreeksen een waarde kan vinden voor de sleutel `lblColors`. In dat geval wordt `"default.lblColors"` gebruikt als waarde voor `innerHTML`. Het gebruik van deze waarde geeft (voor de ontwikkelaar) aan dat er resources ontbreken.

De methode `update()` verzendt de gebeurtenis `resourceNotFound` wanneer een resource niet kan worden gevonden in de keten van landinstellingen. De constante `air.Localizer.RESOURCE_NOT_FOUND` definieert de tekenreeks `"resourceNotFound"`. De gebeurtenis heeft drie eigenschappen: `bundleName`, `resourceName` en `locale`. De eigenschap `bundleName` is de naam van de bundel waarin de resource niet is gevonden. De eigenschap `resourceName` is de resource die niet is gevonden. De eigenschap `locale` is de naam van de landinstelling waarin de resource niet is gevonden.

De methode `update()` verzendt de gebeurtenis `bundleNotFound` wanneer de opgegeven bundel niet kan worden gevonden. De constante `air.Localizer.BUNDLE_NOT_FOUND` definieert de tekenreeks `"bundleNotFound"`. De gebeurtenis heeft twee eigenschappen: `bundleName` en `locale`. De eigenschap `bundleName` is de naam van de bundel waarin de resource niet is gevonden. De eigenschap `locale` is de naam van de landinstelling waarin de resource niet is gevonden.

De methode `update()` werkt asynchroon (en verzendt de gebeurtenissen `resourceNotFound` en `bundleNotFound` asynchroon). Met de volgende code stelt u gebeurtenislisteners in voor de gebeurtenissen `resourceNotFound` en `bundleNotFound`:

```
air.Localizer.localizer.addEventListener(air.Localizer.RESOURCE_NOT_FOUND, rnfHandler);
air.Localizer.localizer.addEventListener(air.Localizer.BUNDLE_NOT_FOUND, rnfHandler);
air.Localizer.localizer.update();
function rnfHandler(event)
{
    alert(event.bundleName + ": " + event.resourceName + ":." + event.locale);
}
function bnfHandler(event)
{
    alert(event.bundleName + ":." + event.locale);
}
```

## AIR HTML Localizer-instellingen aanpassen

Met de methode `setBundlesDirectory()` van het Localizer-object kunt u het pad van de bundelmap aanpassen. Met de methode `setLocalAttributePrefix()` van het Localizer-object kunt u het pad van de bundelmap aanpassen, evenals de waarde van het kenmerk dat wordt gebruikt door de Localizer.

Het standaard bundelpad wordt gedefinieerd als de submap met landinstellingen van de toepassingsmap. U kunt een andere map opgeven door de methode `setBundlesDirectory()` van het Localizer-object op te roepen. Deze methode maakt gebruik van één parameter, `path`. Dit is het pad naar de gewenste bundelmap. Het is een tekenreeks. De parameter `path` kan een van de volgende waarden hebben:

- Een tekenreeks (String) die een pad op basis van de toepassingsmap definieert, bijvoorbeeld `"locales"`
- Een tekenreeks (String) die een geldige URL definieert die gebruikmaakt van de URL-schema's `app`, `app-storage` of `file`, zoals `"app://languages"` (maak *niet* gebruik van het URL-schema `http`)
- Een File-object

Raadpleeg de volgende bronnen voor informatie over URL's en directorypaden:

- Paden van File-objecten (voor ActionScript-ontwikkelaars)
- Paths of File objects (voor HTML-ontwikkelaars)

Met de volgende code stelt u bijvoorbeeld de bundelmap in op de submap languages van de opslagmap van de toepassing (en niet de toepassingsmap):

```
air.Localizer.localizer.setBundlesDirectory("languages");
```

Geef een geldig pad op voor de parameter `path`. Bij een ongeldig pad wordt de uitzondering BundlePathNotFoundError gegenereerd. Deze fout heeft `"BundlePathNotFoundError"` voor de eigenschap `name`. De eigenschap `message` geeft het ongeldige pad aan.

Standaard gebruikt de AIR HTML Localizer `"local_"` als voorvoegsel voor kenmerken die lokalisatie-instellingen voor een element definiëren. Het kenmerk `local_innerHTML` definieert bijvoorbeeld de bundel- en resourcenaam die worden gebruikt voor de waarde `innerHTML` van het volgende `input`-element:

```
<p local_innerHTML="default.greeting" />
```

Met de methode `setLocalAttributePrefix()` van het Localizer-object kunt u een ander kenmerkvoorvoegsel gebruiken dan `"local_"`. Deze statische methode maakt gebruik van één parameter, die bestaat uit de tekenreeks die u wilt gebruiken als kenmerkvoorvoegsel. Met de volgende code stelt u het Localizer-framework zo in dat "loc_" wordt gebruikt als kenmerkvoorvoegsel:

```
air.Localizer.localizer.setLocalAttributePrefix("loc_");
```

U kunt het kenmerkvoorvoegsel aanpassen dat door het Localizer-framework wordt gebruikt. U kunt het voorvoegsel bijvoorbeeld aanpassen als de standaardwaarde (`"local_"`) een conflict veroorzaakt met de naam van een ander kenmerk dat door uw code wordt gebruikt. Gebruik bij het oproepen van deze methode alleen geldige tekens voor HTML-kenmerken. (De waarde mag bijvoorbeeld geen spatie bevatten.)

Zie “DOM-elementen bijwerken met gelokaliseerde inhoud” op pagina 311 voor meer informatie over het gebruik van lokalisatiekenmerken in HTML-elementen.

De instellingen voor bundelmap en kenmerkvoorvoegsel blijven niet bewaard als een nieuwe sessie van de toepassing wordt opgestart. Als u een aangepaste instelling voor bundelmap of kenmerkvoorvoegsel gebruikt, moet u deze opnieuw instellen iedere keer dat de toepassing opnieuw wordt opgestart.

## De keten van landinstellingen definiëren

Wanneer u de AIRLocalizer.js-code laadt, wordt standaard de standaardketen voor landinstellingen ingesteld. Deze keten van landinstellingen wordt gedefinieerd door de landinstellingen die beschikbaar zijn in de bundelmap en de taalinstellingen van het besturingssysteem. (Zie "Landinstellingen beheren" op pagina 310 voor meer informatie.)

U kunt de keten van landinstellingen wijzigen door de statische methode `setLocaleChain()` van het Localizer-object op te roepen. U kunt deze methode bijvoorbeeld oproepen als de gebruiker een voorkeur voor een bepaalde taal opgeeft. De methode `setLocaleChain()` maakt gebruik van één parameter, `chain`, die bestaat uit een array van landinstellingen, bijvoorbeeld `["fr_FR","fr","fr_CA"]`. De volgorde van de landinstellingen in deze array bepaalt de volgorde waarin het framework (in daaropvolgende bewerkingen) zoekt naar resources. Als een resource niet wordt aangetroffen in de eerste landinstelling van de keten, wordt verder gezocht in de resources van de andere landinstellingen. Als het argument `chain` ontbreekt, geen array is of een lege array is, treedt er een fout op en wordt de uitzondering IllegalArgumentsError gegenereerd.

De statische methode `getLocaleChain()` van het Localizer-object retourneert een array met alle landinstellingen in de huidige keten.

De volgende code leest de huidige keten van landinstellingen en voegt boven aan de keten twee Franse landinstellingen toe:

```
var currentChain = air.Localizer.localizer.getLocaleChain();
newLocales = ["fr_FR", "fr"];
air.Localizer.localizer.setLocaleChain(newLocales.concat(currentChain));
```

De methode `setLocaleChain()` verzendt de gebeurtenis `"change"` wanneer de keten van landinstellingen wordt bijgewerkt. De constante `air.Localizer.LOCALE_CHANGE` definieert de tekenreeks `"change"`. Deze gebeurtenis heeft één eigenschap, `localeChain`, een array van landinstellingscodes in de nieuwe keten van landinstellingen. De volgende code stelt een gebeurtenislistener in voor deze gebeurtenis:

```
var currentChain = air.Localizer.localizer.getLocaleChain();
newLocales = ["fr_FR", "fr"];
localizer.addEventListener(air.Localizer.LOCALE_CHANGE, changeHandler);
air.Localizer.localizer.setLocaleChain(newLocales.concat(currentChain));
function changeHandler(event)
{
    alert(event.localeChain);
}
```

De statische eigenschap `air.Localizer.ultimateFallbackLocale` vertegenwoordigt de landinstelling die wordt gebruikt wanneer de toepassing geen gebruikersvoorkeuren ondersteunt. De standaardwaarde is `"en"`. Met de volgende code kunt u een andere landinstelling instellen:

```
air.Localizer.ultimateFallbackLocale = "fr";
```

## Resources ophalen voor een specifieke landinstelling

De methode `getString()` van het Localizer-object geeft als resultaat de tekenreeks die is gedefinieerd voor een resource in een specifieke landinstelling. U hoeft geen waarde voor `locale` op te geven wanneer u deze methode oproept. In dat geval doorzoekt de methode de gehele keten van landinstellingen. Als resultaat wordt de tekenreeks geretourneerd in de eerste landinstelling die de gegeven resourcenaam bevat. De methode heeft de volgende parameters:

| Parameter | Beschrijving |
|---|---|
| `bundleName` | De bundel die de resource bevat. Dit is de bestandsnaam van het eigenschappenbestand zonder de extensie .properties. (Als deze parameter bijvoorbeeld wordt ingesteld op `"alerts"`, zoekt de Localizer-code in lokalisatiebestanden met de naam alerts.properties.) |
| `resourceName` | De resourcenaam. |
| `templateArgs` | Optioneel. Een array van tekenreeksen die de genummerde tags in de vervangende tekenreeks vervangen. Hier volgt bijvoorbeeld een oproep van de functie, waarbij de parameter `templateArgs["Raúl", "4"]` is en de bijbehorende resource de tekenreeks `"Hello, {0}. You have {1} new messages."`. In dit geval retourneert de functie `"Hello, Raúl. You have 4 new messages."`. Als u deze instelling wilt negeren, geeft u de waarde `null` door. |
| `locale` | Optioneel. De landinstellingscode (bijvoorbeeld `"en"`, `"en_us"` of `"fr"`) die moet worden gebruikt. Als er een landinstelling wordt opgegeven en geen corresponderende waarde wordt gevonden, blijft de methode niet doorgaan met zoeken naar waarden in andere landinstellingen in de keten. Als er geen landinstellingscode wordt opgegeven, retourneert de functie de tekenreeks in de eerste keten die een waarde biedt voor de gegeven resourcenaam. |

Het Localizer-framework kan gemarkeerde HTML DOM-kenmerken bijwerken. U kunt gelokaliseerde tekenreeksen echter ook op andere manieren gebruiken. U kunt een tekenreeks bijvoorbeeld gebruiken in dynamisch gegenereerde HTML, of als parameterwaarde in een functieoproep. De volgende code roept bijvoorbeeld de functie `alert()` op met de tekenreeks die is gedefinieerd in de resource `error114` in het standaard eigenschappenbestand van de landinstelling fr_FR:

```
alert(air.Localizer.localizer.getString("default", "error114", null, "fr_FR"));
```

De methode `getString()` verzendt de gebeurtenis `resourceNotFound` wanneer de resource niet kan worden gevonden in de opgegeven bundel. De constante `air.Localizer.RESOURCE_NOT_FOUND` definieert de tekenreeks `"resourceNotFound"`. De gebeurtenis heeft drie eigenschappen: `bundleName`, `resourceName` en `locale`. De eigenschap `bundleName` is de naam van de bundel waarin de resource niet is gevonden. De eigenschap `resourceName` is de resource die niet is gevonden. De eigenschap `locale` is de naam van de landinstelling waarin de resource niet is gevonden.

De methode `getString()` verzendt de gebeurtenis `bundleNotFound` wanneer de opgegeven bundel niet kan worden gevonden. De constante `air.Localizer.BUNDLE_NOT_FOUND` definieert de tekenreeks `"bundleNotFound"`. De gebeurtenis heeft twee eigenschappen: `bundleName` en `locale`. De eigenschap `bundleName` is de naam van de bundel waarin de resource niet is gevonden. De eigenschap `locale` is de naam van de landinstelling waarin de resource niet is gevonden.

De methode `getString()` wordt asynchroon uitgevoerd (en verzendt de gebeurtenissen `resourceNotFound` en `bundleNotFound` asynchroon). Met de volgende code stelt u gebeurtenislisteners in voor de gebeurtenissen `resourceNotFound` en `bundleNotFound`:

```
air.Localizerlocalizer.addEventListener(air.Localizer.RESOURCE_NOT_FOUND, rnfHandler);
air.Localizerlocalizer.addEventListener(air.Localizer.BUNDLE_NOT_FOUND, bnfHandler);
var str = air.Localizer.localizer.getString("default", "error114", null, "fr_FR");
function rnfHandler(event)
{
    alert(event.bundleName + ": " + event.resourceName + ":." + event.locale);
}
function bnfHandler(event)
{
    alert(event.bundleName + ":." + event.locale);
}
```

De methode `getResourceBundle()` van het Localizer-object retourneert een specifieke bundel voor een bepaalde landinstelling. De geretourneerde waarde van de methode is een object waarvan de eigenschappen overeenstemmen met de sleutels in de bundel. (Als de toepassing de gespecificeerde bundel niet kan vinden, geeft deze methode de waarde `null` als resultaat.)

De methode heeft twee parameters, `locale` en `bundleName`.

| Parameter | Beschrijving |
|---|---|
| `locale` | De landinstelling (bijvoorbeeld `"fr"`). |
| `bundleName` | De bundelnaam. |

De volgende code roept bijvoorbeeld de methode `document.write()` aan om de standaardbundel voor de landinstelling fr te laden. Vervolgens wordt de methode `document.write()` aangeroepen om de waarden van de sleutels `str1` en `str2` in die bundel te schrijven:

```
var aboutWin = window.open();
var bundle = localizer.getResourceBundle("fr", "default");
aboutWin.document.write(bundle.str1);
aboutWin.document.write("<br/>");
aboutWin.document.write(bundle.str2);
aboutWin.document.write("<br/>");
```

De methode `getResourceBundle()` verzendt de gebeurtenis `bundleNotFound` wanneer de opgegeven bundel niet kan worden gevonden. De constante `air.Localizer.BUNDLE_NOT_FOUND` definieert de tekenreeks `"bundleNotFound"`. De gebeurtenis heeft twee eigenschappen: `bundleName` en `locale`. De eigenschap `bundleName` is de naam van de bundel waarin de resource niet is gevonden. De eigenschap `locale` is de naam van de landinstelling waarin de resource niet is gevonden.

De methode `getFile()` van het Localizer-object retourneert de inhoud van een bundel, in de vorm van een tekenreeks, voor een bepaalde landinstelling. Het bundelbestand wordt gelezen als UTF-8-bestand. De methode omvat de volgende parameters:

| Parameter | Beschrijving |
|---|---|
| `resourceFileName` | De bestandsnaam van het resourcebestand (bijvoorbeeld `"about.html"`). |
| `templateArgs` | Optioneel. Een array van tekenreeksen die de genummerde tags in de vervangende tekenreeks vervangen. Hier volgt bijvoorbeeld een oproep van de functie waarbij de parameter `templateArgs["Raúl", "4"]` is en waarbij het corresponderende resourcebestand twee regels bevat:<br>`<html>`<br>`<body>Hello, {0}. You have {1} new messages.</body>`<br>`</html>`<br>In dit geval retourneert de functie een tekenreeks van twee regels:<br>`<html>`<br>`<body>Hello, Raúl. You have 4 new messages. </body>`<br>`</html>` |
| `locale` | De landinstellingscode die moet worden gebruikt, bijvoorbeeld `"en_GB"`. Als er een landinstelling wordt opgegeven en geen corresponderend bestand wordt gevonden, blijft de methode niet doorgaan met zoeken in andere landinstellingen in de keten. Als er *geen* landinstellingscode wordt opgegeven, retourneert de functie de tekst in de eerste landinstelling in de keten waarbij een bestand hoort dat overeenkomt met `resourceFileName`. |

De volgende code roept bijvoorbeeld de methode `document.write()` op met gebruikmaking van de inhoud van het bestand about.html van de landinstelling fr:

```
var aboutWin = window.open();
var aboutHtml = localizer.getFile("about.html", null, "fr");
aboutWin.document.close();
aboutWin.document.write(aboutHtml);
```

De methode `getFile()` verzendt de gebeurtenis `fileNotFound` wanneer er geen resource kan worden gevonden in de keten van landinstellingen. De constante `air.Localizer.FILE_NOT_FOUND` definieert de tekenreeks `"resourceNotFound"`. De methode `getFile()` werkt asynchroon (en verzendt de gebeurtenis `fileNotFound` asynchroon). De gebeurtenis heeft twee eigenschappen: `fileName` en `locale`. De eigenschap `fileName` is de naam van het bestand dat niet kan worden gevonden. De eigenschap `locale` is de naam van de landinstelling waarin de resource niet is gevonden. De volgende code stelt een gebeurtenislistener in voor deze gebeurtenis:

```
air.Localizer.localizer.addEventListener(air.Localizer.FILE_NOT_FOUND, fnfHandler);
air.Localizer.localizer.getFile("missing.html", null, "fr");
function fnfHandler(event)
{
    alert(event.fileName + ": " + event.locale);
}
```

## Meer Help-onderwerpen

Building a multilingual HTML-based application (Een meertalige toepassing op HTML-basis maken)

# Hoofdstuk 21: Omgevingsvariabelen van het pad

De SDK van AIR bevat een aantal programma's die kunnen worden gestart via een opdrachtregel of terminal. Het uitvoeren van deze programma's kan vaak handiger zijn wanneer het pad naar de SDK-binmap is opgenomen in de omgevingsvariabele van het pad.

In de informatie die hier wordt gepresenteerd, wordt uitgelegd hoe u het pad voor Windows, Mac en Linux kunt instellen. De informatie dient als handige gids. Computerconfiguraties kunnen echter sterk afwijken, dus de procecure werkt niet voor elk systeem. In deze gevallen moet u de nodige informatie kunnen vinden in de documentatie van uw besturingssysteem of op het internet.

## Het pad instellen op Linux en Mac OS met behulp van de Bash-shell

Wanneer u een opdracht in een terminalvenster invoert, moet de shell, een programma dat uw invoert leest en probeert gepast te reageren, eerst het opdrachtprogramma op uw bestandssysteem vinden. De shell zoekt naar opdrachten in een lijst met mappen die is opgeslagen in een omgevingsvariabele die $PATH heet. Om te zien wat er op dit moment in het pad is ingevoerd, voert u het volgende in:

```
echo $PATH
```

Er wordt nu een lijst weergegeven met mappen, gescheiden door dubbele punten, die er als volgt uitziet:

```
/usr/bin:/bin:/usr/sbin:/usr/local/bin:/usr/x11/bin
```

Het doel is om het pad naar de SDK-binmap van AIR toe te voegen aan de lijst, opdat de shell de hulpmiddelen ADT en ADL kan vinden. Ervan uitgaande dat u de SDK van AIR op de locatie `/Users/fred/SDKs/AIR` hebt geplaatst, voegt de volgende opdracht de nodige mappen aan het pad toe:

```
export PATH=$PATH:/Users/fred/SDKs/AIR/bin:/Users/fred/SDKs/android/tools
```

***Opmerking:*** *als uw pad spaties bevat, voegt u een backslash toe, zoals in het volgende voorbeeld:*

```
/Users/fred\ jones/SDKs/AIR\ 2.5\ SDK/bin
```

U kunt de opdracht `echo` nogmaals gebruiken om er zeker van te zijn dat de opdracht is geslaagd:

```
echo $PATH
/usr/bin:/bin:/usr/sbin:/usr/local/bin:/usr/x11/bin:/Users/fred/SDKs/AIR/bin:/Users/fred/SDK
s/android/tools
```

Tot nu to gaat alles goed. Nu moet u de volgende opdrachten kunnen invoeren om een bemoedigend antwoord te krijgen:

```
adt -version
```

Als u uw $PATH-variabele juist hebt bewerkt, moet de opdracht de versie van ADT weergeven.

Er is echter nog een probleem over: de volgende keer dat u een nieuw terminalvenster opent, zult u zien dat de nieuwe invoer in het pad daar niet meer staat. De opdracht om het pad in te stellen, moet elke keer dat u een nieuwe terminal start, worden uitgevoerd.

Een algemene oplossing voor dit probleem is het toevoegen van een opdracht aan een van de opstartscripts die worden gebruikt door uw shell. Bij Mac OS kunt u het bestand bash_profile maken in de map ~/gebruikersnaam. Zo wordt de opdracht elke keer uitgevoerd dat u een nieuw terminalvenster opent. Bij Ubuntu is het opstartscript dat wordt uitgevoerd wanneer u een nieuw terminalvenster opent, .bashrc. Andere Linuxdistributies en shellprogramma's hebben vergelijkbare conventies.

U voegt als volgt de opdracht aan het shellscript voor opstarten toe:

1. Wijzig uw basismap:

   ```
   cd
   ```

2. Maak (indien nodig) het shellconfiguratieprofiel en voer de tekst aan het einde van bestand in met "`cat >>`". Gebruik het toepasselijke bestand voor uw besturingssysteem en shell. U kunt bijvoorbeeld `.bash_profile` op Mac OS en `.bashrc` op Ubuntu gebruiken.

   ```
   cat >> .bash_profile
   ```

3. Voer de tekst in die u aan het bestand wilt toevoegen:

   ```
   export PATH=$PATH:/Users/cward/SDKs/android/tools:/Users/cward/SDKs/AIR/bin
   ```

4. Sluit het bewerken van de tekst door op het toetsenbord op `CTRL-SHIFT-D` te drukken.

5. Geef het bestand weer om te controleren of alles in orde is:

   ```
   cat .bash_profile
   ```

6. Open een nieuw terminalvenster om het pad te controleren:

   ```
   echo $PATH
   ```

   Als het goed is, zijn uw padtoevoegingen opgenomen.

Als u later een nieuwe versie van een van de SDK's in een andere map maakt, zorgt u ervoor dat u de padopdracht in het configuratiebestand opneemt. Anders blijft de shell de oude versie gebruiken.

# Het pad instellen op Windows

Wanneer u een nieuw opdrachtvenster op Windows opent, worden in dat venster de algemene omgevingsvariabelen overgenomen die zijn gedefinieerd in de systeemeigenschappen. Een van de belangrijkste variabelen is het pad: een lijst met mappen die het opdrachtprogramma doorzoekt wanneer u de naam van een uti te voeren programma invoert. Als u witl zien wat er op dit moment is opgenomen in het pad wanneer u een opdrachtvenster gebruikt, kunt u het volgende invoeren:

```
set path
```

Er wordt een lijst weergegeven met mappen, gescheiden door puntkomma's, die er ongeveer zo uitziet:

```
Path=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem
```

Het doel is om het pad naar de SDK-binmap van AIR toe te voegen aan de lijst, opdat het opdrachtprogramma de hulpmiddelen ADT en ADL kan vinden. Ervan uitgaande dat u de SDK van AIR op de locatie `C:\SDKs\AIR` hebt geplaatst, kunt u de juiste padinvoer met de volgende procedure toevoegen:

1. Open het dialoogvenster Systeemeigenschappen op het Configuratiescherm of door met de rechtermuisknop te klikken op het pictogram Mijn computer en in het menu Eigenschappen te kiezen.

2. Klik op het tabblad Geavanceerd op de knop Omgevingsvariabelen.

3. Selecteer het Pad-element in het gedeelte Systeemvariabelen van het dialoogvenster Omgevingsvariabelen

4 Klik op Bewerken.

5 Scroll naar het einde van de tekst in het veld Waarde van variabele.

6 Voer de volgende tekst in aan het einde van de huidige waarde:

```
;C:\SDKs\AIR\bin
```

7 Klik op OK in alle dialoogvensters om het pad op te slaan.

Let op: als er open opdrachtvensters zijn, wordt de omgeving daarvan niet bijgewerkt. Open een nieuw opdrachtvenster en voer de volgende opdracht in om er zeker van te zijn dat de paden juist zijn ingesteld:

```
adt -version
```

Als u later de locatie van de AIR SDK wijzigt, of een nieuwe versie toevoegt, vergeet dan niet de padvariabele bij te werken.