Erstellen von Adobe AIR-Anwendungen mit dem Packager for iPhone

Rechtliche Hinweise

Rechtliche Hinweise finden Sie unter http://help.adobe.com/de_DE/legalnotices/index.html.

Letzte Aktualisierung 18.5.2011

Inhalt

Kapitel 1: Erste Schritte beim Erstellen von AIR-Anwendungen für das iPhone

Sie können die Adobe® Flash® Platform-Tools und ActionScript® 3.0-Code verwenden, um Adobe® AIR®-Anwendungen für das iPhone und den iPod Touch zu erstellen. Diese Anwendungen werden verteilt, installiert und ausgeführt wie andere iPhone-Anwendungen.

Hinweis: *Im nachfolgenden Text dieses Dokuments werden iPhone und iPod Touch zusammen als „iPhone" bezeichnet.*

Der Packager for iPhone® ist in Adobe® Flash® Professional CS5 enthalten. Der Packager for iPhone kompiliert ActionScript 3.0-Bytecode in nativen iPhone-Anwendungscode. iPhone-Anwendungen („Apps") werden als iPhone-Anwendungsinstallerdateien (.ipa-Dateien) über den iTunes Store vertrieben.

Mit Flash Professional CS5 oder Adobe® Flash® Builder™ 4 können Sie den ActionScript 3.0-Quellinhalt für Ihre Anwendung bearbeiten.

Verwenden Sie Flash Professional CS5, um iPhone-Anwendungen zu entwickeln.

Sie benötigen ebenfalls iPhone-Entwicklerzertifikate von Apple.

Wichtig: Bevor Sie mit der Entwicklung von iPhone-Anwendungen beginnen, lesen Sie die Informationen zum Entwerfen von Anwendungen für das iPhone. Siehe „Überlegungen beim Entwerfen von iPhone-Anwendungen" auf Seite 38. Informieren Sie sich auch über die Entwicklerdateien, die Sie zum Erstellen einer iPhone-Anwendung benötigen. Siehe „Beziehen der Entwicklerdateien von Apple" auf Seite 4.

Wichtige Konzepte

Für die Entwicklung von iPhone-Anwendungen mit ActionScript 3.0 ist das Verständnis der Konzepte und Arbeitsabläufe unerlässlich.

Glossar

Die folgenden Begriffe sind für die Erstellung von iPhone-Anwendungen wichtig.

iPhone Dev Center Die Apple Computer-Website (http://developer.apple.com/iphone/), auf der Sie Folgendes tun können:

- Anmelden als iPhone-Entwickler
- Verwalten und Erstellen von iPhone-Entwicklerzertifikaten, Provisioning-Profilen und App-IDs (nachstehend beschrieben)
- Übermitteln von Anwendungen für den App Store

iPhone-Entwicklerzertifikat Wird verwendet, um einen Entwickler zum Zweck der Entwicklung von Anwendungen zu identifizieren.

Sie können diese Datei von Apple erwerben. Sie konvertieren dieses Zertifikat in eine P12-Zertifikatdatei, um die iPhone-Anwendung, die Sie mit ActionScript 3.0 erstellen, zu signieren. Siehe *P12-Zertifikatdatei*.

Sie benötigen kein iPhone-Entwicklerzertifikat, um Flash Professional CS5-Anwendungen auf dem Entwicklungscomputer zu debuggen und zu testen. Ein Entwicklerzertifikat ist jedoch erforderlich, um die Anwendung auf einem iPhone zu installieren und zu testen.

Das Entwicklerzertifikat unterscheidet sich vom Distributionszertifikat, das Sie zum Erstellen der endgültigen Version Ihrer Anwendung benötigen. Sie beziehen ein Distributionszertifikat von Apple, wenn Sie die endgültige Version Ihrer Anwendung erstellen.

Zertifikatsignaturanforderung Eine Datei mit persönlichen Informationen, die verwendet wird, um ein Entwicklerzertifikat zu generieren. Sie wird auch als CSR-Datei bezeichnet.

Provisioning-Profil Eine Datei, mit der Sie eine iPhone-Anwendung testen oder verteilen können. Sie bekommen die Provisioning-Profildateien von Apple. Ein Provisioning-Profil ist einem bestimmten Entwicklerzertifikat, einer Anwendungs-ID und einer oder mehreren Geräte-IDs zugewiesen. Es gibt verschiedene Arten von Provisioning-Profilen:

- **Entwicklungs-Provisioning-Profil** – Wird verwendet, um eine Testversion einer Anwendung auf dem iPhone des Entwicklers zu installieren.
- **Test-Provisioning-Profil** – Auch als Ad-hoc-Provisioning-Profil bezeichnet. Wird verwendet, um eine Testversion der Anwendung an mehrere Benutzer (und iPhones) weiterzugeben. Mit diesem Provisioning-Profil und der Testanwendung können Benutzer Ihre Anwendung testen, ohne dass diese an den App Store übermittelt wird. Hinweis: Sie können ein Entwicklungs-Provisioning-Profil auch verwenden, um Testanwendungen an mehrere Geräte zu verteilen.
- **Distributions-Provisioning-Profil** – Wird verwendet, um eine iPhone-Anwendung zu erstellen, die an den App Store übermittelt wird.

App-ID Ein eindeutiger String, der eine iPhone-Anwendung (oder mehrere Anwendungen) von einem bestimmten Entwickler identifiziert. Sie erstellen App-IDs im iPhone Dev Center. Jedem Provisioning-Profil ist eine App-ID oder ein App-ID-Muster zugewiesen. Sie benötigen diese App-ID (bzw. das Muster), wenn Sie eine Anwendung entwickeln. Sie verwenden die App-ID im Flash Professional CS5-Dialogfeld „iPhone-Einstellungen“ (oder in der Anwendungsdeskriptordatei).

App-IDs im iPhone Dev Center enthalten eine Bundle-Seed-ID gefolgt von einem Bundle-Bezeichner. Die Bundle-Seed-ID ist eine Zeichenfolge, zum Beispiel 5RM86Z4DJM, die Apple der App-ID zuweist. Der Bundle-Bezeichner enthält eine Zeichenfolge in Form eines umgekehrten Domänennamens, den Sie auswählen. Der Bundle-Bezeichner kann mit einem Sternchen (*) enden, wodurch eine Platzhalter-App-ID angezeigt wird. Beispiele:

- 5RM86Z4DJM.com.example.helloWorld
- 96LPVWEASL.com.example.* (eine Platzhalter-App-ID)

Es gibt zwei Arten von App-IDs im iPhone Dev Center:

- Platzhalter-App-IDs – Im iPhone Dev Center enden diese App-IDs mit einem Sternchen (*), zum Beispiel 96LPVWEASL.com.myDomain.* oder 96LPVWEASL.*. Mit einem Provisioning-Profil, das diese Art von App-ID verwendet, können Sie Testanwendungen entwickeln, die eine App-ID nach diesem Muster verwenden. Für die App-ID der Anwendung können Sie das Sternchen durch eine beliebige Folge gültiger Zeichen ersetzen. Wenn das iPhone Dev Center zum Beispiel „96LPVWEASL.com.example.*“ als App-ID angibt, können Sie „com.example.foo“ oder „com.example.bar“ als App-ID der Anwendung verwenden.

- Spezifische App-ID – Damit wird eine eindeutige App-ID definiert, die in einer Anwendung zu verwenden ist. Im iPhone Dev Center enden diese App-IDs nicht mit einem Sternchen. Ein Beispiel ist 96LPVWEASL.com.myDomain.myApp. Bei einem Provisioning-Profil mit dieser Art App-ID müssen Anwendungen exakt dieselbe App-ID verwenden. Wenn im iPhone Dev Center zum Beispiel „96LPVWEASL.com.example.helloWorld“ als App-ID angegeben ist, müssen Sie „com.example.foo“ als App-ID der Anwendung verwenden.

Wenn Sie Ihre Anwendung entwickeln, geben Sie die App-ID im Dialogfeld „iPhone-Einstellungen“ in Flash Professional CS5 oder in der Anwendungsdeskriptordatei an. Ausführlichere Informationen zu App-IDs finden Sie im Abschnitt „Registerkarte Bereitstellung“ unter „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16 oder unter „Festlegen von iPhone-Anwendungseigenschaften in der Anwendungsdeskriptordatei“ auf Seite 18.

Wichtig: Wenn Sie die App-ID eingeben, ignorieren Sie den Teil mit der Bundle-Seed-ID der App-ID. Wenn Apple Ihre App-ID zum Beispiel als „96LPVWEASL.com.example.bob.myApp“ aufführt, ignorieren Sie 96LPVWEASL und verwenden „com.example.bob.myApp“ als App-ID. Wenn Apple Ihre App-ID als „5RM86Z4DJM.*“ aufführt, ignorieren Sie 5RM86Z4DJM – dies ist eine Platzhalter-App-ID.

Sie finden die mit einem Provisioning-Profil verknüpfte App-ID (oder das Muster für Platzhalter-App-IDs) im iPhone Dev Center (http://developer.apple.com/iphone). Gehen Sie zum Portal des iPhone Developer Program und rufen Sie den Provisioning-Bereich auf.

P12-Zertifikatdatei Eine P12-Datei (eine Datei mit der Erweiterung .p12) ist eine bestimmte Art Zertifikatdatei (eine „Personal Information Exchange“-Datei). Der Packager for iPhone verwendet diesen Zertifikattyp, um eine iPhone-Anwendung zu erstellen. Sie konvertieren das Entwicklerzertifikat, das Sie von Apple erhalten, in diese Zertifikatform.

Eindeutige Geräte-ID Ein eindeutiger Code, der ein bestimmtes iPhone identifiziert. Wird auch als UDID oder Geräte-ID bezeichnet.

Überblick über den Arbeitsablauf bei der Entwicklung

Beim Entwickeln einer Anwendung für das iPhone führen Sie die folgenden Schritte aus:

1. Installieren Sie Flash Professional CS5 von Adobe.
2. Installieren Sie iTunes.
3. Beziehen Sie Entwicklerdateien von Apple. Zu diesen Dateien gehören das Entwicklerzertifikat und Provisioning-Profile. Siehe „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.
4. Konvertieren Sie das Entwicklerzertifikat in eine P12-Zertifikatdatei. Für Flash CS5 ist es erforderlich, dass es sich bei dem Zertifikat um ein P12-Zertifikat handelt. Siehe „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.
5. Verwenden Sie iTunes, um Ihr Provisioning-Profil Ihrem iPhone zuzuordnen.
6. Schreiben Sie die Anwendung in Flash Professional CS5.

   Es ist wichtig, dass Sie sich mit den bewährten Verfahren zum Entwerfen und Optimieren von Code für iPhone-Anwendungen vertraut machen. Siehe „Überlegungen beim Entwerfen von iPhone-Anwendungen“ auf Seite 38.

   Außerdem gilt zu beachten, dass einige ActionScript 3.0-APIs auf dem iPhone eingeschränkt oder gar nicht unterstützt werden. Siehe „ActionScript 3.0-API-Unterstützung für mobile Geräte“ auf Seite 30.

   Sie können den ActionScript 3.0-Code für die Anwendung auch mit Flash Builder bearbeiten.

   Mit Flash Professional CS5 können Sie die Anwendung auf dem Entwicklungscomputer testen.

7 Erstellen Sie Symbole und Grafiken für den Anfangsbildschirm der Anwendung. Jede iPhone-Anwendung enthält verschiedene Symbole, mit denen sie für den Benutzer kenntlich gemacht wird. Das iPhone zeigt den Anfangsbildschirm an, wenn das Programm geladen wird. Siehe „iPhone-Symbole und Bilder für den Startbildschirm“ auf Seite 14.

8 Bearbeiten Sie die iPhone-Einstellungen. Dazu zählen die folgenden Einstellungen:

- Die Identität der Anwendung (einschließlich Dateiname, Anwendungsname, Versionsnummer und App-ID)
- Der Speicherort der Symbolgrafik für die Anwendung
- Das P12-Zertifikat und das Provisioning-Profil, die der Anwendung zugeordnet sind
- Das anfängliche Seitenverhältnis der Anwendung

In Flash Professional CS5 können Sie diese Einstellungen im Dialogfeld „iPhone-Einstellungen“ bearbeiten. Weitere Informationen finden Sie unter „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16.

Sie können diese Einstellungen aber auch direkt in der Anwendungsdeskriptordatei bearbeiten. Weitere Informationen finden Sie unter „Festlegen von iPhone-Anwendungseigenschaften in der Anwendungsdeskriptordatei“ auf Seite 18.

9 Kompilieren Sie die IPA-Datei mit dem Packager for iPhone. Siehe „Kompilieren von Installerdateien für iPhone-Anwendungen (IPA-Dateien)“ auf Seite 22.

10 Installieren und testen Sie die Anwendung auf Ihrem iPhone. Verwenden Sie iTunes, um die IPA-Datei zu installieren.

Für die Ad-hoc-Verteilung wiederholen Sie diesen allgemeinen Prozess, verwenden Sie dabei jedoch ein Test-Provisioning-Profil anstatt eines Entwicklungs-Provisioning-Profils. Für die endgültige Verteilung der Anwendung wiederholen Sie diesen Prozess; verwenden Sie dabei jedoch das Distributions-Provisioning-Profil. (Informationen zu den verschiedenen Arten von Provisioning-Profilen finden Sie im „Glossar“ auf Seite 1.)

Wenn Sie eine Distributionsversion Ihrer Anwendung erstellt haben, lesen Sie die Anweisungen unter „Übermitteln von iPhone-Anwendungen an den App Store“ auf Seite 28.

Ein schnelles Tutorial zum Erstellen einer einfachen iPhone-Anwendung finden Sie unter „Erstellen der iPhone-Anwendung „Hello World“ mit Flash Professional CS5“ auf Seite 8.

Beziehen der Entwicklertools von Adobe

Zum Entwickeln von iPhone-Anwendungen mit ActionScript 3.0 benötigen Sie Flash Professional CS5.

Wichtig: Aktualisieren Sie die Preview-Version von Packager for iPhone, die in Flash Professional CS5 enthalten ist. Wählen Sie in Flash Professional CS5 „Hilfe“ > „Updates“.

Sie können ActionScript-Code auch mit Flash Builder 4 bearbeiten. Flash Builder 4 ist erhältlich unter http://www.adobe.com/de/products/flashbuilder/.

Beziehen der Entwicklerdateien von Apple

Wie beim Entwickeln jeder Anwendung für das iPhone müssen Sie zunächst iPhone-Entwicklerdateien von Apple erwerben. Sie benötigen ein iPhone-Entwicklerzertifikat und ein Mobil-Provisioning-Profil. Außerdem brauchen Sie noch weitere Provisioning-Dateien. Eine Definition dieser Dateien finden Sie im „Glossar“ auf Seite 1.

Hinweis: *Der Bezug dieser Dateien ist ein wichtiger Teil des Anwendungsentwicklungsprozesses. Diesen Schritt müssen Sie abschließen, bevor Sie Ihre Anwendung entwickeln. Das Beziehen der Entwicklerdateien ist kein einfacher Vorgang. Lesen Sie die vorliegenden Anweisungen und die Anweisungen im Apple iPhone Dev Center aufmerksam durch.*

Erwerben und Verwenden von iPhone-Entwicklerdateien

Sie benötigen ein iPhone-Entwicklerzertifikat und Provisioning-Profile von Apple. Außerdem müssen Sie das Zertifikat in ein P12-Zertifikat konvertieren.

Installieren von iTunes

Sie benötigen iTunes, damit Sie Ihre Anwendung auf Ihrem iPhone installieren können. Mithilfe von iTunes finden Sie außerdem die Geräte-ID Ihres iPhones heraus. Diese Geräte-ID brauchen Sie, wenn Sie ein iPhone-Entwicklerzertifikat anfordern.

Anfordern von iPhone-Entwicklerzertifikaten und Erstellen eines Provisioning-Profils

Falls Sie dies noch nicht getan haben, lassen Sie sich beim Apple iPhone Dev Center (http://developer.apple.com/iphone/) als iPhone-Entwickler registrieren.

Hinweis: *Das iPhone SDK und XCode sind nicht erforderlich, um AIR-Anwendungen für das iPhone zu entwickeln. Sie brauchen kein registrierter iPhone-Entwickler zu sein. Sie benötigen ein Entwicklerzertifikat und ein Provisioning-Profil.*

1. Melden Sie sich mit Ihrer iPhone-Entwicklerkonto-ID beim iPhone Dev Center an.
2. Fordern Sie beim iPhone Dev Center ein iPhone-Entwicklerzertifikat an und bezahlen Sie dieses.

   Sie erhalten eine E-Mail mit Ihrem Aktivierungscode für das iPhone Developer Program von Apple.
3. Kehren Sie zum iPhone Dev Center zurück. Folgen Sie den Anweisungen zum Aktivieren Ihres Entwicklerprogramms und geben Sie den Aktivierungscode ein, wenn Sie dazu aufgefordert werden.
4. Nachdem Ihr Aktivierungscode angenommen wurde, gehen Sie zum Abschnitt „iPhone Developer Program Portal“ des iPhone Dev Center.
5. Erstellen Sie eine Zertifikatsignaturanforderungsdatei. Mit dieser Datei können Sie ein iPhone-Entwicklerzertifikat beziehen. Anweisungen hierzu finden Sie unter „Generieren einer Zertifikatsignaturanforderung“ auf Seite 6.
6. Im nächsten Schritt werden Sie aufgefordert, die Geräte-ID (Unique Device ID, UDID) Ihres iPhones einzugeben. Sie finden die UDID in iTunes:
   a. Schließen Sie ein USB-Kabel an das iPhone an. Wählen Sie in iTunes den Bereich „Übersicht“ für das iPhone.
   b. Nachdem Sie das Provisioning-Profil von der iPhone Developer Center-Site heruntergeladen haben, fügen Sie es iTunes hinzu.
   c. Klicken Sie dann auf die angezeigte Seriennummer. Die UDID wird angezeigt. Drücken Sie Befehlstaste+C (Mac) bzw. Strg+C (Windows), um die UDID in die Zwischenablage zu kopieren.
7. Erstellen und installieren Sie ein Provisioning-Profil und ein iPhone-Entwicklerzertifikat.

   Folgen Sie den Anweisungen im iPhone Dev Center. Suchen Sie nach Anweisungen im Abschnitt „iPhone Developer Program Portal“. Sie können den Development Provisioning Assistant verwenden, um Ihr Entwicklerzertifikat zu beziehen und Ihr Provisioning-Profil zu erstellen.

   Ignorieren Sie Schritte, die mit XCode zu tun haben. Sie müssen keinen XCode verwenden, um mit Flash Professional CS5 iPhone-Anwendungen zu entwickeln.
8. Wählen Sie in iTunes „Ablage“ > „Zu Mediathek hinzufügen“. Wählen Sie dann die Provisioning-Profildatei aus (die Datei hat die Dateinamenerweiterung „mobileprovision“). Synchronisieren Sie Ihr iPhone mit iTunes.

Damit können Sie die Anwendung, die diesem Provisioning-Profil zugeordnet ist, auf Ihrem iPhone testen.

Um zu überprüfen, ob ein bestimmtes Provisioning-Profil iTunes bereits hinzugefügt wurde, versuchen Sie es der Mediathek hinzuzufügen. Wenn Sie von iTunes gefragt werden, ob ein vorhandenes Provisioning-Profil ersetzt werden soll, können Sie die Schaltfläche „Abbrechen“ wählen. (Das Profil wurde bereits installiert.) Sie können auch überprüfen, welche Provisioning-Profile auf dem iPhone installiert sind:

a Öffnen Sie die Anwendung „Einstellungen“ auf dem iPhone.

b Wählen Sie die Kategorie „Allgemein“ aus.

c Tippen Sie auf „Profile“. Auf der Seite „Profile“ werden die installierten Provisioning-Profile aufgeführt.

9 Laden Sie die iPhone-Entwicklerzertifikatdatei (.cer) herunter, falls noch nicht geschehen.

Der Development Provisioning Assistant hat unter Umständen einen Link zum Herunterladen dieser Datei angegeben. Sie finden die Datei auch im Abschnitt „Certificates“ des Provisioning Portal im Apple iPhone Dev Center unter (http://developer.apple.com/iphone/).

10 Als Nächstes konvertieren Sie das iPhone-Entwicklerzertifikat in eine P12-Datei. Anweisungen hierzu finden Sie unter „Konvertieren eines Entwicklerzertifikats in eine P12-Datei“ auf Seite 7.

Sie können jetzt eine einfache „Hello World“-Anwendung erstellen. Siehe „Erstellen der iPhone-Anwendung „Hello World“ mit Flash Professional CS5“ auf Seite 8.

Generieren einer Zertifikatsignaturanforderung

Um ein Entwicklerzertifikat zu beziehen, generieren Sie eine Zertifikatsignaturanforderungsdatei, die Sie an das Apple iPhone Dev Center übermitteln.

Generieren einer Zertifikatsignaturanforderung unter Mac OS

Wenn Sie mit Mac OS arbeiten, können Sie die Schlüsselbundverwaltung verwenden, um eine Codesignaturanforderung zu generieren. Sie finden die Schlüsselbundverwaltung im Dienstprogramm-Unterverzeichnis unter „Programme“. Wählen Sie unter Schlüsselbundverwaltung „Zertifikat-Assistent“ > „Zertifikat von einer Zertifizierungsstelle anfordern“.

1 Öffnen Sie die Schlüsselbundverwaltung.

2 Wählen Sie im Menü „Schlüsselbundverwaltung“ den Eintrag „Voreinstellungen“.

3 Klicken Sie im Dialogfeld „Voreinstellungen“ auf „Zertifikate“. Setzen Sie dann „OCSP-Protokoll (Online Certificate Status)“ und „Liste für Zertifikatswiderrufung“ auf „Aus“. Schließen Sie das Dialogfeld.

4 Wählen Sie unter Schlüsselbundverwaltung „Zertifikat-Assistent“ > „Zertifikat von einer Zertifizierungsstelle anfordern“.

5 Geben Sie die E-Mail-Adresse und den Namen ein, der Ihrer iPhone-Entwicklerkonto-ID entspricht. Geben Sie keine E-Mail-Adresse der Zertifizierungsstelle (CA) ein. Wählen Sie „Request is Saved to Disk“ (Anforderung ist auf Festplatte gespeichert) und klicken Sie auf „Fortfahren“.

6 Speichern Sie die Datei (CertificateSigningRequest.certSigningRequest).

7 Laden Sie die CSR-Datei auf der iPhone Entwickler-Site an Apple hoch. (Siehe „Anfordern von iPhone-Entwicklerzertifikaten und Erstellen eines Provisioning-Profils“.)

Generieren einer Zertifikatsignaturanforderung unter Windows

Für Windows-Entwickler ist es vielleicht am einfachsten, das iPhone-Entwicklerzertifikat über einen Mac-Computer zu beziehen. Es ist jedoch auch möglich, für den Erwerb eines Zertifikats einen Windows-Computer zu benutzen. Erstellen Sie zunächst eine Zertifikatsignaturanforderung (eine CSR-Datei) mit OpenSSL:

1. Installieren Sie OpenSSL auf dem Windows-Computer. (Gehen Sie zu http://www.openssl.org/related/binaries.html.)

   Möglicherweise müssen Sie auch die Dateien für Visual C++ 2008 Redistributable installieren, die auf der Open SSL-Downloadseite aufgeführt sind. (Sie brauchen jedoch *nicht* Visual C++ auf Ihrem Computer zu installieren.)

2. Öffnen Sie ein Windows-Befehlseingabefenster und wechseln Sie zum Open-SSL-bin-Verzeichnis (zum Beispiel c:\OpenSSL\bin\).

3. Erstellen Sie den privaten Schlüssel, indem Sie Folgendes in die Befehlszeile eingeben:

   ```
   openssl genrsa -out mykey.key 2048
   ```

   Speichern Sie diese private Schlüsseldatei. Sie benötigen sie später.

   Wenn Sie OpenSSL verwenden, ignorieren Sie etwaige Fehlermeldungen nicht. Wenn OpenSSL eine Fehlermeldung generiert, werden möglicherweise dennoch Dateien ausgegeben. Diese Dateien sind jedoch nicht verwendbar. Wenn Fehler auftreten, überprüfen Sie die Syntax und führen Sie den Befehl erneut aus.

4. Erstellen Sie die CSR-Datei, indem Sie Folgendes in die Befehlszeile eingeben:

   ```
   openssl req -new -key mykey.key -out CertificateSigningRequest.certSigningRequest  -subj
   "/emailAddress=yourAddress@example.com, CN=John Doe, C=US"
   ```

   Ersetzen Sie die E-Mail-Adresse, CN (Zertifikatname) und C (Land) durch die für Sie geltenden Werte.

5. Laden Sie die CSR-Datei auf der iPhone Entwickler-Site an Apple hoch. (Siehe „Anfordern von iPhone-Entwicklerzertifikaten und Erstellen eines Provisioning-Profils“.)

Konvertieren eines Entwicklerzertifikats in eine P12-Datei

Um iPhone-Anwendungen mit Flash Professional CS5 zu entwickeln, müssen Sie eine P12-Zertifikatdatei verwenden. Sie generieren dieses Zertifikat basierend auf der Apple iPhone-Entwicklerzertifikatdatei, die Sie von Apple erhalten.

Konvertieren von iPhone-Entwicklerzertifikaten in P12-Dateien unter Mac OS

Nachdem Sie das Apple-iPhone-Zertifikat von Apple heruntergeladen haben, exportieren Sie es in das P12-Zertifikatformat. Unter Mac® OS gehen Sie dazu folgendermaßen vor:

1. Öffnen Sie die Schlüsselbundverwaltung (unter „Programme/Dienstprogramme“).
2. Wenn Sie das Zertifikat noch nicht dem Schlüsselbund hinzugefügt haben, wählen Sie „Ablage“ > „Importieren“. Navigieren Sie dann zu der Zertifikatsdatei (.cer-Datei), die Sie von Apple erhalten haben.
3. Wählen Sie in der Schlüsselbundverwaltung die Kategorie „Schlüssel“.
4. Wählen Sie den privaten Schlüssel aus, der Ihrem iPhone-Entwicklerzertifikat zugeordnet ist.

   Der private Schlüssel wird durch das öffentliche Zertifikat iPhone-Entwickler: <Vorname> <Nachname> identifiziert, das mit ihm verknüpft ist.

5. Wählen Sie „Ablage“ > „Objekte exportieren“.
6. Speichern Sie den Schlüssel im Dateiformat Personal Information Exchange (.p12).
7. Sie werden aufgefordert, ein Kennwort zu erstellen, das verwendet wird, wenn Sie versuchen, diesen Schlüssel auf einem anderen Computer zu importieren.

Konvertieren von Apple-Entwicklerzertifikaten in P12-Dateien unter Windows

Um iPhone-Anwendungen mit Flash CS5 zu entwickeln, müssen Sie eine P12-Zertifikatdatei verwenden. Sie generieren dieses Zertifikat basierend auf der Apple iPhone-Entwicklerzertifikatdatei, die Sie von Apple erhalten.

1. Konvertieren Sie die Entwicklerzertifikatdatei von Apple in eine PEM-Zertifikatdatei. Führen Sie die folgenden Befehlszeilenanweisungen aus dem OpenSSL-Verzeichnis „bin“ aus:

   ```
   openssl x509 -in developer_identity.cer -inform DER -out developer_identity.pem -outform PEM
   ```

2. Wenn Sie den privaten Schlüssel aus der Schlüsselbundverwaltung auf einem Mac-Computer verwenden, konvertieren Sie ihn in einen PEM-Schlüssel:

   ```
   openssl pkcs12 -nocerts -in mykey.p12 -out mykey.pem
   ```

3. Sie können jetzt eine gültige P12-Datei generieren, die auf dem Schlüssel und der PEM-Version des iPhone-Entwicklerzertifikats basiert:

   ```
   openssl pkcs12 -export -inkey mykey.key -in developer_identity.pem -out iphone_dev.p12
   ```

   Wenn Sie einen Schlüssel aus der Mac OS-Schlüsselbundverwaltung benutzen, verwenden Sie die PE-Version, die Sie im vorherigen Schritt generiert haben. Andernfalls verwenden Sie den OpenSSL-Schlüssel, den Sie zuvor erstellt haben (unter Windows).

Verwalten von Zertifikaten, Geräte-IDs, App-IDs und Provisioning-Profilen

Sie können Zertifikate, Geräte-IDs, App-IDs und Provisioning-Profile im Apple iPhone Dev Center verwalten (http://developer.apple.com/iphone/). Gehen Sie zum Bereich „iPhone Developer Program Portal“ der Site.

- Klicken Sie auf den Hyperlink „Certificates“, um Ihre Entwicklerzertifikate zu verwalten. Sie können Zertifikate erstellen, herunterladen oder widerrufen. Um ein Zertifikat zu erstellen, müssen Sie zunächst eine Zertifikatsignaturanforderung erstellen. Siehe „Generieren einer Zertifikatsignaturanforderung“ auf Seite 6.
- Klicken Sie auf den Hyperlink „Geräte“, um die Liste der Geräte zu verwalten, auf denen Ihre Entwicklungsanwendung installiert werden kann.
- Klicken Sie auf den Hyperlink „App IDs“, um Ihre App-IDs zu verwalten. Wenn Sie ein Provisioning-Profil erstellen, ist es mit einer App-ID verknüpft.
- Klicken Sie auf den Hyperlink „Provisioning“, um Ihre Provisioning-Profile zu verwalten. Sie können Provisioning-Profile für die Entwicklung auch mithilfe des „Development Provisioning Assistant“ erstellen.
- Klicken Sie auf den Hyperlink „Distribution“, wenn Sie Ihre Anwendung an den App Store senden oder eine Ad-hoc-Version Ihrer Anwendung erstellen möchten. In diesem Abschnitt finden Sie einen Hyperlink zur Site „iTunes Connect“, über die Sie eine Anwendung an den App Store senden können.

Erstellen der iPhone-Anwendung „Hello World“ mit Flash Professional CS5

Wichtig: Bevor Sie die Anwendung erstellen, laden Sie die erforderlichen Entwickleranwendungen und -dateien herunter. Siehe „Beziehen der Entwicklertools von Adobe“ auf Seite 4 und „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.

Erstellen eines Flash Professional CS5-Projekts

Sie können eine iPhone-Anwendung direkt in Flash Professional CS5 erstellen:

1 Öffnen Sie Flash CS5.

2 Wählen Sie „Datei“ > „Neu“.

3 Wählen Sie „iPhone“.

4 Klicken Sie auf „OK“.

Hinzufügen von Inhalt zur Anwendung

Fügen Sie der Anwendung als Nächstes den Text „Hello world!“ hinzu.

1 Wählen Sie das Textwerkzeug aus und klicken Sie auf die Bühne.

2 Wählen Sie in den Eigenschaften für das Textfeld die Option „Klassischer Text“ (nicht: „TLF-Text“).

Da es sich um eine einfache Anwendung handelt, ist klassischer Text angemessen. Um TLF-Text zu verwenden, müssten Sie zusätzliche Einstellungen vornehmen. Siehe „Schriftarten und Texteingabe“ auf Seite 41.

3 Geben Sie in das neue Textfeld den Text „Hello World!“ ein.

4 Wählen Sie das Textfeld mit dem Auswahlwerkzeug aus.

5 Öffnen Sie dann den Eigenschafteninspektor und nehmen Sie die folgenden Einstellungen vor:

- Zeichen > Familie: _sans
- Zeichen > Größe: 50
- Position > X: 20
- Position > Y: 20

6 Speichern Sie die Datei.

7 Wählen Sie „Steuerung“ > „Film testen“ > „In AIR Debug Launcher (mobil)“.

Flash Professional CS5 kompiliert den SWF-Inhalt und zeigt eine Version der Anwendung im AIR Debug Launcher (ADL) an. Auf diese Weise können Sie schnell eine Vorschau der Anwendung sehen.

Erstellen von Symbolen und Grafiken für den Anfangsbildschirm der Anwendung

Alle iPhone-Anwendungen haben Symbole, die in der iTunes-Benutzeroberfläche und auf dem iPhone-Bildschirm angezeigt werden.

1 Erstellen Sie im Projektverzeichnis ein Verzeichnis und nennen Sie es „icons“.

2 Erstellen Sie drei PNG-Dateien im Verzeichnis „icons“. Nennen Sie diese Dateien „Icon29.png“, „Icon57.png“ und „Icon512.png“.

3 Bearbeiten Sie die PNG-Dateien, um geeignete Grafiken für Ihre Anwendung zu erstellen. Die Dateien müssen 29 x 29, 57 x 57 bzw. 512 x 512 Pixel groß sein. Für diesen Test können Sie einfach einfarbige Quadrate als Symbole verwenden.

Alle iPhone-Anwendungen zeigen ein Anfangsbild an, wenn die Anwendung auf dem iPhone geladen wird. Sie definieren das Anfangsbild in einer PNG-Datei:

1 Erstellen Sie im Hauptentwicklungsverzeichnis eine PNG-Datei mit dem Namen „Default.png“. (Legen Sie diese Datei *nicht* im Unterverzeichnis „icons“ ab. Achten Sie darauf, den Dateinamen, „Default.png“, mit einem Großbuchstaben-D einzugeben.)

2 Bearbeiten Sie die Datei, sodass sie 320 Pixel breit und 480 Pixel hoch ist. Für den Moment kann es sich bei dem Inhalt um ein einfaches weißes Rechteck handeln. (Dies wird später geändert.)

Hinweis: *Wenn Sie eine Anwendung an den Apple App Store übermitteln, verwenden Sie eine JPG-Version (keine PNG-Version) der 512-Pixel-Datei. Sie verwenden die PNG-Version beim Testen von Entwicklungsversionen der Anwendung.*

Ausführliche Informationen zu diesen Grafiken finden Sie unter „iPhone-Symbole und Bilder für den Startbildschirm“ auf Seite 14.

Bearbeiten der Anwendungseinstellungen

Wichtig: Falls Sie dies noch nicht getan haben, laden Sie die erforderlichen Entwickleranwendungen und -dateien für die iPhone-Entwicklung herunter. Siehe „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.

Im Flash Professional CS5-Dialogfeld „iPhone-Einstellungen“ können Sie viele grundlegende Eigenschaften der iPhone-Anwendung definieren.

1 Wählen Sie „Datei“ > „iPhone-Einstellungen“.

2 Nehmen Sie auf der Registerkarte „Allgemein“ die folgenden Einstellungen vor:

- Ausgabedatei: HelloWorld.ipa

  Dies ist der Dateiname der iPhone-Installerdatei, die generiert werden soll.

- App-Name: Hello World

  Dies ist der Name der Anwendung, der unter dem Anwendungssymbol auf dem iPhone angezeigt wird.

- Version: 1.0

  Die Version der Anwendung.

- Seitenverhältnis: Hochformat
- Vollbild: ausgewählt
- Automatische Ausrichtung: nicht ausgewählt
- Rendering: CPU

  Die anderen Optionen, „GPU“ und „Auto“, verwenden Hardwarebeschleunigung für das Rendering. Diese Funktion kann die Leistung von grafikintensiven Anwendungen (zum Beispiel Spiele), für die die Nutzung der Hardwarebeschleunigung vorgesehen ist, verbessern. Weitere Informationen finden Sie unter „Hardwarebeschleunigung“ auf Seite 38.

- Enthaltene Dateien: Fügen Sie die Datei mit der Anfangsgrafik (Default.pgn) zur Liste der enthaltenen Dateien hinzu.

Hinweis: *Für dieses „Hello World“-Beispiel verwenden Sie bitte die in den Anweisungen angegebenen Einstellungen. Für einige Einstellungen, zum Beispiel die Version, gelten bestimmte Einschränkungen. Diese Einschränkungen sind unter „iPhone-Anwendungseinstellungen“ auf Seite 16 beschrieben.*

3 Nehmen Sie auf der Registerkarte „Bereitstellung“ die folgenden Einstellungen vor:

- Zertifikat: Wählen Sie das .p12-Zertifikat aus, das auf dem von Apple erworbenen Entwicklerzertifikat basiert.

  Mit diesem Zertifikat wird die Datei signiert. Sie müssen das Apple iPhone-Zertifikat in das .p12-Format konvertieren. Weitere Informationen finden Sie unter „Beziehen der Entwicklertools von Adobe“ auf Seite 4.

- Kennwort: Geben Sie das Kennwort für das Zertifikat ein.
- Provisioning-Datei: Wählen Sie die Entwickler-Provisioning-Datei aus, die Sie von Apple erhalten haben. Siehe „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.
- App-ID: Falls dieses Feld ausgewählt werden kann, können Sie eine Anwendungs-ID eingeben, die der Anwendungs-ID entspricht, die Sie Apple angegeben haben (zum Beispiel com.example.as3.HelloWorld).

  Die Anwendungs-ID identifiziert Ihre Anwendung eindeutig.

  Ist das Feld nicht auswählbar, ist das Provisioning-Profil an eine bestimmte Anwendungs-ID gebunden. Die Anwendungs-ID wird im Feld angezeigt.

  Ausführliche Informationen zum Festlegen einer Anwendungs-ID finden Sie im Abschnitt zur Registerkarte „Bereitstellung“ unter „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16.

4 Klicken Sie auf der Registerkarte in der Liste „Symbole“ auf das Symbol „29x29“. Geben Sie dann den Speicherort der Datei „Icon29.png“ an, die Sie zuvor erstellt haben (siehe „Erstellen von Symbolen und Grafiken für den Anfangsbildschirm der Anwendung“ auf Seite 9). Geben Sie dann die entsprechenden PNG-Dateien für die Symbole der Größe 57 x 57 und 512 x 512 an.

5 Klicken Sie auf „OK“.

6 Speichern Sie die Datei.

Nähere Informationen zu den Anwendungseinstellungen finden Sie unter „iPhone-Anwendungseinstellungen“ auf Seite 16.

Kompilieren der IPA-Datei

Sie können jetzt die IPA-Installerdatei kompilieren:

1 Wählen Sie „Datei“ > „Veröffentlichen“.

2 Klicken Sie im Dialogfeld „iPhone-Einstellungen“ auf „OK“.

Der Packager for iPhone generiert die Installerdatei für die iPhone-Anwendung, „HelloWorld.ipa“, im Projektverzeichnis. Das Kompilieren der IPA-Datei kann einige Minuten dauern.

Installieren der Anwendung auf dem iPhone

So installieren Sie die iPhone-Anwendung zu Testzwecken auf einem iPhone:

1 Öffnen Sie die iTunes-Anwendung.

2 Fügen Sie das Provisioning-Profil für diese Anwendung zu iTunes hinzu. Wählen Sie in iTunes „Ablage“ > „Zu Mediathek hinzufügen“. Wählen Sie dann die Provisioning-Profildatei aus (die Datei hat den Dateityp „mobileprovision“).

   Für den Moment verwenden Sie das Entwicklungs-Provisioning-Profil, um die Anwendung auf dem Entwickler-iPhone zu testen.

Später, wenn Sie eine Anwendung in den iTunes Store stellen, verwenden Sie das Distributionsprofil. Um die Anwendung ad hoc zu verteilen (an mehrere Geräte, ohne den iTunes Store zu verwenden), benutzen Sie das Ad-hoc-Provisioning-Profil.

Weitere Informationen zu Provisioning-Profilen finden Sie unter „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.

3 Einige iTunes-Versionen ersetzen die Anwendung nicht, wenn dieselbe Version der Anwendung bereits installiert ist. Löschen Sie in diesem Fall Ihre Anwendung vom Gerät und aus der Liste der Anwendungen in iTunes.

4 Doppelklicken Sie auf die IPA-Datei für Ihre Anwendung. Sie sollte in der Liste der Anwendungen aufgeführt werden.

5 Verbinden Sie das iPhone mit einem USB-Anschluss des Computers.

6 Überprüfen Sie in iTunes die Registerkarte „Programme“ und vergewissern Sie sich, dass die Anwendung in der Liste der zu installierenden Anwendungen ausgewählt ist.

7 Wählen Sie links in der iTunes-Anwendung das Gerät aus. Klicken Sie dann auf „Synchronisieren“. Nach Abschluss der Synchronisierung erscheint die Anwendung „Hello World“ auf Ihrem iPhone.

Wenn die neue Version nicht installiert wurde, löschen Sie sie vom iPhone und aus der Liste der Anwendungen in iTunes und wiederholen Sie den Vorgang. Dies kann vorkommen, wenn die zurzeit installierte Version dieselbe Anwendungs-ID und Version verwendet.

Wenn iTunes beim Versuch, die Anwendung zu installieren, einen Fehler anzeigt, lesen Sie den Abschnitt „Fehlerbehebung bei Problemen mit der Anwendungsinstallation“ unter „Installieren von iPhone-Anwendungen“ auf Seite 24.

Bearbeiten des Anfangsbildschirms

Vor dem Kompilieren der Anwendung haben Sie die Datei „Default.png“ erstellt (siehe „Erstellen von Symbolen und Grafiken für den Anfangsbildschirm der Anwendung“ auf Seite 9). Diese PNG-Datei dient als Startbild, während die Anwendung geladen wird. Beim Testen der Anwendung auf dem iPhone haben Sie möglicherweise den leeren Bildschirm beim Starten bemerkt.

Diesen sollten Sie ändern, damit er zum Startbildschirm Ihrer Anwendung („Hello World!“) passt:

1 Öffnen Sie die Anwendung auf Ihrem Gerät. Wenn der erste „Hello World!“-Text erscheint, drücken Sie die Home-Taste (unter dem Bildschirm) und halten Sie sie gedrückt. Währen Sie die Home-Taste gedrückt halten, drücken Sie auf die Power/Standby-Taste oben auf dem iPhone. Damit wird ein Bildschirmfoto erstellt und an die Camera Roll gesendet.

2 Übertragen Sie das Bild auf Ihren Entwicklungscomputer, indem Sie iPhoto oder eine andere Anwendung zum Übertragen von Fotos verwenden. (Auf dem Mac können Sie auch die Bilderfassung verwenden.)

Sie können das Foto auch per E-Mail an den Entwicklungscomputer senden.

- Öffnen Sie die Fotos-Anwendung.
- Öffnen Sie die Camera Roll.
- Öffnen Sie das Bildschirmfoto, das Sie aufgenommen haben.
- Tippen Sie auf das Bild und dann auf den Vorwärtspfeil unten links. Klicken Sie dann auf „Per E-Mail senden“ und senden Sie das Bild an sich selbst.

3 Ersetzen Sie die Datei „Default.png“ (in Ihrem Entwicklungsverzeichnis) durch eine PNG-Version des Bildschirmfotos.

4 Kompilieren Sie die Anwendung erneut (siehe „Kompilieren der IPA-Datei“ auf Seite 11) und installieren Sie sie erneut auf dem iPhone.

Die Anwendung verwendet beim Laden jetzt den neuen Startbildschirm.

Hinweis: *Sie können beliebige Grafiken für die Datei „Default.png“ verwenden, solange sie die richtigen Abmessungen (320 x 480 Pixel) hat. Häufig ist es jedoch am besten, wenn das Bild in „Default.png“ mit dem Anfangszustand der Anwendung übereinstimmt.*

Kapitel 2: Kompilieren und Debuggen von iPhone-Anwendungen

Sie können eine iPhone-Anwendung mit dem Packager for iPhone kompilieren. Der Packager for iPhone ist in Flash Professional CS5 enthalten.

Das Debugging der Anwendung erfolgt auf dem Entwicklungscomputer. Sie können auch eine Debug-Version auf dem iPhone installieren und die `trace`-Ausgabe in Flash Professional empfangen.

Ein Tutorial zum Erstellen einer iPhone-Anwendung von Anfang bis Ende finden Sie unter „Erstellen der iPhone-Anwendung „Hello World“ mit Flash Professional CS5“ auf Seite 8.

iPhone-Symbole und Bilder für den Startbildschirm

Alle iPhone-Anwendungen haben Symbole, die in der iTunes-Benutzeroberfläche und auf dem iPhone angezeigt werden.

iPhone-Anwendungssymbole

Sie definieren die folgenden Symbole für eine iPhone-Anwendung:

- Ein Symbol mit 29 mal 29 Pixel – Dieses Symbol wird in Spotlight-Suchergebnissen auf dem iPhone und dem iPod touch verwendet.
- Ein Symbol mit 48 mal 48 Pixel – Dieses Symbol wird in Spotlight-Suchergebnissen auf dem iPad verwendet.
- Ein Symbol mit 57 mal 57 Pixel – Dieses Symbol wird in den Startbildschirmen von iPhone und iPod touch verwendet.
- Ein Symbol mit 72 mal 72 Pixel – Dieses Symbol wird im Startbildschirm des iPad verwendet.
- Ein Symbol mit 512 mal 512 Pixeln – Dieses Symbol wird von iTunes verwendet. Die 512-Pixel-PNG-Datei wird nur zum Testen von Entwicklungsversionen Ihrer Anwendung verwendet. Wenn Sie die endgültige Anwendung in den Apple App Store stellen, übermitteln Sie das 512-Bild separat als JPG-Datei. Es ist nicht in der IPA-Datei enthalten.

In Flash Professional CS5 fügen Sie diese Symbole im Dialogfeld „iPhone-Einstellungen“ auf der Registerkarte „Symbole“ hinzu. Siehe „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16.

Sie können die Speicherorte der Symbole auch der Anwendungsdeskriptordatei hinzufügen:

```
<icon>
        <image29x29>icons/icon29.png</image29x29>
        <image57x57>icons/icon57.png</image57x57>
        <image72x72>icons/icon72.png</image72x72>
        <image512x512>icons/icon512.png</image512x512>
</icon>
```

Das iPhone fügt dem Symbol einen Glanzeffekt hinzu. Sie brauchen diesen Effekt nicht in Ihr Quellbild einzuschließen. Um diesen standardmäßigen Glanzeffekt zu entfernen, fügen Sie dem `InfoAdditions`-Element in der Anwendungsdeskriptordatei Folgendes hinzu:

```
<InfoAdditions>
    <![CDATA[
        <key>UIPrerenderedIcon</key>
        <true/>
    ]]>
</InfoAdditions>
```

Siehe „Festlegen von iPhone-Anwendungseigenschaften in der Anwendungsdeskriptordatei“ auf Seite 18.

Der Anfangsbildschirm (Default.png)

Alle iPhone-Anwendungen zeigen ein Anfangsbild an, wenn die Anwendung auf dem iPhone geladen wird. Sie definieren das Anfangsbild in einer PNG-Datei mit dem Namen „Default.png“. Erstellen Sie im Hauptentwicklungsverzeichnis eine PNG-Datei mit dem Namen „Default.png“. (Legen Sie diese Datei *nicht* in einem Unterverzeichnis ab. Achten Sie darauf, den Dateinamen, „Default.png“, mit einem Großbuchstaben-D einzugeben.)

Die Datei „Default.png“ ist 320 Pixel breit und 480 Pixel hoch, unabhängig von der anfänglichen Ausrichtung der Anwendung und davon, ob sie im Vollbild angezeigt wird oder nicht.

Wenn Ihre Anwendung ursprünglich im Querformat ausgerichtet ist, verwenden Sie dieselben Abmessungen, die auch eine Hochformatanwendung verwendet: 320 Pixel breit und 480 Pixel breit. Drehen Sie die Grafik in der PNG-Datei jedoch um 90° gegen den Uhrzeigersinn. Die linke Seite des PNG-Bildmaterials entspricht dem oberen Rand des iPhone-Bildschirms im Querformatmodus. (Informationen zum Einstellen der anfänglichen Ausrichtung der Anwendung finden Sie unter „iPhone-Anwendungseinstellungen“ auf Seite 16.)

Bei Anwendungen, die nicht im Vollbild angezeigt werden, werden die oberen 20 Pixel der Standardgrafik ignoriert. Das iPhone zeigt seine Statusleiste über dem 20 Pixel breiten Rechteck am oberen Rand der Standardgrafik an. Bei einer Anwendung mit querformatiger Ausrichtung entspricht dieser Bereich dem 20 Pixel breiten Rechteck am linken Rand der Datei „Default.png“ (das im Querformat oben angezeigt wird). Bei einer Anwendung mit hochformatiger Ausrichtung ist dieser Bereich das 20 Pixel breite Rechteck am oberen Rand der Datei „Default.png“.

Bei den meisten Anwendungen sollte das Bild „Default.png“ mit dem Startbildschirm der Anwendung übereinstimmen. So erstellen Sie ein Bildschirmfoto des Startbildschirms Ihrer Anwendung:

1. Öffnen Sie die Anwendung auf dem iPhone. Wenn der erste Bildschirm der Benutzeroberfläche erscheint, drücken Sie die Home-Taste (unter dem Bildschirm) und halten Sie sie gedrückt. Währen Sie die Home-Taste gedrückt halten, drücken Sie auf die Power/Standby-Taste oben auf dem Gerät. Damit wird ein Bildschirmfoto erstellt und an die Camera Roll gesendet.
2. Übertragen Sie das Bild auf Ihren Entwicklungscomputer, indem Sie iPhoto oder eine andere Anwendung zum Übertragen von Fotos verwenden. (Auf dem Mac können Sie auch die Bilderfassung verwenden.)

   Sie können das Foto auch per E-Mail an den Entwicklungscomputer senden.

   - Öffnen Sie die Fotos-Anwendung.
   - Öffnen Sie die Camera Roll.
   - Öffnen Sie das Bildschirmfoto, das Sie aufgenommen haben.
   - Tippen Sie auf das Bild und dann auf den Vorwärtspfeil unten links. Klicken Sie dann auf „Per E-Mail senden“ und senden Sie das Bild an sich selbst.

Hinweis: *Sie können beliebige Grafiken für die Datei „Default.png“ verwenden, solange sie die richtigen Abmessungen hat. Häufig ist es jedoch am besten, wenn das Bild in „Default.png“ mit dem Anfangszustand der Anwendung übereinstimmt.*

Verwenden Sie keinen Text im Bild „Default.png“, wenn Ihre Anwendung in mehrere Sprachen lokalisiert wird. „Default.png“ ist statisch; der Text würde also nicht zu den anderen Sprachen passen.

Vergessen Sie nicht, in Flash Professional CS5 die Datei „Default.png“ der Liste der enthaltenen Dateien im Dialogfeld „iPhone-Einstellungen“ hinzuzufügen. Siehe „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16.

Beim Kompilieren mit der PFI-Anwendung in der Befehlszeile müssen Sie in der Liste der einzubeziehenden Elemente auf diese Datei verweisen. Siehe „Erstellen von Installerdateien für iPhone-Anwendungen über die Befehlszeile“ auf Seite 23.

iPhone-Anwendungseinstellungen

Zu den Anwendungseinstellungen gehört Folgendes:

- Der Anwendungsname
- Der Name der IPA-Datei
- Die Anwendungsversion
- Die anfängliche Bildschirmausrichtung der Anwendung und ob die Bildschirmausrichtung automatisch angepasst wird, wenn der Benutzer das iPhone dreht
- Ob die anfängliche Ansicht im Vollbild dargestellt wird
- Informationen zu den Symbolen der Anwendung
- Informationen zur Hardwarebeschleunigung

Sie können die Anwendungseinstellungen in Flash Professional CS5 ändern.

Sie können sie auch in der Anwendungsdeskriptordatei bearbeiten. Die Anwendungsdeskriptordatei ist eine XML-Datei, die Einstellungen für die Anwendung enthält.

Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5

Im Flash Professional CS5-Dialogfeld „iPhone-Einstellungen“ können Sie viele grundlegende Eigenschaften der iPhone-Anwendung definieren.

So öffnen Sie das Dialogfeld „iPhone-Einstellungen“:

❖ Wählen Sie „Datei“ > „iPhone-Einstellungen“.

Registerkarte „Allgemein“

Die Registerkarte „Allgemein“ enthält die folgenden iPhone-relevanten Einstellungen:

- Ausgabedatei – Der Name der Anwendung, der unter dem Anwendungssymbol auf dem iPhone angezeigt wird. Der Name der Ausgabedatei darf kein Pluszeichen (+) enthalten.
- App-Name – Der Name der Anwendung, der unter dem Anwendungssymbol auf dem iPhone angezeigt wird. Der Anwendungsname darf kein Pluszeichen (+) enthalten.
- Version – Hilft Benutzern dabei, festzustellen, welche Version Ihrer Anwendung sie installieren. Die Version wird als CFBundleVersion der iPhone-Anwendung verwendet. Sie muss in einem Format wie nnnnn[.nn[.nn]] vorliegen, wobei n eine Ziffer zwischen 0 und 9 ist und die Klammern optionale Komponenten wie 1, 1.0 oder 1.0.1 angeben. iPhone-Versionen dürfen nur Ziffern und Dezimalpunkte enthalten. iPhone-Versionen können bis zu zwei Dezimalpunkte enthalten.
- Seitenverhältnis – Die anfängliche Ausrichtung der Anwendung (Hochformat oder Querformat).

- Vollbild – Gibt an, ob die Anwendung den gesamten Bildschirm nutzt oder in der iPhone-Statusleiste angezeigt wird.
- Automatische Ausrichtung – Legt fest, dass der Anzeigeinhalt der Anwendung neu ausgerichtet wird, wenn das iPhone neu ausgerichtet wird.

  Wenn Sie die automatische Ausrichtung verwenden, sollten Sie ActionScript-Code hinzufügen, um die `align`-Eigenschaft folgendermaßen einzustellen:

  ```
  stage.align = StageAlign.TOP_LEFT;
  stage.scaleMode = StageScaleMode.NO_SCALE;
  ```

- Rendering – Gibt an, wie Anzeigeobjekte auf dem iPhone dargestellt werden:
  - CPU – Die Anwendung verwendet die CPU für die Darstellung aller Anzeigeobjekte. Es wird keine Hardwarebeschleunigung eingesetzt.
  - GPU – Die Anwendung verwendet die iPhone-GPU, um Bitmaps zusammenzusetzen.
  - Auto – Diese Funktion wurde nicht implementiert.

  Weitere Informationen finden Sie unter „Hardwarebeschleunigung“ auf Seite 38.
- Enthaltene Dateien – Fügen Sie alle Dateien und Verzeichnisse zur Komprimierung in der iPhone-Anwendung hinzu. Die SWF-Hauptdatei und die Anwendungsdeskriptordatei werden automatisch einbezogen. Fügen Sie ggf. weitere erforderliche Bestände zur Liste der enthaltenen Dateien hinzu. Vergessen Sie nicht, die Datei mit der Anfangsgrafik (Default.pgn) zur Liste der enthaltenen Dateien hinzuzufügen.

Registerkarte „Bereitstellung“

Die Registerkarte „Bereitstellung“ enthält Einstellungen zum Signieren und Kompilieren der Anwendung:

- Digitale Signatur für iPhone – Legen Sie eine P12-Zertifikatdatei und das Kennwort für das Zertifikat fest. Sie müssen das Apple iPhone-Zertifikat in das .p12-Format konvertieren. Weitere Informationen finden Sie unter „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.
- Provisioning-Datei – Zeigen Sie auf die Provisioning-Datei für diese Anwendung, die Sie von Apple bekommen haben. Weitere Informationen finden Sie unter „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.
- App-ID – Die App-ID identifiziert Ihre Anwendung eindeutig. Wenn die Provisioning-Datei an eine bestimmte App-ID gebunden ist, füllt Flash Professional CS5 dieses Feld aus und Sie können es nicht bearbeiten. Andernfalls lässt das Provisioning-Profil mehrere App-IDs (Platzhalter) zu. Geben Sie eine App-ID an, die dem Muster der App-ID entspricht, das Sie bei Apple angegeben haben:
  - Wenn Ihre Apple App-ID „com.myDomain.*“ lautet, muss die App-ID im Dialogfeld „iPhone-Einstellungen“ mit „com.myDomain.“ beginnen (zum Beispiel „com.myDomain.myApp“ oder „com.myDomain.app22“).
  - Wenn Ihre Apple App-ID „*“ lautet, kann die App-ID im Dialogfeld „iPhone-Einstellungen“ eine beliebige Folge gültiger Zeichen sein.

  Sie finden die mit Ihrem Provisioning-Profil verknüpfte Apple App-ID (oder das Muster für Platzhalter-App-IDs) im iPhone Dev Center (http://developer.apple.com/iphone). Gehen Sie zum Portal des iPhone Developer Program und rufen Sie den Provisioning-Bereich auf.

  Wichtig: Ignorieren Sie die Zeichen am Anfang der Apple App-ID. Apple nennt diese Zeichenfolge die Bundle-Seed-ID. Wenn Apple Ihre App-ID zum Beispiel als „96LPVWEASL.com.example.bob.myApp“ aufführt, ignorieren Sie 96LPVWEASL und verwenden „com.example.bob.myApp“ als App-ID. Wenn Apple Ihre App-ID als „5RM86Z4DJM.*“ aufführt, ignorieren Sie 5RM86Z4DJM – dies ist eine Platzhalter-App-ID.

- iPhone-Bereitstellungstyp:
  - Schnellveröffentlichung zum Gerätetesten – Wählen Sie diese Option, um schnell eine Version zum Testen der Anwendung auf dem Entwickler-iPhone zu kompilieren.
  - Schnellveröffentlichung zum Geräte-Debugging – Wählen Sie diese Option, um schnell eine Debug-Version zum Testen der Anwendung auf dem Entwickler-iPhone zu kompilieren. Mit dieser Option kann der Flash Professional CS5-Debugger die `trace()`-Ausgabe von der iPhone-Anwendung empfangen. (Siehe „Debuggen von iPhone-Anwendungen“ auf Seite 26.)
  - Bereitstellung - Ad Hoc – Wählen Sie diese Option, um eine Anwendung für die Ad-hoc-Bereitstellung zu erstellen. Siehe dazu das Apple iPhone Developer Center
  - Bereitstellung - Apple App Store – Wählen Sie diese Option, um eine endgültige Version der IPA-Datei für die Bereitstellung im Apple App Store zu erstellen.

Registerkarte „Symbole“

Geben Sie auf der Registerkarte „Symbole“ den Speicherort der Bilddatei für das 29-x-29-Pixel-Symbol, das 48-x-48-Pixel-Symbol, das 57-x-57-Pixel-Symbol, das 72-x-72-Pixel-Symbol und das 512-x-512-Pixel-Symbol an. Siehe „iPhone-Symbole und Bilder für den Startbildschirm“ auf Seite 14.

Hinweis: *Optionen für 48-x-48- und 72-x-72-Pixel-Symbole sind in der Version Packager for iPhone Preview, die in Flash Professional CS5 enthalten ist, nicht verfügbar. Wählen Sie in Flash Professional CS5 „Hilfe“ > „Updates“, um diese Optionen hinzuzufügen.*

Festlegen von iPhone-Anwendungseigenschaften in der Anwendungsdeskriptordatei

Die Anwendungsdeskriptordatei ist eine XML-Datei mit Eigenschaften für die gesamte Anwendung, zum Beispiel Name, Version, Copyright und andere Angaben.

Flash Professional CS5 generiert eine Anwendungsdeskriptordatei basierend auf den Einstellungen im Dialogfeld „iPhone-Einstellungen“. Sie können die Anwendungsdeskriptordatei auch in einem Texteditor bearbeiten. Flash Professional benennt die Anwendungsdeskriptordatei, indem an den Projektnamen „-app.xml“ angehängt wird. Die Anwendungsdeskriptordatei für das Projekt „HelloWorld“ heißt zum Beispiel „HelloWorld-app.xml“. Bearbeiten Sie die Anwendungsdeskriptordatei, wenn Sie Einstellungen definieren möchten, die vom Flash Professional CS5-Dialogfeld „iPhone-Einstellungen“ nicht unterstützt werden. Sie können zum Beispiel das Element `InfoAdditions` definieren, um „info.Plist“-Einstellungen für die Anwendung zu definieren.

Wichtig: Bearbeiten Sie die Anwendungsdeskriptordatei nicht, während das Flash Professional CS5-Dialogfeld geöffnet ist. Speichern Sie Änderungen an der Anwendungsdeskriptordatei, bevor Sie das Dialogfeld „iPhone-Einstellungen“ öffnen.

Es folgt ein Beispiel für eine Anwendungsdeskriptordatei:

```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://ns.adobe.com/air/application/2.0">
    <id>com.example.HelloWorld</id>
    <filename>HelloWorld</filename>
    <name>Hello World</name>
    <version>v1</version>
    <initialWindow>
        <renderMode>gpu</renderMode>
        <content>HelloWorld.swf</content>
        <fullScreen>true</fullScreen>
        <aspectRatio>portrait</aspectRatio>
        <autoOrients>true</autoOrients>
    </initialWindow>
    <supportedProfiles>mobileDevice desktop</supportedProfiles>
    <icon>
        <image29x29>icons/icon29.png</image29x29>
        <image57x57>icons/icon57.png</image57x57>
        <image512x512>icons/icon512.png</image512x512>
    </icon>
    <iPhone>
        <InfoAdditions>
            <![CDATA[
                <key>UIStatusBarStyle</key>
                <string>UIStatusBarStyleBlackOpaque</string>
                <key>UIRequiresPersistentWiFi</key>
                <string>NO</string>
            ]]>
        </InfoAdditions>
    </iPhone>
</application>
```

Nachstehend sind Details zu den Einstellungen in dieser Anwendungsdeskriptordatei aufgeführt:

- Im Element `<application>` ist der AIR 2.0-Namespace erforderlich, um iPhone-Anwendungen zu erstellen:

  <application xmlns="http://ns.adobe.com/air/application/2.0">

- Das Element `<id>`:

  <id>com.example.as3.HelloWorld</id> Die Anwendungs-ID identifiziert Ihre Anwendung eindeutig. Das empfohlene Format ist ein durch einen Punkt getrennter String im umgekehrten DNS-Stil, zum Beispiel `"com.firma.AppName"`. Der Compiler verwendet diesen Wert als Bundle-ID für die iPhone-Anwendung.

  Wenn die Provisioning-Datei an eine bestimmte App-ID gebunden ist, verwenden Sie diese App-ID in diesem Element. Ignorieren Sie die Zeichen, die Apple am Anfang der Apple App-ID zuweist (dies ist die Bundle-Seed-ID). Wenn die App-ID für das Provisioning-Profil zum Beispiel „96LPVWEASL.com.example.bob.myApp“ lautet, verwenden Sie in der Anwendungsdeskriptordatei „com.example.bob.myApp“ als App-ID.

  Wenn das Provisioning-Profil mehrere App-IDs zulässt (Platzhalter), endet die App-ID mit einem Sternchen (zum Beispiel 5RM86Z4DJM.*). Geben Sie eine App-ID an, die dem Muster der App-ID entspricht, das Sie bei Apple angegeben haben:

  - Wenn Ihre Apple App-ID „com.myDomain.*“ lautet, muss die App-ID in der Anwendungsdeskriptordatei mit „com.myDomain“ beginnen. Sie können eine App-ID wie „com.myDomain.myApp“ oder „com.myDomain.app22“ angeben.
  - Wenn Ihre Apple App-ID „*“ lautet, kann die App-ID in der Anwendungsdeskriptordatei eine beliebige Folge gültiger Zeichen sein.

Sie finden die mit Ihrem Provisioning-Profil verknüpfte Apple App-ID (oder das Muster für Platzhalter-App-IDs) im iPhone Dev Center (http://developer.apple.com/iphone). Gehen Sie zum Portal des iPhone Developer Program und rufen Sie den Provisioning-Bereich auf.

Wichtig: Ignorieren Sie die Zeichen am Anfang der Apple App-ID. Apple nennt diese Zeichenfolge die Bundle-Seed-ID. Wenn Apple Ihre App-ID zum Beispiel als „5RM86Z4DJM.*“ aufführt, ignorieren Sie 5RM86Z4DJM – dies ist eine Platzhalter-App-ID. Wenn Apple Ihre App-ID als „96LPVWEASL.com.example.bob.myApp“ aufführt, ignorieren Sie 96LPVWEASL und verwenden „com.example.bob.myApp“ als App-ID.

- Das Element `<filename>`:

  <filename>HelloWorld</filename> Der Name, der für die iPhone-Installerdatei verwendet wird. Der Dateiname darf kein Pluszeichen (*) enthalten.

- Das Element `<name>`:

  <name>Hello World</name> Der Name der Anwendung, der in der iTunes-Anwendung und auf dem iPhone angezeigt wird. Der Name darf kein Pluszeichen (+) enthalten.

- Das Element `<version>`:

  <version>1.0</version> Hilft Benutzern dabei, festzustellen, welche Version Ihrer Anwendung sie installieren. Die Version wird als CFBundleVersion der iPhone-Anwendung verwendet. Sie muss in einem Format wie nnnnn[.nn[.nn]] vorliegen, wobei n eine Ziffer zwischen 0 und 9 ist und die Klammern optionale Komponenten wie 1, 1.0 oder 1.0.1 angeben. iPhone-Versionen dürfen nur Ziffern und Dezimalpunkte enthalten. iPhone-Versionen können bis zu zwei Dezimalpunkte enthalten.

- Das Element `<initialWindow>` enthält die folgenden untergeordneten Elemente, um die Eigenschaften für das anfängliche Erscheinungsbild der Anwendung anzugeben:

  <content>HelloWorld.swf</content> Gibt die SWF-Stammdatei an, die in die iPhone-Anwendung kompiliert werden soll.

  <visible>true</visible> Dies ist eine erforderliche Einstellung.

  <fullScreen>true</fullScreen> Gibt an, dass die Anwendung den gesamten Bildschirm des iPhone nutzt.

  <aspectRatio>portrait</aspectRatio> Gibt an, dass das anfängliche Seitenverhältnis der Anwendung Hochformat ist (statt Querformat). Beachten Sie, dass die Datei „Default.png“, die das Anfangsfenster der Anwendung definiert, unabhängig von dieser Einstellung 320 Pixel breit und 480 Pixel hoch sein sollte. (Siehe „iPhone-Symbole und Bilder für den Startbildschirm“ auf Seite 14.)

  <autoOrients>true</autoOrients> (Optional) Legt fest, ob die Ausrichtung des Inhalts in der Anwendung automatisch angepasst werden soll, wenn die physische Ausrichtung des Geräts geändert wird. Der Standardwert lautet `true`. Sie können die automatische Ausrichtung ändern, indem Sie die `preventDefault()`-Methode eines `orientationChanging`-Ereignisses aufrufen, das vom Stage-Objekt abgesetzt wird. Weitere Informationen finden Sie unter Festlegen und Erkennen der Bildschirmausrichtung.

  Wenn Sie die automatische Ausrichtung verwenden, stellen Sie die `align`-Eigenschaft folgendermaßen ein, um beste Ergebnisse zu erzielen:

  ```
  stage.align = StageAlign.TOP_LEFT;
  stage.scaleMode = StageScaleMode.NO_SCALE;
  ```

  <renderMode>gpu</renderMode> (Optional) Der Renderingmodus, der von der Anwendung verwendet wird. Es gibt drei mögliche Einstellungen:

  - `cpu` – Die Anwendung verwendet die CPU für die Darstellung aller Anzeigeobjekte. Es wird keine Hardwarebeschleunigung eingesetzt.

- `gpu` – Die Anwendung verwendet die iPhone-GPU, um Bitmaps zusammenzusetzen.
- `auto` – Diese Funktion wurde nicht implementiert.

Weitere Informationen finden Sie unter „Hardwarebeschleunigung“ auf Seite 38.

- Das Element `<profiles>`:

  <profiles>mobileDevice</profiles> Beschränkt die Anwendung darauf, in das Profil für mobile Geräte kompiliert zu werden. Dieses Profil unterstützt zurzeit nur iPhone-Anwendungen. Es gibt drei unterstützte Profile:

  - `desktop`– Eine Desktop-AIR-Anwendung.
  - `extendedDesktop` – Eine Desktop-AIR-Anwendung mit Unterstützung für die native Prozess-API.
  - `mobileDevice` – Eine AIR-Anwendung für ein mobiles Gerät. Zurzeit ist das iPhone das einzige unterstützte mobile Gerät.

  Wenn Sie die Anwendung auf ein bestimmtes Profil beschränken, verhindern Sie, dass sie in andere Profile kompiliert werden kann. Wenn Sie kein Profil angeben, kann eine Anwendung für jedes dieser Profile kompiliert werden. Sie können verschiedene Profile spezifizieren, indem Sie durch Leerzeichen getrennt im `<profiles>`-Element angeben.

  Achten Sie darauf, `mobileDevice` als unterstütztes Profil anzugeben (oder lassen Sie das `<profiles>`-Element leer).

- Das Element `<icon>` enthält die folgenden untergeordneten Elemente, um die von der Anwendung verwendeten Symbole anzugeben:

  <image29x29>icons/icon29.png</image29x29> Dies ist das Bild, das in Spotlight-Suchergebnissen verwendet wird.

  <image48x48>icons/icon48.png</image48x48> Dies ist das Bild, das in Spotlight-Suchergebnissen auf dem iPad verwendet wird.

  <image57x57>icons/icon57.png</image57x57> Dies ist das Bild, das für die Startbildschirme von iPhone und iPod Touch verwendet wird.

  <image72x72>icons/icon72.png</image72x72> Dies ist das Bild, das auf Startbildschirm des iPad verwendet wird.

  <image512x512>icons/icon512.png</image512x512> Dies ist das Bild, das in iTunes verwendet wird.

  Der Packager for iPhone verwendet die Symbole (29, 57 und 512), die in der Anwendungsdeskriptordatei definiert sind. Das Tool kopiert sie in die Dateien „Icon-Small.png“, „Icon.png“ bzw. „iTunesArtwork“. Um das Erstellen dieser Kopien zu vermeiden, können Sie diese Dateien direkt „verpacken“. Dazu platzieren Sie sie in dem Verzeichnis, das die Anwendungsdeskriptordatei enthält, und geben die korrekten Namen und Pfade an.

  Das 512-Bild dient nur zu internen Testzwecken. Wenn Sie eine Anwendung an Apple übermitteln, senden Sie das 512-Pixel-Bild separat. Es ist nicht in der IPA-Datei enthalten. Geben Sie das Bild an, um sicherzustellen, dass das 512-Pixel-Bild in iTunes wie gewünscht aussieht, bevor Sie es übermitteln.

- Das Element `<iPhone>` enthält die folgenden untergeordneten Elemente zum Festlegen iPhone-spezifischer Einstellungen:

  <InfoAdditions></InfoAdditions> enthält die untergeordneten Elemente zum Festlegen der Schlüssel-Wert-Paare, die als Info.plist-Einstellungen für die Anwendung verwendet werden:

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

In diesem Beispiel legen die Werte den Statusleistenstil der Anwendung fest und geben an, dass die Anwendung keinen ständigen Wi-Fi-Zugang benötigt.

Die InfoAdditions-Einstellungen sind in einem `CDATA`-Tag enthalten.

Für die iPad-Unterstützung schließen Sie Schlüssel-Wert-Einstellungen für `UIDeviceFamily` mit ein. Die `UIDeviceFamily`-Einstellung ist ein Array von Strings. Jeder String definiert unterstützte Geräte. Die Einstellung `<string>1</string>` definiert Unterstützung für iPhone und iPod Touch. Die Einstellung `<string>2</string>` definiert Unterstützung für das iPad. Wenn Sie nur einen dieser Strings angeben, wird nur die jeweilige Gerätefamilie unterstützt. Die folgende Einstellung limitiert die Unterstützung zum Beispiel auf das iPad:

```
<key>UIDeviceFamily</key>
    <array>
        <string>2</string>
    </array>>
```

Die folgende Einstellung legt die Unterstützung beider Gerätefamilien fest (iPhone/iPod Touch und iPad):

```
<key>UIDeviceFamily</key>
<array>
    <string>1</string>
    <string>2</string>
</array>
```

Informationen zu weiteren Info.plist-Einstellungen finden Sie in der Entwicklerdokumentation von Apple.

Kompilieren von Installerdateien für iPhone-Anwendungen (IPA-Dateien)

Mit dem Packager for iPhone kompilieren Sie eine ActionScript-3.0-Anwendung in eine IPA-Datei.

Erstellen von Installerdateien für iPhone-Anwendungen mit dem in Flash Professional CS5 enthaltenen Packager for iPhone

So verwenden Sie den Packager for iPhone, der in Flash Professional CS5 enthalten ist:

1. Wählen Sie „Datei“ > „Veröffentlichen“.
2. Überprüfen Sie, ob Sie im Dialogfeld „iPhone-Einstellungen“ für alle Einstellungen Werte angegeben haben. Vergewissern Sie sich, dass Sie auf der Registerkarte „Bereitstellung“ die richtigen Optionen gewählt haben. Siehe „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16.
3. Klicken Sie auf die Schaltfläche „Veröffentlichen“.

Der Packager for iPhone generiert die Installerdatei (IPA-Datei) für die iPhone-Anwendung. Das Kompilieren der IPA-Datei kann einige Minuten dauern.

Sie können den Packager for iPhone auch von der Befehlszeile ausführen. Siehe „Erstellen von Installerdateien für iPhone-Anwendungen über die Befehlszeile“ auf Seite 23.

Erstellen von Installerdateien für iPhone-Anwendungen über die Befehlszeile

Sie können den Packager for iPhone von der Befehlszeile ausführen. Der Packager for iPhone konvertiert den Bytecode der SWF-Datei und andere Quelldateien in eine native iPhone-Anwendung.

1 Öffnen Sie eine Befehls-Shell oder ein Terminal und navigieren Sie zum Projektordner der iPhone-Anwendung.

2 Als Nächstes erstellen Sie mit dem PFI-Tool die IPA-Datei, indem Sie die folgende Syntax verwenden:

```
pfi -package -target [ipa-test ipa-debug ipa-app-store ipa-ad-hoc] -provisioning-profile
PROFILE_PATH SIGNING_OPTIONS TARGET_IPA_FILE APP_DESCRIPTOR SOURCE_FILES
```

Ändern Sie den Verweis auf `pfi` so, dass er den vollständigen Pfad zur PFI-Anwendung enthält. Die PFI-Anwendung befindet sich im Unterverzeichnis „pfi/bin“ des Installationsverzeichnisses von Flash Professional CS5.

Wählen Sie die `-target`-Option, die dem Typ der zu erstellenden iPhone-Anwendung entspricht:

- `-target ipa-test` – Wählen Sie diese Option, um schnell eine Version der Anwendung zum Testen auf dem Entwickler-iPhone zu kompilieren.
- `-target ipa-debug` – Wählen Sie diese Option, um eine Debug-Version der Anwendung zum Testen auf dem Entwickler-iPhone zu kompilieren. Mit dieser Option können Sie eine Debug-Sitzung verwenden, um `trace()`-Ausgaben von der iPhone-Anwendung zu erhalten.

  Sie können eine der folgenden `-connect`-Optionen (`CONNECT_OPTIONS`) einschließen, um die IP-Adresse des Entwicklungscomputers, auf dem der Debugger ausgeführt wird, anzugeben:

  - `-connect` – Die Anwendung versucht, eine Verbindung zu einer Debug-Sitzung auf dem Entwicklungscomputer herzustellen, der zum Kompilieren der Anwendung verwendet wird.
  - `-connect IP_ADDRESS` – Die Anwendung versucht, eine Verbindung zu einer Debug-Sitzung auf dem Computer mit der angegebenen IP-Adresse herzustellen. Zum Beispiel:

    ```
    -target ipa-debug -connect 192.0.32.10
    ```

  - `-connect HOST_NAME` – Die Anwendung versucht, eine Verbindung zu einer Debug-Sitzung auf dem Computer mit der angegebenen Hostadresse herzustellen. Zum Beispiel:

    ```
    -target ipa-debug -connect bobroberts-mac.example.com
    ```

  Hinweis: *Die `-connect`-Option ist in Packager for iPhone Preview (in Flash Professional CS5 enthalten) nicht verfügbar. Aktualisieren Sie Packager for iPhone, indem Sie in Flash Professional CS5 „Hilfe“ > „Updates“ wählen.*

  Die `-connect`-Option ist optional. Wenn sie nicht angegeben wird, versucht die resultierende Debug-Anwendung nicht, eine Verbindung zum einem gehosteten Debugger herzustellen.

  Wenn ein Verbindungsversuch fehlschlägt, zeigt die Anwendung ein Dialogfeld an, in dem der Benutzer die IP-Adresse des Debugger-Hostcomputers angeben kann. Ein Verbindungsversuch kann fehlschlagen, wenn das Gerät keine WiFi-Verbindung hat. Dies kann auch passieren, wenn das Gerät zwar verbunden ist, aber nicht hinter der Firewall des Debugger-Hostcomputers.

  Weitere Informationen finden Sie unter „Debuggen von iPhone-Anwendungen“ auf Seite 26.

  Sie können auch die `-renderingdiagnostics`-Option einschließen, um die Diagnosefunktion für die GPU-Darstellung zu aktivieren. Weitere Informationen finden Sie unter „Debuggen mit der GPU-Darstellungsdiagnose“ in „Debuggen von iPhone-Anwendungen“ auf Seite 26.

- `-target ipa-ad-hoc` – Wählen Sie diese Option, um eine Anwendung für die Ad-hoc-Bereitstellung zu erstellen. Siehe dazu das Apple iPhone Developer Center
- `-target ipa-app-store` – Wählen Sie diese Option, um eine endgültige Version der IPA-Datei für die Bereitstellung im Apple App Store zu erstellen.

Ersetzen Sie *PROFILE_PATH* durch den Pfad zur Provisioning-Profildatei Ihrer Anwendung. Weitere Informationen zu Provisioning-Profilen finden Sie unter „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.

Ersetzen Sie *SIGNING_OPTIONS* so, dass auf Ihr iPhone-Entwicklerzertifikat und das entsprechende Kennwort verwiesen wird. Verwenden Sie die folgende Syntax:

```
-storetype pkcs12 -keystore P12_FILE_PATH -storepass PASSWORD
```

Ersetzen Sie *P12_FILE_PATH* durch den Pfad zu Ihrer P12-Zertifikatdatei. Ersetzen Sie *PASSWORD* durch das Zertifikatkennwort. (Siehe dazu auch folgendes Beispiel.) Weitere Informationen zur P12-Zertifikatdatei finden Sie unter „Konvertieren eines Entwicklerzertifikats in eine P12-Datei“ auf Seite 7.

Ersetzen Sie *APP_DESCRIPTOR* so, dass auf die Anwendungsdeskriptordatei verwiesen wird.

Ersetzen Sie *SOURCE_FILES* so, dass auf die Haupt-SWF-Datei des Projekts, ggf. gefolgt von weiteren einzuschließenden Beständen, verwiesen wird. Schließen Sie die Pfade zu allen Symboldateien mit ein, die Sie in Flash CS5 im Dialogfeld „Anwendungseinstellungen“ oder in einer benutzerdefinierten Anwendungsdeskriptordatei definiert haben. Fügen Sie auch die Datei mit der Anfangsgrafik, „Default.png“, hinzu.

Betrachten Sie folgendes Beispiel:

```
pfi -package -target ipa-test -storetype pkcs12 -keystore
"/Users/Jeff/iPhoneCerts/iPhoneDeveloper_Jeff.p12" -storepass dfb7VKL19 "HelloWorld.ipa"
"HelloWorld-app.xml" "HelloWorld.swf" "Default.png" "icons/icon29.png" "icons/icon57.png"
"icons/icon512.png"
```

Das Beispiel kompiliert die Datei „HelloWorld.ipa“ unter Verwendung der folgenden Elemente:

- Ein spezifisches PKCS#12-Zertifikat mit dem Zertifikatkennwort „dfb7VKL19“
- Die Anwendungsdeskriptordatei „HelloWorld-app.xml“
- Die Quelldatei „HelloWorld.swf“
- Spezifische „Default.png“- und Symboldateien

Die pfi-Anwendung kompiliert die Anwendung basierend auf der Anwendungsdeskriptordatei, der SWF-Datei und der anderen Bestände in eine IPA-Datei.

Unter Mac OS können Sie das im Schlüsselbund gespeicherte Zertifikat verwenden, indem Sie dem pfi-Befehl die folgenden Optionen hinzufügen:

```
-alias ALIAS_NAME -storetype KeychainStore -providerName Apple
```

Ersetzen Sie *ALIAS_NAME* durch den Aliasnamen des Zertifikats, das Sie verwenden möchten. Wenn Sie auf ein Zertifikat zeigen, das im Mac-Schlüsselbund gespeichert ist, geben Sie den Aliasnamen an anstatt auf einen Zertifikatspeicherort zu zeigen.

Installieren von iPhone-Anwendungen

Um Ihre Entwicklungsanwendung auf einem iPhone zu installieren, fügen Sie das Provisioning-Profil dem iPhone hinzu und installieren die Anwendung auf dem iPhone.

Hinzufügen des Provisioning-Profils zum iPhone

So fügen Sie das Provisioning-Profil zum iPhone hinzu:

1 Wählen Sie in iTunes „Ablage“ > „Zu Mediathek hinzufügen“. Wählen Sie dann die Provisioning-Profildatei aus (die Datei hat die Dateinamenerweiterung „mobileprovision“).

Stellen Sie sicher, dass Ihr iPhone dem Provisioning-Profil hinzugefügt wird. Sie können Provisioning-Profile im Apple iPhone Dev Center verwalten (http://developer.apple.com/iphone/). Gehen Sie zum Bereich „iPhone Developer Program Portal“ der Site. Klicken Sie auf den Hyperlink „Geräte“, um die Liste der Geräte zu verwalten, auf denen Ihre Entwicklungsanwendung installiert werden kann. Klicken Sie auf den Hyperlink „Provisioning“, um Ihre Provisioning-Profile zu verwalten.

2 Verbinden Sie das iPhone mit dem Computer und synchronisieren Sie es.

Informationen zum Beziehen eines Provisioning-Profils finden Sie unter „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.

Installieren der Anwendung

Sie installieren Ihre Entwicklungsanwendung genau wie jede andere IPA-Datei:

1 Wenn Sie bereits eine Version der Anwendung installiert haben, löschen Sie sie vom Gerät und aus der Liste der Anwendungen in iTunes.

2 Fügen Sie die Anwendung in iTunes mit einer der folgenden Methoden hinzu:

- Wählen Sie in iTunes den Befehl „Ablage“ > „Zu Mediathek hinzufügen“. Wählen Sie dann die IPA-Datei aus und klicken Sie auf „Öffnen“.
- Doppelklicken Sie auf die IPA-Datei.
- Ziehen Sie die IPA-Datei in die iTunes-Mediathek.

3 Verbinden Sie das iPhone mit einem USB-Anschluss des Computers.

4 Überprüfen Sie in iTunes die Registerkarte „Programme“ und vergewissern Sie sich, dass die Anwendung in der Liste der zu installierenden Anwendungen ausgewählt ist.

5 Synchronisieren Sie das iPhone.

Fehlerbehebung bei Problemen mit der Anwendungsinstallation

Wenn iTunes beim Installieren der Anwendung einen Fehler anzeigt, überprüfen Sie Folgendes:

- Vergewissern Sie sich, dass die Geräte-ID dem Provisioning-Profil hinzugefügt wurde.
- Um sicherzustellen, dass das Provisioning-Profil installiert wurde, können Sie es in iTunes ziehen oder den Befehl „Ablage“ > „Zur Mediathek hinzufügen“ verwenden.

Überprüfen Sie auch, ob die App-ID Ihrer Anwendung mit der App-ID von Apple übereinstimmt:

- Wenn Ihre Apple App-ID „com.myDomain.*“ lautet, muss die App-ID in der Anwendungsdeskriptordatei oder im Dialogfeld „iPhone-Einstellungen“ mit „com.myDomain“ beginnen (zum Beispiel „com.myDomain.anythinghere“).
- Wenn Ihre Apple App-ID „com.myDomain.myApp“ lautet, muss die App-ID in der Anwendungsdeskriptordatei oder in Flash Professional CS5 ebenfalls „com.myDomain.myApp“ lauten.
- Wenn Ihre Apple App-ID „*“ lautet, kann die App-ID in der Anwendungsdeskriptordatei oder in Flash Professional CS5 beliebig sein.

Sie legen die App-ID der Anwendung in Flash Professional CS5 im Dialogfeld „iPhone-Einstellungen“ oder in der Anwendungsdeskriptordatei fest.

Debuggen von iPhone-Anwendungen

Sie können die Anwendung auf dem Entwicklungscomputer debuggen, indem Sie die Anwendung in ADL ausführen. Das Debugging auf dem iPhone ist ebenfalls möglich.

Einige AIR-Funktionen, die auf dem iPhone nicht unterstützt werden, sind beim Testen einer Anwendung mit ADL (auf dem Entwicklungscomputer) verfügbar. Beachten Sie diese Unterschiede, wenn Sie Inhalte auf dem Desktop testen. Weitere Informationen finden Sie unter „ActionScript 3.0-APIs, die auf mobilen Geräten nicht unterstützt werden“ auf Seite 30.

Debuggen der Anwendung auf dem Entwicklungscomputer

So debuggen Sie die Anwendung auf dem Entwicklungscomputer mit Flash Professional CS5:

❖ Wählen Sie „Debuggen“ > „Debuggen“ > „In AIR Debug Launcher (mobil)“.

Sie können die Anwendung auch debuggen, indem Sie ADL über die Befehlszeile aufrufen. Verwenden Sie dazu folgende Syntax:

```
adl -profile mobileDevice appDescriptorFile
```

Ersetzen Sie *`appDescriptorFile`* durch den Pfad zur Anwendungsdeskriptordatei.

Schließen Sie die `-profile mobileDevice`-Option mit ein.

Debuggen der Anwendung auf dem iPhone

So führen Sie das Debugging der Anwendung auf dem iPhone aus:

1. Kompilieren Sie die Anwendung mit Debug-Unterstützung:
   - In Flash Professional CS5 kompilieren Sie mit der Einstellung „Schnellveröffentlichung zum Geräte-Debugging“. (Siehe „Erstellen von Installerdateien für iPhone-Anwendungen mit dem in Flash Professional CS5 enthaltenen Packager for iPhone“ auf Seite 22.)
   - Wenn Sie die PFI-Befehlszeilenanwendung verwenden, kompilieren Sie die Anwendung mit der `target ipa-debug`-Option. (Siehe „Erstellen von Installerdateien für iPhone-Anwendungen über die Befehlszeile“ auf Seite 23.)
2. Installieren Sie die Anwendung auf dem iPhone.
3. Schalten Sie beim iPhone die Wi-Fi-Funktion ein und stellen Sie eine Verbindung zu demselben Netzwerk her, mit dem auch der Entwicklungscomputer verbunden ist.
4. Starten Sie eine Debug-Sitzung auf dem Entwicklungscomputer. Wählen Sie in Flash Professional CS5 „Debuggen“ > „Remote-Debug-Sitzung beginnen“ > „ActionScript 3.0“.
5. Führen Sie die Anwendung auf dem iPhone aus.

   Die Debug-Version der Anwendung fordert Sie zur Eingabe der IP-Adresse des Entwicklungscomputers auf. Geben Sie die IP-Adresse ein und tippen Sie auf „OK“. Ermitteln Sie die IP-Adresse des Entwicklungscomputers.
   - Unter Mac OS wählen Sie im Apple-Menü „Systemeinstellungen“. Klicken Sie im Fenster „Systemeinstellungen“ auf das Netzwerksymbol. Im Fenster „Netzwerkeinstellungen“ wird die IP-Adresse angezeigt.
   - Unter Windows rufen Sie die Befehlszeile auf und geben Sie den Befehl `ipconfig` ein.

Die Debug-Sitzung zeigt alle `trace()`-Ausgaben der Anwendung an.

Beim Debuggen einer auf dem iPhone installierten Anwendung unterstützt Flash Professional CS5 alle Debuggingfunktionen einschließlich Haltepunktsteuerung, schrittweise Codeausführung und Variablenüberprüfung.

Debuggen mit der GPU-Darstellungsdiagnose

Mit der GPU-Darstellungsdiagnose können Sie überprüfen, wie die Anwendung die Hardwarebeschleunigung nutzt (bei Anwendungen, die den GPU-Darstellungsmodus verwenden). Um diese Funktion zu verwenden, kompilieren Sie die Anwendung mit dem PFI-Tool in der Befehlszeile und schließen die `-renderingdiagnostics`-Option ein:

```
pfi -package -renderingdiagnostics -target ipa-debug -connect ...
```

Der `-renderingdiagnostics`-Kennzeichner muss direkt nach dem `-package`-Kennzeichner folgen.

Die GPU-Darstellungsdiagnosefunktion zeigt farbige Rechtecke für alle Anzeigeobjekte an:

- Blau – Das Anzeigeobjekt ist keine Bitmap und wird nicht als Bitmap zwischengespeichert, und es wird dargestellt.

  Wenn mehrere blaue Rechtecke für ein Anzeigeobjekt erscheinen, das sich nicht ändert, kann dies daran liegen, dass es sich mit sich bewegenden Anzeigeobjekten überschneidet. Das Anzeigeobjekt könnte zum Beispiel der Hintergrund für sich bewegende Anzeigeobjekte sein. Ziehen Sie das Zwischenspeichern als Bitmap für das Anzeigeobjekt in Betracht.

  Wenn ein blaues Rechteck für ein Objekt angezeigt wird, das eigentlich zwischengespeichert werden soll, kann dies daran liegen, dass das Objekt einen Effekt verwendet, der von der GPU nicht angewendet werden kann. Betroffene Effekte sind zum Beispiel bestimmte Mischmodi, Farbtransformationen, die `scrollRect`-Eigenschaft und Masken.

  Die Anwendung zeigt auch dann ein blaues Rechteck an, wenn zur GPU hochgeladene Anzeigeobjekte die Speichergrenzen überschreiten.

  Die Anwendung protokolliert Meldungen für jedes blaue Rechteck. Die Anwendung gibt diese Meldungen zusammen mit anderen `trace()`- und Debug-Ausgabemeldungen an.

- Grün – Das Anzeigeobjekt ist eine Bitmap oder wird als Bitmap zwischengespeichert, und es wird zum ersten Mal zur GPU hochgeladen.

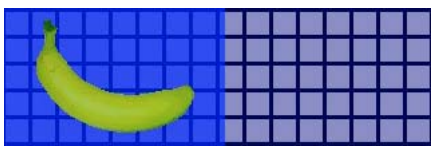
  Wenn für ein Anzeigeobjekt wiederholt ein grünes Rechteck angezeigt wird, erstellt der Code in der Anwendung das Anzeigeobjekt neu. Dies kann zum Beispiel vorkommen, wenn die Zeitleiste zu einem Bild zurückkehrt, das das Anzeigeobjekt erstellt. Ziehen Sie in Betracht, den Inhalt zu ändern, damit keine identischen Objekte neu erstellt werden.

- Rot – Das Anzeigeobjekt ist eine Bitmap oder wird als Bitmap zwischengespeichert, und es wird erneut zur GPU hochgeladen.

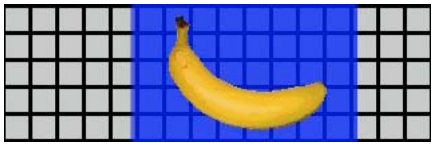
  Ein rotes Rechteck wird jedes Mal angezeigt, wenn sich ein solches Anzeigeobjekt auf eine Weise ändert, die eine Neu-Darstellung der Bitmaprepräsentation erforderlich macht. Zum Beispiel wird ein 2D-Objekt, dessen `cacheAsBitmapMatrix`-Eigenschaft nicht eingestellt ist, beim Skalieren oder Drehen neu dargestellt. Eine Neu-Darstellung erfolgt auch, wenn untergeordnete Anzeigeobjekt verschoben oder geändert werden.

Jedes farbige Rechteck wird ausgeblendet, nachdem die Bühne vier Neuzeichnungszyklen durchlaufen hat, sofern der Grund für die Färbung in diesen Zyklen nicht erneut auftritt. Wenn es keine Änderungen auf der Bühne gibt, ändert sich die Diagnosefarbe nicht.

Als Beispiel stellen Sie sich ein Bitmap-Anzeigeobjekt (eine Banane) vor einem Vektorhintergrund, der nicht als Bitmap zwischengespeichert wird, vor. Wenn die Banane zum ersten Mal dargestellt wird, ist sie grün. Wenn der Hintergrund zum ersten Mal dargestellt wird, ist er blau:

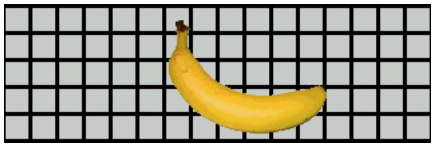


Wenn sich die Banane bewegt, muss die CPU den Hintergrund neu darstellen, sodass die blaue Färbung über dem Hintergrund eingeblendet wird:



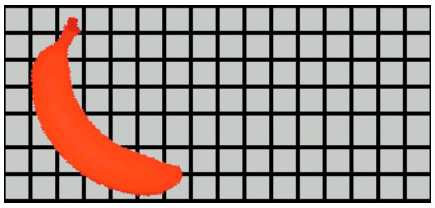
Die blaue Färbung über dem Hintergrund kennzeichnet die neu gezeichneten Bereiche, die an die GPU gesendet werden müssen.

Wenn der Hintergrund jedoch nicht als Bitmap zwischengespeichert wird, zeigt die Darstellungsdiagnosefunktion beim Bewegen der Banane keine Einfärbungen an:



Die Diagnosefunktion zeigt keine Einfärbungen an, da die GPU die Hintergrund-Bitmap weiterhin verwenden kann. Die GPU kann die Banane vor dem Hintergrund zeichnen, ohne die GPU einzubeziehen.

Stellen Sie sich die Banane als 2D-Anzeigeobjekt vor, dessen `cacheAsBitmapMatrix`-Eigenschaft nicht gesetzt ist. Wenn das Anzeigeobjekt gedreht (oder skaliert) wird, zeigt die Diagnosefunktion eine rote Einfärbung an. Damit wird angezeigt, dass die Anwendung eine neue Version des Anzeigeobjekts zur GPU hochladen muss:



Übermitteln von iPhone-Anwendungen an den App Store

So übermitteln Sie Ihre Anwendung an den App Store:

1. Beziehen Sie ein Distributionszertifikat und das Provisioning-Profil vom iPhone Dev Center (http://developer.apple.com/iphone/).

   Ein Distributionszertifikat ist „iPhone Developer: XXX.cer“ benannt, wobei XXX Ihr Name ist.

   Weitere Informationen finden Sie unter „Beziehen der Entwicklerdateien von Apple“ auf Seite 4.

2. Konvertieren Sie das Distributionszertifikat in eine P12-Datei.

   Weitere Informationen finden Sie unter „Konvertieren eines Entwicklerzertifikats in eine P12-Datei“ auf Seite 7.

3. Kompilieren Sie Ihre Anwendung unter Verwendung der P12-Datei und des Provisioning-Profils.

Verwenden Sie die P12-Datei, die Sie basierend auf dem Distributionszertifikat erstellt haben. Verwenden Sie die Anwendungs-ID, die mit dem Provisioning-Profil verknüpft ist.

Weitere Informationen finden Sie unter „Kompilieren von Installerdateien für iPhone-Anwendungen (IPA-Dateien)“ auf Seite 22.

4 Ändern Sie den Dateityp der Installerdatei von .ipa in .zip. (Zum Beispiel von MyApp.ipa in MyApp.zip.)

5 Senden Sie die Anwendung zum iPhone Dev Center (http://developer.apple.com/iphone/).

Kapitel 3: ActionScript 3.0-API-Unterstützung für mobile Geräte

Beim Erstellen von mobilen AIR-Anwendungen können Sie dieselben ActionScript 3.0-APIs verwenden, die auch für die Entwicklung von Adobe Flash Player 10.1- und AIR 2-Desktopanwendungen verfügbar sind. Es gibt jedoch Ausnahmen und Ergänzungen.

ActionScript 3.0-APIs, die auf mobilen Geräten nicht unterstützt werden

Einige ActionScript 3.0-APIs werden für Anwendungen, die mit dem Profil für mobile Geräte ausgeführt werden (zum Beispiel iPhone-Anwendungen), nicht unterstützt.

Wenn Sie denselben ActionScript-Code zum Entwickeln für mehrere Profile (zum Beispiel „desktop“ und „mobile“) schreiben, testen Sie mithilfe von Code, ob eine bestimmte API unterstützt wird. Die NativeWindow-Klasse wird zum Beispiel nicht von iPhone-Anwendungen unterstützt. (iPhone-Anwendungen können keine nativen Fenster verwenden oder erstellen.) Um zu testen, ob eine Anwendung mit einem Profil ausgeführt werden kann, das native Fenster unterstützt (zum Beispiel das Profil „desktop“), überprüfen Sie die `NativeWindow.isSupported`-Eigenschaft.

In der folgenden Tabelle sind APIs aufgeführt, die vom Profil für mobile Geräte nicht unterstützt werden. Außerdem sind Eigenschaften aufgelistet, mit denen Sie feststellen können, ob eine Anwendung auf einer Plattform ausgeführt wird, die die jeweilige API unterstützt.

API	**Zu überprüfende Eigenschaft**
Accessibility	Capabilities.hasAccessibility
Camera	Camera.isSupported
DatagramSocket	DatagramSocket.isSupported
DNSResolver	DNSResolver.isSupported
DockIcon	NativeApplication.supportsDockIcon
DRMManager	DRMManager.isSupported
EncryptedLocalStore	EncryptedLocalStore.isSupported
HTMLLoader	HTMLLoader.isSupported
LocalConnection	LocalConnection.isSupported
Microphone	Microphone.isSupported
NativeApplication.exit()	—
NativeApplication.menu	NativeApplication.supportsMenu
NativeApplication.isSetAsDefaultApplication()	NativeApplication.supportsDefaultApplication
NativeApplication.startAtLogin	NativeApplication.supportsStartAtLogin
NativeMenu	NativeMenu.isSupported

API	Zu überprüfende Eigenschaft
NativeProcess	NativeProcess.isSupported
NativeWindow	NativeWindow.isSupported
NativeWindow.notifyUser()	NativeWindow.supportsNotification
NetworkInfo	NetworkInfo.isSupported
PDF Support	HTMLLoader.pdfCapability
PrintJob	PrintJob.isSupported
SecureSocket	SecureSocket.isSupported
ServerSocket	ServerSocket.isSupported
Shader	—
ShaderFilter	—
StorageVolumeInfo	StorageVolumeInfo.isSupported
XMLSignatureValidator	XMLSignatureValidator.isSupported

Sie können keine HTML- und JavaScript-basierten AIR-Anwendungen für das Profil für mobile Geräte schreiben.

Einige ActionScript 3.0-Klassen werden nur teilweise unterstützt:

File

iPhone-Anwendungen haben nur auf das Anwendungsverzeichnis und das Anwendungsspeicherverzeichnis Zugriff. Sie können auch `File.createTempFile()` und `File.createTempDirectory()` aufrufen. Wenn versucht wird, mit einer Operation auf ein anderes Verzeichnis zuzugreifen (zum Beispiel mit einer FileStream-Methode zum Lesen oder Schreiben), wird eine IOError-Ausnahme ausgegeben.

iPhone-Anwendungen unterstützen keine nativen Dateiauswahl-Dialogfelder wie das von der `File.browseForOpen()`-Methode bereitgestellte.

Loader

In einer iPhone-Anwendung können Sie die `Loader.load()`-Methode nicht verwenden. Sie können keinen ActionScript-Code in SWF-Inhalt ausführen, der mit der `Loader.load()`-Methode geladen wurde. Sie können jedoch Bestände (zum Beispiel Movieclips, Bilder, Schriftarten und Sounds aus der Bibliothek) in der SWF-Datei verwenden. Sie können die `Loader.load()`-Methode auch zum Laden von Bilddateien verwenden.

Video

Innerhalb von AIR-Anwendungen auf dem iPhone werden nur Sorensen-Video und ON2 VP6-Video unterstützt.

Mit der `navigateToURL()`-Methode können Sie ein H.264-Video außerhalb der Anwendung öffnen. Übergeben Sie ein URLRequest-Objekt mit einer URL, die zum Video führt, als `request`-Parameter. Das Video startet den Videoplayer des iPhone.

Textfelder

Es gibt Einschränkungen für Schriftarten und andere Einstellungen von Textfeldern auf dem iPhone. Siehe „Schriftarten und Texteingabe“ auf Seite 41.

Nicht unterstützte APIs und Debuggen mit ADL

Einige AIR-Funktionen, die auf dem iPhone nicht unterstützt werden, sind beim Testen einer Anwendung mit ADL (auf dem Entwicklungscomputer) verfügbar. Beachten Sie diese Unterschiede, wenn Sie Inhalte mit ADL testen.

Zu diesen Funktionen gehören die folgenden Video- und Audiocodecs: Speex (Audio), H.264/AVC (Video) und AAC (Audio). Diese Codecs stehen AIR-Anwendungen, die auf dem iPhone ausgeführt werden, nicht zur Verfügung. Sie funktionieren auf dem Desktop jedoch normal.

Die Unterstützung für Eingabehilfen und Bildschirmlesegeräte funktioniert in ADL unter Windows. Diese APIs werden auf dem iPhone jedoch nicht unterstützt.

Das RTMPE-Protokoll funktioniert bei Verwendung unter ADL auf dem Desktop normal. Eine NetConnection, die versucht, unter Verwendung des RTMPE-Protokolls eine Verbindung herzustellen, bleibt auf dem iPhone jedoch erfolglos.

Die Loader-Klasse funktioniert ohne weitere Einschränkungen, wenn Inhalt mit ADL ausgeführt wird. Beim Ausführen auf dem iPhone resultieren Versuche, SWF-Inhalt zu laden, der ActionScript-Bytecode enthält, jedoch zu einer Fehlermeldung.

Shader-Instanzen werden unter ADL ausgeführt. Auf dem iPhone wird Pixel Bender-Bytecode jedoch nicht interpretiert und Shader haben keinen grafischen Effekt.

Weitere Informationen finden Sie unter „Debuggen von iPhone-Anwendungen“ auf Seite 26.

ActionScript-APIs speziell für mobile AIR-Anwendungen

Die folgenden APIs sind nur in AIR-Anwendungen für mobile Geräte verfügbar. Sie funktionieren zurzeit nicht in Flash Player oder Desktopversionen von AIR.

Bildschirmausrichtungs-API

Mithilfe der Bildschirmausrichtungs-API können Sie mit der Ausrichtung der Bühne und des iPhone arbeiten:

- `Stage.autoOrients` – Ob die Anwendung so eingestellt ist, dass die Bühne automatisch neu ausgerichtet wird, wenn das Gerät gedreht wird. Diese Eigenschaft hat den Wert `true`, wenn im Flash Professional CS5-Dialogfeld „iPhone-Einstellungen“ die Option „Automatische Ausrichtung“ aktiviert ist. (Sie können auch das `autoOrients`-Element in der Anwendungsdeskriptordatei auf `true` einstellen.) Siehe „iPhone-Anwendungseinstellungen“ auf Seite 16. Sie können die automatische Neuausrichtung abbrechen, indem Sie einen `orientationChanging`-Ereignis-Listener für das Stage-Objekt hinzufügen. Der Aufruf der `preventDefault()`-Methode für dieses Ereignisobjekt bricht die automatische Neuausrichtung ab.

  Wenn Sie die automatische Ausrichtung verwenden, stellen Sie die `align`-Eigenschaft folgendermaßen ein, um beste Ergebnisse zu erzielen:

  ```
  stage.align = StageAlign.TOP_LEFT;
  stage.scaleMode = StageScaleMode.NO_SCALE;
  ```

- `Stage.deviceOrientation` – Die physische Ausrichtung des Geräts. Die StageOrientation-Klasse definiert Werte für diese Eigenschaft.
- `Stage.orientation` – Die aktuelle Ausrichtung der Bühne. Die StageOrientation-Klasse definiert Werte für diese Eigenschaft.
- `Stage.supportsOrientationChange` – Hat auf dem iPhone den Wert `true` und in einer AIR-Anwendung den Wert `false`.

- `Stage.setOrientation()` – Legt die Ausrichtung der Bühne fest. Diese Methode hat einen Parameter, einen String, der die neue Bühnenausrichtung definiert. Die Konstanten in der StageOrientation-Klasse definieren mögliche Werte für den Parameter.
- StageOrientation – Definiert Werte für die Bühnenausrichtung. `StageOrientation.ROTATED_RIGHT` gibt zum Beispiel eine Bühne an, die in Relation zur Standardausrichtung des Geräts nach rechts gedreht ist.
- StageOrientationEvent – Definiert Ereignisse, die die Bühne absetzt, wenn sich die Ausrichtung des Bildschirms ändert. Dieses Ereignis tritt auf, wenn der Benutzer das iPhone dreht. Es gibt zwei Arten von Ereignissen. Die Bühne setzt das `orientationChanging`-Ereignis ab, wenn das Gerät gedreht wird. Um die Neuausrichtung der Bühne zu verhindern, rufen Sie die `preventDefault()`-Methode des `orientationChanging`-Ereignisobjekts auf. Die Bühne setzt das `orientationChange`-Ereignis ab, wenn die Neuausrichtung der Bühne abgeschlossen ist.

Derzeit ist die Bildschirmausrichtungs-API nur bei AIR-Anwendungen für mobile Geräte sinnvoll. Wenn Quellcode für eine mobile AIR-Anwendung und eine Desktop-AIR-Anwendung genutzt wird, überprüfen Sie mit der `Stage.supportsOrientationChange`-Eigenschaft, ob die API unterstützt wird.

Das folgende Beispiel zeigt, wie auf die Drehung des Geräts durch den Benutzer reagiert wird:

```
stage.addEventListener(StageOrientationEvent.ORIENTATION_CHANGE,
            onOrientationChange);

function onOrientationChange(event:StageOrientationEvent):void
{
    switch (event.afterOrientation) {
        case StageOrientation.DEFAULT:
            // re-orient display objects based on
            // the default (right-side up) orientation.
            break;
        case StageOrientation.ROTATED_RIGHT:
            // Re-orient display objects based on
            // right-hand orientation.
            break;
        case StageOrientation.ROTATED_LEFT:
            // Re-orient display objects based on
            // left-hand orientation.
            break;
        case StageOrientation.UPSIDE_DOWN:
            // Re-orient display objects based on
            // upside-down orientation.
            break;
}
```

In diesem Beispiel stehen bei unterschiedlichen Bühnenausrichtungen Kommentare anstelle funktionierenden Codes.

Sie können die Ausrichtung der Bühne ändern, indem Sie die `setOrientation()`-Methode des Stage-Objekts aufrufen. Das Festlegen der Ausrichtung ist ein asynchroner Vorgang. Sie können überprüfen, wann die Ausrichtungsänderung abgeschlossen ist, indem Sie einen Listener für das `orientationChange`-Ereignis verwenden. Mit dem folgenden Code wird die Bühne an der rechten Seite ausgerichtet:

```
stage.addEventListener(StageOrientationEvent.ORIENTATION_CHANGE,
            onOrientationChange);
stage.setOrientation(StageOrientation.ROTATED_RIGHT);

function onOrientationChange(event:StageOrientationEvent):void
{
    // Code to handle the new Stage orientation
}
```

Während die Bühne gedreht wird, ändert sich die Größe und das Stage-Objekt setzt ein `resize`-Ereignis ab. Als Antwort auf das `resize`-Ereignis können Sie die Größe und Position der Anzeigeobjekte auf der Bühne ändern.

NativeApplication.systemIdleMode und SystemIdleMode

Mit der `NativeApplication.systemIdleMode`-Eigenschaft können Sie verhindern, dass das iPhone in den Leerlaufmodus wechselt. Standardmäßig wechselt das iPhone in den Leerlaufmodus, wenn der Touchscreen für einen bestimmten Zeitraum nicht verwendet wurde. Im Leerlaufmodus kann der Bildschirm dunkler werden. Möglicherweise wird das iPhone auch in den gesperrten Modus versetzt. Diese Eigenschaft kann auf einen von zwei Werten eingestellt werden:

- `SystemIdleMode.NORMAL` – Das iPhone folgt dem normalen Verhalten für den Leerlaufmodus.
- `SystemIdleMode.KEEP_AWAKE` – Die Anwendung versucht zu verhindern, dass das iPhone in den Leerlaufmodus wechselt.

Diese Funktionen wird nur auf mobilen Geräten unterstützt. In AIR-Anwendungen, die unter Desktopbetriebssystemen ausgeführt werden, wird sie nicht unterstützt. Wenn eine Anwendung auf dem Desktop ausgeführt wird, hat das Einstellen der `NativeApplication.systemIdleMode`-Eigenschaft keine Auswirkungen.

Der folgende Code zeigt, wie der iPhone-Leerlaufmodus deaktiviert wird:

```
NativeApplication.nativeApplication.systemIdleMode = SystemIdleMode.KEEP_AWAKE;
```

CameraRoll

Mit der CameraRoll-Klasse können Sie der iPhone Camera Roll ein Bild hinzufügen. Die `addBitmapData()`-Methode fügt ein Bild der iPhone Camera Roll hinzu. Die Methode weist einen Parameter auf, `bitmapData`. Dieser Parameter ist das BitmapData-Objekt, welches das Bild enthält, das der Camera Roll hinzugefügt werden soll.

Die CameraRoll-Funktion wird nur auf mobilen Geräten unterstützt. In AIR-Anwendungen, die unter Desktopbetriebssystemen ausgeführt werden, wird sie nicht unterstützt. Um zur Laufzeit festzustellen, ob die Anwendung die CameraRoll-Funktionalität unterstützt, überprüfen Sie die statische `CameraRoll.supportsAddBitmapData`-Eigenschaft.

Nachdem Sie die `addBitmapData()`-Methode aufgerufen haben, setzt das CameraRoll-Objekt eines von zwei Ereignissen ab:

- `complete` – Der Vorgang wurde erfolgreich abgeschlossen.
- `error` – Es ist ein Fehler aufgetreten. Es könnte zum Beispiel sein, dass auf dem iPhone nicht genügend Speicherplatz zum Speichern des Bildes frei ist.

Mit dem folgenden Code wird ein Bild der Bühne (eine Bildschirmerfassung) der Camera Roll hinzugefügt:

```
if (CameraRoll.supportsAddBitmapData)
{
    var cameraRoll:CameraRoll = new CameraRoll();
    cameraRoll.addEventListener(ErrorEvent.ERROR, onCrError);
    cameraRoll.addEventListener(Event.COMPLETE, onCrComplete);
    var bitmapData:BitmapData = new BitmapData(stage.stageWidth, stage.stageHeight);
    bitmapData.draw(stage);
    cameraRoll.addBitmapData(bitmapData);
}
else
{
    trace("not supported.");
}

function onCrError(event:ErrorEvent):void
{
    // Notify user.
}

function onCrComplete(event:Event):void
{
    // Notify user.
}
```

DisplayObject.cacheAsBitmapMatrix

Die `cacheAsBitmapMatrix`-Eigenschaft ist ein Matrix-Objekt, das definiert, wie ein Anzeigeobjekt dargestellt wird, wenn `cacheAsBitmap` auf `true` eingestellt ist. Die Anwendung verwendet diese Matrix als Transformationsmatrix, wenn die Bitmapversion des Anzeigeobjekts dargestellt wird.

Wenn `cacheAsBitmapMatrix` festgelegt wurde, behält die Anwendung ein zwischengespeichertes Bitmapbild, das mit dieser Matrix gerendert wurde, anstelle der Anzeigematrix. (Die Anzeigematrix ist der Wert der `transform.concatenatedMatrix`-Eigenschaft des Anzeigeobjekts.) Wenn diese Matrix nicht mit der Anzeigematrix übereinstimmt, wird die Bitmap wie erforderlich skaliert und gedreht.

Ein Anzeigeobjekt, dessen `cacheAsBitmapMatrix`-Eigenschaft festgelegt wurde, wird nur gerendert, wenn sich der Wert von `cacheAsBitmapMatrix` ändert. Die Bitmap wird skaliert oder gedreht, um der Anzeigematrix zu entsprechen.

Sowohl CPU- als auch GPU-basiertes Rendering profitiert von der Verwendung der `cacheAsBitmapMatrix`-Eigenschaft, das GPU-Rendering jedoch in größerem Maße.

Hinweis: *Um die Hardwarebeschleunigung zu verwenden, stellen Sie im Dialogfeld „iPhone-Einstellungen“ von Flash Professional CS5 auf der Registerkarte „Allgemein“ für „Rendering“ die Option „GPU“ ein. (Oder legen Sie die `renderMode`-Eigenschaft in der Anwendungsdeskriptordatei auf `gpu` fest.)*

Der folgende Code verwendet zum Beispiel eine nicht transformierte Bitmapdarstellung des Anzeigeobjekts:

```
matrix:Matrix = new Matrix(); // creates an identity matrix
mySprite.cacheAsBitmapMatrix = matrix;
mySprite.cacheAsBitmap = true;
```

Der folgende Code verwendet eine Bitmapdarstellung, die dem aktuellen Rendering entspricht:

```
mySprite.cacheAsBitmapMatrix = mySprite.transform.concatenatedMatrix;
mySprite.cacheAsBitmap = true;
```

Normalerweise reicht die Identitätsmatrix (`new Matrix()`) oder `transform.concatenatedMatrix` aus. Sie können jedoch auch eine andere Matrix, zum Beispiel eine verkleinerte Matrix, verwenden, um eine andere Bitmap an die GPU hochzuladen. Der folgende Code wendet zum Beispiel eine `cacheAsBitmapMatrix`-Matrix an, die um den Faktor 0,5 für die x- und y-Achsen skaliert wurde. Das Bitmapobjekt, das die GPU verwendet, ist kleiner; die GPU passt die Größe jedoch an, um der `transform.matrix`-Eigenschaft des Anzeigeobjekts zu entsprechen:

```
matrix:Matrix = new Matrix(); // creates an identity matrix
matrix.scale(0.5, 0.5); // scales the matrix
mySprite.cacheAsBitmapMatrix = matrix;
mySprite.cacheAsBitmap = true;
```

Im Allgemeinen wählen Sie eine Matrix, die das Anzeigeobjekt auf die Größe transformiert, in der es in der Anwendung erscheint. Wenn Ihre Anwendung zum Beispiel die Bitmapversion eines Sprites, das um die Hälfte verkleinert wurde, anzeigt, verwenden Sie eine Matrix, die um die Hälfte verkleinert wurde. Wenn die Anwendung das Sprite größer als mit den aktuellen Abmessungen anzeigt, verwenden Sie eine Matrix, die es um den entsprechenden Faktor vergrößert.

Es gibt in der Praxis eine Grenze für die Größe von Anzeigeobjekten, für die die `cacheAsBitmapMatrix`-Eigenschaft eingestellt ist. Die Grenze liegt bei 1020 mal 1020 Pixel. Es gibt in der Praxis eine Grenze für die Gesamtanzahl der Pixel aller Anzeigeobjekte, für die die `cacheAsBitmapMatrix`-Eigenschaft eingestellt ist. Diese Grenze liegt bei ungefähr vier Millionen Pixel.

Bei der Verwendung von `cacheAsBitmapMatrix` und Hardwarebeschleunigung müssen verschiedene Überlegungen berücksichtigt werden. Es ist wichtig zu wissen, für welche Anzeigeobjekte diese Eigenschaft festgelegt werden soll und für welche nicht. Wichtige Informationen zur Verwendung dieser Eigenschaft finden Sie im Abschnitt „Hardwarebeschleunigung“ auf Seite 38.

Mit der Diagnosefunktion für die GPU-Darstellung können Sie die GPU-Nutzung in Debug-Versionen Ihrer Anwendung testen. Weitere Informationen finden Sie unter „Debuggen von iPhone-Anwendungen“ auf Seite 26.

Hinweise zu Netzwerkverbindungen

Bei Verwendung der folgenden URL-Schemas mit der `nativigateToURL()`-Funktion wird ein Dokument in einer externen Anwendung geöffnet:

URL-Schema	Ergebnis des Aufrufs von nativeToURL()	Beispiel

mailto:	Öffnet eine neue Nachricht in der Mail-Anwendung.	`str = "mailto:test@example.com";`
		`var urlReq:URLReq = new URLRequest(str);`
		`navigateToURL(urlReq);`

sms:	Öffnet eine Nachricht in der Textnachrichtanwendung.	`str = "sms:1-415-555-1212";`
		`var urlReq:URLReq = new URLRequest(str);`
		`navigateToURL(urlReq);`

tel:	Wählt eine Telefonnummer (mit Genehmigung durch den Benutzer).	`str = "tel:1-415-555-1212";`
		`var urlReq:URLReq = new URLRequest(str);`
		`navigateToURL(urlReq);`

Eine iPhone-Anwendung ist während einer sicheren Transaktion, zum Beispiel einer https-Anfrage, unter Umständen auf installierte selbst signierte Stammzertifikate für die Serverauthentifizierung angewiesen. Ein Server sollte nicht nur das Endzertifikat senden, sondern auch alle Zwischenzertifikate, die die Verkettung mit dem Stammzertifikat bilden.

Spezielle ActionScript 3.0-APIs für Entwickler von mobilen Anwendungen

Die folgenden ActionScript 3.0-APIs definieren Funktionen, die für mobile Geräte hilfreich sind.

Beschleunigungssensor-API
Die folgenden Klassen ermöglichen es der Anwendung, Ereignisse vom Beschleunigungssensor des Geräts abzurufen:

- Accelerometer
- AccelerometerEvent

Weitere Informationen finden Sie unter Accelerometer-Eingabe.

Lokalisierungs-API
Die folgenden Klassen ermöglichen es der Anwendung, Ereignisse vom Ortungssensor des Geräts abzurufen:

- Geolocation
- GeolocationEvent

Weitere Informationen finden Sie unter Geolokalisierung.

Touch-, Multitouch- und Gesten-API
Die folgenden Klassen ermöglichen es der Anwendung, Berührungs- und Gestenereignisse zu empfangen:

- GestureEvent
- GesturePhase
- MultiTouch
- MultitouchInputMode
- TouchEvent
- TransformGestureEvent

Weitere Informationen finden Sie unter Touch-, Multitouch- und Gesteneingabe.

Kapitel 4: Überlegungen beim Entwerfen von iPhone-Anwendungen

Die Verarbeitungsgeschwindigkeit und die Bildschirmgröße des iPhone müssen beim Entwerfen und Programmieren berücksichtigt werden. Viele dieser Überlegungen gelten jedoch für die Entwicklung aller Anwendungen oder mobiler Anwendungen.

Weitere Informationen zum Optimieren von Anwendungen finden Sie unter Leistungsoptimierung für die Flash-Plattform. Dieses Dokument enthält zahlreiche Vorschläge zur Leistungsoptimieren für mobile Inhalte, Flash Player-Inhalte, AIR-Inhalte und ActionScript-basierte Inhalte allgemein. Die meisten dieser Vorschläge gelten auch für AIR-Anwendungen für das iPhone.

Wichtig: Viele dieser Überlegungen und Optimierungstechniken sind von wesentlicher Bedeutung für das Entwickeln von Anwendungen, die auf dem iPhone problemlos ausgeführt werden können.

Hardwarebeschleunigung

Sie können die OpenGL ES 1.1 Hardwarebeschleunigung verwenden, um die Grafikleistung in einigen Anwendungen zu verbessern. Spiele und andere Anwendungen mit animierten Anzeigeobjekten können von der Hardwarebeschleunigung profitieren. Anwendungen, die Hardwarebeschleunigung verwenden, können einige Grafikprozesse von der CPU auf die iPhone GPU verlagern, wodurch die Leistung erheblich verbessert werden kann.

Wenn Sie eine Anwendung entwickeln, die die GPU nutzen soll, ist die Einhaltung von Regeln wichtig, die sicherstellen, dass die Inhalt effektiv GPU-beschleunigt wird.

Um die Hardwarebeschleunigung zu verwenden, stellen Sie im Dialogfeld „iPhone-Einstellungen“ von Flash Professional CS5 auf der Registerkarte „Allgemein“ für „Rendering“ die Option „GPU“ ein. Sie können auch die `renderMode`-Eigenschaft in der Anwendungsdeskriptordatei auf `gpu` einstellen:

```
<initialWindow>
    <renderMode>gpu</renderMode>
    ...
```

Siehe „Festlegen von iPhone-Anwendungseigenschaften in Flash Professional CS5“ auf Seite 16 und „Festlegen von iPhone-Anwendungseigenschaften in der Anwendungsdeskriptordatei“ auf Seite 18.

Es gibt vier Klassen von Anzeigeobjekten, die mit Hardwarebeschleunigung schnell dargestellt werden, wenn sich ihre Inhalte nicht so häufig ändern:

- Bitmap-Objekte
- 2D-Anzeigeobjekte, deren `cacheAsBitmap`-Eigenschaft auf `true` eingestellt ist und deren `cacheAsBitmapMatrix`-Eigenschaft optional eingestellt ist (siehe unten)
- 3D-Anzeigeobjekte (die `z`-Eigenschaft ist eingestellt)
- Anzeigeobjekte mit einer einfarbigen rechteckigen Füllung und Rändern, die mit den Pixeln auf dem Bildschirm ausgerichtet sind.

Vektorbasierte Objekte werden neu dargestellt, wenn ein anderes Sprite über oder unter ihnen animiert wird. Jedes Objekt, das als Hintergrund oder Vordergrund für eine Animation dient, sollte deshalb ebenfalls einer dieser Kategorien angehören.

Bei Anzeigeobjekten, deren `cacheAsBitmap`-Eigenschaft auf `true` eingestellt ist, führt das Festlegen von `cacheAsBitmapMatrix` dazu, dass die GPU die Bitmap verwendet, die aus der Matrixtransformation resultiert. Die GPU verwendet die Bitmapdarstellung selbst dann, wenn das Objekt gedreht oder skaliert wird. Die GPU kann diese Bitmap viel schneller zusammensetzen und animieren, als die CPU ein vektorbasiertes Objekt neu zeichnen kann.

Wenn Sie `cacheAsBitmap` auf `true` einstellen, werden Anzeigeobjekte (und ihre untergeordneten Elemente) zwischengespeichert. Das Anzeigeobjekt wird nicht neu gezeichnet, wenn neue Bereiche sichtbar werden oder die gesamte Grafik versetzt wird.

Wenn die `cacheAsBitmapMatrix`-Eigenschaft eines Anzeigeobjekts gesetzt ist, kann die Anwendung eine Darstellung des Anzeigeobjekts erstellen, auch wenn dieses nicht sichtbar ist. Die Anwendung erstellt am Anfang des nächsten Bilds eine im Cache gespeicherte Darstellung des Anzeigeobjekts. Wenn Sie das Anzeigeobjekt dann der Bühne hinzufügen, kann es von der Anwendung schnell dargestellt werden. Außerdem kann die Anwendung das Objekt schnell animieren, drehen oder skalieren. Bei Objekten, die gedreht oder skaliert werden, setzen Sie die `cacheAsBitmap`-Eigenschaft nicht, ohne auch die `cacheAsBitmapMatrix`-Eigenschaft einzustellen.

Die Anwendung kann auch Alpha-Transformationen schnell ausführen, wenn ein Objekt als Bitmap zwischengespeichert wurde. Es werden jedoch nur `alpha`-Werte zwischen 0 und 1.0 für hardwarebeschleunigte Alpha-Transformationen unterstützt. Dies entspricht einer `colorTransform.alphaMultiplier`-Einstellung zwischen 0 und 256.

Stellen Sie die `cacheAsBitmap`-Eigenschaft nicht auf `true`, wenn es sich um häufig aktualisierte Objekte handelt, zum Beispiel Textfelder.

Anzeigeobjekte mit häufig geändertem grafischen Inhalt eignen sich im Allgemeinen nicht für die GPU-Darstellung. Dies gilt besonders bei älteren Geräten mit nicht so leistungsstarken GPUs. Aufgrund der Belastung durch das Hochladen der Grafiken zur GPU ist die CPU-Darstellung dann die bessere Wahl.

Strukturieren Sie Anzeigeobjekte mit untergeordneten Anzeigeobjekten, die sich in Relation zum übergeordneten Element bewegen, neu. Ändern Sie sie so, dass die untergeordneten Anzeigeobjekte dem übergeordneten Element gleichgestellt sind. Auf diese Weise haben sie jeweils ihre eigene Bitmapdarstellung. Außerdem kann sich jedes Anzeigeobjekt in Relation zu den anderen Anzeigeobjekten bewegen, ohne dass neue Grafiken zur GPU hochgeladen werden müssen.

Stellen Sie die `cacheAsBitmap`-Eigenschaft auf `true` auf der höchsten Ebene der Anzeigeliste, auf der untergeordnete Anzeigeobjekte nicht animiert werden. Anders ausgedrückt, stellen Sie sie für Anzeigeobjektcontainer ein, die keine bewegten Teile enthalten. Stellen Sie sie nicht für untergeordnete Anzeigeobjekte ein. Stellen Sie sie *nicht* für Sprites ein, die andere animierte Anzeigeobjekte enthalten.

Wenn Sie die `z`-Eigenschaft eines Anzeigeobjekts einstellen, verwendet die Anwendung immer eine zwischengespeicherte Bitmapdarstellung. Nachdem Sie die `z`-Eigenschaft eines Anzeigeobjekts eingestellt haben, verwendet die Anwendung die zwischengespeicherte Bitmapdarstellung auch dann, wenn Sie das Objekt drehen oder skalieren. Die Anwendung verwendet die `cacheAsBitmapMatrix`-Eigenschaft nicht für Anzeigeobjekte, deren `z`-Eigenschaft festgelegt wurde. Dieselben Regeln gelten, wenn Sie Eigenschaften eines dreidimensionalen Anzeigeobjekts festlegen, zum Beispiel `rotationX`, `rotationY`, `rotationZ` und `transform.matrix3D`.

Stellen Sie nicht die `scrollRect`- oder `mask`-Eigenschaft eines Anzeigeobjektcontainers ein, der Inhalt enthält, für den Sie Hardwarebeschleunigung verwenden möchten. Wenn Sie diese Eigenschaften festlegen, wird die Hardwarebeschleunigung für den Anzeigeobjektcontainer und seine untergeordneten Elemente deaktiviert. Als Alternative zum Festlegen der `mask`-Eigenschaft legen Sie ein Masken-Anzeigeobjekt über das maskierte Anzeigeobjekt.

Es gibt Größenbeschränkungen für die Anzeigeobjekte, die für die Hardwarebeschleunigung verfügbar sind. Bei älteren Geräten beträgt diese Grenze für Breite und Höhe jeweils 1024 Pixel oder weniger. Bei neueren Geräten liegt diese Grenze bei 2048 Pixel oder weniger. Mit dem Diagnosetool für die GPU-Darstellung können Sie die Leistung auf einem Gerät testen.

Die GPU verwendet iPhone-RAM zur Speicherung von Bitmapbildern. Sie belegt mindestens so viel Systemspeicher wie für die Bitmapbilder nötig.

Die GPU verwendet Speicherzuordnungen, die für jede Dimension des Bitmapbilds Vielfache von 2 betragen. Beispielsweise kann die GPU Speicher in den Größen 512 x 1024 oder 8 x 32 reservieren. So benötigt ein Bild in der Größe 9 x 15 Pixel genauso viel Speicher wie ein Bild in der Größe 16 x 16 Pixel. Für zwischengespeicherte Anzeigeobjekte sollten Sie am besten in jeder Richtung Abmessungen verwenden, die nahe an Potenzen von 2 (aber nicht mehr) sind. So ist es beispielsweise effizienter, ein Anzeigeobjekt in der Größe 32 x 16 Pixel zu verwenden als eines in der Größe 33 x 17.

Verlassen Sie sich nicht darauf, dass eine Bühne mit geänderter Größe Elemente verkleinert, deren Größe für andere Plattformen (zum Beispiel den Desktop) eingestellt wurde. Verwenden Sie stattdessen die `cacheAsBitmapMatrix`-Eigenschaft oder ändern Sie die Größe der Elemente vor dem Veröffentlichen für das iPhone. 3D-Objekte ignorieren die `cacheAsBitmapMatrix`-Eigenschaft, wenn ein Oberflächenbild zwischengespeichert wird. Aus diesem Grund ist es besser, die Größe von Anzeigeobjekten vor dem Veröffentlichen zu ändern, wenn sie auf einer 3D-Oberfläche dargestellt werden.

Die Vorteile der Hardwarebeschleunigung und der RAM-Verwendung müssen gegeneinander abgewogen werden. Während immer mehr Speicher belegt wird, informiert das iPhone-Betriebssystem andere, native iPhone-Anwendungen, die gerade ausgeführt werden, Speicher freizumachen. Wenn die Anwendungen diese Informationen verarbeiten und versuchen, Speicher freizugeben, können Sie mit Ihrer Anwendung um CPU-Zyklen konkurrieren. Dadurch kann die Leistung Ihrer Anwendung sich vorübergehend verschlechtern. Testen Sie Ihre Anwendung auf älteren Geräten, da auf diesen möglicherweise deutlich weniger Speicher für Ihren Prozess verfügbar ist.

Beim Debuggen der Anwendung auf dem iPhone können Sie die Diagnosefunktion für die GPU-Darstellung aktivieren. Diese Funktion unterstützt Sie dabei, zu sehen, wie Ihre Anwendung die GPU-Darstellung nutzt. Weitere Informationen finden Sie unter „Debuggen mit der GPU-Darstellungsdiagnose“ in „Debuggen von iPhone-Anwendungen“ auf Seite 26.

Informationen zur Verwendung der `cacheAsBitmapMatrix`-Eigenschaft finden Sie im Abschnitt „DisplayObject.cacheAsBitmapMatrix“ unter „ActionScript-APIs speziell für mobile AIR-Anwendungen“ auf Seite 32.

Andere Möglichkeiten zur Verbesserung der Leistung von Anzeigeobjekten

Hardwarebeschleunigung kann die Grafikleistung für bestimmte Klassen von Anzeigeobjekten verbessern. Nachstehend finden Sie Tipps zum Optimieren der Grafikleistung:

- Versuchen Sie, die Anzahl der sichtbaren Elemente auf der Bühne möglichst gering zu halten. Jedes Element braucht Zeit für die Darstellung und die Zusammensetzung mit den umgebenden Objekten.

  Wenn ein Anzeigeobjekt nicht länger angezeigt werden muss, setzen Sie seine `visible`-Eigenschaft auf `false` oder entfernen Sie es von der Bühne (mit `removeChild()`). Stellen Sie nicht einfach seine `alpha`-Eigenschaft auf 0.

- Vermeiden Sie Mischmodi allgemein und den Ebenenmischmodus im Besonderen. Verwenden Sie den normalen Mischmodus, wenn dies möglich ist.

- Anzeigeobjektfilter benötigen viel Rechenleistung. Verwenden Sie sie sparsam. Der Einsatz einiger Filter in einem Einführungsbildschirm ist zum Beispiel akzeptabel. Vermeiden Sie jedoch die Verwendung von Filtern für viele Objekte oder für Objekte, die animiert werden oder wenn Sie eine hohe Bildrate verwenden müssen.
- Vermeiden Sie Morph-Formen.
- Vermeiden Sie Beschneidungen.
- Setzen Sie den `repeat`-Parameter nach Möglichkeit auf `false`, wenn Sie die `Graphic.beginBitmapFill()`-Methode aufrufen.
- Weniger ist mehr. Verwenden Sie die Hintergrundfarbe als Hintergrund. Legen Sie große Formen nicht übereinander ab. Jedes einzelne Pixel, das gezeichnet werden muss, hat seinen Preis. Dies gilt besonders für Anzeigeobjekte, die nicht hardwarebeschleunigt sind.
- Vermeiden Sie Formen mit langen, dünnen Zacken, sich selbst schneidenden Rändern oder zahlreichen feinen Details an den Rändern. Diese Formen werden langsamer dargestellt als Formen mit glatten Rändern. Dies gilt besonders für Anzeigeobjekte, die nicht hardwarebeschleunigt sind.
- Erstellen Sie Bitmaps mit Größen, die kleiner als 2^n mal 2^m Bit sind, aber diese Werte fast erreichen. Die Abmessungen müssen keine Zweierpotenzen sein, aber sie sollten nahe an einer Zweierpotenz sein, jedoch nicht größer. Zum Beispiel wird ein Bild der Größe 31 x 15 Pixel schneller dargestellt als ein Bild der Größe 33 x 17. (31 und 15 sind knapp kleiner als die Zweierpotenzen 32 bzw. 16.) Derartige Bilder können auch Speicher effizienter nutzen.
- Beschränken Sie Anzeigeobjekte auf 1024 x 1024 Pixel (oder 2048 x 2048 Pixel bei neueren Geräten).

Informationsdichte

Die reale Bildschirmgröße ist bei mobilen Geräte kleiner als bei Desktops, die Pixeldichte ist jedoch höher. Scharfer Text ist für das Auge angenehm, die Glyphen müssen jedoch eine physische Mindestgröße aufweisen, um lesbar zu sein.

Mobile Geräte werden häufig unterwegs und bei schlechten Lichtverhältnissen verwendet. Bedenken Sie, wie viele Informationen realistischerweise lesbar auf dem Bildschirm angezeigt werden können. Dies kann weniger sein, als Sie auf einem Bildschirm mit den gleichen Pixelabmessungen auf einem Desktop anzeigen können.

Verwenden Sie typografische Hierarchien, um wichtige Informationen hervorzuheben. Betonen Sie die unterschiedliche Bedeutung von Elementen der Benutzeroberfläche durch Schriftgröße, Stärke, Platzierung und Abstände. Sie können einen oder mehrere Cues auf jeder Hierarchieebene verwenden. Wenden Sie diese Cues konsistent in der ganzen Anwendung an. Ein solches Signal kann räumlich (Einzüge, Zeilenabstand, Platzierung) oder grafisch (Größe, Stil und Farbe der Schrift) sein. Die Verwendung redundanter Signale ist eine effektive Möglichkeit, die Hierarchie auszudrücken. Versuchen Sie jedoch, mit höchstens drei Signalen für jede Ebene auszukommen.

Vereinfachen Sie nach Möglichkeit Bezeichnungen und Erklärungen. Verwenden Sie etwa eine Beispieleingabe in einem Textfeld, um den Inhalt vorzuschlagen. So brauchen Sie das Textfeld nicht separat zu beschriften.

Schriftarten und Texteingabe

Das beste Aussehen erzielen Sie mit Geräteschriftarten. Die folgenden Schriftarten sind zum Beispiel Geräteschriftarten auf dem iPhone:

- Serifen: Times New Roman, Georgia und _serif

- Ohne Serifen: Helvetica, Arial, Verdana, Trebuchet, Tahoma und _sans
- Feste Breite: Courier New, Courier und _typewriter

Verwenden Sie Schriftarten mit einer Größe von mindestens 14 Pixel.

Verwenden Sie Geräteschriftarten für bearbeitbare Textfelder. Geräteschriftarten in Textfeldern werden auch schneller dargestellt als eingebettete Schriftarten.

Verwenden Sie keinen unterstrichenen Text für Texteingabefelder. Legen Sie auch nicht die Ausrichtung des Textfelds fest. Eingabetextfelder auf dem iPhone unterstützen nur die linksbündige Ausrichtung (Standardeinstellung).

Wenn Sie in Flash Professional CS5 die TLF-Texteinstellung für ein Textfeld verwenden, deaktivieren Sie in den ActionScript 3.0-Einstellungen die gemeinsame Laufzeitbibliothek (Runtime Shared Library, RSL) in der Standardverknüpfung. Andernfalls funktioniert die Anwendung nicht auf dem iPhone, da sie versucht, die RSL-SWF-Datei zu verwenden.

1. Wählen Sie „Datei“ > „Einstellungen für Veröffentlichungen“.
2. Klicken Sie im Dialogfeld „Einstellungen für Veröffentlichungen“ auf die Registerkarte „Flash“.
3. Klicken Sie auf die Schaltfläche „Einstellungen“ neben der Dropdownliste „Skript“.
4. Klicken Sie auf die Registerkarte „Bibliothekspfad“.
5. Wählen Sie in der Dropdownliste „Standardverknüpfung“ den Eintrag „Im Code zusammengeführt“.

Ziehen Sie die Implementierung von Alternativen zu Texteingabefeldern in Betracht. Wenn Benutzer zum Beispiel einen numerischen Wert eingeben sollen, brauchen Sie kein Textfeld. Sie können einfach zwei Schaltflächen zum Erhöhen oder Verringern des Werts bereitstellen.

Beachten Sie den Platz, der von der virtuellen Tastatur benötigt wird. Wenn die virtuelle Tastatur aktiviert wird (zum Beispiel, wenn ein Benutzer auf ein Textfeld tippt), passt die Anwendung die Position auf der Bühne an. Die automatische Neupositionierung gewährleistet, dass das ausgewählte Eingabetextfeld sichtbar ist:

- Ein Textfeld am oberen Rand der Bühne wird zum oberen Rand des sichtbaren Bühnenbereichs verschoben. (Der sichtbare Bühnenbereich ist kleiner, weil die virtuelle Tastatur genügend Platz haben muss.)
- Ein Textfeld am unteren Rand der Bühne bleibt am unteren Rand des neuen Bühnenbereichs.
- Ein Textfeld in einem anderen Teil der Bühne wird zur vertikalen Mitte der Bühne verschoben.

Wenn der Benutzer auf ein Textfeld tippt, um es zu bearbeiten (und die Bildschirmtastatur angezeigt wird), setzt das TextField-Objekt ein `focusIn`-Ereignis ab. Sie können einen Ereignis-Listener für dieses Ereignis hinzufügen, um das Textfeld neu zu positionieren.

Ein einzeiliges Textfeld enthält eine Löschen-Schaltfläche (rechts neben dem Text), wenn der Benutzer den Text bearbeitet. Diese Löschen-Schaltfläche wird jedoch nicht angezeigt, wenn das Textfeld zu schmal ist.

Nachdem der Benutzer den Text in einem einzeiligen Textfeld bearbeitet hat, schließt er die Bildschirmtastatur, indem er auf die Schaltfläche „Fertig“ tippt.

Nach dem Bearbeiten von Text in einem mehrzeiligen Textfeld schließt der Benutzer die Bildschirmtastatur, indem er außerhalb des Textfelds auf den Bildschirm tippt. Damit wird der Fokus vom Textfeld genommen. Achten Sie beim Entwerfen der Anwendung darauf, dass bei der Anzeige der Bildschirmtastatur außerhalb des Textfelds noch Platz ist. Wenn das Textfeld zu groß ist, ist unter Umständen kein Bereich zum Tippen außerhalb des Textfelds sichtbar.

Die Verwendung bestimmter Flash Professional CS5-Komponenten verhindert, dass der Fokus von einem Textfeld genommen werden kann. Diese Komponenten sind für den Einsatz auf Desktopcomputern gedacht, wo dieses Fokusverhalten erwünscht ist. Eine dieser Komponenten ist die TextArea-Komponente. Wenn sie den Fokus hat (und bearbeitet wird), können Sie ihr den Fokus nicht durch das Klicken auf ein anderes Anzeigeobjekt nehmen. Auch die Platzierung einiger anderer Flash Professional CS5-Komponenten kann verhindern, dass der Fokus vom bearbeiteten Textfeld genommen wird.

Erstellen Sie keine Abhängigkeiten von Tastaturereignissen. Einige SWF-Inhalte, die für die Webverwendung entwickelt wurden, ermöglichen dem Benutzer zum Beispiel die Steuerung der Anwendung über die Tastatur. Auf dem iPhone ist die Bildschirmtastatur jedoch nur präsent, wenn der Benutzer ein Textfeld bearbeitet. Eine iPhone-Anwendung setzt nur dann Tastaturereignisse ab, wenn die Bildschirmtastatur angezeigt wird.

Speichern des Anwendungszustands

Ihre Anwendung kann jederzeit beendet werden (zum Beispiel, wenn das Telefon klingelt). Ziehen Sie das Speichern des Anwendungszustands bei jeder Änderung in Betracht. Sie können zum Beispiel Einstellungen in einer Datei oder Datenbank im Anwendungsspeicherverzeichnis speichern. Sie können Daten auch in einem lokalen gemeinsamen Objekt speichern. Dann können Sie den letzten Zustand der Anwendung wieder herstellen, wenn sie neu gestartet wird. Wenn eine Anwendung durch einen Telefonanruf unterbrochen wird, wird sie nach dem Ende des Gesprächs neu gestartet.

Verlassen Sie sich nicht darauf, dass das NativeApplication-Objekt ein `exiting`-Ereignis absetzt, wenn die Anwendung beendet wird; dies ist nicht unbedingt der Fall.

Änderungen der Bildschirmausrichtung

iPhone-Inhalte lassen sich im Hochformat oder im Querformat betrachten. Bedenken Sie, wie Ihre Anwendung mit Änderungen der Bildschirmausrichtung umgeht. Weitere Informationen finden Sie unter Erkennen der Geräteausrichtung.

Kollisionsbereiche

Überlegen Sie beim Entwerfen von Schaltfläche und anderen Elementen der Benutzeroberfläche, auf die der Benutzer tippt, wie groß der Kollisionsbereich (das Bedienziel) sein muss (dies ist der Bereich, den der Benutzer treffen muss, um die Schaltfläche zu aktivieren). Gestalten Sie diese Elemente groß genug, dass sie auf dem Touchscreen problemlos mit einem Finger bedient werden können. Achten Sie auch darauf, zwischen den Kollisionsbereichen genügend Platz zu lassen. Kollisionsbereiche sollten ungefähr 44 bis 57 Pixel groß sein.

Speicherzuordnung

Das Zuweisen von neuen Speicherblöcken ist aufwändig. Es kann die Anwendung verlangsamen oder die Leistung bei Animationen oder Interaktionen verschlechtern, da die Garbage Collection (Speicherräumung) ausgelöst wird.

Versuchen Sie, Objekte so oft wie möglich wiederzuverwenden, anstatt eines zu löschen und ein neues zu erstellen.

Bedenken Sie dabei, dass Vektorobjekte unter Umständen weniger Speicher benötigen als Arrays. Siehe Vektorklassen und Arrayklassen im Vergleich.

Weitere Informationen zur Speichernutzung finden Sie unter Einsparen von Speicher.

Zeichnungs-API

Versuchen Sie, die ActionScript-Zeichnungs-API (die Graphics-Klasse) beim Erstellen von Grafiken zu vermeiden. Wenn Sie die Zeichnungs-API verwenden, werden Objekte dynamisch auf der Bühne erstellt und dann für das Rastern dargestellt. Erstellen Sie diese Objekte stattdessen nach Möglichkeit beim Authoring statisch auf der Bühne.

Mit der Zeichnungs-API erstellte Objekte werden, wenn sie wiederholt aufgerufen werden, jedesmal gelöscht und neu erstellt, wenn ActionScript ausgeführt wird. Statische Objekte verbleiben jedoch über verschiedene Zeitleisten hinweg im Speicher.

Ereignis-Bubbling

Bei einem weit verschachtelten Anzeigeobjektcontainer kann das Bubbling (Aufsteigen) von Ereignissen aufwändig sein. Verringern Sie diesen Aufwand, indem Sie das Ereignis vollständig im Zielobjekt verarbeiten und dann die `stopPropagation()`-Methode des Ereignisobjekts aufrufen. Durch das Aufrufen dieser Methode wird das Bubbling des Ereignisses verhindert. Der Aufruf dieser Methode bedeutet auch, dass übergeordnete Objekte das Ereignis nicht empfangen.

Durch weniger tiefe Anzeigeobjektverschachtelung zur Vermeidung langer Ereignisketten lassen sich ebenfalls Vorteile erzielen.

Registrieren Sie nach Möglichkeit MouseEvent-Ereignisse anstelle von TouchEvent-Ereignissen. MouseEvent-Ereignisse belasten den Prozessor weniger als TouchEvent-Ereignisse.

Stellen Sie die Eigenschaften `mouseEnabled` und `mouseChildren` auf `false` ein, wenn dies möglich ist.

Optimieren der Videoleistung

Um die mobile Videowiedergabe zu optimieren, achten Sie darauf, dass möglichst keine anderen Vorgänge in Ihrer Anwendung ablaufen, während das Video abgespielt wird. So können die Prozesse zum Dekodieren und Darstellen des Videos so viel CPU-Leistung wie möglich beanspruchen.

Während der Videowiedergabe sollte wenig oder gar kein ActionScript-Code ausgeführt werden. Vermeiden Sie nach Möglichkeit die Ausführung von Code, der in kurzen Abständen oder auf der Zeitleiste ausgeführt wird.

Minimieren Sie das Neuzeichnen von Anzeigeobjekten außerhalb des Videos. Vermeiden Sie insbesondere das Neuzeichnen von Objekten, die sich mit dem Videobereich überschneiden. Dies gilt selbst dann, wenn die Objekte unter dem Video verborgen sind. Sie werden dennoch neu gezeichnet und benötigen Verarbeitungsressourcen. Verwenden Sie zum Beispiel einfache Formen für den Positionsanzeiger und aktualisieren Sie diesen nur wenige Male pro Sekunde statt in jedem Bild. Die Videosteuerelemente sollten den Videobereich nicht überlappen; positionieren Sie sie direkt darunter. Wenn Sie eine Animation für die Videopufferung verwenden, verstecken Sie sie nicht hinter dem Video, wenn Sie nicht verwendet wird, sondern machen Sie sie unsichtbar.

Flex- und Flash-Komponenten

Viele Flex-Komponenten und Flash-Komponenten wurden zur Verwendung in Desktopanwendungen entwickelt. Diese Komponenten, besonders Anzeigekomponenten zeigen auf mobilen Geräten unter Umständen keine gute Leistung. Neben der langsamen Leistung haben Desktopkomponenten möglicherweise auch Interaktionsmodelle, die für mobile Geräte nicht geeignet sind.

Adobe entwickelt eine für mobile Geräte optimierte Version des Flex-Frameworks. Weitere Informationen finden Sie unter http://labs.adobe.com/technologies/flex/mobile/.

Verringern der Größe von Anwendungsdateien

Nachstehend finden Sie einige Tipps zum Verringern der IPA-Dateigröße:

- Achten Sie darauf, dass Hintergrundbitmaps die richtige Größe haben (nicht größer als erforderlich).
- Überprüfen Sie, ob nicht benötigte Schriftarten eingebettet sind.
- Sehen Sie nach, ob sich in PNG-Elementen Alpha-Kanäle befinden und entfernen Sie sie, falls sie nicht benötigt werden. Verwenden Sie ein Dienstprogramm wie „PNG crunch“, um die Größe von PNG-Elementen zu verringern.
- Konvertieren Sie PNG-Elemente in JPG-Elemente, falls möglich.
- Ziehen Sie das Komprimieren von Sounddateien in Betracht (durch Verwendung einer niedrigeren Bitrate)
- Entfernen Sie alle Elemente, die nicht benötigt werden.