Tworzenie aplikacji ADOBE® AIR® za pomocą programu Packager for iPhone®

Informacje prawne

Informacje prawne można znaleźć na stronie http://help.adobe.com/pl_PL/legalnotices/index.html.

Data ostatniej aktualizacji 30/5/2017

Spis treści

Rozdział 1: Tworzenie aplikacji AIR dla telefonu iPhone — pierwsze kroki

Ważne pojęcia 1
Uzyskiwanie narzędzi programistycznych od firmy Adobe 4
Uzyskiwanie plików dla twórców aplikacji od firmy Apple 5
Tworzenie aplikacji Hello World na telefon iPhone w programie Flash Professional CS5 9

Rozdział 2: Kompilowanie i debugowanie aplikacji na telefon iPhone

Ikona telefonu iPhone i ikony ekranu początkowego 14
Ustawienia aplikacji na telefon iPhone 16
Kompilowanie pliku instalatora aplikacji na telefon iPhone 22
Instalowanie aplikacji na telefon iPhone 24
Debugowanie aplikacji na telefon iPhone 26
Przesyłanie aplikacji na telefon iPhone do sklepu App Store 28

Rozdział 3: Obsługa interfejsu API ActionScript 3.0 na urządzeniach mobilnych

Elementy interfejsu API ActionScript 3.0 nieobsługiwane na urządzeniach mobilnych 30
Elementy interfejsu API ActionScript przeznaczone do tworzenia aplikacji AIR na urządzenia mobilne 32
Elementy interfejsu API ActionScript 3.0 o szczególnym znaczeniu dla programistów tworzących aplikacje na urządzenia mobilne 37

Rozdział 4: Zasady projektowania aplikacji na telefon iPhone

Przyspieszanie sprzętowe 38
Inne sposoby zwiększania wydajności obiektów wyświetlanych 40
Gęstość informacji 41
Czcionki i wprowadzanie tekstu 42
Zapisywanie stanu aplikacji 43
Zmiany orientacji ekranu 43
Cele dotknięć i kliknięć 43
Alokacja pamięci 44
Interfejs API rysowania 44
Propagacja zdarzeń 44
Optymalizacja odtwarzania wideo 44
Składniki środowisk Flex i Flash 45
Redukowanie wielkości pliku aplikacji 45

Rozdział 1: Tworzenie aplikacji AIR dla telefonu iPhone — pierwsze kroki

Korzystając z narzędzi platformy Adobe® Flash® Platform oraz języka ActionScript® 3.0, można tworzyć aplikacje Adobe® AIR® dla telefonu iPhone oraz iPoda Touch. Aplikacje te są udostępniane, instalowane i uruchamiane podobnie jak inne aplikacje na telefon iPhone.

Uwaga: *W dalszej części tej dokumentacji zarówno urządzenie iPhone, jak i urządzenie iPod Touch, są określane po prostu jako „telefon iPhone”.*

Do programu Adobe® Flash® Professional CS5 dołączony jest program Packager for iPhone® Program Packager for iPhone kompiluje kod bajtowy ActionScript 3.0 w rodzimy kod aplikacji na telefon iPhone. Aplikacje na telefon iPhone są udostępniane jako pliki instalatora aplikacji na telefon iPhone (pliki .ipa), za pośrednictwem sklepu iTunes Store.

Do edycji kodu źródłowego aplikacji w języku ActionScript 3.0 można użyć programu Flash Professional CS5 lub Adobe® Flash® Builder™ 4.

Do tworzenia aplikacji dla telefonu iPhone można używać programu Flash Professional CS5.

Konieczne jest również uzyskanie certyfikatu programisty iPhone od firmy Apple.

Ważne: Przed przystąpieniem do opracowywania aplikacji na telefon iPhone należy przejrzeć informacje dotyczące projektowania aplikacji przeznaczonych na telefon iPhone. Więcej informacji zawiera sekcja „Zasady projektowania aplikacji na telefon iPhone” na stronie 38. Należy również zapoznać się z informacjami na temat plików dla twórców aplikacji wymaganych do zbudowania aplikacji na telefon iPhone. Patrz „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.

Ważne pojęcia

Przed przystąpieniem do tworzenia aplikacji na telefon iPhone w języku ActionScript 3.0 szczególnie ważne jest zrozumienie koncepcji i przebiegu pracy, jaką trzeba wykonać.

Słownik

Poniższe terminy mają szczególne znaczenie podczas tworzenia aplikacji na telefon iPhone:

Witryna iPhone Dev Center Witryna firmy Apple (http://developer.apple.com/iphone/) umożliwiająca:

- Złożenie wniosku o certyfikat programisty iPhone.
- Zarządzanie i tworzenie certyfikatów programisty iPhone, profili informacyjnych oraz identyfikatorów aplikacji (zdefiniowanych poniżej).
- Przesyłanie aplikacji do sklepu App Store.

Certyfikat programisty iPhone Służy do identyfikacji programisty na potrzeby tworzenia aplikacji.

Plik ten uzyskuje się od firmy Apple. Certyfikat ten należy przekonwertować do pliku certyfikatu P12 w celu podpisania aplikacji iPhone tworzonej w języku ActionScript 3.0. Więcej informacji zawiera sekcja *Plik certyfikatu P12*.

Do prostego debugowania i testowania aplikacji Flash Professional CS5 na komputerze programisty nie jest potrzebny certyfikat programisty iPhone. Będzie on jednak potrzebny do zainstalowania i przetestowania aplikacji na telefonie iPhone.

Certyfikat programisty różni się od certyfikatu dystrybucyjnego, który jest używany do budowy ostatecznej wersji aplikacji. Certyfikat dystrybucyjny uzyskuje się od firmy Apple z chwilą utworzenia finalnej wersji aplikacji.

Żądanie podpisania certyfikatu Plik zawierający informacje osobiste służące do generowania certyfikatu programisty. Znany również jako plik CSR.

Profil informacyjny Plik umożliwiający testowanie lub dystrybucję aplikacji na telefon iPhone. Pliki profili informacyjnych uzyskuje się od firmy Apple. Profil informacyjny jest przypisany do określonego certyfikatu programisty, identyfikatora aplikacji oraz jednego lub większej liczby identyfikatorów urządzeń. Dostępne są różne typy profili informacyjnych:

- **Programistyczny profil informacyjny** — służy do instalowania wersji testowej aplikacji na telefonie iPhone programisty.
- **Testowy profil informacyjny** — znany również jako profil informacyjny ad-hoc. Służy do dystrybucji wersji testowej aplikacji do wielu użytkowników (oraz urządzeń iPhone). Dzięki temu profilowi informacyjnemu oraz aplikacji testowej użytkownicy mogą przeprowadzać testy aplikacji bez ich przesyłania do sklepu App Store. Uwaga: Programistycznego profilu informacyjnego można również użyć do udostępnienia aplikacji testowych na wielu urządzeniach.
- **Dystrybucyjny profil informacyjny** — służy do budowania aplikacji na telefon iPhone w celu przesyłania aplikacji do sklepu App Store.

Identyfikator aplikacji Unikalny ciąg identyfikujący aplikację na telefon iPhone (lub wiele aplikacji) pochodzącą od określonego programisty. Identyfikatory aplikacji tworzy się w witrynie iPhone Dev Center. Każdy profil Tego identyfikatora aplikacji (lub wzoru) używa się do opracowywania aplikacji. Identyfikatora aplikacji używa się w oknie dialogowym ustawień telefonu iPhone programu Flash Professional CS5 (lub w pliku deskryptora aplikacji).

Identyfikatory aplikacji w witrynie iPhone Dev Center zawierają identyfikator pakunku poprzedzony identyfikatorem wartości początkowej pakunku. Identyfikator wartości początkowej pakietu to ciąg znaków, taki jak 5RM86Z4DJM, przypisywany przez firmę Apple do identyfikatora aplikacji. Identyfikator pakunku zawiera wybierany przez użytkownika ciąg znaków nazwy domeny wstecznej. Identyfikator pakunku może kończyć się znakiem gwiazdki (*) wskazującym wieloznaczny identyfikator aplikacji. Oto przykłady:

- 5RM86Z4DJM.com.example.helloWorld
- 96LPVWEASL.com.example.* (wieloznaczny identyfikator aplikacji)

W witrynie iPhone Dev Center można wyróżnić dwa typy identyfikatora aplikacji:

- Wieloznaczne identyfikatory aplikacji — W witrynie iPhone Dev Center te identyfikatory aplikacji kończą się symbolem gwiazdki (*), np. 96LPVWEASL.com.myDomain.* lub 96LPVWEASL.*. Używając profilu informacyjnego korzystającego z tego rodzaju identyfikatora aplikacji, można generować aplikacje testowe korzystające z identyfikatorów aplikacji odpowiadających temu wzorowi. W przypadku identyfikatora aplikacji możliwe jest zastąpienie symboli gwiazdki dowolnym ciągiem znaków poprawnych znaków. Na przykład, jeśli witryna iPhone Dev Center podaje wartość 96LPVWEASL.com.example.* jako identyfikatora aplikacji można użyć ciągu znaków com.example.foo lub com.example.bar jako identyfikatora aplikacji.

- Określone identyfikatory aplikacji — Definiują one unikalne identyfikatory aplikacji, jakie mają być w niej używane. W witrynie iPhone Dev Center te identyfikatory aplikacji nie kończą się symbolem gwiazdki. Przykład to 96LPVWEASL.com.myDomain.myApp. W przypadku profilu informacyjnego korzystającego z tego rodzaju identyfikatora aplikacji aplikacje muszą dokładnie odpowiadać identyfikatorom aplikacji. Na przykład, jeśli witryna iPhone Dev Center podaje ciąg znaków 96LPVWEASL.com.example.helloWorld jako identyfikatora aplikacji, konieczne jest użycie wartości com.example.foo jako identyfikatora aplikacji.

Podczas opracowywania aplikacji użytkownik określa identyfikator aplikacji w oknie dialogowym ustawień telefonu iPhone w programie Flash Professional CS5 (lub w pliku deskryptora aplikacji). Szczegółowe informacje na temat identyfikatora aplikacji zawiera sekcja „Karta Instalacja” „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5” na stronie 16 (lub „Ustawianie właściwości aplikacji na telefon iPhone w pliku deskryptora aplikacji” na stronie 18).

Ważne: Podczas określania identyfikatora aplikacji należy zignorować fragment identyfikatora aplikacji określający identyfikator wartości początkowej pakunku. Na przykład, jeśli zostanie podany identyfikator aplikacji 96LPVWEASL.com.example.bob.myApp, należy zignorować ciąg znaków 96LPVWEASL — zamiast tego jako identyfikatora aplikacji należy użyć wartości com.example.bob.myApp. Jeśli zostanie podany identyfikator aplikacji 5RM86Z4DJM.*, należy zignorować łańcuch 5RM86Z4DJM — jest to wieloznaczny identyfikator aplikacji.

Identyfikator aplikacji (lub wzór wieloznacznego identyfikatora aplikacji) powiązany z profilem informacyjnym można znaleźć w witrynie iPhone Dev Center (http://developer.apple.com/iphone). Przejdź do sekcji iPhone Developer Program Portal serwisu, a następnie do sekcji Provisioning.

Plik certyfikatu P12 Plik P12 (plik z rozszerzeniem .p12) to rodzaj pliku certyfikatu (Personal Information Exchange). Program Packager for iPhone korzysta z tego typu certyfikatu do budowy aplikacji na telefon iPhone. Certyfikat programisty otrzymany od firmy Apple konwertuje się do tej postaci certyfikatu.

Unikalny identyfikator urządzenia Unikalny kod identyfikujący określony telefon iPhone. Znany również jako UDID lub identyfikator urządzenia.

Obieg pracy przy tworzeniu programowania — przegląd

Proces opracowywania aplikacji dla telefonu iPhone przebiega następująco:

1. Zainstaluj program Flash Professional CS5 firmy Adobe.
2. Zainstaluj program iTunes.
3. Uzyskaj pliki dla twórców aplikacji od firmy Apple. Pliki te obejmują certyfikat programisty oraz profile informacyjne. Patrz „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.
4. Przekształć certyfikat programisty na plik certyfikatu P12. Program Flash CS5 wymaga, aby certyfikat był certyfikatem P12. Patrz „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.
5. Użyj programu iTunes do skojarzenia profilu informacyjnego z telefonem iPhone.
6. Zapisz aplikację w programie Flash Professional CS5.

   Szczególnie ważne jest poznanie najlepszych praktyk w zakresie projektowania i optymalizacji kodu na potrzeby aplikacji iPhone. Więcej informacji zawiera sekcja „Zasady projektowania aplikacji na telefon iPhone” na stronie 38.

   Ponadto niektóre elementy interfejsu API ActionScript 3.0 są obsługiwane w ograniczonym zakresie lub nie są w ogóle obsługiwane w telefonie iPhone. Patrz „Obsługa interfejsu API ActionScript 3.0 na urządzeniach mobilnych” na stronie 30.

   Do edycji kodu aplikacji w języku ActionScript 3.0 można użyć również programu Flash Builder 4.0.

   Można także użyć programu Flash Professional CS5 w celu przetestowania aplikacji na komputerze programisty.

7 Utwórz obraz ikony i obraz ekranu początkowego aplikacji. Każda aplikacja na telefon iPhone obejmuje zestaw ikon identyfikujących ją z perspektywy użytkowników. Obraz ekranu początkowego jest wyświetlany na telefonie iPhone w chwili ładowania programu. Patrz „Ikona telefonu iPhone i ikony ekranu początkowego” na stronie 14.

8 Dokonaj edycji ustawień dla telefonu iPhone. Ustawienia te obejmują następujące elementy:

- Elementy identyfikujące aplikację (w tym nazwa pliku, nazwa aplikacji, numer wersji oraz identyfikator aplikacji)
- Lokalizacja obrazu ikony źródłowej aplikacji
- Certyfikat P12 oraz profil informacyjny przypisany do aplikacji
- Początkowe proporcje aplikacji

W programie Flash Professional CS5 możliwe jest edytowanie tych ustawień w oknie dialogowym ustawień telefonu iPhone. Szczegółowe informacje zawiera sekcja „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5” na stronie 16.

Możliwe jest również edytowanie tych ustawień bezpośrednio w pliku deskryptora aplikacji. Więcej informacji zawiera sekcja „Ustawianie właściwości aplikacji na telefon iPhone w pliku deskryptora aplikacji” na stronie 18.

9 Skompiluj plik IPA, korzystając z programu Packager for iPhone. Patrz „Kompilowanie pliku instalatora aplikacji na telefon iPhone” na stronie 22.

10 Zainstaluj i przetestuj aplikację na telefonie iPhone. Zainstaluj plik IPA za pomocą programu iTunes.

W przypadku dystrybucji ad hoc powtórz ten proces ogólny, używając jednak testowego profilu informacyjnego zamiast programistycznego profilu informacyjnego. Na potrzeby finalnej dystrybucji aplikacji potwórz ten proces, używając dystrybucyjnego profilu informacyjnego. (Więcej informacji na temat różnych typów profili informacyjnych zawiera sekcja „Słownik” na stronie 1.)

Po utworzeniu wersji dystrybucyjnej aplikacji należy zapoznać się z treścią instrukcji w sekcji „Przesyłanie aplikacji na telefon iPhone do sklepu App Store” na stronie 28.

Samouczek prezentujący budowanie prostej aplikacji dla telefonu iPhone zawiera sekcja „Tworzenie aplikacji Hello World na telefon iPhone w programie Flash Professional CS5” na stronie 9.

Uzyskiwanie narzędzi programistycznych od firmy Adobe

Do tworzenia aplikacji dla telefonu iPhone przy użyciu języka ActionScript 3.0 potrzebny jest program Flash Professional CS5.

Ważne: Należy zaktualizować wersję wstępną programu Packager for iPhone dołączoną do programu Flash Professional CS5. W programie Flash Professional CS5 należy wybrać polecenie Pomoc > Aktualizacje.

Do edycji kodu aplikacji w języku ActionScript można użyć również programu Flash Builder 4. Program Flash Builder 4 jest dostępny pod adresem http://www.adobe.com/products/flashbuilder/.

Uzyskiwanie plików dla twórców aplikacji od firmy Apple

Programując aplikację na potrzeby telefonu iPhone, należy uzyskać od firmy Apple pliki dla twórców aplikacji. Konieczne jest uzyskanie certyfikatu programisty iPhone oraz profilu informacyjnego urządzenia mobilnego. Konieczne będzie również uzyskanie innych profili informacyjnych. Definicje tych plików zawiera „Słownik" na stronie 1.

Uwaga: *Uzyskanie tych plików to ważna część procesu opracowywania aplikacji. Należy zakończyć ten proces przed przystąpieniem do opracowywania aplikacji. Uzyskiwanie plików dla twórców aplikacji nie jest prostym procesem. Należy dokładnie przeczytać te instrukcje oraz instrukcje w witrynie iPhone Dev Center firmy Apple.*

Uzyskiwanie i praca z plikami dla twórców aplikacji iPhone

Konieczne jest uzyskanie certyfikatu programisty iPhone oraz profili informacyjnych od firmy Apple. Ponadto konieczna jest również konwersja certyfikatu na certyfikat P12.

Instalowanie programu iTunes

Program iTunes jest niezbędny do zainstalowania aplikacji na telefonie iPhone. Ponadto iTunes służy również do określenia identyfikatora urządzenia danego telefonu iPhone. Identyfikator urządzenia będzie potrzebny podczas starania się o certyfikat programisty iPhone.

Składanie wniosku o certyfikat programisty iPhone i tworzenie profilu informacyjnego

Zarejestruj się jako programista iPhone w witrynie iPhone Dev Center firmy Apple (http://developer.apple.com/iphone/), chyba że rejestracji dokonano już wcześniej.

Uwaga: *Do opracowywania aplikacji na telefon iPhone nie jest koniecznie potrzebny pakiet iPhone SDK ani środowisko XCode. Konieczne jest uzyskanie statusu zarejestrowanego programisty aplikacji na telefon iPhone. Konieczne jest uzyskanie certyfikatu programisty oraz profilu informacyjnego.*

1. Zaloguj się do witryny iPhone Dev Center, korzystając z identyfikatora konta programisty iPhone.
2. W witrynie iPhone Dev Center złóż wniosek (i dokonaj zakupu) certyfikatu programisty iPhone.

   Następnie otrzymasz od firmy Apple wiadomość e-mail zawierającą kod aktywacyjny w programie iPhone Developer Program.
3. Wróć do witryny iPhone Dev Center. Postępuj zgodnie z instrukcjami dotyczącymi aktywacji programu dla programistów (i po wyświetleniu monitu wprowadź swój kod aktywacyjny).
4. Po zaakceptowaniu kodu aktywacyjnego przejdź do sekcji iPhone Developer Program Portal witryny iPhone Dev Center.
5. Utwórz plik z wnioskiem o podpisanie certyfikatu. Plik ten posłuży do uzyskania certyfikatu programisty iPhone. Instrukcje można znaleźć w sekcji „Generowanie wniosku o podpisanie certyfikatu" na stronie 6.
6. W następnym kroku wymagane będzie podanie identyfikatora urządzenia (lub identyfikatora UDID telefonu iPhone. Identyfikator UDID można uzyskać za pośrednictwem iTunes:
   a. Podłącz telefon iPhone za pomocą kabla USB. Następnie, w programie iTunes, wybierz kartę podsumowania dla telefonu iPhone.
   b. Po pobraniu profilu informacyjnego z witryny Apple Dev Center dodaj go do programu iTunes.

c Kliknij wyświetlony numer seryjny. Identyfikator UDID zostanie teraz wyświetlony. Wybierz klawisze Command-C (na komputerze Mac) lub Control-C (na komputerze z systemem Windows) w celu skopiowania identyfikatora UDID do schowka.

7 Utwórz i zainstaluj profil informacyjny oraz certyfikat programisty iPhone.

Postępuj zgodnie z instrukcjami w witrynie iPhone Dev Center. Odszukaj instrukcje w sekcji iPhone Developer Program Portal. Może okazać się przydatne użycie asystenta Development Provisioning Assistant w celu uzyskania certyfikatu programisty i utworzenia profilu informacyjnego.

Zignoruj kroki dotyczące środowiska XCode. Nie jest konieczne używanie środowiska XCode do opracowywania aplikacji na telefon iPhone w programie Flash Professional CS5.

8 W programie iTunes wybierz kolejno opcje Plik > Dodaj do biblioteki. Następnie wybierz plik profilu informacyjnego (dla którego jako rozszerzenie nazwy pliku wybrano „mobileprovision"). Następnie przeprowadź synchronizację telefonu iPhone z programem iTunes.

Pozwala to przetestować aplikację skojarzoną z tym profilem informacyjnym w telefonie iPhone.

W celu sprawdzenia, czy określony profil informacyjny został dodany do programu iTunes, można spróbować dodać go do biblioteki. W przypadku wyświetlenia komunikatu z zapytaniem o zastąpienie istniejącego profilu informacyjnego można nacisnąć przycisk Anuluj. (Profil jest już zainstalowany.) Można również sprawdzić profile informacyjne zainstalowane w telefonie iPhone:

a Otwórz aplikację Ustawienia w telefonie iPhone.

b Otwórz kategorię Ogólne.

c Stuknij opcję Profile. Na stronie Profile zostaną wymienione zainstalowane profile informacyjne.

9 Jeśli jeszcze nie wykonano tej czynności, pobierz plik certyfikatu programisty iPhone (plik .cer).

W celu pobrania tego pliku możesz również skorzystać z łącza udostępnionego przez asystenta Development Provisioning Assistant. Plik można również znaleźć w sekcji Certyfikaty portalu informacyjnego w witrynie iPhone Dev Center firmy Apple (http://developer.apple.com/iphone/).

10 Następnie przekonwertuj certyfikat programisty iPhone na plik P12. Instrukcje zawiera sekcja „Konwersja certyfikatu programisty do pliku P12" na stronie 7.

Można teraz utworzyć prostą aplikację Hello World. Zobacz „Tworzenie aplikacji Hello World na telefon iPhone w programie Flash Professional CS5" na stronie 9.

Generowanie wniosku o podpisanie certyfikatu

Aby uzyskać certyfikat programisty, wygeneruj plik żądania podpisania certyfikatu, który zostanie następnie przesłany do witryny iPhone Dev Center firmy Apple.

Generowanie wniosku o podpisanie certyfikatu w systemie Mac OS

W systemie Mac OS możliwe jest użycie aplikacji Dostęp do pęku kluczy do wygenerowania wniosku o podpisanie certyfikatu. Aplikacja Dostęp do pęku kluczy stanowi podkatalog katalogu Narzędzia w katalogu Programy. W menu Dostęp do pęku kluczy wybierz opcję Asystent certyfikatu > Wniosek o wydanie certyfikatu z urzędu certyfikacji.

1 Otwórz aplikację Dostęp do pęku kluczy.

2 Z menu Dostęp do pęku kluczy wybierz opcję Preferencje.

3 W oknie dialogowym Preferencje kliknij opcję Certyfikaty. Następnie ustaw opcje Protokół statusu certyfikatów sieciowych oraz Lista unieważnień certyfikatów na wartość Wył. Zamknij okno dialogowe.

4 W menu Dostęp do pęku kluczy wybierz opcję Asystent certyfikatu > Wniosek o wydanie certyfikatu z urzędu certyfikacji.

5 Wprowadź adres e-mail oraz nazwę odpowiadającą identyfikatorowi konta programisty iPhone. Nie wprowadzaj adresu e-mail jednostki CA. Wybierz opcję Request is Saved to Disk (Wniosek jest zapisywany na dysku), a następnie kliknij przycisk Kontynuuj.

6 Zapisz plik (CertificateSigningRequest.certSigningRequest).

7 Załaduj plik CSR do witryny Apple iPhone Dev Center. (Patrz „Składanie wniosku o certyfikat programisty iPhone i tworzenie profilu informacyjnego”.)

Generowanie wniosku o podpisanie certyfikatu w systemie Windows

W przypadku programistów pracujących w systemie Windows może okazać się łatwiejsze uzyskanie certyfikatu programisty iPhone na komputerze z systemem Mac. Możliwe jest jednak uzyskanie certyfikatu na komputer z systemem Windows. Najpierw utwórz wniosek o podpisanie certyfikatu (plik CSR), korzystając z warstwy OpenSSL:

1 Zainstaluj warstwę OpenSSL na komputerze z systemem Windows. (Przejdź do witryny http://www.openssl.org/related/binaries.html.)

   Może być również potrzebne zainstalowanie plików redystrybuowalnych Visual C++ 2008, wymienionych na stronie pobierania protokołu Open SSL. (*Nie* ma potrzeby instalowania programu Visual C++ na posiadanym komputerze.)

2 Otwórz sesję wiersza poleceń Windows i przejdź (za pomocą polecenia CD) do podkatalogu bin katalogu OpenSSL (np. c:\OpenSSL\bin\).

3 Utwórz klucz prywatny, wprowadzając w wierszu poleceń:

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

   Zapisz ten plik klucza prywatnego. Będzie on potrzebny później.

   Korzystając z protokołu OpenSSL, nie ignoruj komunikatów o błędach. Mimo że protokół OpenSSL wygeneruje komunikat o błędzie, nadal może on generować pliki. Plik te mogą jednak okazać się bezużyteczne. W przypadku wyświetlenia błędów należy sprawdzić składnię, a następnie uruchomić polecenie ponownie.

4 Utwórz plik CSR, wprowadzając w wierszu poleceń:

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

   Zastąp adres e-mail, CN (nazwę certyfikatu) oraz C (kraj) własnymi wartościami.

5 Załaduj plik CSR do witryny Apple iPhone Dev Center. (Patrz „Składanie wniosku o certyfikat programisty iPhone i tworzenie profilu informacyjnego”.)

Konwersja certyfikatu programisty do pliku P12

Do tworzenia aplikacji na telefon iPhone w programie Flash Professional CS5 potrzebny jest plik certyfikatu P12. Certyfikat ten generuje się w oparciu o plik certyfikatu programisty Apple iPhone otrzymany od firmy Apple.

Konwertowanie certyfikatu programisty telefonu iPhone na plik P12 w systemie Mac OS

Po pobraniu certyfikatu Apple iPhone z telefonu Apple wyeksportuj go do formatu certyfikatu P12. Aby dokonać tego w systemie Mac® OS:

1 Otwórz aplikację Dostęp do pęku kluczy (w folderze Aplikacje/Narzędzia).

2 Jeśli wcześniej nie wykonano tej czynności, dodaj plik certyfikatu programisty do pęku kluczy, wybierając opcje Plik > Importuj. Następnie przejdź do pliku certyfikatu (plik .cer) uzyskanego od firmy Apple.

3 Wybierz kategorię Klucze w aplikacji Dostęp do pęku kluczy.

4 Wybierz klucz prywatny skojarzony z certyfikatem instalacyjnym iPhone.

Klucz prywatny jest identyfikowany przy użyciu skojarzonego z nim certyfikatu publicznego wystawionego dla podmiotu „iPhone Developer: <imię> <nazwisko>".

5 Wybierz polecenie Plik > Eksportuj elementy.

6 Zapisz klucz w formacie pliku .p12.

7 Zostanie wyświetlony monit o utworzenie hasła, które będzie używane podczas próby zaimportowania tego klucza na inny komputer.

Konwertowanie certyfikatu programisty firmy Apple na plik P12 w systemie Windows

Do opracowania aplikacji na telefon iPhone w programie Flash CS5 potrzebny jest plik certyfikatu P12. Certyfikat ten generuje się w oparciu o plik certyfikatu programisty Apple iPhone otrzymany od firmy Apple.

1 Plik certyfikatu programisty otrzymany od firmy Apple konwertuje się do pliku certyfikatu PEM. Uruchom następującą instrukcję wiersza poleceń z podkatalogu bin katalogu OpenSSL:

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

2 W przypadku używania klucza z pęku kluczy na komputerze z systemem Mac dokonaj jego konwersji na klucz PEM:

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

3 Możesz teraz wygenerować prawidłowy plik P12, bazujący na kluczu oraz wersji PEM certyfikatu programisty iPhone:

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

W przypadku używania klucza z pęku kluczy Mac OS użyj wersji PEM wygenerowanej w poprzednim kroku. W przeciwnym przypadku użyj klucza OpenSSL wygenerowanego wcześniej (w systemie Windows).

Zarządzanie certyfikatami, identyfikatorami urządzeń, identyfikatorami aplikacji oraz profilami informacyjnymi

Możliwe jest zarządzanie certyfikatami, identyfikatorami urządzeń, identyfikatorami aplikacji oraz profilami informacyjnymi w witrynie iPhone Dev Center firmy Apple (http://developer.apple.com/iphone/). Przejdź do sekcji iPhone Developer Program Portal witryny.

- Kliknij łącze Certyfikaty, aby zarządzać certyfikatami programistycznymi. Certyfikat można utworzyć, pobrać lub odwołać. Aby utworzyć certyfikat, musisz najpierw utworzyć wniosek o podpisanie certyfikatu. Patrz „Generowanie wniosku o podpisanie certyfikatu" na stronie 6.
- Kliknij łącze Urządzenia, aby zarządzać listą urządzeń, na których może być instalowana opracowywana aplikacja.
- Kliknij łącze identyfikatora aplikacji, aby zarządzać identyfikatorami aplikacji. Tworzony profil informacyjny jest zawsze powiązany z identyfikatorem aplikacji.
- Kliknij łącze Provisioning, aby zarządzać profilami informacyjnymi. Możesz również użyć asystenta Development Provisioning Assistant w celu utworzenia programistycznych profili informacyjnych.
- Kliknij łącze Dystrybucyjny, aby przesłać aplikację do sklepu App Store lub utworzyć wersję Ad Hoc aplikacji. Ta sekcja zawiera łącze do witryny iTunes Connect, która służy do przesyłania aplikacji do sklepu App Store.

Tworzenie aplikacji Hello World na telefon iPhone w programie Flash Professional CS5

Ważne: Przed utworzeniem aplikacji należy pobrać żądane aplikacje programisty i pliki. Patrz „Uzyskiwanie narzędzi programistycznych od firmy Adobe” na stronie 4 oraz „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.

Tworzenie projektu Flash Professional CS5

Aplikację na telefon iPhone można wygenerować bezpośrednio w programie Flash Professional CS5:

1. Otwórz program Flash CS5.
2. Wybierz polecenie Plik > Nowy.
3. Wybierz opcję iPhone OS.
4. Kliknij przycisk OK.

Dodaj treść do aplikacji.

Następnie dodaj tekst „Hello World!” do aplikacji:

1. Wybierz Narzędzie Tekst i kliknij stół montażowy.
2. W ustawieniach Właściwości dla pola tekstowego wybierz opcję Tekst klasyczny (nie zaś Tekst TLF).

   Jest to aplikacja podstawowa, i Tekst klasyczny jest opcją właściwą. W celu użycia opcji Tekst TLF konieczne jest zastosowanie także kilku innych ustawień. Patrz „Czcionki i wprowadzanie tekstu” na stronie 42.
3. W nowym polu tekstowym wpisz „Hello World!”
4. Zaznacz obszar pola tekstowego przy pomocy narzędzia Zaznaczanie.
5. Następnie otwórz Inspektora właściwości i dokonaj następujących ustawień:
   - Znak > Rodzina: _sans
   - Znak > Wielkość: 50
   - Położenie > X: 20
   - Położenie > Y: 20
6. Zapisz plik.
7. Wybierz kolejno opcje Sterowanie > Testuj film > W programie AIR Debug Launcher (zdalnie).

   Program Flash Professional CS5 kompiluje treść SWF i wyświetla wersję aplikacji w programie AIR Debug Launcher (ADL). Pozwala to na szybki podgląd aplikacji.

Tworzenie grafiki ikony i ekranu początkowego aplikacji

Wszystkie aplikacje na telefon iPhone mają ikony pojawiające się w interfejsie użytkownika aplikacji iTunes oraz na telefonie iPhone.

1. Utwórz katalog w katalogu projektu i nadaj mu nazwę icons.
2. Utwórz trzy pliki PNG w katalogu icons. Nadaj im nazwy: Icon29.png, Icon57.png i Icon512.png.

3 Zmodyfikuj pliki PNG, aby utworzyć odpowiednie grafiki dla aplikacji. Pliki muszą mieć rozmiary 29 na 29 pikseli, 57 na 57 pikseli oraz 512 na 512 pikseli. Na potrzeby tego testu możesz po prostu użyć jako grafiki kwadratów o jednolitym wypełnieniu.

W trakcie ładowania każdej aplikacji w telefonie iPhone wyświetlany jest ekran początkowy zawierający obraz. Aby zdefiniować obraz powitalny w pliku PNG.

1 W głównym katalogu na komputerze służącym do programowania utwórz plik PNG o nazwie Default.png. (*Nie* należy umieszczać tego pliku w podkatalogu icons. Należy koniecznie nazwać ten plik Default.png; nazwa powinna rozpoczynać się od wielkiej litery.)

2 Dokonaj edycji pliku, tak aby miał on szerokość 320 pikseli i wysokość 480 pikseli. Tymczasowo treść może stanowić zwykły biały prostokąt. (Zostanie to zmienione później.)

Uwaga: *Przesyłając aplikację do sklepu Apple App Store, korzysta się z wersji JPG (nie zaś wersji PNG) pliku 512 pikseli. Do testowania wersji roboczych aplikacji używa się wersji PNG.*

Szczegółowe informacje na temat tych grafik zawiera sekcja „Ikona telefonu iPhone i ikony ekranu początkowego” na stronie 14.

Edytowanie ustawień aplikacji

Ważne: Jeśli wcześniej nie wykonano tej czynności, należy pobrać wymagane aplikacje programistyczne i pliki iPhone. Patrz „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.

W oknie dialogowym ustawień dla telefonu iPhone w programie Flash Professional CS5 zdefiniuj podstawowe właściwości aplikacji na telefon iPhone.

1 Wybierz polecenie Plik > Ustawienia iPhone OS.

2 Na karcie Ogólne wprowadź następujące ustawienia:

- Plik wyjściowy: HelloWorld.ipa

  Jest to nazwa generowanego pliku instalatora iPhone.

- Nazwa aplikacji: Hello World

  Nazwa aplikacji wyświetlana pod ikoną aplikacji w telefonie iPhone.

- Wersja: 1.0

  Wersja aplikacji.

- Proporcje: pionowy
- Pełny ekran: zaznaczona.
- Automatyczna orientacja: niezaznaczone.
- Rendering: CPU

  W przypadku pozostałych opcji, GPU i automatycznego, do renderingu używany jest akcelerator sprzętowy. Dzięki tej funkcji można uzyskać wyższą wydajność aplikacji intensywnych graficznie (takich jak gry) zaprojektowanych tak, by mogły wykorzystywać akcelerację sprzętową. Więcej informacji można znaleźć w sekcji „Przyspieszanie sprzętowe” na stronie 38.

- Dołączone pliki: Dodaj początkowy plik kompozycji (Default.png) do listy Dołączone pliki.

Uwaga: *Na potrzeby tego przykładu Hello World nie należy modyfikować ustawień podanych w tej instrukcji. Niektóre ustawienia, takie jak Wersja, mają określone ograniczenia. Ograniczenia te omówiono w sekcji „Ustawienia aplikacji na telefon iPhone” na stronie 16.*

3 Na karcie Instalacja dokonaj następujących ustawień:

- Certyfikat: przeglądaj i wybierz certyfikat .p12 w oparciu o certyfikat programisty uzyskany od firmy Apple.

  Certyfikat ten służy do podpisywania pliku. Konieczna jest konwersja certyfikatu Apple iPhone do formatu .p12. Więcej informacji można znaleźć w sekcji „Uzyskiwanie narzędzi programistycznych od firmy Adobe" na stronie 4.

- Hasło: wprowadź hasło dla certyfikatu.
- Plik informacyjny: przeglądaj i wybierz plik informacyjny programisty uzyskany od firmy Apple. Patrz „Uzyskiwanie plików dla twórców aplikacji od firmy Apple" na stronie 5.
- Id. aplikacji: jeśli można wybrać to pole, możesz wprowadzić identyfikator aplikacji zgodny z identyfikatorem aplikacji udostępnionym firmie Apple (np. com.example.as3.HelloWorld).

  Identyfikator aplikacji identyfikuje aplikację w sposób unikalny.

  Jeśli nie można wybrać tego pola, wówczas profil informacyjny jest powiązany z określonym identyfikatorem aplikacji. Identyfikator aplikacji jest wyświetlany w polu.

  Szczegółowe informacje dotyczące określania identyfikatora aplikacji zawiera sekcja „Karta Instalacja" w sekcji „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5" na stronie 16.

4 Na karcie Ikony kliknij opcję Ikona 29 x 29 na liście Ikony. Następnie podaj lokalizację utworzonego wcześniej pliku PNG o wielkości 29 x 29 pikseli (patrz „Tworzenie grafiki ikony i ekranu początkowego aplikacji" na stronie 9). Następnie określ pliki PNG na potrzeby ikony o wielkości 57 x 57 pikseli oraz ikony o wielkości 512 x 512 pikseli.

5 Kliknij przycisk OK.

6 Zapisz plik.

Więcej informacji na temat ustawień aplikacji można znaleźć w sekcji „Ustawienia aplikacji na telefon iPhone" na stronie 16.

Kompilowanie pliku IPA

Można teraz skompilować plik instalatora IPA:

1 Wybierz opcję Plik > Publikuj.

2 W oknie dialogowym ustawień telefonu iPhone kliknij przycisk OK.

Program Packager for iPhone wygeneruje plik instalatora aplikacji na telefon iPhone, plik HelloWorld.ipa, w katalogu projektu. Kompilowanie pliku IPA może zająć kilka minut.

Zainstaluj aplikację na telefonie iPhone.

Aby zainstalować aplikację na telefon iPhone na potrzeby testowania na telefonie iPhone:

1 Otwórz aplikację iTunes.

2 Jeśli wcześniej nie wykonano tej czynności, dodaj plik informacyjny na potrzeby tej aplikacji do programu iTunes. W programie iTunes wybierz kolejno opcje Plik > Dodaj do biblioteki. Następnie wybierz plik profilu informacyjnego (dla którego jako typ pliku wybrano „mobileprovision").

   Do testowania aplikacji na telefonie iPhone programisty użyj programistycznego profilu informacyjnego.

W dalszej części, podczas przekazywania aplikacji do sklepu iTunes Store, użyj profilu dystrybucyjnego. W celu dystrybucji aplikacji ad-hoc (na wiele urządzeń, z wyłączeniem pośrednictwa sklepu iTunes Store), należy użyć profilu informacyjnego ad-hoc.

Więcej informacji na temat profili informacyjnych można znaleźć w sekcji „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.

3. Niektóre wersje iTunes nie dopuszczają do zastępowania aplikacji w sytuacji, gdy ta sama jej wersja jest już zainstalowana. W takim przypadku należy usunąć aplikację z urządzenia oraz z listy aplikacji w iTunes.
4. Kliknij dwukrotnie plik IPA dla swojej aplikacji. Powinien on pojawić się na liście aplikacji.
5. Podłącz swój telefon iPhone do portu USB komputera.
6. W iTunes sprawdź kartę Aplikacja dla urządzenia i upewnij się, że aplikacja została wybrana z listy aplikacji do zainstalowania.
7. Wybierz urządzenie z listy po lewej stronie aplikacji iTunes. Następnie kliknij przycisk Synchronizacja. Po zakończeniu synchronizacji aplikacja Hello World pojawi się na telefonie iPhone.

Jeśli nowa wersja nie została zainstalowana, usuń ją z telefonu iPhone i z listy aplikacji w iTunes, a następnie wykonaj tę procedurę ponownie. Może mieć to miejsce, jeśli obecnie zainstalowana wersja ma ten sam identyfikator aplikacji i numer wersji.

Jeśli w programie iTunes podczas próby zainstalowania aplikacji zostanie wyświetlony błąd, należy zapoznać się z sekcją „Rozwiązywanie problemów z instalacją aplikacji” w punkcie „Instalowanie aplikacji na telefon iPhone” na stronie 24.

Edycja grafiki ekranu początkowego

Przed skompilowaniem aplikacji utworzony został plik Default.png (patrz „Tworzenie grafiki ikony i ekranu początkowego aplikacji” na stronie 9). Obraz zawarty w tym pliku PNG jest wyświetlany w trakcie ładowania aplikacji. Podczas testowania aplikacji na telefonie iPhone można było zaobserwować pusty ekran — to właśnie był obraz początkowy.

Należy teraz zmienić go odpowiednio do potrzeb tworzonej aplikacji („Hello World!”):

1. Otwórz aplikację na urządzeniu. Po pojawieniu się po raz pierwszy napisu „Hello World” naciśnij i przytrzymaj przycisk Home (poniżej ekranu). Przytrzymując przycisk Home, naciśnij przycisk Power/Sleep (w górnej części telefonu iPhone). Zrzut ekranu zostanie przesłany do folderu Rolka z aparatu.
2. Prześlij obraz na komputer programisty, wysyłając zdjęcia z programu iPhoto lub innej aplikacji do transferu zdjęć. (W systemie Mac OS można również skorzystać z aplikacji Pobieranie obrazu.)

   Zdjęcie można także przesłać na komputer programisty za pomocą poczty elektronicznej.

   - Otwórz aplikację Zdjęcia.
   - Otwórz Rolkę z aparatu.
   - Otwórz obraz zrzutu ekranu.
   - Stuknij obraz, a następnie przycisk (strzałkę) „dalej” w dolnym lewym rogu. Następnie kliknij przycisk Wyślij zdjęcie (email), aby wysłać obraz do siebie.
3. Zastąp plik Default.png (w katalogu programistycznym) wersją PNG obrazu zrzutu ekranowego.

4 Ponownie skompiluj aplikację (patrz „Kompilowanie pliku IPA” na stronie 11) i zainstaluj ją ponownie na telefonie iPhone.

Podczas ładowania aplikacji jest teraz wyświetlany nowy ekran powitalny.

Uwaga: *Możliwe jest utworzenie dowolnej grafiki do zapisania w pliku Default.png; jedynym warunkiem są prawidłowe wymiary (320 na 480 pikseli). Najlepiej jednak by treść pliku Default.png odpowiadała początkowemu stanowi aplikacji.*

Rozdział 2: Kompilowanie i debugowanie aplikacji na telefon iPhone

Możliwe jest skompilowanie aplikacji na telefon iPhone za pomocą programu Packager for iPhone. Do programu Adobe Flash Professional CS5 dołączony jest program Packager for iPhone

Aplikacje można debugować na komputerze programisty. Można również zainstalować wersję do debugowania na telefonie iPhone, a następnie odebrać wynik metody `trace()` w programie Flash Professional CS5.

Samouczek dotyczący sposobu budowania aplikacji iPhone od początku do końca zawiera sekcja „Tworzenie aplikacji Hello World na telefon iPhone w programie Flash Professional CS5” na stronie 9.

Ikona telefonu iPhone i ikony ekranu początkowego

Wszystkie aplikacje na telefon iPhone mają ikony pojawiające się w interfejsie użytkownika aplikacji iTunes oraz na telefonie iPhone.

Ikony aplikacji iPhone

Na potrzeby aplikacji dla telefonu iPhone definiuje się następujące ikony:

- Ikona o wymiarach 29 na 29 pikseli — ikona wyników wyszukiwania Spotlight na urządzeniach iPhone oraz iPod Touch.
- Ikona o wymiarach 48 na 48 pikseli — ikona wyników wyszukiwania Spotlight na tablecie iPad.
- Ikona o wymiarach 57 na 57 pikseli — ikona przeznaczona na ekran początkowy urządzeń iPhone i iPod Touch.
- Ikona o wymiarach 72 na 72 piksele (opcjonalna) — ikona przeznaczona na ekran początkowy tabletu iPad.
- Ikona o wymiarach 512 na 512 pikseli — wyświetlana w programie iTunes. 512-pikselowy plik PNG używany jest tylko do testowania wersji roboczych aplikacji. Przesyłając ostateczną wersję aplikacji do serwisu Apple App Store, obraz 512-pikselowy dołącza się osobno jako plik JPG. Nie jest on uwzględniany w pliku IPA.

W programie Flash Professional CS5 należy dodać te ikony do karty Ikony w oknie dialogowym Ustawienia telefonu iPhone. Zobacz „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5” na stronie 16.

Do pliku deskryptora aplikacji można także dodać położenia ikon:

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

Telefon iPhone wzbogaci ikonę o efekt poświaty. Nie jest konieczne uwzględnianie go w obrazie źródłowym. W celu eliminacji tego efektu domyślnego należy do elementu `InfoAdditions` w pliku deskryptora aplikacji dodać następujący kod.

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

Więcej informacji można znaleźć w sekcji „Ustawianie właściwości aplikacji na telefon iPhone w pliku deskryptora aplikacji” na stronie 18.

Kompozycja ekranu początkowego (Default.png)

W trakcie ładowania każdej aplikacji w telefonie iPhone wyświetlany jest ekran początkowy zawierający obraz. Ten obraz powitalny w pliku PNG ma nazwę Default.png. W głównym katalogu na komputerze służącym do programowania utwórz plik PNG o nazwie Default.png. (*Nie* należy umieszczać tego pliku w podkatalogu. Należy koniecznie nazwać ten plik Default.png; nazwa powinna rozpoczynać się od wielkiej litery.)

Plik Default.png ma szerokość 320 pikseli oraz wysokość 480 pikseli, niezależnie od początkowej orientacji aplikacji oraz tego, czy jest on wyświetlany na pełnym ekranie, czy nie.

Jeśli początkowa orientacja aplikacji jest pozioma, należy użyć tych samych wymiarów, jakich używa aplikacja w orientacji pionowej: 320 pikseli szerokości na 480 pikseli wysokości. Należy jednak obrócić grafikę w pliku PNG o 90° w kierunku przeciwnym do ruchu wskazówek zegara. Lewa strona grafiki PNG odpowiada górnej części ekranu telefonu iPhone w orientacji poziomej. (Więcej informacji na temat ustawiania początkowej orientacji aplikacji można znaleźć w sekcji „Ustawienia aplikacji na telefon iPhone” na stronie 16.)

W przypadku aplikacji niepełnoekranowych ignorowanych jest górnych 20 pikseli domyślnego obrazu grafiki. W górnym, prostokątnym pasie ekranu telefonu iPhone o szerokości 20 pikseli, wyświetlany jest pasek stanu, przesłaniając tym samym obraz domyślny. W aplikacjach o orientacji poziomej ten obszar odpowiada lewemu prostokątowi o szerokości 20 pikseli w pliku Default.png (wyświetlanemu w górnej części w orientacji poziomej). W aplikacji o orientacji pionowej ten obszar to prostokąt o szerokości 20 pikseli w pliku Default.png.

W przypadku większości aplikacji obraz Default.png powinien odpowiadać ekranowi początkowemu aplikacji. Aby wykonać zrzut ekranu początkowego aplikacji:

1. Otwórz aplikację na telefonie iPhone. Po pojawieniu się pierwszego ekranu interfejsu użytkownika naciśnij i przytrzymaj przycisk Home (pod ekranem). Przytrzymując przycisk Home, naciśnij przycisk Power/Sleep (w górnej części urządzenia). Zrzut ekranu zostanie przesłany do Rolki z aparatu.
2. Prześlij obraz na komputer programisty, wysyłając zdjęcia z programu iPhoto lub innej aplikacji do transferu zdjęć. (W systemie Mac OS można również skorzystać z aplikacji Pobieranie obrazu.)

   Zdjęcie można także przesłać na komputer programisty za pomocą poczty elektronicznej.

   - Otwórz aplikację Zdjęcia.
   - Otwórz Rolkę z aparatu.
   - Otwórz obraz zrzutu ekranu.
   - Stuknij obraz, a następnie przycisk (strzałkę) „dalej” w dolnym lewym rogu. Następnie kliknij przycisk Wyślij zdjęcie (email), aby wysłać obraz do siebie.

Uwaga: *Możliwe jest utworzenie dowolnej grafiki do zapisania w pliku Default.png; jedynym warunkiem są prawidłowe wymiary. Najlepiej jednak by treść pliku Default.png odpowiadała początkowemu stanowi aplikacji.*

Nie należy zawierać w pliku obrazu Default.png tekstów, jeśli aplikacja ma być lokalizowana na kilka języków. Plik Default.png jest plikiem statycznym i w takiej sytuacji jego treść byłaby nieodpowiednia dla innych wersji językowych.

W programie Flash Professional CS5 koniecznie dodaj plik Default.png do listy Dołączone pliki w oknie dialogowym ustawień telefonu iPhone. Zobacz „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5” na stronie 16.

Kompilując aplikację przy użyciu narzędzia PFI wywoływanego z wiersza poleceń, należy uwzględnić ten plik w liście dołączanych zasobów. Więcej informacji można znaleźć również w sekcji „Tworzenie pliku instalatora aplikacji na telefon iPhone z wiersza poleceń” na stronie 23.

Ustawienia aplikacji na telefon iPhone

Ustawienia aplikacji obejmują:

- Nazwę aplikacji
- Nazwę pliku IPA
- Wersję aplikacji
- Początkową orientację ekranu aplikacji oraz informację, czy orientacja ekranu obraca się automatycznie po obróceniu telefonu iPhone
- Informację, czy widok początkowy jest widokiem pełnoekranowym, czy nie
- Informację o ikonach aplikacji
- Informację o akceleracji sprzętowej

Ustawienia aplikacji można edytować w programie Flash Professional CS5.

Możliwe jest również edytowanie tych ustawień w pliku deskryptora aplikacji. Plik deskryptora aplikacji jest plikiem XML zawierającym ustawienia dla aplikacji.

Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5

Okno dialogowe ustawień dla telefonu iPhone w programie Flash Professional CS5 umożliwia definiowanie wielu podstawowych właściwości aplikacji na telefon iPhone.

Aby otworzyć okno dialogowe Ustawienia telefonu iPhone:

❖ Wybierz polecenie Plik > Ustawienia telefonu iPhone.

Karta Ogólne

Karta Ogólne zawiera następujące ustawienia związane z telefonem iPhone:

- Plik wyjściowy — Nazwa aplikacji wyświetlana pod ikoną aplikacji w telefonie iPhone. W nazwie pliku wyjściowego nie należy uwzględniać znaku plus (+).
- Nazwa programu — Nazwa aplikacji wyświetlana pod ikoną aplikacji w telefonie iPhone. W nazwie aplikacji nie należy używać znaku plus (+).
- Wersja — Ułatwia użytkownikom stwierdzenie, którą wersję aplikacji instalują. Ta wersja jest używana jako obiekt CFBundleVersion aplikacji na telefon iPhone. Musi mieć ona format zbliżony do: nnnnn[.nn[.nn]] (gdzie n jest cyfrą z zakresu 0–9, a nawiasy oznaczają elementy opcjonalne, np. 1, 1.0 czy 1.0.1. Wersje telefonu iPhone muszą zawierać tylko cyfry i kropki dziesiętne. Wersje telefonu iPhone mogą zawierać maksymalnie dwie kropki dziesiętne.

- Proporcje — Początkowe proporcje aplikacji (układ pionowy lub poziomy).
- Pełny ekran — Określa, czy aplikacja jest wyświetlana na pełnym ekranie, czy wyświetla pasek statusu telefonu iPhone.
- Automatyczna orientacja — Wybór tej aplikacji pozwoli na zmianę orientacji wyświetlania treści na ekranie telefonu iPhone z chwilą zmiany orientacji samego urządzenia.

  W przypadku korzystania z funkcji automatycznej orientacji w celu uzyskania najlepszych wyników należy dodać kod ActionScript i ustawić właściwość `align` stołu montażowego na następującą wartość:

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

- Rendering — Sposób, w jaki obiekty wyświetlane są renderowane w telefonie iPhone:
  - CPU — Aplikacja korzysta z procesora CPU do renderowania wszystkich obiektów wyświetlanych. Nie stosuje się akceleratora sprzętowego.
  - GPU — Aplikacja korzysta z procesora GPU telefonu iPhone do kompozycji bitmap.
  - Auto — Ta funkcja nie została zaimplementowana.

  Więcej informacji można znaleźć w sekcji „Przyspieszanie sprzętowe” na stronie 38.
- Dołączone pliki — Umożliwia dodawanie plików i katalogów do pakietu aplikacji na telefon iPhone. Główny plik SWF oraz plik deskryptora aplikacji są dołączone domyślnie. Dodaj wszelkie pozostałe zasoby do listy Dołączone pliki. Koniecznie dodaj początkowy plik kompozycji (Default.png) do listy Dołączone pliki.

Karta Instalacja

Karta Instalacja zawiera ustawienia dotyczące podpisywania i kompilacji aplikacji:

- Podpis cyfrowy iPhone — Określa plik certyfikatu P12 oraz hasło do certyfikatu. Konieczna jest konwersja certyfikatu Apple iPhone do formatu .p12. Więcej informacji można znaleźć w sekcji „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.
- Plik udostępniania usługi — Wskazuje plik udostępniania usługi (profil informacyjny) dla tej aplikacji uzyskany od firmy Apple. Więcej informacji można znaleźć w sekcji „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.
- Identyfikator aplikacji — Identyfikator aplikacji identyfikuje aplikację w sposób unikalny. Jeśli plik informacyjny jest powiązany z określonym identyfikatorem aplikacji, program Flash Professional CS5 ustawia to pole i staje się ono nieedytowalne. W przeciwnym wypadku profil informacyjny dopuszcza wiele identyfikatorów aplikacji (co wyraża się użyciem symboli wieloznacznych). Należy podać identyfikator aplikacji zgodny z symbolem wieloznacznym identyfikatora aplikacji wg firmy Apple:
  - Jeśli identyfikator aplikacji wg firmy Apple to com.myDomain.*, identyfikator aplikacji w oknie dialogowym ustawień telefonu iPhone musi rozpoczynać się od ciągu znaków com.myDomain. (np. com.myDomain.myApp lub com.myDomain.app22).
  - Jeśli identyfikator aplikacji wg Apple to *, wówczas identyfikator aplikacji w oknie dialogowym ustawień telefonu iPhone może być dowolnym ciągiem poprawnych znaków.

  Identyfikator aplikacji wg Apple (lub wzór wieloznacznego identyfikatora aplikacji) powiązany z profilem informacyjnym można znaleźć w witrynie iPhone Dev Center (http://developer.apple.com/iphone). Przejdź do sekcji iPhone Developer Program Portal serwisu, a następnie do sekcji Provisioning.

Ważne: Należy zignorować znaki poprzedzające identyfikator aplikacji podawany przez Apple. Zgodnie z terminologią Apple ten ciąg znaków to ID wartości początkowej pakunku. Na przykład, jeśli zostanie podany identyfikator aplikacji 96LPVWEASL.com.example.bob.myApp, należy zignorować ciąg znaków 96LPVWEASL — zamiast tego jako identyfikatora aplikacji należy użyć wartości com.example.bob.myApp. Jeśli zostanie podany identyfikator aplikacji 5RM86Z4DJM.*, należy zignorować łańcuch 5RM86Z4DJM — jest to wieloznaczny identyfikator aplikacji.

- Typ instalacji na telefonie iPhone:
  - Szybkie publikowanie do testowania na urządzeniu — tę opcję należy wybrać, aby szybko skompilować wersję aplikacji do testowania na telefonie iPhone, którym dysponuje programista.
  - Szybkie publikowanie do debugowania na urządzeniu — tę opcję należy wybrać, aby szybko skompilować wersję aplikacji do debugowania na telefonie iPhone, którym dysponuje programista. Opcja ta umożliwia odebranie wyników metody `trace()` z aplikacji na telefon iPhone przez debugger Flash Professional CS5. (Więcej informacji zawiera sekcja „Debugowanie aplikacji na telefon iPhone” na stronie 26.)
  - Instalacja - Ad Hoc — tę opcję należy wybrać w celu utworzenia aplikacji na potrzeby instalacji ad hoc. Więcej informacji można znaleźć w witrynie Apple Dev Center firmy Apple.
  - Instalacja - Apple App Store — tę opcję należy wybrać w celu utworzenia ostatecznej wersji pliku IPA z myślą o udostępnieniu w serwisie Apple App Store.

Karta Ikony

Na karcie Ikony należy określić położenie obrazu ikony o wymiarach 29 x 29 pikseli, obrazu ikony o wymiarach 48 x 48 pikseli, obrazu ikony o wymiarach 57 x 57 pikseli, obrazu ikony o wymiarach 72 x 72 piksele i obrazu ikony o wymiarach 512 x 512 pikseli. Patrz „Ikona telefonu iPhone i ikony ekranu początkowego” na stronie 14.

Uwaga: *Opcje obrazów o wymiarach 48 x 48 pikseli i 72 x 72 piksele nie są uwzględnione w wersji programu Packager for iPhone Preview dołączonej do produktu Flash Professional CS5. Aby zaktualizować program i uzyskać dostęp do tych opcji, należy wybrać polecenie Pomoc > Aktualizacje w programie Flash Professional CS5.*

Ustawianie właściwości aplikacji na telefon iPhone w pliku deskryptora aplikacji

Plik deskryptora aplikacji to plik XML zawierający właściwości dla całej aplikacji, np. jej nazwę, wersję, prawa autorskie i inne ustawienia.

Program Flash Professional CS5 generuje plik deskryptora aplikacji w oparciu o ustawienia wprowadzone w oknie dialogowym ustawień telefonu iPhone. Plik deskryptora aplikacji można również poddać edycji w edytorze tekstu. Program Flash Professional nadaje nazwę plikowi deskryptora aplikacji, dodając „-app.xml” do nazwy projektu. Na przykład plik deskryptora aplikacji dla projektu HelloWorld otrzymuje nazwę HelloWorld-app.xml. Aby zdefiniować ustawienia nieobsługiwane za pośrednictwem okna dialogowego ustawień telefonu iPhone programu Flash Professional CS5, należy dokonać edycji pliku deskryptora aplikacji. Można, na przykład, zdefiniować element `InfoAdditions` w celu zdefiniowania ustawień info.Plist dla aplikacji.

Ważne: Nie edytuj pliku deskryptora aplikacji, gdy okno dialogowe programu Flash Professional CS5 jest otwarte. Zmiany w pliku deskryptora aplikacji należy zapisać przed otwarciem okna dialogowego ustawień iPhone.

Poniżej przedstawiono przykładowy plik deskryptora aplikacji:

```
<?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>
```

Oto szczegółowe informacje dotyczące ustawień w tym pliku deskryptora aplikacji:

- W elemencie `<application>` na potrzeby budowy aplikacji na telefon iPhone wymagana jest przestrzeń nazw AIR 2.0

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

- Element `<id>`:

  <id>com.example.as3.HelloWorld</id> Identyfikator aplikacji w sposób unikalny identyfikuje aplikację. Zalecaną formą jest rozdzielony kropkami ciąg znaków w stylu odwróconego adresu DNS, taki jak`"com.firma.nazwa_aplikacji"`. Kompilator korzysta z tej wartości jako identyfikatora pakunku na potrzeby aplikacji iPhone.

  Jeśli plik informacyjny jest powiązany z określonym identyfikatorem aplikacji, użyj tego identyfikatora aplikacji w tym elemencie. Zignoruj znaki przypisywane przez Apple na początku identyfikatora aplikacji Apple (znane jako identyfikator wartości początkowej pakunku). Na przykład, jeśli identyfikator aplikacji dla profilu informacyjnego to 96LPVWEASL.com.example.bob.myApp, użyj wartości com.example.bob.myApp jako identyfikatora aplikacji w pliku deskryptora aplikacji.

  Jeśli profil informacyjny dopuszcza wiele (wieloznacznych) identyfikatorów aplikacji, jego identyfikator aplikacji kończy się symbolem gwiazdki (np. 5RM86Z4DJM.*). Należy podać identyfikator aplikacji zgodny ze wzorem symbolu wieloznacznego identyfikatora aplikacji wg firmy Apple:

  - Jeśli identyfikator aplikacji wg firmy Apple to com.myDomain.*, identyfikator aplikacji w pliku deskryptora aplikacji musi rozpoczynać się od ciągu com.myDomain. Możliwe jest określenie identyfikatora aplikacji takiego jak com.myDomain.myApp lub com.myDomain.app22.

- Jeśli identyfikator aplikacji Apple to *, identyfikator aplikacji w pliku deskryptora aplikacji może być dowolnym ciągiem poprawnych znaków.

Identyfikator aplikacji wg Apple (lub wzór wieloznacznego identyfikatora aplikacji) powiązany z profilem informacyjnym można znaleźć w witrynie iPhone Dev Center (http://developer.apple.com/iphone). Przejdź do sekcji iPhone Developer Program Portal serwisu, a następnie do sekcji Provisioning.

Ważne: Należy zignorować znaki poprzedzające identyfikator aplikacji podawany przez Apple. Zgodnie z terminologią Apple ten ciąg znaków to ID wartości początkowej pakunku. Na przykład, jeśli zostanie podany identyfikator aplikacji 5RM86Z4DJM.*, należy zignorować łańcuch 5RM86Z4DJM — jest to wieloznaczny identyfikator aplikacji. Jeśli zostanie podany identyfikator aplikacji 96LPVWEASL.com.example.bob.myApp, należy zignorować ciąg znaków 96LPVWEASL — zamiast tego jako identyfikatora aplikacji należy użyć wartości com.example.bob.myApp.

- Element `<filename>`:

  <filename>HelloWorld</filename> Nazwa używana na potrzeby pliku instalatora iPhone. W nazwie pliku nie należy używać znaku plus (+).

- Element `<name>`:

  <name>Hello World</name> Nazwa aplikacji wyświetlana w aplikacjach iTunes oraz iPhone. W nazwie nie należy używać znaku plus (+).

- Element `<version>`:

  <version>1.0</version> Ułatwia użytkownikom określenie, którą wersję aplikacji instalują. Ta wersja jest używana jako obiekt CFBundleVersion aplikacji na telefon iPhone. Musi mieć ona format zbliżony do: nnnnn[.nn[.nn]] (gdzie n jest cyfrą z zakresu 0–9, a nawiasy oznaczają elementy opcjonalne, np. 1, 1.0 czy 1.0.1. Wersje telefonu iPhone muszą zawierać tylko cyfry i kropki dziesiętne. Wersje telefonu iPhone mogą zawierać maksymalnie dwie kropki dziesiętne.

- Element `<initialWindow>` zawiera następujące elementy podrzędne umożliwiające określenie właściwości początkowego wyglądu aplikacji:

  <content>HelloWorld.swf</content> Identyfikuje główny plik SWF do kompilacji w aplikację na telefon iPhone.

  <visible>true</visible> Jest to ustawienie wymagane.

  <fullScreen>true</fullScreen> Określa, że aplikacja używa pełnego ekranu telefonu iPhone.

  <aspectRatio>portrait</aspectRatio> Określa, że jako początkowe proporcje aplikacji wybrano tryb pionowy (nie zaś poziomy). Należy zwrócić uwagę na fakt, że plik Default.png używany do definiowania początkowego okna aplikacji powinien mieć 320 pikseli szerokości oraz 480 pikseli wysokości, niezależnie od tego ustawienia. (Patrz „Ikona telefonu iPhone i ikony ekranu początkowego” na stronie 14).

  <autoOrients>true</autoOrients> (opcja) Określa, czy orientacja bieżącej treści aplikacji jest ponownie automatycznie zmieniana wraz ze zmianą orientacji fizycznej urządzenia. Wartością domyślną jest `true`. Możliwe jest anulowanie automatycznej orientacji przez wywołanie metody `preventDefault()` zdarzenia `orientationChanging`. wywoływane przez obiekt Stage. Więcej informacji można znaleźć w sekcji Ustawianie i wykrywanie orientacji ekranu.

  W przypadku korzystania z funkcji automatycznej orientacji w celu uzyskania najlepszych wyników należy ustawić właściwość `align` stołu montażowego na następującą wartość:

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

<renderMode>gpu</renderMode> (opcja) Tryb renderowania używany przez aplikację. Dostępne są trzy możliwe ustawienia:

- `cpu` — aplikacja korzysta z procesora CPU do renderowania wszystkich obiektów aplikacji. Nie stosuje się akceleratora sprzętowego.
- `gpu` — aplikacja korzysta z procesora GPU telefonu iPhone do kompozycji bitmap.
- `auto` — ta funkcja nie została zaimplementowana.

Więcej informacji można znaleźć w sekcji „Przyspieszanie sprzętowe” na stronie 38.

- Element `<profiles>`:

  <profiles>mobileDevice</profiles> Ogranicza zakres kompilowanej aplikacji do profilu urządzenia mobilnego. Profil ten obecnie obsługuje jedynie aplikacje na telefon iPhone. Dostępne są trzy obsługiwane profile:

  - `desktop` — aplikacja AIR dla środowiska lokalnego na komputerze stacjonarnym.
  - `extendedDesktop` — aplikacja AIR dla środowiska lokalnego na komputerze stacjonarnym z obsługą interfejsu API procesów rodzimych.
  - `mobileDevice` — aplikacja AIR na urządzenie mobilne. Obecnie telefon iPhone jest jedynym obsługiwanym urządzeniem mobilnym.

  Ograniczenie zakresu aplikacji do określonego profilu zapobiega kompilacji do innych profili. W przypadku nieokreślenia profilu możliwe jest skompilowanie aplikacji dla dowolnego z tych profili. Możliwe jest określenie wielu profili w formie listy. Są one rozdzielone spacjami w elemencie `<profiles>`.

  Należy koniecznie uwzględnić `mobileDevice` jako obsługiwany profil (lub pozostawić element `<profiles>` pusty).

- Element `<icon>` zawiera następujące elementy podrzędne umożliwiające określenie ikon używanych w aplikacji:

  <image29x29>icons/icon29.png</image29x29> Jest to obraz używany w wynikach wyszukiwania Spotlight.

  <image48x48>icons/icon48.png</image48x48> Jest to obraz używany w wynikach wyszukiwania Spotlight na iPadzie.

  <image57x57>icons/icon57.png</image57x57> Jest to obraz używany na ekranie początkowym urządzeń iPhone oraz iPod Touch.

  <image72x72>icons/icon72.png</image72x72> Jest to obraz używany na ekranie początkowym iPada.

  <image512x512>icons/icon512.png</image512x512> Jest to obraz używany w aplikacji iTunes.

  Narzędzie Packager for iPhone korzysta z ikon 29, 57 oraz 512, do których istnieją odwołania w pliku deskryptora aplikacji. To narzędzie kopiuje je do plików o nazwach: Icon-Small.png, Icon.png i iTunesArtwork. Aby uniknąć sporządzania takiej kopii, można spakować te pliki bezpośrednio. W tym celu należy umieścić je w katalogu zawierającym plik deskryptora aplikacji i wymieniając ich prawidłowe nazwy i ścieżki.

  Obraz w rozmiarze 512 przeznaczony jest wyłącznie na potrzeby wewnętrzne związane z testowaniem. Przesyłając ostateczną wersję aplikacji do sklepu Apple App Store, obraz w rozmiarze 512 dostarcza się jako osobny plik. Nie jest on uwzględniany w pliku IPA. Tę opcję należy wybrać, aby móc upewnić się, że obraz w rozmiarze 512 wygląda poprawnie w programie iTunes, jeszcze przed jego przesłaniem.

- Element `<iPhone>` zawiera następujące elementy podrzędne umożliwiające określenie ustawień specyficznych dla telefonu iPhone:

  <InfoAdditions></InfoAdditions> zawiera elementy podrzędne określające pary klucz-wartość do użycia jako ustawienia Info.plist na potrzeby aplikacji:

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

W tym przykładzie wartości powodują ustawienie stylu paska stanu aplikacji oraz określają, że aplikacja nie wymaga stałego dostępu do sieci Wi-Fi.

Ustawienia InfoAdditions są umieszczone w znaczniku `CDATA`.

Aby zapewniona była obsługa iPada, należy uwzględnić klucze i wartości w tablicy `UIDeviceFamily`. `UIDeviceFamily` jest tablicą ciągów znaków. Każdy ciąg definiuje obsługiwane urządzenia. Ustawienie `<string>1</string>` oznacza obsługę urządzeń iPhone i iPod Touch. Ustawienie `<string>2</string>` oznacza obsługę iPada. Ustawienie `<string>3</string>` oznacza obsługę platformy tvOS. W wypadku określenia tylko jednego z tych ustawień, obsługiwana będzie wyłącznie wskazana w ten sposób rodzina urządzeń. Na przykład poniższe ustawienie oznacza, że obsługiwany będzie tylko iPad:

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

Następujące zestawy obsługują obie rodziny urządzeń (iPhone/iPod Touch oraz iPad):

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

Więcej informacji na temat innych ustawień Info.plist zawiera dokumentacja dla programistów Apple.

Kompilowanie pliku instalatora aplikacji na telefon iPhone

Program Packager for iPhone służy do kompilowania aplikacji w języku ActionScript 3.0 do pliku instalatora IPA.

Tworzenie pliku instalatora aplikacji na telefon iPhone przy użyciu programu Packager for iPhone dołączonego do programu Flash Professional CS5

Program Packager for iPhone jest uwzględniony w programie Flash Professional CS5:

1. Wybierz opcję Plik > Publikuj.
2. Sprawdź, czy w oknie dialogowym ustawień telefonu iPhone podano wartości wszystkich ustawień. Upewnij się, że na karcie Instalacja wybrano prawidłowe opcje. Zobacz „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5” na stronie 16.
3. Kliknij przycisk Publikuj.

Program Packager for iPhone generuje plik instalatora aplikacji iPhone (IPA). Kompilowanie pliku IPA może zająć kilka minut.

Program Packager for iPhone można także uruchomić z wiersza poleceń. Więcej informacji można znaleźć również w sekcji „Tworzenie pliku instalatora aplikacji na telefon iPhone z wiersza poleceń” na stronie 23.

Tworzenie pliku instalatora aplikacji na telefon iPhone z wiersza poleceń

Program Packager for iPhone można uruchomić z wiersza poleceń. Program Packager for iPhone przekształca kod bitowy pliku SWF oraz innych plików źródłowych w rodzimą aplikację na telefon iPhone.

1 Otwórz warstwę poleceń lub terminal i przejdź do folderu projektu aplikacji na telefon iPhone.

2 Następnie za pomocą narzędzia pfi utwórz plik IPA, korzystając z następującej składni polecenia:

```
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
```

Zmień odwołanie do aplikacji `pfi` w celu uwzględnienia pełnej ścieżki do aplikacji pfi. Aplikacja pfi jest instalowana w podkatalogu pfi/lib katalogu instalacyjnego programu Flash Professional CS5.

Należy wybrać opcję `-target` odpowiadającą typowi aplikacji na telefon iPhone, którą chce się utworzyć:

- `-target ipa-test` — tę opcję należy wybrać, aby szybko skompilować wersję aplikacji na potrzeby testowania na telefonie iPhone, którym dysponuje programista.
- `-target ipa-debug` — tę opcję należy wybrać, aby skompilować wersję debugowanej aplikacji na potrzeby testowania na telefonie iPhone, którym dysponuje programista. Ta opcja umożliwia odbieranie wyników instrukcji `trace()` z aplikacji na telefon iPhone w sesji debugowania.

  Możliwe jest podanie jednej z następujących opcji `-connect` (`OPCJE_POŁ CZENIA`) w celu określenia adresu IP komputera programistycznego, na którym działa debuger:

  - `-connect` — aplikacja będzie próbowała nawiązać połączenie z sesją debugowania na komputerze programistycznym, na którym została skompilowana.
  - `-connect ADRES_IP` — aplikacja będzie próbowała nawiązać połączenie z sesją debugowania na komputerze o określonym adresie IP. Na przykład:

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

  - `-connect NAZWA_HOSTA` — aplikacja będzie próbowała nawiązać połączenie z sesją debugowania na komputerze o określonej nazwie hosta. Na przykład:

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

  Uwaga: *Opcja `-connect` nie jest obsługiwana przez wersję wstępną (Preview) programu Packager for iPhone dołączoną do produktu Flash Professional CS5. Aby zaktualizować program Packager for iPhone, należy wybrać polecenie Pomoc > Aktualizacje w programie Flash Professional CS5.*

  Opcja `-connect` nie jest wymagana. Jeśli nie zostanie określona, uzyskana aplikacja przeznaczona do debugowania nie będzie próbowała łączyć się z debugerem na hoście.

  Jeśli próba połączenia z sesją debugowania nie powiedzie się, aplikacja wyświetli okno dialogowe z prośbą o wpisanie adresu IP hosta z debugerem. Próba połączenia nie powiedzie się, jeśli urządzenie nie jest podłączone do sieci Wi-Fi. Problem z połączeniem może również wystąpić, jeśli urządzenie jest podłączone, ale nie znajduje się za zaporą sieciową hosta z debugerem.

  Więcej informacji można znaleźć w sekcji „Debugowanie aplikacji na telefon iPhone” na stronie 26.

  Można również skorzystać z opcji `-renderingdiagnostics` w celu włączenia funkcji diagnostyki renderowania przy użyciu procesora GPU. Więcej informacji zawiera sekcja „Diagnostyka renderowania przy użyciu procesora GPU” w rozdziale „Debugowanie aplikacji na telefon iPhone” na stronie 26.

- `-target ipa-ad-hoc` — tę opcję należy wybrać w celu utworzenia aplikacji na potrzeby instalacji ad hoc. Więcej informacji można znaleźć w witrynie Apple Dev Center firmy Apple.
- `-target ipa-app-store` — tę opcję należy wybrać w celu utworzenia ostatecznej wersji pliku IPA z myślą o udostępnieniu w serwisie Apple App Store.

  Zastąp element *`PROFILE_PATH`* ścieżką do pliku profilu informacyjnego dla tworzonej aplikacji. Więcej informacji na temat profili informacyjnych można znaleźć w sekcji „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.

  Zastąp element *`SIGNING_OPTIONS`* odwołaniem do certyfikatu i hasła programisty aplikacji na telefon iPhone. Należy użyć poniższej składni:

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

  Zastąp element *P12_FILE_PATH* ścieżką do pliku certyfikatu P12. Zastąp element *PASSWORD* hasłem dla certyfikatu. (Patrz przykład poniżej.) Więcej informacji na temat pliku certyfikatu P12 zawiera sekcja „Konwersja certyfikatu programisty do pliku P12” na stronie 7.

  Zastąp element *`APP_DESCRIPTOR`* odwołaniem do pliku deskryptora aplikacji.

  Zastąp element *`SOURCE_FILES`* odwołaniem do głównego pliku SWF projektu wraz z ewentualnymi innymi zasobami, które mają zostać uwzględnione. Koniecznie uwzględnij ścieżki do wszystkich plików ikon zdefiniowanych w oknie dialogowym ustawień aplikacji w programie Flash CS5 lub też w niestandardowym pliku deskryptora aplikacji. Następnie dodaj plik kompozycji ekranu początkowego, Default.png.

Należy rozważyć następujący przykład:

```
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"
```

Powoduje on kompilację pliku HelloWorld.ipa za pomocą następujących elementów:

- specyficznego certyfikatu PKCS#12, z użyciem hasła certyfikatu dfb7VKL19
- pliku deskryptora aplikacji HelloWorld-app.xml
- źródłowego pliku HelloWorld.swf
- specyficznego pliku Default.png oraz plików ikon

Aplikacja pfi kompiluje aplikację — w oparciu o plik deskryptora aplikacji, plik SWF oraz inne zasoby — do pliku IPA.

W systemie Mac OS możliwe jest użycie certyfikatu zapisanego w pęku kluczy przez dodanie następujących opcji do polecenia pfi:

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

Zastąp element *ALIAS_NAME* aliasem certyfikatu, którego chcesz użyć. Wskazując certyfikat zapisany w pęku kluczy Mac, nie trzeba już wskazywać lokalizacji pliku certyfikatu — wystarczy wskazać alias.

Instalowanie aplikacji na telefon iPhone

Aby zainstalować opracowywaną aplikację na telefonie iPhone, dodaj profil informacyjny do telefonu iPhone, a następnie zainstaluj aplikację na telefonie iPhone.

Następnie dodaj profil informacyjny do telefonu iPhone

Aby dodać profil informacyjny do telefonu iPhone:

1 W programie iTunes wybierz kolejno opcje Plik > Dodaj do biblioteki. Następnie wybierz plik profilu informacyjnego (dla którego jako rozszerzenie nazwy pliku wybrano „mobileprovision").

   Upewnij się, że posiadany telefon iPhone jest dodawany do profilu informacyjnego. Możliwe jest zarządzanie profilami informacyjnymi w witrynie iPhone Dev Center firmy Apple (http://developer.apple.com/iphone/). Przejdź do sekcji iPhone Developer Program Portal witryny. Kliknij łącze Urządzenia, aby zarządzać listą urządzeń, na których może być instalowana opracowywana aplikacja. Kliknij łącze Provisioning, aby zarządzać profilami informacyjnymi.

2 Podłącz swój telefon iPhone do komputera i przeprowadź synchronizację.

Więcej informacji na temat uzyskiwania profilu informacyjnego można znaleźć w sekcji „Uzyskiwanie plików dla twórców aplikacji od firmy Apple" na stronie 5.

Instalowanie aplikacji

Opracowywaną aplikację instaluje się w ten sam sposób, co każdy inny plik IPA:

1 Jeśli zainstalowano już wcześniej wersję aplikacji, usuń aplikację z urządzenia oraz z listy aplikacji w iTunes.

2 Dodaj aplikację do iTunes w dowolny z poniższych sposobów:

   - W menu Plik (w iTunes) wybierz polecenie Dodaj do biblioteki. Następnie wybierz plik IPA i kliknij przycisk Otwórz.
   - Kliknij dwukrotnie plik IPA.
   - Przeciągnij plik IPA do biblioteki iTunes.

3 Podłącz swój telefon iPhone do portu USB komputera.

4 W iTunes sprawdź kartę Aplikacja dla urządzenia i upewnij się, że aplikacja została wybrana z listy aplikacji do zainstalowania.

5 Przeprowadź synchronizację telefonu iPhone.

Rozwiązywanie problemów z instalowaniem aplikacji

Jeśli w programie iTunes podczas próby zainstalowania aplikacji zostanie wyświetlony błąd, należy sprawdzić następujące elementy:

- Upewnij się, że ID urządzenia został dodany do profilu informacyjnego.
- Aby upewnić się, że profil informacyjny został zainstalowany, można spróbować przeciągnąć go do iTunes lub użyć polecenia Plik > Dodaj do biblioteki.

Sprawdź również czy identyfikator aplikacji danej aplikacji odpowiada identyfikatorowi aplikacji wg Apple.

- Jeśli identyfikator aplikacji wg firmy Apple to com.myDomain.*, identyfikator aplikacji w pliku deskryptora aplikacji lub identyfikator aplikacji w oknie dialogowym ustawień telefonu iPhone musi rozpoczynać się od ciągu znaków com.myDomain (np. com.myDomain.anythinghere).
- Jeśli identyfikator aplikacji wg Apple to com.myDomain.myApp, identyfikator aplikacji w pliku deskryptora aplikacji lub w interfejsie użytkownika programu Flash Profession CS5 musi brzmieć com.myDomain.myApp.
- Jeśli identyfikator aplikacji wg Apple to *, identyfikator aplikacji w pliku deskryptora aplikacji lub w interfejsie użytkownika programu Flash Professional CS5 może mieć dowolne brzmienie.

Identyfikator aplikacji ustawia się w oknie dialogowym ustawień programu iPhone w programie Flash Professional CS5 lub w pliku deskryptora aplikacji.

Debugowanie aplikacji na telefon iPhone

Aplikacje można debugować na komputerze programisty, uruchamiając aplikację w programie ADL. Możliwe jest również debugowanie aplikacji na telefonie iPhone.

Niektóre elementy funkcjonalności środowiska AIR nieobsługiwane w telefonie iPhone są mimo to dostępne podczas testowania aplikacji za pomocą narzędzia ADL (na komputerze programisty). Należy pamiętać o tych różnicach podczas testowania treści na komputerze stacjonarnym. Więcej informacji można znaleźć w sekcji „Elementy interfejsu API ActionScript 3.0 nieobsługiwane na urządzeniach mobilnych” na stronie 30

Debugowanie aplikacji na komputerze programisty

W celu debugowania aplikacji na komputerze programisty za pomocą programu Flash Professional CS5:

❖ Wybierz kolejno opcje Debuguj > Debuguj film > W programie AIR Debug Launcher (zdalnie).

Możesz również debugować aplikacje, wywołując program ADL z wiersza poleceń: Oto jego składnia:

```
adl -profile mobileDevice appDescriptorFile
```

Zastąp element *`appDescriptorFile`* ścieżką do pliku deskryptora aplikacji.

Należy koniecznie uwzględnić opcję `-profile mobileDevice`.

Debugowanie aplikacji w telefonie iPhone

Aby debugować aplikację na telefonie iPhone:

1. Kompilowanie aplikacji z obsługą debugowania:
   - W programie Flash Professional CS5 należy przeprowadzić kompilację z użyciem ustawienia szybkiego publikowania do debugowania na urządzeniu. (Zobacz „Tworzenie pliku instalatora aplikacji na telefon iPhone przy użyciu programu Packager for iPhone dołączonego do programu Flash Professional CS5” na stronie 22).
   - Korzystając z narzędzia PFI wywoływanego z wiersza polecenia, należy skompilować aplikację z opcją `target ipa-debug`. (Więcej informacji można znaleźć również w sekcji „Tworzenie pliku instalatora aplikacji na telefon iPhone z wiersza poleceń” na stronie 23.)
2. Zainstaluj aplikację na telefonie iPhone.
3. W telefonie iPhone włącz sieć Wi-Fi i podłącz się do tej samej sieci, do której podłączony jest komputer służący do programowania.
4. Uruchom sesję debugowania na komputerze używanym do tworzenia aplikacji. W programie Flash Professional CS5 wybierz kolejno opcje Debuguj > Rozpocznij zdalną sesję debugowania > ActionScript 3.0.
5. Uruchom aplikację na telefonie iPhone.

   Zostanie wyświetlony monit wersji debugowanej aplikacji o podanie adresu IP komputera służącego do programowania. Wprowadź adres IP, a następnie stuknij przycisk OK. Aby odczytać adres IP komputera służącego do programowania:
   - W systemie Mac OS, w menu Apple, wybierz Preferencje systemowe. W oknie preferencji systemu kliknij ikonę Sieć. W oknie preferencji sieci zostanie wyświetlony adres IP.
   - W systemie Windows uruchom sesję wiersza poleceń, a następnie uruchom polecenie `ipconfig`.

W sesji debugowania będą wyświetlane wszelkie wyniki działania metody `trace()` z aplikacji.

W przypadku debugowania aplikacji zainstalowanej na telefonie iPhone program Flash Professional CS5 obsługuje wszystkie funkcje debugowania, w tym kontrolę punktów przerwań, przechodzenie kodu krok po kroku oraz monitorowanie zmiennych.

Diagnostyka renderowania przy użyciu procesora GPU

Funkcja diagnostyki renderowania przy użyciu procesora GPU umożliwia sprawdzenie, w jaki sposób aplikacja korzysta z akceleracji sprzętowej (dotyczy to aplikacji korzystających z trybu renderowania przy użyciu procesora GPU). Aby skorzystać z tej funkcji, należy skompilować aplikację za pomocą narzędzia PFI wywoływanego z wiersza poleceń, podając opcje `-renderingdiagnostics`:

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

Flaga `-renderingdiagnostics` musi następować bezpośrednio po fladze `-package`.

Funkcja diagnostyki renderowania przy użyciu procesora GPU wyświetla kolorowe prostokąty w tle wszystkich obiektów wyświetlanych:

- Niebieskie — obiekt wyświetlany nie jest bitmapą ani nie jest buforowany jako bitmapa i jest renderowany.

  Jeśli niebieski prostokąt wyświetlany jest stale w tle obiektu wyświetlanego, który nie ulega zmianom, przyczyną może być przecinanie się tego obiektu z obiektami ruchomymi. Obiekt może być na przykład tłem ruchomych obiektów wyświetlanych. Należy rozważyć buforowanie takiego obiektu jako bitmapy.

  Jeśli niebieski prostokąt wyświetlany jest w tle obiektu, który powinien być buforowany, to być może do renderowania obiektu stosowany jest efekt nieobsługiwany przez procesor GPU. Do efektów tych należą niektóre tryby mieszania, przekształcenia kolorów, właściwość `scrollRect` i maski.

  Kolor niebieski oznacza również przekroczenie limitu pojemności pamięci przez obiekty przesłane do procesora GPU.

  Dla każdego niebieskiego prostokąta aplikacja rejestruje komunikat. Komunikaty te są generowane razem z innymi komunikatami funkcji `trace()` i komunikatami debugowania.

- Zielone — obiekt wyświetlany jest bitmapą lub jest buforowany jako bitmapa i po raz pierwszy jest przekazywany do procesora GPU.

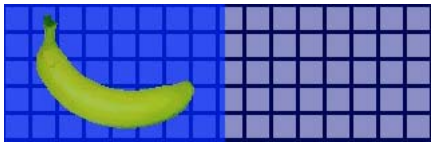
  Jeśli zielony prostokąt wyświetlany jest stale w tle obiektu wyświetlanego, wówczas kod w aplikacji ponownie tworzy obiekt wyświetlany. Taka sytuacja może wystąpić na przykład w momencie, gdy nastąpi powrót na osi czasu do klatki, która tworzy obiekt wyświetlany. Należy rozważyć modyfikację treści, aby zapobiec ponownemu tworzeniu identycznych obiektów.

- Czerwone — obiekt wyświetlany jest bitmapą lub jest buforowany jako bitmapa i jest ponownie przekazywany do procesora GPU.

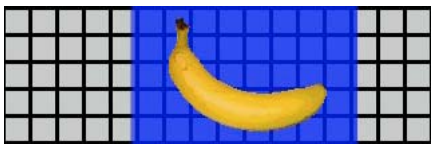
  Czerwony prostokąt jest wyświetlany przy każdej zmianie obiektu wyświetlanego, która wymaga od aplikacji ponownego renderowania reprezentacji bitmapowej. Jeśli na przykład obiekt dwuwymiarowy nie ma ustawionej właściwości `cacheAsBitmapMatrix`, zostanie ponownie wyrenderowany w momencie skalowania lub obrotu. Ponowne renderowanie występuje także w przypadku przesunięcia lub zmiany podrzędnych obiektów wyświetlanych.

Kolorowy prostokąt zniknie po czterech cyklach przerysowania ekranu, jeśli ustąpi przyczyna wyświetlenia prostokąta. Jeśli jednak na ekranie nie wystąpią żadne zmiany, kolorowe prostokąty diagnostyczne nie ulegną zmianie.

Weźmy na przykład pod uwagę obiekt wyświetlany (banan) w postaci bitmapy wyświetlanej na wektorowym tle, które nie jest buforowane jako bitmapa. Przy pierwszym renderowaniu banan uzyska kolor zielony. Przy pierwszym renderowaniu tło uzyska kolor niebieski:

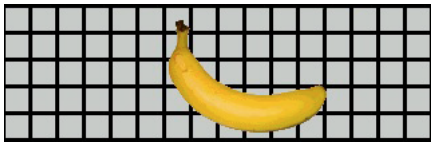


W trakcie przesuwania banana procesor główny musi ponownie renderować tło, co powoduje, że nad tłem zostaną wyświetlone niebieskie cienie:



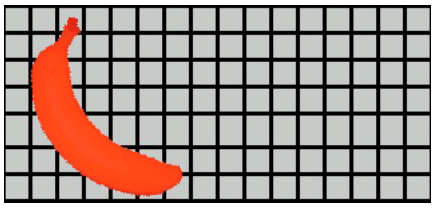
Niebieskie cienie nad tłem odzwierciedlają regiony, które należy przekazać do procesora GPU.

Jeśli jednak tło jest buforowane jako bitmapa i banan zostanie przesunięty, wówczas funkcja diagnostyki renderowania nie wyświetli kolorowych odcieni:



Funkcja diagnostyki nie wyświetli kolorowych odcieni, ponieważ procesor GPU zachowa bitmapę tła. Procesor GPU może złożyć banan z tłem bez użycia procesora głównego.

Załóżmy, że banan jest dwuwymiarowym obiektem wyświetlanym, który nie ma ustawionej właściwości `cacheAsBitmapMatrix`. Każdy obrót (lub skalowanie) obiektu wyświetlanego spowoduje wyświetlenie przez funkcję diagnostyki renderowania czerwonego prostokąta. Takie wskazanie sygnalizuje konieczność przekazania przez aplikację nowej wersji obiektu wyświetlanego do procesora GPU:



Przesyłanie aplikacji na telefon iPhone do sklepu App Store

Aby przesłać aplikację do sklepu App Store:

1. Uzyskaj certyfikat dystrybucyjny i profil informacyjny z witryny iPhone Dev Center (http://developer.apple.com/iphone/).

   Certyfikat dystrybucyjny ma nazwę „iPhone Developer: XXX.cer”, gdzie XXX to nazwa użytkownika.

   Więcej informacji można znaleźć w sekcji „Uzyskiwanie plików dla twórców aplikacji od firmy Apple” na stronie 5.

2. Przekształć certyfikat dystrybucyjny na plik certyfikatu P12.

   Więcej informacji zawiera sekcja „Konwersja certyfikatu programisty do pliku P12” na stronie 7.

3. Skompiluj aplikację, korzystając z pliku P12 i profilu informacyjnego.

Użyj pliku P12 utworzonego w oparciu o certyfikat dystrybucyjny. Użyj identyfikatora aplikacji dołączonego do profilu informacyjnego.

Więcej informacji można znaleźć w sekcji „Kompilowanie pliku instalatora aplikacji na telefon iPhone” na stronie 22.

4 Prześlij aplikację do witryny iPhone Dev Center (http://developer.apple.com/iphone/).

Ważne: *Firma Apple wymaga używania programu Apple Application Loader do przesyłania programów do sklepu App Store. Firma Apple publikuje program Application Loader wyłącznie dla systemu Mac OS X. Opracowując aplikację AIR na telefon iPhone na komputerze z systemem Windows, w celu przesłania aplikacji do sklepu App Store konieczny jest dostęp do komputera z systemem OS X (w wersji 10.5.3 lub nowszej). Program Application Loader można uzyskać w serwisie iOS Developer Center firmy Apple.*

Rozdział 3: Obsługa interfejsu API ActionScript 3.0 na urządzeniach mobilnych

Tworząc aplikacje AIR dla urządzeń mobilnych, można korzystać z elementów interfejsu API języka ActionScript 3.0 dostępnego na potrzeby aplikacji dla środowisk stacjonarnych Adobe Flash Player 10.1 i AIR 2. Istnieje jednak kilka wyjątków.

Elementy interfejsu API ActionScript 3.0 nieobsługiwane na urządzeniach mobilnych

Niektóre elementy interfejsu API nie są obsługiwane w aplikacjach działających w profilu urządzenia mobilnego (takich jak aplikacje działające na telefonie iPhone).

W przypadku używania tego samego kodu ActionScript do pisania programu z myślą o wielu profilach (np. komputerze stacjonarnym oraz urządzeniu mobilnym) należy użyć kodu w celu sprawdzenia, czy interfejs API jest obsługiwany. Na przykład klasa NativeWindow nie jest obsługiwana w aplikacjach na telefon iPhone. (W aplikacjach iPhone nie można używać ani tworzyć okien rodzimych.) W celu sprawdzenia, czy aplikacja działa w profilu obsługującym okna rodzime (takim jak profil aplikacji lokalnej na komputerze stacjonarnym), należy sprawdzić właściwość `NativeWindow.isSupported`.

W poniższej tabeli wymieniono interfejsy API, które nie są obsługiwane w profilu urządzenia mobilnego. Ponadto wymieniono tu również właściwości, które można sprawdzić w celu stwierdzenia, czy aplikacja działa na platformie oferującej obsługę interfejsu API.

Interfejs API	**Test obsługi**
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

Interfejs API	Test obsługi
NativeApplication.isSetAsDefaultApplication()	NativeApplication.supportsDefaultApplication
NativeApplication.startAtLogin	NativeApplication.supportsStartAtLogin
NativeMenu	NativeMenu.isSupported
NativeProcess	NativeProcess.isSupported
NativeWindow	NativeWindow.isSupported
NativeWindow.notifyUser()	NativeWindow.supportsNotification
NetworkInfo	NetworkInfo.isSupported
Obsługa formatu PDF	HTMLLoader.pdfCapability
PrintJob	PrintJob.isSupported
SecureSocket	SecureSocket.isSupported
ServerSocket	ServerSocket.isSupported
Shader	—
ShaderFilter	—
StorageVolumeInfo	StorageVolumeInfo.isSupported
XMLSignatureValidator	XMLSignatureValidator.isSupported

Nie jest możliwe tworzenie aplikacji AIR opartych na językach HTML i JavaScript w profilu urządzenia mobilnego.

Niektóre klasy ActionScript 3.0 są obsługiwane tylko częściowo:

File
Aplikacje telefonu iPhone mają dostęp wyłącznie do katalogu aplikacji oraz katalogu magazynu aplikacji. Ponadto możliwe jest wywołanie metod `File.createTempFile()` oraz `File.createTempDirectory()`. Wywołanie operacji w celu uzyskania dostępu do innego katalogu (np. wywołanie metody odczytu lub zapisu FileStream) skutkuje wyjątkiem IOError.

Aplikacje iPhone nie obsługują okien dialogowych przeglądarki plików rodzimych, takich jak ten udostępniany przez metodę `File.browseForOpen()`.

Loader
W aplikacji na telefon iPhone nie można używać metody `Loader.load()`. Nie można jednak uruchomić kodu ActionScript w treści SWF załadowanej metodą `Loader.load()`. Można jednak użyć zasobów z pliku SWF (takich jak klipy wideo, czcionki oraz dźwięki z biblioteki). Możliwe jest również użycie metody `Loader.load()` do załadowania plików obrazów.

Wideo
W aplikacjach AIR na telefon iPhone obsługiwane jest wyłącznie wideo w formacie Sorensen oraz ON2 VP6.

Można użyć metody `navigateToURL()` do otwarcia wideo w formacie H.264 poza aplikacją. Jako parametr `request` należy przekazać obiekt URLRequest wraz z adresem URL wskazującym na materiał wideo. Wideo uruchamia się w odtwarzaczu wideo telefonu iPhone.

Pola tekstowe

Istnieją ograniczenia dotyczące czcionek i innych ustawień pól tekstowych na telefonie iPhone. Patrz „Czcionki i wprowadzanie tekstu” na stronie 42.

Nieobsługiwane interfejsy API oraz debugowanie za pomocą narzędzia ADL

Niektóre elementy funkcjonalności środowiska AIR nieobsługiwane w telefonie iPhone są mimo to dostępne podczas testowania aplikacji za pomocą narzędzia ADL (na komputerze programisty). Należy pamiętać o tych różnicach podczas testowania treści za pomocą narzędzia ADL.

Funkcjonalność ta obejmuje następujące kodeki audio i wideo: Speex (audio), H.264/AVC (wideo) oraz AAC (audio). Kodeki te są niedostępne dla aplikacji AIR działających w telefonie iPhone. Działają one jednak w sposób normalny na komputerach stacjonarnych.

Ułatwienia dostępu i obsługa czytnika ekranu działają w programie ADL w systemie Windows. Te interfejsy API nie są obsługiwane w telefonie iPhone.

Protokół RTMPE działa normalnie w przypadku używania za pomocą narzędzia ADL na komputerze stacjonarnym. Połączenie NetConnection podejmujące próbę nawiązania połączenia za pomocą protokołu RTMPE nie działa jednak w telefonie iPhone.

Klasa Loader działa bez dodatkowych ograniczeń w przypadku uruchomienia treści za pomocą narzędzia ADL. Jednak podczas wykonywania aplikacji na telefonie iPhone próby załadowania treści SWF zawierającej kod bajtowy ActionScript skutkują komunikatami o błędzie.

Instancje Shader są wykonywane w programie ADL. Jednak w telefonie iPhone kod bajtowy modułu Pixel Bender nie jest interpretowany, zaś moduły cieniujące nie wpływają na wyświetlaną grafikę.

Więcej informacji można znaleźć w sekcji „Debugowanie aplikacji na telefon iPhone” na stronie 26.

Elementy interfejsu API ActionScript przeznaczone do tworzenia aplikacji AIR na urządzenia mobilne

Następujące elementy interfejsu API są dostępne tylko w aplikacjach AIR na urządzeniach mobilnych. Obecnie nie działają one w odtwarzaczu Flash Player ani w wersjach środowiska AIR dla komputerów stacjonarnych.

Interfejs API orientacji ekranu

Interfejs API orientacji ekranu umożliwia pracę orientacją stołu montażowego oraz telefonu iPhone:

- `Stage.autoOrients` — Określa, czy aplikacja automatycznie zmienia orientację stołu montażowego po obróceniu urządzenia. Właściwość ta przyjmuje wartość `true` w przypadku zaznaczenia opcji Automatyczna orientacja w oknie dialogowym ustawień dla telefonu iPhone w programie Flash Professional CS5. (Można również ustawić element `autoOrients` na wartość `true` w pliku deskryptora aplikacji.) Więcej informacji zawiera sekcja „Ustawienia aplikacji na telefon iPhone” na stronie 16. Automatyczną zmianę orientacji można anulować, dodając detektor zdarzenia `orientationChanging` dla obiektu Stage. Wywołanie metody `preventDefault()` tego obiektu zdarzenia powoduje anulowanie automatycznej zmiany orientacji.

  W przypadku korzystania z funkcji automatycznej orientacji w celu uzyskania najlepszych wyników należy ustawić właściwość `align` stołu montażowego na następującą wartość:

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

- `Stage.deviceOrientation` — Fizyczna orientacja urządzenia. Wartości tej właściwości są zdefiniowane w klasie StageOrientation.
- `Stage.orientation` — Bieżąca orientacja stołu montażowego. Wartości tej właściwości są zdefiniowane w klasie StageOrientation.
- `Stage.supportsOrientationChange` — Wartość `true` w telefonie iPhone, oraz `false` w aplikacji AIR.
- `Stage.setOrientation()` — Ustawia orientację stołu montażowego. Metoda ta ma jeden parametr, który jest ciągiem znaków definiującym nową orientację stołu montażowego. Stałe w klasie StageOrientation definiują możliwe wartości parametru.
- StageOrientation — Definiuje wartości orientacji stołu montażowego. Na przykład `StageOrientation.ROTATED_RIGHT` określa stół montażowy obracany w prawo względem domyślnej orientacji urządzenia.
- StageOrientationEvent — Definiuje zdarzenia wywoływane przez stół montażowy wraz ze zmianą orientacji ekranu. Zdarzenie to występuje, gdy użytkownik obróci telefon iPhone. Istnieją dwa typy zdarzeń. Stół montażowy wywołuje zdarzenie `orientationChanging` z chwilą obrócenia urządzenia. Aby uniemożliwić zmianę orientacji stołu montażowego, należy wywołać metodę `preventDefault()` obiektu zdarzenia `orientationChanging`. Stół montażowy wywołuje zdarzenie `orientationChange` tuż po zmianie orientacji stołu montażowego.

Elementy interfejsu API związane z orientacją ekranu są obecnie użyteczne wyłącznie w aplikacjach AIR na urządzeniach mobilnych. W wypadku gdy ten sam kod źródłowy jest używany w aplikacji AIR dla urządzenia mobilnego i aplikacji AIR dla środowiska stacjonarnego, należy odczytać właściwość `Stage.supportsOrientationChange`, aby sprawdzić, czy te elementy interfejsu API są obsługiwane.

W poniższym przykładzie zaprezentowano sposób reagowania na obracanie urządzeniem przez użytkownika:

```
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;
}
```

W tym przykładzie, w przypadku różnych orientacji stołu montażowego zamiast funkcjonalnego kodu wstawiono komentarze.

Możliwa jest zmiana orientacji stołu montażowego przez wywołanie metody `setOrientation()` obiektu Stage. Ustawienie orientacji jest operacją asynchroniczną. Możliwe jest sprawdzenie, czy orientacja została przestawiona, przez wykrycie zdarzenia `orientationChange`. Poniższy kod ilustruje sposób ustawienia stołu montażowego w orientacji dla użytkowników praworęcznych.

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

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

Obrót stołu montażowego powoduje zmianę jego wymiarów, czemu towarzyszy wywołanie zdarzenia `resize` przez obiekt Stage. W odpowiedzi na zdarzenie `resize` można zmienić wielkość i położenie obiektów wyświetlanych.

NativeApplication.systemIdleMode i SystemIdleMode

Właściwość `NativeApplication.systemIdleMode` pozwala zapobiegać przechodzeniu telefonu iPhone do trybu bezczynności. Domyślnie telefon iPhone przechodzi do trybu bezczynności w przypadku braku przez pewien czas interakcji z ekranem dotykowym. Tryb bezczynności może spowodować przyciemnienie ekranu. Ponadto może również spowodować przejście telefonu iPhone do trybu blokady. Właściwość tę można ustawić na jedną z dwu wartości:

- `SystemIdleMode.NORMAL` — Telefon iPhone działa zgodnie z normalnym stanem trybu bezczynności.
- `SystemIdleMode.KEEP_AWAKE` — Aplikacja podejmuje próbę zapobieżenia przechodzeniu telefonu iPhone do trybu bezczynności.

Ta funkcjonalność jest obsługiwana tylko na urządzeniach mobilnych. Nie jest obsługiwana w aplikacjach AIR działających w stacjonarnych systemach operacyjnych. W aplikacji działającej w środowisku stacjonarnym ustawianie właściwości `NativeApplication.systemIdleMode` nie odnosi skutku.

Poniższy kod ilustruje, w jaki sposób można wyłączyć tryb bezczynności telefonu iPhone:

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

CameraRoll

Klasa CameraRoll umożliwia dodawanie obrazu do rolki z aparatu w telefonie iPhone. Metoda `addBitmapData()` dodaje obraz do rolki z aparatu w telefonie iPhone. Metoda ta ma jeden parametr, `bitmapData`. Parametr ten to obiekt BitmapData zawierający obraz, który jest dodawany do rolki z aparatu.

Funkcjonalność obiektu CameraRoll jest obsługiwana tylko na urządzeniach mobilnych. Nie jest obsługiwana w aplikacjach AIR działających w stacjonarnych systemach operacyjnych. W celu sprawdzenia w środowisku wykonawczym, czy aplikacja obsługuje funkcjonalność CamerRoll, należy sprawdzić statyczną właściwość `CameraRoll.supportsAddBitmapData`.

Po wywołaniu metody `addBitmapData()` obiekt CameraRoll wywołuje jedno z dwu zdarzeń:

- `complete` — Operacja została zakończona pomyślnie.
- `error` — Wystąpił błąd. Możliwe na przykład, że na telefonie iPhone brak wystarczającej ilości miejsca na przechowanie obrazu.

Poniższy kod dodaje obraz stołu montażowego (przechwycony obraz ekranu) do rolki z aparatu:

```
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

Właściwość `cacheAsBitmapMatrix` to obiekt Matrix definiujący sposób renderowania obiektu wyświetlanego, gdy właściwość `cacheAsBitmap` została ustawiona na wartość `true`. Aplikacja korzysta z tej matrycy jako matrycy transformacji podczas renderowania wersji bitmapy obiektu wyświetlanego.

W przypadku ustawienia właściwości `cacheAsBitmapMatrix` aplikacja zachowuje przechwycony obraz bitmapy zrenderowany przy użyciu tej matrycy zamiast matrycy wyświetlania. (Matryca wyświetlania to wartość `transform.concatenatedMatrix` obiektu wyświetlanego.) Jeśli ta matryca nie jest zgodna z matrycą wyświetlania, bitmapa jest skalowana i obracana odpowiednio do potrzeb.

Obiekt wyświetlany o ustawionej właściwości `cacheAsBitmapMatrix` jest renderowany wyłącznie, gdy wartość `cacheAsBitmapMatrix` ulegnie zmianie. Bitmapa jest skalowana lub obracana odpowiednio do potrzeb, tak aby była zgodna z matrycą wyświetlania.

Zarówno w przypadku renderowania bazującego na CPU, jak i bazującego na GPU właściwość `cacheAsBitmapMatrix` okazuje się przydatna, choć zazwyczaj więcej korzyści zauważa się w przypadku renderowania GPU.

Uwaga: *Aby użyć akceleracji sprzętowej, ustaw opcję Rendering dla GPU na karcie Ogólne w oknie dialogowym Ustawienia telefonu iPhone w programie Flash Professional CS5. (Można również ustawić właściwość* `renderMode` *na wartość* `gpu` *w pliku deskryptora aplikacji.)*

Na przykład poniższy kod korzysta z niepoddanej transformacji reprezentacji bitmapy obiektu wyświetlanego:

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

Poniższy kod wykorzystuje reprezentację bitmapy odpowiadająca bieżącemu renderingowi:

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

Zazwyczaj wystarcza identyczna matryca (`new Matrix()`) lub `transform.concatenatedMatrix`. Można jednak użyć innej matrycy, takiej jak matryca przeskalowana w dół, w celu załadowania innej bitmapy do GPU. Na przykład w poniższym przykładzie zastosowano matrycę `cacheAsBitmapMatrix` przeskalowaną z użyciem współczynnika 0,5 względem osi x i y. Obiekt bitmapy używany przez GPU jest mniejszy, jednak GPU dostosowuje jego wielkość tak, by odpowiadała właściwości `transform.matrix` obiektu wyświetlanego:

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

W ogólnym przypadku należy wybrać matrycę przekształcającą obiekt wyświetlana na rozmiar, w którym będzie się ona pojawiać w aplikacji. Na przykład, jeśli w aplikacji wyświetlana jest wersja bitmapy ikonki przeskalowanej w dół o połowę, należy użyć matrycy skalującej w dół o połowę. Jeśli w aplikacji ikonka ma być wyświetlana jako większa niż jej obecny rozmiar, należy użyć matrycy skalującej w górę o zadany współczynnik.

Istnieje jednak praktyczny limit wielkości obiektów wyświetlanych, dla których ustawiono właściwość `cacheAsBitmapMatrix`. Limit ten to 1020 x 1020 pikseli. Istnieje też praktyczny limit łącznej liczby pikseli dla wszystkich obiektów wyświetlanych, dla których ustawiono właściwość `cacheAsBitmapMatrix`. Limit ten wynosi około czterech milionów pikseli.

Istnieje wiele kwestii, które należy wziąć pod uwagę, korzystając z właściwości `cacheAsBitmapMatrix` oraz akceleracji sprzętowej. Szczególnie ważne jest stwierdzenie, dla których obiektów wyświetlanych właściwość ta powinna być ustawiona, zaś dla których nie. Ważne informacje na temat używania tej właściwości zawiera sekcja „Przyspieszanie sprzętowe” na stronie 38.

Istnieje możliwość skorzystania z funkcji diagnostyki renderowania przy użyciu procesora GPU do diagnozowania wykorzystania procesora GPU w aplikacjach skompilowanych na potrzeby debugowania. Więcej informacji można znaleźć w sekcji „Debugowanie aplikacji na telefon iPhone” na stronie 26.

Uwagi dotyczące pracy w sieci

Użycie poniższych schematów adresów URL w funkcji `nativigateToURL()` powoduje otwarcie dokumentu w aplikacji zewnętrznej:

Schemat URL	Wynik wywołania funkcji nativeToURL()	Przykład

mailto:	Otwiera nową wiadomość w aplikacji pocztowej.	`str = "mailto:test@example.com";`
		`var urlReq:URLReq = new URLRequest(str);`
		`navigateToURL(urlReq);`

sms:	Otwiera wiadomość w aplikacji do edycji wiadomości SMS.	`str = "sms:1-415-555-1212";`
		`var urlReq:URLReq = new URLRequest(str);`
		`navigateToURL(urlReq);`

tel:	Wybiera numer na telefonie (z koniecznością potwierdzenia przez użytkownika).	`str = "tel:1-415-555-1212";`
		`var urlReq:URLReq = new URLRequest(str);`
		`navigateToURL(urlReq);`

Aplikacja na telefon iPhone może polegać na zainstalowanych certyfikatach głównych z podpisem własnym na potrzeby uwierzytelniania serwera w trakcie transakcji bezpiecznej, takiej jak żądanie https. Serwer powinien wysyłać nie tylko certyfikat końcowy (liść), lecz także certyfikaty pośrednie łączące z certyfikatem głównym.

Elementy interfejsu API ActionScript 3.0 o szczególnym znaczeniu dla programistów tworzących aplikacje na urządzenia mobilne

Poniższe elementy interfejsu API ActionScript 3.0 udostępniają funkcjonalność przydatną na urządzeniach mobilnych.

Interfejs API przyspieszeniomierza
Poniższe klasy umożliwiają aplikacji odbieranie zdarzeń z przyspieszeniomierza urządzenia:

- Accelerometer
- AccelerometerEvent

Więcej informacji można znaleźć w sekcji Dane z przyspieszeniomierza.

Interfejs API geolokalizacji
Poniższe klasy umożliwiają aplikacji odbieranie zdarzeń z czujnika lokalizacji urządzenia:

- Geolocation
- GeolocationEvent

Więcej informacji można znaleźć w sekcji Geolokalizacja.

Zdarzenia dotknięcia, Multi-Touch i gestów w interfejsie API
Poniższe klasy umożliwiają odbieranie przez aplikację zdarzeń dotknięcia i gestów:

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

Więcej informacji można znaleźć w sekcji Interakcje z wykorzystaniem dotknięć, Multi-Touch i gestów.

Rozdział 4: Zasady projektowania aplikacji na telefon iPhone

Szybkość przetwarzania oraz wielkość ekranu telefonu iPhone prowadzą do pewnych specyficznych zasad projektowania i pisania kodu aplikacji na ten telefon. Wiele z tych zasad ma zastosowanie zarówno nie tylko do aplikacji na urządzenia mobilne, ale i do aplikacji w ogóle.

Więcej informacji na temat optymalizacji aplikacji zawiera sekcja Optymalizacja treści mobilnej dla platformy Flash. Ten dokument zawiera wiele sugestii dotyczących optymalizacji wydajności treści urządzeń mobilnych, treści Flash Player, treści AIR oraz treści opartej na języku Action Script w ogóle. Większość z tych sugestii ma zastosowanie również do aplikacji AIR opracowanym z myślą o telefonie iPhone.

Ważne: Wiele z tych zaleceń oraz technik optymalizacji ma duże znaczenie dla opracowywania aplikacji poprawnie działających na telefonie iPhone.

Przyspieszanie sprzętowe

Możliwe jest wykorzystanie sprzętowej akceleracji OpenGL ES-1.1 w celu zwiększenia wydajności graficznej niektórych aplikacji. W wyniku zastosowania akceleratora sprzętowego mogą wiele zyskać gry oraz inne aplikacje, w których występują animowane obiekty wyświetlane. Aplikacje, w których stosuje się akcelerację sprzętową, mogą przerzucać niektóre zadania graficzne z procesora CPU na procesor GPU telefonu iPhone, co często skutkuje znacznym wzrostem wydajności.

Projektując aplikację korzystającą z procesora GPU, należy przestrzegać reguł zapewniających efektywną sprzętową akcelerację wyświetlania treści.

Aby użyć akceleracji sprzętowej, ustaw opcję Rendering dla GPU na karcie Ogólne w oknie dialogowym Ustawienia telefonu iPhone w programie Flash Professional CS5. Można również ustawić właściwość `renderMode` na wartość `gpu` w pliku deskryptora aplikacji:

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

Więcej informacji można znaleźć w sekcjach „Ustawianie właściwości aplikacji na telefon iPhone w programie Flash Professional CS5” na stronie 16 i „Ustawianie właściwości aplikacji na telefon iPhone w pliku deskryptora aplikacji” na stronie 18.

Wyróżnia się cztery klasy obiektów wyświetlanych, które mogą być szybko renderowane przy wykorzystaniu akceleracji sprzętowej, jeśli ich treść rzadko ulega zmianom:

- Obiekty bitmapowe.
- Dwuwymiarowe obiekty wyświetlane, których właściwość `cacheAsBitmap` ma wartość `true` i których właściwość `cacheAsBitmapMatrix` jest (opcjonalnie) ustawiona — zobacz niżej.
- Trójwymiarowe obiekty wyświetlane (tj. obiekty z ustawioną właściwością `z`).
- Obiekty wyświetlane o jednokolorowym prostokątnym wypełnieniu i krawędziach wyrównanych względem pikseli na ekranie.

Obiekty wektorowe są renderowane przy każdej animacji ikonki nad lub pod nimi. Każdy obiekt pełniący rolę pierwszego lub ostatniego planu animacji powinien również należeć do jednej z tych kategorii.

W przypadku obiektów wyświetlanych, dla których właściwość `cacheAsBitmap` została ustawiona na wartość `true`, ustawienie `cacheAsBitmapMatrix` powoduje użycie przez GPU bitmapy będącej rezultatem transformacji matrycy. GPU korzysta z reprezentacji bitmapy nawet, jeśli obiekt jest obracany albo skalowany. GPU może komponować i animować tę bitmapę znacznie szybciej niż procesor CPU może przerysowywać obiekt renderowany wektorowo.

Samo przypisanie właściwości `cacheAsBitmap` wartości `true` powoduje, że obiekt wyświetlany (i wszelkie jego obiekty podrzędne) jest buforowany w pamięci podręcznej. Obiekt wyświetlany nie jest przerysowywany, gdy odsłaniane są nowe regiony lub gdy następuje przemieszczenie całej kompozycji graficznej.

Po ustawieniu właściwości obiektu wyświetlanego `cacheAsBitmapMatrix` aplikacja może stworzyć reprezentację obiektu wyświetlanego nawet wówczas, gdy nie jest on widoczny. Aplikacja buduje reprezentację przechwytywanego obiektu wyświetlanego wraz z początkiem następnej klatki. Następnie, podczas dodawania obiektu wyświetlanego na stół montażowy, aplikacja szybko go renderuje. Aplikacja umożliwia także szybkie animowanie obrotu lub skalowania obiektu. W przypadku obiektów obracanych lub skalowanych nie należy ustawiać właściwości `cacheAsBitmap` bez ustawiania także właściwości `cacheAsBitmapMatrix`.

Aplikacja może również szybko przeprowadzać transformacje kanału alfa na obiekcie buforowanym jako bitmapa. Jednak w przypadku transformacji kanału alfa przyspieszanych sprzętowo obsługiwane są tylko wartości `alpha` z przedziału od 0 do 1,0. Przedział ten odpowiada ustawieniom właściwości `colorTransform.alphaMultiplier` z przedziału od 0 do 256.

Nie należy ustawiać właściwości `cacheAsBitmap` na wartość `true` w przypadku często aktualizowanych obiektów, takich jak pola tekstowe.

Obiekty wyświetlane o często zmieniającej się treści graficznej z reguły niezbyt dobrze nadają się do renderowania przy użyciu procesora GPU. Zasada ta dotyczy szczególnie starszych urządzeń ze słabszymi procesorami GPU. Narzut związany z przekazywaniem danych graficznych do procesora GPU może spowodować, że renderowanie przy użyciu procesora głównego (CPU) będzie bardziej efektywnym rozwiązaniem.

Należy zoptymalizować strukturę obiektów wyświetlanych zawierających podrzędne obiekty wyświetlane poruszające się względem obiektu nadrzędnego. Należy zmienić je w taki sposób, aby stały się obiektami równorzędnymi względem dotychczasowego obiektu nadrzędnego. Dzięki temu każdy z nich będzie miał swoją własną reprezentację bitmapową. Ponadto każdy obiekt wyświetlany będzie mógł przemieszczać się względem pozostałych bez konieczności przekazywania nowych danych graficznych do procesora GPU.

Właściwość `cacheAsBitmap` należy ustawić na wartość `true` na najwyższym poziomie listy wyświetlania, na którym nie są animowane podrzędne obiekty wyświetlane. Innymi słowy, należy ustawić je dla kontenerów obiektów wyświetlanych, które nie zawierają ruchomych części. Nie należy ustawiać ich dla podrzędnych obiektów wyświetlanych. *Nie* należy ustawiać ich dla ikonek zawierających inne obiekty wyświetlane, podlegające animacji.

W przypadku ustawienia właściwości `z` obiektu wyświetlanego aplikacja zawsze używa zbuforowanej reprezentacji bitmapy. W przypadku ustawienia właściwości `z` obiektu wyświetlanego aplikacja używa zbuforowanej reprezentacji bitmapy, nawet w przypadku obracania lub skalowania obiektu. Aplikacja nie używa właściwości `cacheAsBitmapMatrix` dla obiektów wyświetlanych, dla których ustawiono właściwość `z`. Ta sama reguła ma zastosowanie w przypadku ustawienia dowolnych właściwości trójwymiarowego obiektu wyświetlanego, w tym właściwości `rotationX`, `rotationY`, `rotationZ` i `transform.matrix3D`.

Nie należy ustawiać właściwości `scrollRect` ani `mask` kontenera obiektu wyświetlanego zawierającego treść, dla której ma zostać użyta akceleracja sprzętowa. Ustawienie tych właściwości dezaktywuje akcelerację sprzętową dla kontenera obiektu wyświetlanego i jego obiektów podrzędnych. Zamiast ustawiać wartość właściwości `mask`, można także nałożyć obiekt wyświetlany maski na obiekt maskowany.

Obowiązuje ograniczenie wymiarów obiektów wyświetlanych, które mogą być objęte akceleracją sprzętową. Na starszych urządzeniach obowiązuje ograniczenie szerokości i wysokości do 1024 pikseli (lub mniejszej wartości). Na nowszych urządzeniach obowiązuje ograniczenie do 2048 pikseli (lub mniejszej wartości). Narzędzie do diagnostyki renderowania przy użyciu procesora GPU umożliwia przetestowanie wydajności renderowania na urządzeniu.

Procesor GPU korzysta również z pamięci RAM telefonu iPhone do zapisywania obrazów bitmapowych. Korzysta on z co najmniej takiej ilości pamięci, jaka jest używana na potrzeby obiektów bitmapowych.

Procesor GPU korzysta z przydziałów pamięci stanowiących potęgę liczby 2 dla każdego z wymiarów obiektu bitmapowego. Na przykład procesor GPU może rezerwować pamięć na obrazy o wymiarach 512 x 1024 lub 8 x 32. Obraz o wielkości 9 x 15 pikseli zajmuje wówczas tę samą ilość pamięci, co obraz o wymiarach 16 x 16 pikseli. W przypadku buforowanych obiektów wyświetlanych celowe może być użycie obu wymiarów bliskich potędze 2 (ale nie większych). Na przykład bardziej efektywne jest użycie obiektu wyświetlanego o wielkości 32 x 16 pikseli niż obiektu o wielkości 33 x 17 pikseli.

Nie należy polegać na zmianie wymiarów obiektu Stage w przypadku konieczności zmniejszenia wielkości zasobów dostosowanych pod tym kątem do innych platform (takich jak pulpit). Zamiast tego należy użyć właściwości `cacheAsBitmapMatrix` lub zmienić wielkość zasobów przed opublikowaniem treści na potrzeby telefonu iPhone. Obiekty 3D ignorują właściwość `cacheAsBitmapMatrix` w przypadku przechwytywania obrazu powierzchni. Z tego względu bardziej przydatna może okazać się zmiana wielkości obiektów wyświetlanych przed ich opublikowaniem, o ile będą one renderowane na powierzchni 3D.

Jest to swego rodzaju kompromis pomiędzy zaletami akceleracji sprzętowej a wykorzystaniem pamięci RAM. W miarę zapełniania się pamięci system operacyjny iPhone OS informuje inne, działające aplikacje rodzime platformy iPhone o zwolnieniu pamięci. W miarę przetwarzania tej informacji przez aplikacje i zwalniania pamięci mogą one konkurować z bieżącą aplikacją o cykle procesora. Może to na krótki czas obniżyć wydajność bieżącej aplikacji. Należy koniecznie przetestować aplikację na urządzeniach starszego typu, ponieważ mogą one dysponować znacznie mniejszymi zasobami pamięci na potrzeby uruchamiania procesów.

Podczas debugowania aplikacji na telefonie iPhone można włączyć funkcję diagnostyki renderowania przy użyciu procesora GPU. Funkcja ta ułatwia ocenę wykorzystania procesora GPU do renderowania treści w aplikacji. Więcej informacji zawiera sekcja „Diagnostyka renderowania przy użyciu procesora GPU” w rozdziale „Debugowanie aplikacji na telefon iPhone” na stronie 26.

Więcej informacji na temat używania właściwości `cacheAsBitmapMatrix` zawiera sekcja „DisplayObject.cacheAsBitmapMatrix” w sekcji „Elementy interfejsu API ActionScript przeznaczone do tworzenia aplikacji AIR na urządzenia mobilne” na stronie 32.

Inne sposoby zwiększania wydajności obiektów wyświetlanych

Akcelerator sprzętowy pozwala przyspieszyć wydajność grafiki w niektórych klasach obiektów wyświetlanych. Oto kilka wskazówek dotyczących sposobów maksymalizacji wydajności graficznej:

- Należy podjąć próbę ograniczenia liczby elementów widocznych na stole montażowym. Renderowanie każdego elementu oraz skomponowanie go z innymi elementami zajmuje pewien czas.

  Gdy wyświetlanie danego obiektu wyświetlanego staje się zbędne, należy ustawić jego właściwość `visible` na wartość `false` lub usunąć go ze stołu montażowego (`removeChild()`). Nie należy po prostu ustawiać jego właściwości `alpha` na wartość 0.

- Należy w ogóle unikać trybów mieszania, a zwłaszcza trybów mieszania warstw. Tam, gdzie to tylko możliwe, należy używać trybów zwykłego mieszania.
- Należy pamiętać, że filtry obiektów wyświetlanych są obciążające obliczeniowo. Należy starać się rzadko ich używać. Może okazać się na przykład akceptowalne użycie kilku filtrów na ekranie wprowadzającym. Należy jednak unikać używania filtrów wobec wielu obiektów wyświetlanych lub obiektów, które są animowane, lub w sytuacjach, gdy niezbędne jest zastosowanie wysokiej częstości klatek.
- Należy unikać kształtów morficznych.
- Należy unikać przycinania.
- W miarę możliwości należy, wywołując metodę `Graphic.beginBitmapFill()`, ustawić parametr `repeat` na wartość `false`.
- Nie należy wymuszać zbyt wielu operacji rysowania. Jako tła należy używać koloru tła. Nie należy układać dużych kształtów warstwami — jeden na drugim. Narysowanie każdego piksela pociąga bowiem za sobą pewne obciążenie. Dotyczy to szczególnie obiektów wyświetlanych, które nie są objęte akceleracją sprzętową.
- Należy też unikać kształtów o długich cienkich końcach, krawędzi przecinających się i wielu drobnych detali wzdłuż krawędzi. Renderowanie tych kształtów zajmuje znacznie więcej czasu, niż wyświetlanie obiektów o łagodnych kształtach. Dotyczy to szczególnie obiektów wyświetlanych, które nie są objęte akceleracją sprzętową.
- Należy tworzyć bitmapy o wymiarach zbliżonych do 2^n na 2^m bitów, ale mniejszych. Wymiary nie muszą być potęgą liczby 2, ale powinny być do niej zbliżone i nie powinny być od niej większe. Na przykład obraz o wielkości 31 na 15 pikseli jest renderowany szybciej niż obraz o wielkości 33 na 17 pikseli. (31 i 15 to o jeden mniej niż potęgi liczby 2: 32 i 16). Takie obrazy pozwalają również na bardziej efektywne wykorzystanie pamięci.
- Wymiary obiektów wyświetlanych nie powinny przekraczać 1024 x 1024 pikseli (lub 2048 x 2048 na nowszych urządzeniach).

Gęstość informacji

Wielkość fizyczna ekranu urządzeń mobilnych jest dużo mniejsza niż komputerów stacjonarnych, ale gęstość pikseli na ich ekranach jest wyższa. Tekst charakteryzujący się wyostrzonymi krawędziami będzie wyglądał dobrze, lecz glify muszą mieć minimalną wielkość fizyczną, aby były wystarczająco czytelne.

Urządzenia przenośne są często używane w ruchu, a także przy nieodpowiednim oświetleniu. Należy więc zastanowić się, jak wiele informacji można w sposób czytelny wyświetlić na ekranie takiego urządzenia. Może być to mniej niż w przypadku monitora komputera stacjonarnego o tych samych wymiarach w pikselach.

Do wyróżnienia szczególnie ważnych informacji należy użyć hierarchii typograficznej. W celu podkreślenia znaczenia poszczególnych elementów interfejsu użytkownika można użyć wielkości czcionki, grubości i odstępów. Można jednak użyć jednego lub więcej wskaźników na każdym poziomie hierarchii. Wskaźniki te należy zastosować spójnie w całej aplikacji. Wskaźnik może dotyczyć szpalty (np. wcięcie, odstęp między liniami, umieszczenie) lub graficzny (wielkość, styl, kolor czcionki). Stosowanie nadmiarowych wskaźników może być skutecznym sposobem upewnienia się, że hierarchia jest wyrażona w sposób wyraźny. Należy jednak próbować używać nie więcej niż trzech wskaźników dla każdego poziomu grupowania.

Należy próbować upraszczać etykiety i tekst wyjaśnień. Należy na przykład używać przykładowych danych w polach tekstowych, w celu zasugerowania treści i uniknięcia osobnej etykiety.

Czcionki i wprowadzanie tekstu

W celu uzyskania najlepszych efektów należy używać czcionek urządzenia. Na przykład następujące czcionki są czcionkami urządzenia na telefonie iPhone:

- Szeryfowe: Times New Roman, Georgia, oraz _serif
- Bezszeryfowe: Helvetica, Arial, Verdana, Trebuchet, Tahoma oraz _sans
- O stałej szerokości: Courier New, Courier oraz _typewriter

Należy używać czcionek o wielkości 14 pikseli lub większej.

Należy używać czcionek urządzenia w przypadku edytowalnych pól tekstowych. Czcionki urządzenia w polach tekstowych również renderowane są znacznie szybciej niż czcionki osadzone.

Nie należy używać podkreślonego tekstu na potrzeby pól tekstu wejściowego. Nie należy również ustawiać wyrównania pola tekstowego. Pola tekstu wejściowego telefonu iPhone obsługują tylko wyrównanie do lewej (ustawienie domyślne).

W przypadku używania dla pola tekstowego w programie Flash Professional CS5 ustawienia Tekst TLF należy wyłączyć wspólną bibliotekę wykonawczą w powiązaniu domyślnym w ustawieniach języka ActionScript 3.0. W przeciwnym wypadku aplikacja nie będzie działać na telefonie iPhone, ponieważ będzie podejmowała próby użycia pliku SWF biblioteki współużytkowanej czasu wykonywania:

1. Wybierz polecenie Plik > Ustawienia publikowania.
2. W oknie dialogowym Ustawienia publikowania kliknij kartę Flash.
3. Kliknij przycisk Skrypt na prawo od listy rozwijanej Skrypt (ActionScript 3.0).
4. Kliknij kartę Ścieżka biblioteki.
5. Na liście rozwijanej Powiązanie domyślne wybierz opcję Scalone z kodem.

Należy rozważyć zastosowanie opcji alternatywnych w stosunku do używania pól tekstu wejściowego. Na przykład do wprowadzenia wartości numerycznej nie jest potrzebne pole tekstowe. Ten sam efekt można osiągnąć za pomocą dwu przycisków, jednego do zwiększania, zaś drugiego do zmniejszania wartości.

Należy mieć na uwadze obszar zajmowany przez wirtualną klawiaturę. Z chwilą aktywacji klawiatury wirtualnej (na przykład po stuknięciu pola tekstowego przez użytkownika) aplikacja dostosowuje położenie stołu montażowego. Automatyczna zmiana położenia zapewnia pełną widoczność wybranego pola tekstu wejściowego:

- Pole tekstowe w górnej części stołu montażowego jest przenoszone do górnej części widocznego obszaru stołu montażowego. (Obszar widoczny stołu montażowego jest mniejszy, co pozwala na zmieszczenie klawiatury wirtualnej.)
- Pole tekstowe w dowolnej części stołu montażowego pozostaje w dolnej części nowego obszaru stołu.
- Pole tekstowe w innej części stołu jest przenoszone w kierunku środka (w pionie) stołu montażowego.

W przypadku kliknięcia przez użytkownika pola tekstowego w celu jego edycji (oraz w przypadku wyświetlenia wirtualnej klawiatury) obiekt TextField wywoła zdarzenie `focusIn`. Możliwe jest dodanie detektora zdarzeń dla tego zdarzenia w celu zmiany położenia pola tekstowego.

Jednowierszowe pole tekstowe obejmuje przycisk kasowania (na prawo od tekstu) wyświetlany wówczas, gdy użytkownik edytuje pole tekstowe. Ten przycisk kasowania nie jest jednak wyświetlany, jeśli pole tekstowe jest zbyt wąskie.

Po edycji tekstu w jednowierszowym polu tekstowym użytkownik ukrywa wirtualną klawiaturę, dotykając klawisza Done na klawiaturze.

Po zakończeniu edycji tekstu w wielowierszowym polu tekstowym użytkownik ukrywa wirtualną klawiaturę, dotykając obszaru poza polem tekstowym. Powoduje to usunięcie aktywności z pola tekstowego. Upewnij się, że podczas wyświetlania wirtualnej klawiatury projekt obejmuje obszar spoza pola tekstowego. Jeśli pole tekstowe jest za duże, może nie być widoczny żaden inny obszar.

Korzystanie z niektórych składników programu Flash Professional CS5 może uniemożliwić usuwanie aktywności z pola tekstowego. Te składniki są przeznaczone do używania na komputerach stacjonarnych, na których takie działania związane z aktywnością są pożądane. Jednym z takich składników jest składnik TextArea. Gdy jest on w stanie aktywności (i jest edytowany), nie można przenieść aktywności przez kliknięcie innego obiektu wyświetlanego. Umieszczenie innych składników programu Flash Professional CS5 na stole montażowym może również uniemożliwić przeniesienie aktywności z poziomu edytowanego pola tekstowego.

Nie należy polegać na zdarzeniach klawiatury. Niektóre treści SWF projektowane z myślą o wykorzystaniu w sieci WWW mogą na przykład umożliwiać użytkownikowi sterowanie ich działaniem za pośrednictwem klawiatury. W telefonie iPhone klawiatura wirtualna jest jednak wyświetlana wyłącznie wówczas, gdy użytkownik edytuje pole tekstowe. W trakcie, gdy wyświetlana jest klawiatura wirtualna, aplikacja na telefon iPhone wywołuje tylko zdarzenia klawiatury.

Zapisywanie stanu aplikacji

Aplikacja może zostać zamknięta w dowolnej chwili (na przykład z chwilą wykrycia połączenia przychodzącego). Należy więc rozważyć celowość zapisywania stanu aplikacji z chwilą każdej jej zmiany. Można na przykład zapisać ustawienia w pliku lub bazie danych w katalogu magazynu aplikacji. Można również zapisać dane w lokalnym obiekcie współużytkowanym. Następnie można przywrócić stan aplikacji po jej ponownym uruchomieniu. Jeśli działanie aplikacji zostanie przerwane przez połączenie przychodzące, zostanie ono wznowione po zakończeniu połączenia.

Nie należy przy tym polegać na obiekcie NativeApplication wywołującym zdarzenie `exiting` z chwilą zamknięcia aplikacji; taki mechanizm może nie zadziałać.

Zmiany orientacji ekranu

Treść wyświetlaną na telefonie iPhone można oglądać w orientacji pionowej lub poziomej. Należy więc rozważyć sposób obsługi zmian orientacji ekranu przez tworzoną aplikację. Więcej informacji można znaleźć w sekcji Wykrywanie orientacji urządzenia.

Cele dotknięć i kliknięć

Projektując przyciski oraz inne elementy interfejsu użytkownika przeznaczone do stukania przez użytkownika, należy przemyśleć ich rozmiar. Elementy te powinny być odpowiednio duże, tak aby ich aktywacja możliwa była w wygodny sposób za pomocą palca na ekranie dotykowym. Należy również upewnić się, że między miejscami docelowymi jest odpowiednia ilość miejsca. Elementy będące celem dotknięć i kliknięć powinny mieć wymiary około 44 do 57 pikseli.

Alokacja pamięci

Alokacja nieużywanych wcześniej bloków pamięci jest mocno obciążająca. Może ona spowodować spowolnienie aplikacji lub problemy z wydajnością podczas animacji lub interakcji z chwilą uruchomienia procesu porządkowania pamięci.

Należy próbować ponownie wykorzystywać już istniejące obiekty wszędzie tam, gdzie to tylko możliwe, zamiast pozbywania się ich i tworzenia nowych.

Należy pamiętać, że obiekty wektorowe zabierają zwykle mniej pamięci, niż tablice. Więcej informacji zawiera sekcja Klasa Vector a klasa Array.

Więcej informacji na temat wykorzystania pamięci zawiera sekcja Oszczędzanie pamięci.

Interfejs API rysowania

Należy próbować unikać interfejsu API rysowania ActionScript (klasy Graphics) do tworzenia grafiki. Używanie interfejsu API rysowania powoduje tworzenie obiektów w sposób dynamiczny na stole montażowym, a następnie ich renderowanie do postaci rastrowej. W miarę możliwości należy tworzyć takie obiekty statycznie, w czasie tworzenia aplikacji.

Obiekty utworzone za pomocą interfejsów API, wywoływane w sposób powtarzalny, są niszczone i tworzone ponownie przy każdym wykonaniu kodu ActionScript. Obiekty statyczne pozostają jednak w pamięci na różnych osiach czasu.

Propagacja zdarzeń

W przypadku głęboko zagnieżdżonych kontenerów obiektów wyświetlanych propagacja zdarzeń może okazać się kosztowna obliczeniowo. Obciążenie to można zredukować, obsługując zdarzenie w pełni w obiekcie docelowym, a następnie wywołując metodę `stopPropagation()` obiektu zdarzenia. Wywołanie tej metody eliminuje propagację zdarzenia w górę hierarchii detektorów. Wywołanie tej metody oznacza również, że obiekty nadrzędne nie odbierają zdarzeń.

Powiązane cele można zrealizować również, spłaszczając zagnieżdżenie obiektu wyświetlanego w celu uniknięcia długich ciągów zdarzeń.

W miarę możliwości zamiast zdarzeń TouchEvent należy rejestrować zdarzenia MouseEvent. Zdarzenia MouseEvent obciążają procesor w mniejszym stopniu niż zdarzenia TouchEvent.

W miarę możliwości należy ustawić właściwości `mouseEnabled` i `mouseChildren` na wartość `false`.

Optymalizacja odtwarzania wideo

Aby zoptymalizować odtwarzanie wideo na urządzeniu przenośnym, należy zadbać o to, aby podczas odtwarzania wideo uruchomionych było jak najmniej aplikacji. Umożliwia to maksymalne wykorzystanie procesora na potrzeby procesów dekodowania i renderowania wideo.

Podczas odtwarzania wideo nie powinny być uruchomione duże ilości kodu ActionScript. Należy starać się unikać kodu uruchamianego z zegarem o wysokiej częstotliwości między interwałami lub na osi czasu.

Należy minimalizować operacje przerysowywania obiektów wyświetlanych innych niż materiał wideo. W szczególności należy unikać przerysowywania obiektów wyświetlanych przecinających się z obszarem wideo. Zasada ta obowiązuje również, jeśli są one ukryte pod materiałem wideo. Będą one mimo wszystko przerysowywane i będą zużywały zasoby potrzebne do przetwarzania. Należy na przykład dążyć do stosowania prostych kształtów wskaźników położenia a także obniżyć częstotliwość aktualizacji położenia do kilku razy na sekundę zamiast aktualizacji co 1 klatkę. Elementy sterujące materiałem wideo nie powinny zachodzić na obraz wideo. Należy umieścić je bezpośrednio poniżej. W przypadku animacji z buforowaniem wideo nie należy ukrywać jej za materiałem wideo, gdy nie jest ona używana. Należy ją ustawić jako niewidoczną.

Składniki środowisk Flex i Flash

Wiele składników środowisk Flex i Flash zaprojektowano z myślą o zastosowaniu w lokalnym środowisku stacjonarnym. Składniki te, a zwłaszcza składniki wyświetlane, mogą działać mało wydajnie na urządzeniach mobilnych. Innym problemem związanym z takimi składnikami, obok niskiej wydajności, mogą być modele interakcji niedostosowane do specyfiki urządzeń mobilnych.

Firma Adobe opracowuje wersję środowiska Flex zoptymalizowaną dla urządzeń mobilnych. Więcej informacji można znaleźć pod adresem http://labs.adobe.com/technologies/flex/mobile/.

Redukowanie wielkości pliku aplikacji

Oto niektóre wskazówki dotyczące redukcji wielkości pliku IPA:

- Należy sprawdzić bitmapy w tle, upewniając się, że mają one właściwy rozmiar (nie większy niż potrzeba).
- Należy sprawdzić, czy osadzono jakiekolwiek dodatkowe czcionki.
- Należy wyszukać kanały Alfa w zasobach PNG i usunąć je, jeśli nie są potrzebne. Należy użyć narzędzia takiego jak PNG Crunch do zmniejszenia wielkości zasobów PNG.
- Należy tam, gdzie to możliwe, przekształcić zasoby PNG w zasoby JPG.
- Należy rozważyć kompresję plików dźwiękowych (z zastosowaniem niższej szybkości transmisji)
- Należy usunąć wszelkie nieużywane zasoby.