AirPlayの概要

Developer

目次

図、表、リスト

レイヤについて

オーディオ/ビデオ素材をウェブサイト上に置いてユーザに提供する、あるいはこれを再生するiOSアプリケーションを開発するためには、AirPlayについて知っておかなければなりません。ユーザはAirPlayを利用して、オーディオ/ビデオ素材を、高精細（HD）のビデオ表示装置や高忠実度（Hi-Fi）のオーディオシステムで再生できます。また、iTunesやiOSベースのデバイスからApple TVに（そしてホームシアターシステムに）、あるいはAirPlay対応のサウンドシステムに転送することも可能です。その素材は、インターネット経由でライブ放送されているものでも、iTunesやiOSベースのデバイスに格納されたものでも構いません。AirPlayでは、インターネット経由で配信されるオーディオ/ビデオを、iOSアプリケーション、iOSベースのデバイス上のSafariブラウザ、あるいはiTunes（プラットフォームを問わない）で再生する場合に、ストリーミング配信が可能です。

AirPlayは、iOS 4.3以降、iTunes 10.2以降、Mac OS X、Windows上で動作します。

At a Glance

ユーザはAirPlayを使って、Apple TVやAirPlay対応サウンドシステム、リモートスピーカーなどに、オーディオ/ビデオ素材を転送、再生できます。AirPlayはユーザ側が制御するシステムです。コンテンツプロバイダ側の主な役割は、供給する素材をAirPlayに適合した形式にすること、アプリケーションやウェブサイトがAirPlayと正しく連携できるようにすることです。また、ホームシアターシステムでの再生を想定すれば、高精細ビデオやサラウンドサウンドオーディオを供給する必要もあるでしょう。

アプリケーション開発者の立場であれば、AirPlayがより魅力的になるよう、次のようなことを考慮してください。

- AirPlay出力デバイスの選択ツール（ピッカ）を提供する。
- オーディオのメタデータを供給する（これはAirPlay対応出力デバイスに表示される）。
- AirPlay出力デバイスからのリモートコントロールイベント（再生/一時停止など）を監視し、応答する。
- セカンドディスプレイが使える場合、iOSデバイス側には別の画面を表示する。

オーディオ/ビデオの再生方法

AirPlayに適合したオーディオ/ビデオは、次の4通りの方法でiOSデバイスに配信できます。

- ウェブサイト上に置いたオーディオ/ビデオ素材を、HTML5の`<audio>`タグ、`<video>`タグを使って埋め込む方法。この方法は、iOSベースのデバイス上で、Safariブラウザや、Web Viewが組み込まれたアプリケーションを介してアクセスする場合に適しています。
- オーディオ/ビデオをアプリケーションに直接配信し、AV Foundationの`MPMoviePlayer`クラスや`UIWebView`クラスで再生する方法。
- 素材をiOSベースのデバイス上に、ローカルに保存する方法。
- iTunesを利用して、商用的に、またはポドキャストとして配信する方法。

AirPlayは配信方法として、プログレッシブダウンロードにも、HTTP Live Streamingにも対応していますが、HTTP Live Streamingの方を推奨します。

- Safari上で再生することを想定してウェブサイトを設計する場合、ビデオ素材をプログレッシブダウンロードやHTTP Live Streamingで配信すれば、iOSベースのデバイス上で、AirPlayに処理させることができます。
- 少なからぬ量のビデオ素材を再生するiOSアプリケーションを開発する場合、その認定を受けるためには、HTTP Live Streamingに対応する必要があります。これは、可搬デバイス上では、HTTP Live Streamingの方がユーザにとって使い勝手がよいからです。ネットワーク接続速度の変化に応じ、帯域幅の異なるストリームを動的に切り替えることが可能です。

  アプリケーションに対する要件その他、詳しくは『*HTTP Live Streaming Overview*』を参照してください。

AirPlayで再生するためには、ビデオならばH.264、オーディオならばAACやMP3のように、AirPlayに適合した形式にする必要があります。サーバの設定は特に必要ありません。素材の送信時に用いるファイルの拡張子に応じ、MIMEタイプを正しく関連づけるだけで充分です。

ホームシアターシステムには、高解像度ディスプレイを装備し、AC3サラウンドサウンドを再生できるものがあるので、代替のオーディオ/ビデオストリームを提供すれば、ユーザにとってAirPlayがより魅力的になるでしょう。代替ストリームには、AC3コーデックと1280×720の解像度を指定した、代替ストリームプレイリストを使います。

メディアの形式、MIMEタイプ、プレイリストについては、「オーディオ/ビデオおよびサーバをAirPlayに適合させる手順」（8 ページ）を参照してください。

AirPlayのビデオ処理機能はオプトインやオプトアウトができる

AirPlayは、ユーザが意識的にその機能を利用起動するもの、すなわち、ユーザ側で制御するものです。（システムサウンド以外の）オーディオは、いつでもユーザが操作して、AirPlay対応サウンドシステムに転送できます。iTunes用のオーディオ/ビデオも同様です。しかし、ビデオをウェブサイト上

で配布し、あるいはアプリケーション内で表示する場合に、AirPlayにオプトインしてユーザが転送できるようにすること、あるいは、AirPlayをオプトアウトして、受信したデバイスでのビデオ再生を制限することも可能です。

iOS 5.0までは、ユーザがビデオを転送できるようにするためには、明示的にAirPlayにオプトインする必要がありました。これに対し、ベースSDKをiOS 5.0以降（iOS 5.0のSafariを含む）に設定してコンパイルしたアプリケーションは、初めからAirPlayが有効になっています。したがって、ユーザがビデオをApple TVに転送できないようにしたいならば、明示的にAirPlayをオプトアウトする必要があります。

AirPlayをオプトイン、オプトアウトする具体的な手順は、ビデオの再生方法によって異なります。詳しくは「AirPlayのオプトインとオプトアウト」（11 ページ）を参照してください。

魅力的なシステムの提供

アプリケーションのユーザインターフェイスにAirPlayピッカを組み込めば、ユーザは、別のアプリケーションに切り替えたりフォーカスを移したりすることなく、出力先AirPlayデバイスを選択できるようになります。

アプリケーションのオーディオ/ビデオをAirPlay対応デバイスに転送する場合、そのデバイスにも表示画面や制御ボタン類がついているかも知れません。したがって、AirPlay対応デバイス側に表示する、メタデータ（曲名、アルバムのジャケット画像など）を供給するようにしてください。また、アプリケーションは、転送先デバイスから送られてくるリモートコントロールイベント（再生/一時停止、次のトラックへのジャンプなど）にも応答できるようにする必要があります。

アプリケーションのビデオを、ユーザが外づけディスプレイで視聴している場合、ホストデバイスの組み込みディスプレイ側には代替または追加の画像を表示することにすれば、2つの画面を有効活用できます。

実装の詳細については、「より魅力的なAirPlay対応アプリケーション」（14 ページ）を参照してください。

暗号化と認証の機能により、オーディオ/ビデオ素材を保護できる

オーディオ/ビデオをHTTP Live Streamingで配信する際には、組み込みの暗号化機能（暗号化キーと初期化ベクトルを定期的に自動生成する処理を含む）が使えます。詳細については、『*HTTP Live Streaming Overview*』を参照してください。一方、HTTPSプロトコルでキーを配信して認証機能を利用する場合、初期認証ハンドシェイクの処理はアプリケーション側の責任になります。

詳細については、「暗号化と認証」（20 ページ）を参照してください。

関連項目

AirPlayのオーディオ/ビデオについてさらに詳しくは、Appleが発行している次の資料を参照してください。

- *HTTP Live Streaming Overview* - HTTP Live Streamingをセットアップし、一般的なウェブサーバからHTTPプロトコルでライブ/オンデマンドのオーディオ/ビデオを配信する手順を説明しています。
- *Safari HTML5 Audio and Video Guide* — ウェブサイト上で、オーディオ/ビデオを埋め込んで配信する方法を説明しています。
- *AVPlayer Class Reference* — `AVPlayer`クラスについて、AirPlay特有のプロパティを含め説明しています。
- *MPMoviePlayerController Class Reference* — `MPMoviePlayerController`クラスについて、AirPlay特有のプロパティを含め説明しています。
- *UIWebView Class Reference* — `UIWebView`クラスについて、AirPlay特有のプロパティを含め説明しています。

オーディオ/ビデオおよびサーバをAirPlayに適合させる手順

AirPlayで再生するオーディオ/ビデオは、iPhone、iPod touch、iPadなど、iOSベースのデバイスで再生できるものでなければなりません。サーバがAirPlayと適切に連携できるよう、HTTPプロトコル上でオーディオ/ビデオを送信する設定が必要です。

オーディオ/ビデオ素材の準備

AirPlayで再生する素材に対して、次のような処理が必要です。

- オーディオはモノラル/ステレオのAACまたはMP3で圧縮する。
- ビデオはH.264形式に圧縮する。

  iPhone 3G以前のデバイスで再生する場合は、ベースラインプロファイル3.0に従います。

  iPhone 4以降、iPod touch、iPad、Apple TVで再生する場合は、ベースラインプロファイル3.1に従ってください。

  もっぱらiPad、Mac OS X、Apple TVでのみ再生する場合は、メインプロファイル3.1に従います。

- ホームシアターシステムでの再生用に、代替オーディオ/ビデオを用意する。これは、QuickTime参照ムービー、または代替ストリームプレイリストを使い、AC3サラウンドサウンドトラックと高精細ビデオにしたものです。

HTTP Live Streamingにより、ネットワークの帯域幅に応じて配信するビデオを切り替える場合は、推奨するビデオレートおよびエンコーダ設定について、『*HTTP Live Streaming Overview*』を参照してください。

オーディオデータは、`.mp3`、`.aac`、`.m4a`、`.m4v`、`.mp4`、`.mov`の、いずれかの形式のファイルとします。あるいは、HTTP Live Streamingメディアセグメンタに、AAC圧縮オーディオとH.264圧縮ビデオから成るMPEG-2トランスポートストリームの形で、直接配信することも可能です（ライブ放送など）。

また、`.m3u8`プレイリストを作成することもできますが、これは通常、HTTP Live Streamingのサーバソフトウェアが自動生成します。このソフトウェアは、オーディオ/ビデオ素材から`.ts`ファイルも生成します。

AC3オーディオ、高精細ビデオの配信

オーディオ素材を再生するAirPlay対応サウンドシステムは、AC3サラウンドサウンドオーディオにも対応しているかも知れません。AC3も選択肢にある代替ストリームプレイリストを配信すれば、出力デバイスがこれに対応している場合、より深みのあるサウンドを再生できるようになります。同様にAirPlay出力デバイスは、高精細ビデオも表示できるかも知れません。1280×720のビデオストリームを配信すれば、ユーザはより迫力のある画像を楽しめます。

代替ストリームプレイリストはHTTP Live Streamingの機能です。代替ストリームは通常、ビットレートで指定しますが、画面解像度や必要なコーデックでも指定できます。こうしておけば、出力デバイスが対応している場合に限り、AC3オーディオや高精細ビデオストリームを選択できるようになります。

AC3オーディオを代替ストリームとして再生するための要件は、マスタプレイリストの`CODECS`パラメータで指定します。AC3オーディオのコーデックは「`ac-3`」という文字列で表します。同様に、ビデオストリームの再生に最低限必要な解像度は、`RESOLUTION`パラメータで設定します。

たとえば次のプレイリストはそれぞれ、狭帯域ストリーム、広帯域ストリーム、そして、1280×720のディスプレイ、AC3オーディオの再生機能、1.5Mbit/sのインターネット接続を要するAirPlayストリームを指定しています。

```
#EXTM3U
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=150000
http://example.com/low/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=640000
http://example.com/high/index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=1500000,
RESOLUTION=1280x720,CODECS="avc1.42e01e,ac-3"
http://example.com/airplay/index.m3u8
```

代替ストリームプレイリストについて詳しくは、『*HTTP Live Streaming Overview*』を参照してください。

サーバの設定

サーバは、HTTPプロトコルでオーディオ/ビデオを送信できるよう設定する必要があります。このとき最も重要なのは、ファイル拡張子に応じて正しいMIMEタイプを割り当てることです。表 1-1に、MIMEタイプとファイル拡張子の対応を示します。

表 1-1 ファイル拡張子とMIMEタイプ

ファイル拡張子	MIMEタイプ
`.m3u8`	`application/x-mpegURL`または `application/vnd.apple.mpegURL`
`.ts`	`video/MP2T`
`.mov`	`video/quicktime`
`.mp3`	`audio/MPEG3`
`.aac`	`audio/aac`
`.m4a`	`audio/mpeg4`
`.m4v`, `.mp4`	`video/mpeg4`

ヒント HTTP Live Streamingでオーディオ/ビデオを配信する場合、サーバから`.m3u8`ファイルを配信する際に実行時gzip圧縮を有効にすることにより、伝送効率を改善できます。

AirPlayのオプトインとオプトアウト

iTunesやiPodアプリケーションで再生可能なオーディオコンテンツ（システムサウンドを除く）やオーディオメディアは、AirPlayで常時、利用できます。アプリケーションで再生するビデオ、ウェブサイト上に置いて提供するビデオは、提供者側の判断で、AirPlayで扱えるようにすることも、扱えないようにすることも可能です。

iOS 5.0より前のバージョンでは、オーディオ/ビデオをApple TVで再生できるようにするためには、AirPlayにオプトインする必要がありました。ベースSDKをiOS 5.0以降に設定してコンパイルしたアプリケーションでは、初めからAirPlayが有効になっています。したがって、Apple TVに転送し、iOSベースのデバイスでビデオを再生することを禁じたい場合は、明示的にAirPlayを無効にしなければなりません。

iOS 5.0以降には初めから、ウェブコンテンツを扱えるようSafariがオプトインされています。

ウェブサイト側でAirPlayの有効/無効を切り替える

ウェブサイト上に置いたビデオを、AirPlay経由で表示できるようにしたい場合は、HTML5の`<video>`タグを使ってウェブページに埋め込みます。詳細については、『*Safari HTML5 Audio and Video Guide*』を参照してください。

明示的にAirPlayにオプトインするためには、`video`タグの`x-webkit-airplay`属性、または`embed`タグの`airplay`属性を、`"allow"`と設定します（リスト 2-1を参照）。

リスト 2-1　AirPlayでビデオを扱えるようにする記述

```
<video src="myPlaylist.m3u8"
       height="300" width="400"
       x-webkit-airplay="allow" >

  <embed airplay="allow"
       src="movie.mov"
       width=400
       height=300
       mime-type="video/quicktime">
```

```
  </embed>

</video>
```

明示的にAirPlayをオプトアウトしたい場合は、videoタグのx-webkit-airplay属性、またはembedタグのairplay属性を、"disallow"と設定します（リスト 2-2を参照）。

リスト 2-2　AirPlayでビデオを扱えないようにする記述

```
<video src="myPlaylist.m3u8"
       height="768" width="1024"
       x-webkit-airplay="disallow" >

</video>

<!-- or -->

<embed airplay="disallow"
       src="movie.mov"
       width=320
       height=240
       mime-type="video/quicktime">

</embed>
```

アプリケーション側でAirPlayの有効/無効を切り替える

iOS 5.0以降、アプリケーションがビデオの表示にAV Foundation、MPMoviePlayerControllerクラス、UIWebViewクラスのいずれかを使っていれば、AirPlayによるビデオの取り扱いは自動的に有効になります。ただし、旧iOSとの互換性を維持するため、明示的にAirPlayにオプトインするとよいでしょう。一方、ビデオを扱えないよう、明示的にAirPlayを無効にすることも可能です。その具体的な方法は、ビデオ再生に用いるAPIによって異なります。

AVPlayerを使って実装したアプリケーションの場合

- `AVPlayer`クラスを使ってビデオ表示機能を組み込んでいるアプリケーションでは、`allowsAirPlayVideo`プロパティを`YES`とすることにより、明示的にAirPlayを有効にすることができます。
- 無効にしたい場合は、`allowsAirPlayVideo`プロパティを`NO`としてください。
- AirPlay上でビデオを再生しているかどうか確認したい場合は、`airPlayVideoActive`プロパティの状態を調べます。

注 `airPlayVideoActive`プロパティはキー値監視に対応しています。アプリケーションのオブジェクトをオブザーバとして登録すれば、このプロパティの値が変わったとき、通知が届くようになります。

MPMoviePlayerControllerを使って実装したアプリケーションの場合

- `MPMoviePlayerController`クラスを使ってビデオ表示機能を組み込んでいるアプリケーションでは、`allowsAirPlay`プロパティを`YES`とすることにより、明示的にAirPlayを有効にすることができます。
- 無効にしたい場合は、`allowsAirPlay`プロパティを`NO`としてください。
- AirPlay上でビデオを再生しているかどうか確認したい場合は、`airPlayVideoActive`プロパティの状態を調べます。

注 `MPMoviePlayerIsAirPlayVideoActiveDidChangeNotification`通知を登録することにより、`airPlayVideoActive`プロパティの変化を監視できます。

UIWebViewを使って実装したアプリケーションの場合

- `UIWebView` APIを使ってビデオ表示機能を実装しているアプリケーションでは、`mediaPlaybackAllowsAirPlay`プロパティを`YES`とすることにより、明示的にAirPlayを有効にすることができます。さらに、Web Viewに表示するためには、コンテンツ側でAirPlayを無効にしてはなりません。
- 無効にしたい場合は、`mediaPlaybackAllowsAirPlay`プロパティを`NO`としてください。

より魅力的なAirPlay対応アプリケーション

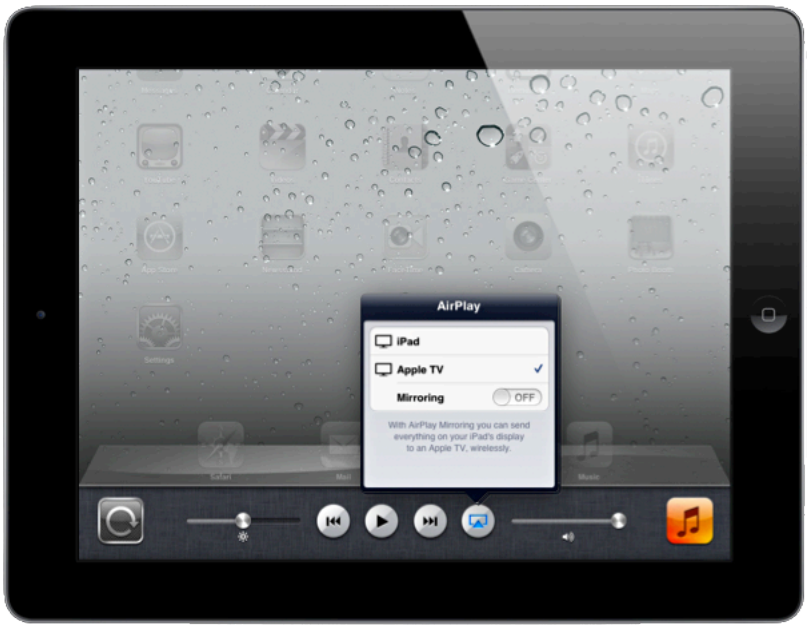
AirPlayに対応したアプリケーションに次のような工夫を施すと、より魅力的になるでしょう。

- AirPlay出力デバイスの選択ツール（ピッカ）を組み込む。
- アプリケーションオーディオはすべてAirPlay出力デバイスに転送して再生する一方、ユーザインターフェイスに関係するサウンドは、転送せずホストデバイス側で鳴るようにする。
- オーディオメタデータを提供し、AirPlay出力デバイスに表示されるようにする。
- AirPlay出力デバイスからの遠隔イベントに応答する。
- ホストデバイスの画面用とAirPlay出力画面用にそれぞれ別々のビデオを配信して、画面が2つあることを有効に活用する。
- 高精細ビデオとAC3オーディオの代替データストリームを配信する（「オーディオ/ビデオおよびサーバをAirPlayに適合させる手順」（8 ページ）の「AC3オーディオ、高精細ビデオの配信」（9 ページ）を参照）。

AirPlayピッカを組み込む

ユーザは、iOSマルチタスキングインターフェイスのシステムAirPlayピッカを使って、AirPlayの出力先を選択できます（図 3-1を参照）。AirPlay出力デバイスが使える場合、AirPlayボタンが音量スライダの隣に現れます。このボタンを押すと、切り替え可能なデバイスのリストが表示されます。

図 3-1　システムAirPlayピッカ



オーディオ/ビデオを再生するアプリケーションにはAirPlay出力ピッカを組み込んで、フォーカスを移さなくても切り替えができるようにするとよいでしょう。AirPlayピッカは、次のコード例のように`MPVolumeView`を使って、メディア再生コントロールに追加できます。

```
MPVolumeView *volumeView = [ [MPVolumeView alloc] init] ;
[view addSubview:volumeView];
```

メディアコントローラを独自に組み込んでいて、標準の音量コントローラは使いたくない場合は、次のようにすればAirPlayピッカだけを単独で追加できます。

```
MPVolumeView *volumeView = [ [MPVolumeView alloc] init] ;
[volumeView setShowsVolumeSlider:NO];
[volumeView sizeToFit];
[view addSubview:volumeView];
```

ピッカが表示されるのは、AirPlay出力デバイスが使える場合に限ります。

オーディオをその目的に応じて振り分ける

アプリケーションは通常、2種類のサウンドを使います。アプリケーションオーディオ（環境音、バックグラウンド音楽、偶発的なノイズなど）と、システムサウンド（クリック音、警告音など）です。AirPlayは、アプリケーションオーディオを遠隔サウンドシステムに配信する一方、システムサウンドはホスト側に残して、フィードバック音が入力デバイスの近くで鳴るようにするものと想定しています。

アプリケーションオーディオの再生にシステムサウンドAPIを使うと、AirPlay対応サウンドシステムに転送されなくなってしまいます。これはユーザにとっては望ましい動作とは言えないでしょう。システムサウンドAPIは、システムサウンドの再生にのみ用いることが大切です。アプリケーションオーディオの再生には、`AVAudioPlayer`などのAPIを使ってください。

オーディオメタデータを配信する

オーディオが再生されるのは、大画面のホームシアターシステム、あるいはLCDディスプレイつきのサウンドシステムかも知れません。アーティスト名、曲名、アルバムのジャケット画像など、AirPlayデバイスの画面に表示されるメタデータを配信すれば、ユーザの楽しみが増すことでしょう。

メタデータの追加は、ディクショナリを`MPNowPlayingInfoCenter`の`setNowPlayingInfo`メソッドに渡して行います。`MPNowPlayingInfoCenter`クラスは`MediaPlayer`フレームワークの一部ですが、`MediaPlayer`、`AVFoundation`、`AudioQueue`など、あらゆる再生フレームワークと連携して動作します。

一般的な曲情報（文字列）だけでなく、再生レート、経過時間、総再生時間も渡してください。再生デバイスでは、経過時間と再生レートの情報に基づき、現在の再生位置を表すバーを表示できます。再生レートが変わったら、経過時間と再生レートを更新してください。

リモートコントロールイベントに応答する

AirPlayの使用中、オーディオ/ビデオを再生しているのは、ホストデバイスとは別の部屋かも知れません。AirPlay出力デバイスには、独自のコントローラが附属し、あるいはApple Remoteに対応していることがあります。したがってアプリケーションは、再生、一時停止、早送りなどのリモートコントロールイベントを監視し、応答できるようにするべきでしょう。リモートコントロールイベントを有効にすると、ホストデバイスに物理的に差し込んだヘッドフォンやイヤホンでもコントロールできるようになります。

リモートコントロールイベントを受信するコード例を示します。

```
- (BOOL) canBecomeFirstResponder {return YES;}
- (void) viewDidAppear: (BOOL) animated {
    [super viewDidAppear:animated];
    [ [UIApplication sharedApplication] beginReceivingRemoteControlEvents];
    [self becomeFirstResponder];
}
```

再生が終わったら、以後はリモートコントロールイベントを受信する必要がない旨を、次のコードでシステムに通知します。

```
- (void) viewWillDisappear: (BOOL) animated {
    [super viewWillDisappear:animated];
    [ [UIApplication sharedApplication] endReceivingRemoteControlEvents];
    [self resignFirstResponder];
}
```

アプリケーションの動作に応じ、`UIEventTypeRemoteControl`およびそのサブタイプのイベントに応答します。そのコード例を以下に示します。

```
- (void) remoteControlReceivedWithEvent: (UIEvent *) receivedEvent {
    if (receivedEvent.type == UIEventTypeRemoteControl) {
        switch (receivedEvent.subtype) {
            case UIEventSubtypeRemoteControlTogglePlayPause:
                [self playPauseToggle: nil]
                break;
            case UIEventSubtypeRemoteControlNextTrack:
                [self nextTrack: nil]
                break;
...
```

イベントを表すサブタイプの列挙値については`UIEvent.h`を参照してください。

セカンドディスプレイを最大限に活用する

デフォルトでは、AirPlayビデオ出力デバイスを選択しても、そこにはアプリケーションのビデオがそのまま表示されるだけです。つまり、ホストデバイスの画面と外部ディスプレイの表示内容は同じなのです。

Important アプリケーションがステータスバーの向きを正しく設定するように注意してください。リモートのディスプレイでは、ホストデバイスの加速度計を読み取って向きを変えることができません。アプリケーションが設定した、ステータスバーの向きに応じて決まります。

2つの画面の表示内容が同じで構わないのであれば、アプリケーションは何をする必要もありません。以下、セカンドディスプレイを別個に扱う方法を解説します。

それぞれに異なる内容を表示できれば、画面が2つあることを有効活用できます。たとえば、ホストデバイス側にUIを表示し、外部ディスプレイには映像コンテンツのみを高精細で表示する、という使い方です。2つのディスプレイを独立に扱うためには、`UIView`を使ってセカンドディスプレイの有無を判断した上で、それぞれに何を表示するか指定することになります。

アプリケーション起動時に、`applicationDidFinishLaunching`内で、セカンドディスプレイの有無を確認してください。次のコード例のように、`UIScreen screens`配列に複数のスクリーンが設定されているかどうかを調べます。

```
if ([[UIScreen screens] count] > 1) { ... }
```

さらに、セカンドディスプレイがアクティブになったら通知されるよう登録します。通知の登録をするコード例を以下に示します。

```
[[NSNotificationCenter defaultCenter]
    addObserver:self
        selector:@selector(screenDidConnect:)
            name:UIScreenDidConnectNotification
            object:nil];

- (void) screenDidConnect:(NSNotification *)notification {
    [self myScreenInit:[notification object]];
}
```

この例では、ユーザ定義のメソッド`myScreenInit`が呼び出されます。その際に渡される通知オブジェクトは、新たにアクティブになったスクリーンを表しています（新しいスクリーンオブジェクトは、`UIScreen screens`配列の末尾のオブジェクトとしても取得できます）。

新しいスクリーンオブジェクトを取得したので、そこに描画する準備ができます。スクリーンのboundsを参照して画面の大きさを調べ、これを使って適切な大きさのUIWindowを初期化します。最後に、次のコード例のように、描画対象をこの新しいスクリーンに切り替え、ウインドウのhiddenプロパティをNOとします。

```
- (void) myScreenInit:(UIScreen *)connectedScreen {
    CGRect frame = connectedScreen.bounds;
    UIWindow *window = [[UIWindow alloc] initWithFrame:frame];
    [window setScreen:connectedScreen];
    window.hidden = NO;
}
```

このようにセカンドディスプレイを設定すると、以降の描画処理はすべて、このセカンドディスプレイが対象になります。ホストデバイスのディスプレイに戻って描画したい場合は、window setScreen:でウインドウを内部スクリーンに設定してください。これはUIScreen screens配列の先頭にあるオブジェクトです。このように、2つのスクリーンはいつでも切り替え可能です。描画処理は、OpenGLの処理も含め、こうして切り替えた「現」スクリーンが対象となります。

暗号化と認証

HTTP Live Streamingに適用できるよう、iOSにはメディアの暗号化機能が組み込まれているほか、iOS 5.0以降は認証の機能も加わりました。暗号化キーを保護されたドメインから配布する場合、アプリケーションはサーバとの初期認証ハンドシェイクを処理しなければなりません。

メディアダウンロード時の暗号化

HTTP Live Streamingを使って、「メディアセグメンタ」ソフトウェアに、メディアを暗号化するよう指示することができます。セグメンタは、暗号化キーとその初期化ベクトルを、所定の間隔で生成します。

このキーを使って、所定の期間、ビデオをすべて暗号化します。クライアントソフトウェアはキーを取得し、これを使ってビデオを復号します。初期化ベクトルはキーを使う際に必要となります。

セグメンタに対し、キーを変更する頻度と、現行のキーの初期化ベクトルを変更する頻度を指定する必要があります。また、キーや初期化ベクトルに使うベースURLも指定します。セグメンタは、生成するプレイリストファイルに、キーや初期化ベクトルのURLを記述するようになっています。

キーは通常、それほど頻繁に変更するものではなく、セキュリティ保護のためHTTPS上で受け渡しします。初期化ベクトルは頻繁に変更するのが普通であり、HTTPで高速に受け渡しします。HTTPであってもセキュリティが損なわれることはありません。初期化ベクトル単独では、何の役にも立たないからです。

詳細については、『*HTTP Live Streaming Overview*』を参照してください。

暗号化キーをダウンロードするための認証

暗号化キーを保護するため、サーバはビデオクライアントに対し、自分自身を認証するよう要求します。認証の方法はアプリケーションによって異なります。AirPlayで暗号化されたストリーミングコンテンツを処理できるのは、iOS 5.0以降に限ります。

（iOSの）Safariでウェブベースのビデオを再生する場合、手動でユーザ認証をしてからでないと、保護されたドメインから配布されるキーで暗号化したビデオを再生することはできません。認証処理をJavaScriptで実装することも可能ですが、ログイン情報が漏れる危険がないとは言えません。手動での認証は、ペイパービューやサブスクリプションビデオに対してはうまくいきますが、広告収入で賄われるビデオの場合、閲覧にはログインが必要なため、普通はうまくいきません。

AirPlayを介してビデオを再生する場合、視聴者はテレビ画面で見ているので、パスワードの入力はできないことに注意してください。視聴者は、AirPlay経由でビデオを中継するデバイス側で認証しなければなりませんが、このデバイスは別の部屋にあるかも知れません。この状況ではほかにも、Cookieやその他の認証トークンの有効期間をどう決めるかなど、さまざまな問題があります。以上のような点を考慮してサーバを設定しなければなりません。

アプリケーション上でビデオを再生する場合、認証キーの配布方法として、次の3つをお勧めします。

- **保護されたHTTPS区画からキーを配布。**再生に先立ち、アプリケーションは`NSURLConnection`を用い、秘匿された証明書を使って自分自身を認証できます。
- **HTTPS上でCookieを使用して配布。**アプリケーションはHTTPSサーバに接続し、アプリケーションが定義した方法で自分自身を認証できます。サーバ側では、キーURLに適用するCookieを発行できます。Cookieの有効期間は、再生終了後、相当の時間経過後に無効になるよう設定してください。サーバは、以後、キーを取得する`GET`要求に対して、有効なセッションCookieを提示するよう要求しなければなりません。

  最高度の信頼性を確保するため、有効期間が近々経過するのであれば、以後の`GET`要求に応じ、Cookieの有効期間を更新しなければなりません。
- **「.m3u8」ファイルに、アプリケーションが定義したURLスキームでキーを指定**。この場合、アプリケーションは、独自の`NSURLProtocol`を登録して、当該URLに対する要求を処理することになります。プレーヤーは、以後、キーURLをロードする必要が生じると、アプリケーション側のコードを呼び出します。アプリケーションは安全なサイドチャネルを使ってキーを取得し、プレーヤーに渡します。

詳細については、『*HTTP Live Streaming Overview*』を参照してください。

書類の改訂履歴

この表は「*AirPlayの概要*」の改訂履歴です。

日付	メモ
2011-10-12	ビデオコンテンツ、Webサイト、アプリケーションをAirPlayに対応させる方法について説明した新規文書。

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
U.S.A.

アップルジャパン株式会社
〒163-1450 東京都新宿区西新宿
3 丁目20 番2 号
東京オペラシティタワー
http://www.apple.com/jp/