Video APIs

- Objective
- Contents
    - Fullscreen and embedded players
    - Movie events
    - Hands-on practice
        - Goal
        - Steps
    - Windows development considerations
        - Grant access to video stream and audio stream
        - Grant access to music library
    - References and further reading
- Summary

Objective

In this chapter, you'll learn how you can embed videos within your apps. You'll see how to play both local files and how to stream remote videos.

Contents

You can use the `Titanium.Media.VideoPlayer` to play videos in your Titanium app. This object is returned when you call `Titanium.Media.createVideoPlayer()`. It provide useful methods, such as `play()`, `pause()`, and `stop()`.

You can play local video files by calling the `setMedia()` method or by setting the `media` property. Either accepts a File or Blob object. You can play a remote video by calling the `setUrl()` method or by setting the `url` property. Either accepts the URL of the media to play. (There's also a deprecated contentURL property, but you should use `url` instead.) Finally, you can set the `autoplay` property to `true` to automatically play the video when the VideoPlayer is rendered.

The VideoPlayer is treated like a regular Titanium View. Thus, you can overlay images and views over the VideoPlayer by adding them to the view hierarchy. Like this:

```
var activeMovie = Titanium.Media.createVideoPlayer();
// create a label
var movieLabel = Titanium.UI.createLabel({
 text: 'Do not try this at home',
 height: 35,
 color: 'white',
 font: {fontSize: 24, fontFamily: 'Helvetica Neue'}
});
// add label to view
activeMovie.add(movieLabel);
```

Fullscreen and embedded players

On Android, the VideoPlayer must be used fullscreen. It cannot be embedded into a smaller view. This is because on Android, the VideoPlayer is not technically a proxy for a native view object like it is on iOS. Instead, creating the VideoPlayer fires an Intent which launches the native video player Activity. It's for this reason that on Android, you don't call `win.add(videoPlayer)`. If you did so, your app would throw an error and crash.

On iOS, you can embed the video player within a window or view; it doesn't have to be shown full size. Simply set `height` and `width` properties on the VideoPlayer. To display the player fullscreen, you can set `videoPlayer.fullscreen=true`.

You can control the way in which the video playback controls are shown for the player. You can "embed" the within the player, which removes them from the player's surrounding chrome giving more space to your video. You do so by setting the `movieControlStyle` property, like this

```
var activeMovie = Titanium.Media.createVideoPlayer({
 url: '../movie.mp4',
 movieControlStyle: Titanium.Media.VIDEO_CONTROL_EMBEDDED
});
```

Movie events

The VideoPlayer supports quite a few events that you can use to control the playback experience. See the `Ti.Media.VideoPlayer` API docs page for full information. But here are a few of the events you might monitor:

- `complete` – fired when the playback ends or the user exits playback, use the `e.reason` property to determine the actual stop condition, such as `Ti.Media.VIDEO_FINISH_REASON_PLAYBACK_ENDED`
- `load` – fired when the movie finishes loading
- `fullscreen` – fired when the movie changes to or from fullscreen, use the `e.entering` property to determine if the player is entering or leaving fullscreen mode

A technique you should consider is stopping the movie when the video it's embedded in closes. You can do that simply with code like this:

```
win.addEventListener('close', function() {
 activeMovie.stop();
});
```

Hands-on practice

Goal

In this activity, you will write a simple app to stream a movie from a remote URL.

Steps

1. Create a new Titanium Mobile app. Delete all of the code from the app.js file.
2. Define a window with a black background. Optionally, set the orientation of the window to landscape.
3. Define a VideoPlayer with these characteristics:
    - backgroundColor = black
    - default video controls (not embedded)
    - scalingMode = fill the screen
    - fullscreen
    - Play the http://assets.appcelerator.com.s3.amazonaws.com/video/media.m4v movie. Depending on your Android environment, that movie might not play well (due to encoding issues from the creation process). If you have playback troubles, you can play http://assets.appcelerator.com.s3.amazonaws.com/video/media.3gp on Android instead.
4. Add the branching logic so that on iOS you add the VideoPlayer to the window object.
5. Add the code to play the video stream and open the window.
6. Build and test your app in the simulator/emulator or on device.

Simulator / emulator rotation
Rotate the iPhone simulator by pressing Command and either the left or right arrow key. Rotate the Android emulator by pressing Control+F12.

Windows development considerations

Support for Windows 8.1 and Windows Phone SDKs has been deprecated as of SDK 6.3.0.GA and has be removed in SDK 7.0.0.GA.

In order to enable camera and audio recording for Windows Phone, you need to provide appropriate Capabilities in your `tiapp.xml`. Windows Phone users are prompted to grant or deny permission when your application attempt to use it.

Grant access to video stream and audio stream

```
<ti:app>
  ...
  <windows>
    ...
    <Capabilities>
        <DeviceCapability Name="microphone" />
        <DeviceCapability Name="webcam" />
    </Capabilities>
    ...
  </windows>
  ...
</ti:app>
```

Grant access to music library

```
<ti:app>
  ...
  <windows>
    ...
    <Capabilities>
        <Capability Name="musicLibrary" />
    </Capabilities>
    ...
  </windows>
  ...
</ti:app>
```

For more information about audio configuration in `tiapp.xml`, see Windows-specific section in tiapp.xml and timodule.xml Reference.

References and further reading

- Finished code
- API docs: Ti.Media.VideoPlayer
- iOS Developer Docs: Supported video formats
- Android Developer Docs: Supported media formats
- Windows 10 Mobile: Audio, video, and camera

Summary

In this chapter, you learned how you can embed videos within your apps. You saw how to play both local files and how to stream remote videos.