Flex Video
Tips & Tricks

Flex Video: About Me

About Me

- **Adobe Certified Master Instructor**
- **Adobe User Group Manager**
- **Over 10 years design/dev experience**
  - Post Production Video Compositing/Editing
  - Motion Graphics
  - Interactive Design etc.
- **Owner & Principal of RealEyes Media**
- **Taught/developed many successful advanced Flash/Flex application courses**

Flex Video: Overview

- **Capabilities**
  - Basics, Codecs etc.
- **Creating Video**
  - Rules, tools & settings
- **Integration**
  - Time for some code
- **Interactivity**
  - Play with your video

Codecs etc…

CAPABILITIES

Flex Video: Capabilities

Why Flash?

- 98% of Internet users in the US/Canada already have Flash Player 7 or higher.
- Flash Player 7 introduced expanded support for video. Flash Player 9 has even more.
- It allows easy integration with Flex and AIR applications.
- Flash is cross platform, even to many mobile devices.
- 70%+ of video streaming providers use Flash video

Flex Video: Capabilities

Why Flash Video?

- Ubiquity
- Expressiveness
- Customization/Control
- Quality
- Advanced Runtime Visual Effects & Capabilities
- Streaming capability
- Ease of implementation
- Already the standard for video delivery by sites like YouTube.

Flex Video: Capabilities

- **Codecs Overview: Sorenson**
  - Developed by Sorenson Media and licensed to Macromedia in 2002 for release with Flash MX.
  - The current version is version 3, known as Sorenson Spark, and it can be used to create FLV files. Spark has Basic and Pro versions. Pro allows for higher image quality than the Basic version.
  - Required for Flash 6 or 7 video deployment.
  - Better for large frame sizes or video with high bit rates (over 1 Mbps), because it is less processor intensive.
  - Good for targeting a wide range of devices, with varying processor power.

Flex Video: Capabilities

- **Codecs Overview: On2 VP6**
  - Developed by On2 Technologies and adopted by Macromedia for video in Flash Player 8.
  - Produces high image quality, but is processor intensive. Comes in the normal and Simple flavors. Simple is basically uncompressed and can be HD, but is supported only in the latest Flash Player release.
  - Good for target audiences with modern processors (Pentium IV and up) and Flash Player 8 and higher.
  - Good for video smaller than 640 x 480.
  - Supports alpha channels.

Flex Video: Capabilities

- **Codecs Overview: H.264 & AAC/AAC+**
  - Developed by the MPEG and VCEG group as part of MPEG-4 and it includes high definition capability. It is used a lot in HD DVD/Blu Ray DVD.
  - Available in the most recent release of Flash Player 9 (Moviestar). Can only be streamed with FMS 3.
  - Supports true HD video, multi-core support, many devices, and lots of metadata.
  - Can be processor intensive, but hardware acceleration and in full screen mode allows for smooth playback of large video.
  - Note: moov atom issue

Rules, tools and settings

CREATING VIDEO

Flex Video: Creating Video

- **Creating (compression)**
  - The rules
  - The tools
  - The settings

Kevin Towes -- http://www.adobe.com/devnet/flash/articles/flv_encoding.html

Creating Video: Rules

General Rules

(frame height x frame width x frame rate (fps)) / Compression = total bits/sec.

- **As the target bandwidth (data rate) decreases, reduce the keyframe rate.**
- **As motion increases, you must increase the keyframe rate, the frame rate, and the data rate.**
- **Dial-up connections consume almost all of the available bandwidth**

Creating Video: Rules

Rules Cont.

- **Frame rates are calculated at half-rate, quarter-rate, and third-rate**
- **Reduce frame size when bandwidth is limited and frame rate and quality are important.**
- **Always enable de-interlacing and set the Flash Video encoder to the upper field.**
- **Extra video noise in lower quality video requires additional data rate.**

Creating Video: Tools

- **Tools**
  - RIVA (free)
  - FFMpeg (free)
  - On2Flix
  - Sorenson Squeeze
  - Creative Suite & Adobe Media Encoder

Creating Video: Settings

Encoding FLV Sorenson

Target Connection Speed	Video Bitrate	Width	Height	FPS	Keyframe Interval	Audio Bitrate
High Motion	High motion; lots of zooms, fades and people moving around					
1.5 Mbps	750 Kbps	320	240	29.97	60	96 Kbps
768 kbps	575 Kbps	320	240	29.97	60	64 Kbps
384 Kbps	329 Kbps	320	240	14.98	30	32 Kbps
56k Dial Up	40 Kbps	192	144	9.99	20	8 Kbps
Low Motion	Low motion; general talking heads					
1.5 Mbps	650 Kbps	320	240	29.97	60	96 Kbps
768 kbps	230 Kbps	320	240	14.98	30	64 Kbps
384 Kbps	153 Kbps	320	240	9.99	20	32 Kbps
56k Dial Up	40 Kbps	192	144	9.99	20	8 Kbps

Creating Video: Settings

Encoding FLV On2 (High Motion)

High-Motion Video	Total Bit Rate	Video Bit Rate	Audio Bit Rate	Frame Size	Frame Rate Video/Film Source	Keyframe Interval*
Modem – Streaming**	40K	24K	16K mono	160 x 120	7.5/8 fps	8 sec.
Modem – Progressive	80K	64K	16K mono	160 x 120	7.5/8 fps	8 sec.
Small	220K	188K	32K mono	240 x 180	15/12 fps	8 sec.
Medium	400K	336K	64K mono	320 x 240	30/24 fps	8 sec.
Large	850K	754K	96K stereo	480 x 360	30/24 fps	8 sec.
X-Large	1400K	1304K	96K stereo	640 x 480	30/24 fps	8 sec.

Creating Video: Settings

Encoding FLV On2 (Low Motion)

Low-Motion Video	Total Bit Rate	Video Bit Rate	Audio Bit Rate*	Frame Size	Frame Rate Video/Film Source	Keyframe Interval
Modem – Streaming	40K	24K	16K mono	160 x 120	7.5/8 fps	8 sec.
Modem – Progressive	50K	34K	16K mono	160 x 120	7.5/8 fps	8 sec.
Small	100K	68K	32K mono	240 x 180	15/12 fps	8 sec.
Medium	180K	132K	48K mono	320 x 240	30/24 fps	8 sec.
Large	350K	286K	64K mono	480 x 360	30/24 fps	8 sec.
X-Large	600K	504K	96K stereo	640 x 480	30/24 fps	8 sec.

Time for some code

INTEGRATION

Flex Video: Integration

- **Integration**
  - Component
  - Custom code
    - Connection > Stream: Video & Audio
    - Streaming H.264/AAC+ from FMS
  - Meta-data
  - Full-screen & hardware acceleration
  - Playback Media Player Style
  - Seamless display
    - Alpha
    - Layer Transform/Filter
    - Compositing

Integration: Flex Video Display

Component: Flex Video Display

- **No playback component for Flex, just the Video Display.**
- **Lots of methods and events exposed for control and cue point management.**
- **Use the VideoDisplay tag, specify a source, set dimensions to start.**
- **For deeper use, create a listener for video events to control playback.**

Integration: Flex Video Display

- **Using the existing/updated Flex 2.0 video display component**
- **The Flex 3.0 VideoDisplay component will play H.264 (.mov and .mp4 files, etc.) without any modifications.**
- **Using ActionScript 3.0 in Flex, you can create a custom class extending the `UIComponent` class that creates a Video object and uses the `NetConnection` and `NetStream` classes to play the video through the Video display object.**

```
<mx:VideoDisplay  id="reVid"
   width="320" height="240"
   autoBandWidthDetection="false|true"
   autoPlay="true|false"
   autoRewind="true|false"
   bufferTime="0.1"
   cuePointManagerClass=""
   cuePoints=""
   idleTimeout="300000"
   live="false|true"
   maintainAspectRatio="true|false"
   playheadTime=""
   playheadUpdateInterval="250"
   progressInterval="250"
   source="video/realeyes.flv"
   totalTime=""
   volume="0.75"/>
```

Integration: ActionScript 3

[Flex Code Sample]

Integration: ActionScript 3

ActionScript 3

- **NetStream object has pause and resume methods rather than just pause.**
- **Flash will throw an error if you don't have a handler for the AsyncError event.**
- **Have to use Event Listeners for your click events for controls.**

Integration: Async Error Handler

```
var video:Video = new Video(); //create the video display obj

var connect_nc:NetConnection = new NetConnection(); //create the
   net conn for external media
connect_nc.connect(null); //conn to null for progressive
   download

connect_nc.addEventListener(AsyncErrorEvent.ASYNC_ERROR, errorHandler);

var stream_ns:NetStream = new NetStream(connect_nc);//create a
   NetStream and hook it into the conn

var _client:Object  = new Object();    //create a generic client
   object to receive all NetStream events
```

Integration: Async Error Handler

```
stream_ns.client = this; //associate the client object with the
   NetStream. This handles asynchronous errors too.

function onMetaData(){…}; //have the client handle metadata
   events

video.attachNetStream(stream_ns); //attach the stream to the
   video display so we can see it

addChild(video); //attach the video display to the stage so we
   really see it

stream_ns.play("fileToPlay.mp4");//specify which file to play
```

Integration: ActionScript 3

[ActionScript Code Sample]

Integration: Streaming H.264/AAC+

- **Using FMS with Flex for Video Streaming**
  - VideoDisplay Component streaming features
    - Auto bandwidth detection capable
    - Easy capable usage for FMS video streaming
  - Custom NetConnection Code for Streaming
    - H.264
    - Playlisting
    - Bandwidth Detection
    - Auto-Bandwidth Switching

Integration: Streaming H.264/AAC+

- **Streaming H.264**
  - Requires a streaming media server
  - Can stream or seek to any point non-linear
  - Quicker playback and much more secure
  - Flash Media Server will be the ONLY server to support H.264 and AAC+ streaming to Flash for now
  - Must use 'mp4:' prefix for stream play commands
  - Position of moov atom doesn't matter

```
stream_ns.play( "mp4:fileToPlay.mp4" );
```

Integration: Streaming H.264/AAC+

- **Coding process stays similar as existing streaming.**
- **True streaming video does not have to have the moov atom at the beginning.**
- **FMS 3 supports encrypted streaming and enhanced seeking for MPEG-4 content, as well as all features currently supported for streaming FLV content.**
- **Capable of streaming HD quality video as well as live recording of HD quality video into H.264 format (via FME)!! *(Now that's pretty slick)***

Integration: Streaming H.264/AAC+

H.264 Meta data

- **Using the onMetaData event of the NetStream object, you can extract information such as (depending on the encoder and codec):**
  - Dimensions
  - Length
  - Codec
  - seek points
- **Other data that could potentially be stored in the meta data are:**
  - cover art
  - subtitles
  - Audio book chapters
- ***If seek points aren't included in the meta data, you will be unable to seek in the file at all.***
- ***Cue points currently are not supported.***

Integration: Streaming H.264/AAC+

[metadata sample]

Integration: Using Full Screen

- **Full screen mode with hardware scaling**
- **The visual rendering process moves to the graphical processing unit (GPU).**
- **Hardware acceleration can be disabled in the new Display tab of the Flash Settings dialog box.**
- **API to specify a rectangular portion of the stage to be scaled:**
  - Fewer pixels that need to be drawn to the screen
  - Better user experience
  - Lower CPU utilization for full-screen video playback.

Integration: Using Full Screen

```
private function goFullScreen(p_evt:Object):void
{
   try
   {
        var scalingRect:Rectangle = new Rectangle(vidPlayer.x,
                      vidPlayer.y, vidPlayer.width,
                      vidPlayer.height);
        stage["fullScreenSourceRect"] = scalingRect;

        if(stage.displayState == NORMAL)
        {
               stage.displayState = FULL_SCREEN;
        }

   //Continued on next slide
```

Integration: Using Full Screen

```
        else
        {
                stage.displayState = NORMAL;
        }

    }
    catch(e:Error)
    {
         Alert.show("Error Code: " + e.message, "FULL SCREEN
                ERROR");
    }
}
```

Integration: Using Full Screen

- **[DEMO – Full Screen and hardware acceleration]**

Integration: Custom Playback Controls

- **Custom Playback Controls**
  - For play/pause controls, use the NetStream's **play** and **pause** methods. For toggling play controls, you can check the class' **togglePause** method.
  - The **close** method of the NetStream can act as stop.
  - For fast forward and rewind, use the **seek** method.
  - There is no playheadUpdate event equivalent for NetStream, so for scrub bars, you'll need to monitor playhead position (**time** property) through an enterFrame event or timer.
  - Use the **seek** method for navigating with scrub bars.
  - The **HSlider** component makes a handy scrub bar.

Integration: Custom Playback Controls

```
addEventListener( Event.ENTER_FRAME, _onEnterFrame, false, 0, true );
private function _onEnterFrame( evt:Event ):void
{
    position = _stream_ns.time;
}
public function get position():Number
{
    return _position;
}
public function set position(p_value:Number):void
{
    _position = p_value;
    _scrubBar.value = _position;
    _stream_ns.seek( _position );
}
```

Integration: Streaming H.264/AAC+

- **Seamless display**
  - Alpha Video
    You can have an 8-bit alpha channel in videos encoded with the On2 VP6 codec
  - SAMPLE: Combining H.264 & VP6
    - [Alpha VP6 video overlay on H.264]

Play with your video

INTERACTIVITY

Flex Video: Interactivity

- **Interactivity**
  - Controls
    - REFlexVid Player
  - Cue points
    - Overview
    - Internal/AMF
    - XML
  - Playlisting
    - Avatars
      - Guided usage

Interactivity: REFlexVid Player

REFlexVid Player

[In progress…will be part of REDBug eventually]

Interactivity: REFlexVid Player

Cue Points

- Cue points
  - Overview
  - Internal/AMF
  - XML

Interactivity: Cue Points

- **There are three types of cue points:**
  - **Navigation** - allows you to seek a particular frame in the FLV file by creating a keyframe in the FLV file as near as possible to the specified time.
  - **Event** - enables you to synchronize a point in time within the FLV file with an external event.
  - **ActionScript** - Used with the Flash's FLVPlayback component
- **The navigation and event cue points are also known as *embedded* cue points because they are embedded in the FLV file stream and in the FLV file's metadata packet.**

Interactivity: Cue Points

- **To handle the onCuePoint() call. The simplest solution is to set the `NetStream` client property equal to the `this` scope.**
- **Then the NetStream will look in the current scope for the onMetaData() and onCuePoint() methods.**

```
var nc:NetConnection = new NetConnection();
nc.connect(null);
var ns:NetStream = new NetStream(nc);
ns.client = this;
…
function onCuePoint(infoObject:Object):void
{
     trace("cue point");
}
```

Interactivity: REFlexVid Player

Playlisting

- Avatars
  - Guided usage
  - XML Samples for the play list (progressive & w/ start/end)

Interactivity: REFlexVid Player

Progressive sample

```
<data>
  <playlist>
      <video name="vid1.flv" title="Video 1" />
      <video name="vid2.flv" title="Video 2" />
      <video name="vid3.flv" title="Video 3" />
      <video name="vid4.flv" title="Video 4" />
      <video name="vid5.flv" title="Video 5" />
  </playlist>
</data>
```

Interactivity: REFlexVid Player

Streaming sample

```
<data>
  <playlist>
      <video name="vid1.flv" title="Video 1">
            <clip start="20" end="45" />
            <clip start="60" end="65" />
            <clip start="100" end="150" />
      </video>
      <video name="vid2.flv" title="Video 2">
            <clip start="5" end="14" />
            <clip start="32" end="110" />
      </video>
  </playlist>
</data>
```

Interactivity: Cue Points

- **TOP 10 TIPS - kinda:**
  1. Compression: Quality in, quality out
  2. Use the right codec for the job
  3. Be sure to handle your (asynch) events
  4. Video instances can NOT be added to containers – use UIObject holders for light weight
  5. Don't always try and build your application around the video. Use the creative side of the technology to integrate the video around your application.

Interactivity: Cue Points

- **TOP 10 TIPS (continued):**
  6. Code reusability! Make/take a player and build upon it.
  7. Fully explore all the classes associated with video and audio to see all the methods and properties available.
  8. Write your mother. She loves you.
  9. Meta data is your friend, respect it, talk to it, use it – but in a nice way.
  10. Know when you need to make a custom component and when a VideoDisplay component is more appropriate.

Flex Video: Interactivity

- **Flash Video Dev Net - http://www.adobe.com/devnet/flash/video.html**
- **Flash Media Encoder - http://www.adobe.com/products/flashmediaserver/flashmediaencoder**
- **Timed Text XML Format - http://www.w3.org/AudioVideo/TT/**
- **FLV Meta Data Injector http://www.asvguy.com/2005/06/flv_metadata_in.html**

Resources

- **Adobe Labs: http://labs.adobe.com/**
- **Moviestar Update: http://labs.adobe.com/technologies/flashplayer9/**
- **Tinic Uro's Blog: http://www.kaourantin.net/**
- **H.264 on WikiPedia: http://en.wikipedia.org/wiki/H264**
- **AAC on WikiPedia: http://en.wikipedia.org/wiki/Advanced_Audio_Coding**
- **MPEG Industry Forum: www.mpegif.org**
- **MPEG LA: www.mpegla.com**
- **VIA Licensing: www.vialicensing.com**
- **Codec Comparisons:**
  - http://www.streamingmedia.com/article.asp?id=9659&page=1&c=8
  - http://www.compression.ru/video/codec_comparison/subjective_codecs_comparison_en.html.

Resources

- **Moov atom helpers:**
  - QTIndexSwapper (AIR) - http://renaun.com/blog/2007/08/22/234/
  - Qt-faststart (C) - http://svn.mplayerhq.hu/ffmpeg/trunk/tools/qt-faststart.c?revision=9634&view=markup
- **Cool AIR app**
  - RICH FLV http://www.richapps.de/?p=82

Resources

- **Encoders:**
  - AnyStream - http://www.anystream.com/
  - Digital Rapids - http://www.digital-rapids.com/
  - Kulabyte - http://www.kulabyte.com/
  - On2 - http://www.on2.com/
  - Rhozet - http://www.rhozet.com/
  - Sorenson - http://www.sorensonmedia.com/
  - TeleStream - http://www.telestream.net/
  - X264 Encoder - http://www.videolan.org/developers/x264.html

Resources

- **David Hassoun**
  david@realeyes.com
- **Blog**
  http://david.realeyes.com/
- **Adobe Devnet Article**
  http://www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html
- **Presentation & Additional Resources**
  http://labs.realeyesmedia.com