Working with Media in AV Foundation

Overview and best practices

Session 415

Sam Bushell
Media Frameworks Architect

What You Will Learn

- Why and when you should use AV Foundation editing
- Concepts underlying manipulation of timed-based media
- What editing tasks you can accomplish using AV Foundation

Sample Code for This Session

- AVEditDemoIPad
- Materials available at:
  https://developer.apple.com/wwdc/schedule/details.php?id=415

Technology Framework

MediaPlayer

AppKit

AVFoundation

CoreAudio

CoreMedia

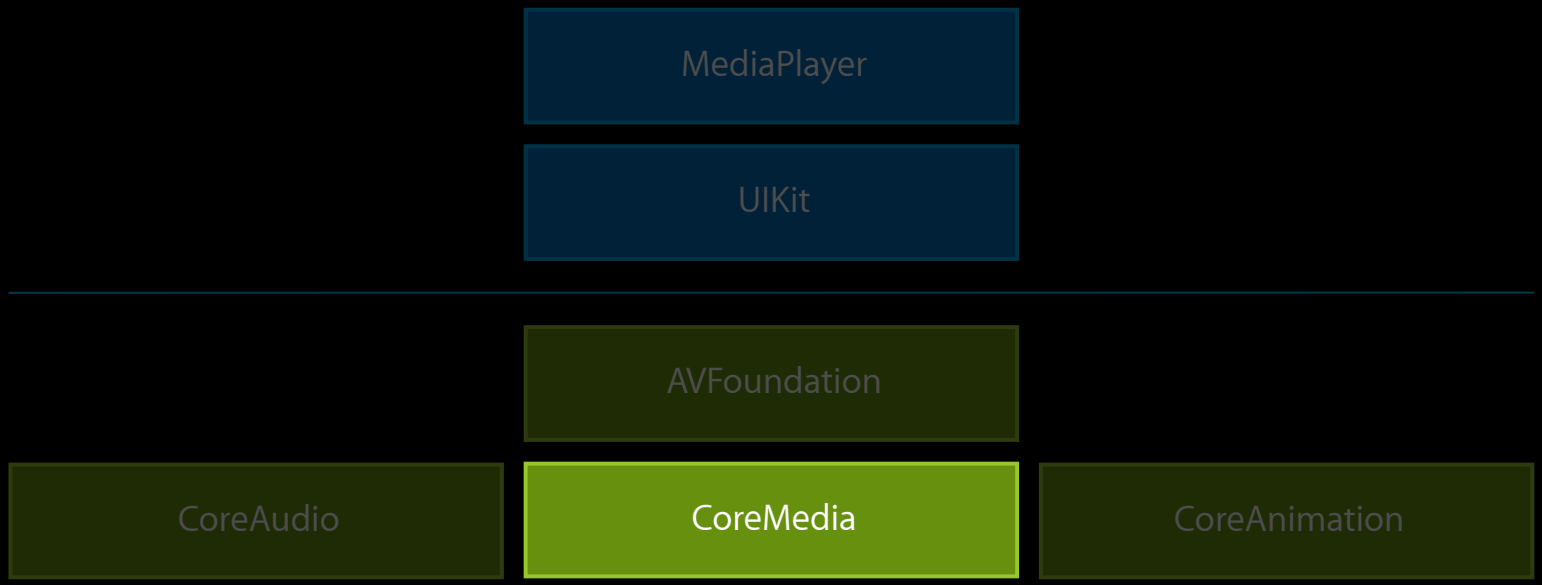
CoreAnimation

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Fundamentals



CMTime

Struct type for rational time

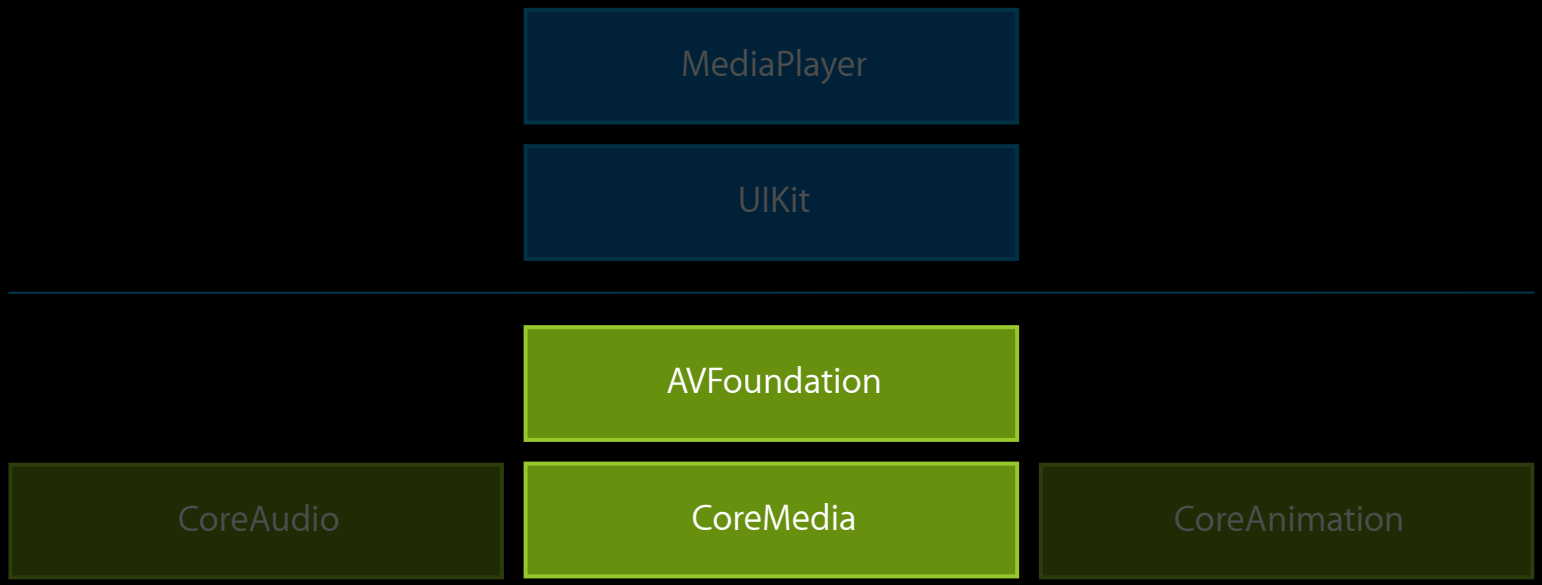
- CMTime t = CMTimeMake(time value, time scale);
- kCMTimeZero, kCMTimeInvalid

- x = CMTimeAdd(y, z);
- if(CMTIME_COMPARE_INLINE(t1, <=, t2)) { ... }

- CMTimeRange r = CMTimeRangeMake(start time, duration);

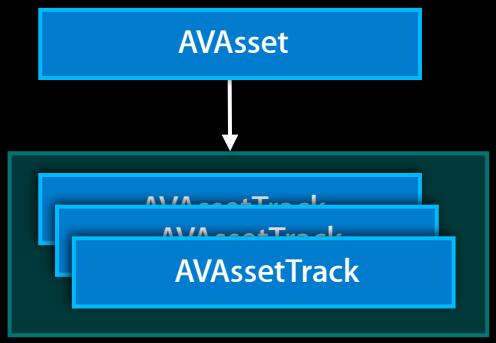
- CMTimeMapping m = CMTimeMappingMake(source range, target range);

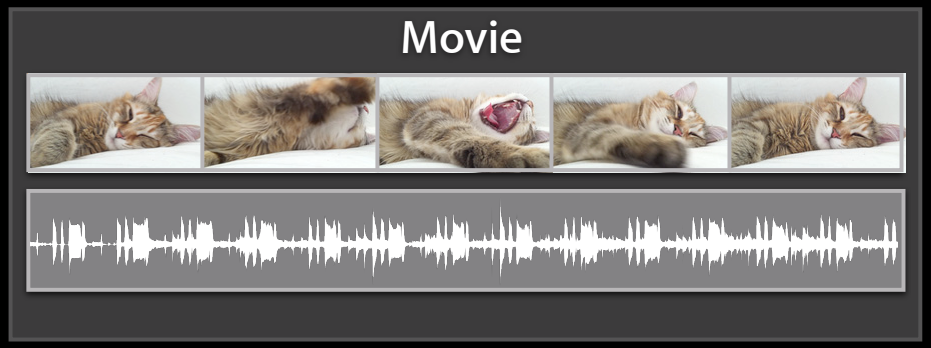
Moving Up


MediaPlayer
UIKit
AVFoundation
CoreAudio
CoreMedia
CoreAnimation

Movies and AVAssets

- AVURLAsset represents movies in files
- An AVAssetTrack represents a particular track inside the movie
- Use AVPlayerItem to play an AVAsset





Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Grab Image at Time

Grab images from an AVAsset using AVAssetImageGenerator

```
AVAssetImageGenerator *imageGenerator =
    [AVAssetImageGenerator
        assetImageGeneratorWithAsset:myAsset];
```

```
[imageGenerator generateCGImagesAsynchronouslyForTimes:timeArray
    completionHandler:handlerBlock];

// need to retain imageGenerator until you get the images
```

Image Generation Completion Block

Check the result

```
AVAssetImageGeneratorCompletionHandler handlerBlock =
  ^(CMTime requestedTime, CGImageRef image, CMTime actualTime,
    AVAssetImageGeneratorResult result, NSError *error){
  switch (result) {
    case AVAssetImageGeneratorSucceeded:
      /* image is valid */
      break;
    case AVAssetImageGeneratorFailed:
      /* error */
      break;
    case AVAssetImageGeneratorCancelled:
      /* cancelled */
      break;
  }
}
```

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Export and Trimming

Export an AVAsset to a new file using AVAssetExportSession

- Presets for different sizes, bitrates, etc.
- Optionally set timeRange to trim
- Optionally add metadata

```
AVAssetExportSession *exportSession = [[AVAssetExportSession alloc]
    initWithAsset:asset presetName:AVAssetExportPresetMediumQuality];
```

```
exportSession.outputURL = ...;
exportSession.outputFileType = AVFileTypeQuickTimeMovie;
```

```
exportSession.timeRange = CMTimeRangeMake(startTime, duration);
```

```
exportSession.metadata = ...;
```

```
[exportSession exportAsynchronouslyWithCompletionHandler:handlerBlock];
```

Export Completion Block

Check the status

```
void (^handlerBlock)(void) = ^{
  switch (exportSession.status) {
    case AVAssetExportSessionStatusCompleted:
      /* export complete */
      break;
    case AVAssetExportSessionStatusFailed:
      /* export error (see exportSession.error) */
      break;
    case AVAssetExportSessionStatusCancelled:
      /* export cancelled */
      break;
  }
}
```

A Word on Error Handling

Handle failures gracefully

- AVAssetExportSession will not overwrite files
- AVAssetExportSession will not write files outside of your sandbox

Export and Multitasking

iOS

Handle failures gracefully

- Other apps that start playback will interrupt a background export
- Even in the foreground, an incoming phone call will interrupt export

Editing APIs in AV Foundation

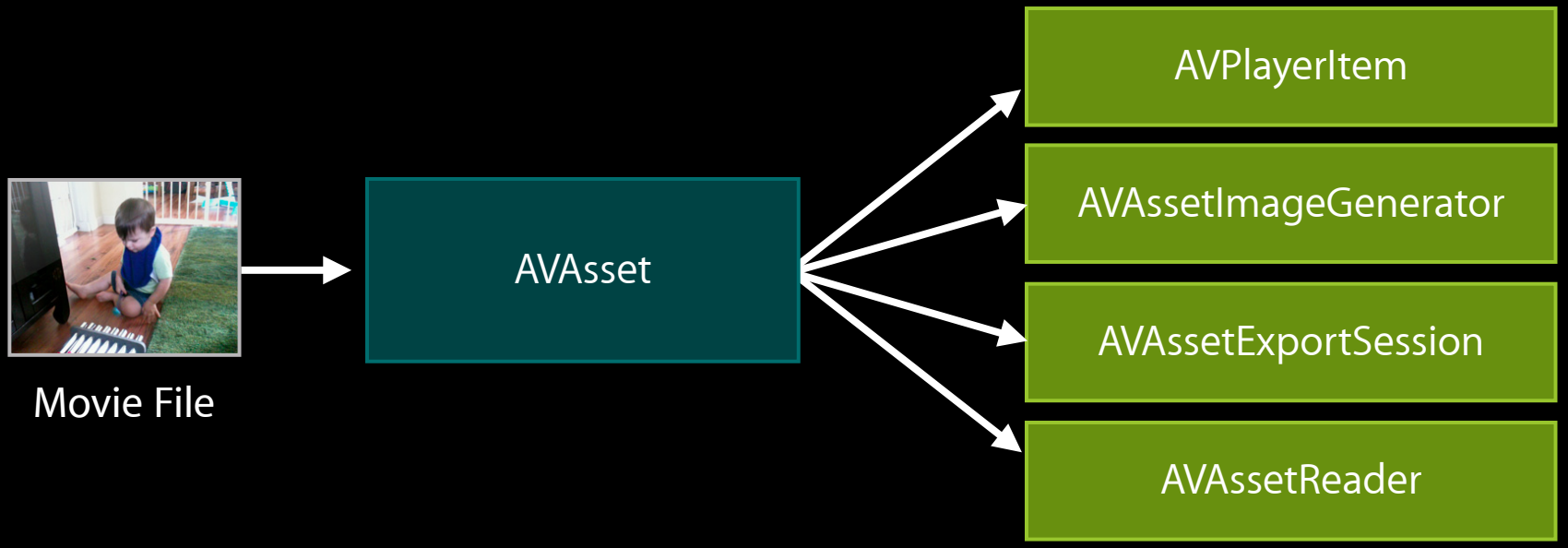
Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

AVAsset and AVComposition

Cornerstones of editing

AVAsset as a Source Object



AVAsset as a Source Object

Movie Files

AVComposition

a subclass of AVAsset

AVPlayerItem

AVAssetImageGenerator

AVAssetExportSession

AVAssetReader

Editing APIs in AV Foundation

Scenarios

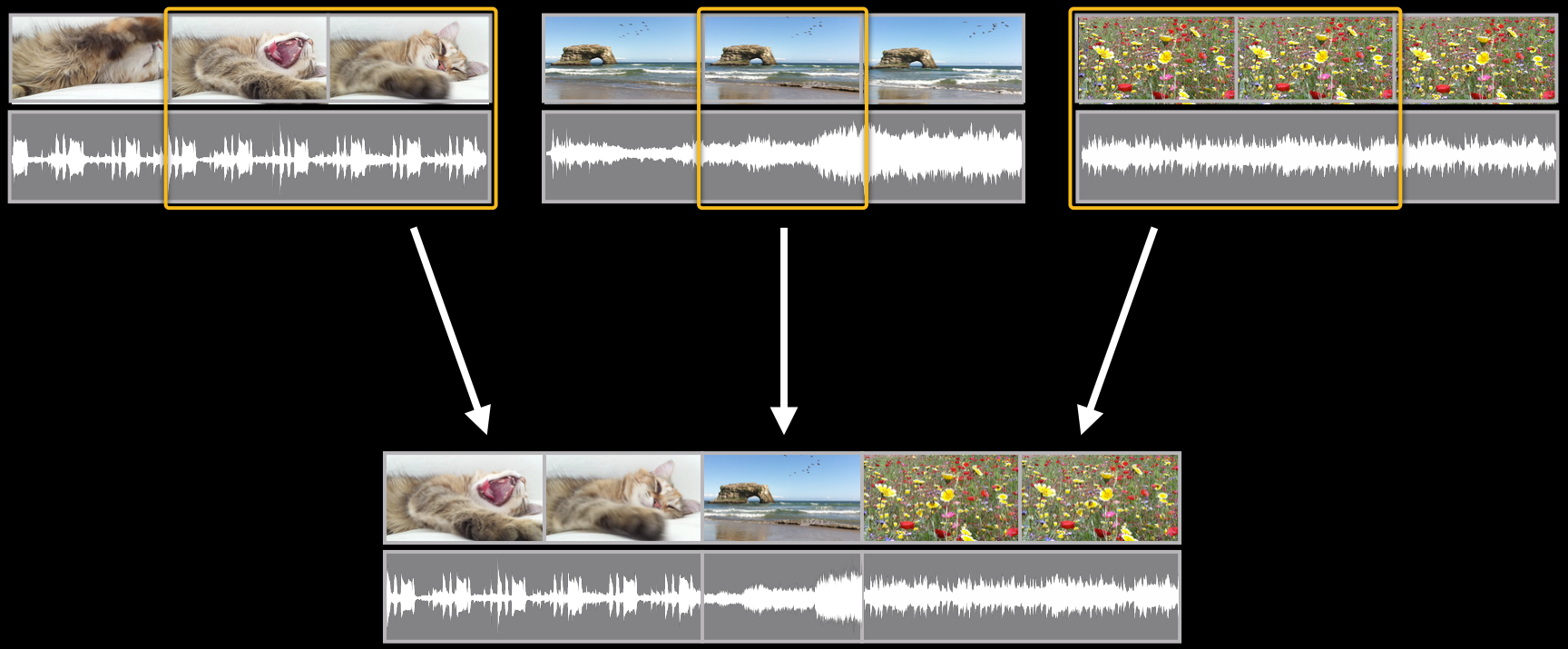
- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Demo

Cutting together movie clips

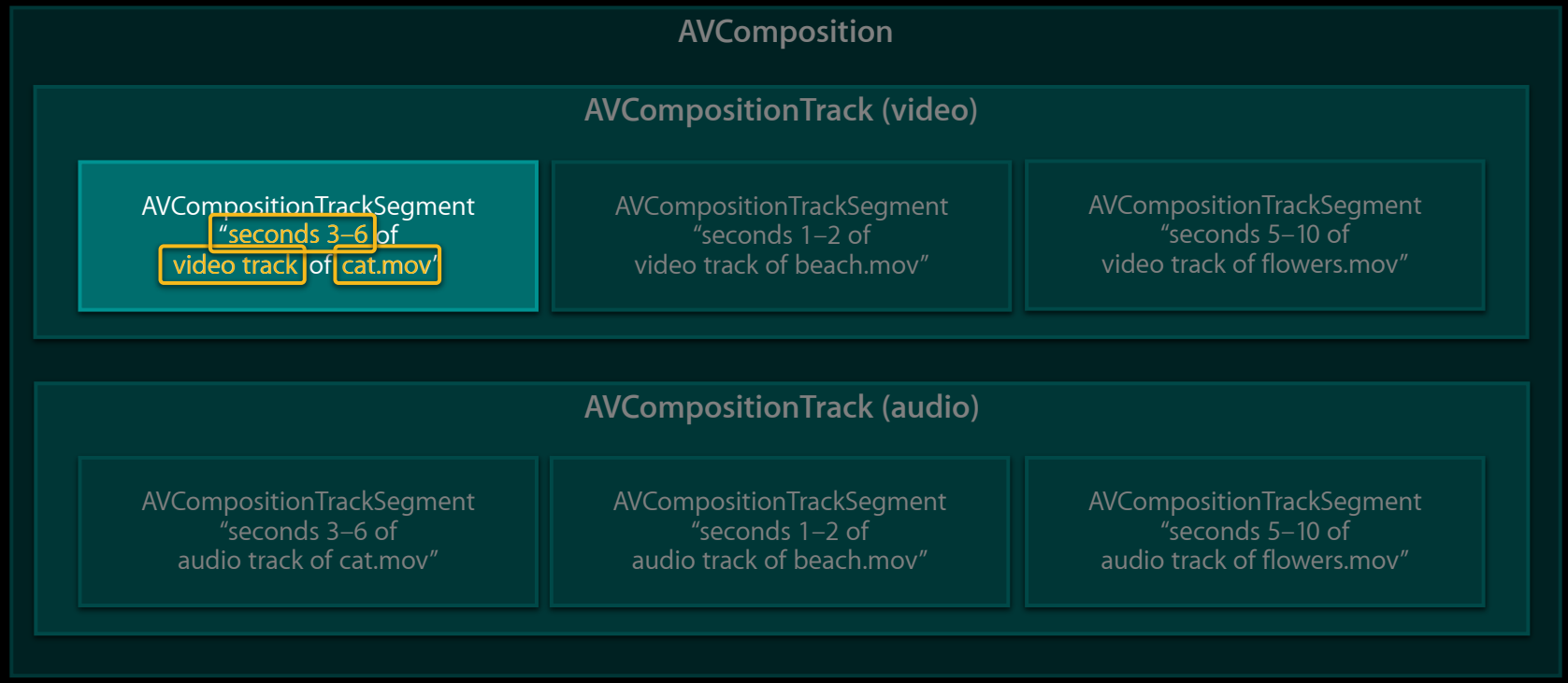
Sample code: AVEditDemoIPad (see SimpleEditor.m)

Composing a Timeline



AVComposition

AVComposition assembles asset segments on a timeline



AVMutableComposition

- You can edit across all tracks of a composition:
  - [composition insertTimeRange:... ofAsset:... atTime:... error:...];
- You can edit a single track:
  - [compositionTrack insertTimeRange:... ofTrack:... atTime:... error:...];
  - [compositionTrack insertTimeRanges:... ofTracks:... atTime:... error:...];

- You can change the segment array directly:
  - [compositionTrack setSegments:...];

AVEditDemo

See buildSequenceComposition in SimpleEditor.m

```
AVMutableComposition *composition = [AVMutableComposition composition];
```

```
AVMutableCompositionTrack *compositionVideoTrack =
    [composition addMutableTrackWithMediaType:AVMediaTypeVideo
        preferredTrackID:...];
```

```
[compositionVideoTrack insertTimeRange:...
    ofTrack:clipVideoTrack atTime:... error:...];
```

When Not to Mutate Compositions

- Do not modify an AVMutableComposition during playback, image generation, export or reading

- Make a copy for these tasks; then it is safe to modify the original:

```
AVPlayerItem *playerItem =
    [AVPlayerItem playerItemWithAsset:
      [[mutableComposition copy] autorelease]];
```

- Switch player to new item:

```
[player replaceCurrentItemWithPlayerItem:playerItem];
```

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

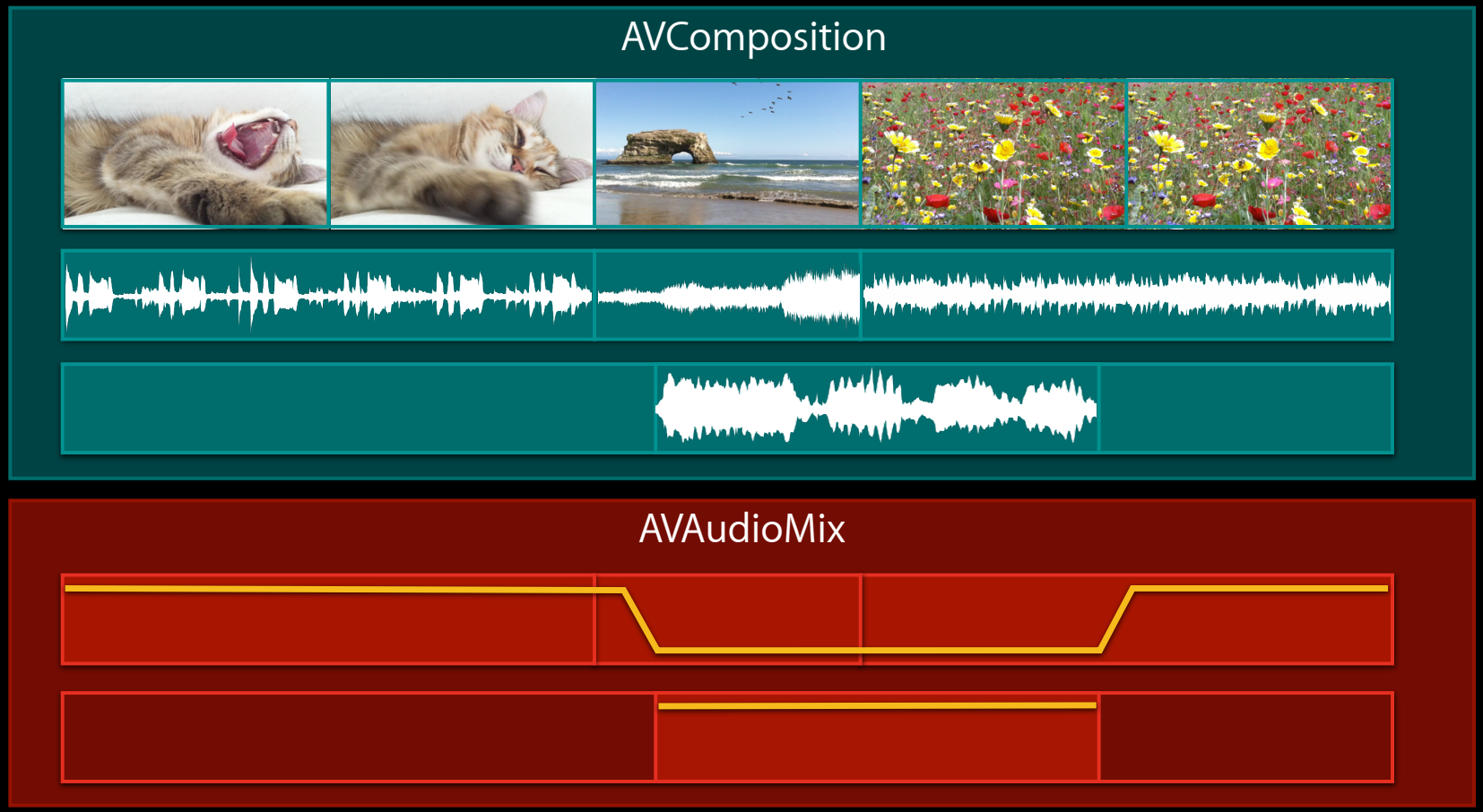
Demo

Mixing in additional audio

Sample code: AVEditDemoIPad (see SimpleEditor.m)

Adding a Commentary Track

AVComposition and AVAudioMix



AVAudioMix

Tool for adding volume adjustments

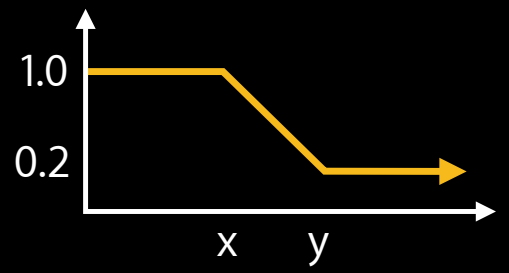
- Has array of AVAudioMixInputParameters
  - Each adjusts the volume level of one track
  - Tracks without AVAudioMixInputParameters get default volume

AVMutableAudioMix

```
AVMutableAudioMixInputParameters *trackMix =
    [AVMutableAudioMixInputParameters
        audioMixInputParametersWithTrack:mainAudioTrack];
```

```
[trackMix setVolume:1.0 atTime:kCMTimeZero];

[trackMix setVolumeRampFromStartVolume:1.0
    toEndVolume:0.2
    timeRange:CMTimeRangeMake(x,y-x)];
...
```




```
AVMutableAudioMix *audioMix = [AVMutableAudioMix audioMix];

audioMix.inputParameters = [NSArray arrayWithObject:trackMix];
```

Using AVAudioMix

- To apply AVAudioMix for playback:
  - playerItem.audioMix = audioMix;
- To apply AVAudioMix for export:
  - exportSession.audioMix = audioMix;
- To apply AVAudioMix for sample reading:
  - assetReaderAudioMixOutput.audioMix = audioMix;

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Editing APIs in AV Foundation

Scenarios

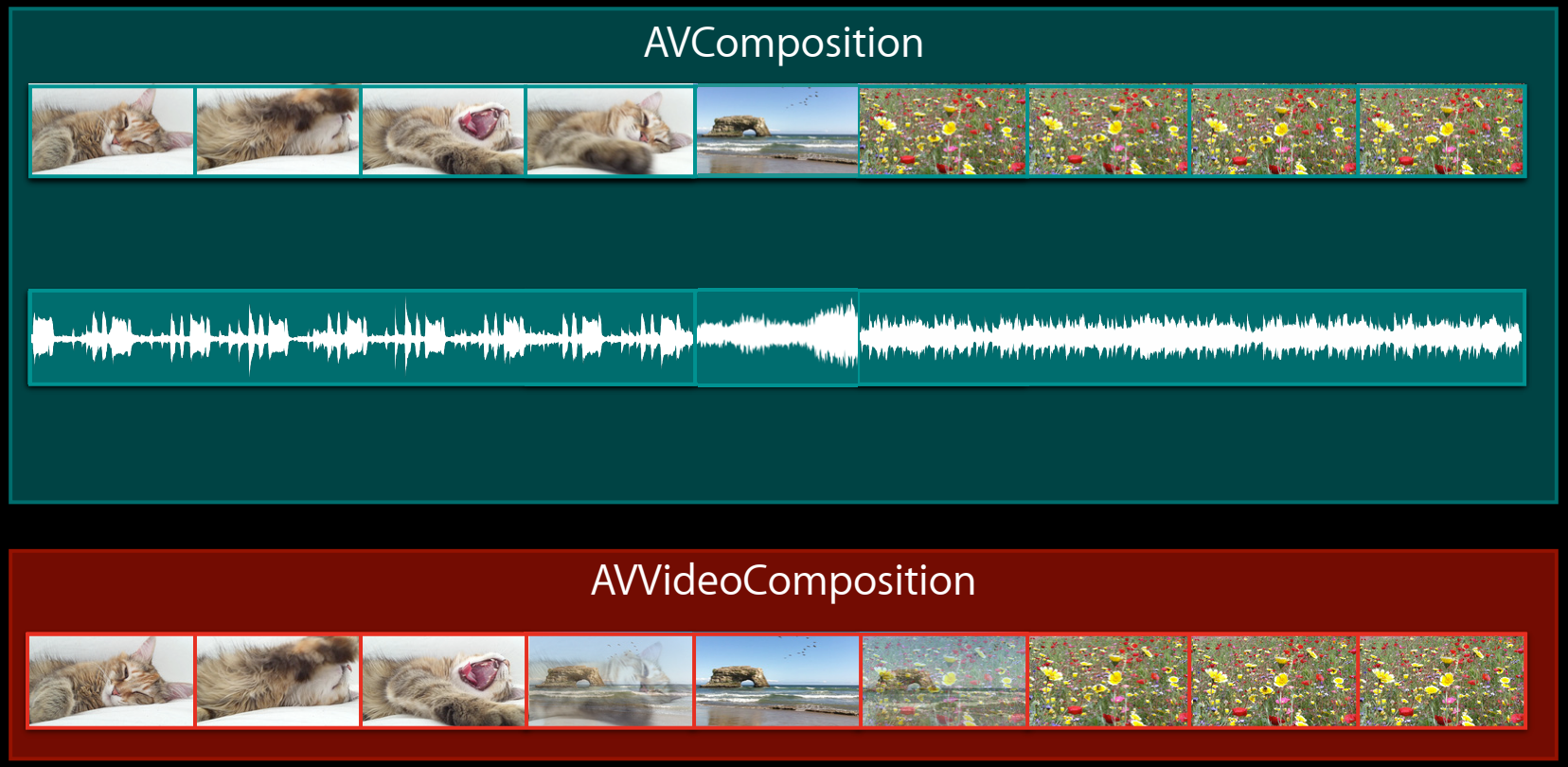
- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Demo

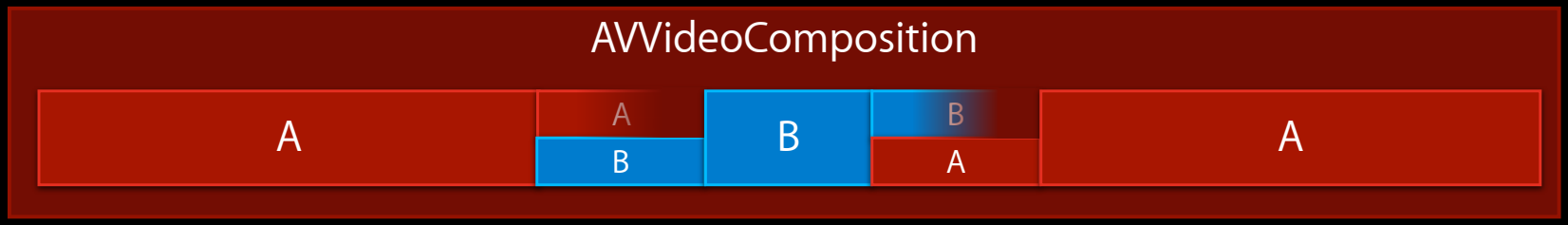
Video transitions

Sample code: AVEditDemoIPad (see SimpleEditor.m)

AVComposition and AVVideoComposition

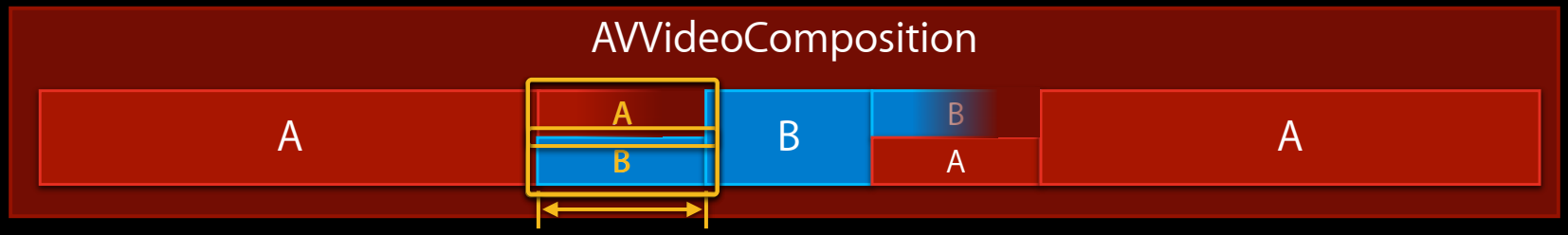


AVVideoComposition



- Has an array of AVVideoCompositionInstructions
  - Each instruction describes the output video in terms of input layers (AVVideoCompositionLayerInstruction)
    - Each layer has an opacity and an affine transform
    - Opacity can be tweened — e.g., for a cross-fade
    - Affine transform can be tweened — e.g., for a push transition

AVVideoCompositionInstruction




```
AVMutableVideoCompositionInstruction *transition =
    [AVMutableVideoCompositionInstruction videoCompositionInstruction];
```

```
transition.timeRange = ...;
```

```
AVMutableVideoCompositionLayerInstruction *fromLayer =
    [AVMutableVideoCompositionLayerInstruction
        videoCompositionLayerInstructionWithAssetTrack:trackA];
```

```
// Fade out trackA by setting a ramp from 1.0 to 0.0.
[fromLayer setOpacityRampFromStartOpacity:1.0 toEndOpacity:0.0
    timeRange:...];
```

```
AVMutableVideoCompositionLayerInstruction *toLayer = ...
```

```
layerInstructions = [NSArray arrayWithObjects:fromLayer, toLayer, nil];
```

AVVideoComposition

```
AVMutableVideoComposition *videoComposition =
    [AVMutableVideoComposition videoComposition];
```

```
videoComposition.instructions = [NSArray arrayWithObject:transition];
```

```
videoComposition.frameDuration = CMTimeMake(1, 30);
```

```
videoComposition.renderSize = CGSizeMake(1280, 720);
```

```
videoComposition.renderScale = 0.5; // for playback only
```

Using AVVideoComposition

- For playback:
  - playerItem.videoComposition = videoComposition;
- For image generation:
  - assetImageGenerator.videoComposition = videoComposition;
- For export:
  - assetExportSession.videoComposition = videoComposition;
- To retrieve rendered frames:
  - assetReaderVideoCompositionOutput.videoComposition
    = videoComposition;

Pitfalls

Troubleshooting

- AVVideoCompositionInstructions must not overlap or contain gaps

AVComposition

AVVideoComposition

- AVVideoComposition must not be shorter than AVComposition

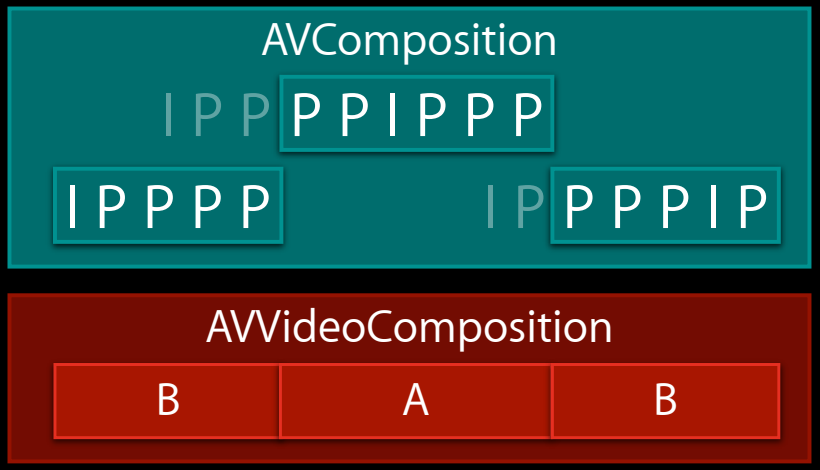
AVComposition

AVVideoComposition

Temporal Video Compression

- Not restricted to editing at key frames
- Use AVVideoComposition and alternating segments between two video tracks to give AV Foundation more time for catchup decoding

I P P P P I P P P P I P P P P I P P P P P ...



Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Editing APIs in AV Foundation

Scenarios

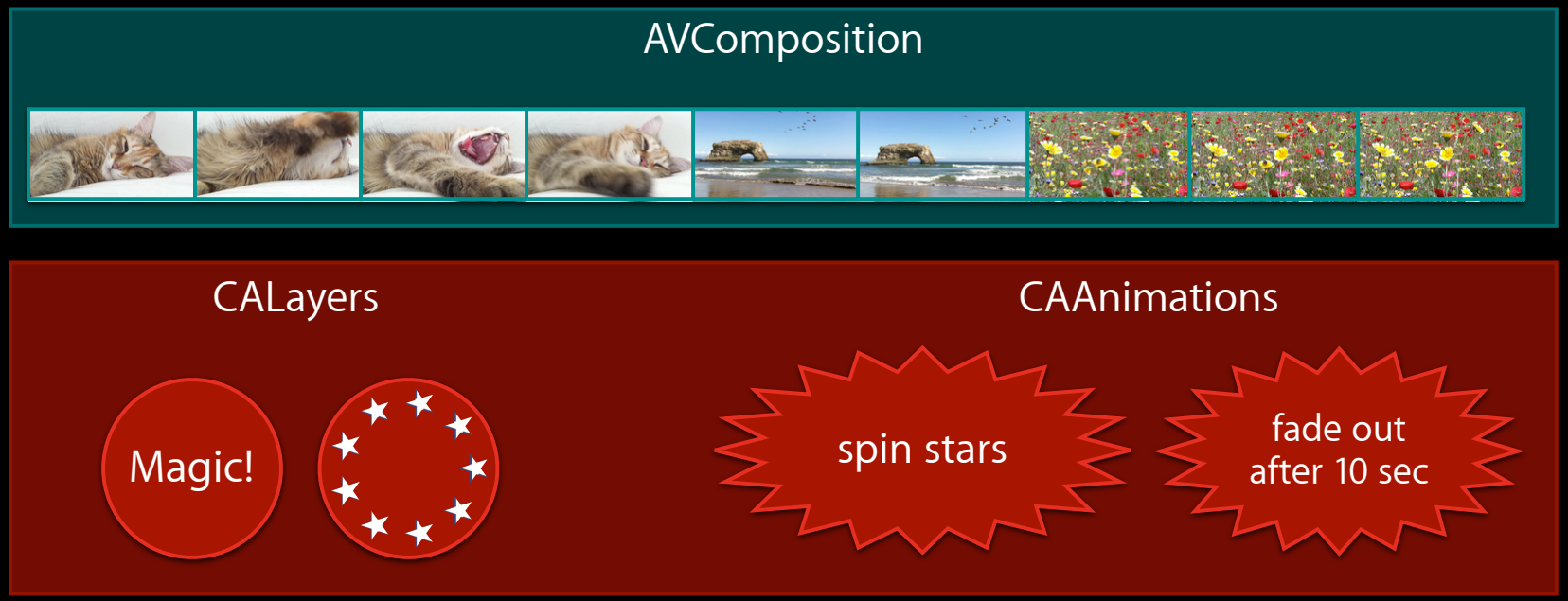
- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Demo

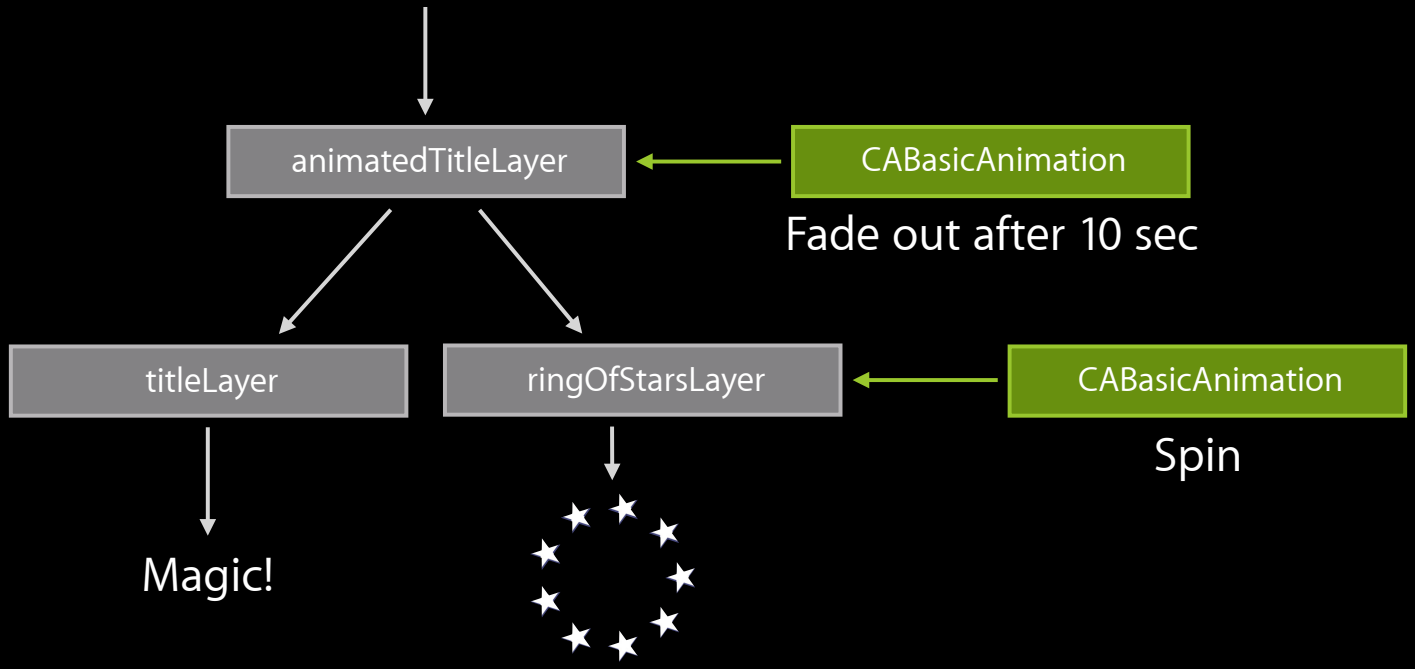
Core Animation in movies

Sample code: AVEditDemoIPad (see SimpleEditor.m)

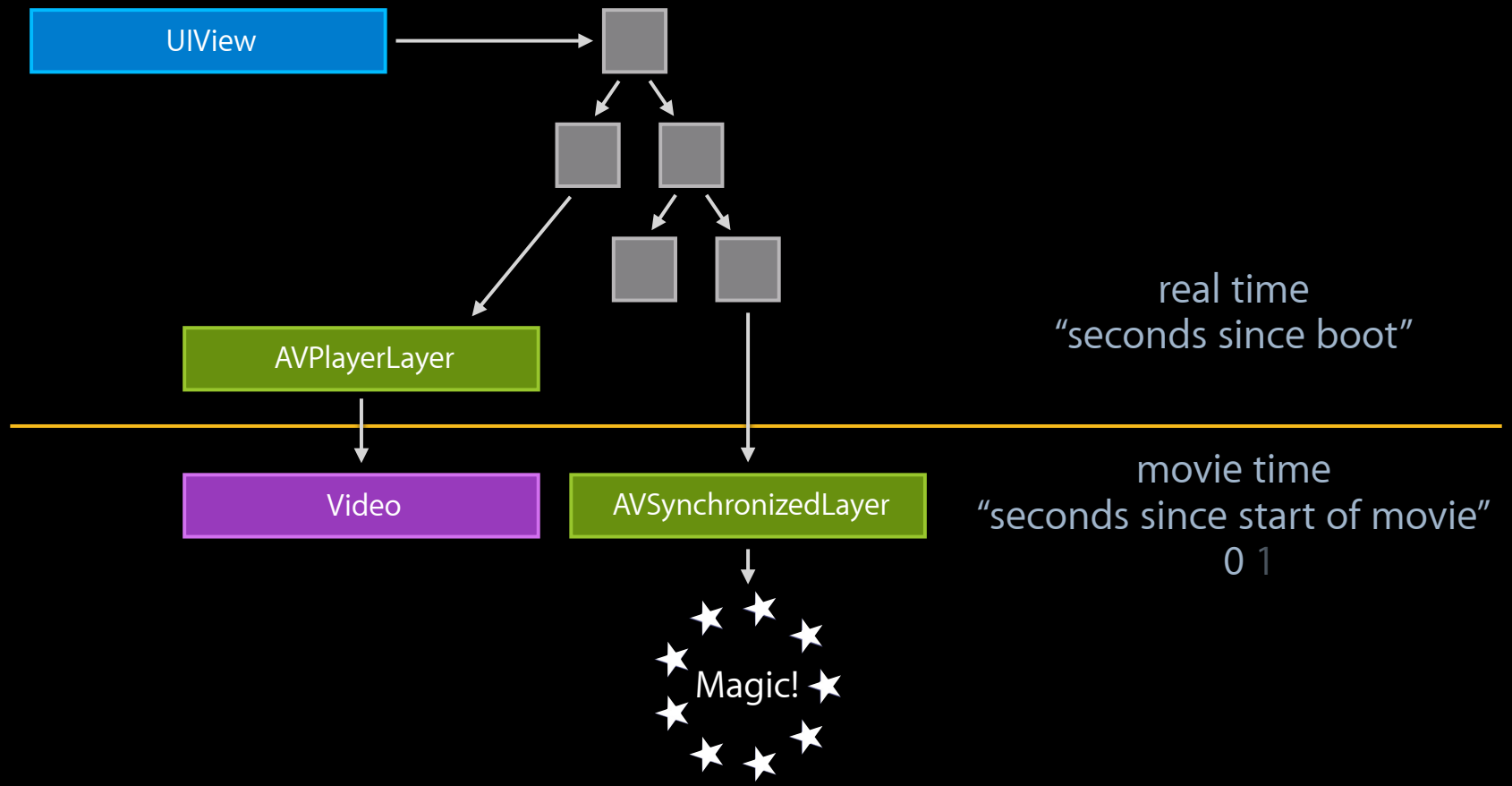
Core Animation in Movies



AVEditDemo: Core Animation



Animation, Video, and Time

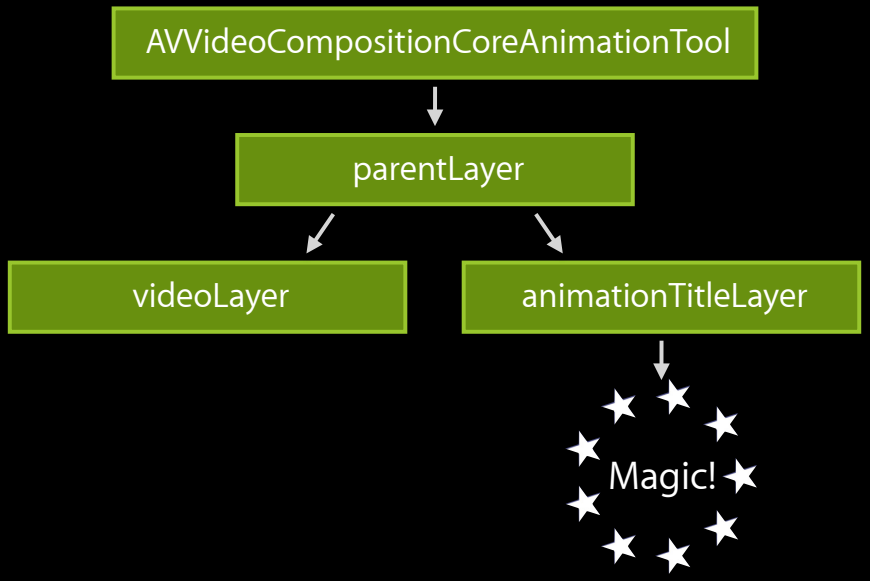


Playback with Core Animation

- Use AVSynchronizedLayer to make animation use movie timing

Export with Core Animation

- Use AVVideoCompositionCoreAnimationTool to integrate Core Animation as a "post-processing" video stage



- Set compositionInstruction.enablePostProcessing to NO to skip Core Animation rendering when not needed

Multitasking

iOS

- Core Animation use in the background will cause the export to fail

Core Animation Gotchas

Troubleshooting

Core Animation features that are convenient for real-time animation but do not work well in movies

- Zero beginTime is automatically translated to CACurrentMediaTime()
  - Use a small nonzero number: e.g., 1e-100 or -1e-100 (AVCoreAnimationBeginTimeZero)
- Animations are automatically removed after Core Animation thinks they have passed

```
animation.removedOnCompletion = NO;
```

Core Animation Gotchas

Core Animation features that are convenient for real-time animation but do not work well in movies

- Disable implicit animations

```
[CATransaction begin];
  [CATransaction setDisableActions:YES];
  // your layer code here
[CATransaction commit];
```

Stretch It Out

- Core Animation contributions may continue past the end of an AVComposition

- Need to explicitly indicate how long playback or export should run
  - Set playerItem.forwardPlaybackEndTime for playback
  - Set assetExportSession.timeRange for export

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

Editing APIs in AV Foundation

Scenarios

- Create an image for a time
- Export or trim a movie
- Cutting together multiple clips
- Audio mixing
- Video transitions
- Incorporating Core Animation in movies
- Reading audio and video from a movie
- Writing a movie with your own audio and video data

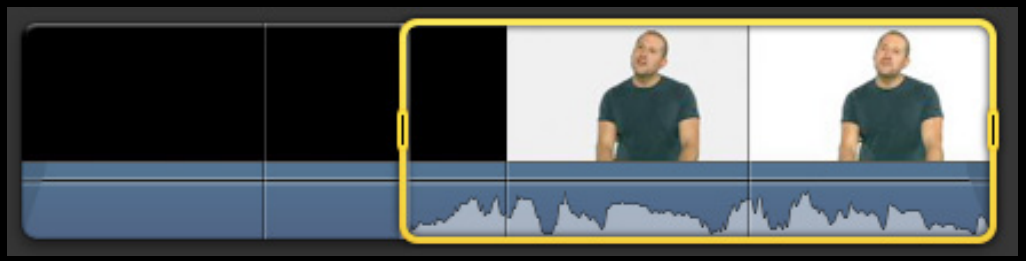
Working Directly with Media Data

Adam Sonnanstine
AV Foundation Engineer

Agenda

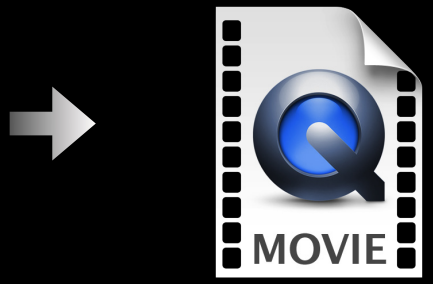
- Why and how
- Read from assets
- Write to file
- File-writing considerations

Precise Graphical Editing



AVAssetReader

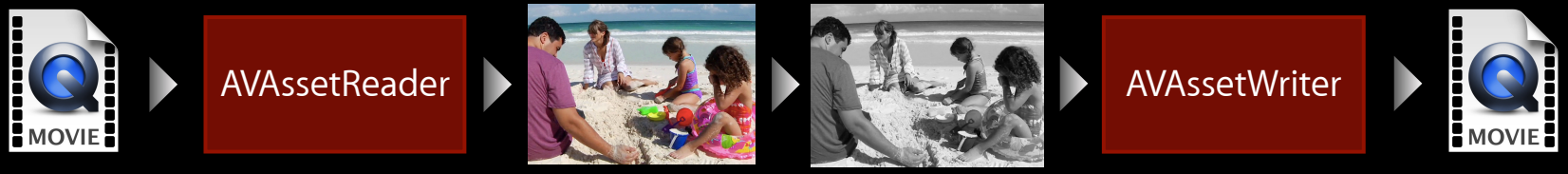
Record Live Game Play



AVAssetWriter

Pixel-Level Visual Effects

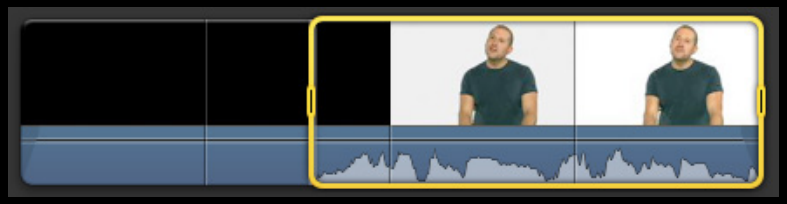
Combine AVAssetReader and AVAssetWriter



Reading Media Data from Assets

AVAssetReader

AVAssetReader Example Use Cases

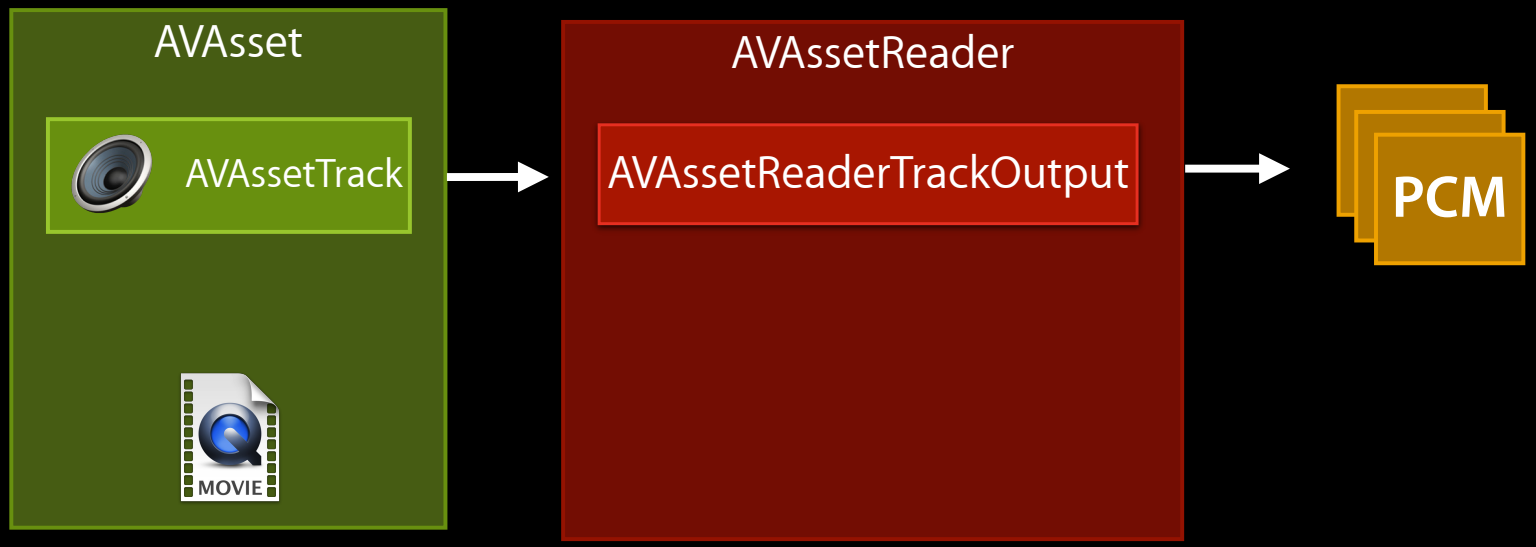


Drawing audio waveform

Source for video frames

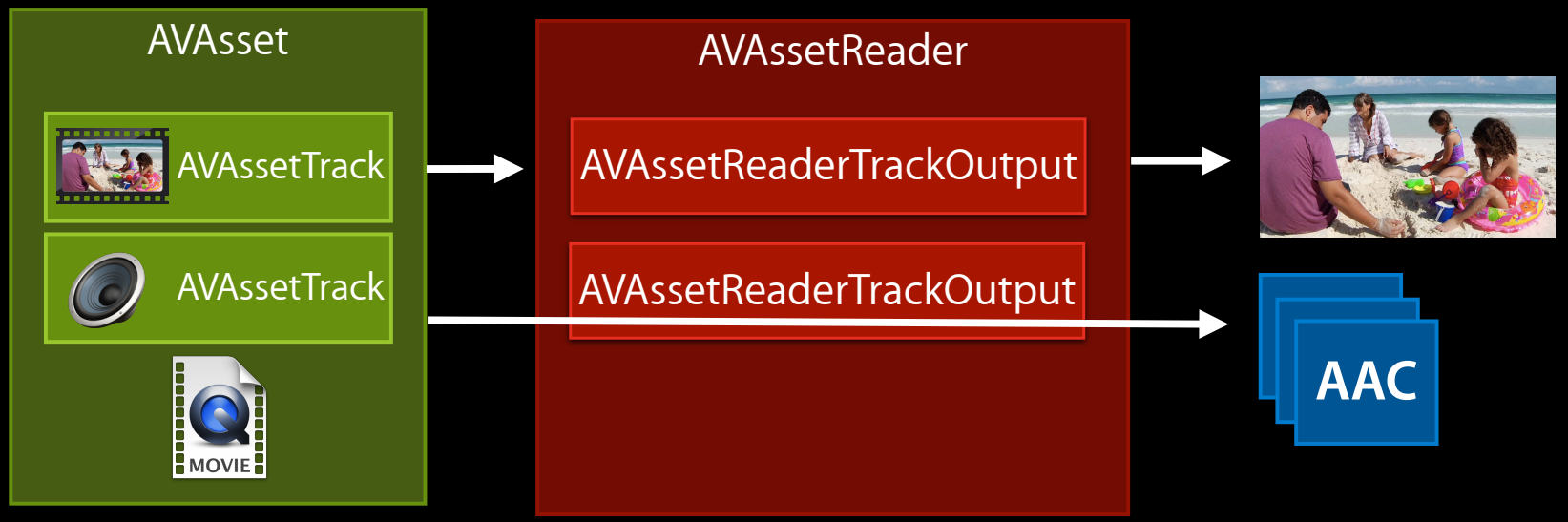
Waveform Scenario

Reading from a single asset track



Pixel Filtering Scenario

AVAssetReaderTrackOutput



How to Choose the Tracks?

- All tracks must come from the same asset
- To combine tracks from multiple assets, use AVComposition

Setting up the Objects

- Instantiate asset reader
- Add outputs
- Configure
- Start reading

Setting up the Objects

- Instantiate asset reader

```
AVAssetReader *assetReader = [AVAssetReader assetReaderWithAsset:asset
  error:&error];
```

- Add outputs
- Configure
- Start reading

Setting up the Objects

- Instantiate asset reader

```
AVAssetReader *assetReader = [AVAssetReader assetReaderWithAsset:asset
  error:&error];
```

- Add outputs

```
AVAssetReaderOutput *trackOutput = [AVAssetReaderTrackOutput
  assetReaderTrackOutputWithTrack:audioTrack
  outputSettings:outputSettings];

[assetReader addOutput:trackOutput];
```

- Configure
- Start reading

Setting up the Objects

- Instantiate asset reader

```
AVAssetReader *assetReader = [AVAssetReader assetReaderWithAsset:asset
  error:&error];
```

- Add outputs

```
AVAssetReaderOutput *trackOutput = [AVAssetReaderTrackOutput
  assetReaderTrackOutputWithTrack:audioTrack
  outputSettings:outputSettings];

[assetReader addOutput:trackOutput];
```

- Configure

```
assetReader.timeRange = CMTimeRangeMake(kCMTimeZero, CMTimeMake(5, 1));
```

- Start reading

Setting up the Objects

- Instantiate asset reader

```
AVAssetReader *assetReader = [AVAssetReader assetReaderWithAsset:asset
  error:&error];
```

- Add outputs

```
AVAssetReaderOutput *trackOutput = [AVAssetReaderTrackOutput
  assetReaderTrackOutputWithTrack:audioTrack
  outputSettings:outputSettings];

[assetReader addOutput:trackOutput];
```

- Configure

```
assetReader.timeRange = CMTimeRangeMake(kCMTimeZero, CMTimeMake(5, 1));
```

- Start reading

```
BOOL success = [assetReader startReading];
```

Reading Samples

```
BOOL done = NO;
while (!done) {
    CMSampleBufferRef sampleBuffer = [trackOutput copyNextSampleBuffer];
    if (sampleBuffer) {
        // Extract & draw waveform sample values
        CFRelease(sampleBuffer);
    } else {
        AVAssetReaderStatus status = assetReader.status;
        // act on asset reader status
        done = YES;
    }
}
```

Reading Samples

```
BOOL done = NO;
while (!done) {
    CMSampleBufferRef sampleBuffer = [trackOutput copyNextSampleBuffer];
    if (sampleBuffer) {
        // Extract & draw waveform sample values
        CFRelease(sampleBuffer);
    } else {
        AVAssetReaderStatus status = assetReader.status;
        // act on asset reader status
        done = YES;
    }
}
```

Reading Samples

```
BOOL done = NO;
while (!done) {
    CMSampleBufferRef sampleBuffer = [trackOutput copyNextSampleBuffer];
    if (sampleBuffer) {
        // Extract & draw waveform sample values
        CFRelease(sampleBuffer);
    } else {
        AVAssetReaderStatus status = assetReader.status;
        // act on asset reader status
        done = YES;
    }
}
```

Reading Samples

```
BOOL done = NO;
while (!done) {
    CMSampleBufferRef sampleBuffer = [trackOutput copyNextSampleBuffer];
    if (sampleBuffer) {
        // Extract & draw waveform sample values
        CFRelease(sampleBuffer);
    } else {
        AVAssetReaderStatus status = assetReader.status;
        // act on asset reader status
        done = YES;
    }
}
```

Reading Samples

```
BOOL done = NO;
while (!done) {
    CMSampleBufferRef sampleBuffer = [trackOutput copyNextSampleBuffer];
    if (sampleBuffer) {
        // Extract & draw waveform sample values
        CFRelease(sampleBuffer);
    } else {
        AVAssetReaderStatus status = assetReader.status;
        // act on asset reader status
        done = YES;
    }
}
```

Specify Format of Decoded Audio

outputSettings parameter

- Example: Linear PCM, 32-bit floating-point

```
[NSDictionary dictionaryWithObjectsAndKeys:
  [NSNumber numberWithUnsignedInt:kAudioFormatLinearPCM], AVFormatIDKey,
  [NSNumber numberWithInteger:32], AVLinearPCMBitDepthKey,
  [NSNumber numberWithBool:YES], AVLinearPCMIsFloatKey,
  nil];
```

- AVAudioSettings.h

Specify Format of Decoded Video

outputSettings parameter

- Example: 32-bit ARGB, 640×480

```
[NSDictionary dictionaryWithObjectsAndKeys:
  [NSNumber numberWithUnsignedInt:kCVPixelFormatType_32ARGB],
     kCVPixelBufferPixelFormatTypeKey,
  [NSNumber numberWithInteger:640], kCVPixelBufferWidthKey,
  [NSNumber numberWithInteger:480], kCVPixelBufferHeightKey,
  nil];
```

- AVVideoSettings.h
- AVAssetReaderOutput.h

AVAssetReader is a Focused Tool

- Real time
- Seeking

- Offline
- Single-use
- Media data enumeration

Writing Audio and Video Data to File

AVAssetWriter

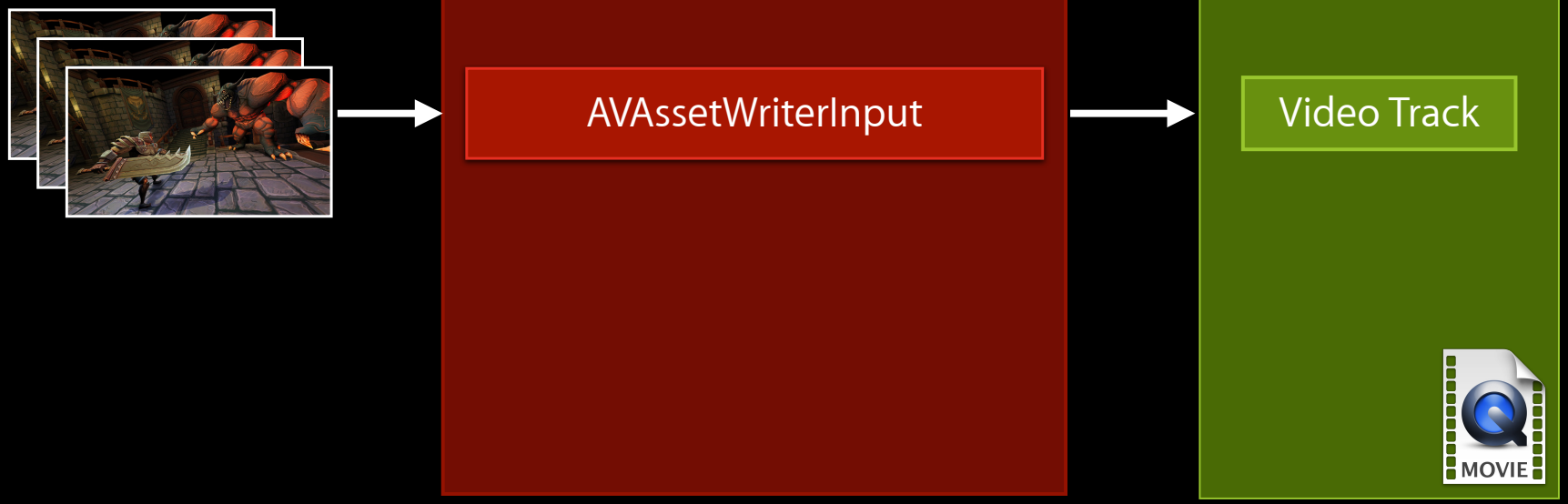
AVAssetWriter Example Use Cases

Recording live gameplay

Writing filtered video frames to file

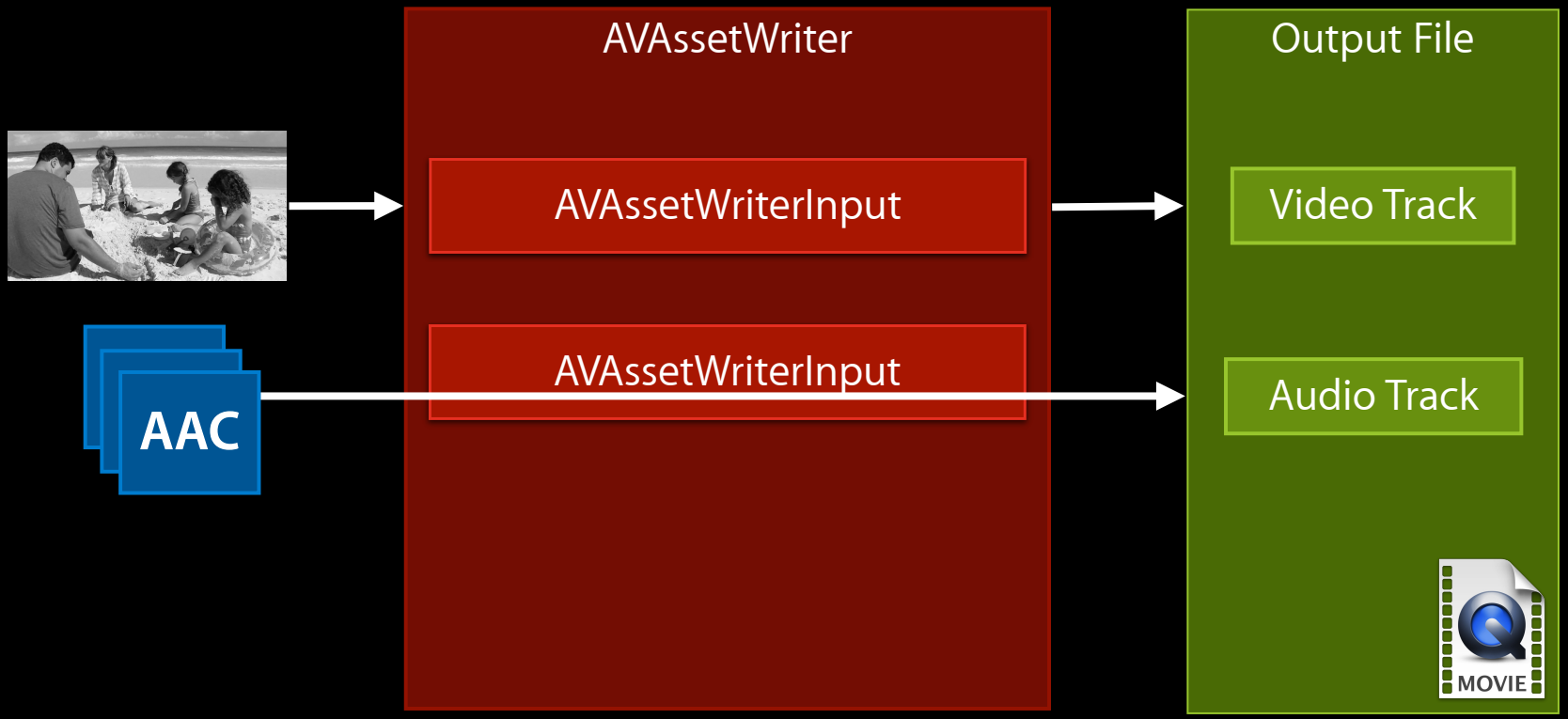
Game Recording Scenario

AVAssetWriterInput



Pixel Filtering Scenario

AVAssetWriterInput



Setting up AVAssetWriter + Input

- Instantiate asset writer
- Add inputs
- Configure
- Start writing

Setting up AVAssetWriter + Input

- Instantiate asset writer

```
AVAssetWriter *assetWriter = [AVAssetWriter
  assetWriterWithURL:localOutputURL fileType:AVFileTypeQuickTimeMovie
  error:&localError];
```

- Add inputs
- Configure
- Start writing

Setting up AVAssetWriter + Input

- Instantiate asset writer

```
AVAssetWriter *assetWriter = [AVAssetWriter
  assetWriterWithURL:localOutputURL fileType:AVFileTypeQuickTimeMovie
  error:&localError];
```

- Add inputs

```
AVAssetWriterInput *videoInput =
  [AVAssetWriterInput assetWriterInputWithMediaType:AVMediaTypeVideo
  outputSettings:compressionVideoSettings];
[assetWriter addInput:videoInput];
```

- Configure
- Start writing

Setting up AVAssetWriter + Input

- Instantiate asset writer

```
AVAssetWriter *assetWriter = [AVAssetWriter
  assetWriterWithURL:localOutputURL fileType:AVFileTypeQuickTimeMovie
  error:&localError];
```

- Add inputs

```
AVAssetWriterInput *videoInput =
  [AVAssetWriterInput assetWriterInputWithMediaType:AVMediaTypeVideo
  outputSettings:compressionVideoSettings];
[assetWriter addInput:videoInput];
```

- Configure

```
assetWriter.shouldOptimizeForNetworkUse = YES;
```

- Start writing

Setting up AVAssetWriter + Input

- Instantiate asset writer

```
AVAssetWriter *assetWriter = [AVAssetWriter
  assetWriterWithURL:localOutputURL fileType:AVFileTypeQuickTimeMovie
  error:&localError];
```

- Add inputs

```
AVAssetWriterInput *videoInput =
  [AVAssetWriterInput assetWriterInputWithMediaType:AVMediaTypeVideo
  outputSettings:compressionVideoSettings];
[assetWriter addInput:videoInput];
```

- Configure

```
assetWriter.shouldOptimizeForNetworkUse = YES;
```

- Start writing

```
BOOL success = [assetWriter startWriting];
```

AVAssetWriter Sample-Writing Phase

- Start session
- Append samples
- Finish writing

AVAssetWriter Sample-Writing Phase

- Start session

```
[assetWriter startSessionAtSourceTime:kCMTimeZero];
```

- Append samples
- Finish writing

AVAssetWriter Sample-Writing Phase

- Start session

```
[assetWriter startSessionAtSourceTime:kCMTimeZero];
```

- Append samples

```
CMSampleBufferRef sampleBuffer = ...;
success = [videoInput appendSampleBuffer:sampleBuffer];
```

- Finish writing

AVAssetWriter Sample-Writing Phase

- Start session

```
[assetWriter startSessionAtSourceTime:kCMTimeZero];
```

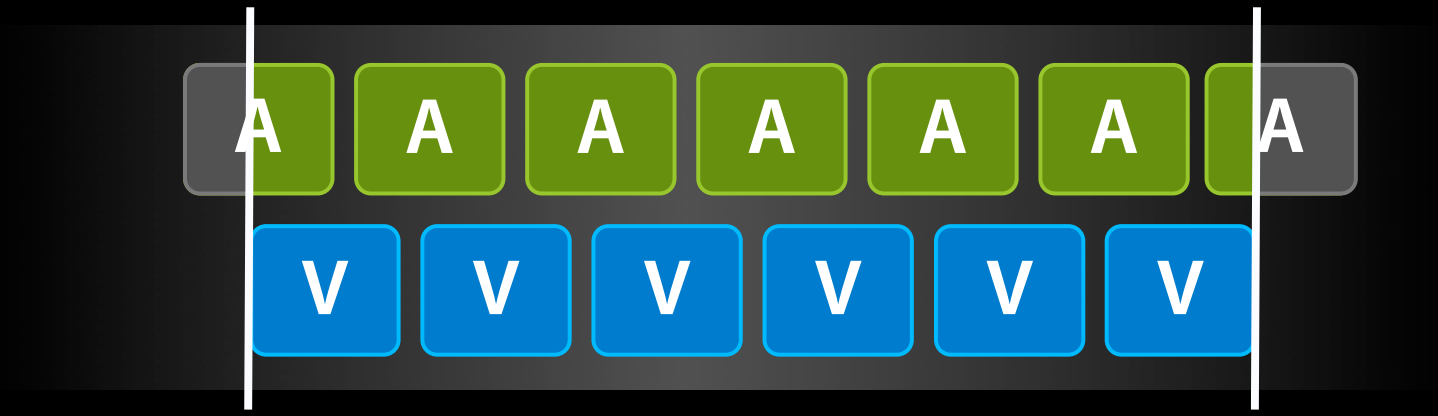
- Append samples

```
CMSampleBufferRef sampleBuffer = ...;
success = [videoInput appendSampleBuffer:sampleBuffer];
```

- Finish writing

```
success = [assetWriter finishWriting];
if (!success) {
  NSError *error = assetWriter.error;
  // deal with error
}
```

Fine-Tuning Start and End Times



-startSessionAtSourceTime:

-endSessionAtSourceTime:

Output Settings for Audio Compression

AVAudioSettings.h

- Example: AAC, 128 kbps, 44100 Hz, Stereo

```
[NSDictionary dictionaryWithObjectsAndKeys:
  [NSNumber numberWithUnsignedInt:kAudioFormatMPEG4AAC], AVFormatIDKey,
  [NSNumber numberWithInteger:128000], AVEncoderBitRateKey,
  [NSNumber numberWithInteger:44100], AVSampleRateKey,
  stereoChannelLayoutAsData, AVChannelLayoutKey,
  [NSNumber numberWithUnsignedInteger:2], AVNumberOfChannelsKey,
  nil];
```

Output Settings for Video Compression

AVVideoSettings.h

- Example: H.264, 640×480

```
[NSDictionary dictionaryWithObjectsAndKeys:
  AVVideoCodecH264, AVVideoCodecKey,
  [NSNumber numberWithInteger:640], AVVideoWidthKey,
  [NSNumber numberWithInteger:480], AVVideoHeightKey,
  nil];
```

Integration with Professional Video

Apple ProRes

Only on Mac OS

- Two Flavors:
  - `AVVideoCodecAppleProRes422`
  - `AVVideoCodecAppleProRes4444`
- More info and white paper:
  - http://www.apple.com/finalcutstudio/finalcutpro/apple-prores.html

Different Modes for Different Sources

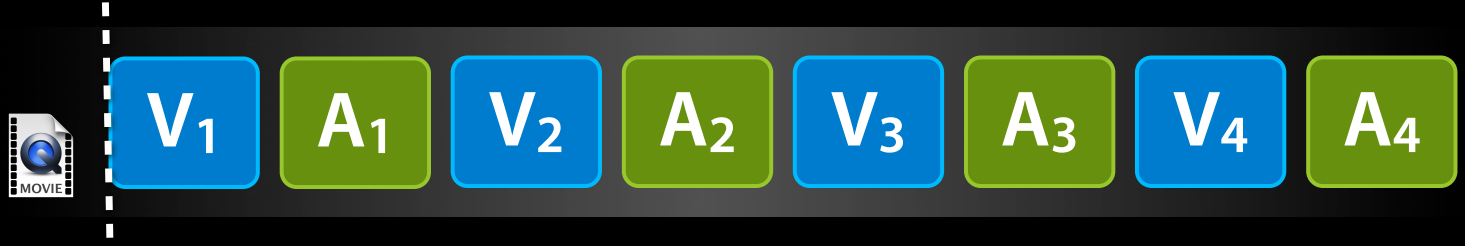
- Different sources of media data have different characteristics:
  - Offline vs. realtime
  - Pull vs. push
- AVAssetWriter can be configured to support each of these

How to Lay Out Media Data in a File?

Audio/video interleaving

- One track after another:

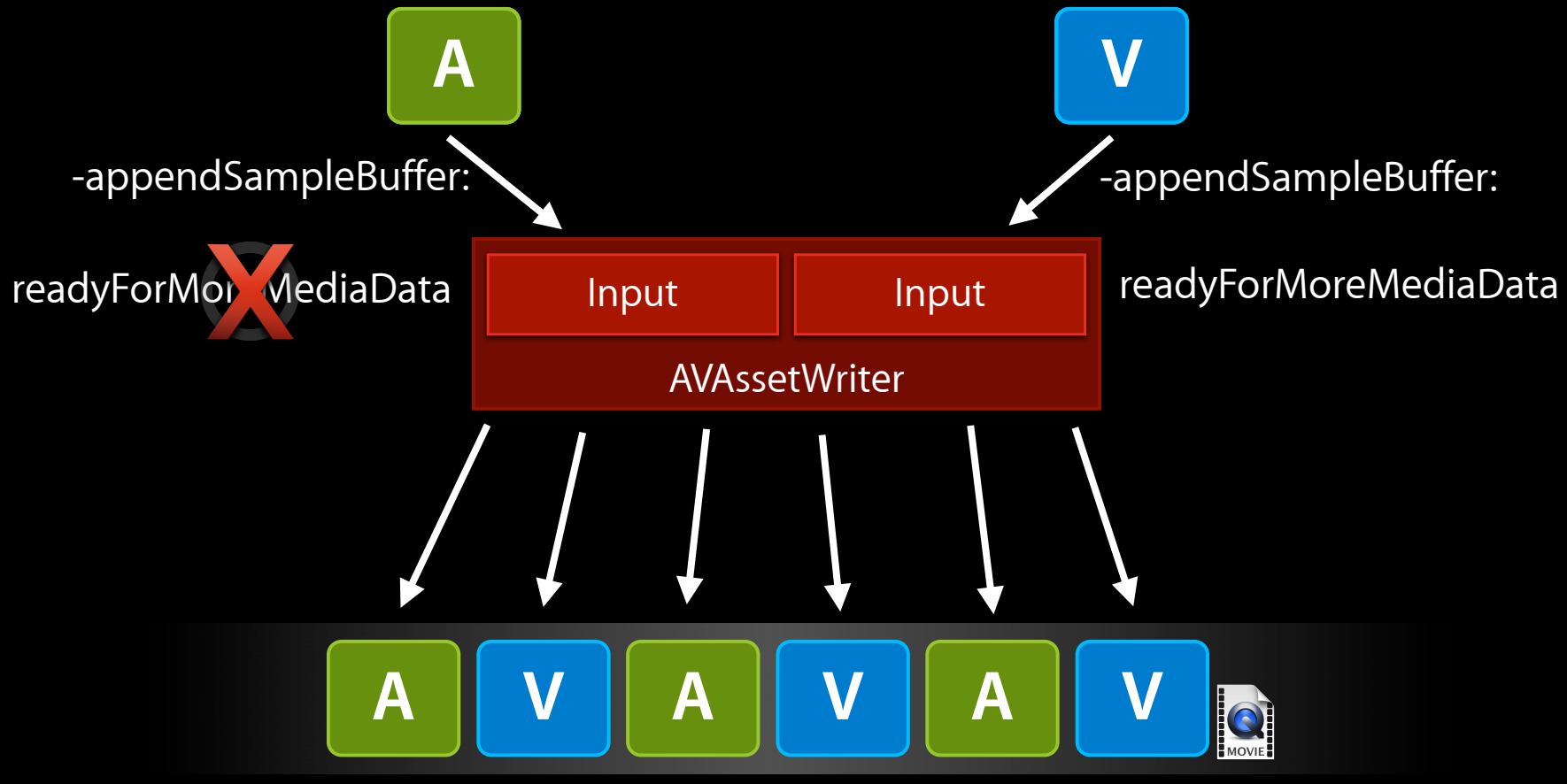
- Mixed together:



- AVAssetWriter tries to interleave tracks for efficient playback I/O

AVAssetWriterInput Feedback

readyForMoreMediaData property



Using AVAssetWriter with Pull-Model Sources

-requestMediaDataWhenReady method

```
[myAVAssetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:^{
  while (myAVAssetWriterInput.readyForMoreMediaData) {
     CMSampleBufferRef sampleBuffer = ...;
     if (sampleBuffer) {
        [myAVAssetWriterInput appendSampleBuffer:sampleBuffer];
        CFRelease(sampleBuffer);
     } else {
        [myAVAssetWriterInput markAsFinished];
        break;
     }
  }
}];
```

Using AVAssetWriter with Pull-Model Sources

-requestMediaDataWhenReady method

```
[myAVAssetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:^{
  while (myAVAssetWriterInput.readyForMoreMediaData) {
     CMSampleBufferRef sampleBuffer = ...;
     if (sampleBuffer) {
        [myAVAssetWriterInput appendSampleBuffer:sampleBuffer];
        CFRelease(sampleBuffer);
     } else {
        [myAVAssetWriterInput markAsFinished];
        break;
     }
  }
}];
```

Using AVAssetWriter with Pull-Model Sources

-requestMediaDataWhenReady method

```
[myAVAssetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:^{
  while (myAVAssetWriterInput.readyForMoreMediaData) {
     CMSampleBufferRef sampleBuffer = ...;
     if (sampleBuffer) {
        [myAVAssetWriterInput appendSampleBuffer:sampleBuffer];
        CFRelease(sampleBuffer);
     } else {
        [myAVAssetWriterInput markAsFinished];
        break;
     }
  }
}];
```

Using AVAssetWriter with Pull-Model Sources

-requestMediaDataWhenReady method

```
[myAVAssetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:^{
  while (myAVAssetWriterInput.readyForMoreMediaData) {
     CMSampleBufferRef sampleBuffer = ...;
     if (sampleBuffer) {
        [myAVAssetWriterInput appendSampleBuffer:sampleBuffer];
        CFRelease(sampleBuffer);
     } else {
        [myAVAssetWriterInput markAsFinished];
        break;
     }
  }
}];
```

Using AVAssetWriter with Pull-Model Sources

-requestMediaDataWhenReady method

```
[myAVAssetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:^{
  while (myAVAssetWriterInput.readyForMoreMediaData) {
     CMSampleBufferRef sampleBuffer = ...;
     if (sampleBuffer) {
        [myAVAssetWriterInput appendSampleBuffer:sampleBuffer];
        CFRelease(sampleBuffer);
     } else {
        [myAVAssetWriterInput markAsFinished];
        break;
     }
  }
}];
```

Push Model Source

- Process + append buffers directly in push callback

```
- captureOutput:... didOutputSampleBuffer:sampleBuffer fromConnection:...
{
    if (assetWriterInput.readyForMoreMediaData) {
        // perform real-time processing
        [assetWriterInput appendSampleBuffer:sampleBuffer];
    } else {
        // drop video frames, potentially throttle upstream
    }
}
```

Push Model Source

- Process + append buffers directly in push callback

```
- captureOutput:... didOutputSampleBuffer:sampleBuffer fromConnection:...
{
    if (assetWriterInput.readyForMoreMediaData) {
        // perform real-time processing
        [assetWriterInput appendSampleBuffer:sampleBuffer];
    } else {
        // drop video frames, potentially throttle upstream
    }
}
```

Push Model Source

- Process + append buffers directly in push callback

```
- captureOutput:... didOutputSampleBuffer:sampleBuffer fromConnection:...
{
    if (assetWriterInput.readyForMoreMediaData) {
        // perform real-time processing
        [assetWriterInput appendSampleBuffer:sampleBuffer];
    } else {
        // drop video frames, potentially throttle upstream
    }
}
```

Push Model Source

- Process + append buffers directly in push callback

```
- captureOutput:... didOutputSampleBuffer:sampleBuffer fromConnection:...
{
    if (assetWriterInput.readyForMoreMediaData) {
        // perform real-time processing
        [assetWriterInput appendSampleBuffer:sampleBuffer];
    } else {
        // drop video frames, potentially throttle upstream
    }
}
```

- Avoid queueing, if possible

Realtime Source

- Writing all incoming data is more important than ideal interleaving
- Interleaving should naturally be OK

```
assetWriterInput.expectsMediaDataInRealTime = YES;
```

- Set for all inputs
- AVAssetWriter will try to stay out of your way

When Not to Use Reader and Writer

- AVAssetReader is not for playback!
- For simple exports and transcoding, use AVAssetExportSession
- For simple capture, use AVCaptureMovieFileOutput

Summary

• Create an image for a time	AVAssetImageGenerator
• Outputting a movie	AVAssetExportSession
• Combining multiple clips	AVComposition
• Audio volume adjustment	AVAudioMix
• Video transitions	AVVideoComposition
• Incorporating Core Animation	AVSynchronizedLayer and AVVideoCompositionCoreAnimationTool
• Reading audio and video data	AVAssetReader
• New movie with your data	AVAssetWriter

More Information

Eryk Vershen
Media Technologies Evangelist
evershen@apple.com

Documentation
AV Foundation Programming Guide
http://developer.apple.com/library/ios/#documentation/AudioVideo/Conceptual/AVFoundationPG/

Apple Developer Forums
http://devforums.apple.com

Related Sessions

Exploring AV Foundation	Presidio Tuesday 2:00PM
AirPlay and External Displays in iOS Apps	Presidio Tuesday 3:15PM
HTTP Live Streaming Update	Nob Hill Tuesday 4:30PM
Introducing AV Foundation Capture for Lion	Pacific Heights Wednesday 3:15PM
Capturing from the Camera using AV Foundation on iOS 5	Pacific Heights Wednesday 4:30PM

Labs

AV Foundation Lab	Graphics, Media & Games Lab B Thursday 9:00AM
QuickTime Lab	Graphics, Media & Games Lab D Thursday 9:00AM


WWDC2011