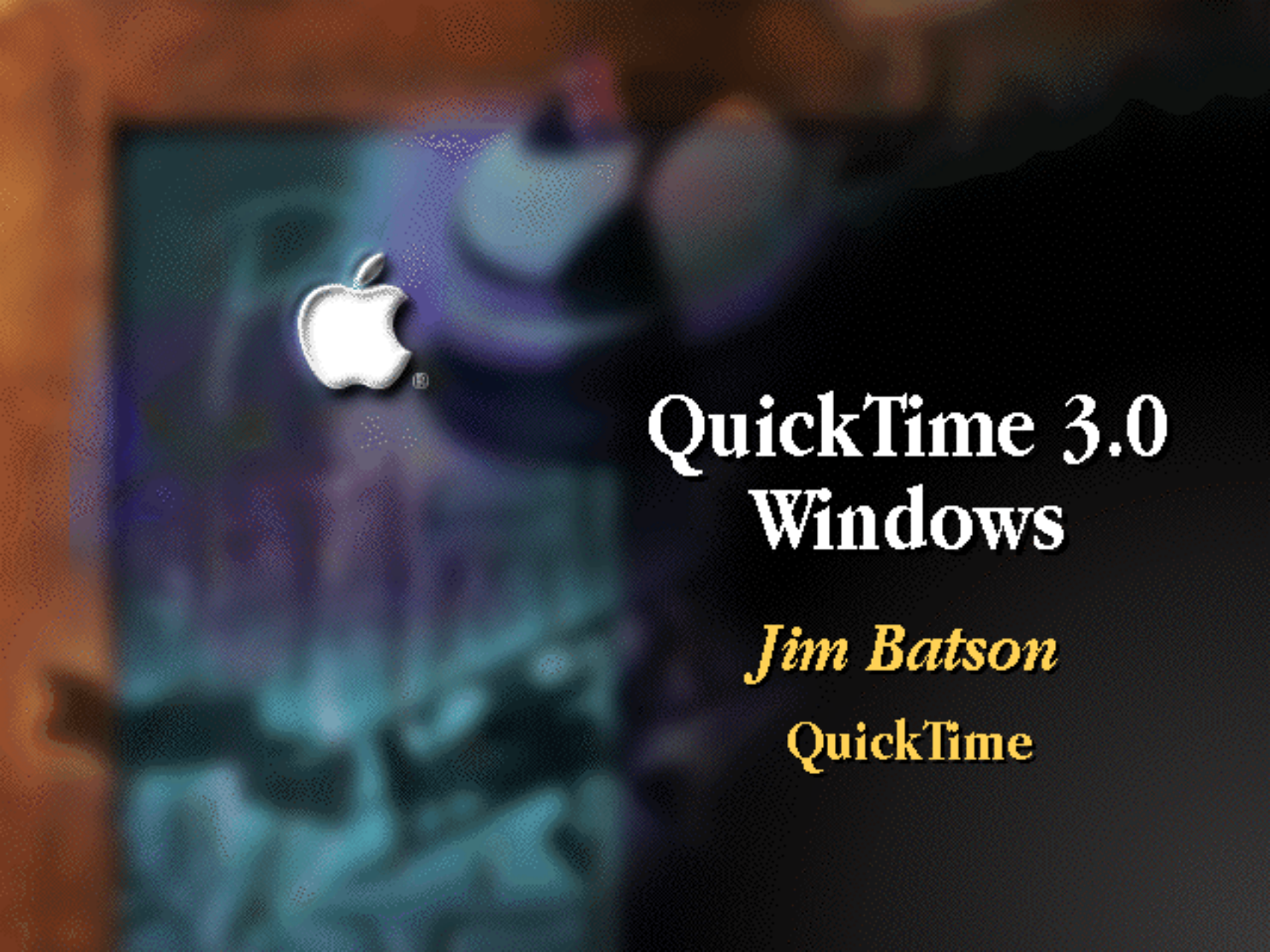

Worldwide
Developers
Conference


QuickTime 3.0
Windows
Jim Batson
QuickTime

QuickTime for Windows 1.0 - 2.1.2

- Play only solution
- Limited media types
- Only one video and one audio track
- Separate code base

QuickTime 3.0 Goals

- Windows is full parity with Macintosh
  - Movie editing, creation, and capturing
- All media types supported
- Same behaviour and features of Macintosh implementation
- Integration with standard Windows services
  - eg. MCI, OCX, VBX

Target Windows Environment

- 32-bit applications
  - Windows 95
  - Windows NT 4.0
- Video support
  - DirectDraw
  - GDI
- Sound support
  - DirectSound or WaveOut

QuickTime 3.0 Goals

- Provides a unified API across all platforms
  - Primary goal is to support cross platform code
    - Applications
    - QuickTime extensions
  - Your QuickTime code is almost the same if not identical
- Common code base
  - Simultaneous release
  - Identical api's vs. similar api's (eg. using HWND)
  - Incorporates Macintosh data types (Graphics, Files, Memory, ...)

QuickTime 3.0

- **Partial implementation of Mac Toolbox**
  - Portions required for QuickTime and QuickTime based applications
  - NOT a full implementation
  - NOT intended for porting Macintosh applications

QTW 2.1.2 to QuickTime 3.0 Migration

- Both can co-exist on a machine
  - Shipping apps do not have to worry about compatibility when 3.0 is installed


Demo

QTW 2.1.2 to QuickTime 3.0 Migration *(cont.)*

- Applications are easy
  - QTInitialize, QTTerminate => INIT_QTML, TERM_QTML
  - HWND => WindowPtr
  - Create/DestroyPortAssociation and conversion routines
  - Windows Messages => Macintosh Events
    - MCIsPlayerMessage => Win2MacEvent,MCIsPlayerEvent
  - Macintosh File conventions
    - LPCSTR => FSSpec for OpenMovieFile

QTW 2.1.2 to QuickTime 3.0 Migration *(cont.)*

- Codecs are totally different
  - The new model is like the Mac
  - Hard to code for the old model

QuickTime Graphics Environment

Macintosh

QuickTime/QuickDraw

HAL (GDevices, PixMaps, …)

On/OffScreen Memory

QuickTime Graphics Environment

Macintosh	Windows
QuickTime/QuickDraw	QuickTime/QuickDraw
HAL (GDevices, PixMaps, …)	HAL (GDevices, PixMaps, …)
	DirectDraw/GDI
On/OffScreen Memory	On/OffScreen Memory

Pixel Formats

- Macintosh
  - 8-bit index, 5-5-5 RGB BigEndian, 32-bit ARGB.

Pixel Formats

- Macintosh
  - 8-bit index, 5-5-5 RGB BigEndian, 32-bit ARGB.
- Windows
  - 8-bit index
  - 5-5-5, 5-6-5 RGB Little/Big Endian
  - 24-bit RGB, BGR
  - 32-bit ARGB, BGRA, ABGR, RGBA

Pixel Formats

- Macintosh
  - 8-bit index, 5-5-5 RGB BigEndian, 32-bit ARGB.
- Windows
  - 8-bit index
  - 5-5-5, 5-6-5 RGB Little/Big Endian
  - 24-bit RGB, BGR
  - 32-bit ARGB, BGRA, ABGR, RGBA
- PixMap's planeBytes encodes the pixel format

Pixel Formats

- Macintosh
  - 8-bit index, 5-5-5 RGB BigEndian, 32-bit ARGB.
- Windows
  - 8-bit index
  - 5-5-5, 5-6-5 RGB Little/Big Endian
  - 24-bit RGB, BGR
  - 32-bit ARGB, BGRA, ABGR, RGBA
- PixMap's planeBytes encodes the pixel format
- If you just support Macintosh Pixel formats, you need to endian flip

Endian Issues

- In CPU Register

High byte (15…8)	Low byte (7…0)

Endian Issues

- In CPU Register

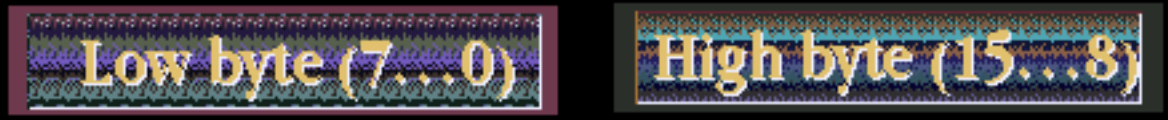
- What is Big Endian?

Endian Issues

- In CPU Register

- What is Big Endian?

- What is Little Endian?



Endian Issues

- What is Native Endian?
- Endian.h is your friend
- Cross-platform code must be Endian aware

QuickDraw

- GWorlds created using native bitmaps (DIB)
- Fully featured
- Handles all pixel formats
- NewGWorldEx for creating non-Mac pixel format GWorlds

QuickTime Audio Environment

Macintosh

QuickTime Audio Environment

Macintosh

SoundManager

HAL (SDev Component Interface)

Sound Hardware SDev

Windows

SoundManager

HAL (SDev Component Interface)

DirectSound SDev

WaveOut SDev

Audio Data Path

- Sound formats supported
  - 8-bit offset
  - 8-bit signed
  - 16-bit signed Big/Little Endian
  - 32/64-bit Float
  - IMA, Mace, ADPCM, uLaw, aLaw
  - Plug in your own compressors
- Internal formats
  - Data exchange with Sound Manager is in native format
  - Compressors, Decompressors, H/W SDevs

Audio Input

- Macintosh
  - Sound device drivers
- Windows
  - Sound input components (instead of a driver reference, a component instance)
  - WaveIn provided
  - Must produce Native Endian data

Video Input

- Macintosh and Windows
  - Common video digitizer API (VDIG)
- QuickTime requires some user interface elements
  - Sequence Grabber panels
    - Sound and video
  - Standard Compression extensions
  - Movie info panels
- Toolbox elements exist to support this level of UI


Demo

Resources

- Yes, Virginia, Mac resources can exist on Windows
- Available for your code
  - Required for components
- Available for getting data stored in resource files
  - eg. Sound Designer II, PICS
- Dual fork files on Macintosh
  - Hides the separate resource file from users

Resources

- Single fork files on Windows
  - If there is a resource file, it is visible to the user
- Movies
  - Movie resource can be in the resource fork or data fork
  - Reference movies vs. self-contained movies
  - Data fork only movies
  - Data fork only movies with external references

Resources

- Single fork files on Windows
  - If there is a resource file, it is visible to the user
- Movies
  - Movie resource can be in the resource fork or data fork
  - Reference movies vs. self-contained movies
  - Data fork only movies
  - Data fork only movies with external references
- QuickTime can use separate resource fork files
  - But don't hurt the users!

Resources

- For data files, all the popular forms
  - Parallel resource directory
    - resource.frk, .rsrc
  - Parallel resource file
    - !filename, filename.#res, filename.qtr
  - Multi-stream NT files
  - Apple Single

Resources

- For data files, all the popular forms
  - Parallel resource directory
    - resource.frk, .rsrc
  - Parallel resource file
    - !filename, filename.#res, filename.qtr
  - Multi-stream NT files
  - Apple Single
- For executable files (.QTX, .DLL, .EXE)
  - .QTR resource files can be used
  - The resource fork can be merged with the executable file

Component Manager

- Mechanism to plug your code into QuickTime
- Thing Resource
  - Type, SubType, Manufacturer
  - Code Reference
    - Code Resource/Code Fragment
    - DLL entry point
- Components delivered as .QTX files
  - Is a renamed DLL file
    - Component entry points exported
  - Thing resource

Files

- QuickTime expects Macintosh file references and aliases
- Files can be manipulated using the File Manager
- Window's file handles can be converted to Mac file references

Scheduling Model

- Execution sequencing
- Shared data protection

Macintosh Interrupts

- Interrupt 1 — Timer tasks
- Interrupt 2 — Sound tasks, Async I/O tasks
- Interrupt 3 — Deferred tasks
- Application tasks

Thread Model

- Timer thread—priority A
- Sound thread—priority B
- Async I/O thread—priority B
- Deferred task thread—priority C
- Application threads—priority D

Data protection

- Data shared across threads must be protected
- Queues
  - InitializeQHdr and TerminateQHdr
  - QHdr struct has a mutex field
- Cross-platform mutex implementation
  - Protect your own data

Re-Entrancy

- Mac Toolbox is generally not re-entrant
- Only one thread can be executing in QTML at a time
  - Managed transparently by QuickTime

Re-Entrancy

- Mac Toolbox is generally not re-entrant
- Only one thread can be executing in QTML at a time
  - Managed transparently by QuickTime
- Threads emulating Macintosh Interrupts are the exception
  - MUST follow Macintosh interrupt rules
    - Only call interrupt safe toolbox routines
  - MUST register and unregister with QuickTime

Event Model

- Macintosh
  - Classic WaitNextEvent loop
  - Application dispatches to appropriate code based on the event type
- Windows
  - Message loop
  - Application sends messages to system to dispatch to appropriate code
  - Some window messages get dispatched to WndProcs without appearing in the message loop

Event Model

- QuickTime uses the Macintosh Event Model
- WinToMacEvent converts a Windows message to a Mac EventRecord
- Then call MCIsPlayerEvent

Window/Port Association

- Implicit connection between HWNDs and WindowPtr
- Given a WindowPtr created by QuickTime, you can get the HWND
  - You can use GDI to draw
- Can create a GrafPort from an HWND
  - You can use QuickDraw to draw

Use the Paradigm Best for You

- System types
  - Windows Message/EventRecord
  - HWND/WindowPtr
  - HFILE/file reference number
  - DIB/GWorld
- Can mix and match

QuickTime 3.0

- Summary


For More Info
cwiltgen@apple.com
www.quicktime.apple.com