QTCaptureDevice Class Reference

QuickTime > Cocoa

2009-05-06

Contents

QTCaptureDevice Class Reference 7

Overview 7
Tasks 8
    Finding Devices 8
    Using a Device 8
    Getting Information About a Device 9
Class Methods 9
    defaultInputDeviceWithMediaType: 9
    deviceWithUniqueID: 10
    inputDevices 10
    inputDevicesWithMediaType: 11
Instance Methods 11
    attributeForKey: 11
    attributeIsReadOnly: 12
    close 12
    deviceAttributes 12
    formatDescriptions 13
    hasMediaType: 13
    isConnected 14
    isInUseByAnotherApplication 14
    isOpen 14
    localizedDisplayName 15
    modelUniqueID 15
    open: 16
    setAttribute:forKey: 16
    setDeviceAttributes: 17
    uniqueID 17
Constants 17
    Device Attributes 17
    Enumunerations 20
Notifications 22
    QTCaptureDeviceWasConnectedNotification 22
    QTCaptureDeviceWasDisconnectedNotification 23
    QTCaptureDeviceFormatDescriptionsWillChangeNotification 23
    QTCaptureDeviceFormatDescriptionsDidChangeNotification 23
    QTCaptureDeviceAttributeWillChangeNotification 23
    QTCaptureDeviceAttributeDidChangeNotification 23

Document Revision History 25

Index 27

Tables

QTCaptureDevice Class Reference 7

Table 1 Media types supported by `QTCaptureDevice` 7

QTCaptureDevice Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSObject (NSObject)
Framework	/System/Library/Frameworks/QTKit.framework
Availability	Available in QuickTime 7.2.1 and later.
Declared in	QTCaptureDevice.h
Related sample code	LiveVideoMixer3 QT Capture Widget QTRecorder

Overview

This class represents an available capture device. Each instance of `QTCaptureDevice` corresponds to a capture device that is connected or has been previously connected to the user's computer during the lifetime of the application. Instances of `QTCaptureDevice` cannot be created directly. A single unique instance is created automatically whenever a device is connected to the computer and can be accessed using the `deviceWithUniqueID:` (page 10) class method. An array of all currently connected devices can also be obtained using the `inputDevices` (page 10) class method.

Devices can provide one or more stream of a given media type. Applications can search for devices that provide media of a specific type using the `inputDevicesWithMediaType:` (page 11) and `defaultInputDeviceWithMediaType:` (page 9) class methods. Table 1 details the media types supported by `QTCaptureDevice` and examples of devices that support them:

Table 1 Media types supported by `QTCaptureDevice`

Media Type	Description	Example Devices
`QTMediaTypeVideo`	Media that only contains video frames.	iSight cameras (external and built-in); USB and FireWire webcams
`QTMediaTypeMuxed`	Multiplexed media that may contain audio, video, and other data in a single stream.	DV cameras

Media Type	Description	Example Devices
`QTMediaTypeSound`	Media that only contains audio samples.	Built-in microphones and line-in jacks; the microphone built-in to the external iSight; USB microphones and headsets; any other device supported by Core Audio.

`QTCaptureDevice` objects can have extended attributes that applications can read using the `attributeForKey:` and `deviceAttributes` methods. Some attributes, for which the `attributeIsReadOnly:` method returns NO, can be edited using the `setAttribute:forKey:` and `setDeviceAttributes:` methods. In addition to these explicit methods, applications can use key-value coding to get and set extended attributes. For an object that supports a given attribute, `valueForKey:` will be functionally identical to `attributeForKey:`, and `setValue:forKey:` will be identical to `setAttribute:forKey:`. Applications wishing to observe changes for a given attribute can add a key-value observer where the key path is the attribute key.

Tasks

Finding Devices

`+ defaultInputDeviceWithMediaType:` (page 9)
Returns a `QTCaptureDevice` instance for the default device connected to the user's system of the given media type.

`+ deviceWithUniqueID:` (page 10)
Returns a `QTCaptureDevice` instance with the identifier device UID.

`+ inputDevices` (page 10)
Returns an array of devices currently connected to the computer that can be used as input sources.

`+ inputDevicesWithMediaType:` (page 11)
Returns an array of input devices currently connected to the computer that send a stream with the given media type.

Using a Device

`- close` (page 12)
Releases application control over the device acquired in the `open:` method.

`- isConnected` (page 14)
Returns `YES` if the device is connected to the computer.

`- isInUseByAnotherApplication` (page 14)
Returns YES is the device is connected, but being exclusively used by another application.

`- open:` (page 16)
Attempts to give the application control over the device so that it can be used for capture.

`- isOpen` (page 14)
Returns `YES` if the device is open in the current application.

Getting Information About a Device

- `attributeForKey:` (page 11)
  Returns a device attribute for the given key.
- `attributeIsReadOnly:` (page 12)
  Returns whether the given attribute for the device cannot be modified.
- `deviceAttributes` (page 12)
  Returns a dictionary of the device's current attirbutes.
- `formatDescriptions` (page 13)
  Returns an array of stream formats currently in use by the device.
- `hasMediaType:` (page 13)
  Returns whether the receiver sends a stream with the given media type.
- `setAttribute:forKey:` (page 16)
  Sets a device attribute for the given key.
- `setDeviceAttributes:` (page 17)
  Sets attributes on the device from the key-value pairs in the given dictionary.
- `localizedDisplayName` (page 15)
  Returns a localized human-readable name for the receiver's device.
- `modelUniqueID` (page 15)
  Returns the unique ID of the model of the receiver's device.
- `uniqueID` (page 17)
  Returns the unique ID of the receiver's device.

Class Methods

defaultInputDeviceWithMediaType:

Returns a `QTCaptureDevice` instance for the default device connected to the user's system of the given media type.

```
+ (QTCaptureDevice *)defaultInputDeviceWithMediaType:(NSString *)mediaType
```

Parameters

mediaType

The media type, such as `QTMediaTypeVideo`, `QTMediaTypeSound`, or `QTMediaTypeMuxed`, supported by the returned device.

Return Value

The default device with the given media type on the user's system, or `NIL` if no device with that media type exists.

Discussion

This method returns the default device of the given media type connected to the user's system. For example, for `QTMediaTypeSound`, this method will return the default sound input device selected in the Sound Preference Pane. If there is no device for the given media type, this method will return nil.

Media types are defined in `QTMedia.h`.

Availability
Mac OS X v10.5 and later.

Related Sample Code
QT Capture Widget

Declared In
`QTCaptureDevice.h`

deviceWithUniqueID:

Returns a `QTCaptureDevice` instance with the identifier device UID.

```
+ (QTCaptureDevice *)deviceWithUniqueID:(NSString *)deviceUID
```

Parameters

deviceUID
: The unique identifier of the device instance to be returned.

Return Value
If a device with unique identifier `deviceUID` was connected to the computer at some point during the lifetime of the application, this method returns a `QTCaptureDevice` instance for that identifier. Otherwise, this method returns `NIL`.

Discussion
Every capture device available to the computer is assigned a unique identifier that persists on one computer across device connections and disconnections, as well as across reboots of the computer. This method can be used to recall or track the status of a specific device, even if it has been disconnected.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

inputDevices

Returns an array of devices currently connected to the computer that can be used as input sources.

```
+ (NSArray *)inputDevices
```

Return Value
An NSArray of `QTCaptureDevice` instances for each connected device. If there are no available devices, the returned array will be empty.

Discussion
This method queries the device system and builds an array of `QTCaptureDevice` instances for input devices currently connected and available for capture. The returned array contains all devices that are available when the method is called. Applications should observe `QTCaptureDeviceWasConnectedNotification` and `QTCaptureDeviceWasDisconnectedNotification` to be notified when the list of available devices has changed.

Availability
Mac OS X v10.5 and later.

Related Sample Code
LiveVideoMixer3

Declared In
`QTCaptureDevice.h`

inputDevicesWithMediaType:

Returns an array of input devices currently connected to the computer that send a stream with the given media type.

```
+ (NSArray *)inputDevicesWithMediaType:(NSString *)mediaType
```

Parameters

mediaType
: The media type, such as `QTMediaTypeVideo`, `QTMediaTypeSound`, or `QTMediaTypeMuxed`, supported by each returned device.

Return Value
An array of `QTCaptureDevice` instances for each connected device with the given media type. If there are no available devices, the returned array will be empty.

Discussion
This method queries the device system and builds an array of `QTCaptureDevice` instances for input devices that are currently connected and output streams of the given media type.

Media types are defined in `QTMedia.h`.

Availability
Mac OS X v10.5 and later.

Related Sample Code
QTRecorder

Declared In
`QTCaptureDevice.h`

Instance Methods

attributeForKey:

Returns a device attribute for the given key.

```
- (id)attributeForKey:(NSString *)attributeKey
```

Discussion
Use this method to get attributes of a device. The keys that can be used with this method are described in the Constants section. Applications using key-value coding can also get an attribute for a given key by passing that key to the NSObject `valueForKey:` method.

Availability
Mac OS X v10.5 and later.

Related Sample Code
LiveVideoMixer3

QTRecorder

Declared In
`QTCaptureDevice.h`

attributeIsReadOnly:

Returns whether the given attribute for the device cannot be modified.

```
- (BOOL)attributeIsReadOnly:(NSString *)attributeKey
```

Return Value
Returns `YES` if the attribute cannot be modified; otherwise, `NO`.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

close

Releases application control over the device acquired in the `open:` method.

```
- (void)close
```

Discussion
This method should be called to match each invocation of `open:` when an application no longer needs to use a device for capture. If a device is disconnected or turned off while it is open it will be closed automatically. Applications should check if a device has not been closed automatically by registering to receive `QTCaptureDeviceWasDisconnectedNotification` or by checking `isOpen` before manually closing the device using this method.

Applications can use key value coding with the `@"connected"` and `@"inUseByAnotherApplication"` keys to be notified of changes.

Availability
Mac OS X v10.5 and later.

Related Sample Code
QTRecorder

Declared In
`QTCaptureDevice.h`

deviceAttributes

Returns a dictionary of the device's current attirbutes.

```
- (NSDictionary *)deviceAttributes
```

Return Value
An dictionary of attributes supported by the device.

Discussion
Applications can use this method to determine what attributes a specific device supports.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

formatDescriptions

Returns an array of stream formats currently in use by the device.

```
- (NSArray *)formatDescriptions
```

Return Value
An array of `QTFormatDescription` objects describing the current stream formats of the device.

Discussion
Applications can use this method to determine what kind of media the receiver outputs. Applications can be notified of format changes by registering to receive `QTCaptureDeviceFormatDescriptionsWillChangeNotification` and `QTCaptureDeviceFormatDescriptionsDidChangeNotification` notifications or by adding a key value observer for the key `@"formatDescriptions"` .

Availability
Available in Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

hasMediaType:

Returns whether the receiver sends a stream with the given media type.

```
- (BOOL)hasMediaType:(NSString *)mediaType
```

Parameters

mediaType

A media type, such as `QTMediaTypeVideo`, `QTMediaTypeSound`, or `QTMediaTypeMuxed`.

Return Value
Returns YES if the device outputs the given media type, NO otherwise.

Discussion
Media types are defined in `QTMedia.h`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

isConnected

Returns `YES` if the device is connected to the computer.

```
- (BOOL)isConnected
```

Return Value
Returns `YES` if the device is connected and available to applications; otherwise, `NO`.

Discussion
This method checks whether the receiver's device is currently connected to the computer and available for use by applications.

Applications can use key value coding with the `@"connected"` and `@"inUseByAnotherApplication"` keys to be notified of changes.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

isInUseByAnotherApplication

Returns YES is the device is connected, but being exclusively used by another application.

```
- (BOOL)isInUseByAnotherApplication
```

Return Value
Returns YES if another process has exclusive control over a connected device; otherwise, NO.

Discussion
If the device can only be accessed by one process at a time, this method checks if the process has exclusive control over the current process.

Applications can use key value coding with the `@"connected"` and `@"inUseByAnotherApplication"` keys to be notified of changes.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

isOpen

Returns `YES` if the device is open in the current application.

```
- (BOOL)isOpen
```

Return Value
Returns `YES` if the device was previously opened by the receiver's `open:` method. Returns `NO` otherwise.

Discussion
The method checks if the device was previously succcessfully opened with the receiver's `open:` method. If this method returns `YES`, the device can be used immediately for capture.

Applications can use key value coding with the `@"connected"` and `@"inUseByAnotherApplication"` keys to be notified of changes.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

localizedDisplayName

Returns a localized human-readable name for the receiver's device.

```
- (NSString *)localizedDisplayName
```

Return Value
The localized name of the receiver's device.

Discussion
This method can be used when displaying the name of a capture device in the user interface.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

modelUniqueID

Returns the unique ID of the model of the receiver's device.

```
- (NSString *)modelUniqueID
```

Return Value
The unique identifier of the model of device corresponding to the recevier.

Discussion
The unique identifier returned by this method is unique to all devices of the same model. The value is persistent across device connections and disconnections, and across different computers.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

open:

Attempts to give the application control over the device so that it can be used for capture.

```
- (BOOL)open:(NSError **)errorPtr
```

Parameters

errorPtr

If not equal to `NIL`, points to an NSError describing why the device could not be opened, or points to `NIL` if the device was opened successfully.

Return Value

Returns `YES` if the device was opened successfully; otherwise, `NO`.

Discussion

This method attempts to open the device for control by the current application. If the device is connected and no other processes have exclusive control over it, then the application starts using the device immediately, taking exclusive control of it if necessary. Otherwise, this method returns `NO` and sets errorPtr to point to an error describing why the device could not be opened. Applications that call `open:` should also call the `close` method to relinquish access to the device when it is no longer needed. Multiple calls to this method can be nested. Each call to this method must be matched by a call to `close`. Applications that capture from a device using `QTCaptureDeviceInput` must call this method before creating the `QTCaptureDeviceInput` to be used with the device. If a device is disconnected or turned off while it is open, it will be closed automatically.

Applications can use key value coding with the `@"connected"` and `@"inUseByAnotherApplication"` keys to be notified of changes.

Availability

Available in Mac OS X v10.5 and later.

Related Sample Code

QT Capture Widget

QTRecorder

Declared In

`QTCaptureDevice.h`

setAttribute:forKey:

Sets a device attribute for the given key.

```
- (void)setAttribute:(id)attributeforKey
    :(NSString *)attributeKey
```

Discussion

Use this method to set attributes of a device. The keys that can be used with this method are described in the Constants section. This method raises an NSInvalidArgumentException if the attribute is read-only or not supported by the receiver. Applications using key value coding can also set an attribute for a given key by passing that key to the NSObject `setValue:forKey:` method.

Availability

Available in Mac OS X v10.5 and later.

Related Sample Code

QTRecorder

Declared In
`QTCaptureDevice.h`

setDeviceAttributes:

Sets attributes on the device from the key-value pairs in the given dictionary.

```
- (void)setDeviceAttributes:(NSDictionary *)deviceAttributes
```

Discussion
This method allows application to set multiple attributes on a device at once. This method raises an NSInvalidArgumentException if any of the attributes in the dictionary are read-only or not supported by the receiver. Applications using key-value coding can also set multiple attributes using the NSObject `setValuesForKeysWithDictionary:` method using attribute keys as keys in the dictionary.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

uniqueID

Returns the unique ID of the receiver's device.

```
- (NSString *)uniqueID
```

Return Value
The unique identifier of the device corresponding to the receiver.

Discussion
The unique identifier returned by this method is persistent on one computer across device connections and disconnections, as well as across reboots of the computer. It can be passed to the `deviceWithUniqueID:` class method to get the `QTCaptureDevice` instance for the device with that unique identifier.

Availability
Mac OS X v10.5 and later.

Declared In
`QTCaptureDevice.h`

Constants

Device Attributes

Constants for different device attributes.

```
NSString * const QTCaptureDeviceChangedAttributeKey;
NSString * const QTCaptureDeviceAvailableInputSourcesAttribute;
NSString * const QTCaptureDeviceInputSourceIdentifierAttribute;
NSString * const QTCaptureDeviceInputSourceIdentifierKey;
NSString * const QTCaptureDeviceInputSourceLocalizedDisplayNameKey;
NSString * const QTCaptureDeviceSuspendedAttribute;
NSString * const QTCaptureDeviceLinkedDevicesAttribute;
NSString * const QTCaptureDeviceLegacySequenceGrabberAttribute;
NSString * const QTCaptureDeviceAVCTransportControlsAttribute;
NSString * const QTCaptureDeviceAVCTransportControlsSpeedKey;
NSString * const QTCaptureDeviceAVCTransportControlsPlaybackModeKey;
```

Constants

`QTCaptureDeviceChangedAttributeKey`

Indicates the key of the attribute that changed. Used as a key in the userInfo dictionary passed to `QTCaptureDeviceAttributeWillChangeNotification`, and `QTCaptureDeviceAttributeDidChangeNotification` to indicate the key of the attribute that changed.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAvailableInputSourcesAttribute`

For devices with multiple possible input sources, returns an array of dictionaries describing each available input source. Some devices can capture data from one of multiple input sources (different input jacks on the same audio device, for example). The value is an `NSArray` of `NSDictionary` objects. The keys in each dictionary are described in Input Source Dictionary Keys. This string value can be used in key paths for key value coding, key value observing, and bindings.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceInputSourceIdentifierAttribute`

Used to get and set the currently used input source for the device. Some devices can capture data from one of multiple input sources (different input jacks on the same audio device, for example). The value is an object returned by the `QTCaptureDeviceInputSourceIdentifierKey` key in one of the dictionaries returned by `QTCaptureDeviceAvailableInputSourcesAttribute`. This string value can be used in key paths for key value coding, key value observing, and bindings.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceInputSourceIdentifierKey`

An object representing a unique ID for the input source. This ID is not guaranteed to persist between device connections or changes in device configuration. To set the input source for a device, set `QTCaptureDeviceInputSourceIdentifierAttribute` to the value returned by this key. This string value can be used in key paths for key value coding, key value observing, and bindings.

This key, along with the `QTCaptureDeviceInputSourceLocalizedDisplayNameKey` key, comprises the NSDictionary objects describing input sources returned by `QTCaptureDeviceAvailableInputSourcesAttribute`.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceInputSourceLocalizedDisplayNameKey`

The localized display name of an input source, suitable for display in a user interface. This string value can be used in key paths for key value coding, key value observing, and bindings.

This key, along with the `QTCaptureDeviceInputSourceIdentifierKey` key, comprises the NSDictionary objects describing input sources returned by `QTCaptureDeviceAvailableInputSourcesAttribute`.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceSuspendedAttribute`

Returns whether or not data capture on the device is suspended due to a feature on the device. For example, this attribute is `YES` for the external iSight when its privacy iris is closed, or for the internal iSight on a notebook when the notebook's display is closed.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceLinkedDevicesAttribute`

Returns an array of `QTCaptureDevice` objects that, although they are separate devices on the system, are a part of the same physical device as the receiver. For example, for the external iSight camera, this attribute returns an array containing a `QTCaptureDevice` for the external iSight microphone.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceLegacySequenceGrabberAttribute`

An NSValue interpreted as a ComponentInstance for the legacy sequence grabber component used by the device. Some older devices are opened and conreolled by legacy Sequence Grabber components. Applications that need to configure legacy devices directly through the Sequence Grabber configuration dialog can access an open component instance with this attribute.

This string value can be used in key paths for key-value coding, key-value observing, and bindings.

If the device is being used in a capture session, do not modify properties of the returned Sequence Grabber component (by displaying the configuration dialog, for example) while the session is running. Doing so will prevent the capture session from capturing more frames.

Available in Mac OS X v10.5 and later.

Not available to 64-bit applications.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsAttribute`

For AVC devices that read data from linear media, such as tapes, specifies the mode and speed at which that media is playing.

The value is an NSDictionary with keys and values described under `QTCaptureDevice` AVC Transport Controls.

This string value can be used in key paths for key-value coding, key-value observing, and bindings.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsSpeedKey`
Specifies the approximate rate at which the device runs through linear media. The value is an NSNumber interpreted as a `QTCaptureDeviceAVCTransportControlsSpeed`. This is one of the keys that comprise the NSDictionary that specifies the linear media playback mode and rate given by the `QTCaptureDeviceAVCTransportControlsAttribute`.

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsPlaybackModeKey`
A value provided with the `QTCaptureDeviceAVCTransportControlsPlaybackModeKey` key that specifies whether the device previews audio and displays video while it is running through linear media. `QTCaptureDeviceAVCTransportControlsNotPlayingMode` is equivalent to the Play mode on most cameras and tape decks, while `QTCaptureDeviceAVCTransportControlsPlayingMode` is equivalent to Stop on most cameras and tape decks. If the device is connected to a session, the video at the current location on the device's media will only be captured if this attribute is set to `QTCaptureDeviceAVCTransportControlsNotPlayingMode`.

```
enum {
    QTCaptureDeviceAVCTransportControlsNotPlayingMode       = 0,
    QTCaptureDeviceAVCTransportControlsPlayingMode          = 1
};
```

Available in Mac OS X v10.5 and later.

Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsSpeed`
A value provided with the `QTCaptureDeviceAVCTransportControlsSpeedKey` key that specifies whether the device previews audio and displays video while it is running through linear media. The actual speed at which the media is run for a given value will depend on the manufacturer and model of the device, as well as the value of `QTCaptureDeviceAVCTransportControlsPlaybackModeKey` (in general, when `QTCaptureDeviceAVCTransportControlsPlaybackModeKey` is set to `QTCaptureDeviceAVCTransportControlsNotPlayingMode`, the media will run faster than when it is set to `QTCaptureDeviceAVCTransportControlsPlayingMode`).

Enumunerations

These are the values for the dictionary passed to `QTCaptureDeviceAVCTransportControlsAttribute`. For most cameras and tape decks, different speeds will affect the media speed.

```
enum {
    QTCaptureDeviceAVCTransportControlsFastestReverseSpeed  = -19000,
    QTCaptureDeviceAVCTransportControlsVeryFastReverseSpeed = -16000,
    QTCaptureDeviceAVCTransportControlsFastReverseSpeed     = -13000,
    QTCaptureDeviceAVCTransportControlsNormalReverseSpeed   = -10000,
    QTCaptureDeviceAVCTransportControlsSlowReverseSpeed     = -7000,
    QTCaptureDeviceAVCTransportControlsVerySlowReverseSpeed = -4000,
    QTCaptureDeviceAVCTransportControlsSlowestReverseSpeed  = -1000,
    QTCaptureDeviceAVCTransportControlsStoppedSpeed         = 0,
    QTCaptureDeviceAVCTransportControlsSlowestForwardSpeed  = 1000,
    QTCaptureDeviceAVCTransportControlsVerySlowForwardSpeed = 4000,
    QTCaptureDeviceAVCTransportControlsSlowForwardSpeed     = 7000,
    QTCaptureDeviceAVCTransportControlsNormalForwardSpeed   = 10000,
    QTCaptureDeviceAVCTransportControlsFastForwardSpeed     = 13000,
    QTCaptureDeviceAVCTransportControlsVeryFastForwardSpeed = 16000,
    QTCaptureDeviceAVCTransportControlsFastestForwardSpeed  = 19000,
};
```

Constants

`QTCaptureDeviceAVCTransportControlsFastestReverseSpeed`
: Media runs in reverse at greater than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsVeryFastReverseSpeed`
: Media runs in reverse at greater than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsFastReverseSpeed`
: Media runs in reverse at greater than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsNormalReverseSpeed`
: Media runs in reverse at normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsSlowReverseSpeed`
: Media runs in reverse at less than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsVerySlowReverseSpeed`
: Media runs in reverse at less than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsSlowestReverseSpeed`
: Media runs in reverse at less than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsStoppedSpeed`
: Media is paused.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsSlowestForwardSpeed`
: Media runs forward at less than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsVerySlowForwardSpeed`
: Media runs forward at less than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsSlowForwardSpeed`
: Media runs forward at less than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsNormalForwardSpeed`
: Media runs forward at normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsFastForwardSpeed`
: Media runs forward at greater than than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsVeryFastForwardSpeed`
: Media runs forward at greater than than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

`QTCaptureDeviceAVCTransportControlsFastestForwardSpeed`
: Media runs forward at greater than than normal speed.

  Available in Mac OS X v10.5 and later.

  Declared in `QTCaptureDevice.h`.

Notifications

QTCaptureDeviceWasConnectedNotification

Posted when a device is connected or turned on.

Availability
QuickTime 7.2.1 and later

Declared In
`QTCaptureDevice.h`

QTCaptureDeviceWasDisconnectedNotification

Posted when a device is disconnected or turned off.

Availability
QuickTime 7.2.1 and later

Declared In
`QTCaptureDevice.h`

QTCaptureDeviceFormatDescriptionsWillChangeNotification

Posted when the device's formats that are returned by the `formatDescriptions` method are about to change.

Availability
QuickTime 7.2.1 and later

Declared In
`QTCaptureDevice.h`

QTCaptureDeviceFormatDescriptionsDidChangeNotification

Posted when the device’s formats that are returned by the `formatDescriptions` method have just changed.

Availability
QuickTime 7.2.1 and later

Declared In
`QTCaptureDevice.h`

QTCaptureDeviceAttributeWillChangeNotification

Posted when one of the device's attributes is about to change.

The notification’s user info dictionary will contain the attribute key of the changed attribute for the key `QTCaptureDeviceChangedAttributeKey`.

Availability
QuickTime 7.2.1 and later

Declared In
`QTCaptureDevice.h`

QTCaptureDeviceAttributeDidChangeNotification

Posted when the one of device’s attributes has changed.

The notification’s user info dictionary will contain the attribute key of the changed attribute for the key `QTCaptureDeviceChangedAttributeKey`.

Availability
QuickTime 7.2.1 and later

Declared In
`QTCaptureDevice.h`

Document Revision History

This table describes the changes to *QTCaptureDevice Class Reference*.

Date	Notes
2009-05-06	Fixed enumeration and constant listings. Minor fixes.
2009-04-08	Fixed constant listings. Corrected attribute descriptions. Minor fixes.
2009-03-04	Added availability information and media type definitions; added text on using key value coding with certain attributes; minor edit and linking fixes.
2007-07-22	Updated information on behavior of open: and close: methods; added new modelUniqueID: method; removed device control attributes.
2006-08-07	New document that describes the Objective-C class for an available capture device.

Index

A

attributeForKey: instance method 11
attributeIsReadOnly: instance method 12

C

close instance method 12

D

defaultInputDeviceWithMediaType: class method 9
Device Attributes 17
deviceAttributes instance method 12
deviceWithUniqueID: class method 10

E

Enumunerations 20

F

formatDescriptions instance method 13

H

hasMediaType: instance method 13

I

inputDevices class method 10
inputDevicesWithMediaType: class method 11
isConnected instance method 14
isInUseByAnotherApplication instance method 14
isOpen instance method 14

L

localizedDisplayName instance method 15

M

modelUniqueID instance method 15

O

open: instance method 16

Q

QTCaptureDeviceAttributeDidChangeNotification notification 23
QTCaptureDeviceAttributeWillChangeNotification notification 23
QTCaptureDeviceAvailableInputSourcesAttribute constant 18
QTCaptureDeviceAVCTransportControlsAttribute constant 19
QTCaptureDeviceAVCTransportControlsFastestForward-Speed constant 22
QTCaptureDeviceAVCTransportControlsFastestReverse-Speed constant 21
QTCaptureDeviceAVCTransportControlsFastForward-Speed constant 22
QTCaptureDeviceAVCTransportControlsFastReverse-Speed constant 21

QTCaptureDeviceAVCTransportControlsNormalForwardSpeed constant 22
QTCaptureDeviceAVCTransportControlsNormalReverseSpeed constant 21
QTCaptureDeviceAVCTransportControlsPlaybackModeKey constant 20
QTCaptureDeviceAVCTransportControlsSlowestForwardSpeed constant 22
QTCaptureDeviceAVCTransportControlsSlowestReverseSpeed constant 21
QTCaptureDeviceAVCTransportControlsSlowForwardSpeed constant 22
QTCaptureDeviceAVCTransportControlsSlowReverseSpeed constant 21
QTCaptureDeviceAVCTransportControlsSpeed constant 20
QTCaptureDeviceAVCTransportControlsSpeedKey constant 20
QTCaptureDeviceAVCTransportControlsStoppedSpeed constant 22
QTCaptureDeviceAVCTransportControlsVeryFastForwardSpeed constant 22
QTCaptureDeviceAVCTransportControlsVeryFastReverseSpeed constant 21
QTCaptureDeviceAVCTransportControlsVerySlowForwardSpeed constant 22
QTCaptureDeviceAVCTransportControlsVerySlowReverseSpeed constant 21
QTCaptureDeviceChangedAttributeKey constant 18
QTCaptureDeviceFormatDescriptionsDidChangeNotification notification 23
QTCaptureDeviceFormatDescriptionsWillChangeNotification notification 23
QTCaptureDeviceInputSourceIdentifierAttribute constant 18
QTCaptureDeviceInputSourceIdentifierKey constant 18
QTCaptureDeviceInputSourceLocalizedDisplayNameKey constant 19
QTCaptureDeviceLegacySequenceGrabberAttribute constant 19
QTCaptureDeviceLinkedDevicesAttribute constant 19
QTCaptureDeviceSuspendedAttribute constant 19
QTCaptureDeviceWasConnectedNotification notification 22
QTCaptureDeviceWasDisconnectedNotification notification 23

S

setAttribute:forKey: instance method 16
setDeviceAttributes: instance method 17

U

uniqueID instance method 17