QuickTime® VR
Macromedia Director Xtra API v1.0

Introduction

This document provides a reference guide to QuickTime VR's Director Xtra, which provides a Lingo level API to QuickTime VR 1.0 movies on Mac™ OS and Windows™ computers. This document explains how the Xtra can be used to display movies in Macromedia Director™. Once you have read this manual and examined the QTVR Xtra Testbed, you will be able to create Director movies which make extensive use of QuickTime VR movies.

The QuickTime VR Xtra handles both QuickTime VR panoramic and object movies. Note that the commands and properties can apply to either or both types of movies. After loading the Xtra with the Lingo `xtra` command, use the Enter command to initialize the QuickTime VR Xtra. Use the Lingo `new` command to create an instance of the Xtra. Look at the Open command to see how a QuickTime VR movie is opened. Use GetQTVRType if you need to determine the type of movie it is. When you are finished with the movie, use the Close command and when you are finished using the Xtra use the Exit command.

element name

Type of element: COMMAND or PROPERTY

DESCRIPTION

A high-level description of the element's function.

SYNTAX

A summary of the element's syntax. Functions are generally explained using a `put` statement, but can be used in any expression in Lingo.

EXAMPLES

One or more examples of the key uses of this element.

Most examples assume that a panoramic VR movie is open, and that there is a global variable `gQTVRinstance` which contains the reference to the Xtra's instance The Open page describes how this global variable is created.

CAUTIONS

A list of the most frequent problems encountered or mistakes made when using this element, along with suggested fixes.

KNOWN BUGS

A list of known bugs relating to this element.

SEE ALSO

A list of other elements which relate to this element.

ClickLoc

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets a program-controlled mouse click location in window coordinates. Gets the last location at which the mouse was clicked (under program or user control) in window coordinates.

The clickLoc property's values are measured within the window's coordinate system. The ClickLoc property provides a display-centric method for programmatically selecting a hot spot.

Setting the CickLoc to a valid hot spot's location sets up the ClickPanAngles, ClickPanLoc, HotSpotID, HotSpotName, HotSpotType, HotSpotViewAngles, ObjectViewAngles and ObjectZoomRect properties.

SYNTAX

```
QTVRSetClickLoc(<QTVRInstance>, <clickLoc>)
```

`<clickLoc>` is a string containing a point (two comma-separated values) measured in the panoramic window's coordinate system. If `<clickLoc>` is outside the panoramic window bounds, behavior is unpredictable.

```
put QTVRGetClickLoc(<QTVRInstance>) into tClickLoc
```

Returns a string containing the last point (measured in the panoramic window's coordinate system) at which a hot spot was selected.

EXAMPLES

To display the last mouse click location:

```
put QTVRGetClickLoc(gQTVRInstance) into tClickLoc
put "Last mouse click at:" && tClickLoc
```

To set a mouse click location:

```
QTVRSetClickLoc(gQTVRInstance, tclickLoc)
```

CAUTIONS

The ClickLoc property is only reset after a hot spot is selected interactively or the ClickLoc is set programmatically.

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickPanAngles
ClickPanLoc
HotSpotID
HotSpotName
HotSpotType
HotSpotViewAngles
MouseDown
MouseOver
ObjectViewAngles
ObjectZoomRect

ClickPanAngles

Pano Only

GET PROPERTY

DESCRIPTION

Gets the last location at which the mouse was clicked (under program or user control) in panoramic angle coordinates.

The ClickPanAngles property's values are measured within the panorama itself, given the rotated orientation in which the panorama is stored.

SYNTAX

```
put QTVRGetClickPanAngles(<QTVRInstance>) into tClickPanAngles
```

Returns a string containing a pair of angles (two comma-separated values) measured in the panorama's angular coordinate system:

```
<PanAngle>,<TiltAngle>
```

EXAMPLES

To display the last mouse click location in angular coordinates:

```
put QTVRGetClickPanAngles(gQTVRInstance) into tClickPanAngles
put "Last mouse click in the panorama at angles:" && tClickPanAngles
```

CAUTIONS

The ClickPanAngles property is only reset after a hot spot is selected interactively or the ClickLoc is set programmatically.

SEE ALSO

ClickLoc	PanAngle
ClickPanLoc	TiltAngle

ClickPanLoc

Pano Only

GET PROPERTY

DESCRIPTION

Gets the last location at which the mouse was clicked (under program or user control) in panoramic pixel coordinates.

The ClickPanLoc property's values are measured within the panorama itself, given the rotated orientation in which the panorama is stored.

SYNTAX

```
put QTVRGetClickPanLoc(<QTVRInstance>) into tClickPanLoc
```

Returns a string containing a point (two comma-separated values) measured in the panorama's coordinate system. Note that panoramas are rotated 90° counter-clockwise relative to a horizontal orientation, and that `<clickPanLoc>` is measured in this rotated coordinate system.

EXAMPLES

To display the last mouse click location in panoramic coordinates:

```
put QTVRGetClickPanLoc(gQTVRInstance) into tClickPanLoc
put "Last mouse click in the panorama at:" && tClickPanLoc
```

CAUTIONS

The clickPanLoc property is only reset after a hot spot is selected interactively or the clickLoc is set programmatically.

SEE ALSO

ClickLoc ClickPanAngles

Close

COMMAND

DESCRIPTION

Closes a QuickTime VR movie. This command must be performed before the Lingo `forget` command is called for a given instance.

SYNTAX

```
QTVRClose(<QTVRInstance>)
```

EXAMPLES

To close a movie :

```
QTVRClose(gQTVInstance)
set gQTVRInstance = 0
```

CAUTIONS

After calling QTVRClose, you should use the Lingo `forget` command to remove the QTVR instance. Alternately, you can call QTVROpen to reuse the instance for another QTVR movie. You should not make any more QTVR Xtra calls with an instance after calling QTVRClose.

SEE ALSO

Enter Exit Open

Column

Object Only

SET/GET PROPERTY

DESCRIPTION

Sets the column number for the next view. Gets the column number to be used for the next view.

If the column has not been changed (directly or indirectly) since the last update, the Column property will return the column displayed in the last view.

Normally you should use PanAngle and TiltAngle to specify the view of an object. The object controller would then display the closest captured view matching the pan and tilt angles. However, you may use the Row and Column properties to explicitly specify which captured view should be displayed. The Column property specifies the horizontal position of the camera during capture.

The column value should be in the range 0 to (number of horizontal positions -1):
0 specifies the first column, 1 the second column, etc. The value is clipped if it is out of range.

SYNTAX

```
QTVRSetColumn(<QTVRInstance>, <columnNumber>)
```

`<columnNumber>` is an integer number.

```
put QTVRGetColumn(<QTVRInstance>) into tColumnNumber
```

Returns a string containing an integer column number.

EXAMPLES

To change the view of an object:

```
QTVRSetColumn(gQTVRInstance, 5)
```

To display the current column number:

```
put QTVRGetColumn(gQTVRInstance) into tColumnNumber
put "Current column number is:" && tColumnNumber
```

CAUTIONS

The Row and Column properties are not as general as the PanAngle and TiltAngle properties since you must have explicit knowledge of how the object was captured. Use the Column property only when necessary.

SEE ALSO

PanAngle Row

TiltAngle Update

Enter

COMMAND

DESCRIPTION

Initializes the QuickTime VR Xtra. This must be called after the Xtra is loaded using the Lingo `xtra` command, and before creating any instances of the Xtra. There should only be one call to QTVREnter after you've loaded the QTVR Xtra. You must call QTVRExit when you are finished with using the QTVR Xtra.

SYNTAX

```
QTVREnter(<QTVRXtra>)
```

EXAMPLES

```
-- Load the xtra
put xtra "QTVRXtra" into gQTVRXtra
-- Enter
QTVREnter(gQTVRXtra)
-- Create the first Instance
put new(gQTVRXtra) into gQTVRInstance
```

Please see the StartMovie script in the Xtra Testbed for a complete example of the use of these commands.

CAUTIONS

This call must be made in the appropriate order as described above. If it is not, unpredictable results may occur.

SEE ALSO

Close Exit Open

Exit

COMMAND

DESCRIPTION

De-initializes the QuickTime VR Xtra. Exit must be called after disposing of any instances of the Xtra and before the movie finishes. You must call Exit when you are done with the QTVR Xtra.

SYNTAX

```
QTVRExit(<QTVRXtra>)
```

EXAMPLES

```
-- Destroy any instances we've created
forget (gQTVRInstance)
-- Exit
QTVRExit(gQTVRXtra)
```

Please see the StopMovie script in the Xtra Testbed for a complete example of the use of this command.

CAUTIONS

This call must be made in the appropriate order as described above. If it is not, unpredictable results may occur.

SEE ALSO

Close Enter Open

ExitMouseOver

Pano Only

COMMAND

DESCRIPTION

Used during a MouseOverHandler or RolloverHotSpotHandler callback, instructs QuickTime VR to exit from the MouseOver routine immediately. When control returns from Director to QuickTime VR at the end of the handler execution, QuickTime VR immediately finishes the MouseOver call and returns control back to Director.

ExitMouseOver has no effect if it is called at times other than during a MouseOverHandler or RolloverHotSpotHandler callback.

SYNTAX

```
QTVRExitMouseOver(<QTVRInstance>)
```

EXAMPLES

To exit QuickTime VR's MouseOver routine whenever the user rolls over a particular hot spot:

```
on RolloverHotSpotHandler pHotSpotID
    global gQTVRInstance, gSpecialHotSpotID
    if pHotSpotID = gSpecialHotSpotID then
        QTVRExitMouseOver(gQTVRInstance)
    end if
end rolloverHotSpotHandler
```

SEE ALSO

MouseOver MouseOverHandler RolloverHotSpotHandler

FOV

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the field of view to be used when next computing an image. Gets the FOV to be used when next computing an image.

If the FOV has not been changed (directly or indirectly) since the last update, the FOV property will return the FOV used to compute the last image.

The FOV , PanAngle, TiltAngle, and NodeID properties completely describe a view within a particular panoramic movie.

The FOV value is measured in degrees which increase as the view becomes more wide angle, and decrease as the view zooms in more tightly. The field of view measures the angle between a line from the viewpoint to the top of the image, and a line from the viewpoint to the bottom of the image. Values which exceed the node's FOV constraint angles are clipped. Pan and tilt angles are clipped in order to show images which satisfy the node's constraints at the specified FOV .

SYNTAX

```
QTVRSetFOV(<QTVRInstance>, <angle>)
```

<angle> is a string containing an angle measured in floating point degrees.

```
put QTVRGetFOV(<QTVRInstance) into tFOV
```

Returns a string containing an angle measured in floating point degrees.

EXAMPLES

To zoom to a different view within the node:

```
QTVRSetFOV(gQTVRInstance, "45.0")
QTVRUpdate(gQTVRInstance)
```

To display the current zoom angle:

```
put QTVRGetFOV(gQTVRInstance) into tFOV
put "Current zoom angle is:" && tFOV
```

CAUTIONS

When setting up a new view for display, it is important to realize that the FOV property is itself clipped as well as clipping the PanAngle and TiltAngle properties at the time it is set, not at the time it is imaged. Consequently, it is important to set the FOV, PanAngle and TiltAngle properties in the correct order:

```
QTVRSetFOV(gQTVRInstance, tFOV)
QTVRSetTiltAngle(gQTVRInstance, tTiltAngle)
QTVRSetPanAngle(gQTVRInstance, tPanAngle)
```

If the FOV property is changing, it should be changed first.

SEE ALSO

NodeID PanAngle TiltAngle

HotSpotID

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets a program-controlled hot spot selection. Gets the last hot spot selection made (under program or user control).

Setting the HotSpotID to a valid hot spot ID sets up the HotSpotName, HotSpotType, and HotSpotViewAngles properties.

SYNTAX

```
QTVRSetHotSpotID(<QTVRInstance>, <hotSpotID>)
```

`<hotSpotID>` is a value containing a positive integer between 1 and 255. If a hot spot with ID `<hotSpotID>` does not exist in the current node, does nothing.

```
put QTVRGetHotSpotID(<QTVRInstance>) into tHotSpotID
```

Returns a string containing the ID of the last hot spot selected. If a hot spot is not selected, returns the string `"0"`.

EXAMPLES

To display the last hot spot selected:

```
put QTVRGetHotSpotID(gQTVRInstance) into tHotSpotID
if tHotSpotID <> 0 then
    put "Last hot spot selected was:" && tHotSpotID
else
    put "No hot spot currently selected."
end if
```

To select a particular hot spot:

```
-- Assumes hot spot ID 10 exists in the current node
QTVRSetHotSpotID(gQTVRInstance, 10)
```

CAUTIONS

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickLoc
HotSpotName
HotSpotType
HotSpotViewAngles
MouseDown
MouseOver
ObjectViewAngles
ObjectZoomRect

HotSpotName

Pano Only

GET PROPERTY

DESCRIPTION

Gets the name of the selected hot spot.

The HotSpotName can be used as the filename of a matching VR object movie or still. Alternatively, the HotSpotName can be used as a descriptive field for display to the user.

SYNTAX

```
put QTVRGetHotSpotName(<QTVRInstance>) into tHotSpotName
```

Returns a string containing the name of the last hot spot selected. If no name was provided for the selected hot spot at authoring time or if a hot spot is not selected, the string will be empty.

EXAMPLES

To display the name of the last hot spot selected:

```
-- Assume that there is a hot spot selected
put QTVRGetHotSpotName(gQTVRInstance) into ¬
    tHotSpotName
if tHotSpotName <> empty then
    put "Name of last hot spot selected was:" && tHotSpotName
else
    put "Selected hot spot is unnamed."
end if
```

CAUTIONS

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickLoc
HotSpotID
MouseDown
MouseOver

HotSpotType

Pano Only

GET PROPERTY

DESCRIPTION

Gets the hot spot type of the selected hot spot.

SYNTAX

```
put QTVRGetHotSpotType(<QTVRInstance>) into tHotSpotType
```

Returns a string containing a four letter code containing the type of the selected hot spot. Valid hot spot types include:

link	if a link hot spot was selected
navg	if a navigable object hot spot was selected
stil	if a still hot spot was selected
misc	if a miscellaneous hot spot was selected
undf	if an undefined hot spot was selected

Note that link hot spots can only be selected programmatically. Other hot spot types specified at authoring time can also be returned.

If a hot spot is not selected, the returned string will be `"undf"`.

EXAMPLES

To display the type of the last hot spot selected:

```
-- Assume a hot spot is selected
put QTVRGetHotSpotType(gQTVRInstance) into tHotSpotType
put "Type of last hot spot selected was:" && tHotSpotType
```

CAUTIONS

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickLoc	HotSpotID
MouseDown	MouseOver

HotSpotViewAngles

Pano Only

GET PROPERTY

DESCRIPTION

Gets the poster view of the selected hot spot within the current node.

The HotSpotViewAngles property returns a string containing a triplet of comma-separated angles: the horizontal pan angle, the vertical pan angle and the zoom angle. These values can be used to set the panoramic view to display the poster view for the selected hot spot.

SYNTAX

```
put QTVRGetHotSpotViewAngles(<QTVRInstance>) into tHotSpotViewAngles
```

Returns a string containing a triplet of comma-separated angles:

```
<PanAngle>,<TiltAngle>,<FOV>
```

If a hot spot is not selected, the returned string will be empty.

EXAMPLES

To show the best view in the current node of the last hot spot selected:

```
put QTVRGetHotSpotViewAngles(gQTVRInstance) into ¬
    tHotSpotViewAngles
if tHotSpotViewAngles <> empty then
    put item 1 of tHotSpotViewAngles into tPanAngle
    put item 2 of tHotSpotViewAngles into tTiltAngle
    put item 3 of tHotSpotViewAngles into tFOV
    QTVRSetFOV(gQTVRInstance, tFOV)
    QTVRSetTiltAngle(gQTVRInstance, tTiltAngle)
    QTVRSetPanAngle(gQTVRInstance, tPanAngle)
    QTVRUpdate(gQTVRInstance)
else
    put "No hot spot currently selected."
end if
```

CAUTIONS

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickLoc HotSpotID FOV

MouseDown MouseOver PanAngle

TiltAngle

Idle

COMMAND

DESCRIPTION

Provides idle time to the panoramic engine. The engine uses idle time to provide hot spot cursor feedback and handle mouseless zooming if the mouse is within a panoramic movie, to display a high quality update if required (when the quality property is set to 0), and to perform paging to preload the image cache if memory permits. Idle returns after one iteration, and must be called repeatedly to manage hot spot cursor feedback and handle mouseless zooming properly. MouseOver is the preferred routine for this functionality.

In panoramic movies, if the mouse is located outside the movie when a panning, tilting or FOV action finishes, Idle should be used to allow the QTVR engine to display a high-quality update if the quality property is set to 0.

In object movies, Idle provides idle time to the QTVR object controller. Currently you only need to call Idle if you have an animated object movie where each view has a set of frames that should be cycled through while the user is not interacting with the object. You should call idle only if the mouse is not over the object movie. If the mouse is over the movie, you should call MouseOver instead.

SYNTAX

```
QTVRIdle(<QTVRInstance>)
```

EXAMPLES

To send an idle to the panoramic engine:

```
QTVRIdle(gQTVRInstance)
```

To use idle to perform a high-quality update after a panning operation at quality 0, see the sample frame script in the Xtra Testbed.

CAUTIONS

The cursor command in Director only changes the cursor when Director itself thinks that the currently displayed cursor is different from the one the script is requesting. Director is not aware of QuickTime VR's cursor changes, and cursor commands designed to reset the cursor when it has moved outside the panoramic movie may be ignored. It may be necessary to change the cursor twice in order to effect the desired cursor change.

SEE ALSO

MouseOver

Quality

IsQTVRMovie

GET PROPERTY

DESCRIPTION

Determines whether given instance contains a valid, open QTVR movie that is ready for display.

SYNTAX

```
put ISQTVRMovie(<QTVRInstance>) into tValidMovie
```

Returns a value containing a 1 if the movie is Valid and 0 if it is not.

EXAMPLES

```
if ISQTVRMovie(gQTVRInstance) <> 0 then
    --perform some action on the instance
else
    put "A valid movie is not currently open."
end if
```

SEE ALSO

QTVRType

MouseDown

COMMAND

DESCRIPTION

If the movie is an object movie, MouseDown passes control to the object controller following a mouse-down event detected in Director. An optional parameter specifies the cursor location at mouse-down time. The controller retains control until mouse-up, at which time control returns to Director.

Normally you would call mouseOver when the cursor is inside the object movie, and mouse-down handling would be done automatically for you. Use the mouseDown command for the special case where you were not able to call mouseOver.

If the movie is a panoramic movie, MouseDown passes control to the panoramic engine following a mouse-down event detected in Director. An optional parameter specifies the cursor location at mouse-down time.

The engine retains control until mouse-up, at which time control returns to Director. Control returns before a high-quality update is performed (if the quality property is set to 0). MouseDown returns a string describing the user action during the mouse-down period.

If the user clicked on a hot spot during the mouseDown command, that hot spot is selected.

The MouseStillDownHandler, NodeLeaveHandler, and PanZoomStartHandler properties can be used to provide time to Director while the mouse is down. The MouseDown command is not needed if the MouseOver command is being used.

SYNTAX

```
QTVRMouseDown(<QTVRInstance>)
```

If the movie is an object movie, returns nothing.

If the movie is a panoramic movie, returns a one- or two-token string describing the user's actions during this mouse-down event:

`pan ,0`	if panning or zooming
`jump,<newNodeID>`	if the user moved between nodes
`navg,<hotSpotID>`	if a navigable object hot spot was selected
`stil,<hotSpotID>`	if a still hot spot was selected
`misc,<hotSpotID>`	if a miscellaneous hot spot was selected
`<type>,<hotSpotID>`	if a hot spot of developer-defined type was selected
`undf,<hotSpotID>`	if an undefined hot spot was selected

Note that the first token of the pan/zoom result is "pan " with a space at the end.

EXAMPLES

After detecting a mouse-down event, pass it to QuickTime VR and handle the return string:

```
put QTVRMouseDown(gQTVRInstance) into tMouseDownAction
```

CAUTIONS

If the movie is a panormic movie, after a panning operation with the quality property set to 0, MouseDown returns before a high-quality update is performed. QuickTime VR must be given time, either through an idle call, or through a mouseOver call, in order to perform the required high-quality update. The sample frame script in the QTVRXtra Testbed provides a fully fleshed out example of how this can be achieved.

The cursor command in Director only changes the cursor when Director itself thinks that the currently displayed cursor is different from the one the script is requesting. Director is not aware of QuickTime VR's cursor changes, and cursor commands designed to reset the cursor when it has moved outside the panoramic movie may be ignored. It may be necessary to change the cursor twice in order to effect the desired cursor change.

SEE ALSO

MouseOver MouseStillDownHandler

NodeLeaveHandler PanZoomStartHandler

MouseDownHandler

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the name of the Lingo handler to be called when the mouse is pressed during MouseOver. Gets the name of the Lingo handler called when the mouse is pressed during MouseOver.

The MouseDownHandler is only called while a call to MouseOver is in progress. It is not called when the MouseDown command is used. If a MouseDownHandler is set, the default assumption is that the handler will manage the entire mouse-down event, and QuickTime VR's internal MouseDown code is not executed after the MouseDownHandler returns. If QuickTime VR's internal MouseDown code needs to be executed, a PassMouseDown command must be sent to QuickTime VR before the MouseDownHandler returns.

If the MouseDownHandler property value is set to empty, the current callback is deactivated.

SYNTAX

```
QTVRSetMouseDownHandler(<QTVRInstance>, <handlerName>)
```

`<handlerName>` is a string containing a Lingo handler name.

```
put QTVGetMouseDownHandler(<QTVRInstance>) into tMouseDownHandler
```

Returns a string containing a Lingo handler name. Returns empty if no callback has been set.

EXAMPLES

To set a handler which plays a sound on mouse-down during mouseOver:

```
QTVRSetMouseDownHandler(gQTVRInstance, "BeepOnClick")
...
on BeepOnClick pMouseDownLoc
    global gQTVRInstance
    -- pMouseDownLoc is the location of the mouse click
    -- BeepOnClick is called whenever the mouse button is
    -- is pressed during a mouseOver call
    beep
    QTVRPassMouseDown(gQTVRInstance)
end BeepOnClick
```

To deactivate the callback:

```
QTVRSetMouseDownHandler(gQTVRInstance, empty)
```

CAUTIONS

If PassMouseDown is not called within a MouseDownHandler, QuickTime VR assumes that the MouseDownHandler has managed the mouse-down event, and does not handle it itself.

On the Macintosh, the XCMD Callback Factory must be installed in order for the MouseDownHandler callback to work properly.

SEE ALSO

MouseOverHandler
MouseStillDownHandler
PanZoomStartHandler
PassMouseDown
RolloverHotSpotHandler

MouseOver

Pano Only

COMMAND

DESCRIPTION

If the movie is an object movie, MouseOver passes control to the object controller when the cursor is within the object movie. Control returns to Director when the cursor leaves the object movie, or when a Toolbox event not recognized by the controller is received.

MouseOver provides cursor feedback when the mouse is over the object movie. Mouse-down events are handled automatically to manipulate the object. Arrow keys events are handled to change the view of the object. A "looping" movie continues to animate during MouseOver.

If the movie is a panoramic movie, MouseOver passes control to the panoramic engine while the cursor remains within the panoramic movie or until a mouse-down action is completed. Control returns to Director when the cursor leaves the panoramic movie or when the mouse is released.

MouseOver provides hot spot cursor feedback, supports mouseless zooming, executes a high-quality update if required (when the quality property is set to 0), and performs paging to preload the image cache. If a mouse-down event within the panoramic movie occurs during MouseOver, the event is processed automatically, with control returning to Director upon mouse-up. MouseOver returns a string describing the user action during the mouse-over and possible mouse-down period.

MouseOver functions by repeatedly sending idle events while the cursor remains within the panoramic movie, and sending a mouse-down event if necessary.

If the user clicked on a hot spot during the mouse-down portion of a MouseOver call, that hot spot is selected.

The MouseOverHandler and RolloverHotSpotHandler properties can be used to provide time to Director during MouseOver. The ExitMouseOver command can be used to instruct MouseOver to return control to Director immediately. The MouseDownHandler property can be used to notify and provide time to Director when a mouse-down occurs during MouseOver. The MouseDownHandler must send a PassMouseDown command to QuickTime VR in order for QuickTime VR to handle the mouse-down event. The MouseStillDownHandler, PanZoomStartHandler and NodeLeaveHandler properties can be used to provide time to Director during the mouse-down action.

SYNTAX

```
QTVRMouseOver(gQTVRInstance)
```

If the movie is an object movie, returns nothing.

If the movie is a panoramic movie, returns a one- or two-token string describing the user's actions during mouseOver:

`pan ,0`	if panning or zooming
`jump,<newNodeID>`	if the user moved between nodes
`navg,<hotSpotID>`	if a navigable object hot spot was selected
`stil,<hotSpotID>`	if a still hot spot was selected

misc,<hotSpotID>	if a miscellaneous hot spot was selected
<type>,<hotSpotID>	if a hot spot of developer-defined type was selected
undf,<hotSpotID>	if an undefined hot spot was selected
0	user did not mouse-down: cursor moved outside panoramic movie, or exitMouseOver command used

Note that the first token of the pan/zoom result is "pan " with a space at the end.

EXAMPLES

To detect if the cursor is within the panoramic movie and pass control to QuickTime VR:

```
-- gMovieSprite is a cast member under the panoramic movie
if rollover(gMovieSprite) then
    put QTVRMouseOver(gQTVRInstance) into tMouseOverAction
    ...
```

See the sample frame script in the QTVRXtra Testbed for a fully fleshed out example.

CAUTIONS

If the movie is a panoramic movie, after a panning operation with the quality property set to 0, MouseOver returns before a high-quality update is performed. QuickTime VR must be given time, either through another MouseOver call, or through an Idle call, in order to perform the required high-quality update. The sample frame script in the QTVRXtra Testbed provides a fully fleshed out example of how this can be achieved.

The cursor command in Director only changes the cursor when Director itself thinks that the currently displayed cursor is different from the one the script is requesting. Director is not aware of QuickTime VR's cursor changes, and cursor commands designed to reset the cursor when it has moved outside the panoramic movie may be ignored. It may be necessary to change the cursor twice in order to effect the desired cursor change.

SEE ALSO

ExitMouseOver
Idle
MouseDown
MouseDownHandler
MouseOverHandler
MouseStillDownHandler
NodeLeaveHandler
PanZoomStartHandler
PassMouseDown
Quality

MouseOverHandler

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the name of the Lingo handler to be called periodically during MouseOver. Gets the name of the Lingo handler called periodically during MouseOver.

The ExitMouseOver command can be used to terminate the current mouseOver call.

If the MouseOverHandler property value is set to empty, the current callback is deactivated.

SYNTAX

```
QTVRSetMouseOverHandler(<QTVRInstance>, <handlerName>)
```

`<handlerName>` is a string containing a Lingo handler name.

```
put QTVRGetMouseOverHandler(<QTVRInstance>)into tMouseOverHandler
```

Returns a string containing a Lingo handler name. Returns empty if no callback has been set.

EXAMPLES

To set a handler that changes the frame and updates the stage on a regular basis during MouseOver:

```
QTVRSetMouseOverHandler(gQTVRInstance, "PeriodicUpdate")
...
on PeriodicUpdate
    -- Called periodically while the mouse is over
    -- the panoramic movie during a mouseOver call
    global gStartFrame, gEndFrame
    if the frame = gEndFrame then
        go to gStartFrame
    else
        go to the frame + 1
    end if
    updateStage
end PeriodicUpdate
```

To deactivate the callback:

```
QTVRSetMouseOverHandler(gQTVRInstance, empty)
```

CAUTIONS

On the Macintosh, the XCMD Callback Factory must be installed in order for the MouseOverHandler callback to work properly.

SEE ALSO

ExitMouseOver

MouseDownHandler

MouseStillDownHandler

NodeLeaveHandler

PanZoomStartHandler

RolloverHotSpotHandler

MouseStillDownHandler

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the name of the Lingo handler to be called periodically while the mouse is down. Gets the name of the Lingo handler called periodically while the mouse is down.

If the MouseStillDownHandler property value is set to empty, the current callback is deactivated.

SYNTAX

```
QTVRSetMouseStillDownHandler(<QTVRInstance>, <handlerName>)
```

`<handlerName>` is a string containing a Lingo handler name.

```
put QTVRGetMouseStillDownHandler(<QTVRInstance>) into tMouseStillDownHandler
```

Returns a string containing a Lingo handler name. Returns empty if no callback has been set.

EXAMPLES

To set a handler that changes the frame and updates the stage on a regular basis while the mouse is down:

```
QTVRSetMouseStillDownHandler(gQTVRInstance,"PeriodicUpdate")
...
on PeriodicUpdate
    -- Called periodically while the mouse is down
    -- during a MouseOver or MouseDown call
    global gStartFrame, gEndFrame
    if the frame = gEndFrame then
        go to gStartFrame
    else
        go to the frame + 1
    end if
    updateStage
end PeriodicUpdate
```

To deactivate the callback:

```
QTVRSetMouseStillDownHandler(gQTVRInstance, empty)
```

CAUTIONS

On the Macintosh, the XCMD Callback Factory must be installed in order for the MouseStillDownHandler callback to work properly.

SEE ALSO

MouseDownHandler

MouseOverHandler

NodeLeaveHandler

PanZoomStartHandler

RolloverHotSpotHandler

NodeID

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the node ID. Gets the current node ID.

The nodeID, hPanAngle, vPanAngle and zoomAngle properties completely describe a view within a particular VR movie.

Node IDs are positive integers set at authoring time. The set of node IDs in a VR movie need not be contiguous. Setting a node ID immediately changes the current node, although imaging waits until the next update is sent.

SYNTAX

```
QTVSetNodeID(<QTVRInstance>, <nodeID>)
```

`<nodeID>` is a value containing a positive integer. If `<nodeID>` is invalid, does nothing.

```
put QTVGetNodeID(<QTVRInstance>) into tNodeID
```

Returns a string containing the current node ID.

EXAMPLES

To look in the default direction in a new node:

```
QTVSetNodeID(gQTVRInstance, 3)
QTVRUpdate(gQTVRInstance)
```

To display the current node ID:

```
put QTVGetNodeID(gQTVRInstance) into tNodeID
put "Current node ID is:" && tNodeID
```

CAUTIONS

When setting up a new view for display, it is important to realize that the Node ID property is evaluated immediately, and that changing the Node ID automatically sets up default pan, tilt and FOV values set at authoring time. It is important to change the node ID before setting FOV, TiltAngle and PanAngle property values:

```
QTVSetNodeID(gQTVRInstance, tNodeID)
QTVRSetFOV(tFOV)
QTVRSetTiltAngle(tTiltAngle)
QTVRSetPanAngle(tPanAngle)
```

KNOWN BUGS

If `<nodeID>` is invalid while setting the nodeID property value, the pan, tilt and FOV values are all reset to the defaults for the current node.

SEE ALSO

FOV PanAngle TiltAngle

NodeLeaveHandler

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the name of the Lingo handler to be called when a node is exited. Gets the name of the Lingo handler called when a node is exited.

If the nodeLeaveHandler property value is set to empty, the callback is deactivated.

SYNTAX

```
QTVRSetNodeLeaveHandler(<QTVRInstance>, <handlerName>)
```

`<handlerName>` is a string containing a Lingo handler name.

```
put QTVRSetNodeLeaveHandler(<QTVRInstance>) into tNodeLeaveHandler
```

Returns a string containing a Lingo handler name. Returns empty if no callback has been set.

EXAMPLES

To set a handler that plays a sound whenever a node is exited:

```
QTVRSetNodeLeaveHandler(gQTVRInstance,"PlayJumpSound")
...
on PlayJumpSound pNewNodeID
    -- Called when a node is being exited
    beep
end PlayJumpSound
```

To deactivate the callback:

```
QTVRSetNodeLeaveHandler(gQTVRInstance, empty)
```

CAUTIONS

On the Macintosh, the XCMD Callback Factory must be installed in order for the NodeLeaveHandler callback to work properly.

SEE ALSO

MouseDownHandler

MouseOverHandler

MouseStillDownHandler

PanZoomStartHandler

RolloverHotSpotHandler

NodeName

Pano Only

GET PROPERTY

DESCRIPTION

Gets the current node's name.

Each node is named during the authoring process. The NodeName property provides the name of the current node.

The NodeName property can not be used to go to a particular named node. It can only be retrieved for the current node. The NodeID property must be used to change or set nodes.

SYNTAX

```
put QTVRGetNodeName(<QTVRInstance>) into tNodeName
```

Returns a string containing the name of the current node. If no name was assigned to the node during the authoring process, the returned string will be empty.

EXAMPLES

To display the current node's name:

```
put QTVRGetNodeName(gQTVRInstance) into tNodeName
put "Current node's name is:" && tNodeName
```

SEE ALSO

NodeID

Nudge

Object Only

COMMAND

DESCRIPTION

Turn the object one step in a particular direction and display the new view. Since an object movie contains a discrete number of views of the object, a range of pan and tilt angles will map to a single view. The Nudge command allows you to change to an adjacent view of the object. The new view is displayed immediately when the command is executed.

SYNTAX

```
QTVRNudge(<QTVRInstance>,<direction>)
```

`<direction>` is one of the following strings: `up`, `down`, `left` or `right`. The string indicates the direction the object should turn.

EXAMPLES

To turn the object one step to the right and display the new view:

```
QTVRNudge(gQTVRInstance,"right")
-- Update does not need to be called
```

SEE ALSO

PanAngle

TiltAngle

ObjectViewAngles

Pano Only

GET PROPERTY

DESCRIPTION

Gets the poster view for use in the matching VR object movie of the selected hot spot.

If the selected hot spot is not of type 'navg', the ObjectViewAngles property will be empty.

The ObjectViewAngles property returns a string containing a pair of comma-separated angles: the pan angle and the tilt angle. This pair of values can be used to display the matching VR object so that it appears at the same angle as the object appears within the current node of panoramic movie.

SYNTAX

```
put QTVRGetObjectViewAngles(<QTVRInstance>) into tNavgViewAngles
```

Returns a string containing a pair of comma-separated angles:

```
<PanAngle>,<TiltAngle>
```

If a hot spot is not selected, the returned string will be empty.

EXAMPLES

To open an object movie with a view that matches the orientation of the object within the panoramic movie:

```
-- Assume a 'navg' hot spot is selected
put QTVRGetObjectViewAngles(gQTVRInstance) into tObjectViewAngles
put QTVRGetHotSpotName(gQTVRInstance) into tObjectName
if tObjectViewAngles<> empty then
    put item 1 of tObjectViewAnglesinto tObjectPanAngle
    put item 2 of tObjectViewAnglesinto tObjectTiltAngle
    -- Use these two values to display the first frame of the matching
    -- object movie
    OpenObjectMovie tObjectName, tObjectPanAngle, tObjectTiltAngle
else
    put "Object hot spot not currently selected."
end if
```

CAUTIONS

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickLoc HotSpotID

MouseDown MouseOver

ObjectZoomRect

Pano Only

GET PROPERTY

DESCRIPTION

Gets the starting rectangle for the selected hot spot for the zooming effect used to transition from panoramas to objects.

If the selected hot spot is not of type 'navg', the ObjectZoomRect property will be empty.

The ObjectZoomRect property returns a string containing a rect. This rect can be used as the starting point for the zooming effect which transitions from the panorama to the matching QuickTime VR object. The rect is based on coordinates provided at authoring time, which are reprocessed based on the current view. The rect may lie partially outside the panorama movie.

SYNTAX

```
put QTVRGetObjectZoomRect(<QTVRInstance>) into tObjectZoomRect
```

Returns a string containing a rect.

If a hot spot is not selected, or the selected hot spot is not of type 'navg', the returned string will be unpredictable.

EXAMPLES

To open an object movie by zooming to a view that matches the orientation of the object within the panoramic movie:

```
put QTVRGetObjectZoomRect(gQTVRInstance) into tObjectZoomRect
put QTVGetObjectViewAngles(gQTVRInstance) into tObjectViewAngles
put QTVRGetHotSpotName(gQTVRInstance) into tObjectName
if tObjectZoomRect <> empty and tObjectViewAngles <> empty then
    put item 1 of tObjectViewAngles into tObjectPanAngle
    put item 2 of tObjectViewAngles into tObjectTiltAngle
    -- Use these two values to display the first frame of the matching
    -- object movie
    ZoomObjectMovie tObjectName, tObjectPanAngle, tObjectTiltAngle, ¬
        tObjectZoomRect
else
    put "Object movie hot spot not currently selected."
end if
```

CAUTIONS

The hot spot selection is only reset after a new hot spot is selected interactively or a hot spot is set programmatically. The persistence of the hot spot selection may lead to invalid hot spot property values if the current node has changed since the last hot spot selection.

SEE ALSO

ClickLoc HotSpotID

MouseDown MouseOver

Open

COMMAND

DESCRIPTION

Opens a QuickTime VR movie within Director.

Before calling Open, you must have loaded the Xtra using the Lingo command `xtra "QTVRXtra"`, called the Enter command (see Enter), and created an instance of that Xtra using the Lingo `new` command which is fully explained in the Director documentation.

OpenMovie specifies the VR file to be opened, the location and optionally the size of the window in which to display the movie. If the call is successful, it returns empty. Otherwise it returns an error message.

SYNTAX

```
QTVROpen(<QTVRInstance> <filePath>, <loc>|<rect>, <visibleString>)
```

`<filePath>` is a string which specifies either the full path to a QuickTime VR movie, or the filename of a QuickTime VR movie found in the same directory as the current application.

`<loc>|<rect>` is a string which specifies either the top left corner of the window, which will be opened at the default size stored in the movie itself, or the window rect it is to occupy. The coordinates are measured in Director's coordinate system.

If the movie is a panoramic movie, the visibleString parameter specifies whether or not the movie is displayable to the screen after it is opened. If the visibleString patrameter is set to "invisible", QuickTime VR will not image the panorama until its visible property is set to true. If the visibleString patrameter is set to "visible", QuickTime VR will image the movie to the screen when it receives an update command or as soon as it receives idle time from the QuickTime scheduler (under no circumstances will the movie will be displayed as part of the OpenMovie command itself). Because idle time will be provided automatically to QuickTime VR if there are any other QuickTime movies open, if the visibleString patrameter is set to "visible", QuickTime VR may image the movie unexpectedly when another movie is being used.

Returns empty or an error message. If the first word in the returned string is "Error", an error occurred and the movie was not opened. The rest of the string contains the error message.

EXAMPLES

To open a movie and anchor its window at a specific location:

```
QTVROpen(gQTVRInsatnce, "HD:Test Movie", "100,100", "visible")
```

To open a movie and set its window to a specific size:

```
-- Opens a 400 x 300 window anchored at 100,100
QTVROpen(gQTVRInsatnce, "HD:Test Movie", "100,100,500,400", "visible")
```

CAUTIONS

Multiple movies can be open at the same time. Every call to OpenMovie should eventually be balanced with a Close call. If, during development, you encounter script

errors which cause Director to stop executing, make sure you close any open QuickTime VR movies. You may have to do this by hand in the Message window. If you do not close open QuickTime VR movies, the memory used by those movies will NOT be freed.

On the Windows™ Xtra, it is also important that the size of the display Rect specified as well as the VR movie itself (both in pixel coordinates) be divisible by 4. If not, your VR movies can take a serious performance hit while panning and unpredictable results can occur.

SEE ALSO

Close Enter

Exit Visible

PanAngle

SET/GET PROPERTY

DESCRIPTION

Sets the pan angle to be used when next computing an image. Gets the pan angle to be used when next computing an image. If the PanAngle has not been changed (directly or indirectly) since the last update, the PanAngle property will return the PanAngle used to compute the last image.

The PanAngle and TiltAngle properties completely describe a view of a particular QuickTime VR object movie. Increasing the pan angle rotates the object to the left. The pan angle value is measured in degrees. The value is clipped to the minimum or maximum pan angle or the value is wrapped around into the range 0°-360° if the pan angle is out of range.

The PanAngle, TiltAngle, FOV and NodeID properties completely describe a view within a particular panoramic movie. Changing the pan angle rotates the viewpoint left and right.

The pan angle value is measured in degrees increasing to the left, and is constrained by the node's minimum and maximum horizontal pan angles. Values which exceed the node's horizontal constraint angles are clipped. Values which, when coupled with the current TiltAngle and FOV property values, would cause an image to lie beyond the current node's constraints are also clipped. If the panorama wraps around, values less than 0° or greater than 360° are automatically wrapped around into the range 0°-360°.

SYNTAX

```
QTVRSetPanAngle(<QTVRInstance>, <angle>)
```

`<angle>` is a string containing an angle measured in floating point degrees.

```
put QTVRGetPanAngle(<QTVRInstance>) into tPanAngle
```

Returns a string containing an angle measured in floating point degrees.

EXAMPLES

```
QTVRSetPanAngle(gQTVRInstance, "132.2")
QTVUpdate(gQTVRInstance)
```

To display the current horizontal pan angle:

```
put QTVRGetPanAngle(gQTVRInstance) into tPanAngle
put "Current pan angle is:" && tPanAngle
```

CAUTIONS

If the movie is a panoramic movie, when setting up a new view for display, it is important to realize that the PanAngle value is clipped at the time it is set, not at the time it is imaged. Consequently, it is important to set the PanAngle, TiltAngle and FOV properties in the correct order:

```
QTVRSetFOV(gQTVRInstance, tFOV)
QTVRSetTiltAngle(gQTVRInstance, tVTiltAngle)
QTVRSetPanAngle(gQTVRInstance, tPanAngle)
```

If the FOV property is changing and is not set first, the PanAngle property will be clipped against the old FOV instead of the new one, and the resulting view may be incorrect.

SEE ALSO

FOV	TiltAngle
NodeID	Update

PanZoomStartHandler

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the name of the Lingo handler to be called once when a user starts panning or zooming. Gets the name of the Lingo handler called once when a user starts panning or zooming.

The handler specified by the PanZoomStartHandler property is only called as a result of interactive mouse actions. It is not called during a swing update.

If the PanZoomStartHandler property value is set to empty, the callback is deactivated.

SYNTAX

```
QTVRSetPanZoomStartHandler(<QTVRInstance>,<handlerName>)
```

`<handlerName>` is a string containing a Lingo handler name.

```
put QTVRGetPanZoomStartHandler(<QTVRInstance>) into tPanZoomStartHandler
```

Returns a string containing a Lingo handler name. Returns empty if no callback has been set.

EXAMPLES

To set a handler that plays a sound whenever the user starts panning or zooming:

```
QTVRSetPanZoomStartHandler(gQTVRInstance,"PlayPanningSound")
...
on PlayPanningSound
    -- Called when the user starts panning or zooming
    beep
end PlayPanningSound
```

To deactivate the callback:

```
QTVRSetPanZoomStartHandler(gQTVRInstance, empty)
```

CAUTIONS

On the Macintosh, the XCMD Callback Factory must be installed in order for the PanZoomStartHandler callback to work properly.

SEE ALSO

MouseDownHandler

MouseOverHandler

MouseStillDownHandler

NodeLeaveHandler

RolloverHotSpotHandler

PassMouseDown

Pano Only

COMMAND

DESCRIPTION

Used during a MouseDownHandler callback, instructs QuickTime VR to execute its internal MouseDown code when control returns from Director. If PassMouseDown is not called in the MouseDownHandler, QuickTime VR will assume that the MouseDownHandler has managed the MouseDown activity itself and will return immediately.

PassMouseDown has no effect if it is called at times other than during a MouseDownHandler callback.

SYNTAX

```
QTVRPassMouseDown(<QTVRInstance>)
```

EXAMPLES

To instruct QuickTime VR to execute its internal mouseDown code after the mouseDownHandler has returned:

```
on mouseDownHandler
    global gQTVRInstance
    ...

    QTVRPassMouseDown(gQTVRInstance)
end mouseDownHandler
```

SEE ALSO

MouseOver

MouseDownHandler

QTVRType

GET PROPERTY

DESCRIPTION

Returns the type of QTVR movie currently loaded.

SYNTAX

```
put QTVRGetQTVRType(<QTVRInstance>) into tMovieType
```

Returns the string "QTVRPanorama" if movie is a panoramic movie, "QTVRObject" if movie is an object movie, and "NotAQTVRType" otherwise.

EXAMPLES

```
if QTVRGetQTVRType(gQTVRInstance) = "QTVRPanorama" then
    --perform some action specific to panoramas like QTVRSetFOV
end if
```

SEE ALSO

IsQTVRMovie

Quality

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the quality level to be used when imaging a panorama. Gets the quality level to be used when imaging a panorama.

If the quality has not been changed (directly or indirectly) since the last update, the quality property will return the quality used to compute the last image.

The quality property is a floating point number, passed as a string. Currently recognized values are:

`4.0`	Highest quality rendering. The rendered image is fully anti-aliased.
`2.0`	The rendered image is partially anti-aliased.
`1.0`	The rendered image is not anti-aliased.
`0.0`	Images are rendered at quality 1.0 when the user is interactively panning or zooming, and are automatically updated at quality 4.0 during idle time when the user stops panning or zooming. All other updates are rendered at quality 4.0.

Higher quality values result in longer imaging times, and slower update speeds.

SYNTAX

```
QTVRSetQuality(<QTVRInstance>, <quality>)
```

`<quality>` is a string containing a floating point number, which can also be passed as a string containing an integer, or as an integer.

```
put QTVRGetQuality(<QTVRInstance>) into tQuality
```

Returns a string containing a floating point number.

EXAMPLES

To set a low quality level:

```
QTVRSetQuality(gQTVRInstance, "1")
```

To do a high-quality update, as well as setting up for automatic low-quality/high-quality interactive updates:

```
QTVRSetQuality(gQTVRInstance, "0")
QTVRUpdate(gQTVRInstance)
```

SEE ALSO

Update

RolloverHotSpotHandler

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the name of the Lingo handler to be called during MouseOver when the user mouses over a different hot spot. Gets the name of the Lingo handler called during MouseOver when the user mouses over a different hot spot.

The handler specified by the RolloverHotSpotHandler property is called every time the hot spot under the mouse changes during MouseOver. The handler is passed the hot spot ID of the hot spot under the mouse. If the mouse is not over a hot spot, the handler is passed an ID of 0. The handler is only called when the hot spot ID changes, as well as when mouseOver is first called.

If the RolloverHotSpotHandler property value is set to empty, the callback is deactivated.

SYNTAX

```
QTVRSetRolloverHotSpotHandler(<QTVRInstance>,<handlerName>)
```

`<handlerName>` is a string containing a Lingo handler name.

```
put QTVRSetRolloverHotSpotHandler(<QTVRInstance>) into¬
tRolloverHotSpotHandler
```

Returns a string containing a Lingo handler name. Returns empty if no callback has been set.

EXAMPLES

To set a handler that displays the hot spot ID the user is currently over:

```
QTVRSetRolloverHotSpotHandler(<QTVRInstance>,"DisplayHotSpotID")
...
on DisplayHotSpotID pHotSpotID
    -- Called when the cursor rolls over a new hot spot
    if pHotSpotID <> 0 then
        put "Cursor is over:" && pHotSpotID
    else
        put "Cursor is over nothing"
    end if
end DisplayHotSpotID
```

To deactivate the callback:

```
QTVRSetRolloverHotSpotHandler(<QTVRInstance>, empty)
```

CAUTIONS

On the Macintosh, the XCMD Callback Factory must be installed in order for the rolloverHotSpotHandler callback to work properly.

SEE ALSO

ExitMouseOver
MouseDownHandler
MouseOverHandler
MouseStillDownHandler
NodeLeaveHandler
PanZoomStartHandler

Row

Object Only

SET/GET PROPERTY

DESCRIPTION

Sets the row number for the next view. Gets the row number of the current view.

Normally you should use PanAngle and TiltAngle to specify the view of an object movie. The object controller would then display the closest captured view matching the pan and tilt angles. However, you may use the Row and Column commands to explicitly specify which captured view should be displayed. The row property specifies the vertical position of the camera during capture.

The Row value should be in the range 0 to (number of vertical positions -1): 0 specifies the top-most row, 1 the next row down, etc. The value is clipped if it is out of range.

SYNTAX

```
QTVRSetRow(<QTVRInstance>, <rowNumber>)
```

`<rowNumber>` is an integer number.

```
put QTVRGetRow(<QTVRInstance>) into tRowNumber
```

EXAMPLES

To change the view of an object:

```
QTVRSetRow(gQTVRInstance, 5)
QTVRUpdate(gQTVRInstance)
```

To display the current row number:

```
put QTVRGetRow(gQTVRInstance) into tRowNumber
put "Current row number is:" && tRowNumber
```

CAUTIONS

The Row and Column properties are not as general as the PanAngle and TiltAngle properties since you must have explicit knowledge of how the object was captured. Use the Row property only when necessary.

SEE ALSO

Column PanAngle

TiltAngle Update

TiltAngle

SET/GET PROPERTY

DESCRIPTION

Sets the tilt angle to be used when next computing an image. Gets the tilt angle to be used when next computing an image.

If the TiltAngle has not been changed (directly or indirectly) since the last update, the TiltAngle property will return the TiltAngle used to compute the last image.

The PanAngle and TitlAngle properties completely describe a view of a particular QuickTime VR object movie. Increasing the tilt angle rotates the object down. The tilt angle value is measured in degrees. The value is clipped to the minimum or maximum tilt angle if it is out of range.

The TiltAngle, PanAngle, FOV and NodeID properties completely describe a view within a particular panoramic movie. Changing the tilt angle rotates the viewpoint up and down.

The tilt angle value is measured in degrees increasing up and decreasing down from the horizontal center line of the panorama, and is constrained by the node's minimum and maximum tilt angles. Values which exceed the node's vertical constraint angles are clipped. Values which, when coupled with the current PanAngle and FOV property values, would cause an image to lie beyond the current node's constraints are also clipped.

SYNTAX

```
QTVRSetTiltAngle(<QTVRInstance>, <angle>)
```

`<angle>` is a string containing an angle measured in floating point degrees.

```
put QTVRGetTiltAngle(<QTVRInstance>) into tVPanAngle
```

Returns a string containing an angle measured in floating point degrees.

EXAMPLES

To jump to a different view within the node:

```
QTVRSetTiltAngle(gQTVRInstance, "14.2")
QTVRUpadte(gQTVRInstance)
```

To display the current vertical pan angle:

```
put QTVRGetTiltAngle(gQTVRInstance) into tTiltAngle
put "Current vertical pan angle is:" && tTiltAngle
```

CAUTIONS

If the movie is a panoramic movie, when setting up a new view for display, it is important to realize that the TiltAngle property is clipped at the time it is set, not at the time it is imaged. Consequently, it is important to set the TiltAngle, PanAngle and FOV properties in the correct order:

```
QTVRSetFOV(gQTVRInstance,  tFOV)
QTVRSetTiltAngle(gQTVRInstance,  tVTiltAngle)
QTVRSetPanAngle(gQTVRInstance,  tPanAngle)
```

If the FOV property is changing and is not set first, the PanAngle property will be clipped against the old FOV instead of the new one, and the resulting view may be incorrect.

SEE ALSO

FOV	PanAngle
NodeID	TransitionMode

TransitionMode

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the type of transition from the current view to the view that will be displayed on the next update command. Gets the type of transition that will be used in the next update command.

If the TransitionMode has not been changed (directly or indirectly) since the last update, the TransitionMode property will return the transitionMode used during the last update.

In combination with the UpdateMode property, TransitionMode determines the type of update that will take place. The TransitionMode can be:

`normal`	The new view is imaged and displayed. No transition effect is used. The user sees a "cut" from the current view to the new view.
`swing`	If the new view is in the current node, the view point moves smoothly from the current view to the new view, taking the shortest path if the panorama wraps around. The speed of the swing is controlled by the transitionSpeed property. If the new view is in a different node, the new view is imaged and displayed with a "normal" transition.

SYNTAX

```
QTVRSetTransitionMode(<QTVRInstance>, <transitionMode>)
```

`<transitionMode>` is one of the following strings: `normal` or `swing`. If `<transitionMode>` is invalid, behavior is unpredictable.

```
put QTVRGetTransitionMode(<QTVRInstance>) into tTransitionMode
```

Returns a string describing the current transition mode.

EXAMPLES

To swing to a particular view:

```
on SwingPanoMovie pHPan, pVPan, pZoom, pSwingSpeed, pSwingQuality, pFinalQuality
  global gQTVRInstance
  if IsQTVRMovie(gQTVRInstance) then
    -- Set up the destination pan and zoom angles
    QTVRSetFOV(gQTVRInstance,pZoom)
    QTVRSetTiltAngle(gQTVRInstance,pVPan)
    QTVRSetPanAngle(gQTVRInstance,pHPan)

    -- For performance, you can use lower quality during the swing
    QTVRSetQuality(gQTVRInstance, pSwingQuality)
    QTVRSetTransitionMode(gQTVRInstance, "swing")
    QTVRSetTransitionSpeed(gQTVRInstance, pSwingSpeed)
    QTVRUpdate(gQTVRInstance)

    -- Set transition mode to normal
    QTVRSetTransitionMode(gQTVRInstance, "normal")

    -- Only do a reupdate if the quality values are different
```

```
    if pFinalQuality <> pSwingQuality then
      QTVRSetQuality(gQTVRInstance, pFinalQuality)
      QTVRUpdate(gQTVRInstance)
    end if
  end if
end SwingPanoMovie
```

KNOWN BUGS

If the TransitionMode property is set to an invalid value, behavior during the next update is unpredictable. Furthermore, until the TransitionMode property value is set to a valid mode, a get TransitionMode command can return unpredictable values.

CAUTIONS

If the TransitionMode property is set to "swing" when a MouseOver or MouseDown command is sent to QuickTime VR, unusual behavior may result.

SEE ALSO

TransitionSpeed UpdateMode Update

TransitionSpeed

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the speed of the next swing transition performed by the update command. Gets the speed of the next swing transition.

In combination with the TransitionMode property, TransitionSpeed determines the type of swing update that will take place.

The TransitionSpeed is a floating point quantity that provides the slowest swing transition at 1.0 and faster transitions at higher values. On mid-range computers, a rate of 4.0 performs well off CD-ROM.

SYNTAX

```
QTVRSetTransitionSpeed(<QTVRInstance>, <speed>)
```

`<speed>` is a string containing a positive floating-point value, which can also be passed as a string containing an integer. If `<speed>` is invalid, updates will behave unpredictably.

```
put QTVRGetTransitionSpeed(<QTVRInstance> into tTransitionSpeed
```

Returns a string containing a floating-point value.

EXAMPLES

To slowly swing to a new view at low-quality:

```
-- Set up the transition
QTVRSetTransitionSpeed(gQTVRInstance, "1")
QTVRSetTransitionModegQTVRInstance, "swing")
-- Set quality low for faster imaging during swing
QTVRSetQuality(gQTVRInstance, "1")
QTVRUpadte(gQTVRInstance)
QTVRSetTransitionModegQTVRInstance, "normal")
```

KNOWN BUGS

If the TransitionSpeed property value is set to a negative number, swing updates will act the same as normal updates.

SEE ALSO

TransitionMode

Update

Update

COMMAND

DESCRIPTION

For object movies, Update instructs the object controller to redraw the current view of the object on the display screen.

The most common usage of update are:

- to display the current view if the openMovie command included the "invisible" parameter.
- to display an alternate view of the object view after calls to change the hPanAngle, and vPanAngle.
- to refresh the current view if the movie was previously drawn over by other cast members, covered up by other windows, or if Director becomes the foreground application again.

For panoramic images, Update instructs the panoramic imaging system to update or reimage the display screen, the offscreen buffer, or both. The exact action taken depends on the settings of the UpdateMode, TransitionMode and TransitionSpeed properties.

The most common usages of update are to:

- compute and display an image into the offscreen buffer and onto the screen based on the current NodeID, PanAngle, TiltAngle, FOV and Quality settings. This can be used when first opening a panoramic movie, and when jumping to a new view within a node or to a new node under program control.
- compute and display an image into the offscreen buffer based on the current nodeID, PanAngle, TiltAngle, FOV and quality settings without changing what appears on the screen (UpdateMode "offscreenOnly"). This is useful when wanting to precompute an image which will need to be rapidly displayed, such as when changing screens while opening a panoramic movie.
- display or redisplay the offscreen buffer onto the screen without recomputing it (UpdateMode "fromOffscreen"). This allows extremely rapid display of a precomputed panoramic image.
- animate the viewpoint within a node by swinging from the last hPanAngle, vPanAngle and zoomAngle settings used to the current PanAngle, TiltAngle and FOV settings (TransitionMode "swing"). Each image in the sequence is computed and displayed into the offscreen buffer and onto the screen at the current quality setting. The speed at which the viewpoint swings is governed by the transitionSpeed property.

SYNTAX

```
QTVRUpdate(<QTVRInstance>)
```

EXAMPLES

To set up and display a particular viewpoint in the current node or object at the current FOV and quality settings:

```
QTVRSetTiltAngle(gQTVRInstance,  tVTiltAngle)
QTVRSetPanAngle(gQTVRInstance,  tPanAngle)
QTVRUpdate(gQTVRInstance)
```

CAUTIONS

If a panoramic movie is currently invisible (opened invisible or set invisible using the Visible property), the Update command will make the movie visible.

SEE ALSO

FOV	NodeID	PanAngle
Quality	TiltAngle	TransitionMode
TransitionSpeed	UpdateMode	Visible

UpdateMode

Pano Only

SET/GET PROPERTY

Mac™ OS Only

DESCRIPTION

Sets the combination of offscreen buffer and display screen updated during the next update command or interactive action. Gets the combination of offscreen and display screen buffer update that will be used during the next update command or interactive action.

In combination with the TransitionMode property, UpdateMode determines the type of update that will take place. The UpdateMode can be:

`normal`	Images are computed and displayed directly onto the screen while interactively panning and zooming. Provides the fastest overall frame rate, but individual frame draw times may be relatively slow for high-quality images on lower performance systems. Programmatic updates are displayed into the offscreen buffer, and then onto the screen.
`updateBoth`	Images are computed and displayed into the offscreen buffer,and then onto the screen. The use of the back buffer reduces the overall frame rate but provides the fastest imaging time for each frame.
`offscreenOnly`	Images are computed and displayed into the offscreen buffer only, and are not copied to the screen. Useful for preparing for a screen refresh that will have to happen quickly.
`fromOffscreen`	The offscreen buffer is copied directly to the screen. The offscreen buffer is refreshed only if it is out of date. The offscreen buffer is automatically updated if necessary to keep it in sync with the screen image.
`directToScreen`	Images are computed and displayed directly to the screen, without changing the offscreen buffer. Provides the fastest overall frame rate, but individual frame draw times may be relatively slow for high-quality images on lower performance systems.

SYNTAX

```
QTVRSetUpdateMode(<QTVRInstance>, <updateMode>)
```

`<updateMode>` is one of the following strings: `normal`, `updateBoth`, `offscreenOnly`, `fromOffscreen`, or `directToScreen`. If `<updateMode>` is invalid, behavior is unpredictable.

```
put QTVRSetUpdateMode(gQTVRInstance) into tUpdateMode
```

CAUTIONS

If the UpdateMode property is set to "offscreenOnly" when a MouseOver or MouseDown command is sent to QuickTime VR, no screen updates will be shown.

KNOWN BUGS

If <updateMode> is invalid while setting the UpdateMode property value, behavior during the next update is unpredictable. Furthermore, until the UpdateMode property value is set to a valid mode, a GetUpdateMode command can return unpredictable values.

SEE ALSO

TransitionMode

Update

Visible

SET/GET PROPERTY

DESCRIPTION

Sets the visibility property which determines whether or not the movie is imaged. Gets the current value of the visibility property.

SYNTAX

```
QTVRSetVisible(<QTVRInstance>, <boolean>)
```

`<boolean>` is one of the following values: zero (`false`) or non-zero (`true`).

```
put QTVRSetVisible(gQTVRInstance) into tVisible
```

Returns one of the following values: zero (`false`) or one (`true`).

EXAMPLES

To display a panoramic or object movie which was opened with the "invisible" parameter specified:

```
QTVRSetVisible(gQTVRInstance, 1)
-- You can also use:  QTVRSetVisible(gQTVRInstance, true)
QTVRUpdate(gQTVRInstance)
```

To get and display the current movie visibility:

```
if QTVRGetVisible(gQTVRInstance) then
  put "Movie is visible"
else
  put "Movie is invisible"
end if
```

To prevent a panoramic or object movie from imaging while another QuickTime movie is being used:

```
QTVRSetVisible(gQTVRInstance, 0)
-- You can also use:  QTVRSetVisible(gQTVRInstance, false)
```

SEE ALSO

OpenMovie

Update

WarpMode

Pano Only

SET/GET PROPERTY

DESCRIPTION

Sets the type of warping to be used when imaging the panorama. Gets the type of warping to be used when imaging the panorama.

The WarpMode property is an integer, which can be passed as a string. Currently recognized values are:

`2`	Two-dimensional warping. This produces perspectively correct images from a panoramic source picture.
`1`	One-dimensional warping.
`0`	No warping. This reproduces the source panorama directly, without warping it at all.

Higher warping values require longer imaging times. The default value for the WarpMode property is 2.

SYNTAX

```
QTVRSetWarpMode(<QTVRInstance>, <warpMode>)
```

`<warpMode>` is a string containing a integer, which can be passed as an integer. If warpMode is invalid, behavior is unpredictable.

```
put QTVRSetWarpMode(gQTVRInstance) into tWarpMode
```

Returns a string containing a integer.

EXAMPLES

To do a single update without warping and switch back to two-dimensional warping:

```
QTVRSetWarpMode(gQTVRInstance, "0")
QTVRUpdate(gQTVRInstance)
QTVRSetWarpMode(gQTVRInstance, "2")
```

To display the current warping type:

```
put QTVRSetWarpMode(gQTVRInstance) into tWarpMode
put "Current warping type:" && tWarpMode
```

CAUTIONS

If the WarpMode property is set to an invalid value, behavior during the next update is unpredictable.

SEE ALSO

Update