

tiltGallery

User Guide revision 1.1

www.flashloaded.com

Table of Contents

Installation	3
Getting started	4
Component Inspector parameters	5
General	5
Layout	5
Preloading	6
Camera	6
Animations	7
Easing	8
Large elements	9
Thumbnails	9
XML	10
Skinning	12
Displaying RSS feeds	15
Crossdomain file	16
ActionScript events	17
Performing any action on click	19
Opening a URL on click	20
ActionScript methods	21
Help	22

Installation

You will need Adobe Extension Manager in order to install this component. Extension Manager should have been installed by default when you installed Flash. You may download the latest version of Extension Manager for free from the [Adobe website](#).

1. Ensure that Flash is closed before installing the tiltGallery component.
2. Unzip/extract the tiltGallery.zip file that you downloaded. You will find a file called tiltGallery.mxp. Double click on this file in order to install the component using Extension Manager.

tiltGallery should now be installed in your Flash Components Panel in the *Flashloaded* folder.

Getting started

1. Having installed tiltGallery using the Adobe Extension Manager, start a new Flash ActionScript 3.0 file and save it.
2. Locate the **Flashloaded** folder in the Components panel and double click on it to expand it. You will find the **tiltGallery** component inside this folder.
3. Drag and drop the **tiltGallery** component onto the stage. The component initially has a default skin containing all of the fields for RSS feeds. Please refer to the [skinning](#) section for instructions on skinning the component and removing information that you don't wish to display in RSS feeds.
4. Click on the tiltGallery component that's on the stage and open the Component Inspector panel (shift +F7).
5. If you are using the tiltGallery to display an RSS feed, enter the URL of the RSS feed in the **XML/RSS Source** parameter of the component and set the **feedType** parameter to *RSS*.

If you wish to display your own images, SWF's or videos in the tiltGallery, you should create an XML file which contains the file info and enter the name of the XML file in the **XML/RSS Source** parameter of the component and set the **feedType** parameter to *XML*. Please refer to the [XML](#) section for instructions on creating the XML file.

6. At this stage, you can already test tiltGallery with the default parameters, to ensure that you have set it up correctly. Press Ctrl+Enter (win) or Cmnd+Enter (mac) to test your project.
7. You can change the various parameter settings in the Component Inspector to obtain the desired look and feel. Please see the [Component Inspector parameters](#) section for a description on each setting.

Component Inspector parameters

General

Parameter	Description	Example
XML/RSS Source	The URL of the RSS feed display or the path and filename of the XML file containing the data of the images, videos and SWF's to display. Please see the XML section for instructions on creating the XML file.	gallery.xml
feedType	Sets whether to the XML/RSS Source is an XML or RSS feed.	XML
Proxy file URL	This parameter is only necessary when displaying an RSS feed that is located on a different domain. Enter the url to the PHP or ASP proxy file to use (depending on whether your server supports PHP or ASP).	tiltgalleryproxy.php
tiltEnabled	Sets whether the tilt is enabled for only horizontal, vertical or both directions.	both
tiltStrength	The strength of tilt to apply when tilt is enabled. A lower number will give a more subtle tilt.	7

Layout

Parameter	Description	Example
columnLimit	The maximum number of columns to display the content over. The rows are calculated automatically.	3
itemLimit	The number of entries to display from the RSS or XML feed.	30
stretchThumbnailsToFill	Sets whether to stretch the thumbnail images to fill the defined thumbnail size or not. The thumbnail size is set in the skin .	true

Parameter	Description	Example
thumbnailSpacingHorizontal	The horizontal space (in pixels) between the thumbnails.	15
thumbnailSpacingVertical	The vertical space (in pixels) between the thumbnails.	15
showFullScreenButton	Sets whether to display the full screen button or not.	true

Preloading

Parameter	Description	Example
preloadAll	<p>If set to true, the large images will preload and be cached. This makes for a smoother user experience as is the animation will move from the thumbnail to large image without displaying a preloader.</p> <p><i>Note: This only applies to images. SWF and flv files will still require a preloader.</i></p>	TRUE
preloaderType	Sets whether to display a text preloader while the thumbnails load or a graphical preloader in the placeholder of each loading thumbnail.	text
preloaderText	The customizable text which appears for the text preloader. The words %NUM% and %TOTAL% are variables that are replaced by the current number loaded and total number of images to load.	Loading %NUM% of %TOTAL% images

Camera

Parameter	Description	Example
cameraZoom	The amount that the camera is zoomed into the component.	4
cameraMoveSpeed	The speed at which the camera moves.	0.4
cameraReactionTime	The response time of the camera.	0.6

Animations

Parameter	Description	Example
clickedItemFadeInOutSpeed	The speed at which a clicked item fades out. Set to 0 for no fading.	1
clickedItemGoTargetSpeed	The speed at which the thumbnails regroup to the starting position.	3
thumbnailOutHorizontalDistanceRatio	Sets the distance ratio that the thumbnails move out horizontally, relative to the width of the component.	3.5
thumbnailOutVerticalDistanceRatio	Sets the distance ratio that the thumbnails move out vertically, relative to the height of the component.	3.5
thumbnailFadeOutSpeed	The speed at which the thumbnails fade out when moving away to reveal the large element. Set to 0 for no fading.	3
thumbnailResizeSpeed	The speed at which the thumbnails resize when moving away to reveal the large element.	1
largePlaneRSSFadeInSpeed	The speed at which the detailed RSS item fades in.	1
largePlaneRSSFadeOutSpeed	The speed at which the detailed RSS item fades out. Set to 0 for no fading.	1
largePlaneZoomInSpeed	The speed at which the large element zooms in.	1
largePlaneZoomOutSpeed	The speed at which the large element zooms out.	1.5

Easing

Parameter	Description	Example
thumbnailResizeOut	The easing effect to apply to the thumbnails as they are resizing on their way out. <i>Note: This setting will only work if thumbnailResizeSpeed is not 0.</i>	Cubic.easeOut
thumbnailResizeIn	The easing effect to apply to the thumbnails as they are resizing on their way in. <i>Note: This setting will only work if thumbnailResizeSpeed is not 0.</i>	Cubic.easeOut
itemGoTargetOut	The easing effect to apply to the thumbnail motion as they are moving out of the screen.	Sine.easeOut
itemGoTargetIn	The easing effect to apply to the thumbnail motion as they are moving into the screen.	Sine.easeIn
largePlaneZoomOut	The easing effect to apply to the zoom out animation of the large element.	Cubic.easeOut
largePlaneZoomIn	The easing effect to apply to the zoom in animation of the large element.	Cubic.easeOut

Large elements

Parameter	Description	Example
largeImageHandCursor	Sets whether to display a hand cursor when the mouse is over a large image - indicating to the user that they may click on it.	true
mainItemMouseEnabled	If set to <i>true</i> , clicking on the large element will close it and return to thumbnail view. If the large elements are interactive SWF's, you would set this to <i>false</i> in order to allow the user to click on the controls in the SWF.	true

Thumbnails

Parameter	Description	Example
thumbnailSegmentsHorizontal	Sets the number of horizontal segments to use for the display of the thumbnails. In most cases this should be left at the default setting.	3
thumbnailSegmentsVertical	Sets the number of vertical segments to use for the display of the thumbnails. In most cases this should be left at the default setting.	3

XML

The list of images, SWF's and FLV videos that you wish to display in the tiltGallery must be specified in an XML file. By defining this in an external XML file, you can publish the SWF file once and change the content whenever you wish.

1. Open your favorite plain text editor (for example Notepad on Windows or TextEdit on Mac) and start a new file. *Note: If you are using TextEdit on Mac, choose Format > Make Plain Text*
2. Begin your file with the following line:

```
<?xml version="1.0"?>
```

This is the standard xml declaration.

3. Add the following lines to your xml file (the bold lines are the new additions)

```
<?xml version="1.0"?>  
<content imagePath="images/">  
</content>
```

imagePath = the path to the folder containing the images

4. Add the image tags to your XML file (the bold lines are the new additions).

```
<?xml version="1.0"?>  
<content imagePath="images/">  
  <image thumbnail="1.jpg" large="movie1.flv" width="640" height="360" />  
  <image thumbnail="2.jpg" large="photo2.jpg" />  
  <image thumbnail="3.jpg" large="file3.swf" />  
</content>
```

thumbnail = the filename of the thumbnail image

large = the filename of the corresponding large image, swf or video

width = The width of the flv should be specified (when loading videos only)

height = The height of the flv should be specified (when loading videos only)

5. You can also add an optional parameter to each image which can be read through an ActionScript event when clicking on a large element or thumbnail (the bold lines are the new additions).

```
<?xml version="1.0"?>
<content imagePath="images/">
    <image thumbnail="1.jpg" param="http://www.flashloaded.com"
large="movie1.flv" width="640" height="360" />
    <image thumbnail="2.jpg"param="http://www.fontsforflash.com"
large="photo2.jpg" />
</content>
```

param = the value of the parameter. For example, a URL, frame name or number.

6. Save the XML file to the same folder as your Flash file. In this example, we have given the XML file the name: *gallery.xml*

7. Return to your Flash file. Enter the name and path to the XML file that you just created in the **XML/RSS Source** parameter of the tiltGallery that's on the stage.

Note: If your .swf file will be in a different folder to the HTML file in which it is embedded, you should enter the path to the XML file, relative to the location of the .html file.

8. Press Ctrl+Enter (Win) or Cmnd+Enter (Mac) to test your movie.

Setting component parameters in the XML file

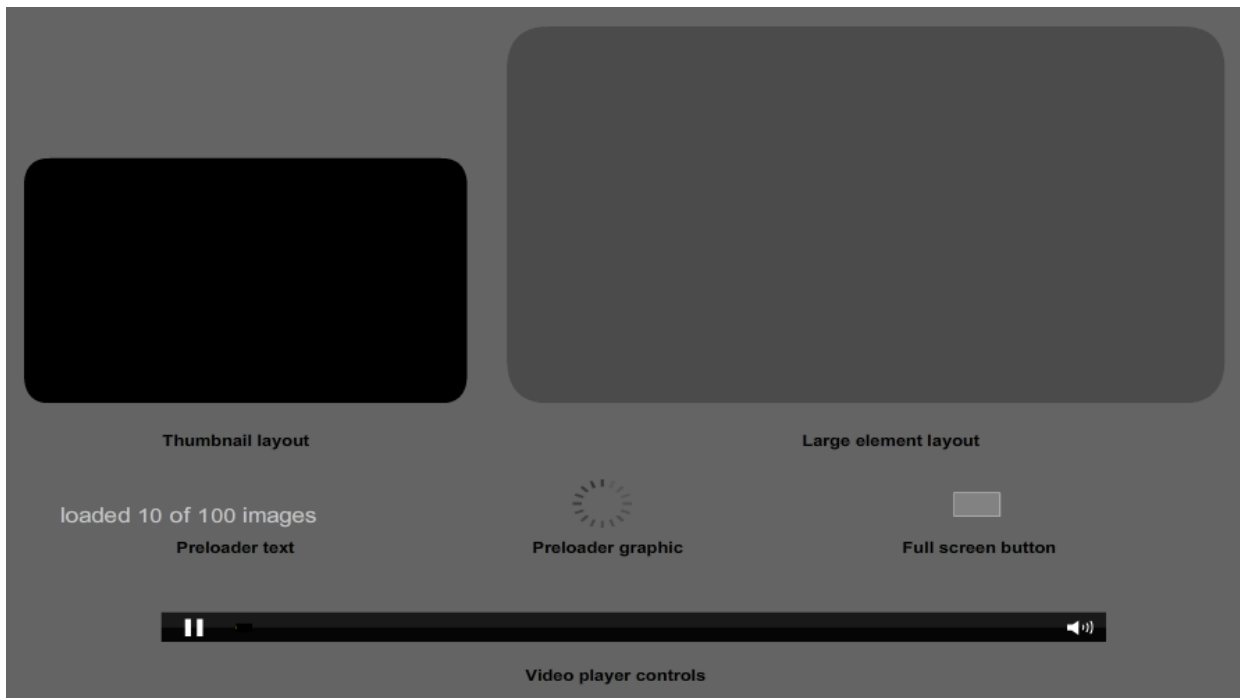
All of the parameters that appear in the Component Inspector can be set in the XML file. Any parameter that is set in the XML will override the value for that parameter that has been set in the Component Inspector. In order to set a parameter in the XML file, simply define the parameter and the value in the *content* tag like this:

```
<content imagePath="images/" rowLimit=5 thumbnailSpacingHorizontal="20"
thumbnailSpacingVertical="20" >
```

Skinning

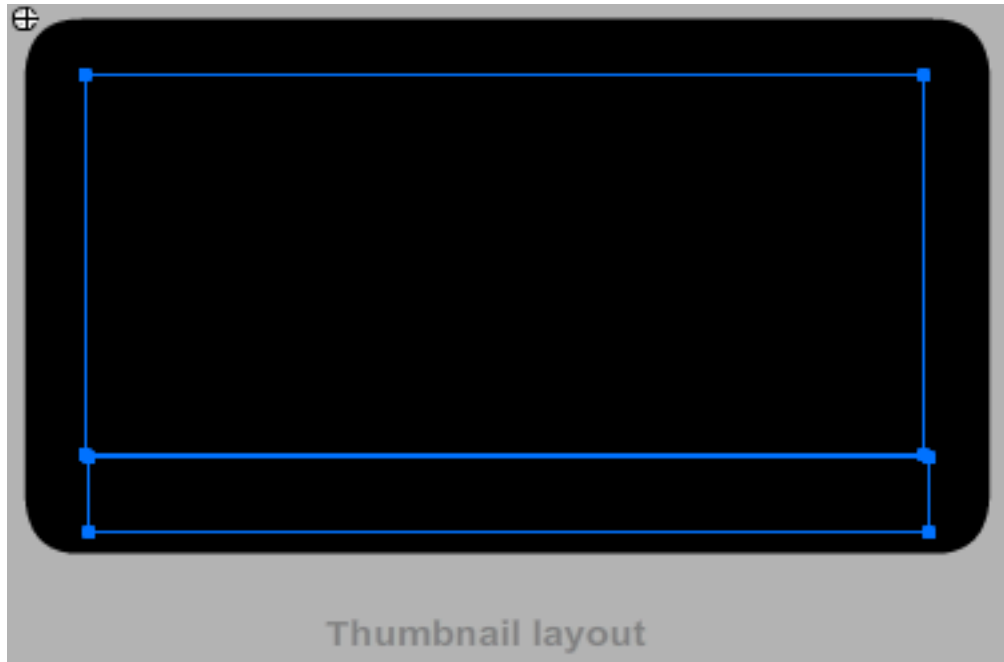
The entire look of the thumbnail and large elements can be skinned. This includes the text formatting and layout for RSS feeds. You can also delete textfields that you do not wish to display for RSS feeds. The preloaders, RSS scrollbar, video player controls and full screen button can also be skinned.

To start skinning, double click on the tiltGallery component that's on the stage. You will now see all of the elements that make up the display of this component. They are located in a layer called "assets". Ensure that this layer is unlocked in order to edit any of these elements.



Skinning the thumbnail appearance

The skin for the image and RSS thumbnails are all contained in the Thumbnail layout movie clip. Double click on this movie clip to edit each of the elements.



You will find the following elements in this movie clip:

A textfield with the instance name *title*

This is the textfield for the title of the RSS entry. You can edit the font properties for this textfield.

A textfield with the instance name *dt*

This is the textfield for the date of the RSS entry. You can edit the font properties for this textfield.

A movie clip with the instance name *video_mc*

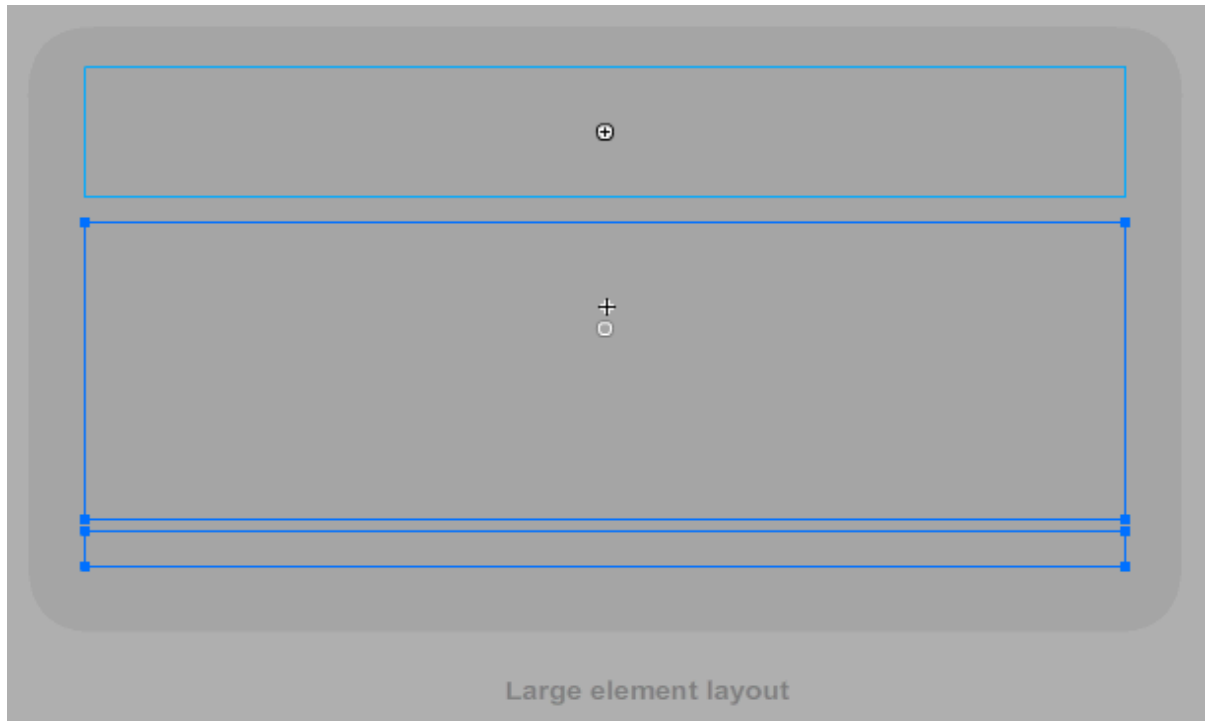
This is a holder movie clip for videos.

A shape for the thumbnail background

Resize this to suit the desired size and proportions of your thumbnails and the thumbnail background.

Skinning the large element appearance

The skin for the large elements and detailed RSS entries are all contained in the Large element layout movie clip. Double click on this movie clip to edit each of the elements.



You will find the following elements in this movie clip:

A movie clip called *title_mc*

This movie clip contains the title textfield for the detailed RSS entries. There are two frames in this textfield, for the mouse up and over positions. The title text can be styled differently for each of these states.

A textfield with the instance name *desc*

This is the textfield for the date of the RSS entry. You can edit the font properties for this textfield.

A textfield with the instance name *pubDate*

This is the textfield for the date of the RSS entry. You can edit the font properties for this textfield.

A movie clip called *rssBG*

This is the background of the detailed RSS entries. You can resize or restyle this background to suit your design.

Displaying RSS feeds

In order to display the content of an RSS feed that's on another domain, the feed must be accessed through a proxy file. This is due to a security feature of the Flash player.

The following two proxy files are included with your download, which you will find in the *proxy* folder:

tiltgalleryproxy.php - For servers that support PHP

tiltgalleryproxy.asp - For servers that support ASP

Upload the appropriate proxy file for your server and enter the path or URL of the proxy file in the ***Proxy file URL*** parameter of the tiltGallery.

Crossdomain file

The Flash Player includes an additional security feature that does not allow scripts to be read across domains. It also does not recognize two URLs for the same site, with and without "www", as being on the same domain. For example, *http://www.yourdomain.com* or not recognized as being the same as *http://yourdomain.com*

If you chose to specify the full URL to the proxy file, for example *http://www.yourdomain.com/tiltgalleryproxy.php*, then you would need a *crossdomain.xml* policy file on your server. Without it, visitors who enter your site without the "www" in the URL will not see the feeds. You might already have a *crossdomain.xml* file on your server.

This is how you would create a *crossdomain.xml* file:

1. Open a new document in a text editor, such as Notepad on Windows or TextEdit on Mac (if you are using TextEdit, select Format > Make Plain Text).
2. Paste the following code into the document:

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM "http://www.macromedia.com/xml/dtds/
cross-domain-policy.dtd">
  <cross-domain-policy>
    <allow-access-from domain="yourdomain.com" />
    <allow-access-from domain="www.yourdomain.com" />
  </cross-domain-policy>
```

3. Replace "yourdomain.com" with your own domain (both with and without the www).
4. Save this file under the name: ***crossdomain.xml***
5. Upload this file to the root of your webserver (usually the same place where the HTML files reside).

For more information on the cross-domain policy file, please see the following Adobe article: http://kb2.adobe.com/cps/142/tn_14213.html

ActionScript events

Events are called whenever the tiltGallery performs the specified action. The component includes an event class called TiltGalleryEvent in the com.flashloaded.TiltGallery package. This can be used, for example, to display a title for a large element.

The event has an item property which holds the param property for elements in which the param value was defined.

The following events are included:

TiltGalleryEvent.THUMB_CLICK

Broadcasted when a thumbnail image is clicked upon.

TiltGalleryEvent.THUMB_MOUSE_OVER

Broadcasted when a thumbnail image is moused over.

TiltGalleryEvent.THUMB_MOUSE_OUT

Broadcasted when the mouse moves off a thumbnail image.

TiltGalleryEvent.LARGE_CLICK

Broadcasted when the current large element is clicked upon.

TiltGalleryEvent.LARGE_MOUSE_OVER

Broadcasted when the current large element is moused over.

TiltGalleryEvent.LARGE_MOUSE_OUT

Broadcasted when the mouse moves off the current large element.

TiltGalleryEvent.LARGE_DESELECTED

Broadcasted when a large element is deselected and returns to thumbnail view.

TiltGalleryEvent.LARGE_LOADED

Broadcasted when a large element has loaded.

TiltGalleryEvent.CAMERA_MOVEMENT_STARTED

Broadcasted whenever the camera starts to move .

TiltGalleryEvent.COMERA_MOVEMENT_COMPLETE

Broadcasted whenever the camera movements has completed.

Example 1- This outputs a trace for the THUMB_CLICK and LARGE_CLICK events:

```
import com.flashloaded.TiltGallery.TiltGalleryEvent;

tgInstance.addEventListener(TiltGalleryEvent.THUMB_CLICK, traceMe);
tgInstance.addEventListener(TiltGalleryEvent.LARGE_CLICK, traceMe);

function traceMe(e:TiltGalleryEvent):void {
    trace("action: " + e.param);
}
```

Performing any action on click

You can have perform any action when clicking on an image. This is done by specifying the a parameter for the image in the parameter element of the XML file, or in the Component Inspector and by reading the parameter in the **LARGE_CLICK** or **THUMB_CLICK** events. This is how you would do this:

1. Add a value in the param element of each image in the XML file. In this example, we're specifying a frame number to go to in param element:

```
<image thumbnail="t1.jpg" param="5" large="1.jpg" />
```

2. Give the tiltGallery that's on the stage an instance name, e.g: *gallery*

3. Type the following ActionScript code on the timeline, in the first frame in which tiltGallery appears:

```
import com.flashloaded.TiltGallery.TiltGalleryEvent;

gallery.addEventListener(TiltGalleryEvent.LARGE_CLICK, traceMe);

function traceMe(e:TiltGalleryEvent) {
    // Perform any action by using any parameters
    trace("action: " + e.param);
}
```

*Note: In this example, you would replace **gallery** with the instance name of your tiltGallery.*

Opening a URL on click

You can have a URL open when clicking on an image. This is done by specifying the URL in the **param** element of the XML file and by reading the URL in either the **LARGE_CLICK** or **THUMB_CLICK** events. This is how you would do this:

1. Add URL's in the param element of each image in the XML file. For example:

```
<image thumbnail="t1.jpg" param="http://www.flashloaded.com" large="1.jpg" />
```

2. Give the tiltGallery that's on the stage an instance name, e.g: *gallery*

3. Type the following ActionScript code on the timeline, in the first frame in which tiltGallery appears:

```
import com.flashloaded.TiltGallery.TiltGalleryEvent;

gallery.addEventListener(TiltGalleryEvent.LARGE_CLICK, largeImageClicked);

function largeImageClicked(e:TiltGalleryEvent) {
    var url:String = e.param;
    if(url != '') {
        var request:URLRequest = new URLRequest(e.param);
        navigateToURL(request, "_blank");
    }
}
```

*Note: In this example, you would replace **gallery** with the instance name of your 3D Stack.*

You should now see that the URL opens when clicking on a large image that has a URL assigned to it.

ActionScript methods

deselectElement

Availability

Flash Player 9

Description

Method; if a large element is currently displayed, it will return to thumbnail view.

Example

```
tiltGalleryInstance.deselectElement();
```

getCurrentState

Availability

Flash Player 9

Description

Method; returns the current state of the tiltGallery view: ***thumbnail*** or ***large***

Example

```
tiltGalleryInstance.getCurrentState();
```

Help

This component is fully supported by the Flashloaded support team through our support forum. You will also find tips and additional information in the forum as well as announcements for version updates:

[tiltGallery Support Forum](#)

Note: In order to post a question in the forum, you will need to [register](#) by creating a username and password. This registration differs from your account login.