

# 3D Stack

User Guide revision 1.1  
[www.flashloaded.com](http://www.flashloaded.com)

# Table of Contents

<b>Installation</b>	<b>3</b>
<b>Getting started</b>	<b>4</b>
<b>XML</b>	<b>6</b>
<b>Component Inspector Parameters</b>	<b>10</b>
General settings	10
Animation settings	11
Slideshow settings	12
Arrow settings	12
Scrollbar settings	12
ThumbnailScroller settings	13
Scrolling settings	14
ImageSettings settings	14
Rotation settings	15
Rollover zoom settings	15
Reflection settings	15
Shadow settings	15
Camera settings	16
Textfield settings	16
<b>Skinning</b>	<b>17</b>
<b>Enabling mouse wheel scrolling for Mac browsers</b>	<b>18</b>
<b>ActionScript events</b>	<b>19</b>
<b>ActionScript properties</b>	<b>21</b>
<b>Performing any action on click</b>	<b>22</b>
<b>Opening a URL on click</b>	<b>23</b>
<b>ActionScript methods</b>	<b>24</b>
<b>Help</b>	<b>28</b>

# 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 3D Stack component.
2. Unzip/extract the 3DStack.zip file that you downloaded. You will find a file called 3DStack.mxp. Double click on this file in order to install the component using Extension Manager.

3D Stack should now be installed in your Flash Components Panel.

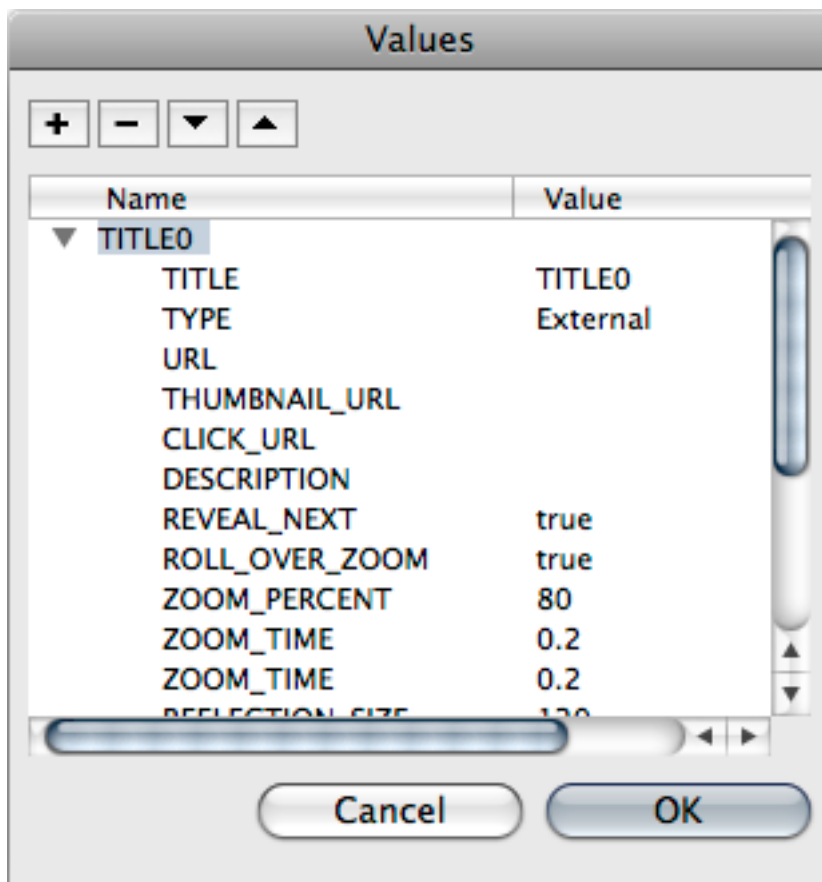
# Getting started

1. Having installed 3D Stack using the Adobe Extension Manager, start a new Flash ActionScript 3 file and save it.
2. Prepare your images:  
3D Stack can use the same or different images for the large and thumbnail images. In order to set this up, create two folders called, for example, *thumbs* and *images* in the same location as your Flash file. Place the thumbnail images in the *thumbs* folder and the large images in the *images* folder. It's much simpler and quicker to set up if the large and thumbnail images have the same filenames.
3. Drag and drop the **3D Stack** component onto the stage. Use the Free Transform tool or the properties panel to resize the component to the desired display area.
4. Click on the component and open the Component Inspector panel (shift +F7).
5. The list of images can be entered either directly into to the Component Inspector, or through an external XML file. An advantage in using an external XML file is that the images can be changed without the need to republish the SWF. It is also much quicker and easier to use an XML file.

Please refer to the [XML](#) section for instructions on creating an XML playlist. If you are using the XML option, enter the name and path to the XML file in the **xml source** parameter.

To enter the image list directly in the Component Inspector:

- a. Double click on the empty square brackets in the **Image collection** parameter. This will open the values window.
- b. Press the + button to define an image file.
- c. Enter details of the image.
- d. Repeat steps b and c to add more file to the image list. Press OK when done.



6. At this stage, you can already test 3D Stack with the default parameters, to ensure that you have set it up correctly. Press Ctrl+Enter (win) or Cmnd+Enter (mac) to test your movie.
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.

*Note: In order for the animations to be smooth it's recommended to set your movie speed to 31 fps.*

# XML

All of the images for 3D Stack can be specified using an XML file instead of the Component Inspector. You can also set all of the parameters in the XML file. By defining the images and parameters in an external XML file, you can publish the SWF file once and change the images or parameters 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" encoding="utf-8"?>
```

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" encoding="utf-8"?>
<stack3d>
  <stackSettings>
  </stackSettings>
  <images>
  </images>
</stack3d>
```

4. Add the following lines between the *images* tags (the bold lines are the new additions):

```
<?xml version="1.0" encoding="utf-8"?>
<stack3d>
  <stackSettings>
  </stackSettings>
  <images>
    <image type="external" URL="images/1.jpg" thumbnailURL=""></image>
    <image type="external" URL="images/2.jpg" thumbnailURL=""></image>
  </images>
</stack3d>
```

Here's an explanation of the tags used in the above examples:

- type:** defines whether the image is an external file or an exported symbol in the library.  
**URL:** the path and filename to the image file or the exported class name.  
**thumbnailURL:** the path and filename of thumbnail image which appears in the thumbnail scroller.

*Note: The **stackSettings** tags must be included in the XML file, even if it does not have any content. The optional content for this tag set will be explained later.*

5. Each image can include additional *optional* information. Here is an examples of optional information that can be added to the image tags, with an explanation of each tag (the bold lines are the new additions):

```

<?xml version="1.0" encoding="utf-8"?>
<stack3d>
  <stackSettings>
  </stackSettings>
  <images>
    <image type="external" URL="images/1.jpg" thumbnailURL="">
      <general title="First title" description="First description"
        revealNext="false" rollOverZoom="true" zoomPercent="80"
        zoomTime="0.2" />
      <reflection size="120" opacity="80" />
      <shadow distance="10" angle="10" opacity="50" color="0x000000" />
      <rotation x="10" y="10" z="0" />
      <clickURL>http://www.flashloaded.com</clickURL>
    </image>
    <image type="external" URL="images/2.jpg" thumbnailURL="">
      <general title="Second title" description="Second description"
        revealNext="false" rollOverZoom="true" zoomPercent="80"
        zoomTime="0.2" />
      <reflection size="120" opacity="80" />
      <shadow distance="10" angle="10" opacity="50" color="0x000000" />
      <rotation x="10" y="10" z="0" />
      <param>5</param>
    </image>
  </images>
</stack3d>

```

Here's an explanation of the tags used in the above examples:

- general:** Contains the following general settings for the image:
- title: A title for the image which can appear in a textfield on the stage.
  - description: A description for the image which can appear in a textfield on the stage.
  - revealNext: Sets whether to reveal the next image which clicking on this image or not.
  - rollOverZoom: Sets whether to zoom the image on rollover or not.
  - zoomPercent: Percentage to zoom when rolling over the image. This value can also be set globally in the component inspector.
  - zoomTime: Speed of the rollover zoom (if rollOverZoom is set to true).
- reflection:** Contains the following reflection settings for the image:
- size: Size of the reflection, in pixels.
  - opacity: Opacity of the reflection.
- shadow:** Contains the following shadow settings for the image:
- distance: Distance between the shadow and the image.
  - angle: Angle of the shadow from the image.
  - opacity: Opacity of the shadow,
  - color: Color of the shadow in the format: 0xHEX-VALUE.
- rotation:** Contains the x, y and z rotation settings for the image.
- clickURL:** The optional URL to open to when clicking on the image.

**param:** An optional parameter that can be read through an ActionScript event when clicking on the image.

6. You can also *optionally* set any of the 3D Stack's parameters in the XML file instead of the Component Inspector. Setting them in the XML file will override the settings for the same parameter in the Component Inspector. Here's an example as to how to do this (the bold lines are the new additions):

```
<?xml version="1.0" encoding="utf-8"?>
<stack3d>
  <stackSettings>
    <general noOfImages="10" mode="linear" angleBetImages="12"
      radius="600" initialX="350" initialY="350" />
    <animation initStyle="firsttolast" initDirection="down"
      initTime="5" initEasing="easeOutSine" mainEasing=""
      mainTime="" />
    <slideShow show="false" timeDelay="1" />
    <userInterface arrows="true" scrollbar="true"
      thumbnailScroller="true" />
    <scrolling clickDrag="true" mouseWheel="true" scrollSpeed="10" />
    <preloaders type="circular" setTextPreloader="true" />
    <imageSettings randomise="false" loop="true" blur="0"
      opacity="50" spacingX="80" spacingY="50" spacingZ="100"
      frontImageWidth="200" frontImageHeight="150"
      maintainRatio="exactfit" border="true" borderColor="0xFEFEFE"
      borderSize="2" useGlobalRotation="true" rotationX="0"
      rotationY="0" rotationZ="0" useGlobalRollOver="true"
      rollOverInterval="" rollOverPercent=""
      useGlobalReflection="true" reflectionOpacity=""
      reflectionSize="" />
    <thumbnailScroller x="20" y="20" width="500" height="50"
      thumbnailWidth="70" thumbnailHeight="50" spacing="10"
      moveAmount="80" direction="horizontal" />
    <scrollbar direction="" x="" y="" />
    <cameraSettings x="" y="" z="" zoom="100" />
  </stackSettings>
  <images>
    .....
  </images>
</stack3d>
```

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

8. Return to your Flash file. Enter the name and path to the XML file that you just created in the **XML path** parameter of the 3D Stack 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.*

9. Press Ctrl+Enter (Win) or Cmnd+Enter (Mac) to test your 3D Stack.

# Component Inspector Parameters

## General settings

Parameter	Description	Example	AS Name
Image collection	This parameter is used only if you are defining the images directly in the Component Inspector as opposed to XML or RSS. Please refer to point #5 of the <a href="#">Getting started</a> section for instructions on using this.		imageCollection
XML path	The path and filename of the XML file containing the image information.	images.xml	xmlPath
RSS path	If you are loading a Flickr feed, you can specify the URL of the Flickr RSS feed here.		rssPath
Position X	The X coordinate for the front image.	350	initialX
Position Y	The Y coordinate for the front image.	350	initialY
Image width	The width of the front image.	200	frontImageWidth
Image height	The height of the front image.	150	frontImageHeight
Last image width	The width of the last visible image in the stack. The images in the stack will change in size gradually between the first and last images.	20	
Last image height	The height of the last visible image in the stack. The images in the stack will change in size gradually between the first and last images.	15	
Image to display	The number of images to display in the stack at a time. Set this number to "0" to display all of the images.	10	noOfImages
Direction	Sets whether to display the stack in linear or circular direction.	LINEAR	mode
Angle between images	For circular mode only: Sets the angular spacing between two contiguous images. This parameter works together with radius.	100	angleBetImages

Parameter	Description	Example	AS Name
Radius	For circular mode only: Sets the radius of the circle around which the images are arranged. The radius and angle are used together to create the desired curvature.	1000	radius
Randomise images	Sets whether to display the images in a random order or not.	false	randomize
Loop images	Sets whether to scroll the images in an endless loop or not.	true	loop
Preloader type	Sets whether to display a text or circular preloader. When set to text, a textual message will appear counting the images loaded. In circular mode, a spinning circle preloader will appear in the middle of each image that is still loading.	TEXT	preloaderType

## Animation settings

Parameter	Description	Example	AS Name
Opening animation order	The order in which the images appear in the stack when opening.	FIRSTTOLAST	initStyle
Opening animation direction	The direction at which the image come in for the opening animation. Options are: <i>DOWN, UP, LEFT, RIGHT</i>	DOWN	initDirection
Opening animation easing	The easing effect to apply to the opening animation.	easeOutBack	initEasing
Opening animation interval	The time taken for the opening animation.	3	initInterval
Animation easing	The easing effect to apply to the general animation.	easeOutSine	mainEasing
Animation interval	The speed (in seconds) for the general animation.	3	mainInterval

## Slideshow settings

Parameter	Description	Example	AS Name
Show slideshow	Sets whether to display the images as an automated slideshow or not.	false	slideShow
slideshow delay	The delay interval (in seconds) before showing the next image when in slideshow mode.	0.5	slideshowDelay

## Arrow settings

Parameter	Description	Example	AS Name
Show arrows	Sets whether to show <i>next</i> and <i>previous</i> image arrows or not.	true	showArrows

## Scrollbar settings

Parameter	Description	Example	AS Name
Show scrollbar	Sets whether to show the scrollbar or not.	true	showScrollbar
Scrollbar X	X coordinate of scrollbar.	80	scrollbarX
Scrollbar Y	Y coordinate of scrollbar.	530	scrollbarY
Scrollbar direction	Sets whether the scrollbar should be horizontal or vertical.	HORIZONTAL	scrollbarDirection

## ThumbnailScroller settings

Parameter	Description	Example	AS Name
Show thumbnail scroller	Sets whether to show the thumbnail scroller or not.	true	showThumbnailScroller
Thumbnail scroller X	X coordinate of thumbnail scroller.	10	thumbnailScrollerX
Thumbnail scroller Y	Y coordinate of thumbnail scroller.	200	thumbnailScrollerY
Thumbnail scroller width	Width of thumbnail scroller (in pixels).	70	thumbnailScrollerWidth
Thumbnail scroller height	Height of thumbnail scroller (in pixels).	290	thumbnailScrollerHeight
Thumbnail width	Width of the thumbnails that appear in the thumbnail scroller (in pixels).	70	thumbnailWidth
Thumbnail height	Height of the thumbnails that appear in the thumbnail scroller (in pixels).	50	thumbnailHeight
Thumbnail move amount	Number of pixels that the thumbnail scroller moves with each arrow click.	60	thumbnailMoveAmount
Thumbnail spacing	Spacing (in pixels) between thumbnails.	10	thumbnailSpacing
Thumbnail scroller direction	Sets whether the thumbnail scroller should be vertical or horizontal.	VERTICAL	thumbnailDirection

## Scrolling settings

Parameter	Description	Example	AS Name
Use drag scroll	Enables or disables click and drag stage scrolling.	true	clickDragScrolling
Drag scroll pixels	Number of pixels that the mouse must be dragged in order to scroll to the next image.	20	scrollSpeed
Use mousewheel scroll	Enabled or disables mouse wheel scrolling.	true	mouseWheelScrolling

## ImageSettings settings

Parameter	Description	Example	AS Name
Aspect ratio	Sets if images should appear at their original size or at the size specified in the <i>image width</i> and <i>image height</i> properties.	ORIGINALSIZE	maintainRatio
Blur	Amount of blur to apply to the most distant image. Images in the stack will be progressively more blurry as they get further to the back.	3	blur
Opacity	Opacity of each image.	80	opacity
Spacing X	Linear mode only: X spacing between each image.	100	spacingX
Spacing Y	Linear and circular mode: Y spacing between each image.	100	spacingY
Spacing Z	Linear and circular mode: Z spacing between each image.	300	spacingZ
Show border	Sets whether to display a border around images or not.	true	border
Border size	Size of border around images (if <i>show border</i> is <i>true</i> ).	2	borderSize
Border color	Color of border around images (if <i>show border</i> is <i>true</i> ).	#EFEFEF	borderColor

## Rotation settings

Parameter	Description	Example	AS Name
Use global rotation	Set this value to <i>true</i> to override any individual image rotation settings with the global settings.	true	useGlobalRotation
Global rotation X	Rotation of images around the X axis.	0	globalRotationX
Global Rotation Y	Rotation of images around the Y axis.	0	globalRotationY
Global rotation Z	Rotation of images around the Z axis.	0	globalRotationZ

## Rollover zoom settings

Parameter	Description	Example	AS Name
Use global rollover zoom	Set this value to <i>true</i> to override any individual image rollover settings with the global settings.	true	globalRolloverZoom
Rollover zoom percentage	Percentage to zoom the image on rollover, with 100 being the regular size.	120	rolloverZoomPercent
Rollover zoom interval	The delay interval after which the image should start zooming.	0.2	rolloverZoomInterval

## Reflection settings

Parameter	Description	Example	AS Name
Use global reflection	Set this value to <i>true</i> to override any individual image reflection settings with the global settings.	true	globalReflection
Reflection size	Size of reflection.	120	reflectionSize
Reflection opacity	Opacity of reflection.	120	reflectionOpacity

## Shadow settings

Parameter	Description	Example	AS Name
Use global shadow	Set this value to <i>true</i> to override any individual image shadow settings with the global settings.	true	globalShadow
Shadow distance	Distance (in pixels) between the shadow and the image.	10	shadowDistance
Shadow angle	Angle at which the shadow is slanted.	10	shadowAngle
Shadow opacity	Opacity of the shadow.	50	shadowOpacity
Shadow color	Color of the shadow.	#000000	shadowColor

## Camera settings

Parameter	Description	Example	AS Name
Camera X	Camera angle around X axis.	0	cameraX
Camera Y	Camera angle around Y axis.	0	cameraY
Camera Z	Camera angle around Z axis.	-1000	cameraZ
Camera zoom	Amount at which the camera should be zoomed.	100	cameraZoom

## Textfield settings

Parameter	Description	Example	AS Name
Title textfield	Instance name of a textfield on the stage in which the title text of the selected image should appear.	title_txt	imageTitleTextfield
Description textfield	Instance name of a textfield on the stage in which the description text of the selected image should appear.	desc_txt	imageDescriptionTextfield

# Skinning

The next and previous buttons for the thumbnail scroller and images navigation as well as the scrollbar and preloader, can all be skinned to match your desired look.

## Skinning the next buttons, previous buttons and scrollbar

Double click anywhere on the 3D Stack component that's on the stage in order to skin these elements. Unlock the *assets* layer and double click on the movie clip of each element that you wish to skin.

## Skinning the text preloader

Double click anywhere on the 3D Stack component that's on the stage. Unlock the *assets* layer. Double click on the loadingTxt textfield in order to change the font, size, color and text.

The text contains variables which are replaced dynamically:

%NUM% = the number of the current image loading.

%TOTAL% = the total number of images to load.

Here's an example of the complete text:

```
Loading %NUM% of %TOTAL% images.
```

## Skinning the animated preloader

Double click on the *preloader* movie clip. You should see the preloaderGraphic on the timeline with a motion tween. You can change the color of this preloader by changing the tint or you can change the symbols and animation entirely.

Note: The preloader movie clip must have a center registration point.

# Enabling mouse wheel scrolling for Mac browsers

The current version of the Flash Player does not natively support mouse wheel scrolling in Mac browsers. We have built a solution for this into 3D Stack. In order to use this solution, you must construct your HTML file like this:

1. Copy the **js** folder, that was included with your download, to the same folder in which your HTML file will reside.
2. Write the following code in the <head></head> section of your HTML file:

```
<script type="text/javascript" src="js/swfobject.js"></script>
<script type="text/javascript" src="js/swfmacmousewheel2.js"></script>
<script type="text/javascript">
    var vars = {};
    var params = { scale:'noScale', salign:'lt', menu:'true' };
    var attributes = { id:'stackObject', name:'stackObject' };
    swfmacmousewheel.registerObject(attributes.id);
</script>
```

3. Write the following code in the body of your HTML file, where you would like the Flash SWF to be located:

```
<script>swfobject.embedSWF("stackexample.swf", "flashContent", "1000",
"600", "9.0.0", "js/expressInstall.swf", vars, params, attributes );</
script>
```

Note: Change the items marked in bold to match your SWF filename, height and width.

This will work when testing online only. You must ensure that you upload the **js** folder with your HTML file.

# ActionScript events

Events are called whenever the 3DStack performs the specified action. The component includes an event class called Stack3DEvent in the com.flashloaded.Stack3D.events package.

The event has an item property which holds the following image properties:

**type:** the type of event that was initiated (e.g: *IMAGE\_CLICKED*, *IMAGE\_OVER*, *IMAGE\_OUT*)

**item:** this contains the ID of the image.

## The following events are included:

### ***Stack3DEvent.XML\_COMPLETE***

Broadcasted when XML has completed loading.

### ***Stack3DEvent.RSS\_COMPLETE***

Broadcasted when RSS has completed loading.

### ***Stack3DEvent.FRONT\_IMAGE\_CLICKED***

Broadcasted when viewer clicks on the font image in the stack.

### ***Stack3DEvent.IMAGE\_CLICKED***

Broadcasted when viewer clicks any large image.

### ***Stack3DEvent.IMAGE\_OVER***

Broadcasted when mouse rolls over a large image.

### ***Stack3DEvent.IMAGE\_OUT***

Broadcasted when mouse rolls out a large image.

### ***Stack3DEvent.IMAGE\_LOADED***

Broadcasted when any of the main images has completed loading.

### ***Stack3DEvent.STACK\_LOADED***

Broadcasted when all the images have been loaded and the 3D Stack is initializing.

### ***Stack3DEvent.THUMBNAIL\_CLICKED***

Broadcasted when thumbnail gets clicked.

### ***Stack3DEvent.SCROLLBAR\_DRAG\_STARTED***

Broadcasted when Scrollbar starts dragging.

### ***Stack3DEvent.SCROLLBAR\_DRAG\_STOPPED***

Broadcasted when Scrollbar stops dragging.

### ***Stack3DEvent.SLIDE\_SHOW***

Broadcasted when slide show brings a new image to the front.

### ***Stack3DEvent.IMAGE\_SELECTED***

Broadcasted when the viewer clicks on an image/thumbnaill in the stack and the image comes to the front.

### ***Stack3DEvent.IMAGE\_DESELECTED***

Broadcasted when the the image in the front leaves the front row.

Example - This outputs a trace for the events:

```
import com.flashloaded.Stack3D.events.*;

Stack3DInstance.addEventListener(Stack3DEvent.IMAGE_CLICKED, traceMe);
Stack3DInstance.addEventListener(Stack3DEvent.IMAGE_OVER, traceMe);
Stack3DInstance.addEventListener(Stack3DEvent.IMAGE_OUT, traceMe);
Stack3DInstance.addEventListener(Stack3DEvent.STACK_LOADED, traceMe);
Stack3DInstance.addEventListener(Stack3DEvent.THUMBNAİL_CLICKED, traceMe);
Stack3DInstance.addEventListener(Stack3DEvent.SCROLLBAR_DRAG_STARTED, traceMe);
Stack3DInstance.addEventListener(Stack3DEvent.SLIDE_SHOW, traceMe);

function traceMe(e:Stack3DEvent):void {
    trace(e.type);
    trace(e.item);
}
```

## **item Object**

The following parameters can be obtained from the item object:

- ID
- TYPE
- URL
- THUMBNAİL\_URL
- TITLE
- DESCRIPTION
- REVEAL\_NEXT
- ROLL\_OVER\_ZOOM
- ZOOM\_PERCENT
- ZOOM\_TIME
- REFLECTION\_SIZE
- REFLECTION\_OPACITY
- SHADOW\_DISTANCE
- SHADOW\_ANGLE
- SHADOW\_OPACITY
- SHADOW\_COLOR
- CLICK\_URL
- ROTATION\_X
- ROTATION\_Y
- ROTATION\_Z
- PARAM

The following events pass the item object parameters:

Stack3DEvent.IMAGE\_RELEASED  
Stack3DEvent.FRONT\_IMAGE\_CLICKED  
Stack3DEvent.IMAGE\_CLICKED  
Stack3DEvent.IMAGE\_OVER  
Stack3DEvent.IMAGE\_OUT  
Stack3DEvent.DOUBLE\_CLICK  
Stack3DEvent.IMAGE\_DESELECTED  
Stack3DEvent.SLIDE\_SHOW  
Stack3DEvent.IMAGE\_SELECTED

## ActionScript properties

You can find the full list of ActionScript properties for each parameter in the **AS Name** column of the [Component Inspector parameters](#) table.

## 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 **FRONT\_IMAGE\_CLICKED** event. 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 type="external" URL="images/2.jpg" thumbnailURL="">
  <param>5</param>
</image>
```

2. Give the 3D Stack that's on the stage an instance name, e.g: *stack*

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

```
import com.flashloaded.Stack3D.events.Stack3DEvent;

stack.addEventListener(Stack3DEvent.FRONT_IMAGE_CLICKED,
onFrontImageClicked);

function onFrontImageClicked(e:Stack3DEvent) {
    // Perform any action by using any parameters
    trace(e.item.PARAM);
}
```

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

## 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 **IMAGE\_CLICKED** or **FRONT\_IMAGE\_CLICKED** 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 type="external" URL="images/2.jpg" thumbnailURL="">
  <param>http://www.flashloaded.com<param>
</image>
```

2. Give the 3D Stack that's on the stage an instance name, e.g: *stack*
3. Type the following ActionScript code on the timeline, in the first frame in which 3D Stack appears:

```
import com.flashloaded.Stack3D.events.Stack3DEvent;

stack.addEventListener(Stack3DEvent.FRONT_IMAGE_CLICKED,
onFrontImageClicked);

function onFrontImageClicked(e:Stack3DEvent) {
    var url:String = e.item.PARAM;
    var request:URLRequest = new URLRequest(url);
    navigateToURL(request, "_blank");
}
```

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

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

# ActionScript methods

## addImage()

### Availability

Flash Player 9

### Description

Method; Adds an ImageObject object with the specified parameters. This image appears at the end of the stack. Returns nothing.

### Syntax

```
3DStackInstance.addImage(imageObj:Object);
```

### Example

```
var imageObj :Object = new Object();
imageObj["type"] = "external";
imageObj["url"] = "images/1.jpg";
imageObj["thumbnailUrl"] = "images/thumbs/1.jpg";
imageObj["title"] = "test";
imageObj["description"] = "test";
imageObj["revealNext"] = "true";
imageObj["rollOverZoom"] = "true";
imageObj["zoomPercent"] = "80";
imageObj["zoomSpeed"] = "0.2";
imageObj["reflectionSize"] = "120";
imageObj["reflectionOpacity"] = "80";
imageObj["shadowDistance"] = "10"
imageObj["shadowAngle"] = "10";
imageObj["shadowOpacity"] = "50";
imageObj["shadowColor"] = "0x000000";
imageObj["rotationX"] = "0";
imageObj["rotationY"] = "0";
imageObj["rotationZ"] = "0";
imageObj["clickUrl"] = "http://www.flashloaded.com";
3DStackInstance.addImage(imageObj);
```

## addImageAt()

### Availability

Flash Player 9

### Description

Method; Adds at given index an ImageObject object with the specified parameters. Returns nothing.

### Syntax

```
3DStackInstance.addImageAt(imageObj:Object, index:Number);
```

### Example

```
3DStackInstance.addImageAt(imageObj, 3);
```

## removeImage()

### Availability

Flash Player 9

### Description

Method; Removes an ImageObject object. This method can only remove the image at the end of stack. Returns nothing.

### Syntax

```
3DStackInstance.removeImage();
```

### Example

```
3DStackInstance.removeImage();
```

## removeImageAt()

### Availability

Flash Player 9

### Description

Method; removes an ImageObject object at given index. Returns nothing.

### Syntax

```
3DStackInstance.removeImageAt(index: Number);
```

### Example

```
3DStackInstance.removeImageAt(3);
```

## nextImage()

### Availability

Flash Player 9

### Description

Method; Scrolls the stack to show next image. Returns nothing.

### Syntax

```
3DStackInstance.nextImage( );
```

### Example

```
3DStackInstance.nextImage( );
```

## prevImage()

### Availability

Flash Player 9

### Description

Method; Scrolls the stack to show previous image. Returns nothing.

### Syntax

```
3DStackInstance.prevImage( );
```

### Example

```
3DStackInstance.prevImage( );
```

## getCurrentImageId()

### Availability

Flash Player 9

### Description

Method; Returns the ID of the current image.

### Syntax

```
3DStackInstance.getCurrentImageId( );
```

### Example

```
trace(3DStackInstance.getCurrentImageId( ));
```

## gotoImage()

### Availability

Flash Player 9

### Description

Method; Scrolls the stack to show image specified by index. Returns nothing.

### Syntax

```
3DStackInstance.gotoImage(index:Number);
```

### Example

```
3DStackInstance.gotoImage(4);
```

## playSlideshow()

### Availability

Flash Player 9

### Description

Method; Starts scrolling the stack continuously in an infinite loop. Uses SlideshowDelay as the interval to scroll to the next image. Returns nothing.

### Syntax

```
3DStackInstance.playSlideshow();
```

### Example

```
3DStackInstance.playSlideshow();
```

## stopSlideshow()

### Availability

Flash Player 9

### Description

Method; Stops playing slideshow if it is already playing. Returns nothing.

### Syntax

```
3DStackInstance.stopSlideshow();
```

### Example

```
3DStackInstance.stopSlideshow();
```

# 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: [3D Stack 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.*