

# TN2Gallery

User Guide revision 1.4

[www.flashloaded.com](http://www.flashloaded.com)

# Table of Contents

<b>Installation</b>	<b>3</b>
<b>Getting Started</b>	<b>4</b>
<b>Using XML</b>	<b>5</b>
<b>Component Inspector Parameters</b>	<b>11</b>
Thumbnailer2	11
ThumbImager2	14
TN2Controls	17
<b>Important HTML settings</b>	<b>19</b>
<b>Using the TN2Gallery components with the fCMSPro</b>	<b>20</b>
Customizing the preloader	21
Customizing the slideshow timer	21
Starting a slideshow automatically	21
Setting relative location for the gallery, tags and slideshow buttons	23
Setting relative location for the slideshow timer	24
<b>ActionScript Reference</b>	<b>26</b>
ThumbImager2 Methods	26
ThumbImager2 Events	27
Thumbnailer2 Properties	30
Thumbnailer2 Methods	32
Thumbnailer2 Events	36
TN2Controls Properties	42
TN2Controls Methods	48
TN2Controls Events	50
<b>Help</b>	<b>51</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 TN2Gallery components.
2. Unzip/extract the TN2Gallery.zip file that you downloaded. You will find a file called TN2Gallery.mxp. Double click on this file in order to install the components using Extension Manager.

The Thumbnailer2, ThumbImager2 and TN2Controls components should now be installed in your Flash Components Panel.

# Getting Started

1. First you will need to prepare three image files for each image that you wish to display, one for the thumbnail, one for the preview and one for the main image. Save the images as gifs, pngs or jpgs. Jpgs are usually best for photographs.
2. Open Flash and start a new file. If you are using Flash CS3 make sure you start a new *Actionscript 2* file.
3. Press *Ctrl+F7* (Windows) or *Cmd+F7* (Mac) to open the components panel and drag a copy of the **Thumbnailer2** out of the **TN2Gallery** folder onto the stage. With the component still selected press *Ctrl+F3* (Windows) or *Cmd+F3* (Mac) to open the properties panel and give it the instance name *tn*.
4. Return to the Components panel and drag a copy of the **ThumbImager2** out of the **TN2Gallery** folder and onto the stage. With the component still selected, open the properties panel and give it the instance name *ti*.
5. Select the **Thumbnailer2** component that's on the stage and press *Alt+F7* to open the Component Inspector. Double-click the value box of the **Data** parameter to open its dialog box.
6. Click the plus button to add a thumbnail to the Thumbnailer2. Enter a title for the image. Next, in the **image\_src** field enter the name of a main image you prepared in step 1. Then, in the **thumb\_src** field enter the name of the thumbnail you prepared. Finally, in the **preview\_src** field enter the name of the preview image that you prepared.
7. Repeat step 6 for each set of images that you wish to display.
8. Click **OK** to close the dialog box.
9. With the **Thumbnailer2** instance still selected locate the Preview parameter and set it to Bottom.
10. Save and test your file.

# Using XML

We recommend that you use an external XML file with the TN2Gallery because it allows for greater flexibility and allows you to modify your file without republishing the SWF file.

1. First you will need to prepare three image files for image you wish to display, one for the thumbnail, one for the preview and one for the main image. Save the images as *gifs*, *pngs* or *jpgs*. Jpgs are usually best for photographs.
2. Open your favourite plain text editor, such as Notepad on Windows or TextEdit on Mac. Start a new file and save it as 'thumbnailer.xml' (without the quotes).

Enter the following line of code:

```
<?xml version="1.0" encoding="utf-8"?>
```

This is the standard opening line for an XML file.

3. Add the following two lines to create an XML element named 'tn2' (the new lines are shown in bold):

```
<?xml version="1.0" encoding="utf-8"?>  
<tn2>  
</tn2>
```

4. Next, add a new element named 'gallery' inside the 'tn2' element (again, new lines highlighted in bold):

```
<?xml version="1.0" encoding="utf-8"?>  
<tn2>  
  <gallery>  
  </gallery>  
</tn2>
```

5. The next step is to add a title for the 'gallery' like so:

```
<?xml version="1.0" encoding="utf-8"?>  
<tn2>  
  <gallery>  
    <title>My Gallery</title>  
  </gallery>  
</tn2>
```

6. Add a **description** element inside the **gallery**, below the **title**:

```
<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
    <description>My First XML Driven Gallery</description>
  </gallery>
</tn2>
```

7. The next step is to add a **file\_root** element. This element contains the path to the directory that contains your image files. In order to load images from the same directory set it to `./`:

```
<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
    <description>My First XML Driven Gallery</description>
    <file_root>./</file_root>
  </gallery>
</tn2>
```

8. Now add a **thumb\_src** element. This element is used to specify a thumbnail for the gallery. The image should be specified relative to the path specified in the **file\_root** element:

```
<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
    <description>My First XML Driven Gallery</description>
    <file_root>./</file_root>
    <thumb_src>gallery_thumb.jpg</thumb_src>
  </gallery>
</tn2>
```

9. The final element to add to your **gallery** element is an **images** element. This element is used to contain the images you will display in the Thumbnailer2:

```
<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
    <description>My First XML Driven Gallery</description>
    <file_root>./</file_root>
    <thumb_src>gallery_thumb.jpg</thumb_src>
    <images>
    </images>
  </gallery>
</tn2>
```

10. To add an image to your gallery you need to add an **image** element to the **images** element like so:

```
<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
    <description>My First XML Driven Gallery</description>
    <file_root>./</file_root>
    <thumb_src>gallery_thumb.jpg</thumb_src>
    <images>
      <image>
      </image>
    </images>
  </gallery>
</tn2>
```

11. The **image** element contains several nodes that describe the image. The minimum requirement for the **image** node is to contain the **image\_src** node which is the path to image with highest resolution that will be displayed using the thumbImager. The location is specified relative to the **file\_root** element you defined in step 7. If the other nodes do not exist, the Thumbnailer2 component will automatically create the values for the **thumb\_src** and **preview\_src** nodes. For example, if you define this:

```
<image_src>image.jpg</image_src>
```

The values that are created automatically will be as if you had defined this:

```
<thumb_src>thumbs/image_1.jpg</thumb_src>
<preview_src>thumbs/image_2.jpg</preview_src>
```

If you define the **thumb\_src** and **preview\_src** nodes, these values will not be created automatically.

```
<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
```

```

    <description>My First XML Driven Gallery</description>
    <file_root>./</file_root>
    <thumb_src>gallery_thumb.jpg</thumb_src>
    <images>
        <image>
            <title>My First Image</title>
            <image_src>image.jpg</image_src>
        </image>
    </images>
</gallery>
</tn2>

```

12. If you define the **title** and **description** nodes they will be used to display the image info when using the TN2Controls component.

```

<?xml version="1.0" encoding="utf-8"?>
<tn2>
    <gallery>
        <title>My Gallery</title>
        <description>My First XML Driven Gallery</description>
        <file_root>./</file_root>
        <thumb_src>gallery_thumb.jpg</thumb_src>
        <images>
            <image>
                <title>My First Image</title>
                <description>Image desc</description>
                <image_src>image_1.jpg</image_src>
            </image>
        </images>
    </gallery>
</tn2>

```

13. There are five more optional elements that can be defined inside the 'image' element, they are **description**, **image\_action**, **thumb\_action**, **thumb\_src** and **preview\_src**. If used they are passed to the component's event handler functions. You may choose to leave them empty or omit them altogether if you do not wish to use them. If you define the **thumb\_action** node, clicking on corresponding thumbnail will try to open the value of node as a url. The example below will open <http://flashloaded.com> in new browser window. Defining the **image\_action** node will open the value of the node as url when a user clicks on the ThumbImager2 component.

```

<?xml version="1.0" encoding="utf-8"?>
<tn2>
    <gallery>
        <title>My Gallery</title>
        <description>My First XML Driven Gallery</description>
        <file_root>./</file_root>
        <thumb_src>gallery_thumb.jpg</thumb_src>
        <images>
            <image>
                <title>My First Image</title>

```

```

        <description>This is my first image</description>
        <image_src>image_1.jpg</image_src>
        <thumb_action>http://flashloaded.com</thumb_action>
        <image_action>http://adobe.com</image_action>
        <thumb_src></thumb_src>
        <preview_src></preview_src>
    </image>
</images>
</gallery>
</tn2>

```

*Note: You can add more nodes inside the 'image' tag. The value of each defined node will be passed to component's event handler functions. For the purpose of this tutorial leave them empty or omit them altogether.*

14.Repeating steps 10 to 14, add an **image** element and its contents for each of the images you wish to add to your Thumbnailer2:

```

<?xml version="1.0" encoding="utf-8"?>
<tn2>
  <gallery>
    <title>My Gallery</title>
    <description>My First XML Driven Gallery</description>
    <file_root>.</file_root>
    <thumb_src>gallery_thumb.jpg</thumb_src>
    <images>
      <image>
        <title>My First Image</title>
        <image_src>image_1.jpg</image_src>
      </image>
      <image>
        <title>My Second Image</title>
        <image_src>image_2.jpg</image_src>
      </image>
      <image>
        <title>My Third Image</title>
        <image_src>image_3.jpg</image_src>
      </image>
    </images>
  </gallery>
</tn2>

```

15.Save your XML file and close your plain text editor.

16.Open Flash and start a new Flash file. If you are using Flash CS3, you should choose to start a new *Actionscript 2.0* file.

17.Press *Ctrl+F7* (Windows) or *Cmd+F7* (Mac) to open the Components panel and drag a copy of the **Thumbnailer2** out of the **TN2Gallery** folder onto the stage. With the component still selected

press *Ctrl+F3* (Windows) or *Cmd+F3* (Mac) to open the properties panel and give it the instance name *tn*.

18. Return to the Components panel and drag a copy of the **ThumbImager2** out of the **TN2Gallery** folder and onto the stage. With the component still selected open the properties panel and give it the instance name *ti*.

19. Select the **Thumbnailer2** component that's on the stage and press *Alt+F7* to open the Component Inspector. Locate the *XML File* parameter and enter the name of the XML file you saved in step 15.

20. Save your Flash file in the same directory as your XML file and test it.

# Component Inspector Parameters

## Thumbnailer2

Parameter	Description	Example
Data	Thumbnail Data. If no data is set, the component will try to acquire data from an XML file or the fCMS.	
- Orientation	Sets the direction in which the thumbnails will be displayed.	horizontal
- Thumbnail Distance	The distance between thumbnails measured in pixels.	2
- Center on Load	If 'true' and the thumbnailer area is bigger than total width of the thumbnails, the component will center them inside the component's area.	true
- Display on Load	The index number of the thumbnail to be triggered automatically when thumbnails are loaded.	
- Display While Loading	If 'false' the Thumbnailer component is hidden until all thumbnails are loaded.	true
Use Numbers	If set to 'true' numbers will be used instead of thumbnails.	false
- Over Color	The color of a number when the mouse is over it.	#CCCCCC
- Selected Color	The color of a number when it is selected.	#CCCCCC
AutoAlign	If the value is set to "true", the thumbnailer will auto-position itself on every stage size change (bottom of the screen when orientation is "horizontal" or at right edge of the screen when orientation is "vertical").	false
Autosize	The thumbnailer can be set to resize automatically in length or height (depending on orientation) when there is enough space to do so.	Disabled

Parameter	Description	Example
- Difference	When Autosize is set to true, increasing this value will decrease the size of the thumbnail. For example, if Autosize is set to true and the thumbnail is positioned at X=20, you would set the difference to 20 in order to keep the thumbnail centered.	0
- Movement Mode	Sets movement type of the thumbnail. Options available: <b>Default:</b> thumbnails are scrollable with the mouse movement <b>Endless:</b> thumbnails scroll in an endless loop. <b>Combined:</b> thumbnails are scrollable with buttons and mouse movement <b>Slideshow:</b> the next thumbnail and large image is shown automatically at the set interval <b>Static:</b> thumbnails scroll using arrows	Default
- Default Acceleration	The amount of acceleration applied to the thumbnails' movement.	10
- Default Buffer Width	in "Default Mode", the size of the 'dead area' on the thumbnail's edges where mouse position is not relevant.	50
- Default Slowdown Ratio	in "Default Mode", when the mouse cursor is outside of the thumbnail's area, this value determines the time needed for the movement to stop (slowdown).	0
- Delay	The delay until "Endless Mode" movement starts, or the delay until transition in "Slideshow Mode".	1000
- Endless Speed	The speed of movement in "Endless Mode".	20
Preview	The position of the preview movieclip, relative to the position of thumbnail.	none
- BG Color	The color of the preview background.	white
- BG Alpha	The opacity of the preview background.	100
- BG Size	The size of the preview border.	2
- Duration	The duration of the preview box animation in milliseconds.	150

Parameter	Description	Example
- Thumbnail Size	The size of the thumbnails to use in preview mode. Dependent on fCMS thumbnail configuration (0 is the original size etc).	2
Background	Enables or disables the background of thumbnailer. This must be set to "true" if <b>AutoHide</b> is enabled in the <b>TN2Controls</b> component.	Disabled
- Size	The margin sizes of the background relative to the thumbnailer's area. <i>Note: If you have set AutoAlign to "true", you can use this setting to add padding between the thumbnailer and the side of the stage.</i>	{60,2,30,2}
Behaviour	Predefined thumbnail behavior modes.	ShaderBorder
- Shader Color	The color of shaded thumbnails.	Black
- Shader Alpha	The alpha value of shader color.	20
- Shader Fade Out	The number of milliseconds that it takes the Shader to fade in.	10
- Shader Fade In	The number of milliseconds that it takes the Shader to fade out.	10
- Border Color	The color of the border for non-selected thumbnails.	Black
- Border Over	The color of the border for thumbnails when the mouse cursor is over them.	Black
- Border Selected	The color of the border when the thumbnail is selected.	White
- Border Size	The size of the thumbnail's border in pixels.	2
XML File	The path and name of the external XML file if you intend to set parameters from an XML file instead of the parameters panel.	
Auto Load Gallery	If set to "true", first gallery from xml definition will be loaded.	true

Parameter	Description	Example
FCMS Server Path	The path to the folder on the server containing the TN2Gallery files ( fCMSBackend ).	http:// www.flashload ed.com/ flashcompone nts/tn2gallery/ fCMSBackend /
- File root	The path to the folder containing uploaded images relative to the server path (the same value as in the config.xml file).	images/
- Gallery ID	The unique identifier (id) of the gallery to load after initialization.	0
- Tag ID	The unique identifier (id) of the tag to load after initialization.	0
- Thumbnail Size	The size of the thumbnails to use. Dependent on the fCMS thumbnail configuration (0 is the original size etc).	1
- Source Field	For use with fCMSPro only. The name of the document field with thumbnail information.	
- Admin Hide	For use with fCMSPro only. If set to "true", the thumbnailer will be hidden in administration mode.	true

## ThumbImager2

Parameter	Description	Example
- Horizontal Align	The horizontal alignment of the loaded image.	Center
- Vertical Align	The vertical alignment of the loaded image.	Middle

Parameter	Description	Example
Container	The area within which loaded images are contained. Area - the area of component defined in the Flash IDE. No masking is applied. <b>MaskedArea</b> - the same as Area, but the image will be masked. <b>Stage</b> - the Flash stage. <b>Nailer Stage</b> - the Flash stage without the space used by the Thumbnailer. <b>Stage Cover</b> - the whole stage is covered and the image is loaded on top of the cover. <b>FullScreen Cover</b> - the same as <i>Stage Cover</i> but in fullscreen mode.	Masked Area
- FS Align	The stage alignment when in fullscreen mode.	TopLeft
- FS Container	The area within which loaded images are contained when in fullscreen mode.	Stage
- Cover Alpha	The alpha value of the cover in 'Stage Cover' and 'FullScreen Cover' container mode.	60
- Cover Color	The color of the cover in 'Stage Cover' and 'FullScreen Cover' container mode.	black
- Cover Blur Amount	If the blur amount is bigger than 0, the current stage snapshot will be used as the cover and blurred.	0
Autosize	If 'true', the component will resize the loaded image according to the container size.	Always
- Maximum Zoom	The maximum magnification of the loaded image.	1.3
Transition	The type of image transition. Set this toe BGResize of the images are of different sizes.	BGResize
- Duration	The duration of transition in milliseconds.	800
- BG Color	The color of the background when 'BGResize' transition is selected.	White
- BG Size	The size of image border (the difference between the background and image size).	4
- BG Resize Duration	The duration of background resize in milliseconds.	200

Parameter	Description	Example
Percentage Variable	Reference to the variable in which the loading percentage is stored.	MyVariable
Percentage Text	The structure of the text that will be displayed through the Percentage Variable. The default format is 'Loading XX%' where XX represents a percentage. The rest of the text can be customized.	Loading XX %
fCMS Image Size	The size of the image to load. Dependent on fCMS thumbnail configuration (0 is original size etc).	0

## TN2Controls

Parameter	Description	Example
Controls	Turns controls on (value 1) or off (value 0).	1
Image Controls	Defines the position of the next and previous image controls.	Image Sides
Page Controls	Defines the position of the next and previous page controls.	Thumbnailer Sides
Fullscreen	Defines the position of the fullscreen control.	Image Corner
Timer	Defines the position of the timer animation used in the slideshow mode.	Image Corner
Preloader	Defines the position of the preloader animation.	Image Center
Image Info	Defines the position of the image information.	TopLeft
- Info Button	Defines the position of the info control.	TopRight
- Show Info	Defines whether image information will be displayed when the mouse cursor is over the info control, or when it is released over the info control.	Release
Button Distance	The distance between the controls (in pixels).	4
AutoHideImagerControls	If set to "true", controls dependant on image position will be hidden on mouse roll out.	true
AutoHide	Enables the autohide option. Requires the TN2Gallery background option to be enabled.	Disabled
- Hide Alpha	The alpha value of the hidden controls.	0
- Hide Delay	Duration of time the cursor is out of the background area before until hiding controls begins.	1000

Parameter	Description	Example
Tooltip	If 'true', tooltips will appear over the buttons.	true
- Tip Text	Defines the tooltip text for each of the buttons.	
Gallery	Defines the active image navigation buttons.	Both
- BG Color	The background color of the tag text field.	Black
- BG Alpha	The alpha value for the background color of the tag text field.	60
- Grid Size	Defines the number of rows and columns in the gallery view. The default value (2 x 3) will display 2 rows across 3 columns.	2x3
- Tag Over Color	The color of the tag text when the mouse cursor hovers over it.	red
- AutoDisplay	If set to "true", gallery grid or tag cloud will be displayed automatically.	true

## Important HTML settings

In order to fullscreen mode to work correctly in a browser, the HTML file in which the Flash SWF containing the TN2Gallery is embedded must be edited to ensure that **scale** is set to **noscale** and **allowFullScreen** is set to **true**.

For example:

```
<script language="javascript">
  if (AC_FL_RunContent == 0) {
    alert("This page requires AC_RunActiveContent.js.");
  } else {
    AC_FL_RunContent(
      .....
      .....
      'scale', 'noscale',
      .....
      .....
      .....
      .....
    );
  }
</script>
<noscript>
  <object .....>
    .....
    <param name="    allowFullScreen" value="true" />
    .....
    <embed .....    allowFullScreen="true" ...../>
  </object>
</noscript>
```

# Using the TN2Gallery components with the fCMSPro

In order to use the TN2Gallery components with the fCMSPro, you must have the fCMSPro version 1.3.0 or later installed.

1. Drag the **fFile** component into the same movieclip that is controlled by **fTemplate**. Give its field a name, set the number of files (images) that you want to allow for each document and set **visibility** to *for admin*.
2. Drag the **Thumbnailer2** component from TN2 Gallery component folder into the same movieclip. Set the **fCMS -> Source Field** parameter of the Thumbnailer2 to the field name of **fFile** component and leave the *Admin Hide* parameter set to *true*.
3. Ensure that auto thumbnail creation is enabled by renaming *fCMSBackend/config/thumbs-example.xml* to **fCMSBackend/config/thumbs.xml**

You can now export the SWF and define some images as content for the fFile field. After saving, the Thumbnailer2 will display those images.

# TN2Gallery Customization

All of the TN2 controls, preloader and timer can be customized. To do this in Flash CS3 or Flash 8, open **Window->Common Libraires->TN2Assets** and drag the **TN2Assets** folder to the library of your current .fla, then double click on the desired movieclip to edit it.

The example file *gallery.fla* (which is included with your download) contains different control button skins. To use them instead of the default buttons, open *gallery.fla* and drag the TN2Assets folder onto the stage, then select the newly added movieclips and delete them from the stage.

## Customizing the preloader

The **TN2\_preloader** movieclip needs to have the *onProgress* function defined. That function will be called whenever the percentage of currently loaded image changes. For example, this is how you would create a simple textfield that will display the loaded percentage:

1. Create a dynamic text field inside the *TN2\_preloader* movieclip and give it an instance name *percentage\_txt*
2. Add the following code on the timeline of the *TN2\_preloader* movieclip:

```
function onProgress( per:Number ) {  
    percentage_txt.text = per + "%";  
}
```

## Customizing the slideshow timer

The **TN2\_timer** movieclip needs to have the *onTimer* function defined. This function will be called as the time of the slideshow timer passes. Two parameters are passed to function: **time passed** and **total time**.

## Starting a slideshow automatically

1. Add the following code to the timeline:

```
var listenerObject = new Object();  
listenerObject.thumbsLoaded = function( eventObject ){  
    myTN2Controls.slideshow.onRelease();  
}  
myThumbnailer2.addEventListener("thumbsLoaded", listenerObject )
```

## Setting the Color of the Title and Description

In order to set the color of the title and description you must use the following code on your timeline:

```
var currentDesc = "";
var currentTitle = "";
myThumbImager2.loadHandler = function()
{
  myTN2Controls.imgInfo.title_txt.htmlText = "";
  myTN2Controls.imgInfo.description_txt.htmlText = "";
}

this.onEnterFrame = function()
{
  myTN2Controls.imgInfo.title_txt.htmlText = currentTitle;
  myTN2Controls.imgInfo.description_txt.htmlText = currentDesc;
}

var tnailerListener = new Object();
tnailerListener.release = function( eventObject ){
  currentTitle = eventObject.data.title;
  currentDesc = eventObject.data.description;
}
tnailer.addEventListener("release", tnailerListener )
```

Note that this assumes you are using the instance names 'myThumbImager2' and 'myTN2Controls' (without the quotes).

# Customizing the controls

The buttons in the TN2Controls as well as the information and full screen buttons can be positioned in any location relative to one of the TN2Gallery components. The slideshow timer can also be positioned in this way.

## Setting relative location for the gallery, tags and slideshow buttons

By default, the buttons for the TN2Controls component will remain wherever they are placed on the stage. This means that in fullscreen mode, these buttons may appear in the middle of the screen. If they are positioned in the top left corner of the stage, you should not encounter this problem in full screen mode.

In this tutorial, we explain how to ensure that the gallery, tags and slideshow components will always appear in the relative location to the thumbNailer.

1. Give the ThumbNailer2 component that's on your stage an instance name of *tnailer*.
2. Give the TN2Controls component that's on your stage an instance name of *tn2ctrls*.
3. Click on the first frame in which the thumbNailer and TN2Controls components appear and apply the following ActionScript code:

*// This positions the gallery button 80px to the left and 8px down relative to the ThumbNailer*

```
tn2ctrls.definePosition( "gallery", {  
    ref:tnailer,  
    horizontal:"left",  
    vertical:"top",  
    x: -80,  
    y: 8  
});
```

*// This positions the tags button 80px to the left and 8px down relative to the ThumbNailer*

```
tn2ctrls.definePosition( "tags", {  
    ref:tnailer,  
    horizontal:"left",  
    vertical:"top",  
    x: -80,  
    y: 8  
});
```

*// This positions the slideshow button 54px to the left and 8px down relative to the ThumbNailer*

```
tn2ctrls.definePosition( "slideshow", {  
    ref:tnailer,  
    horizontal:"left",  
    vertical:"top",  
    x: -54,  
    y: 8  
});
```

## Setting relative location for the slideshow timer

The slideshow timer can be positioned relative to any of the TN2Gallery components (besides the preset positions available).

In this tutorial, we explain how to position the slideshow component in the bottom left corner of the large image.

1. Give the ThumbImager2 component that's on your stage an instance name of *timager*.
2. Give the TN2Controls component that's on your stage an instance name of *tn2ctrls*.
3. Click on the first frame in which the thumbNailer and TN2Controls components appear and apply the following ActionScript code:

```
tn2ctrls.definePosition( "timer", {
    ref:timager,
    horizontal:"left",
    vertical:"bottom",
    x: 3,
    y: -28
});
```

## Setting a uniform tag size

1. Give the TN2Controls component that's on your stage an instance name of *tn2ctrls*.
2. Click on the first frame in which the TN2Controls components appear and apply the following ActionScript code:

```
lo = new Object();
lo.controlClick = function( eventObject ){

    if ( eventObject.id == "tags" ) {
        var htmlt = tn2ctrls.tagsField._txt.htmlText;
        htmlt = htmlt.split('SIZE="20"').join('SIZE="12"');
        htmlt = htmlt.split('SIZE="16"').join('SIZE="12"');
        tn2ctrls.tagsField._txt.htmlText = htmlt;
    }
}
tn2ctrls.addEventListener("controlClick", lo );
```

## Adding and Removing Controls

1. Select the TN2Controls component on the stage.
2. Open the component inspector panel
3. For each control you wish to appear set a value of 1, for any you wish hidden set a value of 0.

# ActionScript Reference

## ThumbImager2 Methods

ThumbImager2.[load](#)

### Availability

Flash Player 8

### Usage

```
myThumbImager2.load( path:String )
```

### Parameters

path - String; path to the image.

### Returns

Nothing.

### Description

Method; loads Image.

### Example

The following code will load the specified image:

```
myThumbImager2.load( "myImage.jpg" );
```

# ThumbImager2 Events

ThumbImager2.[progress](#)

## Availability

Flash Player 8

## Usage

```
listenerObject = new Object();

listenerObject.progress = function( eventObject ){
    // insert your code here
    trace( value + "%");
}
myThumbImager2.addEventListener("progress", listenerObject )
```

## Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

value - Number. Percentage loaded.

mc - Movieclip. Reference to the movieclip that is loading.

## Description

Event. Broadcast to all registered listeners when image loading progress changes.

ThumbImager2.[load](#)

## Availability

Flash Player 8

## Usage

```
listenerObject = new Object();

listenerObject.load = function( eventObject ){
    // insert your code here
    trace( "loaded");
}
myThumbImager2.addEventListener("load", listenerObject )
```

## Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

mc - Movieclip. Reference to the movieclip that contains the loaded image.

## Description

Event. Broadcast to all registered listeners when an image is loaded.

## ThumbImager2.change

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();  
listenerObject.change = function( eventObject ){  
    // insert your code here  
    trace( "changed" );  
}  
myThumbImager2.addEventListener("change", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

### Description

Event. Broadcast to all registered listeners when image size or position changes.

## ThumbImager2.transition

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();  
listenerObject.transition = function( eventObject ){  
    // insert your code here  
    trace( "transition event" );  
}  
myThumbImager2.addEventListener("transition", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

start - Boolean.

end - Boolean.

### Description

Event. Broadcast to all registered listeners when a transition is starts or finishes.

ThumbImager2.[cover](#)

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();  
listenerObject.cover = function( eventObject ){  
    // insert your code here  
    trace( "cover event" );  
}  
myThumbImager2.addEventListener("cover", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

start - Boolean.

end - Boolean.

### Description

Event. Broadcast to all registered listeners when a cover is displayed and closed.

## Thumbnailer2 Properties

### Thumbnailer2.data

#### Availability

Flash Player 8

#### Usage

myThumbnailer2.data:Array

#### Description

Object. Returns the data collection (array of objects) of currently loaded thumbnails.

#### Example

The following code will output the data of the currently loaded thumbnails:

```
for ( var i = 0; i < myThumbnailer2.data.length; i++ ) {  
    for ( var j in myThumbnailer2.data[ i ] ) {  
        trace( "Field '" + j + "' value: " +  
myThumbnailer2.data[ i ][ j ] );  
    }  
}
```

### Thumbnailer2.galleries

#### Availability

Flash Player 8

#### Usage

myThumbnailer2.galleries:Array

#### Description

Object. Returns the data collection (array of objects) of galleries.

#### Example

The following code will output the data of the current galleries:

```
for ( var i = 0; i < myTN2GalleryThumbnailer2.galleries.length; i++ )  
{  
    for ( var j in myThumbnailer2.galleries[ i ] ) {  
        trace( "Field '" + j + "' value: " +  
myThumbnailer2.galleries[ i ][ j ] );  
    }  
}
```

## Thumbnailer2.tags

### Availability

Flash Player 8

### Usage

myThumbnailer2.tags:Array

### Description

Object. Returns the data collection (array of objects) of tags.

### Example

The following code will output the data of the current tags:

```
for ( var i = 0; i < myThumbnailer2.tags.length; i++ ) {  
    for ( var j in myThumbnailer2.tags[ i ] ) {  
        trace( "Field '" + j + "' value: " +  
myThumbnailer2.tags[ i ][ j ] );  
    }  
}
```

## Thumbnailer2.enabled

### Availability

Flash Player 8

### Usage

myThumbnailer2.enabled:Boolean

### Description

Property. indicates whether the component is enabled ( true ) or not ( false ).

### Example

The following example sets the enabled property of the component to false :

```
myThumbnailer2.enabled = false;
```

## Thumbnailer2 Methods

### Thumbnailer2.[loadGallery](#)

#### Availability

Flash Player 8

#### Usage

```
myThumbnailer2.loadGallery( id:Number )
```

#### Parameters

id - Number. id of the fCMSPro gallery document.

#### Returns

Nothing.

#### Description

Method. Loads TN2Admin defined gallery.

#### Example

The following code will load gallery with id 24:

```
myThumbnailer2.loadGallery(24);
```

### Thumbnailer2.[loadTag](#)

#### Availability

Flash Player 8

#### Usage

```
myThumbnailer2.loadTag( id:Number )
```

#### Parameters

id - Number. id of the fCMSPro tag.

#### Returns

Nothing.

#### Description

Method. Loads images tagged with TN2Admin defined tag.

#### Example

The following code will load images tagged with tag id 27:

```
myThumbnailer2.loadTag(27);
```

## Thumbnailer2.loadXML

### Availability

Flash Player 8

### Usage

```
myThumbnailer2.loadXML( url:String, old:Boolean )
```

### Parameters

url - String. Path of the TN2 xml file.

old - Boolean. If 'true', ThumbnailerPro compatible xml is expected.

### Returns

Nothing.

### Description

Method. Loads TN2 xml definition.

### Example

The following code will load an xml file that defines the TN2 Gallery:

```
myThumbnailer2.loadXML( "myTN2.xml" );
```

## Thumbnailer2.setSize

### Availability

Flash Player 8

### Usage

```
myThumbnailer2.setSize( width:Number, height:Number )
```

### Parameters

width - Number. New width of the component.

height - Number. New height of the component.

### Returns

Nothing.

### Description

Method; Resizes the component.

### Example

The following code will resize component:

```
myThumbnailer2.setSize( 512, 50 );
```

## Thumbnailer2.nudge

### Availability

Flash Player 8

### Usage

```
myThumbnailer2.nudge( value )
```

### Parameters

value - Number (pixels) or String ("prev", "next")

### Returns

Nothing.

### Description

Method; Moves the thumbnails.

### Example

The following code will move thumbnails to the left:

```
myThumbnailer2.nudge( "next" );
```

## Thumbnailer2.setNudgeProperties

### Availability

Flash Player 8

### Usage

```
myThumbnailer2.setNudgeProperties(tweenEquation:Function, tweenDuration:Number)
```

### Parameters

tweenEquation - Function. Tween equation function

tweenDuration - Number. Duration of the tween in milliseconds

### Returns

Nothing.

### Description

Method; Sets tween properties for the nudge method.

### Example

The following code will set the nudge tween function to Strong.easeOut and the tween duration to 300 milliseconds:

```
myThumbnailer2.setNudgeProperties(mx.transitions.easing.Strong.easeOut  
, 300);
```

## Thumbnailer2.triggerThumb

### Availability

Flash Player 8

### Usage

```
myThumbnailer2.triggerThumb(index:Number)
```

### Parameters

index - index number of the thumbnail to trigger.

### Returns

Nothing.

### Description

Method; triggers the thumbnail. Same actions will be performed as if a user has clicked on the thumbnail.

### Example

The following code will trigger the third thumbnail:

```
myThumbnailer2.triggerThumb(2);
```

## Thumbnailer2 Events

Thumbnailer2.[over](#)

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.over = function( eventObject ){
    // insert your code here
    trace(eventObject.id);
}
myThumbnailer2.addEventListener("over", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

data - Object. Reference to the object containing thumbnail data with properties:

id - Number. ID of the TN2Admin image document. (undefined if data is loaded from XML file)

title - String. Title of the image.

description - String. Description of the image.

image\_src - String. URL of the image.

thumb\_src - String. URL of the thumbnail image.

Other properties are possible, depending on definition (preview\_src, extra1, extra2, width, height...)

### Description

Event. Broadcast to all registered listeners when mouse is over the thumbnail.

Thumbnailer2.[out](#)

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.out = function( eventObject ){
    // insert your code here
    trace(eventObject.id);
}
myThumbnailer2.addEventListener("out", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

data - Object. Reference to the object containing thumbnail data with the following properties:

id - Number. ID of the TN2Admin image document. (undefined if data is loaded from an XML file)

title - String. Title of the image.

description - String. Description of the image.  
image\_src - String. URL of the image.  
thumb\_src - String. URL of the thumbnail image.

Other properties are possible, depending on definition (preview\_src, extra1, extra2, width, height...)

## Description

Event. Broadcast to all registered listeners when mouse is out of the thumbnails area.

## Thumbnailer2.press

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();  
listenerObject.press = function( eventObject ){  
    // insert your code here  
    trace(eventObject.id);  
}  
myThumbnailer2.addEventListener("press", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

data - Object. Reference to the object containing thumbnail data with the following properties:

id - Number. ID of the TN2Admin image document. (undefined if data is loaded from an XML file)

title - String. Title of the image.

description - String. Description of the image.

image\_src - String. URL of the image.

thumb\_src - String. URL of the thumbnail image.

Other properties are possible, depending on the definition (preview\_src, extra1, extra2, width, height...)

## Description

Event. Broadcast to all registered listeners when the mouse is pressed over the thumbnail.

## Thumbnailer2.release

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.release = function( eventObject ){
    // insert your code here
    trace(eventObject.id);
}
myThumbnailer2.addEventListener("release", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

data - Object. Reference to the object containing thumbnail data with the following properties:

id - Number. ID of the TN2Admin image document. (undefined if data is loaded from XML file)

title - String. Title of the image.

description - String. Description of the image.

image\_src - String. URL of the image.

thumb\_src - String. URL of the thumbnail image.

Other properties are possible, depending on the definition (preview\_src, extra1, extra2, width, height...)

### Description

Event. Broadcast to all registered listeners when the mouse is released over the thumbnail.

## Thumbnailer2.releaseOutside

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.releaseOutside = function( eventObject ){
    // insert your code here
    trace(eventObject.id);
}
myThumbnailer2.addEventListener("releaseOutside", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

data - Object. Reference to the object containing thumbnail data with properties:

id - Number. ID of the TN2Admin image document. (undefined if data is loaded from the XML file)

title - String. Title of the image.

description - String. Description of the image.

image\_src - String. URL of the image.

thumb\_src - String. URL of the thumbnail image.

Other properties are possible, depending on definition (preview\_src, extra1, extra2, width, height...)

### Description

Event. Broadcast to all registered listeners when the mouse is released outside of the thumbnails' area.

## Thumbnailer2.infoLoad

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.infoLoad = function( eventObject ){
    // insert your code here
    trace("loaded");
}
myThumbnailer2.addEventListener("infoLoad", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

galleries - Object. Reference to the object containing gallery data.

tags - Object. Reference to the object containing tag data.

### Description

Event. Broadcast to all registered listeners when gallery and tag data are loaded.

## Thumbnailer2.load

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.load = function( eventObject ){
    // insert your code here
    trace(eventObject.id);
}
myThumbnailer2.addEventListener("load", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

data - Object. Reference to the object containing thumbnail data with the following properties:

id - Number. ID of the TN2Admin image document. (undefined if data is loaded from XML file)

title - String. Title of the image.

description - String. Description of the image.

image\_src - String. URL of the image.

thumb\_src - String. URL of the thumbnail image.

Other properties are possible, depending on definition (preview\_src, extra1, extra2, width, height...)

## Description

Event. Broadcast to all registered listeners when each thumbnail is loaded.

Thumbnailer2.[thumbsLoaded](#)

## Availability

Flash Player 8

## Usage

```
listenerObject = new Object();
listenerObject.thumbsLoaded = function( eventObject ){
    // insert your code here
    trace("loaded");
}
myThumbnailer2.addEventListener("thumbsLoaded", listenerObject )
```

## Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

## Description

Event. Broadcast to all registered listeners when all thumbnails are loaded.

Thumbnailer2.[resize](#)

## Availability

Flash Player 8

## Usage

```
listenerObject = new Object();
listenerObject.resize = function( eventObject ){
    // insert your code here
    trace("size changed");
}
myThumbnailer2.addEventListener("resize", listenerObject )
```

## Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

width - Number. New width of the component

height - Number. New height of the component

## Description

Event. Broadcast to all registered listeners when component is resized.

Thumbnailer2.[time](#)

## Availability

Flash Player 8

## Usage

```
listenerObject = new Object();
```

```
listenerObject.time = function( eventObject ){
    // insert your code here
    trace("changed");
}
myThumbnailer2.addEventListener("time", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

start - Boolean.

end - Boolean.

value - Number.

total - Number.

### Description

Event. Broadcasts the time event to all registered listeners.

Thumbnailer2.[change](#)

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.change = function( eventObject ){
    // insert your code here
    trace("changed");
}
myThumbnailer2.addEventListener("change", listenerObject )
```

### Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

newMovement - String. Old movement type.

oldMovement - String. New movement type.

### Description

Event. Broadcast to all registered listeners when movement type is changed.

## TN2Controls Properties

TN2Controls.[enabled](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.enabled:Boolean

### Description

Property; indicates whether the component is enabled ( true ) or not ( false ).

### Example

The following example sets the enabled property of the component to false.

```
myTN2Controls.enabled = false;
```

TN2Controls.[prevImage](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.prevImage:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'prevImage' movie clip.

```
trace(myTN2Controls.prevImage);
```

TN2Controls.[nextImage](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.nextImage:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'nextImage' movie clip.

```
trace(myTN2Controls.nextImage);
```

TN2Controls.[prevPage](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.prevPage:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'prevPage' movie clip.

```
trace(myTN2Controls.prevPage);
```

TN2Controls.[nextPage](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.nextPage:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'nextPage' movie clip.

```
trace(myTN2Controls.nextPage);
```

## TN2Controls.[gallery](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.gallery:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'gallery' movie clip.

```
trace(myTN2Controls.gallery);
```

## TN2Controls.[tags](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.tags:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'tags' movie clip.

```
trace(myTN2Controls.tags);
```

## TN2Controls.[download](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.download:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'download' movie clip.

```
trace(myTN2Controls.download);
```

## TN2Controls.[slideshow](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.slideshow:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'slideshow' movie clip.

```
trace(myTN2Controls.slideshow);
```

## TN2Controls.[fullscreen](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.fullscreen:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'fullscreen' movie clip.

```
trace(myTN2Controls.fullscreen);
```

## TN2Controls.[close](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.close:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'close' movie clip.

```
trace(myTN2Controls.close);
```

## TN2Controls.info

### Availability

Flash Player 8

### Usage

myTN2Controls.info:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'info' movie clip.

```
trace(myTN2Controls.info);
```

## TN2Controls.timer

### Availability

Flash Player 8

### Usage

myTN2Controls.timer:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'timer' movie clip.

```
trace(myTN2Controls.timer);
```

## TN2Controls.preloader

### Availability

Flash Player 8

### Usage

myTN2Controls.preloader:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'preloader' movie clip.

```
trace(myTN2Controls.preloader);
```

## TN2Controls.[imgInfo](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.imgInfo:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'imgInfo' movie clip.

```
trace(myTN2Controls.imgInfo);
```

## TN2Controls.[grid](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.grid:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'grid' movie clip.

```
trace(myTN2Controls.grid);
```

## TN2Controls.[tagsField](#)

### Availability

Flash Player 8

### Usage

myTN2Controls.tagsField:MovieClip

### Description

Property; reference to the control movie clip.

### Example

The following example traces a reference to the 'tagsField' movie clip.

```
trace(myTN2Controls.tagsField);
```

## TN2Controls Methods

### TN2Controls.[positionAll](#)

#### Availability

Flash Player 8

#### Usage

```
myTN2Controls.positionAll()
```

Parameters

None.

Returns

Nothing.

#### Description

Method; Positions all controls according to definitions.

#### Example

The following code will position all active controls:

```
myTN2Controls.positionAll();
```

### TN2Controls.[definePosition](#)

#### Availability

Flash Player 8

#### Usage

```
myTN2Controls.definePosition( controlName:String, posDef:Object )
```

Parameters

controlName - String. TN2 control identifier. Possible values: prevImage, nextImage, prevPage, nextPage, gallery, tags, download, slideshow, fullscreen, close, info, timer, preloader, imgInfo, grid and tagsField

posDef - Object. Defines control position. Properties:

ref - MovieClip. Position reference movieclip.

horizontal - String. Possible values: "left", "center" and "right".

vertical - String. Possible values: "top", "middle" and "bottom".

x - Number. Difference from horizontal value. Optional.

y - Number. Difference from vertical value. Optional.

rotation - Number. Optional.

callback - Function. Executes when position is set. Optional.

#### Returns

Nothing.

## Description

Method; Defines control position.

## Example

The following code will define position for "prevImage" button:

```
myTN2Controls.definePosition( "prevImage", {
    ref:myThumbImager2,
    horizontal:"left",
    vertical:"middle",
    x: 4,
    callback: function ( justLoaded:Boolean ){
        if ( justLoaded ) {
            _alpha = 100;
            gs.TweenLite.from( this, 0.4, {delay:0.3, _x:_x + 5,
_alpha:0, ease:Strong.easeOut});
        }
    }
});
```

TN2Controls.[setPosition](#)

## Availability

Flash Player 8

## Usage

```
myTN2Controls.definePosition( controlName:String, useCallback:Boolean )
```

Parameters

controlName - String. The TN2 control identifier. Possible values: prevImage, nextImage, prevPage, nextPage, gallery, tags, download, slideshow, fullscreen, close, info, timer, preloader, imgInfo, grid and tagsField  
useCallback - Boolean.

## Returns

Nothing.

## Description

Method; Positions control.

## Example

The following code will set the position of the "prevImage" button:

```
myTN2Controls.setPosition( "prevImage" );
```

## TN2Controls Events

TN2Controls.[controlClick](#)

### Availability

Flash Player 8

### Usage

```
listenerObject = new Object();
listenerObject.controlClick = function( eventObject ){
    // insert your code here
    if ( eventObject.id == "info" ) trace("show info");
}
myTN2Controls.addEventListener("controlClick", listenerObject )
```

Event Object Parameters

target - Movieclip. Reference to the movieclip that triggers the event.

id - String. Control name. Possible values: prevImage, nextImage, prevPage, nextPage, gallery, tags, download, slideshow, fullscreen, close, info, timer, preloader, imgInfo, grid and tagsField

### Description

Event. Broadcast to all registered listeners when mouse is released over a control movieclip.

# 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: [TN2Gallery 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.*