FSI VIEWER

Let your website visitors have a closer look

User Manual



Version 2018

FSI Viewer

FSI TouchZoom

FSI QuickZoom

FSI ThumbBar

FSI Showcase

FSI Pages

Last update: March 2018

Latest Build: 18.03.15

Developed by:
NeptuneLabs GmbH
Lagesche Str. 32
D-32657 Lemgo
Germany

© 2009-2018 NeptuneLabs. All rights reserved.

All brands and product names are trademarks or registered trademarks of the respective producers. FSI Viewer, FSI Server and NeptuneLabs are registered trademarks of NeptuneLabs GmbH, Germany.

1 Introduction 8

2 FSI Viewer 9

2.1 Usage 9

2.2 How it works 11

2.3 Using FSI Viewer 12

2.3.1 The Menu Bar 13

Main Functions 13

2.3.2 Mouse Modes 14

2.3.3 Additional Buttons 15

2.3.4 Using the navigator window 15

2.4 Index structure of FSI Viewer 16

2.5 Configuration 17

2.5.1 Enabling the Debug Mode 17

2.5.2 Using XML Configuration Files (*.xml) 18

2.5.3 Hierarchy of Configuration Parameters 21

Configuration Hierarchy 22

Choosing the adequate Configuration Method 22

2.5.4 Conditional Configuration Options 23

Cascaded Conditions 24

Conditions and Groups 25

Building Conditions 26

2.5.5 Retrieving Parameters from FSI Server 29

2.5.6 Using Custom Language Files 30

2.6 FSI Viewer Parameters 31
2.6.1 Basic Parameters 35
General 35
Display Options 48
Preview Options 55
Global Parameters for Hot Spots 57
Camera Section 61
VirtualSpace Section 69
HotSpot Section 72
Video Section 85
2.6.2 Plugin Parameters 90
FullScreen Plug-In 91
AutoSpin Plug-In 91
2.7 Creating and using Spin Videos 95
2.8 Embedding the viewer at runtime 96
2.9 JavaScript Interface 98
2.9.1 Public Methods 98
2.9.2 Callbacks 111
Using Callbacks 111
3 FSI TouchZoom 120
3.1 Usage 120
3.2 Initializing FSI TouchZoom 122
3.3 FSI TouchZoom Parameters 123

3.4 FSI TouchZoom JavaScript Interface 128
3.4.1 Public Methods 129
3.4.2 Callbacks 134
4 FSI QuickZoom 136
4.1 Usage 136
4.2 How FSI QuickZoom works 138
4.3 FSI QuickZoom Parameters 138
4.4 Initializing FSI QuickZoom manually 147
4.5 FSI QuickZoom JavaScript Interface 149
4.5.1 Public Methods 150
4.5.2 Callbacks 153
5 FSI ThumbBar 155
5.1 Usage 155
5.2 How FSI ThumbBar works 158
5.3 FSI ThumbBar Parameters 159
5.4 Customization 176
5.4.1 CSS Classes 176
5.4.2 Adding Control Buttons 177
5.4.3 Defining a custom presentation type 178
5.5 ThumbBar JavaScript Interface 180
5.5.1 Public Methods 180
5.5.2 Callback functions 186
6 FSI Showcase 190

6.1 Usage 190
7 FSI Pages 193
7.1 Introduction 193
7.2 How FSI Pages works 193
7.3 Usage of FSI Pages 194
7.3.1 Defining Image Collections 197
7.3.2 Aspect Ratio of the Pages 199
7.3.3 Presets in FSI Server Interface 200
7.3.4 Printing Pages 200
7.3.5 Saving Pages 201
Saving a complete PDF 202
7.3.6 Special URL Values 203
7.4 FSI Pages Parameters 204
7.4.1 General Parameters 207
Search 217
7.4.2 Layout/Appearance Parameters 222
7.4.3 Link Parameters 236
7.4.4 Plug-in Parameters 242
FullScreen 242
PagesThumbBar 243
Bookmarks 244
Chapters 246
7.5 FSI Pages JavaScript Interface 247

7.5.1 FSI Pages Public Methods 248
7.5.2 FSI Pages Callbacks 258
8 Appendix 267
8.1 Escaping or URL-encoding parameter values 267
8.2 HTML Tags available in FSI Viewer 267
8.3 Example of a custom skin CSS file 268
9 Index 274

1 Introduction

The JavaScript based viewers FSI Viewer, FSI TouchZoom, FSI QuickZoom, FSI Showcase, FSI Pages and FSI ThumbBar bring outstanding zoom and viewer qualities to almost all devices. They are highly customizable and work independently from FSI Viewer Flash.

FSI Server is the basis of all viewers which request the image data from it. By using FSI Server, only one high resolution source image is required for each image to be displayed.

Thank you for using NeptuneLabs software!

www.neptunelabs.com **: Online Resources for FSI Viewer**
Visit www.neptunelabs.com for software updates, regularly updated samples, tutorials and downloads.

2 FSI Viewer

FSI Viewer is a JavaScript based tool to display 2D image zoom and 360° Object spins on desktop computers and mobile devices. Depending on the device capabilities it offers in place zoom or pinch zoom support. FSI Viewer also supports optional multiple axis spin.

Using an optional video, the rotation of the 360° object can be displayed extremely smooth on devices that support video capturing.

Optional hot spots with HTML content can be added by providing 3D coordinates (x,y,z) in millimeters.

Imaging Servers

FSI products covered by this manual can at present securely be run with FSI Server from version 2016.

Supported devices

- iOS 4 or higher
- Android 4.x based devices
- Windows Phone 8 and Windows RT devices
- Windows 8 with touch enabled browsers
- Mouse zoom: desktop or laptop computers & all major browsers

2.1 Usage

I) Add the fsiviewer.js script to the head of your document:

Adding the script:

```
<head>
[...]
//adding the fsiviewer.js script
```

```
<script src="viewer/applications/viewer/js/fsiviewer.js" type="text/
javascript"></script>
</head>
```

II) Add a <fsi-viewer> tag to the part of your document body where the zoom and/or the 360° product spin should be displayed. Add the desired dimension and the imagesource directory to it:

Integrate FSI Viewer JS (2D Zoom):

```
<body>
[…]
<fsi-viewer width="500" height="600" src="/images/foo.tiff">
// here you can place the fallback content in case the browser
// cannot display the viewer, e,g,:
<!-- Fallback Content -->
<b>Sorry</b>, your browser is not capable of running FSI Viewer JS.
</fsi-viewer>
</body>
```

Integrate FSI Viewer JS 360 (360° Spins):

```
<body>
[…]
<fsi-viewer width="500" height="600" dir="/images/spin/">
// here you can place the fallback content in case the browser
// cannot display the viewer, e,g,:
<!-- Fallback Content -->
<b>Sorry</b>, your browser is not capable of running FSI Viewer JS.
</fsi-viewer>
</body>
```

Parameters are added to the <fsi-viewer> tag. Note that you can as well use FSI Viewer configuration files (required e.g. when using hot spots):

Adding Parameters to the viewer:

```
<head>
//adding the fsiviewer.js script
<script src="js/fsiviewer.js" type="text/javascript"></script>
</head>

<body>

//Adding parameters directly to the tag:
<fsi-viewer width="500" height="600" src="/images/sample.jpg"
debug="true" skin="silver">
</fsi-viewer>


//Example with using a FSI Viewer configuration file:
<fsi-viewer width="500" height="600" src="/images/sample.jpg"
cfg="sample/sample_configuration">
</fsi-viewer>

</body>
```

2.2 How it works

The script uses the given <fsi-viewer> tag, which is similar to an <img> tag, to display a single image zoom and/or 360° object spin.

If a video has been defined, the script captures video frames on start to provide an ultra smooth rotation.

Without a video (or on devices not supporting video capturing) a fixed number of images from different viewing angles will be used to display the object spin.

The user can drag the image left and right to spin and zoom in/out either using pinch zoom or by clicking on the object.

Note on Internet Explorer:

In order to make FSI Viewer JS work, Internet Explorer must run in IE8 quirks, IE9 or higher quirks or standards mode. You can ensure that by adding an X-UA-Compatible meta tag to the head section of your web page, for example:

```
<meta http-equiv="X-UA-Compatible" content="IE=Edge">
```

The HTML code generated in the FSI Server interface contains all necessary tags to ensure maximum compatibility on all devices.

2.3 Using FSI Viewer

FSI Viewer is navigated by the menu bar (here seen below the image) and by using the mouse directly on the image. For example, you can select a section of the image you want to magnify.

The mouse functions are determined by the corresponding buttons on the menu bar (zoom, pan, rotate, etc.).

The optional small navigator window (bottom right) displays the position of the image section currently viewed.



2.3.1 The Menu Bar



> ⚠ The menu bar might look different depending on skins or additional plugins.

Main Functions

	**Back to Initial View**. Undoes zoom, pan and rotation and restores the initial view. (identical with pressing space bar)
	Zoom In
	Zoom Out

2.3.2 Mouse Modes

The MouseModes plug-in is required to display the following mouse mode buttons.

	**Mouse - Zoom In** Choose this function to enlarge segments using the mouse. Click on the image and drag the frame over the desired segment. Alternatively you can click on the image, without marking a segment. The image will then be magnified in steps. To zoom out in steps, hold down the CTRL-key and click on the image.
	**Mouse - Pan** Choose this function to pan the image using the mouse. Click on the image and drag in the desired direction. To return to the original view, hold down the CTRL-key and click on the image.
	**Mouse - Turn** Choose this function to rotate the image around the z-axis. Click on the image and drag in whatever direction you want to turn the object (to the right or the left). To reset the rotation, hold down the CTRL-key and click on the image.

	**Mouse - Rotate (only for 360° presentations)** Choose this function to rotate the object around the y-axis or the x-axis if available. Click on the image and drag to the left or to the right. Move the mouse up or down to rotate the object around the x-axis. To reset rotation, hold down the CTRL-key and click on the image.

2.3.3 Additional Buttons

	**Hide / Display menu** Displays or hides the user interface.
	**Information** Displays information about the viewer (optional, can be hidden by altering the skin CSS)

2.3.4 Using the navigator window

In the navigator window you can see a miniature presentation of the entire image. The segment which is currently viewed is framed. In the navigator window you can change the current segment either by dragging the frame or by clicking on the desired area of the image.



2.4 Index structure of FSI Viewer

The index structure of the JS viewers as found in WEBAPPS/fsi/viewer/:

Directory	Content Description
applications	contains the .js scripts for the according viewers
config	contains the configuration files
languages	User interface language files
skins	FSI Skins containing the visible parts of the user interface

Please note:
You need to make sure that your web server properly serves files with the following extensions: *.js, *.xml

2.5 Configuration

2.5.1 Enabling the Debug Mode

The debug mode can be enabled or disabled by passing the "Debug" parameter either in the code snip or in an XML configuration file.

Please use "debug=1" or "debug=true" to enable the debug mode and "debug=0" (or omit the debug parameter) to disable the debug mode. To disable the debug mode permanently you can safely delete the "/debug" directory of your FSI setup directory.

Example:

```
<body>
<fsi-viewer width="500" height="600" src="/images/foo.tiff"
debug="true">
</fsi-viewer>
</body>
```

Example:

If FSI Viewer does not work or look as expected in an HTML page you can quickly debug FSI Viewer as follows:

- Open the HTML page containg FSI Viewer
- Open the "Inspect Element" feature of your browser

- In the "Console" and "Network" part you can now check if everything is implemented and loaded correctly

2.5.2 Using XML Configuration Files (*.xml)

FSI Viewer uses XML configuration files which can easily be edited with any text editor.

The default configuration file "_default.xml" - which is located in WEBAPPS/fsi/viewer/ is the most important XML configuration file and will always be evaluated first.

To use FSI Viewer with additional configuration files you have to place the file in the WEBAPPS/fsi/viewer/config/ folder and provide the name of the configuration file using the "cfg" parameter in the code:

Using FSI Viewer Configuration Files

```
<head>
//adding the fsiviewer.js script
<script src="js/fsiviewer.js" type="text/javascript"></script>
</head>
<body>
//Example with using a FSI Viewer configuration file:
<fsi-viewer width="500" height="600"
src="/images/sample.jpg" cfg="sample_configuration">
</fsi-viewer>
</body>
```

This will load the WEBAPPS/fsi/viewer/config/sample_configuration.xml file as the FSI Viewer JS configuration.

Configuration files are structured into configuration groups (XML nodes) containing parameter names and values. Only parameters in these groups will be recognized by FSI Viewer. Configuration files do not need to contain all groups though. Each parameter has to be provided as an individual XML child node of a

group where the node name is the parameter name and the parameter value needs to be provided as "value" attribute of the node.

> **Note:**
>
> It is possible to implement viewer parameters in the XML configuration using specific viewer nodes, e.g.
>
> - <Viewer>
> - <TouchZoom>
> - <QuickZoom>
> - <Pages>
> - <Thumbbar>
>
> This enables you for example to implement globally used FSI Viewer, FSI TouchZoom, FSI Pages, FSI ThumbBar or FSI QuickZoom parameters in the _default.xml or for certain viewers with an individual configuration .xml file.

Structure of XML configuration files

```
<fsi_parameter>
          <image>
             <parameter value="" />
          </image>

        <Viewer>
                 <parameter value="" />
        </Viewer>

          <plugins>
            <parameter value="" />
```

```
        </plugins>

        <options>
          <parameter value="" />
        </options>

</fsi_parameter>
```

Simple Configuration File

```
<fsi_parameter>
  <image>
     <path value="samples/Watch.jpg" />
  </image>
  <Viewer>
     <skin value="silver" />
     <debug value="1" />
         <noNav value="true" />
  </Viewer>
  <TouchZoom>
          <useDevicePixelRatio value="true" />
          <enableByCSSClass value="zoom-image" />
          <zoomPrecision value="3" />
  </TouchZoom>
</fsi_parameter>
```

Please keep in mind the following rules when using XML configuration files:

- Attribute values must be provided in double quotes
- The last occurrence of a given parameter will be used
- Configuration files might be cached by your browser, so that changes in these files require flushing the browser cache. This does not apply when using the debug mode.
- Make sure to save configuration files in UTF-8 format if your configuration file contains language specific diacritics, Japanese characters and alike

You can comment on or disable sections of an XML configuration file using the usual XML comment syntax:

Comments

```
<image>

  <!-- This is a comment -->
  <path value="image.tif" />

  <!-- The following section will be ignored -->
  <!--
  <path value="anotherimage.tif" />
  -->
</image>
```

2.5.3 Hierarchy of Configuration Parameters

As explained before parameters can be provided in two different ways:

1. The default configuration file "_default.xml" - located in WEBAPPS/fsi/viewer/ - will always be evaluated first
2. You can store image specific parameters in additional *.xml files in WEBAPPS/fsi/viewer/config/
3. You can override parameters from either configuration file by specifing them in the <fsi-viewer> tag

Parameters specified in multiple ways – for example in the default configuration and by custom configuration files– will be evaluated in the following way:

- Parameters defined in configuration files override parameters defined in the default configuration file "_default.xml".
- Parameters defined in the <fsi-viewer> tag override parameters defined in the custom configuration file.

The list below shows the hierarchy resulting from the sequential evaluation of the different configuration methods:

Configuration Hierarchy

1. Parameters set in the HTML tag
2. Configuration file (*.xml)
3. Default configuration file (_default.xml)

Choosing the adequate Configuration Method

So far you learned that FSI Viewer can be configured by any combination of:

- Parameters set in the HTML tag
- Additional configuration files (*.xml)
- The default configuration file (_default.xml)

These configuration options are not only alternate methods. Combining these methods offers the possibility to provide configuration parameters as easily as possible and as dynamically as required. The following considerations might be helpful when deciding which method to choose for a certain parameter.

Concerning the configuration via configuration files or the default configuration file, in earlier versions it was possible to define global parameters via the <Options> node (e.g. skin). This is still possible, but if a parameter is defined in the custom viewer node (e.g. skin is set to white in the <Viewer> node), this parameter is superior to the one defined in the <Options> node.

Default Configuration

You can easily change global parameters that apply to all FSI Viewer instances by editing a single file. This is therefore the recommended configuration method for:

General configuration options like:

- Skin
- Language
- MenuAlign, AnimationSpeed and alike

Configuration Files

Separate configuration files should be used for parameters specific to a group of images or a specific way to display images. You might for example want to setup individual configuration files for different types of FSI Viewer JSinstances, like "small_no_interface. xml", "large_with_hotspots.xml" as an example.

Usually you don‘t need to create a configuration file for individual images, as you can specify the image path or other unique parameters dynamically in the code.

2.5.4 Conditional Configuration Options

Conditional options are parameter blocks being evaluated if one or more conditions are met.

Example:

```
<fsi_parameter>

  <if parameter="renderer" value="png">
    <options>
      <effects value="Sepia()" />
      <renderer value="jpeg" />
    </options>
  </if>
```

The "options" block containing the "effects" and "renderer" parameters applies only, if "renderer" equals "png".

When are the conditions evaluated?

Conditional options are evaluated after the non-conditional parameters and after applying parameters by query and/or arguments.

When to use conditions in configurations?

Conditions are evaluated client side each time a viewer instance is loaded. Therefore do not use too complex conditions and use them if required only.

You could e.g. define a specific UI skin depending on the directory a source image resides in. The recommended way in this case though would be to assign the "skin" attribute to the viewer's HTML tag.

A good example **when** to use conditions is using the "@mobile" parameter in conditions. If you use FSI Viewer and would like to hide the UI on mobile devices, this condition does the trick:

Example:

```
<fsi_parameter>

  <if parameter="@mobile" value="true">
    <viewer>
      <HideUI value="true" />
    </viewer>
  </if>
```

Cascaded Conditions

If multiple conditions should be met to apply parameters, you can cascade conditions:

Example:

```
<fsi_parameter>

  <if parameter="renderer" value="sw1">
    <if parameter="quality" operator="lessThanEqual" value="50" >
      <options>
        <effects value="Sepia()" />
      </options>
    </if>
  </if>
```

The "options" block containing the "effects" and "renderer" parameters applies only, if

- "renderer" equals "sw1"

AND

- "quality" <= 50

Conditions and Groups

Conditions can appear within groups like <options>, <viewer>, <pages> and alike.

In this case they are only evaluated, if the group matches for the viewer in use.

That means a condition in <pages> will be ignored by FSI Viewer. This is the recommended way if a conditions applies to a single viewer type only.

Example:

```
<fsi_parameter>
```

```
<pages>
  <if parameter="renderer" value="png">
    <effects value="Sepia()" />
    <renderer value="jpeg" />
  </if>
</pages>
```

If a condition should apply to all viewer types you can add the condition to the **<options>** group.

In case you want a condition to apply to multiple viewer types, but NOT to all, you can add the groups in question to the condition. The content of the groups does not need to be the same. If it should be the same, you need to repeat it in each viewer group:

Example:

```
<fsi_parameter>

  <if parameter="renderer" value="sw1">
    <pages>
      <effects value="Sepia()" />
    </pages>

    <viewer>
      <effects value="Sepia()" />
    </viewer>
  </if>
```

Building Conditions

The tag name of conditions must be "if".

The following attributes are supported:

- **parameter**: Mandatory. The name of the configuration parameter to compare. E.g. "language", "imageSrc", "dir", etc.
- **value**: Mandatory. The value to compare the configuration parameter to.
- **operator**: Optional. Defaults to "=". The operator to use for the comparison.

Supported Operators in Conditions:

=	equals (case insensitive)	(aliases: eq)
==	equals (case sensitive)	(aliases: none)
!=	not equal (case insensitive)	(aliases: not)
!==	not equal (case sensitive)	(aliases: none)
<	less than (numerical)	(aliases: lt, lessThan)
<=	less than or equal (numerical)	(aliases: lte, lessThanEqual)
>	greater than (numerical)	(aliases: gt, greaterThan)
>=	greater than or equal (numerical)	(aliases: gte, greaterThanEqual)
match	match a JS regular expression	(aliases: none)
notmatch	not match a JS regular expression	(aliases: none)

Please note:

As "<" and ">" may not be used in XML attribute values, you should use the alias instead (e.g. "lt" instead of "<". You can as well use the entity "<", but this doesn't look to good.

Regular Expressions (operator "match" and "notmatch")

Please assign a JavaScript regular expression like you would in a browser to the "value" attribute of the condition. The following example matches all source images ending with ".tif":

Example:

```
<if parameter="imagesrc" operator="match" value="/.*\.tif$/gi" >
    <options>
      <effects value="Sepia()" />
    </options>
  </if>
```

What parameter values can I use for conditions?

You can use all configuration parameters as you would use them in an URL.
E.e. use "imagesrc" for <image><src value="foo" />

In addition you can use a small number of device dependent parameters prefixed with "@".
The following table lists possible values:

Name	Possible values	Description
@mobile	true/false	true for mobile devices, false for desktop devices
@devicepixelratio	>= 1.0	device pixel ration of the user's device

@navigatorUserAgent	String	User agent value of the user's browser
@navigatorLanguage	String	navigator.language of the user's browser
@navigatorPlatform	String	navigator.platform of the user's browser

2.5.5 Retrieving Parameters from FSI Server

FSI Viewer is capable of retrieving image specific parameters from FSI Server dynamically. This way you do not need to define these parameters explicitly. The list below shows some typical parameters that can be retrieved from FSI Server automatically:

- ImageWidth
- ImageHeight
- ImageTilesX
- ImageTilesY
- SceneSets
- Image Collections
- Hyperlinks
- Hotspots

How to access image parameters from FSI Server?

FSI Server can be queried for information on a specific folder or image by simple HTTP Request, e.g.

http://[domain]/fsi/server?type=info&tpl=info.json&source=samples/sample_image.tif

and will return a JSON file containing the related information. Depending on the type= and tpl= parameters, different informations can be retrieved dynamically. You can read more about the server adressing in the Appendix of the FSI Server manual.

In which case will these values be retrieved automatically?

FSI Viewer tries to retrieve the parameters ImageWidth and ImageHeight from the server if one of these values has not been defined manually. If the server delivers additional parameters (e.g. ImageTilesX, ImageTilesY) these parameters will only have an effect if they have not been defined manually.

2.5.6 Using Custom Language Files

In case you want to edit tool tips for existing languages or add support for additional languages you can edit or create language files in the subdirectory "/languages" of your FSI Viewer setup directory.

Each language file contains a list of phrases used in FSI Viewer user interface.The language files are XML files and need to be saved in UTF-8 format. The attribute "id" specifies the context of each phrase. Please note that you need to use XML entities for characters not allowed in XML files, like e.g.

&	for "&"
<	for "<"
>	for ">"
"	for "
'	for ,

You can use basic HTML-tags to apply text formatting to the tool tips.

Excerpt from a language file:

```
<Data language="english" color="000000" font="_sans" offsetTop="0"
offsetLeft="0"
offsetWidth="0" offsetHeight="0">


        <tip id="ZoomOut">
                <b>Zoom Out</b>
                <br/>Click to decrease magnification
        </tip>

        <tip id="ZoomIn">
                <b>Zoom In</b>
                <br/> Click to increase magnification
        </tip>

        <tip id="Downloading Data">Downloading:</tip>

        [...]

</Data>
```

You can duplicate existing language files or create new files as needed.

To use custom language files, please use the "language" parameter and set the parameter value to the filename without the ".xml" extension.

It is also possible to map the available language files to the language of the browser (the file mapping.json in the \languages folder of your FSI Viewer installation can be adapted to your liking for that). In order to use the mapping defined in the JSON file, the parameter language needs to be set to "auto".

2.6 FSI Viewer Parameters

Parameters are added to the <fsi-viewer> tag. Note that you can as well use FSI Viewer configuration files (required e.g. when using hot spots):

Adding Parameters to the viewer

```
<head>
//adding the fsiviewer.js script
<script src="js/fsiviewer.js" type="text/javascript"></script>
</head>
<body>

//Adding parameters directly to the tag:
<fsi-viewer width="500" height="600" src="/images/sample.jpg"
debug="true" skin="silver">
</fsi-viewer>
//Example with using a FSI Viewer configuration file:
<fsi-viewer width="500" height="600" src="/images/sample.jpg"
cfg="sample/sample_configuration">
</fsi-viewer>

</body>
```

You can define individual configuration .xml files like this:

Simple Configuration File

```
<fsi_parameter>

  <image>
     <path value="samples/Watch.jpg" />
  </image>
```

```
  <Viewer>
     <skin value="silver" />
     <debug value="1" />
         <noNav value="true" />
  </Viewer>

</fsi_parameter>
```

If you would like to use global parameters for all FSI Viewer instances, the corresponding parameters can be defined in the _default.xml.

Example _default.xml:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <Viewer>
                <skin value="white" />
                <inPlaceZoom value="true" />
                <hideUI value="false"/>
                <adaptiveuisize value="true"/>
        </Viewer>
...

<Options>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

You can optionally modify the way FSI Viewer works by passing parameters to the script. To do so, pass an object containing the parameters and value as an object to the constructor:

Example:

```
var node = document.createElement("fsi-viewer");


node.style.width = "600px";
node.style.height = "400px";

var parameters =
{"imagesrc" : "images/Rocker Shoe.tif", "skin":"silver", "debug":true};
$FSI.setParameters(node, parameters);
document.body.appendChild(node);


// Important: call this AFTER adding the node to the DOM
$FSI.initCustomNode(node);
```

Note:

$FSI.initCustomNode(node) initializes the given node.

When adding multiple viewers you can initialize all new FSI Viewer nodes in one go, using this command:

```
$FSI.initCustomTag("fsi-viewer");
```

or, when using different custom tags (fsi-viewer, fsi-imageflow,…):

```
    $FSI.initCustomTags(); to initialize all new FSI
    custom tags.
```

2.6.1 Basic Parameters

General

debug	
Description	display debug and status information in the JavaScript console
Syntax	Boolean
Default	false
Context	FSI Viewer

Display debug and status information in the browser's JavaScript console.

headers	
Description	"&headers=" parameter to use for image requests
Syntax	String
Default	"cors"
Context	FSI Viewer

The "&headers=" parameter to use for image requests.

The default value "cors" ensures cors compatibility for cross-origin scenarios.

cfg	
Description	relative path to an XML configuration file in FSI Viewer format
Syntax	String
Default	""
Context	FSI Viewer

The relative path to an XML configuration file in FSI Viewer format.

When using a relative path the location specified by the FSI Viewer "FSIBase" parameter will be used, which is [viewer]/config/ by default.

navWidth	
Description	defines the width of the navigation frame
Syntax	String
Default	"120"
Context	FSI Viewer

Defines the width of the navigation frame displayed in the right corner on the bottom.

navHeight	
Description	defines the height of the navigation frame
Syntax	String
Default	"120"
Context	FSI Viewer

Defines the height of the navigation frame displayed in the right corner on the bottom.

urlLoadingAnimation	
Description	defines the loading animation shown
Syntax	string
Default	"/skins/resources/loading.svg"
Context	FSI Viewer

Defines the loading animation shown. Custom animations can be placed here: / skins/resources/. Setting the parameter to "false" discards the display.

InvisibleLoadingPurposes	
Description	defines if the loading animation should be excluded under certain circumstances
Syntax	string
Default	-
Context	FSI Viewer

Defines if the loading animation should be excluded under certain circumstances. Per default it is activated whenever the viewer loads images. Setting the parameter to "asyncResize" omits the loading animation when asynchronic loading takes place, e.g. when the viewer is switched to FullScreen and the higher resolution images need to be loaded. "tile" omits the presence of the loading animation when tiles are loaded, e.g. when the user zooms into the picture. It is also possible to combine both as "tile,asyncResize" - in this case the loading animation is only shown in the initial loading sequence.

skin	
Description	defines the skin to display
Syntax	string

skin	
Default	"black"
Context	FSI Viewer

Defines the skin which will be displayed. FSI Viewer comes with three default skins, "black", "white" and "silver".

Skins are defined via CSS and can be found here: WEBAPPS/fsi/viewer/skins.

It is also possible to alter existing skins or to define custom skins in this location.

In order to use a custom skin, the easiest way would be to duplicate one of the default skin CSS files and adapt them to your liking. Make sure to use a custom class in front af ALL classes defined in the custom CSS, otherwise you will break other skins on the same HTML document.

Too see an example file, please go to the Appendix.

General appearance CSS rules, which are valid on an overall level, (e.g. the logo container, height of the menubar, etc.) are defined in the corresponding core CSS files, which can be found in WEBAPPS/fsi/viewer/skins/resources. (fsi-core.css, fsi-viewer-core.css)

IMPORTANT:

Never change the classes of the core CSS in the core files directly. If you would like to adapt the CSS rules listed there, please overwrite them in a custom skin file or in the HTML document instead.

You can find detailed tutorials for changing the appearance of the skin in the support section of our website:

- Create Basic Skins for FSI Viewer/ FSI Pages JS
- Create Advanced Skins for FSI Viewer/ FSI Pages JS
- Create Custom Buttons for Skins
- Create a Custom Skin using JavaScript

For a detailed example of a basic custom skin, please go to "Example of a custom skin CSS file" .

dir	
Description	path to the source images for the object spin on FSI Server
Syntax	String
Default	""
Context	FSI Viewer

The path to the source images for the object spin on FSI Server.

The viewer will load the list of images from the given directory.

src	
Description	path to the single images used for FSI Viewer
Syntax	String
Default	""
Context	FSI Viewer

Defines the path to single images used for FSI Viewer.

imagesources	
Description	paths to single images used for FSI Viewer
Syntax	String

imagesources	
Default	""
Context	FSI Viewer

Defines the paths to single images used for FSI Viewer. Example: "dir1/1.jpeg, dir1/2.jpeg, dir2/1.jpeg,..."

language	
Description:	Defines the language of the tooltips
Syntax:	String
Default:	—
Context:	FSI Viewer

The language parameter can be used to adapt the tooltips which are shown while hovering over a button to a certain language. The available languages can be found in the \languages folder of your FSI Viewer installation. (See Using Custom Language Files for more info).

It is also possible to map the available language files to the language of the browser (the file mapping.json in the \languages folder of your FSI Viewer installation can be adapted to your liking for that). In order to use the mapping defined in the JSON file, the parameter language needs to be set to "auto".

effects	
Description:	Image manipulation parameters for all images
Syntax:	String
Default:	—
Context:	FSI Viewer

The effects parameter can be used to pass image manipulation parameters to the imaging server. Typical manipulation parameters include the image saturation

("saturation") and image sharpening ("sharpen"). If no effect is specified, the default settings of the server (profile) apply.

Please refer to your FSI Server documentation for a list of available effect parameters.

initialView	
Description	defines the initial image segment
Syntax	String
Default	""
Context	FSI Viewer

Image section (and rotation) to display on startup.

The first two parameters are required as they specify the scene set and the scene. For 2D images both values default to 1.

initialViewPersistent	
Description	Keep the InitialView as default view
Syntax	boolean
Default	false
Context	FSI Viewer

If this parameter is activated, clicking the "Reset" button will display the image section defined by the InitialView parameter instead of the entire image. If this parameter is set to "false", the initial view will be displayed on startup, but pressing the Reset button will reset the view to the entire image.

panLimits	
Description	deterrmines how far the user is allowed to pan

panLimits	
Syntax	String
Default	"strict"
Context	FSI Viewer

Determines how far the user is allowed to pan. Possible settings are: "loose", "medium", "strict"

preventImageTransparency	
Description	prevents the image transparency
Syntax	Boolean
Default	false
Context	FSI Viewer

Prevents using the image transparency if a transparency channel exists.

useDevicePixelRatio	
Description	Use the device's pixel ratio to display even sharper images.
Syntax	Boolean
Default	true
Context	FSI Viewer

Use the device's pixel ratio to automatically display even sharper images.

drawCube	
Description	draws a 3D cube wireframe representing the spin object space

drawCube	
Syntax	boolean
Default	false
Context	FSI Viewer

Draws a 3D cube wireframe representing the object space of the spin as defined by the "virtualSpace" parameters.

The drawCube option can be useful to validate the "virtualSpace" parameters.

> ⚠ This option only works on desktop computers using the HTML canvas element.

drawTable	
Description	draws a virtual rotating table plate
Syntax	boolean
Default	false
Context	FSI Viewer

Draws a virtual rotating table plate.

> ⚠ This option only works on desktop computers using the HTML canvas element.

tableDiameter	
Description	defines diameter of virtual rotating table plate

tableDiameter	
Syntax	Float
Default	950
Context	FSI Viewer

Defines the diameter of the virtual rotating table plate in mm.

> ⚠ This option only works on desktop computers using the HTML canvas element.

drawGroundPlate	
Description	draws a virtual ground plate
Syntax	boolean
Default	false
Context	FSI Viewer

Draws a virtual ground plate.

> ⚠ This option only works on desktop computers using the HTML canvas element.

initialMouseMode	
Description	Mouse mode on startup
Syntax	String

initialMouseMode	
Default	"auto"
Context	FSI Viewer

Specifies the selected mouse mode on startup.

Possible values:	
0:	(Zoom)
1:	(Pan)
2:	(Spin 360°)

menuButtonOrder	
Description	defines the button order of the menu bar buttons
Syntax	String
Default	-
Context	FSI Viewer

Defines the button order of the menu bar buttons. The buttons need to be listed in a string with the corrsponding button IDs.

The following button IDs currently exist:

Reset, ZoomOut, ZoomIn, MouseMode_0, MouseMode_1, MouseMode_2, MaxZoom, HotSpots, ToggleAutoSpin, ToggleFullScreen

autoDestroy	
Description	automatically destroy instances
Syntax	boolean
Default	true

autoDestroy	
Context	FSI Viewer

Automatically destroy instances created with the fsi-viewer tag upon removing the tag from the document DOM.

spinDirectionImage	
Description	defines direction of rotation
Syntax	String
Default	"left"
Context	FSI Viewer

Specifies the direction of the spin rotation.

SceneSets	
Description	Scene sequence definition
Syntax	String
Default	1 – n tiles
Context	FSI Viewer

Defines the minimum instance size of the viewer.

Using this parameter you can define scene sets for 360° views. Scene sets define the order of scenes to be displayed.

The default is a single scene set containing all scenes (image tiles) from scene 1 to the last scene image. The default sequence is left to right, top to bottom in a tile image and the sequence of the <image> nodes for 360° views based on individual source images.

In the viewer scenes can be selected by moving the mouse horizontally and scene sets can be selected by moving the mouse vertically.

Syntax:

Individual sets have to be separated with ";" (semicolon).

Scenes within the sets are separated with "," (comma) and specify the 1-based index of the image from left to right, top to bottom up to the total number of scenes

(TilesX x TilesY or individual source images).

Instead of individual scenes you can also specify scene ranges, e.g. "5-10".

If you leave out the start or the end of a scene range, 1 or respectively the last scene will be assumed in this case. If you specify a range using "->" the scenes from the first number to the last scene image plus the first scene image up to the second number will be assumed.

Example:

SceneSets="1-12;13-24;25-36;37-48"

reverseSceneSets	
Description	reverses the Scene Sets
Syntax	boolean
Default	false
Context	FSI Viewer

Reverses the order of the Scene Sets.

NoSetLoop	
Description	No 360° rotation around the y-axis
Syntax	Boolean

NoSetLoop	
Default	false
Context	FSI Viewer

The first and the last scene set will be considered ending points. (E.g. for partial rotations < 360°).

Please note: The blur animation is turned off if NoSetLoop is activated.

NoSceneLoop	
Description	No 360° rotation around the x-axis
Syntax	Boolean
Default	false
Context	FSI Viewer

The first and last scene in the each scene set will be considered ending points. (E.g. for partial rotations < 360°).

Please note: The blur animation is turned off if NoSceneLoop is activated.

Display Options

hideUI	
Description	hides the user interface
Syntax	string
Default	false
Context	FSI Viewer

Hides or displays the user interface.

Possible values:	
true	
false	
onMobileDevices:	UI is hidden on mobile devices
ifNoInPlaceZoom:	UI is hidden if in place zoom is deactivated

noNav	
Description	hides the navigation window
Syntax	boolean
Default	false
Context	FSI Viewer

Hides or displays the navigation window.

autoHideNav	
Description	automatically hides the navigation window
Syntax	boolean
Default	false
Context	FSI Viewer

Hides or displays the navigation window automatically if the magnification is at minimum.

autoSpinSpeed	
Description	Time in seconds for a full rotation
Syntax	float
Default	-

autoSpinSpeed	
Context	FSI Viewer

Time in seconds for a full rotation. Negative values reverse spin direction.

autoSpinInterval	
Description	defines pause between every step in ms
Syntax	float
Default	-
Context	FSI Viewer

Can be used instead of autoSpinSpeed to define the duration of the AutoSpin. autoSpinInterval defines the pause between every step of the rotation in ms. Negative values reverse spin direction. The hierarchy of the parameters is as follows:

- autoSpin_interval (Plugin Parameter)
- autoSpinInterval (Config Parameter)
- autoSpin_speed (Plugin Parameter)
- autoSpinSpeed (Config Parameter)

enableZoom	
Description	enable the user to zoom in and out
Syntax	Boolean
Default	true
Context	FSI Viewer

Enable the user to zoom in and out.

On touch enabled devices the zoom is based on the native pinch zoom gesture. On non touch enabled devices, in place zoom by clicking the object or dragging

up/down will be enabled instead. Alternatively, the menu bar can be used for zooming.

desktopClickZoomScale	
Description	the magnification level to use when clicking the object
Syntax	Float
Default	3
Context	FSI Viewer

For desktop browsers only. The magnification level to use when clicking the object.

backgroundColor	
Description	the background color to use
Syntax	6-digit hex-color
Default	#FFFFF
Context	FSI Viewer

The background color to use. The color should match the video's or image's background color (usually white or black).

mouseSensitivity	
Description	sensitivity of the mouse movement when rotating the object
Syntax	float
Default	1.0
Context	FSI Viewer

Sensitivity of the mouse (finger) movement when rotating the object.

pellets	
Description	viewer draws optionally small dots on the video
Syntax	boolean
Default	true
Context	FSI Viewer

When using a spin video you can optionally have the viewer draw small dots on the

video. This improves the visual impression of the video in higher resolutions.

pelletsOpacity	
Description	defines opacity of the dots ("pellets") drawn on the video
Syntax	float
Default	0.09
Context	FSI Viewer

Opacity of the dots ("pellets") drawn on the video, if the pellets parameter is enabled.

adaptiveUISize	
Description	determines if interface scales according to the zoom level .
Syntax	Boolean
Default	true
Context	FSI Viewer

When set to true, the interface scales according to the zoom level.

inPlaceZoom	
Description	activates zooming in the FSI Viewer instance
Syntax	string
Default	"true"
Context	FSI Viewer

The parameter defines if in-place-zoom (zooming directly in the FSI Viewer instance)is activated.

Possible values:	
true	
false	
auto:	in-place-zoom is activated, but not on mobile devices

documentZoom	
Description	allows pinch-zoom on FSI Viewer instance
Syntax	string
Default	"auto"
Context	FSI Viewer

Activates pinch document zoom on FSI Viewer instance.

Possible values:	
true	

false	
auto:	pinch document zoom is activated on all touch-enabled devices

autoDisablePointerActions	
Description	disables events if viewer is larger than viewport
Syntax	Boolean
Default	true
Context	FSI Viewer

Disables mouse and touch events if viewer's height OR width is larger than the viewport width or height.

loadExactPreviewImagesAfterResize	
Description	Reload images in the new required resolution after a resize event
Syntax	Boolean
Default	"true"
Context	FSI Viewer

Reload images in the new required resolution after a resize event.

maxZoom	
Description	defines the maximum magnification level
Syntax	Float
Default	100
Context	FSI Viewer

Defines the maximum magnification level.

minInstanceSize	
Description	defines the minimum instance size
Syntax	Integer
Default	60
Context	FSI Viewer

Defines the minimum instance size of the viewer.

Preview Options

showPreviewWhileLoading	
Description	shows/hides preview images for spins while loading
Syntax	boolean
Default	true
Context	FSI Viewer

If this parameter is set to false, the preview images which are shown while the spin is loaded are hidden.

spinPreviewWhileLoading	
Description	shows/hides preview images for spins while loading
Syntax	boolean
Default	true
Context	FSI Viewer

If this parameter is set to false,a static image will be shown instead of the spin while loading. If initialView is defined, the image defined there will be shown.

showProgressbar	
Description	shows/hides progess bar while loading
Syntax	boolean
Default	true
Context	FSI Viewer

If this parameter is set to false, the progress bar which isshown while the spin is loaded is hidden.

progressBarWidth	
Description	defines the width of the progress bar
Syntax	float
Default	35
Context	FSI Viewer

Defines the width of the progress bar in percent of the viewport width.

progressBarHeight	
Description	defines the height of the progress bar
Syntax	float
Default	14
Context	FSI Viewer

Defines the height of the progress bar in pixel.

progressBarHAlign	
Description	defines the horizontal position of the progress bar
Syntax	string
Default	"center"
Context	FSI Viewer

Defines the horizontal position of the progress bar, possible values: left, center, right.

progressBarVAlign	
Description	defines the vertical position of the progress bar
Syntax	string
Default	"bottom"
Context	FSI Viewer

Defines the horizontal position of the progress bar, possible values: top, middle (or center), bottom.

Global Parameters for Hot Spots

enableHotspots	
Description	enables or disables hot spots
Syntax	boolean
Default	true
Context	FSI Viewer

You can set this value to false if hot spots have been defined, but should not be displayed.

hotspotCallbackFunction	
Description	callback function for hot spots
Syntax	string/ function
Default	""
Context	FSI Viewer

The name of a custom JavaScript callback function in window scope to call each time a hot spot mouse action occurs.

Alternatively you can pass the function directly as a JavaScript parameter.

Example of a callback function prototype:

Callback Function Prototype:

```
function onFSIViewerHSAction(strAction, viewer, idViewer, idHotspot,
node, param, angle) {
  if (strAction == "click") {
    alert("clicked on hot spot " + idHotspot);
  }
}
```

Attribute	Description
action	a string representing the action "mouseover", "mouseout", "mousedown" or "click"

Attribute	Description
viewer	the FSI Viewer viewer instance calling the method
idViewer	the value of the "id" attribute of the viewer's tag
idHotspot	the 0-based id of the hotspot
node	the node containing the hot spot content
param	the content of the "actionParameter" parameter of the hot spot
angle	the hot spot's perimeterAngle value (the angle at which it's visible best)

hotspotLineColor1	
Description	lighter color of the hot spot marker lines
Syntax	6-digit hex color
Default	#FFFF00
Context	FSI Viewer

The lighter color of the hot spot marker lines.

hotspotLineColor2	
Description	darker color of the hot spot marker lines
Syntax	6-digit hex color
Default	#000000
Context	FSI Viewer

The darker color of the hot spot marker lines.

hotspotDotColor1	
Description	lighter color of the hot spot marker lines
Syntax	6-digit hex color
Default	#FFFF00
Context	FSI Viewer

The lighter color of the hot spot marker circle's gradient.

hotspotDotColor2	
Description	darker color of the hot spot marker lines
Syntax	6-digit hex color
Default	#000000
Context	FSI Viewer

The darker color of the hot spot marker circle's gradient.

decodeHTMLEntitiesInHotSpotTexts	
Description	enables html entities in the hotspots text
Syntax	6-digit hex color
Default	#000000
Context	FSI Viewer

When enabled, basic HTML entities will be enabled in the hotspot text.

hotspotDotRadius	
Description	radius of the hot spot marker circle in px
Syntax	float

hotspotDotRadius	
Default	4.0
Context	FSI Viewer

The radius of the hot spot marker circle in px.

Camera Section

The camera section contains parameters describing the technical specifications and the position of the camera when shooting the 360° spin.

You only need to specify these values if you are using the virtual space, e.g. for hot spots.

For 2D zoom and spins without hot spots you can omit this parameter group.

digitalCropFactor	
Description	depends on the size of the optical sensor chip of the camera
Syntax	float
Default	1.0
Context	FSI Viewer

The value depends on the size of the optical sensor chip of the camera.

For full frame cameras the value is "1.0". Common values are between 1.5 to 1.6. The crop factor is the same factor that affects the focal length of your camera's objective. E.g. the focal length of an objective with a full frame camera is 50mm. With a camera that has a crop factor of 1.5 the effective focal length would be 1.5 x 50mm = 75mm.

Please note that for an accurate calculation the rounded figure (e.g. 1.5) is not precise enough, since two to four decimal places are required.

In order to ensure an exact smooth video and hotspot implementation, please do the following calculation for specific sensor dimensions of the camera used. The size of the optical sensor chip can be found in the cameras technical specifications.

$$\frac{\sqrt{36^2 + 24^2}}{\sqrt{x^2 + y^2}} = \text{digitalCropFactor}$$

$36^2 + 24^2 =$ Reference value (35mm "full frame" format; 36mm x 24mm, crop factor: 1.0)

$x^2 + y^2 =$ Sensor dimensions of your camera (width and height)

Please find below the calculation for a Canon 7D camera (which has the sensor dimensions of 22.3mm x 14.9mm) as an example:

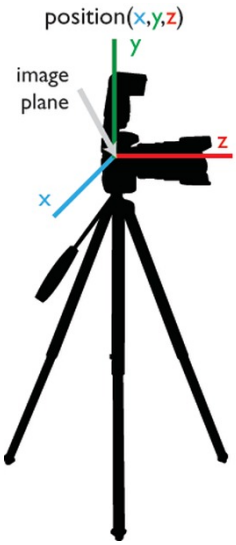
$$\frac{\sqrt{36^2 + 24^2}}{\sqrt{22.3^2 + 14.9^2}} = 1.6132$$

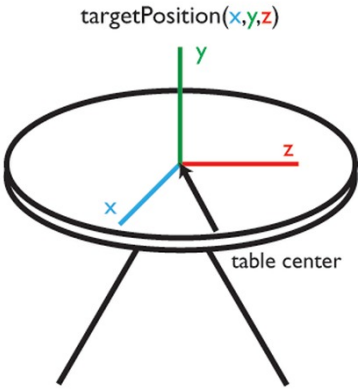
focalLength	
Description	focal length of the camera objective
Syntax	float (in mm)
Default	50
Context	FSI Viewer

The focal length of the camera objective.

position, targetPosition	
Description	defines the position of the camera and the object
Syntax	float (in mm)
Default	
Context	FSI Viewer

postion(x,y,z) and targetPosition(x,y,z) can be used instead of distanceToTableCenter, heightAboveTable and targetHeightOffset. (see illustration below)





Explanation position(x,y,z)	
x	if image plane is in line with table center, x is 0
y	height of the image plane in reference to the table center
z	the distance between image plane and table center

Explanation targetPosition(x,y,z)	
x	if target is is in table center, x is 0
y	height of the target in reference to the table center
z	the distance between target and table center

distanceToTableCenter	
Description	distance from cameras image plane to the center of the turntable
Syntax	float (in mm)
Default	0
Context	FSI Viewer

The straight distance from the cameras image plane to the center of the turn table in mm (see illustration below).

heightCamera	
Description	distance from image plane of the camera to the ground
Syntax	float (in mm)
Default	0
Context	FSI Viewer

The straight distance from the optical center of your camera to the ground in mm (see illustration below).

heightTable	
Description	distance from top of the turn table to the ground
Syntax	float (in mm)
Default	0
Context	FSI Viewer

The straight distance from the top of the turn table to the ground in mm. (see illustration below)

targetHeightOffset	
Description	height of the camera target spot at the center of the turntable
Syntax	float (in mm)
Default	0
Context	FSI Viewer

The height of the camera target spot at the center of the turn table in mm.

If you vertically target the table center the value is 0.0 mm.

(see illustration below)

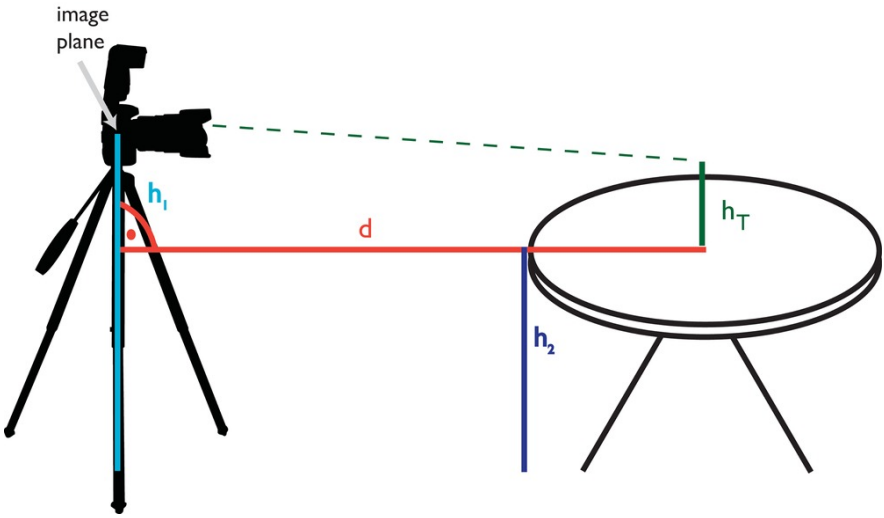
targetWidthOffset	
Description	horizontal offset of the camera target spot
Syntax	float (in mm)
Default	0
Context	FSI Viewer

The horizontal offset of the camera target spot at the vertical the center of the turn table related to the center of the turn table in mm. If you target the horizontal table center (default) the value is 0.0 mm.

heightAboveTable	
Description	height of the image plane above the table
Syntax	float (in mm)
Default	500
Context	FSI Viewer

Alternative parameter if heightCamera and heightTable are unkown. Only applies if both parameters are undefined.

Camera Positioning Illustration



Attribute	Description
hT	Parameter: targetHeightOffset
h1	Parameter: heightCamera
h2	Parameter:heightTable
d	Parameter: distanceToTableCenter

Usage of Robotic Camera Arm for rotations around the x-axis

armLength	
Description	length of the robot arm
Syntax	float (in mm)
Default	
Context	FSI Viewer

Defines the lenght of the robot arm; default value is the distance of the camera to the camera target. Only relevant if hotspots are used.

armJointPosition	
Description	position of the arm joint
Syntax	float (in mm)
Default	
Context	FSI Viewer

Defines the position (x,y,z) of the robot arm joint; default value is equal to the camera targetPostion. Only relevant if hotspots are used.

armAngleMeasureSetup	
Description	angle where distance from camera to target is measured
Syntax	float (in degrees)
Default	0
Context	FSI Viewer

Defines the angle from which distance from camera to target was measured. Only relevant if hotspots are used.

armStartAngle	
Description	defines the start angle
Syntax	float (in degrees)
Default	-90
Context	FSI Viewer

Defines the start angle of the rotation around the x-asis. Only relevant if hotspots are used.

armEndAngle	
Description	defines the end angle
Syntax	float (in degrees)
Default	90
Context	FSI Viewer

Defines the end angle of the rotation around the x-axis. Only relevant if hotspots are used.

VirtualSpace Section

The virtualSpace section is optional and only required if you want to verify the 3D projections for use with hot spots. Using the camera parameters, the viewer calculates the position of the hot spots based on the given 3D coordinates only.

cubeSizeX, cubeSizeY and cubeSizeZ	
Description	"cube" is the bounding box of the spin object in millimeters
Syntax	float
Default	-

cubeSizeX, cubeSizeY and cubeSizeZ	
Context	FSI Viewer

The "cube" is the bounding box of the spin object in millimeters.

cubeLineWidth	
Description	line width of the cube
Syntax	float
Default	-
Context	FSI Viewer

Line width of the drawn cube. Only applies if the parameter "drawCube" is activated.

baseRotationY, baseRotationX, baseRotationZ	
Description	specifies a rotation offset in degrees
Syntax	float
Default	0
Context	FSI Viewer

You can specify a rotation offset in degrees.

This is required only, if you started the 360° shooting session at an angle other than 0°.

rotationXRange, rotationYRange	
Description	specifies the range of the rotation
Syntax	float
Default	180 (X) or 360 (Y)

rotationXRange, rotationYRange	
Context	FSI Viewer

Specifies the range of the rotation of a certain axis.

rotationXSets	
Description	defines the inital amount of scene sets
Syntax	integer
Default	number of scene sets
Context	FSI Viewer

Defines the inital amount of scene sets in order to ensure the correct posture of hot spot in case your presentation requires the value in the parameter SceneSets to differ from the initial amount.

rotationYScenes	
Description	defines the inital amount of scenes
Syntax	integer
Default	number of scenes
Context	FSI Viewer

Defines the inital amount of scenes in order to ensure the correct posture of hot spot in case your presentation requires the value in the parameter SceneSets to differ from the initial amount.

shiftX, shiftY and shiftZ	
Description	defines an offset of the spin object on the turn table in mm

shiftX, shiftY and shiftZ	
Syntax	float
Default	-
Context	FSI Viewer

Optionally defines an offset of the spin object on the turn table in mm.

position	
Description	defines an offset of the spin object on the turn table in mm
Syntax	float
Default	-
Context	FSI Viewer

Alternative parameter to shiftX,shiftY, shiftZ - positon(x,y,z) is optional and defines an offset of the spin object on the turn table in mm.

HotSpot Section

The hotspots3d section is optional and only required if you want to display hot spots on your 360° object spin. Hot Spots consist of a marker line and a marker dot as well as content area. The content area may contain all kind of HTML content, e.g. formatted text, hyperlinks, images, videos and alike.

Each hotspot requires an individual hot spot definition node.

In addition a <defaults> node can be used to define default parameter values.

HotSpot definition mode

```
<hotspots3d>
   <defaults>
      <normalClass value="hotspotNormal"/>
      <hoverClass value="hotspotOver"/>
      <activeClass value="hotspotAction"/>
      <perimeterAngleFadeOut value="15" />
   </defaults>
   <hotspot>
      <text value="Hot Spot A "/>
      <x value="11"/>
      <y value="223"/>
      <z value="-116"/>
      <perimeterAngle value="0"/>
      <perimeterAngleRange value="60"/>
   </hotspot>
   <hotspot>
      <text value="Hot Spot B "/>
      <x value="11"/>
      <y value="223"/>
      <z value="-116"/>
      <perimeterAngle value="90"/>
      <perimeterAngleRange value="30"/>
      <perimeterAngleFadeOut value="5" />
   </hotspot>
[...]
```

Hot Spot Parameters:

x, y and z	
Description	x, y, z distance on the real object measured from center of object
Syntax	float (in mm)
Default	-

x, y and z	
Context	FSI Viewer

The x, y and z distance in mm on the real object measured from the center of the object (usually equal to the center of the turn table).

Combined with the required <camera> parameters, the viewer calculates the rotated and projected position of the hot spot on screen.

text	
Description	HTML content to display for this hot spot
Syntax	string
Default	""
Context	FSI Viewer

The HTML content to display for this hot spot.

Note that the value needs to be XML encoded if the content contains invalid characters for XML attribute values (e.g. '&' for '&', '<' for '<', '>' for '>' and '"' for ").

horizontalMarkerSize	
Description	length of the horizontal line of the hot spot's marker
Syntax	integer
Default	30
Context	FSI Viewer

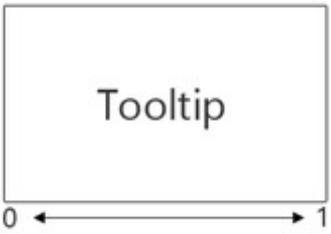
Length of the horizontal line of the hot spot's marker. Range: [-x ... 0 ... x]

verticalMarkerSize	
Description	length of the vertical line of the hot spot's marker
Syntax	integer
Default	100
Context	FSI Viewer

Length of the vertical line of the hot spot's marker. Range: [0 ... x]

horizontalMarkerPosition	
Description	horizontal position of the marker line
Syntax	float
Default	0
Context	FSI Viewer

Defines the horizontal position of the marker line on the tooltip. Range: [0.0 ... 1.0] The value 0.0 marks the left corner of the tooltip while 1 marks the right corner.



verticalMarkerPosition	
Description	vertical position of the marker line
Syntax	float
Default	0.5

verticalMarkerPosition	
Context	FSI Viewer

Defines the vertical position of the marker line on the tooltip. Range: [0.0 ... 1.0] The value 0.0 marks the top corner of the tooltip while 1 marks the bottom corner.



markerAngle	
Description	angle of the vertical hot spot marker
Syntax	float
Default	45
Context	FSI Viewer

Defines the angle of the vertical hot spot marker.

perimeterAngle	
Description	viewing angle in degrees at which to display the hot spot
Syntax	float
Default	-
Context	FSI Viewer

The viewing angle in degrees at which to display the hot spot.

E.g. for a hot spot on the left side of the object use "90".

For a hot spot at the back of the object use "180".

Alias: perimeterAngleY

perimeterAngleX	
Description	viewing angle in degrees at which to display the hot spot
Syntax	float
Default	-
Context	FSI Viewer

The viewing angle in degrees at which to display the hot spot.

Only applies if a rotation around the x-axis is used.

perimeterAngleRange	
Description	angle range in which the hot spot shall be visible
Syntax	float
Default	30
Context	FSI Viewer

The angle range in which the hot spot shall be visible.

E.g. a perimeterAngle of "90" and a perimeterAngleRange of "40" would display the hot spot from 70 to 110 degrees (+/- 20°).

Alias: perimeterAngleRangeY

perimeterAngleRangeX	
Description	angle range in which the hot spot shall be visible
Syntax	float

perimeterAngleRangeX	
Default	30
Context	FSI Viewer

The angle range in which the hot spot shall be visible.

Only applies if a rotation around the x-axis is used.

perimeterAngleFadeOut	
Description	angle range in which the hot spot shall be faded in and out
Syntax	float
Default	10
Context	FSI Viewer

The angle range in which the hot spot shall be faded in and out.

A value of "0" means that the hot spot will be displayed or hidden immediately when entering or leaving the perimeterAngleRange. Values greater than 0 extend the perimeterAngleRange by the given degrees in which the hot spot's opacity changes.

Alias: perimeterFadeOutY

perimeterAngleFadeOutX	
Description	angle range in which the hot spot shall be faded in and out
Syntax	float
Default	10
Context	FSI Viewer

The angle range in which the hot spot shall be faded in and out.

A value of "0" means that the hot spot will be displayed or hidden immediately when entering or leaving the perimeterAngleRange. Values greater than 0 extend the perimeterAngleRange by the given degrees in which the hot spot's opacity changes.

Only applies if a rotation around the x-axis is used.

perimeterZoom	
Description	specifies from which magnification level on a hot spot is displayed
Syntax	float
Default	1.0
Context	FSI Viewer

An optional parameter specifying from which magnification level on a hot spot should be displayed. E.g. 2.5 would hide the hot spot as long as the magnification level is below 2.5. This parameters has no effect if zoom is disabled by the global "enableZoom" parameter.

perimeterImageWidth	
Description	defines the absolute width of spin instance for displaying hot spot
Syntax	float
Default	1.0
Context	FSI Viewer

Similar to perimeterZoom. While perimeterZoom depends on the magnification this parameter defines the absolute width of the (zoomed) spin instance to decide whether to show or hide a tooltip. It is recommend to set this parameter as a default hot spot parameter when using relative instance sizes.

interactive	
Description	enables or disables reactions of hot spot areas to interactive actions
Syntax	boolean
Default	false
Context	FSI Viewer

The hot spot content area will not react on mouse over, click actions and alike when setting "interactive" to false.

actionParameter	
Description	value will be passed on to the hot spot callback function
Syntax	string
Default	""
Context	FSI Viewer

An optional parameter value that will be passed on to the hot spot callback function.

url	
Description	An optional URL to open if the user clicks the hot spot label.
Syntax	string
Default	""
Context	FSI Viewer

An optional URL to open if the user clicks the hot spot label.

You can use any URL you can use with the window.open command, like regular URLs as well as "javascript:" or "email:" URLs.

urlTarget	
Description	target frame or window to open the URL in
Syntax	string
Default	_self
Context	FSI Viewer

The target frame or window to open the URL in.

Use e.g. "_blank" to open hyperlinks in a new window.

focusOnClick	
Description	zoom and/or rotate to hotspot position on click
Syntax	boolean
Default	true
Context	FSI Viewer

Enables zooming and/or rotating to hotspot position on click.

rotateOnFocus	
Description	rotate to sceneSet and scene
Syntax	boolean
Default	true
Context	FSI Viewer

Setting this parameter to true rotates to sceneSet and scene matching the coordinates defined in hotspot.perimeterAngle, while false keeps the current sceneSet and scene.

zoomOnFocus	
Description	zoom to hotspot
Syntax	boolean
Default	true
Context	FSI Viewer

Setting this parameter to true zooms to thehotspot, while false keeps the current zoom.

focusRadius	
Description	size of section when zooming to a hotspot
Syntax	float
Default	0.25
Context	FSI Viewer

Defines the size of the sectionwhen zooming to a hotspot (0...1).

focusAngleX, focusAngleY	
Description	defines scene with given focus angle (optional)
Syntax	float
Default	-
Context	FSI Viewer

If set, the target scene or sceneSet will not be determined by the perimeterAngle parameter, but with the given focusAngle.

normalClass	
Description	CSS class name to use for content tag in normal state
Syntax	string
Default	"FSI360HotSpot_Normal"
Context	FSI Viewer

A CSS class name to use for the content tag in normal state.

activeClass	
Description	CSS class name to use for content tag in mouse onclick state
Syntax	string
Default	"FSI360HotSpot_Active"
Context	FSI Viewer

A CSS class name to use for the content tag in onclick state.

hoverClass	
Description	CSS class name to use for content tag in mouse over state
Syntax	string
Default	"FSI360HotSpot_Active"
Context	FSI Viewer

A CSS class name to use for the content tag in mouse over state.

Example CSS classes:

It is recommended to add the following CSS classes to the HTML document containing FSI Viewer instances with hot spots. These classes will be used if you do not define "normalClass", "hoverClass" and "activeClass" parameters for your hot spots. The _Mobile und _SmallMobile classes will be added to the hot spot class names on mobile device or small mobile devices (phones). You can use them to adjust the font size to increase the readability of text within the hot spots.

Example CSS classes

```
<style type="text/css">


div.FSI360HotSpot_Normal{
   border:2px solid rgba(255, 255, 0, 0.8);
   font-family:Arial,Helvetica,sans-serife;
   color:#000;
   font-size:13px;
   background: rgba(255, 255, 200, 0.8);
   padding:3px;
   border-radius: 3px;
   cursor:pointer;
   box-shadow: 4px 4px 4px rgba(0,0,0,0.3);
   white-space:nowrap;
}

div.FSI360HotSpot_Active{
   background: #FFF;
}

div.FSI360HotSpot_Mobile{
   font-size:13px;
}
```

```
div.FSI360HotSpot_SmallMobile{
   font-size:32px;
}
</style>
```

cssClass	
Description	CSS class name to add to the hot spot's content tag
Syntax	string
Default	""
Context	FSI Viewer

An additional CSS class name to add to the hot spot's content tag.

cssStyle	
Description	CSS style value to add to the hot spot's content tag
Syntax	string
Default	""
Context	FSI Viewer

An additional CSS style value to add to the hot spot's content tag.

Video Section

The video section is optional and only required if you want to enable ultra smooth rotations by providing a video of the spin.

source	
Description	CSS style value to add to the hot spot's content tag
Syntax	string
Default	""
Context	FSI Viewer

The value may be an absolute URL. Otherwise it will be relative to the [viewer]/applications/spin360/videos/ folder.

> **NOTE:** Just provide the file name WITHOUT the file extension.

Please refer to "Creating Videos" below for more information on spin videos.

sceneSet	
Description	defines in which Scene Set the video is used
Syntax	string
Default	""
Context	FSI Viewer

If Scene Sets are used, this parameter defines in which Scene Set the video will be used for smooth rotation.

base	
Description	optional prefix for all "source" parameters
Syntax	string
Default	""
Context	FSI Viewer

An optional prefix for all "source" parameters.

spinDirection	
Description	defines the spin direction of the video
Syntax	String "left" or "right"
Default	"left"
Context	FSI Viewer

Defines the spin direction of the video.

You might want to change the spinDirection option if the object spins into the wrong direction when dragging the mouse left or right.

captureFrames	
Description	number of frames to capture from the video
Syntax	integer
Default	120
Context	FSI Viewer

The number of frames to capture from the video.

The more frames you define the smoother the rotation will appear. More frames require more time to capture the videos on startup. 120 frames (default) means 3° per frame.

captureOffsetStart	
Description	offset at start of video to use when capturing the video frames
Syntax	float (in seconds)
Default	0
Context	FSI Viewer

An optional offset at the start of the video to use when capturing the video frames. You might want to define an offset if the video does not start exactly at the position of the first image (0°).

captureOffsetEnd	
Description	offset at end of video to use when capturing the video frames
Syntax	float (in seconds)
Default	0
Context	FSI Viewer

An optional offset at the end of the video to use when capturing the video frames. You might want to define an offset if the video does not start exactly at the position of the first image (0°).

widthCorrectionFactor	
Description	adjusts width of the video to width of the images on screen
Syntax	float
Default	1.0
Context	FSI Viewer

An optional factor to adjust the width of the video to the width of the images on screen. You might want to adjust the value if the image do not match the video dimension.

heightCorrectionFactor	
Description	adjusts height of the video to height of the images on screen
Syntax	float
Default	1.0
Context	FSI Viewer

An optional factor to adjust the height of the video to the height of the images on screen. You might want to adjust the value if the image do not match the video dimension.

xCorrectionFactor	
Description	adjusts horizontal position of video to horizontal position of images
Syntax	float
Default	0.0
Context	FSI Viewer

An optional factor to adjust the horizontal position of the video to the horizontal position of the images on screen. You might want to adjust the value if the image do not match the video position.

yCorrectionFactor	
Description	adjusts vertical position of video to vertical position of images

yCorrectionFactor	
Syntax	float
Default	0.0
Context	FSI Viewer

An optional factor to adjust the vertical position of the video to the vertical position of the images on screen. You might want to adjust the value if the image do not match the video position.

frameOffset	
Description	determines a frame offset if required
Syntax	integer
Default	0
Context	FSI Viewer

Determines a frame offset at the start if required.

2.6.2 Plugin Parameters

plugins	
Description	adds a Plug-in (e.g. Fullscreen) to the viewer
Syntax	String
Default	""
Context	FSI Viewer

Adds a plug-in to the viewer.

Please note:

If you would like to add specific parameters to the plugins via the fsi-viewer tag, you can do this by prefixing the parameter name with the name of the plug-in, e.g. autospin_enabled="true"

FullScreen Plug-In

Adding the parameter plugins:"FullScreen, Resize" enables a full screen view option in FSI Viewer. The Resize parameter needs to be set in order to ensure proper adjusting to the full screen.

Note:

If the FSI Viewer is embedded in an iFrame, the attribute allowfullscreen="true" needs to be added to the iFrame. Otherwise the FullScreen Plug-in will not work due to security restrictions of the browser.

AutoSpin Plug-In

available from version 16.09.30

Adding the parameter plugins:"AutoSpin" enables a button which lets you play/pause the animation that is set with the autospinspeed parameter. Parameters are listed as the following in the fsi-viewer tag: ***autoSpin_Speed***, ***mouseModes_MenuOffset***, etc.

The following parameters can be set:

Speed	
Description	Time in seconds for a full rotation
Syntax	float
Default	-
Context	FSI Viewer

Has the same function as "autoSpinSpeed", but is used here in the context of the plug-in. Time in seconds for a full rotation. Negative values reverse spin direction.

Interval	
Description	Time in seconds for a full rotation
Syntax	float
Default	-
Context	FSI Viewer

Has the same function as "autoSpinInterval", but is used here in the context of the plug-in.

Can be used instead of autoSpinSpeed to define the duration of the AutoSpin. autoSpinInterval defines the pause between every step of the rotation in ms. Negative values reverse spin direction. The hierarchy of the parameters is as follows:

- autoSpin_interval (Plugin Parameter)
- autoSpinInterval (Config Parameter)
- autoSpin_speed (Plugin Parameter)
- autoSpinSpeed (Config Parameter)

preventClickZoomWhileSpinning	
Description	prevents zooming while Autospin is activated
Syntax	Boolean

preventClickZoomWhileSpinning	
Default	false
Context	FSI Viewer

Prevents zooming while clicking into the Autospin.

Button	
Description	enables Play/Pause button
Syntax	Boolean
Default	true
Context	FSI Viewer

Enables the Play/ Pause button for the autospin animation

MouseModes Plug-In

The MouseMode plugin is activated by default with the _default.xml.

The following parameters can be set:

MenuOffset	
Description	Indentation of the menu buttons
Syntax	Integer
Default	0
Context	FSI Viewer

Specifies the space in pixels left of the menu button(s) of the plug-in.

Mode *n*	
Description	Removes the button for mouse mode n from the menu bar
Syntax	Boolean
Default	-
Context	FSI Viewer

Provides the possibility to hide specific mouse mode buttons.

Mode n	**Mouse Mode**
 Mode0	Zoom
 Mode1	Pan
 Mode2	Rotate 3D (X/Y Axis)
 Mode3	Rotate 2D (Z Axis)

Sequence	
Description	Sequence of the buttons
Syntax	String
Default	0,1,3,2
Context	FSI Viewer

You can alter the sequence of the buttons by providing the modes separated by commas (see the table above).

Example: "1,0,2,3" alters the sequence of the buttons to "Pan, Zoom, Rotate 3D, Rotate 2D".

2.7 Creating and using Spin Videos

When creating spin videos for ultra smooth rotations, it is strongly recommended to use a camera that can take photos and record videos so that the position and technical specifications apply to both, the video and the photos.

Make sure you do not modify the object or camera position between the video and photo shooting session.

1. Take the photos from different viewing angles
2. Record a video of a full, continuous object spin starting at the angle you took the first photo from until reaching the same angle again. You might want to take a video of 2 full spins and cut the video later on.
3. Cut the video if required making sure the video starts and ends at the viewing angle you took the first photo at
4. Export the video in the following formats:
   webm (for Chrome, Firefox, Safari, etc.)
   mp4 (for MS IE)
5. Copy the video files to a static source connector on FSI Server using the same file names just with the file extension matching the video format
6. Add the <video> section to your configuration file and add the "source" parameter (just the file name WITHOUT the file extension).

2.8 Embedding the viewer at runtime

It is also possible to embed the viewer at runtime via JavaScript.

I) Passing parameters via node attribute:

Example:

```
var node = document.createElement("fsi-viewer");

node.setAttribute("width", "400");
node.setAttribute("height", "600");
node.setAttribute("src", "images/Rocker Shoe.tif");
node.setAttribute("debug", "true");

document.body.appendChild(node);

// Important: required if you add the node AFTER the DOMContent has
already been loaded
$FSI.initCustomNode(node);
```

II) Passing parameters via JavaScript object:

⚠ NOTE: In this case any parameters passed via node attribute will be ignored.

Example:

```
var node = document.createElement("fsi-viewer");
```

```
node.style.width = "600px";
node.style.height = "400px";

var parameters =
{"imagesrc" : "images/Rocker Shoe.tif", "skin":"silver", "debug":true};

$FSI.setParameters(node, parameters);
document.body.appendChild(node);

// Important: required if you add the node AFTER the DOMContent has
already been loaded
$FSI.initCustomNode(node);
```

Note:

$FSI.initCustomNode(node) initializes the given node.

When adding multiple viewers you can initialize all new FSI Viewer nodes in one go, using this command:

`$FSI.initCustomTag("fsi-viewer");`

or, when using different custom tags (fsi-viewer, fsi-imageflow,…):

`$FSI.initCustomTags();`

to initialize all new FSI custom tags.

Destroying Instances

Before finally removing <fsi-viewer> nodes from the DOM, you need to call the destroy() method of the node.

This will release all event handlers and free allocated resources.

Example:

```
var node = document.getElementById("myViewer");

node.destroy();

node.parentNode.removeChild(node);
```

2.9 JavaScript Interface

2.9.1 Public Methods

The JavaScript interface of FSI Viewer can be useful if you want to change the current image or config or add additional content depending on the object spin position or hot spot actions.

void	changeConfig(strCfgFileName, oParameters);
	Reset the viewer object and change the configuration to the given configuration file. strCfgFileName (optional): path to the configuration file (see "cfg" parameter) oParameters (optional): an object containing parameters If present, these parameters will overwrite the parameters defined upon initialization.

Example:

```
var oParameters = {};
   oParameters src = "images/bluemarble.tif";
/*
// optional:
   oParameters.imageWidth;
   oParameters.imageHeight;
   oParameters.cropRect;
   oParameters.cropValues;
   oParameters.effects;
*/

fsiviewer.changeConfig(oParameters);
```

bSuccess	changeImage(cfg, strView);
	Resets the viewer object and changes the image displayed to the given image file.

void	**start();**
	Starts the viewer.

void	**printAPI();**
	Shows all API methods in the console in alphabetical order.

bEnabled	**debugEnabled();**
	Returns if the debug mode is enabled.

mixed	**debugError();**
	Returns the corresponding viewer error messages in the console.

mixed	**debugLog();**
	Returns the corresponding viewer log messages in the console.

mixed	**debugWarn();**
	Returns the corresponding viewer warning messages in the console.

void	**init(elementID, oParameters, bDebug);**
	Initialises the chosen viewer element.

mixed	**getConfigValue(strName);**
	Returns the value of the configuration parameter "strName".

mixed	**traceConfigValue(strName);**
	Traces a config value and enables you to see where (in config files, the _default.xml, etc) a specific value is set.

bInitDone	**getInitDone();**
	Returns if the initialization process is done.

void	**spinToDegree(fDegree);**
	Spins to the position indicated by degree. Note that you can only spin to a position that has a corresponding HD image, not to a position that shows a video only. fDegree: the desired position in degree (0 .. 360)

void	**spinToTarget(nTargetFrame, nStopSceneSet, nFrames);**
	Spins to the position indicated by the target parameters. Note that you can only spin to a position that has a corresponding HD image, not to a position that shows a video only.

nMousemode	**getMouseMode();**
	Returns the current active mouse mode.

void	**setMouseMode(nMode);**
	Sets the active mouse mode.

nInstanceID	**getInstanceID();**
	Returns the number of the viewer object.

strInstanceName	**getInstanceName();**
	Returns the name of the viewer object.

void	**destroy();**
	Destroys the given FSI Viewer object. You should destroy the instance before you remove the object spins tag from the DOM tree.

strVersion	**getVersion();**
	Returns a string containing the FSI Viewer software version.

strVersion	**getBuild();**
	Returns a string containing the FSI Viewer software build number.

nHeight	**getMenuHeight(strPurpose);**
	Returns a string containing the height of the menu bar.

bSuccess	**addListener(strListenerName, fn, iScope);**
	Calls a specific listener. Listeners which are added via addListener(), remain valid when changeConfig() is called.

bSuccess	**removeListener(strListenerName, fn);**
	Removes a specific listener.

void	**showAboutWindow();**
	Shows the "About" window.

void	**showMenuButton(strButtonID, bShow);**
	Shows a specific menu button.

void	**updateSize();**
	Updates the size of a fsi-viewer element.

void	**resetView();**
	Reset the viewer to the initial magnification and rotation.

fZoomInPercent	**getZoom();**
	Returns the current magnification level..

void	**setZoom(fPercent, bPreliminary, bAnimate);**
	Set magnification to the amount indicated by fPercent [0...100], where fPercent = 0 means normal scale and fPercent = 100 for maximum magnification. bPreliminary: Set to true if the magnification level is temporary only. In this case no additional image data will be loaded. bAnimate:Set to true to animate from the current to the given magnification level.

strViewString	**getViewString();**
	Returns the string representing the current image section and rotation. The format is: SceneSet, Scene, left, top, right, bottom ***SceneSet*** is 1 for rotations around one axis ***Scene*** describes the rotation. If you use 36 images for a spin, scene 18 means a rotation of 180 degree. ***left, top, right, bottom***: image coordinates in the range [0.0 to 1.0].

bSuccess	**gotoViewString(strViewString, bAnimated);**
	Display the image section and rotation as specified by "strViewString". Please refer to the description of the "getViewString" method for a format description. The boolean "bAnimated" can be used to specify, wether the view should be displayed immediately or in form of an animated movement from the current view to the specified view.

strViewString	**getHotspotViewString(nHotspotID)**
	returns the view string for the given hotspot. FSIHotspotID is the consecutive number of the Hotspot (1,2,3,..)

bSuccess	**gotoHotspot(FSIHotspotID, bAnimate)**
	displays the image section as specified by the Hotspot. FSIHotspotID is the consecutive number of the Hotspot (1,2,3,..) The boolean "bAnimated" can be used to specify, wether the view should be displayed immediately or in form of an animated movement from the current view to the specified view.

bPresent	**getMenuButtonPresent(strButtonID)**
	Shows active menu buttons, can be used to remove inactive buttons from the menu if you are using a custom skin. NOTE: Method only delivers correct results after the callback "onInitMenu" is called.

void	**pressMenuButton(strButtonID)**
	Sets the menu button to active.

void	**clickMenuButton(strButtonID)**
	Sets the menu button to active.

void	**releaseMenuButton(strButtonID)**
	Disables the active menu button.

void	**registerExternalMenuButton(elButton, strButtonID, strTip)**
	Registers an external menu button.

void	**unregisterExternalMenuButton(elButton)**
	Unregisters an external menu button.

void	**registerExternalMenuButtonsFromContainer(elContainer)**
	Registers an external menu button from a specific container.

void	**unregisterExternalMenuButtonsFromContainer(elContainer)**
	Unregisters an external menu button from a specific container.

strClassName	**getSkinClassName()**
	Returns the name of the current used skin class, e.g. fsi-skin-black

bFullscreen	**getInFullScreenMode()**
	Enables the FullScreen mode.

bSuccess	**setFullScreenElement(elFullScreenContainer)**
	Defines the FullScreen element.

nFrames	**getSpinFrames()**
	Returns the amount of spin frames.

bTransparent	**getImageIsTransparent()**
	Returns the info if the image is transparent.

oDimension	**getSrcDim()**
	Returns the dimensions of the source image.

void	**showMenuToolTip(strTip, elSrc, x, y)**
	Shows custom tooltips.. strTip: HTML code of thetooltip or ID in language.xml files elSrc: the DOM element wich is used to align the tooltip, usually event.target x,y: (optional) offset for tooltip position

void	**hideMenuToolTip()**
	Shows custom tooltips.. strTip: HTML code of thetooltip or ID in language.xml files elSrc: the DOM element wich is used to align the tooltip, usually event.target x,y: (optional) offset for tooltip position

oParameters	**getParameters();**
	Returns an object containing all parameters set via javascript or custom tag attributes.

bResult	**makeBoolean(value, bDefaultValue);**
	Transforms a value into a boolean value, e.g. "1" to true, "false" to false ***bDefaultValue*** is optional and false if not defined.

mixed	**getPluginParameter(strPluginName, strParameterName, defaultValue, bMakeBoolean);**
	Returns the parameter defined for the given plugin. ***defaultValue*** and ***bMakeBoolean*** are optional. example: iViewer.getPluginParameter("pages", "dir");

void	**testAPIListenersStart()**
	Starts the API test and shows all called listeners and the parameters which they return.

void	**testAPIListenersStop()**
	Stops the API test.

bSuccess	**startAutoSpin()**
	Starts the auto spin.

bSuccess	stopAutoSpin()
	Stops the auto spin.

2.9.2 Callbacks

You can define callback functions by assigning methods to the FSI Viewer instance.

- Using Callbacks
- Method A – Create a function in window scope
- Method B - Assignment by parameter
- Method C - Assignment by instance property

Using Callbacks

Method A – Create a function in window scope

Example:

```
<script>
   function onFSIViewerViewChanged(strView){
   console.log(strView);
}
<script>
```

a) assignment in Custom Tag:

Example:

```
<fsi-viewer onViewChanged="onFSIViewerViewChanged" [...]/>
```

or

b) assignment in XML configuration file:

Example:

```
<fsi_parameter>
   <onViewChanged value="onFSIViewerViewChanged"/>
</fsi_parameter>
```

Method B - Assignment by parameter

Example:

```
oParameters= {};
oParameters.onViewChanged = function(strView){
console.log(strView);
}

var myViewer = new $FSI.Viewer(elContainer, oParameters);
myViewer.start();
// all callback assignments must take place before starting
```

Method C - Assignment by instance property

Example:

```
var myViewer = new $FSI.Viewer(elContainer, oParameters);
myViewer.onViewChanged = function(strView){
console.log(strView);
}

myViewer.start();
// all callback assignments must take place before starting
```

⚠ Please note: You can modify or add new callbacks when calling changeConfig(). You cannot change, add or remove callbacks after the viewer started.

The following callback methods exist:

	onReady();
	Called as soon as FSI Viewer finished loading data and gets interactive.

	onConfigsReady(oConfigValues);
	Called as soon as FSI Viewer finished loading the configuration.

	onInit();
	Called as soon as FSI Viewer is initialized.

	onDestroy();
	Called when the fsi-viewer element is destroyed.

	onInitFailed(strReason);
	Called when the initialization is failed. This could happen if e.g. an image cannot be loaded or the dimensions of an image cannot be requested. The listener returns a string with the reason why the initialization failed.

	onMenuEnabled(bEnabled);
	Called if the menu is enabled/disabled.

	onMenuReady();
	Called as soon as the menu is ready.

	onMenuDestroy();
	Called if the menu is destroyed.

	onZoomChanged(fScale, fScaleMax);
	This optional callback function will be called each time the magnification level changes. ***fScale*** is the magnification level from 0 to 100. ***fScaleMax*** sets the maximum zoom level.

	onAnimationComplete();
	Called when a spin and/or zoom animation finishes.

	onAnimationStart();
	Called when a spin and/or zoom animation starts.

	onChangeConfig(strCfgFileName, oParameters);
	Called when the config file is changed.

	onChangeImage(oParameters, strView);
	Called when the image is changed.

	onResize(nWidth, nHeight, bResizeDone, BFullScreen);
	Called when the fsi-viewer element is resized.

	onSkinChanged(strCurrentSkinClass, strCurrentSkinClassBefore)
	This optional callback function can be used to call an action as soon as the skin class is changed.

	onRegisterExternalMenuButtons()
	This optional callback function can be used to call an action as soonexternal menu buttons are registered.

	onAfterRegisterExternalButton(elButton, strButtonID, bButtonPresentInViewer)
	This optional callback function can be used to remove inactive buttons from the menu bar if you are using a custom skin.

	onProgress(fPercent);
	This optional callback function can be called while the loading progress.

	onMenuButtonEnabled(strButtonID, null, bEnable);
	This optional callback function can be used call an action when a button is enabled or disabled.

	onMenuButtonPressed(strButtonID, evt);
	This optional callback function can be used to set an action that starts when a button is pressed.

	onMenuButtonReleased(strButtonID, evt);
	This optional callback function can be used to set an action when a certain button is released..

	onMenuButtonClicked(strButtonID, evt);
	This optional callback function can be used to set an action when a certain button is released..

	onMenuButtonSetPressed(bPressed);
	This optional callback function can be used to call an action when a button is pressed.

	onReset();
	This optional callback function can be used to call an action when the configuration is changed.

	onInitMenu();
	This optional callback function can be used to set an action when the menu is built (after all buttons are added).

	onMouseModeChanged(nMouseMode);
	This optional callback function can be used to call an action when the mouse mode is changed.

	onStartDragging(bInNavWindow);
	This optional callback function can be used to call an action when the user starts dragging the image.

	onStopDragging(bInNavWindow);
	This optional callback function can be used to call an action when the user stops dragging the image.

	onDataComplete();
	This optional callback function can be used to call an action while the configuration data loading is completed. Images/videos are not loaded yet at this point.

	onFullScreen(bFullScreenActive);
	This optional callback function can be used to call an action when FullScreen mode is enabled or disabled.

	onViewChanged(strViewString);
	This optional callback function can be used to call an action as soon as the viewed image section changes.

	onHotspotEvent(strAction, nHotspotID, strHotspotAttributeID, strContent)
	This optional callback function can be used to call an action as soon as a certain Hotspot event occurs. The following events are possible: mouseover, mousedown, mouseout & click.

3 FSI TouchZoom

FSI TouchZoom is an HTML5 and Javascript based tool adding image zoom to touch enabled devices like Apple iPad ™. When pinch zooming into a website containing single source images provided by FSI Server, the images will be displayed in the resolution matching the current magnification.

Supported devices

- iOS 4 or higher
- Android 4.x based devices
- Windows 8 with touch enabled browsers
- Mac OS with touch device

3.1 Usage

To use FSI TouchZoom, make sure the respective images on your website come from FSI Server and add the following script to the <head> section of your website:

Example:

```
<head>
<script src="viewer/applications/touchzoom/js/fsitouchzoom.js"
type="text/javascript">
</script>
</head>
```

If you would like to use global parameters for all images displayed with FSI TouchZoom, the corresponding parameters can be defined in the _default.xml.

Example:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <TouchZoom>
                <useDevicePixelRatio value="true" />
                <enableByCSSClass value="zoom-image" />
                <zoomPrecision value="3" />
        </TouchZoom>
...

<Options>
        <Skin value="black"/>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

It is possible to pass optional parameters via JavaScript. In this case it is crucial to embed the TouchZoom script before defining the parameters:

Example:

```
<script type="text/javascript
src="/viewer/applications/touchzoom/js/fsitouchzoom.js"></script>

<script type="text/javascript>
$FSI.touchZoomParameters =
{debug:true, useDevicePixelRatio:false};
```

```
</script>
```

Alternatively it is possible to use an external JavaScript for defining the parameters. This is useful if the parameters are valid for multiple HTML documents.

In case the script is automatically initialised (default), the automatically created instance can be adressed via

`$FSI.touchZoom`

3.2 Initializing FSI TouchZoom

Initializing or adding images manually

If you are adding images by script, you can decide to:

I) let FSI.TouchZoom initialize automatically and add image manually:

Example:

```
<script type="text/javascript
src="/viewer/applications/touchzoom/js/fsitouchzoom.js"></script>

<script type="text/javascript>
addMyImagesToTheDom(); // add your images here

// make FSI.TouchZoom look for new images
$FSI.touchZoom.scanForNewImages();
</script>
```

II) initialize FSI.TouchZoom manually after adding your images:

Example:

```
<script type="text/javascript
src="/viewer/applications/touchzoom/js/fsitouchzoom.js"></script>

<script type="text/javascript>
// prevent $FSI.TouchZoom from initializing automatically
// on document load
$FSI.touchZoomParameters = {autoInit:false};
function onBodyLoaded() {

  addMyImagesToTheDom(); // add your images here


  // create and initialize FSI.TouchZoom
  var parameters = {debug:true, useDevicePixelRatio:true};

  myTouchZoom = new $FSI.TouchZoom();
  myTouchZoom.init(parameters);

}
</script>
```

3.3 FSI TouchZoom Parameters

If you would like to use global parameters for all images displayed with FSI TouchZoom, the corresponding parameters can be defined in the _default.xml.

Example:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <TouchZoom>
                <useDevicePixelRatio value="true" />
                <enableByCSSClass value="zoom-image" />
                <zoomPrecision value="3" />
        </TouchZoom>
...

<Options>
        <Skin value="black"/>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

You can optionally modify the way FSI TouchZoom works by passing parameters to the script. To do so, pass an object containing the parameters and value as an object to the constructor:

Example:

```
<script type="text/javascript">
FSITouchZoom = new $FSI.TouchZoom({debug:true, zoomPrecision:6});
FSITouchZoom.init();
</script>
```

debug	
Description	displays debug & status information in the javascript console.
Syntax	Boolean
Default	false
Context	FSI TouchZoom

Display debug and status information in the browser's javascript console.

useDevicePixelRatio	
Description	Use the device's pixel ratio to display even sharper images.
Syntax	Boolean
Default	true
Context	FSI TouchZoom

Use the device's pixel ratio to display even sharper images. For devices with a pixel ratio >1 (e.g. Apple iPad 3) the images will be displayed in higher resolutions. If the aspect ratio is 2, the images loaded when pinching a page will be 2 x the resolution compared to devices with a pixel aspect ratio of 1.

monitorPositions	
Description	The script monitors the position of all single source <img> tags
Syntax	Boolean
Default	true

monitorPositions	
Context	FSI TouchZoom

By default the script monitors the position of all single source <img> tags to ensure that a higher resolution image will be loaded if the image is within the viewport. For huge amounts of images on a single web page you might want to disable this feature, if the positions of the images do not change (e.g. there is no javascript modifying the DOM tree).

useTiledImages	
Description	Zoomed images will be tiled into multiple image tiles
Syntax	Boolean
Default	true
Context	FSI TouchZoom

By default, zoomed images of 1000 pixel in width or height will be tiled into multiple image tiles. While this increases the user experience and the loading performance, you might want to disable this feature if you e.g. change the style of an image (e.g. style.display) at runtime. In this case the change would not effect the tiled image.

zoomPrecision	
Description	This parameter defines the precision of loading zoomed images.
Syntax	Integer
Default	4
Context	FSI TouchZoom

This parameter defines the precision of loading zoomed images.

Using the value "0" will load images exactly matching the current magnification. This does on the other hand prevent effective caching.

Values greater than 0 increase the precision how often the script loads new images when changing the magnification. The value 4 means that the script loads new images at 1.0, 1.25, 1.5, 1.75, 2.0 etc. magnification levels.

loadingImage	
Description	Specifies which image to display at the bottom right of the viewport
Syntax	Boolean or URL
Default	true
Context	FSI TouchZoom

Specifies whether or which image to display at the bottom right of the viewport while new images are being retrieved from FSI Server. You can use true or false to enable/disable the loading image display or provide an URL to a custom image.

imgSrcAttributeName	
Description	Specifies an alternative image source attribute
Syntax	string
Default	-
Context	FSI TouchZoom

Specifies an alternative image source attribute if required. This is useful for example if your would like to have SEO optimized image URLs in the "src" attribute, which would lead to FSI TouchZoom not recognizing the image. In this case, you can add an alternative image src attribute which contains the FSI Server single source image URL.

enableByCSSClass	
Description	provides a comma separated list of CSS class names
Syntax	String
Default	""
Context	FSI TouchZoom

Using the enableByCSSClass parameter you can provide a comma separated list of CSS class names. Only images containing at least one of the given class names will be modified by this script. All other images normal images that do not increase the resolution dynamically.

You can combine this parameter with the disableByCSSClass parameter.

disableByCSSClass	
Description	provides a comma separated list of CSS class names
Syntax	String
Default	""
Context	FSI TouchZoom

Using the disableByCSSClass parameter you can provide a comma separated list of CSS class names. Images containing at least one of the given class names will NOT be modified by this script. All other images containing single source image URLs will be modified.

You can combine this parameter with the enableByCSSClass parameter.

3.4 FSI TouchZoom JavaScript Interface

Usually you will not need to use the javascript interface.

Even if you modify an images src parameter at runtime, the tool will notice that and replace the magnified image accordingly.

If you on the other hand add images to the DOM tree at runtime or modify the position of images by script, the following interface is available to update the images and positions:

Initialization:

```
FSITouchZoom = new $FSI.TouchZoom();
FSITouchZoom.init();
```

3.4.1 Public Methods

void	**scanForNewImages();**
	Calling the scanForNewImages method will make the script look for single source images that have not been present in the DOM tree when the document finished loading. If you e.g. add <img> tags at runtime, you need to call this method in order to make the script aware of new image(s).

void	**addImage(elImg);**
	This adds a single <img> node to the list of dynamic touch zoom images. elImg: the <img> node object **Note:** you can as well call scanForNewImages() after adding an <img> node.

void	**removeImages(arImageDOMElements, bSilent);**
	Removes one or more touch zoom images and restores their original state. arImageDOMElements: an <img> node object or an array of <img> node objects

nImagesRemoved	**removeAllImages();**
	Removes all touch zoom images and restores their original state.

void	**updateImagePositions();**
	This makes the tool update all image positions and refresh images if needed. You might want to call this after modifying the DOM tree by script.

void	**printAPI();**
	Shows all API methods in the console in alphabetical order.

mixed	**getConfigValue(strName);**
	Returns the value of the configuration parameter "strName".

mixed	**traceConfigValue(strName);**
	Traces a config value and enables you to see where (in config files, the _default.xml, etc) a specific value is set.

void	**setZoomPrecision(fPrecision);**
	Use this method to modify the parameter zoomPrecision at runtime.

void	**destroy();**
	Destroys FSI TouchZoom.

void	**init();**
	Initializes FSI TouchZoom.

strVersion	**getVersion();**
	Returns a string containing the FSI ImageFlow software version.

strVersion	**getBuild();**
	Returns a string containing theFSI ImageFlow software build number.

strInstanceName	**getInstanceName();**
	Returns the name of the viewer object.

nInstanceID	**getInstanceID();**
	Returns the number of the viewer object.

void	**enable(bEnable);**
	Enables FSI TouchZoom.

nRegisteredImages	**getImageCount();**
	Returns the amount of registered images available for FSI TouchZoom.

void	**lockImage(elImage, bLock);**
	Locks a certain image .

void	**lockImageSourceDimension(elImage, bLock);**
	Locks a certain image to it's source dimensions.

void	**lockUpdates(bLock);**
	Locks updates .

void	**resetImage(elImg);**
	Resets a certain image.

void	**triggerRefresh();**
	Triggers a refresh.

bSuccess	**addListener(strListenerName, fn, iScope);**
	Adds a listener.

bSuccess	**removeListener(strListenerName, fn)**
	Removes a listener.

void	**testAPIListenersStart()**
	Starts the API test and shows all called listeners and the parameters which they return.

void	**testAPIListenersStop()**
	Stops the API test.

3.4.2 Callbacks

void	**onConfigsReady(oConfigValues);**
	Called when the configuration is ready.

void	**onDestroy();**
	Called when the instance is destroyed.

void	**onInit();**
	Called as soon as the viewer is initialized.

void	**onZoomChanged(fDocumentScale, fDocumentScalePixelRatio);**
	Called as soon as the zoom scope is changed.

4 FSI QuickZoom

FSI QuickZoom is an HTML5 and Javascript based tool adding image zoom to desktop and laptop computers on mouseover. When hovering over a single source image provided by FSI Server, the part of the image the cursor is currently pointing at will be displayed enlarged in a zoom window next to the original image.

Supported devices

- Desktop or laptop computers & all major browsers

4.1 Usage

To use FSI QuickZoom, make sure the respective images on your website come from FSI Server and add the following script to the <head> section of your website:

Example:

```
<head>
<script src="viewer/applications/quickzoom/js/fsiquickzoom.js"
type="text/javascript">
</script>
</head>
```

If you would like to use global parameters for all images displayed with FSI QuickZoom, the corresponding parameters can be defined in the _default.xml.

Example:

```
<fsi_parameter>
        <Image>
```

```
                <ServerType value="FSI"/>
        </Image>


        <QuickZoom>
                <useDevicePixelRatio value="true" />
                <enableByCSSClass value="zoom-image" />
                <zoomImageOffset value="10"/>
                <inPlaceZoom value="true"/>
                <moveRangeCenterCorners value="false"/>
        </QuickZoom>
...

<Options>
        <Skin value="black"/>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

It is possible to pass optional parameters via JavaScript. In this case it is crucial to embed the QuickZoom script before defining the parameters:

Example:

```
<script type="text/javascript
src="/viewer/applications/quickzoom/js/fsiquickzoom.js"></script>

<script type="text/javascript>
$FSI.quickZoomParameters =
{debug:true, useDevicePixelRatio:false};
</script>
```

Alternatively it is possible to use an external JavaScript for defining the parameters. This is useful if the parameters are valid for multiple HTML documents.

In case the script is automatically initialised (default), the automatically created instance can be adressed via

```
$FSI.quickZoom
```

4.2 How FSI QuickZoom works

FSI QuickZoom searches for all <img> tags in the HTML document containing FSI Server single source image URLs and adds event handlers to them.

When moving the mouse cursor over these images a magnified image will be displayed next to the original image. The user can move the cursor across the original image to display any image section of interest.

4.3 FSI QuickZoom Parameters

If you would like to use global parameters for all images displayed with FSI QuickZoom, the corresponding parameters can be defined in the _default.xml.

Example:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>
```

```
        <QuickZoom>
                <useDevicePixelRatio value="true" />
                <enableByCSSClass value="zoom-image" />
                <zoomImageOffset value="10"/>
                <inPlaceZoom value="true"/>
                <moveRangeCenterCorners value="false"/>
        </QuickZoom>
...

<Options>
        <Skin value="black"/>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

It is also possible to pass optional parameters via JavaScript. In this case it is crucial to embed the QuickZoom script before defining the parameters:

Example:

```
<script type="text/javascript
src="/viewer/applications/quickzoom/js/fsiquickzoom.js"></script>

<script type="text/javascript>
$FSI.quickZoomParameters = {debug:true, useDevicePixelRatio:false};
</script>
```

debug	
Description	displays debug & status information in the javascript console.
Syntax	Boolean
Default	false
Context	FSI QuickZoom

Display debug and status information in the browser's javascript console.

useDevicePixelRatio	
Description	Use the device's pixel ratio to display even sharper images.
Syntax	Boolean
Default	true
Context	FSI QuickZoom

Use the device's pixel ratio to display even sharper images. For devices with a pixel ratio >1 (e.g. Apple iPad 3) the images will be displayed in higher resolutions. If the aspect ratio is 2, the images loaded when pinching a page will be 2 x the resolution compared to devices with a pixel aspect ratio of 1.

headers	
Description	display debug and status information in the JavaScript console
Syntax	String
Default	"cors"
Context	FSI QuickZoom

The "&headers=" parameter to use for image requests.

imgSrcAttributeName	
Description	Specifies an alternative image source attribute
Syntax	string
Default	-
Context	FSI QuickZoom

Specifies an alternative image source attribute if required. This is useful for example if your would like to have SEO optimized image URLs in the "src" attribute, which would lead to FSI QuickZoom not recognizing the image. In this case, you can add an alternative image src attribute which contains the FSI Server single source image URL.

enableByCSSClass	
Description	provides a comma separated list of CSS class names
Syntax	String
Default	""
Context	FSI QuickZoom

Using the enableByCSSClass parameter you can provide a comma separated list of CSS class names. Only images containing at least one of the given class names will be modified by this script. All other images are displayed as normal images that do not use FSI QuickZoom.

You can combine this parameter with the disableByCSSClass parameter.

disableByCSSClass	
Description	provides a comma separated list of CSS class names
Syntax	String
Default	""
Context	FSI QuickZoom

Using the disableByCSSClass parameter you can provide a comma separated list of CSS class names. Images containing at least one of the given class names will NOT be modified by this script. All other images containing single source image URLs will be modified.

You can combine this parameter with the enableByCSSClass parameter.

addCSSClass	
Description	adds an additional CSS class
Syntax	String
Default	""
Context	FSI QuickZoom

Using the addCSSClass parameter you can add an additional CSS class. Only images containing the given class names will be modified by this script. All other images are displayed as normal images that do not use FSI QuickZoom.

You can combine this parameter with the disableByCSSClass parameter.

magnification	
Description	defines the magnification level
Syntax	Float
Default	""
Context	FSI QuickZoom

Defines the magnification level of the image section displayed in the zoom window.

It is also possible to define the magnification for each image tag individually by adding ***FSIQuickZoomMagnification***.

Example:

```
<img FSIQuickZoomMagnification="4" src= [...]>
```

inPlaceZoom	
Description	Shows zoom in place of the original image when hovered over
Syntax	Boolean
Default	false
Context	FSI QuickZoom

Show zoomed image instead of the original image when hovered over(true) or show zoom image next to original image (false).

autoZoomDimension	
Description	defines on which source the zoom image dimension depends on
Syntax	Boolean
Default	true
Context	FSI QuickZoom

Zoom dimension depends on source image size and screen dimension (true) or zoom dimension corresponds to original image dimension (false).

Setting “inPlaceZoom” to “true” overwrites this parameter to “false”.

moveRangeCenterCorners	
Description	defines if corners of the zoom image will be centered

moveRangeCenterCorners	
Syntax	Boolean
Default	true
Context	FSI QuickZoom

The zoomed image can be moved until a corner of the image is at the center of the zoom area (true), or the image can only be move until reaching the corner (false).

Setting “inPlaceZoom” to “true” overwrites the default value to “false”.

maxZoomWindowWidth	
Description	defines the maximum zoom window width
Syntax	Float
Default	-
Context	FSI QuickZoom

Defines the maximum width of the zoom window displayed next to the original image.

maxZoomWindowHeight	
Description	defines the maximum zoom window height
Syntax	Float
Default	-
Context	FSI QuickZoom

Defines the maximum height of the zoom window displayed next to the original image.

minZoomWindowWidth	
Description	defines the minimum zoom window width
Syntax	Float
Default	-
Context	FSI QuickZoom

Defines the minimum width of the zoom window displayed next to the original image.

minZoomWindowHeight	
Description	defines the minimum zoom window height
Syntax	Float
Default	-
Context	FSI QuickZoom

Defines the minimum height of the zoom window displayed next to the original image.

skipImageWidth	
Description	defines minimum image width for QuickZoom use
Syntax	Boolean
Default	-
Context	FSI QuickZoom

If the image width is smaller than the width defined with this parameter, FSI QuickZoom will not be applied to it.

skipImageHeight	
Description	defines minimum image height for QuickZoom use
Syntax	Boolean
Default	-
Context	FSI QuickZoom

If the image height is smaller than the height defined with this parameter, FSI QuickZoom will not be applied to it.

pageMargin	
Description	offset for the zoom window to each side of the page
Syntax	Float
Default	-
Context	FSI QuickZoom

Defines the offset for the zoom window to each side of the page.

zoomImageBorderWidth	
Description	defines the image border width
Syntax	Float
Default	-1
Context	FSI QuickZoom

Defines the image border width of the zoom window.

zoomImageOffset	
Description	defines the offset of the zoom window to the original image

zoomImageOffset	
Syntax	Float
Default	-1
Context	FSI QuickZoom

Defines the horizontal offset of the zoom window to the original image.

Defining box-shadow and fade-in

The box-shadow and the fade-in of the FSI QuickZoom zoom window can be defined via CSS.

Example:

```
div.FSIQuickZoom {
        border:1px solid #666;
        background-color: white;
        box-shadow: 3px 3px 3px rgba(0,0,0,0.25);
        animation-name: QuickZoomFadeIn;
        animation-duration: 0.5s;
        }

@keyframes QuickZoomFadeIn {
        from {
                        opacity: 0;
                }
        to {
                        opacity: 1;
                }
}
```

4.4 Initializing FSI QuickZoom manually

If you are adding images by script, you can decide to:

I) let FSI.QuickZoom initialize automatically and add image manually:

Example:

```
<script type="text/javascript
src="/viewer/applications/quickzoom/js/fsiquickzoom.js"></script>

<script type="text/javascript>

addMyImagesToTheDom(); // add your images here

// make FSI.QuickZoom look for new images

$FSI.quickZoom.scanForNewImages();

</script>
```

II) initialize FSI.QuickZoom manually after adding your images:

Example:

```
<script type="text/javascript
src="/viewer/applications/quickzoom/js/fsiquickzoom.js"></script>

<script type="text/javascript>

// prevent $FSI.QuickZoom from initializing automatically
// on document load
$FSI.quickZoomParameters = {autoInit:false};

function onBodyLoaded()

{
```

```
addMyImagesToTheDom(); // add your images here

// create and initialize FSI.QuickZoom
var parameters = {debug:true, useDevicePixelRatio:true};

myQuickZoom = new $FSI.QuickZoom();
myQuickZoom.init(parameters);

}

</script>
```

4.5 FSI QuickZoom JavaScript Interface

Usually you will not need to use the javascript interface.

Even if you modify an images src parameter at runtime, the tool will notice that and replace the magnified image accordingly.

If you on the other hand add images to the DOM tree at runtime or modify the position of images by script, the following interface is available to update the images and positions:

Initialization:

```
FSIQuickZoom = new $FSI.QuickZoom();
FSIQuickZoom.init();
```

4.5.1 Public Methods

void	**init();**
	Initializes FSI QuickZoom.

void	**printAPI();**
	Shows all API methods in the console in alphabetical order.

void	**destroy();**
	Destroys FSI QuickZoom.

void	**addImage(elImg);**
	This adds a single <img> node to the list of dynamic touch zoom images. elImg: the <img> node object **Note:** you can as well call scanForNewImages() after adding an <img> node.

mixed	**getConfigValue(strName);**
	Returns the value of the configuration parameter "strName".

mixed	**traceConfigValue(strName);**
	Traces a config value and enables you to see where (in config files, the _default.xml, etc) a specific value is set.

nInstanceID	**getInstanceID();**
	Returns the number of the viewer object.

strInstanceName	**getInstanceName();**
	Returns the name of the viewer object.

strVersion	**getVersion();**
	Returns a string containing the FSI QuickZoomsoftware version.

strVersion	**getBuild();**
	Returns a string containing theFSI QuickZoom software build number.

void	**removeImages(arImageDOMElements)**
	Removes one or more quickzoom images and restores their original state.

void	**scanForNewImages();**
	Calling the scanForNewImages method will make the script look for single source images that have not been present in the DOM tree when the document finished loading. If you e.g. add <img> tags at runtime, you need to call this method in order to make the script aware of new image(s).

bSuccess	**addListener(strListenerName, fn, iScope);**
	Adds a listener.

bSuccess	**removeListener(strListenerName, fn)**
	Removes a listener.

void	**testAPIListenersStart()**
	Starts the API test and shows all called listeners and the parameters which they return.

void	**testAPIListenersStop()**
	Stops the API test.

4.5.2 Callbacks

void	**onConfigsReady(oConfigValues);**
	Called when the configuration is ready.

void	**onDestroy();**
	Called when the instance is destroyed.

void	**onInit();**
	Called as soon as the viewer is initialized.

void	**onStart(nImagesPrepared, arImageDOMElements);**
	Called as soon as the viewer is initialized.

void	**onZoomEnd();**
	Called as soon as the zooming ends.

void	**onZoomStart(elImage, strZoomImage);**
	Called as soon as the zoom is started.

void	onModifyPositionAndSize(oModify, oInfo);
	Returns the position and size.

Example:

```
function setQuickZoomPos(oModify,oInfo){

oModify.left = 0;
// relative to the left of the image's clientRect
oModify.top = 0;
// relative to the top of the image's clientRect

oModify.width = 200;
oModify.height = 200;
oModify.srcZoomImage += "&effects=Sepia()";

// oInfo contains various information you can read
// modifying oInfo does not change anything
// e.g.
// oInfo.elementPosition
(clientPosition of the original
image tag)
// oInfo.imgPaddings (margin and border of the image)

}
```

5 FSI ThumbBar

FSI ThumbBar is a Javascript based image thumb bar with optional image zoom.

Supported devices test

- iOS 4 or higher
- Android 4.x based devices
- Windows 8 with touch enabled browsers
- Mac OS with touch device

5.1 Usage

I) Add the fsithumbbar.js script to the head of your document:

Example:

```
<head>
<script src="viewer/applications/thumbbar/js/fsithumbbar.js" type="text/
javascript">
</script>
</head>
```

II) Add a<fsi-thumbbar> tag to the part of your document body where FSI ThumbBar should be displayed. Add the desired dimension to it and image source (directory) to it:

Example:

```
<body>
<fsi-thumbbar width="400" height="150" dir="/images/foo/" >
</fsi-thumbbar>
```

```
</body>
```

Parameters are added to the <fsi-thumbbar> tag. Note that you can as well use configuration files:

Example:

```
<body>

//Adding parameters directly to the tag:
<fsi-thumbbar width="100%" height="150" src="/images/foo/"
backgroundcolor="#000" presentationtype="flat">
</fsi-thumbbar>

//Example with using a FSI configuration file:
<fsi-thumbbar width="100%" height="150" dir="/images/foo/" cfg="sample/
sample_configuration>
</fsi-thumbbar>

</body>
```

You can define individual configuration .xml files like this:

Simple Configuration File

```
<fsi_parameter>
  <image>
            <path value="samples/Watch.jpg" />
  </image>
  <ThumbBar>
```

```
            <presentationtype value="stacks" />
            <debug value="1" />
                <enablezoom value="false" />
                <preloadcount value="8" />
  </ThumbBar>
</fsi_parameter>
```

If you would like to use global parameters for all FSI ThumbBar instances, the corresponding parameters can be defined in the _default.xml.

Example _default.xml:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <ThumbBar>
             <presentationtype value="stacks" />
             <debug value="1" />
                 <enablezoom value="false" />
                 <preloadcount value="8" />
   </ThumbBar>
...

<Options>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

5.2 How FSI ThumbBar works

The script retrieves an image list from FSI Server and display the images in form of an image thumbbar in the given <fsi-thumbbar> tag.

Note on Internet Explorer:

In order to make FSI ThumbBar work, Internet Explorer must run in IE8 quirks, IE9 or higher quirks or standards mode. You can ensure that by adding an X-UA-Compatible meta tag to the head section of your web page, for example:

```
<meta http-equiv="X-UA-Compatible"
content="IE=Edge,chrome=1">
```

5.3 FSI ThumbBar Parameters

Parameters are added to the <fsi-thumbbar> tag. Note that you can as well use configuration files:

Example:

```
<body>

//Adding parameters directly to the tag:
<fsi-thumbbar width="100%" height="150" src="/images/foo/"
 backgroundcolor="#000" presentationtype="flat">
</fsi-thumbbar>

//Example with using a FSI configuration file:
<fsi-thumbbar width="100%" height="150" dir="/images/foo/"
 cfg="sample/sample_configuration>
</fsi-thumbbar>

</body>
```

You can define individual configuration .xml files like this:

Simple Configuration File

```
<fsi_parameter>
  <image>
            <path value="samples/Watch.jpg" />
  </image>
  <ThumbBar>
            <presentationtype value="stacks" />
            <debug value="1" />
                <enablezoom value="false" />
                <preloadcount value="8" />
  </ThumbBar>
</fsi_parameter>
```

If you would like to use global parameters for all FSI ThumbBar instances, the corresponding parameters can be defined in the _default.xml.

Example _default.xml:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <ThumbBar>
             <presentationtype value="stacks" />
             <debug value="1" />
                 <enablezoom value="false" />
                 <preloadcount value="8" />
```

```
    </ThumbBar>
...

<Options>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

You can optionally modify the way FSI ThumbBar works by passing parameters to the script. You can either use an XML configuration passing the path using a "cfg" parameter or pass parameters and values as an object to the constructor:

Example:

```
<script type="text/javascript">

thumb = new $FSI.ThumbBar({debug:true,
backgroundColor:"#ff0"});

thumb.init();

</script>
```

debug	
Description	displays debug & status information in the javascript console.
Syntax	Boolean
Default	false

debug	
Context	FSI ThumbBar

Display debug and status information in the browser's JavaScript console.

headers	
Description	header parameter to use for image requests
Syntax	String
Default	"cors"
Context	FSI ThumbBar

The "&headers=" parameter to use for image requests.

The default value "cors" ensures cors compatibility for cross-origin scenarios.

dir	
Description	The path to the source image folder on FSI Server to display.
Syntax	string
Default	""
Context	FSI ThumbBar

The path to the source image folder on FSI Server to display.

renderer	
Description	The renderer parameter to use for image requests (e.g. jpeg).
Syntax	string
Default	""
Context	FSI ThumbBar

The renderer parameter to use for image requests (e.g. jpeg).

format	
Description	The format parameter to use for image requests (e.g. png).
Syntax	string
Default	""
Context	FSI ThumbBar

The format parameter to use for image requests (e.g. png).

overlays	
Description	The overlays parameter to use for image requests.
Syntax	string
Default	""
Context	FSI ThumbBar

The overlays parameter to use for image requests.

quality	
Description	JPEG quality level when retrieving JPEG images from the server.
Syntax	Integer
Default	-
Context	FSI ThumbBar

The JPEG quality level when retrieving JPEG images from the server.

effects	
Description	Optional FSI Server effects to use for all images.
Syntax	String
Default	""
Context	FSI ThumbBar

Optional FSI Server effects to use for all images.

Example: effects:"Sepia()"

cropValues	
Description	Defines the amount that should be cut off the image.
Syntax	String
Default	""

cropValues	
Context	FSI ThumbBar

Defines the amount that should be cut off the image from each side.

Values can be provided as pixel (e.g. "20px") or percent (e.g. "10%") or floating point numbers (e.g. "0.25").

Use 4 values for left, top, right, bottom.

Example: 10%,0,10%,0

cropRect	
Description	Defines a crop rectangle
Syntax	String
Default	"0,0,1,1"
Context	FSI ThumbBar

Defines the amount that should be cut off the image from each side.

Values can be provided as pixel (e.g. "20px") or percent (e.g. "10%") or floating point numbers (e.g. "0.25").

Use 4 values for left, top, right, bottom.

Note

You cannot use "CropRect" AND "CropValues" parameters at a time.

If you use both, "CropRect" will be ignored.

elementWidth	
Description	The maximum width of images in the slideshow.
Syntax	String
Default	10 (px)
Context	FSI ThumbBar

The maximum width of images in the slideshow. The value can be provided in pixels or in percent of the instance width.

elementSpacing	
Description	Spacing between the images in pixel.
Syntax	String
Default	0.25 (%)
Context	FSI ThumbBar

Spacing between the images in pixel. The spacing needs to be either stated in pixels or in percent of the instance width.

Note: If the resulting image height limits the image dimension, there is a spacing even if you define elementSpacing with a value of "0". In this case you either need to:

a) Increase the container element height

b) Decrease the value of the elementWidth parameter

paddingTop	
Description	Top padding in pixel.
Syntax	Integer
Default	10
Context	FSI ThumbBar

Top padding in pixel.

paddingBottom	
Description	Bottom padding in pixel.
Syntax	Integer
Default	0
Context	FSI ThumbBar

Use numbers greater or smaller than 0 to make the slide show rotate on start with the Bottom padding in pixel.

autoResize	
Description	defines if instance is resized when the DOM element is resized
Syntax	Boolean
Default	true
Context	FSI ThumbBar

Defines if the instance is resized when the DOM element is resized.

initialImage	
Description	defines the image index to focus on start
Syntax	Integer
Default	0
Context	FSI ThumbBar

Defines the image index to focus on start.

vertical	
Description	defines if the thumb bar is displayed horizontally or vertically
Syntax	Boolean
Default	false
Context	FSI ThumbBar

Defines if the thumb bar is displayed horizontally (default) or vertically.

presentationType	
Description	The overall image presentation appearance.
Syntax	String
Default	"flat"
Context	FSI ThumbBar

Selects the overall image presentation appearance in FSI ThumbBar. Available types:

- "flat"
- "stacks"
- "outerRing"
- "innerRing"
- "custom" (see chapter "Using custom presentation type")

enableZoom	
Description	Enable or disable zooming the focused image with the built-in image zoom.
Syntax	Boolean
Default	true
Context	FSI ThumbBar

Enable or disable zooming the focused image with the built-in image zoom.

When zooming an image, FSI TumbBar adds a element to the document body, holding the zoomed image.

useTouchZoom	
Description	Uses FSI TouchZoom on thumbnail images.
Syntax	Boolean
Default	true
Context	FSI ThumbBar

Uses FSI TouchZoom on thumbnail images and requires adding the FSI TouchZoom script to the document.

ScrollBar	
Description	enable or disable the scroll bar.
Syntax	Boolean
Default	true
Context	FSI ThumbBar

This value enables or disables the scroll bar.

centerZoomedImage	
Description	scrolls to the selected (zoomed) image
Syntax	Boolean
Default	true
Context	FSI ThumbBar

Moves the currently zoomed image to the center of the bar.

Alignment	
Description	Alignment of the thumbnails (vertical alignment if horizontal thumb bar and vice versa).
Syntax	Float
Default	1.0
Context	FSI ThumbBar

Alignment of the thumbnails (vertical alignment if horizontal thumb bar and vice versa):

- 1.0: align at bottom (or right)
- 0.5: center images
- 0.0: align at top (left)
- ...or any value inbetween

endlessScrolling	
Description	defines if endless scroll is activated
Syntax	Boolean
Default	false
Context	FSI ThumbBar

Defines if the endless scroll is activated and duplicates images if required.

preloadCount	
Description	The number of images to pre-load.
Syntax	Integer
Default	40
Context	FSI ThumbBar

The number of images to pre-load although they are currently not visible.

Disabling preload (set value to 0) will only load images visible on stage.

autoRotateSpeed	
Description	Auto scroll on start.
Syntax	(Float)
Default	0
Context	FSI ThumbBar

Auto scroll on start; use positive or negative values to specify the direction. Auto-rotation stops as soon as the user interacts with the bar.

depthFadeout	
Description	Darkens images that are further away in z-direction.
Syntax	Boolean
Default	true
Context	FSI ThumbBar

Darkens images that are further away in z-direction (for 3D presentation types).

zoomShaderStartOpacity	
Description	The start opacity of the shader when zooming an image.
Syntax	Float
Default	0
Context	FSI ThumbBar

The start opacity (0 to 100) of the shader when zooming an image. See CSS Classes for more options like color.

zoomShaderEndOpacity	
Description	The end opacity of the shader when zooming an image.
Syntax	Float
Default	0
Context	FSI ThumbBar

The end opacity (0 to 100) of the shader when zooming an image. See CSS Classes for more options like color.

autoDestroy	
Description	automatically destroy instances
Syntax	Boolean
Default	true
Context	FSI ThumbBar

Automatically destroy instances created with the fsi-thumbbar tag upon removing the tag from the document DOM.

viewerSelector	
Description	alias for viewerID
Syntax	String
Default	
Context	FSI ThumbBar

Defines a selector (e.g. "#myViewer" or "fsi-viewer" or ".myClass") of an FSI Viewer instance to be used to display the selected image. You may use "auto" to use the first <fsi-viewer> tag in the document (if there is only one).

cmdButtonSelector	
Description	a selector to one or more DOM elements that should be used as control buttons
Syntax	String
Default	
Context	FSI ThumbBar

Defines a selector to one or more DOM elements that should be used as control buttons.

The matched DOM elements need to contain the attribute "x-fsi-cmd" wth one of the following values:
firstImage, previousPage, previousImage, nextImage, nextPage, lastPage

See "Adding control buttons" for more information.

viewerSelector	
Description	alias for viewerID
Syntax	String
Default	
Context	FSI ThumbBar

Defines a selector (e.g. "#myViewer" or "fsi-viewer" or ".myClass") of an FSI Viewer instance to be used to display the selected image. You may use "auto" to use the first <fsi-viewer> tag in the document (if there is only one).

PerspectiveAlignment	
Description	Defines the perspective origin for 3D presentation types ("innerRing", "outerRing", "custom").
Syntax	Float
Default	1.0
Context	FSI ThumbBar

Defines the perspective origin for 3D presentation types ("innerRing", "outerRing", "custom").

- 1.0: perspective origin at bottom (or right)
- 0.5: perspective origin at center

- 0.0: perspective origin at top (left)
- ...or any other value

placeHolderImage	
Description	Defines the CSS Class and type of the placeholders while images are loading.
Syntax	String
Default	"default"
Context	FSI ThumbBar

Defines the CSS Class and type of the placeholders while images are loading.

- none" (CSS class: "none") - no placeholder at all
- "blank" (CSS class: "blank") - a blank white placeholder
- "default" (CSS class: "default") - the default placeholder with a background image
- "custom" (CSS class: "custom") - a custom placeholder class (add the CSS class as described below)
- [path to an image on FSI Server] (CSS class: "image") - a custom image with a pad effect (see parameter placeHolderPadEffect)

To customize the CSS for the placeholder, please use the following CSS rule:

```
div.fsi-thumbbar-root div.fsi-thumbbar-container
img.placeholder.image
```

(replace ".image" with the CSS class listed above).

placeHolderPadEffect	
Description	Defines the pad effect to use when using a custom placeholder image.
Syntax	String

placeHolderPadEffect	
Default	Pad(CC,FFFFFF)
Context	FSI ThumbBar

Defines the pad effect to use when using a custom placeholder image (see parameter placeHolderImage).

garbageCollectionSize	
Description	Specifies how many images will be kept in memory although they ar not visible anymore.
Syntax	Integer
Default	-1
Context	FSI ThumbBar

Specifies how many images will be kept in memory although they ar not visible anymore.
Use -1 for auto sizing the garbage collection size.
Use 0 to disable garbage collections (all images will be kept in memory, not recommended for huge collections)

Any other positive number:
At least the specified number of images will be kept in memory.
Can be more images if required by instance size and preloadCount parameter.

5.4 Customization

5.4.1 CSS Classes

The following CSS classes can be used to modify the appearance:

- fsi-thumbbar - The entire thumbbar instance.
- div.fsi-thumbbar-shader - The shader below a zoomed image.
- div.fsi-thumbbar-zoom-div -The div containing the zoomed image.
- div.fsi-thumbbar-container img.placeholder - The place holder shown when a thumbnail loads. See parameter "placeHolderImage" for more details.

5.4.2 Adding Control Buttons

There are two ways to add control buttons:

- using the registerButton(element, command) API method
- using the cmdButtonSelector parameter and HTML attributes

The following commands are available:

- firstImage
- previousPage
- previousImage
- nextImage
- nextPage
- lastPage

Example:

```
<div id="myThumbBarControls">
<input type="button" x-fsi-cmd="previousImage" value="previous" />
<input type="button" x-fsi-cmd="nextImage" value="next"/>
</div>


<fsi-thumbbar
cmdButtonSelector="#myThumbBarControls > input"
width="800"
height="400"
dir="images/someFolder"></fsi-thumbbar>
```

Please note:

You can use any HTML element, not just buttons. The CSS class "fsi-pressed" will be added to the DOM element, while the button is pressed.

5.4.3 Defining a custom presentation type

You can create a custom presentation type in addition to the built-in types (see "presentationType").
To use a custom presentation you need to set "presentationType" to "custom" and define 2 functions.

The function defined by "CustomPresentationInitFunction" will be called on start and on resize.
The function defined by "CustomPresentationFunction" will be called each time the presentation needs to be drawn.
You therefore need to make sure that this function takes very little time to complete, otherwise the presentation will be laggy.

Example:

```
<presentationType value="custom" />
<customPresentationInitFunction value="onThumbBarPresentationInit" />
<customPresentationFunction value="onThumbBarPresentationCalc" />
```

Plus the script:

Example:

```
function onThumbBarPresentationInit(oAnimation){
// oAnimation contains information on the presentation
// You can add your own properties
// oAnimation will be passed to your custom calculation function
console.log("init custom FSI ThumbBar presentation");
console.dir(oAnimation);
oAnimation.myLeftLimit = -oAnimation.totalWidth /2 - 2 *
oAnimation.nMaxThumbWidth;
oAnimation.myRightLimit = oAnimation.totalWidth /2 + 2 *
oAnimation.nMaxThumbWidth;
oAnimation.myZDistance = oAnimation.nPerspective /8;
// nHorizontalFactor is 1 for horizontal thumb bars and -1 for vertical
thumb bars
oAnimation.myRotationFactor = oAnimation.nHorizontalFactor * 2;
}


function onThumbBarPresentationCalc(oAnimation, oItem){
var ret = 0;
if (oItem.position > oAnimation.myLeftLimit && oItem.position <
oAnimation.myRightLimit){
ret = 1;
// oItem.position is 0 for the centered image, negative for images left
of the center
// delta is -1 for images at the left margin, 0 at center and 1 at the
right margin
var delta = oItem.position / oAnimation.totalWidth;
var piDelta = 2 * delta * Math.PI;
var zRel = Math.cos(piDelta);
oItem.ry = Math.sin(piDelta)/oAnimation.myRotationFactor;

oItem.z = zRel * oAnimation.myZDistance;
oItem.brightness = 1 + (zRel -1) / 6;

// set to true only, if items overlap (items will be z-sorted, which is
expensive)
```

```
// oItem.bSetZIndex = true;
}
// return 1 to indicate that item is on stage, 0 to indicate it's not
visible
return ret;
}
```

5.5 ThumbBar JavaScript Interface

Initialization:

```
FSIThumbBar = new $FSI.FSIThumbBar ();
FSIThumbBar.init(dir:"images/foo");
```

5.5.1 Public Methods

void	**init(elementID, oParameters, bDebug);**
	Initializes the FSI ThumbBar instance with the given parameters.

void	**printAPI();**
	Shows all API methods in the console in alphabetical order.

void	**destroy();**
	Destroys the FSI ThumbBar instance leaving the empty container only. You can re-use the same container calling FSIThumbBar.init()

bInitDone	**getInitDone();**
	Returns if the initialization process is done.

nInstanceID	**getInstanceID();**
	Returns the number of the viewer object.

strVersion	**getVersion();**
	Returns a string containing the FSI ThumbBar software version.

strBuild	**getBuild();**
	Returns a string containing the FSI ThumbBar software build number.

void	**assignFSIViewer(iFSIViewerInstance);**
	Assigns a certain FSI Viewer JS instance to FSI ThumbBar for building FSI Showcase JS.

void	**focusImage(nImage);**
	Scrolls the bar until the image with the given index is in the foreground.

void	**zoomImage(nImage);**
	Scrolls the bar until the image with the given index is in the foreground and zoom the given image afterwards.

void	**closeZoom(evt, bImmediate);**
	Remove the zoomed image if any.

bSuccess	**addListener(strListenerName, fn, iScope);**
	Calls a specific listener.

bSuccess	**removeListener(strListenerName, fn);**
	Removes a specific listener.

strInstanceName	**getInstanceName();**
	Returns the name of the viewer object.

void	testAPIListenersStart()
	Starts the API test and shows all called listeners and the parameters which they return.

void	testAPIListenersStop()
	Stops the API test.

bSuccess	buttonDown(strCommand)
	Presses the down button.

bSuccess	buttonUp(strCommand)
	Presses the up button.

bSuccess	changeConfig(strCfgFileName, oParameters)
	Reset the viewer object and change the configuration to the given configuration file. strCfgFileName (optional): path to the configuration file (see "cfg" parameter) oParameters (optional): an object containing parameters If present, these parameters will overwrite the parameters defined upon initialization.

bSuccess	changeDir(strDirectory)
	Reset the viewer object changes the directory to the given directory.

void	firstImage(nAnimationFrames)
	Selects the first image of the range.

mixed	getConfigValue(strName)
	Returns the set config value for the given parameters.

nImages	getImageCount()
	Returns the amount of images used in FSI ThumbBar.

fPosition	getScrollPos()
	Returns the current scroll position.

bScrolling	isScrolling()
	Returns if FSI ThumbBar is currently scrolling.

void	**lastImage(nAnimationFrames)**
	Selects the last image of the range.

void	**nextImage(nAnimationFrames)**
	Selects the next image of the range.

void	**nextPage(nAnimationFrames)**
	Selects the next pages of the range.

void	**previousImage(nAnimationFrames)**
	Selects the previous image of the range.

void	**previousPage(nAnimationFrames)**
	Selects the previous page of the range.

bSuccess	**registerButton(DOMElement, strCommand)**
	Registers a button with the given specifications. (See Adding Control Buttons.)

fPosition	setScrollPos(fPosition, nAnimationFrames)
	Sets the scroll position based on the given specifications

void	startAutoRotate(fSpeed)
	Starts the automatic rotation.

void	stopAutoRotate()
	Stops the automatic rotation.

void	traceConfigValue(strName)
	Traces a config value and enables you to see where (in config files, the _default.xml, etc) a specific value is set.

bSuccess	unRegisterButton(DOMElement)
	Unregisters a button. (See Adding Control Buttons.)

5.5.2 Callback functions

The optional JavaScript callback functions enables developers to react on events fired by FSI ThumbBar. To use a callback you need to implement a corresponding JavaScript function AND pass the name of the callback function to the FSI ThumbBarinstance, e.g.:

Example:

```
function onStart(strDirectory, nImages, arImagePaths){
        console.log("The FSI ThumbBar instance with the id " + idElement
        + " displays a total of " + nImages + " images".}


FSIThumbBar = new $FSI.ThumbBar();

FSIThumbBar.init(dir:"images/foo", callBackStart:onStart);
```

Note:
You can either pass the function or the function name in window scope as a string.

The following callback functions can be used:

	onStart(strDirectory, nImages, arImagePaths);
	Called once after loading the image list.

	onClick(nImageIndex, strImagePath);
	Called each time the user clicks an image.

	onFocus(nImageIndex, strImagePath);
	Called each time the foreground image changes.

	onZoomReady(nImageIndex,strImageURL,elZoomedImage);
	Called as soon as the zoom is ready.

	onZoomStart(nImageIndex, strImageURL);
	Called each time a zoomed image starts loading.

	onZoomEnd();
	Called each time a zoomed image has been closed.

	onDestroy();
	Called as soon as the viewer is destroyed.

	onInit(oParameters);
	Called as soon as the viewer is initialized.

	onConfigsReady(oConfigValues);
	Called as soon as the config is ready.

	onMouseEnter(nIndex, oItem);
	Called as soon the mouse enters the defined area.

	onMouseLeave(nIndex, oItem);
	Called as soon the mouse leaves the defined area.

	onScrollStart(fPosition);
	Called as soon as the scrolling is started.

	onScrollEnd(fPosition);
	Called as soon as the scrolling is ended.

	onSetScrollRange(fTotalSize, fRange, bEndlessScrolling, nTotalImages);
	Called as soon as the defined scroll range is set.

6 FSI Showcase

FSI Showcase is a combination of the HTML5 and Javascript based viewers FSI Viewer and FSI ThumbBar, for displaying multiple images in a scrollable area with a FSI Viewer instance above.

Supported devices

- iOS 4 or higher
- Android 4.x based devices
- Windows Phone 8 and Windows RT devices
- Windows 8 with touch enabled browsers
- Mouse zoom: desktop or laptop computers &all major browsers

> **Note on Parameters**
>
> As FSI Showcase is build from both FSI Viewer and FSI ThumbBar, the related parameters are not re-listed in this chapter.
>
> You can find the parameters in FSI Viewer Parameters and FSI ThumbBar Parameters.

6.1 Usage

To use FSI Showcase, make sure the respective images on your website come from FSI Server and add the following scripts to the <head> section of your website:

Example:

```
<head>
```

```
<script src="applications/viewer/js/fsiviewer.js" type="text/
javascript"></script>
<script src="applications/thumbbar/js/fsithumbbar.js" type="text/
javascript"></script>
</head>
```

FSI Showcase is build of combining the FSI Viewer and FSI ThumbBar tags, assigning the Viewer to ImageFlow with a specific viewer id:

Example:

```
<body>


//Adding FSI Viewer
<fsi-viewer id="viewer1" width="100%" height="600" noNav="1">
</fsi-viewer>

//Adding FSI ThumbBar, assigning the Viewer with the viewerID
<fsi-thumbbar viewerID="viewer1" width="100%" height="150"
  dir="/images/foo/" presentationType="flat">
</fsi-thumbbar>

</body>
```

If you want to add the fullscreen plugin to FSI Showcase, you need to wrap the Viewer and ThumbBar into a container DIV to reserve the proper space for the ThumbBar Element in fullscreen mode, like:

Example:

```
<body>
//Adding Container
```

```
<div id="Container_FSI_123" // Used for fullscreen element parameter of
FSI Viewer
  style="position:relative; width:500px; height:500px; box-
sizing:border-box;
  padding-bottom:150px;">  // Height of the Imageflow!
  //Adding FSI Viewer
  <fsi-viewer id="viewer1" width="100%" height="600" noNav="1"
plugins="fullScreen"
    fullscreenelement="Container_FSI_123">
  </fsi-viewer>
  //Adding FSI ThumbBar , assigning the Viewer with the viewerID
  <fsi-thumbbar viewerID="viewer1" width="100%" height="150" dir="/
images/foo/"
    presentationType="flat">
  </fsi-thumbbar>

</div>
</body>
```

> **Note on HTML Codes**
>
> The HTML Code generated in the FSI Server interface will change accordingly to the plugins selected.

7 FSI Pages

7.1 Introduction

FSI Pages is a JavaScript based tool for presenting image collections as an interactive catalog, booklet or photo album on desktop computers and mobile devices. FSI Pages automatically creates a page index with thumbnails and supports hyperlinks and interactive zooming.



7.2 How FSI Pages works

The script uses the given <fsi-pages> tag, which is similar to an <img> tag, to display an interactive catalog.

In the tag, a directory is defined which will contain all images displayed. It is possible to add tooltips and hyperlinks to the pages for displaying product descriptions or to link to an online shop. A chapters plugin enables users to guide through large catalogs easily.

The user can drag the pages left and right to browse through the catalog and zoom in/out either using pinch zoom or by double-clicking on the page.

Hovering over embedded areas displays a tooltip and/or hyperlink if defined.

7.3 Usage of FSI Pages

I) Add the fsipages.js script to the head of your document:

Adding the script:

```
<head>
//adding the fsipages.js script
<script src="viewer/applications/pages/js/fsipages.js"
type="text/javascript"></script>
</head>
```

II) Add a <fsi-pages> tag to the part of your document body where the catalog should be displayed. Add the desired dimension and the image directory to it:

Integrate FSI Pages JS:

```
<body>
<fsi-pages width="100%" height="100%" dir="/images/catalog/">
//here you can place the fallback content (or image) in case the browser
```

```
// cannot display the viewer, e,g,:
<b>Sorry</b>, your browser is not capable of running FSI Pages
</fsi-pages>
</body>
```

Parameters are added to the <fsi-pages> tag. Note that you can as well use FSI Pages configuration files:

Adding Parameters to FSI Pages JS:

```
<head>
//adding the fsipages.js script
<script src="viewer/applications/pages/js/fsipages.js"
type="text/javascript"></script>
</head>

<body>

//Adding parameters directly to the tag:
<fsi-pages width="500" height="600" dir="/images/catalog/" debug="true"
skin="silver" PageLayout="flip">
</fsi-pages>

//Example with using a FSI Pages configuration file:
<fsi-pages width="500" height="600" dir="/images/catalog/" cfg="sample/
catalog.xml">
</fsi-pages>

</body>
```

You can define individual configuration .xml files like this:

Simple Configuration File

```
<fsi_parameter>
  <image>
     <path value="samples/Watch.jpg" />
  </image>
  <Pages>
     <skin value="silver" />
     <debug value="1" />
         <pagelayout value="flip" />
         <thumbsize value="100" />
         <bendeffect value="none" />
  </Pages>
</fsi_parameter>
```

If you would like to use global parameters for all FSI Pages instances, the corresponding parameters can be defined in the _default.xml.

Example _default.xml:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <Pages>
                <skin value="silver" />
            <debug value="1" />
                 <pagelayout value="flip" />
                 <thumbsize value="100" />
                 <bendeffect value="none" />
        </Pages>
...
```

```
<Options>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

7.3.1 Defining Image Collections

The images to be displayed in FSI Pages have to be defined in an <images> node added to your XML configuration file.

There are two different ways of defining image lists:

Method 1: Retrieving automatically generated image lists from FSI Server

The most convenient way to create image collections is using the template system of FSI Server.
You can do so by using the "Publish to Web" option in FSI Server web interface.

Retrieving Image Collections from Imaging Server:

```
<Pages>
<ListTemplate value="catalog_list" />
```

```
<dir value="images/catalogs/foo" />
</Pages>
```

This way FSI Server uses the specified ListTemplate to return the matching images in XML format. Depending on the list template, FSI Server serves IPTC data belonging to an image (e.g. for links) as well. (catalog_list is the default ListTemplate)

Method 2: Creating collections manually using XML configuration files

Creating collections manually using XML configuration files:

```
<Images>
...
<Image file="image1.xml"
label="My First Image" />
<Image file="image2.xml"
label="My Second Image" />
...
</Images>
```

Method 1 retrieves a complete image list ignoring previously defined <images> nodes.

Image Order

The order of the images (this is the page order) corresponds to the order of the <image> declarations within the <images> section.

Inserting Blank Pages

You can insert blank pages by adding <image empty="true" /> nodes anywhere in your image collection. Alternatively you can use the FSI Pages parameter to pass the page numbers to insert blank pages before seperated by commas. This is especially useful if you retrieve the image collection from FSI Server (Method 1).

Removing certain Images from a List

Using the parameter removePages you can disable (remove) certain images from an image list. In this case you pass the page number(s) of the pages you want to remove separated by commas.

7.3.2 Aspect Ratio of the Pages

By default FSI Pages uses the aspect ratio of the first image in the image collection to determine the aspect ratio for all pages.

Alternatively you can manually define an aspect ratio for all pages using the "ratio" parameter.

Examples:

```
ratio="1:2" or ratio="640:480"
```

Note on Aspect Ratio

FSI Pages assumes all images in the given folder will have the same dimensions, thus the first image (cover) is used.
If the following images/pages are having different dimensions (width & height), they may appear distorted.

7.3.3 Presets in FSI Server Interface

You can choose from the following presets when using the "Publish as FSI Pages" option in the FSI Server web interface "Publish To Web" Tab:

- **Catalog with links** Each image represents a page, no page margins, no overlays. If the image contains XML data in the "IPTC_FSI" data field, links will be presented on the page.
- **Catalog without links** Each image represents a page, no page margins, no overlays.
- **Teaser Thumbnail** No user interface, zoom is disabled. Pages flip automatically when idle. You can specify a link for the entire viewer area so that you can use this template to publish a thumbnail of a catalog linking to the full size catalog.

7.3.4 Printing Pages

If you want to enable users to print pages from within FSI Pages you can enable the built-in print function by setting the FSI Pages parameter Print to "true".

Example:

```
// In the configuration file:

...
<Pages>
<Print value="true" />
<PrintResolution value="1024" />
</Pages>
...


// Directly in the tag :
<fsi-pages dir="..." print="true" ...></fsi-pages>
```

This adds a "Print" button to the menu bar. Clicking the "Print" button displays the "Print" dialog of FSI Pages. The Print dialog offers different printing options to the user, depending on the current page and if the user zoomed into a page.

The printing options cover:

- Selected Range (e.g. 1-3; 9-20 or 1,6,8,12)
- All pages currently in view
- All pages

After the user has chosen an option and clicks the "Print" button in the dialog, FSI Pages downloads the required image data (pixel data) from the imaging server and opens the print dialog on the user's system once the download has finished.

You can limit the resolution used for printing using the FSI Pages parameter PrintResolution. Lower resolution leads to faster downloads at the cost of decreased print quality.

You can additionally pass effect parameters PrintEffects to sharpen the image or to specify the image compression.

7.3.5 Saving Pages

If you want to enable users to download a file related to the displayed page, the current section or the entire document, you can enable the save options by setting the FSI Pages parameter Save to "true".

Example:

```
// In the configuration file:

...
<Pages>
<Save value="true" />
<SaveResolution value="1024" />
</Pages>
```

```
...


// Directly in the tag :
<fsi-pages dir="..." save="true" ...></fsi-pages>
```

This adds a "Save" button to the menu bar. Clicking the button displays the "Save" dialog of FSI Pages. Without any additional preparation this allows users to download an image of the left or right page or of both.

After the user has chosen an option and clicks the "Save" button in the dialog, FSI Pages downloads the required image data (pixel data) from the imaging server.

You can limit the resolution used for downloading using the FSI Pages parameter SaveResolution.

You can additionally pass effect parameters SaveEffects to colorize, pixelize or sharpen the image or to specify the image compression.

Saving a complete PDF

In addition to downloading images you might want to enable users to download the complete source PDF document. This documentation refers to PDF documents only, but you can also provide other file types for download.

Enabling this option is as easy as adding the FSI Pages parameter SaveDocumentFile, providing the absolute URL to a PDF document to download.

Example:

```
// In the configuration file:
```

```
...
<Pages>
<Save value="true" />
<SaveResolution value="1024" />
<SaveDocumentFile value="https://foo.com/file.pdf" />
</Pages>
...


// Directly in the tag :
<fsi-pages save="true" savedocumentfile="https://foo.com/file.pdf"></fsi
-pages>
```

7.3.6 Special URL Values

For some special URL values FSI Pages ignores the URLPrefixes and URLSuffixes as well. The following table lists these special URLs:

URL	Description
javascript:foo()	Call the JavaScript function foo()
mailto:email@domain.tld	Send an email to email@domain.tld
#none	Do nothing. Handy if you just want to display a tool tip. Available in FSI Pages version 4.0.7 or above
#page=n	Forces FSI Pages to flip to page n
#showimage=[imageURL]	Display a custom JPEG image. The value [URL] needs to be an absolute URL to a JPEG image.

URL	Description
#zoom	Forces FSI Pages to zoom to the area covered by the link's shape(s)
#zoomarea	Magnifies the area covered by the link‘s shape(s) in place
#zoom=1,1,0,0,0.5,0.5	Forces FSI Pages to zoom to the image area following "="
#zoomonpage=n;strView	Combination of "#page=" and "#zoom=". This is: first flip to page n, then zoom to area strView. Available in FSI Pages version 4.0.7 or above

Please note that the special links starting with "#" need to be provided exactly as listed in the table above. Using e.g. "#zoomOnPage=[…]" or "foo.html#zoom" will link to the HTML anchor provided rather than executing an FSI Pages action.

For "mailto:" URLs the target frame is always "_self", except the XML <area> node specifies a different target.

7.4 FSI Pages Parameters

Parameters are added to the <fsi-pages> tag. Note that you can as well use FSI Pages configuration files:

Adding Parameters to FSI Pages:

```
<head>
//adding the fsipages.js script
```

```
<script src="viewer/applications/pages/js/fsipages.js"
type="text/javascript"></script>
</head>

<body>

//Adding parameters directly to the tag:
<fsi-pages width="500" height="600" dir="/images/catalog/" debug="true"
skin="silver" PageLayout="flip">
</fsi-pages>

//Example with using a FSI Pages configuration file:
<fsi-pages width="500" height="600" dir="/images/catalog/" cfg="sample/
catalog.xml">
</fsi-pages>

</body>
```

You can define individual configuration .xml files like this:

Simple Configuration File

```
<fsi_parameter>
  <image>
     <path value="samples/Watch.jpg" />
  </image>
  <Pages>
     <skin value="silver" />
     <debug value="1" />
         <pagelayout value="flip" />
         <thumbsize value="100" />
         <bendeffect value="none" />
  </Pages>
</fsi_parameter>
```

If you would like to use global parameters for all FSI Pages instances, the corresponding parameters can be defined in the _default.xml.

Example _default.xml:

```
<fsi_parameter>
        <Image>
                <ServerType value="FSI"/>
        </Image>


        <Pages>
                <skin value="silver" />
            <debug value="1" />
                 <pagelayout value="flip" />
                 <thumbsize value="100" />
                 <bendeffect value="none" />
        </Pages>
...

<Options>
        <FSIBase value="config/"/>
        <Language value="english"/>
        <ScenePreload value="true"/>
</Options>

</fsi_parameter>
```

You can optionally modify the way FSI Pages works by passing parameters to the script. To do so, pass an object containing the parameters and value as an object to the constructor:

Example:

```
var node = document.createElement("fsi-pages");


node.style.width = "600px";
node.style.height = "400px";

var parameters =
{"dir" : "images/mycatalog/", "skin":"silver", "debug":true};

$FSI.setParameters(node, parameters);
document.body.appendChild(node);


// Important: call this AFTER adding the node to the DOM
$FSI.initCustomNode(node);
```

7.4.1 General Parameters

debug	
Description	display debug and status information in the JavaScript console
Syntax	Boolean
Default	false
Context	FSI Pages

Display debug and status information in the browser's JavaScript console.

skin	
Description	defines the skin to display
Syntax	string
Default	"black"
Context	FSI Pages

Defines the skin which will be displayed. FSI Pages comes with three default skins, "black", "white" and "silver".

Skins are defined via CSS and can be found here: WEBAPPS/fsi/viewer/skins.

It is also possible to alter existing skins or to define custom skins in this location.

In order to use a custom skin, the easiest way would be to duplicate one of the default skin CSS files and adapt them to your liking. Make sure to use a custom class in front af ALL classes defined in the custom CSS, otherwise you will break other skins on the same HTML document.

Too see an example file, please go to the Appendix.

General appearance CSS rules, which are valid on an overall level, (e.g. the logo container, height of the menubar, etc.) are defined in the corresponding core CSS files, which can be found in WEBAPPS/fsi/viewer/skins/resources. (fsi-core.css, fsi-viewer-core.css)

IMPORTANT:

Never change the classes of the core CSS in the core files directly. If you would like to adapt the CSS rules listed there, please overwrite them in a custom skin file or in the HTML document instead.

You can find detailed tutorials for changing the appearance of the skin in the support section of our website:

- Create Basic Skins for FSI Viewer JS/ FSI Pages
- Create Advanced Skins for FSI Viewer JS/ FSI Pages

- Create Custom Buttons for Skins
- Create a Custom Skin using JavaScript

For a detailed example of a basic custom skin, please go to "Example of a custom skin CSS file" .

dir	
Description	path to the source images for the catalog on FSI Server
Syntax	String
Default	""
Context	FSI Pages

The path to the source images for the catalog on FSI Server.

The viewer will load the list of images from the given directory.

language	
Description	User interface language
Syntax	String
Default	""
Context	FSI Pages

Defines the language of tool tips for the user interface (menu bar).

Possible values depend on the XML based language files located in the sub-index "/languages" of the FSI Viewer setup path. To specify a language, please enter the filename without file extension, e.g. "german".

You can modify existing or add your own "language files".

cfg	
Description	relative path to an XML configuration file in FSI Pages format
Syntax	String
Default	""
Context	FSI Pages

The relative path to an XML configuration file in FSI Pages format.

hideUI	
Description	hides the user interface
Syntax	string
Default	false
Context	FSI Pages

Hides or displays the user interface.

Possible values:	
true	
false	
onMobileDevices:	UI is hidden on mobile devices
ifNoInPlaceZoom:	UI is hidden if in place zoom is deactivated

adaptiveUISize	
Description	determines if interface scales according to the zoom level .
Syntax	Boolean
Default	true
Context	FSI Pages

When set to true, the interface scales according to the zoom level.

Zoom	
Description	Enables/ disables page zoom
Syntax	Boolean
Default	true
Context	FSI Pages

Enables/ disables page zoom.

MaxZoom	
Description	Maximum zoom in percent of the source image.
Syntax	Float
Default	100
Context	FSI Pages

Maximum zoom in percent of the source image.

MenuButtonOrder	
Description	determines the button order in the user interface.

MenuButtonOrder	
Syntax	string
Default	
Context	FSI Pages

Determines the button order in the menu bar. Use the button ID which can be found in the respective skin.css (viewer\skins\resources\fsi-pages-core.css) for listing them.

Example MenuButtonOrder="ToggleFullscreen, FirstPage, LastPage"

Currently the following button IDs exist:

PrintPage, FirstPage, PreviousPage, PageInputEnter, NextPage, LastPage, ToggleBookmarkList, ToggleBookmark, ToggleTableOfContents, TogglePageIndex, ToggleFullScreen

ButtonsFirstLastPage	
Description	Enables/ disables First Page and Last Page Buttons
Syntax	Boolean
Default	true
Context	FSI Pages

When set to true, the buttos for directing to the first or the last page appear.

Index	
Description	Enables / disables page overview
Syntax	Boolean
Default	true
Context	FSI Pages

Enables or disables the page index containing thumbnails of all pages.

PageInput	
Description	Enables / disables page input field
Syntax	Boolean
Default	true
Context	FSI Pages

Specifies whether a text box for the page input is displayed in the menu bar.

PageInputEnterButton	
Description	Enables / disables page overview
Syntax	Boolean
Default	true
Context	FSI Pages

Specifies whether the enter button for the page input field is displayed in the menu bar.

ListTemplate	
Description	XML based template to use when retrieving image lists from the image server
Syntax	String
Default	"image_list.xml"
Context	FSI Pages

The XML based template to use when retrieving image lists from the image server via "dir" or "query" parameters. Available templates are located in the "WEB-INF/templates/fsi/" directory of your server setup.

PageDataTemplate	
Description	XML based template used for the Page Data
Syntax	String
Default	
Context	FSI Pages

The XML based template used for the Page Data.

DemoURL	
Description	Global link when clicking FSI Pages
Syntax	String
Default	-
Context	FSI Pages

Specifies a link to open when the user clicks anywhere on the FSI Pages instance. This parameter is especially useful when using a small preview of a catalog to lead the user to the full size version of FSI Pages. You can use URLs starting with "javascript:" to have a Javascript function called when the user clicks the viewer.

DemoURLTarget	
Description	HTML frame in which DemoURL is opened
Syntax	String
Default	-
Context	FSI Pages

HTML frame in which to open the URL specified by "DemoURL" parameter in.

You can use default frame names like "_self" or "_blank" or use a custom frame of your HTML frame set.

Print	
Description	Enables or disables the user to print
Syntax	Boolean
Default	true
Context	FSI Pages

Enable or disable the user to print current pages, the complete catalog or specific pages from the image collection shown with FSI Pages.

PrintResolution	
Description	Specifies the print resolution
Syntax	Integer
Default	2000
Context	FSI Pages

Specifies in which resolution the user is allowed to print the pages.

PrintEffects	
Description	Specifies print effects
Syntax	String
Default	
Context	FSI Pages

Defines image modification parameters to apply to images used for printing.

You can sharpen the image and define the JPEG compression level, e.g. "sharpen=100&quality=95".

Save	
Description	Enables or disables the user to save pages
Syntax	Boolean
Default	false
Context	FSI Pages

Enable or disable the download dialog where the user can save the current pages or the complete catalog, depending on the settings.

SaveResolution	
Description	Specifies the save resolution
Syntax	Integer
Default	2000
Context	FSI Pages

Specifies in which resolution the user is allowed to save the pages.

SaveEffects	
Description	Specifies save effects
Syntax	String
Default	
Context	FSI Pages

Defines image modification parameters to apply to images used for saving.

You can sharpen the image and define the JPEG compression level, e.g. "sharpen=100&quality=95".

SaveDocumentFile	
Description	URL to entire document (PDF)
Syntax	String
Default	
Context	FSI Pages

If you want to enable the user to save the PDF of your catalog, you can link the URL to the PDF here. This enables the option "complete document" in the download dialog.

Search

Search	
Description	Enable searching a catalog
Syntax	Boolean
Default	false
Context	FSI Pages

Enables or disables the search interface of FSI Pages.

SearchInput	
Description	Shows input for search
Syntax	Boolean
Default	true
Context	FSI Pages

Shows the input for search.

SearchType	
Description	Use "simplesearch" or "search" type for FSI Server
Syntax	String
Default	"simplesearch"
Context	FSI Pages

Using "simplesearch" makes FSI Server search in the data fields "iptc.fsi_search_data" and "iptc.caption" only by default rather than searching in all meta data fields.

SearchContext	
Description	meta data fields to search
Syntax	String
Default	"iptc.fsi_search_data,iptc.caption, iptc.fsi_extra"
Context	FSI Pages

Comma separated list of meta data fields names to search in.FSI Server with FSI Viewer parameter SearchType="simplesearch" only.

SearchAutoWildCards	
Description	Add wildcards ("*") to keywords automatically
Syntax	Boolean
Default	true
Context	FSI Pages

By default FSI Pages adds wildcards ("*") to the keywords the user entered into the search input of the search dialog. Set this parameter to "false" if you don‘t want FSI Pages to add wildcards automatically.

SearchAutoSelectFirstResult	
Description	Search Complete Action
Syntax	Boolean
Default	false
Context	FSI Pages

Select the first search result item when the search is complete (true, default) or just show the results (false).

SearchResultClickAction	
Description	Defines action when search result is clicked
Syntax	ENum
Default	flip
Context	FSI Pages

flip:	flip to result page (default
zoom:	zoom the result page

SearchTextOnStart	
Description	Search text on startup
Syntax	String
Default	—
Context	FSI Pages

Define a text to search for on start or an empty string (default) to not initiate a text search on start.

SearchThumbSize	
Description	Tumbnail size of search results
Syntax	Integer
Default	320
Context	FSI Pages

Defines the size of the Search result thumbnails.

SearchParameters	
Description	Adds parameters (modifiers) to the search query
Syntax	String
Default	—
Context	FSI Pages

This parameter provides the possibility to add parameters to the search query within FSI Pages. The parameters need to be provided in the regular HTTP request format: param1=value1&param2=value2[…].

SearchCustomURL	
Description	URL to request the search result from
Syntax	URL
Default	—
Context	FSI Pages

You can assign a custom URL to direct search requests to using this parameter. For example you might want to use a database and server side script to return pages related to the search request. This is the default way if you are not using imaging server.

SearchResultServerTemplate	
Description	Template for the search result
Syntax	String
Default	pages_searchresult" (FSI Server)
Context	FSI Pages

Specifies the server template to request in search queries.

SearchUseMethodGet	
Description	Use HTTP GET method to request search results
Syntax	Boolean
Default	false
Context	FSI Pages

By default FSI Pages requests search results using the HTTP POST method. Using this parameter you can force FSI Pages to use the HTTP GET method when requesting search results.

SearchSortResults	
Description	Sort search results
Syntax	Boolean
Default	True
Context	FSI Pages

If set to true, the search results are ordered by page number.

If set to false, the search results are listed in the order the imaging server sends them to FSI Pages.

7.4.2 Layout/Appearance Parameters

PageLayout	
Description	Animation when moving to another page
Syntax	String
Default	"flip"
Context	FSI Pages

Defines the animation when moving to another page. "scroll" slides the pages as if they were positioned next to each other (like in FSI Pages mobile)."flip" shows a page turn effect.

InitialPage	
Description	Page to display on startup
Syntax	Integer
Default	1
Context	FSI Pages

Specifies the page to display on startup, e.g. "1" for the front cover.

Ratio	
Description	Page to display on startup
Syntax	String
Default	1:1
Context	FSI Pages

The aspect ratio of pages displayed in FSI Pages.

You can specify any aspect ration (e.g. "320:240" or "1:2") or specify "auto" to use the aspect ratio of the first image in the image collection.

cropRect	
Description	Defines a crop rectangle
Syntax	String
Default	"0,0,1,1"
Context	FSI Pages

Crop rectangle (left, top, right, bottom) as 0.0..1.0 floating point values to display a given section of an image only.

Default: "0,0,1,1"

> **Note:**
>
> You cannot use "CropRect" AND "CropValues" parameters at a time.
>
> If you use both, "CropRect" will be ignored.

AutoFlipCropRect	
Description	If active, the right page uses a horizontally flipped crop rect
Syntax	Boolean
Default	true

AutoFlipCropRect	
Context	FSI Pages

If this parameter is enabled, the right page uses a horizontally flipped crop rect. If it is set to false, both pages use the same crop rect.

cropValues	
Description	Defines crop margin values for a page
Syntax	String
Default	"0,0,0,0"
Context	FSI Pages

Defines the amount that should be cut off the image from each side.

Values can be provided as pixel (e.g. "20px") or percent (e.g. "10%") or floating point numbers (e.g. "0.25").

effects	
Description	Image manipulation parameters for all pages
Syntax	String
Default	
Context	FSI Pages

The effects parameter can be used to pass image manipulation parameters to the imaging server. Typical manipulation parameters include the image saturation ("saturation") and image sharpening ("sharpen"). If no effect is specified, the default settings of the server (profile) apply.

Please refer to your FSI Server documentation for a list of available effect parameters.

MousewheelNavigation	
Description	defines if navigation with the mousewheel is possible
Syntax	Boolean
Default	true
Context	FSI Pages

Defines if navigation with the mousewheel is possible. If activated, you can turn pages while scrolling with the mousewheel.

ThumbSize	
Description	Max. size of thumbnails in the page index
Syntax	Integer
Default	160
Context	FSI Pages

The maximum size of thumbnails displayed in the index. The actual width and height depend on the aspect ratio of the pages.

Parameters only valid if PageLayout is set to scroll:

verticalLayout	
Description	moves pages vertical
Syntax	Boolean
Default	"false"
Context	FSI Pages

When set to true, the pages scroll in the vertical direction instead of the usual horizontal.

Note: only valid if PageLayout is set to "scroll"!

onePageInView	
Description	sets only one page in view
Syntax	Boolean
Default	"false"
Context	FSI Pages

Note: only valid if PageLayout is set to "scroll"!

doublePageMode	
Description	enables double page view
Syntax	Boolean
Default	"true"
Context	FSI Pages

If set to "false", the pages will not be displayed as coherent double pages.

Note: only valid if PageLayout is set to "scroll"!

autoDoublePageMode	
Description	
Syntax	Boolean
Default	"true"
Context	FSI Pages

Note: only valid if PageLayout is set to "scroll"!

bendEffect	
Description	gives pags a glossy bend effect
Syntax	String
Default	"glossy"
Context	FSI Pages

Defines if the pages are shown with a bend effect. "none" disables the effect.

FrontCover	
Description	enables/ disables the front cover
Syntax	Boolean
Default	"true"
Context	FSI Pages

Setting "FrontCover" parameter to false disables access to the front cover and forces FSI Pages to display the first image on the first inner left page.

BackCover	
Description	Enables/disables the back cover
Syntax	Boolean
Default	"true"
Context	FSI Pages

Setting "BackCover" parameter to false disables access to the back cover and forces FSI Pages to add a blank content page if required.

FrontCoverImage	
Description	ath to an image to use for the front cover

FrontCoverImage	
Syntax	String
Default	-
Context	FSI Pages

A path to an image to use for the front cover. FSI Pages will use the "ImageServer" parameter of FSI Viewer for relative paths.

The cover page(s) will be added to the images in the image list.

BackCoverImage	
Description	ath to an image to use for the back cover
Syntax	Boolean
Default	"true"
Context	FSI Pages

A path to an image to use for the back cover. FSI Pages will use the "ImageServer" parameter of FSI Viewer for relative paths.

The cover page(s) will be added to the images in the image list.

BlankBackCover	
Description	Use a blank page for back cover
Syntax	Boolean
Default	"true"
Context	FSI Pages

Use a blank page as back cover.

emptyImages	
Description	add blank pages
Syntax	String
Default	
Context	FSI Pages

Using this parameter you can add blank pages. The value of the parameter must contain one or more image page numbers of the blank pages to be inserted.

E.g. <EmptyImages value="2,10" /> adds two blank pages at page 2 and page 10, moving the other images in your collection correspondingly.

Alternatively you can add <image empty="true" /> nodes to your image collection.

removePages	
Description	Removes pages from the image collection
Syntax	String
Default	
Context	FSI Pages

Using this parameter you can remove images from the image collection. The behavior of FSI Pages is exactly as if the images would not be listed in the image collection.

E.g. <EmptyImages value="1,3,4" /> removes the first, third and fourth image in the collection. Please note that removing images takes place before inserting blank pages using the "EmptyImages" parameter.

IdleAutoTurn	
Description	Turns pages automatically when idle
Syntax	Boolean

IdleAutoTurn	
Default	false
Context	FSI Pages

Run the FSI Pages in demonstration mode and turn pages automatically when the user does not interact (= moves the mouse) for 2 seconds.

IdleAutoTurnMinPage	
Description	Defines the range for automatical page turning
Syntax	Integer
Default	-
Context	FSI Pages

Loop the auto flip action between IdleAutoTurnMinPage and IdleAutoTurnMaxPage. This way you can restrict the auto flip to a range of pages. If the initial page is outside the range specified, FSI Pages flips towards the range before looping between the range specified.

IdleAutoTurnMaxPage	
Description	Defines the range for automatical page turning
Syntax	Integer
Default	-
Context	FSI Pages

Loop the auto flip action between IdleAutoTurnMinPage and IdleAutoTurnMaxPage. This way you can restrict the auto flip to a range of pages. If the initial page is outside the range specified, FSI Pages flips towards the range before looping between the range specified.

IdleAutoTurnDelay	
Description	Specifies how long pages are shown when IdleAutoTurn is activated
Syntax	float
Default	2
Context	FSI Pages

Show each page for at least n seconds when flipping pages automatically. Please note that a page might be displayed for longer if loading the page takes longer than the time specified.

PageNumbers	
Description	Display page numbers on pages
Syntax	Boolean
Default	false
Context	FSI Pages

Show or hide page numbers on pages.

FirstPageNumber	
Description	Defines with which page number the catalog starts
Syntax	Integer
Default	1
Context	FSI Pages

Defines with which page number the catalog starts. By default (FirstPageNumber=1) the front cover is page number one.

You can specify an offset so that the page numbers start with a different value.

This parameter affects page numbers, the index and the display of the current page number in the FSI Pages user interface.

UseRomanPageNumbersToPage	
Description	Display Roman page numbers up to page number n
Syntax	Integer
Default	0
Context	FSI Pages

Defines the page number up to which FSI Pages uses Roman page Numbers.

The default value of "0" forces FSI Pages to display Roman numbers for negative numbers and 0.

If you set the value to "5" the display will be like this:

I,II,III,IV,V,6,7,8,9…

Note that you can use the Parameter FirstPageNumber to offset all page number values.

RomanPageNumbersOffset	
Description	Offset Roman page numbers by given value
Syntax	Integer
Default	0
Context	FSI Pages

Defines the value of the first Roman page number.

The default value of "0" means no offset so that the Roman numbers start with "I".

CustomPageNumbers	
Description	Comma separated string of custom page numbers
Syntax	String
Default	-
Context	FSI Pages

Use "CustomPageNumbers" to specify custom page numbers as a comma separated string. Items starting with underscore ("_") will not be displayed.

Pages without a custom page number will use the ordinary page numbering.

Example: < CustomPageNumbers value="_cover1,_cover2,3,4,5,_" />

In this case the cover page and the first left page will have no page number.

The first right page will start with page number "3". The sixth page again has no page number and the seventh and following page will be numbered as usual.

Note: The parameter FirstPageNumber has no impact on the custom page numbers. You can therefore use this parameter to modify the value of the first

ordinary page number.

CustomPageNumbersFile	
Description	URL or path to an XML file containing custom page numbers
Syntax	String
Default	-
Context	FSI Pages

Same as CustomPageNumbers, but the page number values will be loaded from an XML file. You can either define a path relative to the fsi/config/pages_xml/ directory or an absolute URL. The required format of the XML file is as follows:

<fsi>

<pagenumbers>3,6,7</pagenumbers>

</fsi>

RememberLastViewedPage	
Description	stores page number and displays page on startup next time
Syntax	Boolean
Default	false
Context	FSI Pages

If you enable this parameter FSI Pages stores the page number of the most recently viewed page on the user's computer and displays this page on start up the next time the user views the same catalogue (image collection) again.

RememberLastViewedPageExpireAfter	
Description	defines how long last viewed page will be stored
Syntax	Integer
Default	-
Context	FSI Pages

By default the last viewed page will be restored on the next visit of a user if you enable the RememberLastViewedPage parameter. You might want to specify an expire time after that the last viewed page will be discarded. Please specify the expire time in seconds, e.g. "3600" for one hour.

By default, the stored pages will be stored infinitely.

fsi-pages-intro-content	
Description	Defines content to be displayed next to front cover

fsi-pages-intro-content	
Syntax	String
Default	-
Context	FSI Pages

You can define some HTML Code to be displayed next to the front cover of the publication. This is useful to display e.g. usage instructions.

The content needs to be defined in the <fsi-pages> tag as following:

Example:

```
<head>
[...]
//adding the fsipages.js script
<script src="viewer/applications/pages/js/fsipages.js"
type="text/javascript"></script>
</head>

<body>
[...]
<fsi-pages width="500" height="600" dir="/images/catalog/" debug="true"
skin="silver" PageLayout="flip" Links="false">

        //Adding intro content:
        <fsi-pages-intro-content>
                <h3>Welcome to our catalog!</h3>
                <br />
                        <ul>
                                <li>Tap the Next Page/Previous Page
buttons </li>
                                <li>Double-click or pinch-zoom on a page
to zoom in</li>
```

```
                                <li>Tap the full-screen icon to switch
to full-screen mode</li>
                                <li>Click on the index icon for a
catalog overview</li>
                                <li>Enter a number in the page input
field to browse to a certain page</li>
                            </ul>
        </fsi-pages-intro-content>
</fsi-pages>
```

7.4.3 Link Parameters

Links	
Description	Enables/disables links
Syntax	Boolean
Default	true
Context	FSI Pages

Specifies whether links are displayed. If set to "false" Links and tool tips will not be displayed.

linkRGBANormal	
Description	Specifies color & opacity of links in normal state
Syntax	String
Default	"0000FFFF"
Context	FSI Pages

8-digit hexadecimal number specifying the color and opacity of links on the pages in normal state in the form "RRGGBBAA".

linkRGBAHover	
Description	Specifies color & opacity of links in hover state
Syntax	String
Default	"FF00FFFF"
Context	FSI Pages

8-digit hexadecimal number specifying the color and opacity of links on the pages in hover state in the form "RRGGBBAA".

linkRGBAActive	
Description	Specifies color & opacity of links in active state
Syntax	String
Default	"FF00FFFF"
Context	FSI Pages

8-digit hexadecimal number specifying the color and opacity of links on the pages in active state in the form "RRGGBBAA".

FollowLinks	
Description	Specifies if links are opened
Syntax	Boolean
Default	true
Context	FSI Pages

Setting this parameter to "false" prevents FSI Pages from opening links on the pages. Links and tool tips will nevertheless be displayed.

LinkTemplates	
Description	Template identifiers for link URLs
Syntax	String
Default	-
Context	FSI Pages

You can provide one or more identifiers that will be replaced in Link URLs.

The identifiers are case sensitive and must be separated by commas. The values to insert have to be defined by the parameter "LinkTemplateData" explained below.

LinkTemplateData	
Description	Values to replace template identifiers in link URLs
Syntax	String
Default	-
Context	FSI Pages

When defining "LinkTemplates" to replace place holders in Link URLs you can use this parameter to specify the values that shall be inserted. The number and sequence of the values must match the number and sequence of the templates you defined using the "LinkTemplates" parameter.

Multiple values have to be separated by commas and the individual values have to be provided url-encoded.

ForceLinkURL	
Description	Force URL for all links on pages
Syntax	String
Default	-
Context	FSI Pages

Force FSI Pages to use this URL for all links on all pages. This overwrites any URLs specified in the page data.

ForceLinkUrlPrefix, ForceLinkUrlSuffix	
Description	Force a link prefix / suffix for all links on pages
Syntax	String
Default	-
Context	FSI Pages

Force FSI Pages to use this prefix /suffix for all links on all pages. This overwrites any prefixes and suffixes specified in the page data.

DefaultLinkURL	
Description	Default URL for links on pages
Syntax	String
Default	-
Context	FSI Pages

Use this URL for links if no URL has been specified in the page data.

DefaultLinkUrlPrefix, DefaultLinkUrlSuffix	
Description	Default prefix / suffix for link URLs on pages
Syntax	String
Default	-
Context	FSI Pages

Use this prefix or suffix for all links on the pages if no prefix / suffix has been specified in the page data.

DefaultLinkTarget	
Description	Default target frame for all hyperlinks on all pages
Syntax	String
Default	-
Context	FSI Pages

Use this HTML target frame for links if no target frame has been specified in the page XML data.

ForceLinkTarget	
Description	Force a target frame for all hyperlinks on all pages
Syntax	String
Default	-
Context	FSI Pages

Force FSI Pages to use this HTML target frame for all links on all pages. Example. "_blank"

This overwrites any target frames specified in the page XML data.

Note: this parameter does not affect "javascript:" and "mailto:" links.

ForceJavascriptTarget	
Description	Force a target frame for all "javascript:" links on pages
Syntax	String
Default	-
Context	FSI Pages

Force FSI Pages to use this HTML target frame for all "javascript:" links on all pages. This overwrites any target frames specified in the page XML data.

Note: this parameter does not affect URL and "mailto:" links.

DefaultJavascriptTarget	
Description	Default target frame for all "javascript:" hyperlinks on all pages
Syntax	String
Default	-
Context	FSI Pages

Use this HTML target frame for "javascript:" links if no target frame has been specified in the page data.

ForceLinkTip	
Description	Force a tool tip for all links on pages
Syntax	String
Default	-
Context	FSI Pages

Force FSI Pages to use this tool tip for all links on the pages. This overwrites any tool tips specified in the page data.

DefaultLinkTip	
Description	Default tool tip for links on pages
Syntax	String
Default	-
Context	FSI Pages

Use this tool tip for all links on the pages if no tool tip has been specified in the page data.

7.4.4 Plug-in Parameters

plugins	
Description	adds a Plug-in (e.g. Fullscreen) to FSI Pages
Syntax	String
Default	-
Context	FSI Pages

Adds a plug-in to the viewer. Multiple plug-ins can be listed seperated by comma.

> **Please note:**
>
> If you would like to add specific parameters to the plugins via the fsi-pages tag, you can do this by prefixing the parameter name with the name of the plug-in, e.g. pagesthumbbar_height="100"

FullScreen

Adding the parameter plugins:"FullScreen, Resize" enables a full screen view option in FSI Pages. The Resize parameter needs to be set in order to ensure proper adjusting to the full screen.

> **Note:**
>
> If the FSI Pages is embedded in an iFrame, the attribute allowfullscreen="true" needs to be added to the iFrame. Otherwise the

FullScreen Plug-in will not work due to security restrictions of the browser.

PagesThumbBar

Adding the parameter plugins:"PagesThumbBar" displays a row of thumbnails at the bottom of the pages.

The following parameters can be used to adapt the PagesThumbBar:

PageNumbers	
Description	Page numbers being displayed below thumbnails
Syntax	Boolean
Default	true
Context	FSI Pages

Enables/ disables the display of page numbers being below the thumbnails.

Autohide	
Description	Auto hides thumbar below 600px
Syntax	Boolean
Default	true
Context	FSI Pages

If set to true, the thumbnails will not be displayed if the FSI Pages Instance is smaller than 600px.

Height	
Description	Defines height of the thumbnails
Syntax	Integer
Default	160
Context	FSI Pages

Defines the height of the thumbnails displayed.

Bookmarks

Adding the parameter plugins:"Bookmarks" adds a bookmark functionality to FSI Pages.

The following parameters can be used to adapt the bookmarks functionality:

ThumbSize	
Description	Defines the thumbnail size in the bookmark list.
Syntax	Float
Default	200
Context	FSI Pages

Defines the maximum thumbnail size in the bookmark list in pixel.

ShowIconsOnPages	
Description	Shows icon on bookmarked pages
Syntax	Boolean
Default	true
Context	FSI Pages

Shows icon on bookmarked pages.

PageNumbers	
Description	Shows page numbers in bookmark list
Syntax	Boolean
Default	true
Context	FSI Pages

Shows page numbers in bookmark list

PersistentStorage	
Description	Defines if the bookmark data is being cached..
Syntax	Boolean
Default	true
Context	FSI Pages

When disabled, the bookmarks will be deleted when the user leaves the FSI Pages instance.

When enabled the data will be cached for the given time (defined by PersistentStorageExpiresAfter)

PersistentStorageExpiresAfter	
Description	Keep note data for given time.
Syntax	Integer
Default	-

PersistentStorageExpiresAfter	
Context	FSI Pages

Defines after which period of time the cached note data expires.

Chapters

Adding the parameter plugins:"Chapters" adds a select box to the user interface of FSI Pages. The user can directly access sections of a catalog by selecting the corresponding item in the select box. You can assign colors and different levels to each chapter using simple XML based data describing the document structures.

Example:

```
<Indexdata>
        <Index label="First Page" page="1" color="FFFF00" />
        <Index label="Chapter 1" page="10" color="00FF00">
                <Index label="Chapter 1.1" page="20" color="00FF00"/>
                <Index label="Chapter 1.2" page="30" color="00FF00"/>
                <Index label="Chapter 1.3" page="40" color="00FF00"/>
        </Index>
        <Index label="Chapter 2" page="50" color="0000FF"/>
        <Index label="Chapter 3" page="60" color="CCCCCC"/>
        <Index label="Chapter 4" page="70" color="FF0000"/>
</Indexdata>
```

The following parameters can be used to adapt the Chapters:

indexDataFile	
Description	URL to an external XML file describing the document structure
Syntax	string
Default	-
Context	FSI Pages

You can use an external XML file to provide the XML document structure data required for this plug-in. The XML file must be located in the same domain as FSI Viewer to avoid cross-domain security restrictions.

Width	
Description	Width of the select box
Syntax	Integer
Default	-
Context	FSI Pages

Usually the FSI Pages skin defines the width and position of the select box. Alternatively you can specify a fixed width in pixels using this parameter.

7.5 FSI Pages JavaScript Interface

The JavaScript interface of FSI Pages can be useful if you - for example - want to set certain actions when FSI Pages is initialized, while pages change, the user interface is used or links are clicked.

7.5.1 FSI Pages Public Methods

void	**start();**
	Starts FSI Pages.

nID	**getInstanceID();**
	Returns the ID of the Pages JS object.

strInstanceName	**getInstanceName();**
	Returns the name of the viewer object.

void	**changeConfig(strCfgFileName, oParameters);**
	Reset the viewer object and change the configuration to the given configuration file. strCfgFileName (optional): path to the configuration file (see "cfg" parameter) oParameters (optional): an object containing parameters If present, these parameters will overwrite the parameters defined upon initialization.

void	**printAPI();**
	Shows all API methods in the console in alphabetical order.

void	**init(domElement, oParameters, bDebug);**
	Initializes the chosen Pages JS element.

bInitDone	**getInitDone();**
	Returns if the initialization process is done.

void	**destroy();**
	Destroys the given FSI Viewer JS object. You should destroy the instance before you remove the object spins tag from the DOM tree.

bEnabled	**debugEnabled();**
	Returns if the debug mode is enabled.

mixed	**debugError();**
	Returns the corresponding viewer error messages in the console.

mixed	**debugLog();**
	Returns the corresponding viewer log messages in the console.

mixed	**debugWarn();**
	Returns the corresponding viewer warning message in the console.

mixed	**getConfigValue(strName);**
	Returns the value of the configuration parameter "strName".

mixed	**traceConfigValue(strName);**
	Traces a config value and enables you to see where (in config files, the _default.xml, etc) a specific value is set.

bResult	**getConfigBoolean(strParameterName, bDefault);**
	Returns the currently configured value of the requested boolean parameter.

fResult	**getConfigFloat(strParameterName, fDefault);**
	Returns the currently configured value of the requested float parameter.

nResult	**getConfigInt(strParameterName, nDefault);**
	Returns the currently configured value of the requested integer parameter.

strResult	**getConfigString(strParameterName, strDefault);**
	Returns the currently configured value of the requested string parameter.

bSuccess	**addListener(strListenerName, fn, iScope);**
	Calls a specific listener. Listeners which are added via addListener(), remain valid when changeConfig() is called.

bSuccess	**removeListener(strListenerName, fn);**
	Removes a specific listener..

void	**showAboutWindow();**
	Shows the "About" window.

void	**nextPage();**
	Goes to next page.

void	**previousPage();**
	Goes to previous page.

void	**firstPage();**
	Goes to first page.

void	**lastPage();**
	Goes to last page.

bPresent	**getMenuButtonPresent(strButtonID)**
	Shows active menu buttons, can be used to remove inactive buttons from the menu if you are using a custom skin.. NOTE: Method only delivers correct results after the callback "onInitMenu" is called.

void	**pressMenuButton(strButtonID)**
	Sets the menu button to active.

void	**clickMenuButton(strButtonID)**
	Sets the menu button to active.

void	**releaseMenuButton(strButtonID)**
	Disables the active menu button.

void	**registerExternalMenuButton(elButton, strButtonID, strTip)**
	Registers an external menu button.

void	**unregisterExternalMenuButton(elButton)**
	Unregisters an external menu button.

void	**registerExternalMenuButtonsFromContainer(elContainer)**
	Registers an external menu button from a specific container.

void	**unregisterExternalMenuButtonsFromContainer (elContainer)**
	Unregisters an external menu button from a specific container.

strClassName	getSkinClassName()
	Returns the name of the current used skin class, e.g. fsi-skin-black

bSuccess	setFullScreenElement(elFullScreenContainer)
	Defines the FullScreen element.

strPageNumber	getCurrentPageNumber()
	Returns the current page number.

strPageNumber	getPageNumberString(nPage)
	Returns the current page number.

oDimension	getPageContainerDim()
	Returns the Page container dimensions.

void	goToPage(strPage, bAnimate)
	Moves to a certain page. strPage is a string and works like the parameter → .

void	**goToPageNumeric(nPage, bAnimate)**
	Moves to a certain page. Works like goToPageNumber, but with nPage being numeric.

nPage	**getTargetPage(nPage)**
	Return the nPage limited to minPage and maxPage. If FSI Pages shows double pages, the value of the left page matching nPage will be returned.

strVersion	**getVersion();**
	Returns a string containing the FSI Viewer JS software version.

strVersion	**getBuild();**
	Returns a string containing the FSI Viewer JS software build number.

strContentID	**getContentID()**
	Returns a string containing the content ID.

void	**showMenuToolTip(strTip, elSrc, x, y)**
	Shows custom tooltips.. strTip: HTML code of thetooltip or ID in language.xml files elSrc: the DOM element wich is used to align the tooltip, usually event.target x,y: (optional) offset for tooltip position

void	**hideMenuToolTip()**
	Shows custom tooltips.. strTip: HTML code of thetooltip or ID in language.xml files elSrc: the DOM element wich is used to align the tooltip, usually event.target x,y: (optional) offset for tooltip position

oPageLayoutData	**getPageLayoutData()**
	oPageLayout has the following properties: boolean bDoublePages int nImages int nMaxPage int nMinPage

oParameters	**getParameters();**
	Returns an object containing all parameters set via javascript or custom tag attributes.

mixed	**togglePageIndex();**
	Toggles the page index.

bResult	**makeBoolean(mixedCOnvert, bDefaultValue);**
	Transforms a value into a boolean value, e.g. "1" to true, "false" to false ***bDefaultValue*** is optional and false if not defined.

mixed	**getPluginParameter(strPluginName, strParameterName, defaultValue, bMakeBoolean);**
	Returns the parameter defined for the given plugin. ***defaultValue*** and ***bMakeBoolean*** are optional. example: iViewer.getPluginParameter("pages", "dir");

bSuccess	**showSearchDialog(bShow)**
	Shows search dialog.

bSuccess	searchForString(strKeyWords)
	Searches for a string.

void	gotoPageAndZoom(strPage, strViewstring)
	Goes to a certain page and zooms in.

void	testAPIListenersStart()
	Starts the API test and shows all called listeners and the parameters which they return.

void	testAPIListenersStop()
	Stops the API test.

7.5.2 FSI Pages Callbacks

You can define callback functions by assigning methods to the FSI Pages instance.

The following callback methods exist:

	onReady();
	Called as soon as FSI Pages finished loading data and gets interactive.

	onDestroy();
	Called when the fsi-pages element is destroyed.

	onInitFailed(strReason);
	Called when the initialization is failed. This could happen if e.g. an image cannot be loaded or the dimensions of an image cannot be requested. The listener returns a string with the reason why the initialization failed.

	onChangeConfig(strCfgFileName, oParameters);
	Called when the config file is changed.

	onMenuEnabled(bEnabled);
	Called if the menu is enabled/disabled.

	onMenuDestroy();
	Called if the menu is destroyed.

	onMenuButtonEnabled(strButtonID, null, bEnable);
	This optional callback function can be used call an action when a button is enabled or disabled.

	onMenuButtonPressed(strButtonID, evt);
	This optional callback function can be used to set an action that starts when a button is pressed.

	onMenuButtonReleased(strButtonID, evt);
	This optional callback function can be used to set an action when a certain button is released..

	onMenuButtonClicked(strButtonID, evt);
	This optional callback function can be used to set an action when a certain button is released..

	onMenuButtonSetPressed(bPressed);
	This optional callback function can be used to call an action when a button is pressed.

	onRegisterExternalMenuButtons()
	This optional callback function can be used to call an action as soonexternal menu buttons are registered.

	onAfterRegisterExternalButton(elButton, strButtonID, bButtonPresentInViewer)
	This optional callback function can be used to remove inactive buttons from the menu bar if you are using a custom skin.

	onMenuReady();
	This optional callback function can be used to set an action when the menu is ready (after all buttons are added).

	onFullScreen(bSet);
	This optional callback function can be used to call an action when FullScreen mode is enabled or disabled.

	onPageChanged(nPage, strPage, strMaxPage);
	This optional callback function can be used to call an action as soon as the viewed page changes.

	onShowPageIndex(bShow);
	This optional callback function can be used to call an action as soon as the page index is opened.

	onShowBookmarkList(bShow);
	This optional callback function can be used to call an action as soon as the bookmark list is opened.

	onShowChapter(bShow);
	This optional callback function can be used to call an action as soon as the chapters opened.

	onShowChaptersData(arIndexItems);
	This optional callback function can be used to return the Chapters data if the Chapters plugin is implemented (arIndexItems can be multi-dimensional if sub-chapters exist).

	onShowPrintDialog(bShow);
	This optional callback function can be used to call an action as soon as the print dialog is opened.

	onAutoPageLayoutChange(bDoublePage);
	This optional callback function can be used to call an action as soon as the page layout is automatically changed.

	onPageLayoutChange(oPageLayoutData);
	This optional callback function can be used to call an action as soon as the page layout is manually changed.

	onBookmarkAdded(nPage);
	This optional callback function can be used to call an action as soon as a bookmark is added to the page.

	onBookmarkRemoved(nPage);
	This optional callback function can be used to call an action as soon as a bookmark is removed of the page.

	onBookmarksLoaded(strLoaded);
	This optional callback function can be used to call an action as soon as the bookmarks are loaded.

	onBookmarksSaved(strLoaded);
	This optional callback function can be used to call an action as soon as the bookmarks are saved.

	onRemoveAllBookmarks();
	This optional callback function can be used to call an action as soon as all bookmarks are removed.

	onPageSizeChanged(nWidth, nHeight);
	This optional callback function can be used to call an action as soon as the page size is changed.

	onPageZoomed(nPage);
	This optional callback function can be used to call an action as soon as the page size is zoomed.

	onPrintPages(arPageIndices, arImageURLs);
	This optional callback function can be used to call an action as soon as the pages are printed.

	onPageToolTip(nLinkAreaIndex, strURL, strTipContent);
	This optional callback function returns a string that is useful if you want to modify the tool tip content.

	onOpenPageLink(nLinkAreaIndex, strURL, strTarget);
	This optional callback function can be used to call an action if links are clicked. Return "false" to prevent opening the link., return {strURL:"foo", strTarget: "bar"} to modifiy the URL or the target (both are optional)

	onResize(nWidth, nHeight, bResizeDone, BFullScreen);
	Called when the fsi-pages element is resized.

	onSkinChanged(strCurrentSkinClass, strCurrentSkinClassBefore)
	This optional callback function can be used to call an action as soon as the skin class is changed.

	onSearchInit(elInput)
	This optional callback function can be used to call an action as soon as the search is initialized.

	onSearchResults(arPageNumbers, arImages)
	This optional callback function can be used to call an action as soon as the search results are returned.

	onSearchStart(strKeyWords)
	This optional callback function can be used to call an action as soon as the search is started.

	onShowSearchDialog(bShow)
	This optional callback function can be used to call an action as soon as the search dialog is showed.

8 Appendix

8.1 Escaping or URL-encoding parameter values

In some situations you need to pass escaped (URL-encoded) parameters values to FSI Viewer, because using unescaped values would break the XML syntax or because you need to use characters prohibited in URLs.

In cases like this you can pass escaped values rather than passing the actual value.

What does escaping (URL-encoding) mean?

URL-encoding (or percent-encoding) means converting the character to its corresponding value in ASCII and then representing that value as a pair of hexadecimal digits. The digits, preceded by a percent sign ("%"), are then used in place of the reserved character. FSI Viewer transforms URL-encoded values back to the original value.

How can I URL-encode a value?

Most scripting languages (JavaScript, ActionScript, PHP and alike) provide methods for this purpose. Please note that FSI Viewer requires Unicode safe encoding rather than non-unicode safe encoding – this is an escaped character might be represented by one or more 2-digit numbers preceded by "%".

Using JavaScript you therefore need to use the method "encodeURIComponent()" instead of "escape()".

8.2 HTML Tags available in FSI Viewer

FSI Viewer supports basic HTML tags in various contexts like when entering custom tool tips, texts for the "LargeToolTips" plug-in, search results and alike. Whenever this documentation refers to "Basic HTML formatting tags", the following HTML tags can be used (variable values printed bold italic):

HTML Tags

```
<b></b>
<i></i>
<u></u>
<br/>
<font color="#FF0000" size="12" face="Arial"></font>
<p align="left|center|right"></p>
```

Please note that you need to use XML syntax in order to avoid XML syntax errors in configuration files - this is e.g. using <br/> instead of <br>.

8.3 Example of a custom skin CSS file

In order to use a custom skin, the easiest way would be to duplicate one of the default skin CSS files and adapt them to your liking. Make sure to use a custom class in front af ALL classes defined in the custom CSS, otherwise you will break other skins on the same HTML document.

General appearance CSS rules, which are valid on an overall level, (e.g. the logo container, height of the menubar, etc.) are defined in the corresponding core CSS files, which can be found in WEBAPPS/fsi/viewer/skins/resources. (fsi-core.css, fsi-viewer-core.css, fsi-pages-core.css)

IMPORTANT:

Never change the classes of the core CSS in the core files directly. If you would like to adapt those CSS rules, please overwrite them in a custom skin file or in the HTML document instead.

Example:

```
/* IMPORTANT NOTES */
/* - make sure to use a custom class in front af ALL classes defined
here
(in this sample: .fsi-skin-example) */
/* otherwise you will break other skins on the same HTML document */
/* enables a floating menu bar which overlaps with the image */
/* useful for transparent menu bars */
/* baseSize defines the size of all skin elements (default 16) */
/* baseEnlargement defines the general enlargement
of the icons when the viewport is changed, default is 8 */


.fsi-skin-example .fsi-ui-info {
  content:'{"floatingMenuBar":true, "floatingNavWindow":false,
"baseEnlargement":10}';
}
/* the main menu bar container */
.fsi-skin-example .fsi-ui-menu-bar {
  background: rgba(255, 255, 255, 0.8);
}
/* The logo displayed in the menu bar.
Custom logos need to be placed into the WEBAPPS/fsi/viewer/skins/
resources/ folder of
your FSI Viewer installation.
NOTE: Originally, this is defined in the fsi-viewer-core.css.
Since the core css files should not be altered, please overwrite the
classes from
there in tyour custom skin CSS. */
.fsi-viewer-root .fsi-skin-example .fsi-ui-logo-image {
background-image:url("[fsi-skin-path]resources/custom_logo.svg");
  width: 3.2em;
  height: auto;
  margin-left:0.15em;
}
/* defines the position of the loading animation */
```

```
.fsi-skin-example .fsi-viewer-loading-animation {
  left:50%;
  top:50%;
  transform:translate(-50%, -50%);
}
/* the base button class */
.fsi-skin-example .fsi-ui-button {
  background: rgba(255,255,255,0.2);
  border:1px solid solid rgba(255,255,255,0.2);
  border-radius: 6px;
}
/* styles for hovered buttons */
.fsi-skin-example .fsi-ui-button-hover {
background-color:#FFF;
box-shadow: 2px 2px 2px rgba(0,0,0,0.3);
border:1px solid rgba(56,176,170,0.2);
}
/* styles for pressed buttons */
.fsi-skin-example .fsi-ui-button-active {
  border:1px solid rgba(56,176,170,0.2);
  background:#ffffff;
  box-shadow: inset 2px 2px 5px rgba(0,0,0,0.3);
}
/* the base button icon class */
.fsi-skin-example .fsi-ui-button-icon {
  color: #33b1aa;
}
.fsi-skin-example .fsi-ui-button-disabled .fsi-ui-button-icon {
  color:#bec7c6;
}
.fsi-skin-example .fsi-ui-button-hover .fsi-ui-button-icon {
}
.fsi-skin-example .fsi-ui-button-active .fsi-ui-button-icon {
}
.fsi-skin-example .fsi-ui-menu-bar-disabled .fsi-ui-button-icon {
  color: #bec7c6;
}
/* select frame */
/* PSEUDO CLASS, only these 2 values will be read */
```

```
.fsi-skin-example .fsi-ui-select-frame {
  border-color:#33b1aa;
  background-color: rgba(255, 255, 255, 0.25);
}
/* progress bar displayed when loading image sequences */
/* PSEUDO CLASS, only these 3 values will be read */
.fsi-skin-example .fsi-ui-progressbar {
  color:#33b1aa;
  background-color:rgba(255,255,255,0.5);
  border-color:#999;
}
/* PSEUDO CLASS, only the color values will be read */
.fsi-skin-example .fsi-ui-progressbar-text {
  color:#555;
}
/* tool tips */
.fsi-skin-example .fsi-ui-tool-tip {
  color:#4d4d4d;
  background-color:rgba(255,255,255,0.8);
  border:1px solid #4d4d4d;
  box-shadow: 2px 2px 2px rgba(0,0,0,0.5);
}
.fsi-skin-example .fsi-ui-tool-tip-arrow {
  border-top: 7px solid #4d4d4d;
}
/* navigation window */
.fsi-skin-example .fsi-ui-nav-window {
  background-color: rgba(255,255,255,0.8);
}
.fsi-skin-example .fsi-ui-nav-window-image {
  border: 1px solid #33b1aa;
  background-color: rgba(255,255,255,0.8);
}
/* PSEUDO CLASS, only these 2 values will be read */
.fsi-skin-example .fsi-ui-nav-frame {
  border-color: #33b1aa;
  background-color: rgba(255,255,255,0.35);
}
/* the text 0.0% to 100.0% below the nav window image */
```

```
.fsi-skin-example .fsi-ui-nav-window-zoom-text {
  color:#4d4d4d;
}
/* HotSpots */
.fsi-skin-example .fsi-ui-hotspot {
  color:#fff;
  border:1px solid rgba(255,255,255, 1.0);
  border-radius: 2px;
  background-color: rgba(88, 168, 255, 0.85);
  box-shadow: 2px 2px 2px rgba(0,0,0,0.3);
}
.fsi-skin-example .fsi-ui-hotspot-hover {
  background-color: rgba(88, 168, 255, 1.0);
}
.fsi-skin-example .fsi-ui-hotspot-active {
background-color: rgba(88, 168, 255, 1.0);
}
.fsi-skin-example .fsi-ui-hotspot-line-color {
  color: #336699;
}
.fsi-skin-example .fsi-ui-hotspot-line-background-color {
  color: #113399;
}
/* dialogs */
.fsi-skin-example .fsi-dialog{
  border:0px none;
  background-color:#FFF;
  color:#206662;
}
.fsi-skin-example .fsi-dialog input[type="button"] {
  font-size: 14px;
  color: #666;
  border:1px solid #eee;
  background-color: transparent;
}
.fsi-skin-example .fsi-dialog input[type="button"]:hover {
  color: #206662;
  background-color: #EEE;
}
```

```
/* FSI Pages */
.fsi-skin-example .fsi-ui-button.PageInput {
  background-color: white;
  border:1px solid #CCC;
  color: #333;
}
.fsi-skin-example .fsi-ui-button.PageInput input:focus {
  border-color:#999;
  color: #000;
}
```

9 Index

A

Appendix
- Encoding Parameter Values 267
- HTML-Tags 267
- Skin Example 268

F

FSI ImageFlowThumbBar
- Callbacks
  - onClick 187
  - onStart 187
- Parameters
  - autoDestroy 173

FSI Pages
- Usage
  - Special URL values 203

FSI Pages JS
- How it works 193
- Public Methods
  - traceConfigValue 250
- Usage 194
  - Aspect Ratio 199
  - Defining Image Collections 197
  - Presets 200

FSI Pages JS JavaScript Interface
- Callbacks
  - onAfterRegisterExternalButton 261
  - onAutoPageLayoutChange 262
  - onBookmarkAdded 263
  - onBookmarkRemoved 263
  - onBookmarksLoaded 263
  - onBookmarksSaved 263
  - onChangeConfig 259
  - onDestroy 259
  - onFullScreen 261
  - onInitFailed 259
  - onMenuButtonClicked 260
  - onMenuButtonEnabled 259
  - onMenuButtonPressed 260
  - onMenuButtonReleased 260
  - onMenuButtonSetPressed 260
  - onMenuDestroy 259
  - onMenuEnabled 259
  - onMenuReady 261
  - onOpenPageLink 265

- onPageChanged 261
- onPageLayoutChange 263
- onPageSizeChanged 264
- onPageToolTip 264
- onPageZoomed 264
- onPrintPages 264
- onReady 258
- onRegisterExternalMenuButtons 260
- onRemoveAllBookmarks 264
- onResize 265
- onSearchInit 265
- onSearchResults 265
- onSearchStart 266
- onShowBookmarkList 262
- onShowChapter 262
- onShowChaptersData 262
- onShowPrintDialog 262
- onShowSearchDialog 261, 266
- onSkinChanged 265

- Public Methods
  - addListener 251
  - changeConfig 248
  - clickMenuButton 252
  - debugEnabled 249
  - debugError 249
  - debugLog 249
  - debugWarn 250
  - destroy 249
  - firstPage 252
  - getBuild 255
  - getConfigBoolean 250
  - getConfigFloat 250
  - getConfigInt 251
  - getConfigValue 250
  - getContentID 255
  - getCurrentPageNumber 254
  - getInitDone 249
  - getInstanceID 248
  - getInstanceName 248
  - getMenuButtonPresent 252
  - getPageContainerDim 254
  - getPageLayoutData 256
  - getPageNumberString 254
  - getParameters 257
  - getPluginParameter 257
  - getSkinClassName 254
  - getTargetPage 255
  - getVersion 255
  - goToPage 254
  - gotoPageAndZoom 258
  - goToPageNumeric 255
  - hideMenuToolTip 256
  - init 249
  - lastPage 252
  - makeBoolean 257
  - pressMenuButton 252
  - previousPage 252
  - printAPI 248
  - registerExternalMenuButton 253
  - registerExternalMenuButtonsFromContainer 253

- releaseMenuButton 253
- searchForString 258
- setFullScreenElement 254
- showAboutWindow 251
- showMenuToolTip 256
- showSearchDialog 257
- start 248
- testAPIListenersStart 258
- testAPIListenersStop 258
- togglePageIndex 257
- unregisterExternalMenuButton 253
- unregisterExternalMenuButtonsFromContainer 253

FSI Pages JS Parameters

- ForceLinkTip
  - ForceLinkTip 241
- General
  - adaptiveUISize 211
  - ButtonsFirstLastPage 212
  - cfg 210
  - debug 207
  - DemoURL 214
  - DemoURLTarget 214
  - dir 209
  - hideUI 210
  - Index 212
  - language 209
  - ListTemplate 213
  - MaxZoom 211
  - MenuButtonOrder 211
  - PageDataTemplate 214
  - PageInput 213
  - PageInputEnterButton 213
  - Print 215
  - PrintEffects 215
  - PrintResolution 215
  - Save 216
  - SaveDocumentFile 217
  - SaveEffects 216
  - SaveResolution 216
  - skin 208
  - Zoom 211
- Layout/ Appearance
  - autoDoublePageMode 226
  - AutoFlipCropRect 223

- BackCover 227
- BackCoverImage 228
- bendEffect 227
- BlankBackCover 228
- cropRect 223
- cropValues 224
- CustomPageNumbers 233
- CustomPageNumbersFile 233
- doublePageMode 226
- effects 224
- emptyImages 229
- FirstPageNumber 231
- FrontCover 227
- FrontCoverImage 227
- fsi-pages-intro-content 234
- IdleAutoTurn 229
- IdleAutoTurnDelay 231
- IdleAutoTurnMaxPage 230
- IdleAutoTurnMinPage 230
- InitialPage 222
- MousewheelNavigation 225
- onePageInView 226
- PageNumbers 231
- Ratio 222
- RememberLastViewedPage 234
- RememberLastViewedPageExpireAfter 234
- removePages 229
- RomanPageNumbersOffset 232
- ThumbSize 225
- UseRomanPageNumbersToPage 232
- verticalLayout 225

- Links 236
  - DefaultJavascriptTarget 241
  - DefaultLinkTarget 240
  - DefaultLinkTip 241
  - DefaultLinkURL 239
  - DefaultLinkUrlPrefix 239
  - DefaultLinkUrlSuffix 239
  - FollowLinks 237
  - ForceJavascriptTarget 240
  - ForceLinkTarget 240
  - ForceLinkURL 238
  - ForceLinkUrlPrefix 239
  - ForceLinkUrlSuffix 239
  - linkRGBAActive 237
  - linkRGBAHover 237
  - linkRGBANormal 236
  - LinkTemplateData 238
  - LinkTemplates 238
- Search 217
  - SearchAutoSelectFirstResult 219
  - SearchAutoWildCards 218
  - SearchContext 218
  - SearchCustomURL 220
  - SearchInput 217
  - SearchParameters 220

- SearchResultClickAction 219
- SearchResultServerTemplate 221
- SearchSortResults 221
- SearchTextOnStart 219
- SearchThumbSize 220
- SearchType 218
- SearchUseMethodGet 221

FSI Pages JS Plugin Parameters
- Bookmarks 244
  - PageNumbers 245
  - PersistentStorage 245
  - PersistentStorageExpiresAfter 245
  - ShowIconsOnPages 244
  - ThumbSize 244
- Chapters 246
  - indexDataFile 247
  - Width 247
- FullScreen 242
- PagesThumbBar 243
  - Autohide 243
  - Height 244
  - PageNumbers 243
- Plugins 242

FSI QuickZoom
- Callbacks
  - onConfigsReady 153
  - onDestroy 153
  - onInit 153
  - onModifyPositionAndSize 154
  - onStart 153
  - onZoomEnd 153
  - onZoomStart 153
- Parameters
  - addCSSClass 142
  - autoZoomDimension 143
  - debug 140
  - disableByCSSClass 141
  - enableByCSSClass 141
  - headers 140
  - imgSrcAttributeName 141
  - inPlaceZoom 143
  - magnification 142
  - maxZoomWindowHeight 144
  - maxZoomWindowWidth 144
  - minZoomWindowHeight 145
  - minZoomWindowWidth 145
  - moveRangeCenterCorners 143
  - pageMargin 146
  - skipImageHeight 146
  - skipImageWidth 145

- useDevicePixelRatio 140
- zoomImageBorderWidth 146
- zoomImageOffset 146

- Public Methods
  - addImage 150
  - addListener 152
  - destroy 150
  - getBuild 151
  - getConfigValue 150
  - getInstanceID 151
  - getInstanceName 151
  - getVersion 151
  - init 150
  - printAPI 150
  - removeImages 151
  - removeListener 152
  - scanForNewImages 152
  - testAPIListenersStart 152
  - testAPIListenersStop 152
  - traceConfigValue 151

FSI Showcase JS
- Supported Devices 190
- Usage 190

FSI ThumbBar
- Callbacks
  - onConfigsReady 188
  - onDestroy 188
  - onFocus 188
  - onInit 188
  - onMouseEnter 189
  - onMouseLeave 189
  - onScrollEnd 189
  - onScrollStart 189
  - onSetScrollRange 189
  - onZoomEnd 188
  - onZoomReady 188
  - onZoomStart 188
- How it works 158
- Parameters
  - Alignment 170
  - autoResize 167
  - autoRotateSpeed 171
  - centerZoomedImage 170
  - cmdButtonSelector 173
  - cropRect 165
  - cropValues 164
  - debug 161
  - depthFadeout 172
  - dir 162
  - effects 164
  - elementSpacing 166
  - elementWidth 166
  - enableZoom 169
  - endlessScrolling 171
  - format 163
  - garbageCollectionSize 176
  - headers 162
  - initialImage 168
  - overlays 163
  - paddingBottom 167
  - paddingTop 167

- PerspectiveAlignment 174
- placeHolderImage 175
- placeHolderPadEffect 175
- preloadCount 171
- presentationType 168
- quality 164
- renderer 163
- ScrollBar 170
- useTouchZoom 169
- vertical 168
- viewerSelector 173, 174
- zoomShaderEndOpacity 172
- zoomShaderStartOpacity 172

- Public Methods
  - addListener 182
  - assignFSIViewer 181
  - buttonDown 183
  - buttonUp 183
  - changeConfig 183
  - changeDir 184
  - closeZoom 182
  - destroy 181
  - firstImage 184
  - focusImage 182
  - getBuild 181
  - getConfigValue 184
  - getImageCount 184
  - getInitDone 181
  - getInstanceID 181
  - getInstanceName 182
  - getScrollPos 184
  - getVersion 181
  - init 180
  - isScrolling 184
  - lastImage 185
  - nextImage 185
  - nextPage 185
  - previousImage 185
  - previousPage 185
  - printAPI 180
  - registerButton 185
  - removeListener 182
  - setScrollPos 186
  - startAutoRotate 186
  - stopAutoRotate 186
  - testAPIListenersStop 183, 183
  - traceConfigValue 186
  - unRegisterButton 186
  - zoomImage 182
- Supported Devices 155
- Usage 155

FSI TouchZoom
- Callbacks
  - onConfigsReady 135
  - onDestroy 135
  - onInit 135
  - onZoomChanged 135
- Initialization 122
- Parameters
  - debug 125
  - disableByCSSClass 128
  - enableByCSSClass 128
  - imgSrcAttributeName 127
  - loadingImage 127
  - monitorPositions 125
  - useDevicePixelRatio 125
  - useTiledImages 126
  - zoomPrecision 126
- Public Methods
  - addImage 130
  - addListener 134
  - destroy 132
  - enable 133
  - getBuild 132
  - getConfigValue 131
  - getImageCount 133
  - getInstanceID 132
  - getInstanceName 132
  - getVersion 132
  - init 132
  - lockImage 133
  - lockImageSourceImage 133
  - lockUpdates 133
  - printAPI 131
  - removeImages 130, 130
  - removeListener 134
  - resetImage 133
  - scanForNewImages 129
  - setZoomPrecision 131
  - testAPIListenersStart 134
  - testAPIListenersStop 134
  - traceConfigValue 131
  - triggerRefresh 134
  - updateImagePositions 130
- Supported Devices 120
- Usage 120

FSI Viewer JS
- Callbacks
  - onAfterRegisterExternalButton 116
  - onAnimationComplete 115
  - onAnimationStart 115
  - onChangeConfig 115
  - onChangeImage 115
  - onConfigsReady 113
  - onDataComplete 118
  - onDestroy 114
  - onFullScreen 119
  - onHotspotEvent 119
  - onInit 114
  - onInitFailed 114
  - onInitMenu 118
  - onMenuButtonClicked 117
  - onMenuButtonEnabled 116
  - onMenuButtonPressed 117
  - onMenuButtonReleased 117
  - onMenuButtonSetPressed 117
  - onMenuDestroy 114
  - onMenuEnabled 114
  - onMenuReady 114
  - onMouseModeChanged 118
  - onProgress 116
  - onReady 113
  - onRegisterExternalMenuButtons 116
  - onReset 117
  - onResize 115
  - onSkinChanged 116
  - onStartDragging 118
  - onStopDragging 118
  - onViewChanged 119
  - onZoomChanged 115
- Conditional Configuration Options 23
  - Building Conditions 26
  - Cascaded Conditions 24
  - Conditions and Groups 25
- Configuration 17
  - Configuration Parameters Hierarchy 21
  - Debug Mode 17
  - Parameter Retrieving from FSI Server 29
  - XML Configuration Files 18
- Embedding at runtime 96
- How it works 11
- Index Structure 16
- Parameters
  - Implementation 31
- Public Methods 98
  - addListener 103

- changeConfig 99
- changeImage 99
- clickMenuButton 106
- debugEnabled 100
- debugError 100
- debugLog 100
- debugWarn 100
- destroy 102
- getBuild 103
- getConfigValue 101
- getHotspotViewString 105
- getImageIsTransparent 108
- getInFullScreenMode 108
- getInitDone 101
- getInstanceID 102
- getInstanceName 102
- getMenuButtonPresent 106
- getMenuHeight 103
- getMouseMode 102
- getParameters 109
- getPluginParameter 110
- getSkinClassName 107
- getSpinFrames 108
- getSrcDim 108
- getVersion 103
- getViewString 105
- getZoom 104
- gotoHotspot 106
- gotoViewString 105
- hideMenuToolTip 109
- init 101
- makeBoolean 110
- pressMenuButton 106
- printAPI 100
- registerExternalMenuButton 107
- registerExternalMenuButtonsFromContainer 107
- releaseMenuButton 107
- removeListener 103
- resetView 104
- setFullScreenElement 108
- setMouseMode 102
- setZoom 104
- showAboutWindow 103
- showMenuButton 103
- showMenuToolTip 109
- spinToDegree 101
- spinToTarget 102
- start 100
- startAutoSpin 110
- stopAutoSpin 111
- testAPIListenersStart 110
- testAPIListenersStop 110
- traceConfigValue 101
- unregisterExternalMenuButton 107
- unregisterExternalMenuButtonsFromContainer 107
- updateSize 104

- Spin Videos 95
- Supported Devices 9
- Usage 9
  - Additional Buttons 15
  - Menu Bar 13
  - Mouse Modes 14
  - Navigator Window 15

J

JavaScript InterfaceFSI Pages JS JavaScript Interface
- Public Methods
  - getConfigString 251
  - nextPage 251
  - removeListener 251

P

Parameter
- Camera
  - armAngleMeasureSetup 68
  - armEndAngle 69
  - armJointPosition 68
  - armLength 68
  - armStartAngle 69
  - digitalCropFactor 61
  - distanceToTableCenter 65
  - focalLength 63
  - heightAboveTable 66
  - heightCamera 65
  - heightTable 65
  - position, targetPosition 63
  - targetHeightOffset 66
  - targetWidthOffset 66
- Display Options
  - adaptiveUISize 52
  - autoDisablePointerActions 54
  - autoHideNav 49
  - autoSpinInterval 50
  - autoSpinSpeed 49
  - backgroundColor 51
  - desktopClickZoomScale 51
  - documentZoom 53
  - enableZoom 50
  - hideUI 48
  - inPlaceZoom 53

- loadExactPreviewImagesAfterResize 54
- maxZoom 54
- minInstanceSize 55
- mouseSensitivity 51
- noNav 49
- pellets 52
- pelletsOpacity 52

- General
  - autoDestroy 45
  - cfg 36
  - debug 35
  - dir 39
  - drawCube 42
  - drawGroundPlate 44
  - drawTable 43
  - effects 40
  - headers 35
  - imagesources 39
  - initialMouseMode 44
  - initialView 41
  - initialViewPersistent 41
  - InvisibleLoadingPurposes 37
  - language 40
  - menuButtonOrder 45
  - navHeight 36
  - navWidth 36
  - NoSceneLoop 48
  - NoSetLoop 47
  - panLimits 41
  - preventImageTransparency 42
  - reverseSceneSets 47
  - SceneSets 46
  - skin 37
  - spinDirectionImage 46
  - src 39
  - tableDiameter 43
  - urlLoadingAnimation 37
  - useDevicePixelRatio 42
- Global Hot Spot Parameter
  - decodeHTMLEntitiesInHotSpotTexts 60
  - enableHotspots 57
  - hotspotCallbackFunction 58
  - hotspotDotColor1 60
  - hotspotDotColor2 60
  - hotspotDotRadius 60
  - hotspotLineColor1 59
  - hotspotLineColor2 59
- Hot Spot
  - actionParameter 80
  - activeClass 83
  - cssClass 85
  - cssStyle 85
  - focusAngleX 82
  - focusAngleY 82
  - focusOnClick 81
  - focusRadius 82
  - horizontalMarkerPosition 75
  - horizontalMarkerSize 74
  - hoverClass 83
  - interactive 80
  - markerAngle 76
  - normalClass 83
  - perimeterAngle 76
  - perimeterAngleFadeOut 78
  - perimeterAngleFadeOutX 78
  - perimeterAngleRange 77

- perimeterAngleRangeX 77
- perimeterAngleX 77
- perimeterImageWidth 79
- perimeterZoom 79
- rotateOnFocus 81
- text 74
- url 80
- urlTarget 81
- verticalMarkerPosition 75
- verticalMarkerSize 75
- x 73
- y 73
- z 73
- zoomOnFocus 82

- Plugin
  - AutoSpin 91
  - AutoSpin_Button 93
  - AutoSpin_Interval 92
  - AutoSpin_preventClickZoomWhileSpinning 92
  - AutoSpin_Speed 92
  - FullScreen 91
  - MouseModes 93
  - MouseModes_MenuOffset 93
  - MouseModes_Mode 94
  - MouseModes_Sequence 95
  - plugins 90
- Preview Options
  - progressBarHAlign 57
  - progressBarHeight 56
  - progressBarVAlign 57
  - progressBarWidth 56
  - showPreviewWhileLoading 55
  - showProgressbar 56
  - spinPreviewWhileLoading 55
- Video
  - base 87
  - captureFrames 87
  - captureOffsetEnd 88
  - captureOffsetStart 88
  - frameOffset 90
  - heightCorrectionFactor 89
  - sceneSet 86
  - source 86
  - spinDirection 87
  - widthCorrectionFactor 88
  - xCorrectionFactor 89
  - yCorrectionFactor 89
- virtualSpace
  - baseRotationX 70
  - baseRotationY 70
  - baseRotationZ 70
  - cubeLineWidth 70
  - cubeSizeX 69
  - cubeSizeY 69

- cubeSizeZ 69
- position 72
- rotationXRange 70
- rotationXSets 71
- rotationYRange 70
- rotationYScenes 71
- shiftX 71
- shiftY 71
- shiftZ 71

Presets
- Usage
- Printing Pages 200