

PAGEFLIP4

USER'S MANUAL

FOR VERSION 4.1

This document is the online user's manual of the IpariGrafika PageFlip version 4.1.

IMPORTANT! Unlike earlier versions, *PageFlip4* includes a basic level copy protection feature preventing unauthorized use of the book on homepages, or publications. This protection does not limit the License owner in any way; the use of the created Book is limited by the License holder. For more information on the configuration of copy protection, refer to the Appendix A.

TABLE OF CONTENTS

1 INTRODUCTION

- 1.1 New features (for users of the 3.6 version)
- 1.2 New *PageFlip4* features
- 1.3 Terms

2 OVERVIEW

- 2.1 The Screen
- 2.2 The Book
- 2.3 Preview
- 2.4 ControlBar
- 2.5 Copyright Message
- 2.6 Zoom View
- 2.7 Thumbnail View
- 2.8 Print
- 2.9 Sounds
- 2.10 Background music
- 2.11 Files

3 CONFIGURATION OF *PAGEFLIP4*

- 3.1 Creating test environment
- 3.2 Modification of the XML file
- 3.3 Use of the XML Validator
- 3.4 Sizes
- 3.5 Customizing the Screen and its structure
 - 3.5.1 Default font
 - 3.5.2 Popup and Tooltip settings
- 3.6 Customizing the ControlBar
- 3.7 Customizing the Book (Hardcover, hardpage, largecover etc.)
- 3.8 PageOrder
- 3.9 Preparing the graphical content
- 3.10 Pages & Spreads
- 3.11 Use of HotSpot
- 3.12 Table of Contents & Info Floating window content
- 3.13 Configuring the Menu language(s)
- 3.14 Publication of the created PageFlip

APPENDIX

- Copy Protection settings
- Interactivity
- Customizing ControlBar Skinning
- Font Embedding
- XML reference
- Hotspot XML reference
- PageFlip4* Error messages
- Tips for Optimization
- PageFlip4* Embedding
- Use of LightBox
- Color Codes
- mostFAQ

1 INTRODUCTION

The main function of *PageFlip4* is to display a document like a book with pages that can be flipped, while all other features are subordinated to that. With the help of *PageFlip4*, User can create a book from simple images or interactive flash movie files, and flip its pages on the screen, publish it on the Internet, or on any kind of media.

PageFlip4 can load any flash compatible content, so display and interactivity is not limited to printed publications or pdf. Since *PageFlip4* can be widely customized, it can even serve as basis for homepages, where the content is displayed on the pages.

PageFlip4 is recommended to everyone eager to share one or more of their documents in a stylish, book-like form, and who has basic knowledge of image manipulation (scaling) and XML file editing (text file modification).

FROM SIMPLE TO ADVANCED

While *PageFlip4* has the simplest possible configuration, it is also extremely customizable to meet all the various needs. The solution is to have most of the settings as default, so the configuration file only needs to include the modified settings from the more than 200 attributes.

By default, our book starts on the first page, the pages can be flipped, there are navigation buttons, thumbnail screen, zooming, printing, and all we have to do is prepare the contents to be loaded to the pages, and adjust the size at the settings.

Or, there is the advanced use, where the interactive content takes the control over the book, where every visible detail is customized, where the sound material can be heard over the background music, and so on. And if we reach the limits of XML configuration, further possibilities are available with the help of the source codes in the Developer package.

1.1 NEW FEATURES (FOR USERS OF THE 3.6 VERSION)

This version is a completely new software. It has only inherited the feature list of its predecessor (not completely identical). There are changes in the programming language and structure: *PageFlip4* is an Object-Oriented software written in AS3. From the outside, speed can strike users, while the inner improvements will be very useful during later developments, which will hopefully be available for Users in the form of free updates.

However, there are spectacular and exciting new features. The most significant are the following:

- Multiflip: flipping simultaneously more than one pages
- Multi Level of Transparency: optional number of consecutive transparent pages
- Real Spread: real spreads working perfectly without tricks
- New perspective Hard Cover
- Tilted Book view
- Content Preview: preview of the pages beside the Book
- Background image and background music
- Background image and background music assigned to pages

Some further modifications with less importance:

- Transitions
- Embedded international fonts (Latin)
- 5 kinds of HotSpot shapes
- Popup windows with images, headlines, texts
- ToolTip
- Immediate menu language switch
- Copy protection

1.2 NEW PAGEFLIP4 FEATURES

PageFlip4.1 is the first update of *PageFlip4*. It includes some important new features, and some bugfixes (for more information see the version.txt in the *PageFlip4* package)

Here is the list of the major new features:

- External Script: allows User to executes custom AS3 scripts, and access *PageFlip4* APIs, and recieves flipping events
- Execute custom function by HotSpot clicking
- Custom Controlbar Layout: let User place the control bar elements on the Screen
- Click to Zoom, and Drag Zoom: access zoom image by simply clicking on the page and then move the Zoomed image by dragging it with the mouse
- Center Single Page: centers the book when closed
- Controlling shadow visibility
- Page Name display

1.3 TERMS

The following concepts need to be clarified for easier understanding:

PageFlip, PageFlip4 – the application itself.

User, Publisher - *PageFlip4* License owner, the person or company that edits, publishes, and uploads the Book to the Internet. Do not mix up with the visitors, who browse the Book with the help of *PageFlip4*.

Screen – The display interface of *PageFlip4*, the area where the elements of *PageFlip4* (the book itself, the buttons, the thumbnails, etc.) are displayed.

Book - the main ingredient of PageFlip is the book with pages that can be flipped (magazine, newspaper, any other publication with pages that can be flipped will be referred to as Book in this manual).

Content Preview, Preview – the series of thumbnails in a scrollable bar, appearing simultaneously with the Book.

Thumbnails – the view of *PageFlip4* when all the pages of the Book appear in small preview images

Page – a page of the Book

Page-file – The file specifying the content of the page

2 OVERVIEW

2.1 THE SCREEN

The following elements can appear on the *Screen* of *PageFlip4*, depending on the configuration and the current view:

- Preview
- Book
- ControlBar
- CopyrightMessage
- Zoom view
- Thumbnail View
- Floating Windows (Table of Content, Info)
- Tooltips

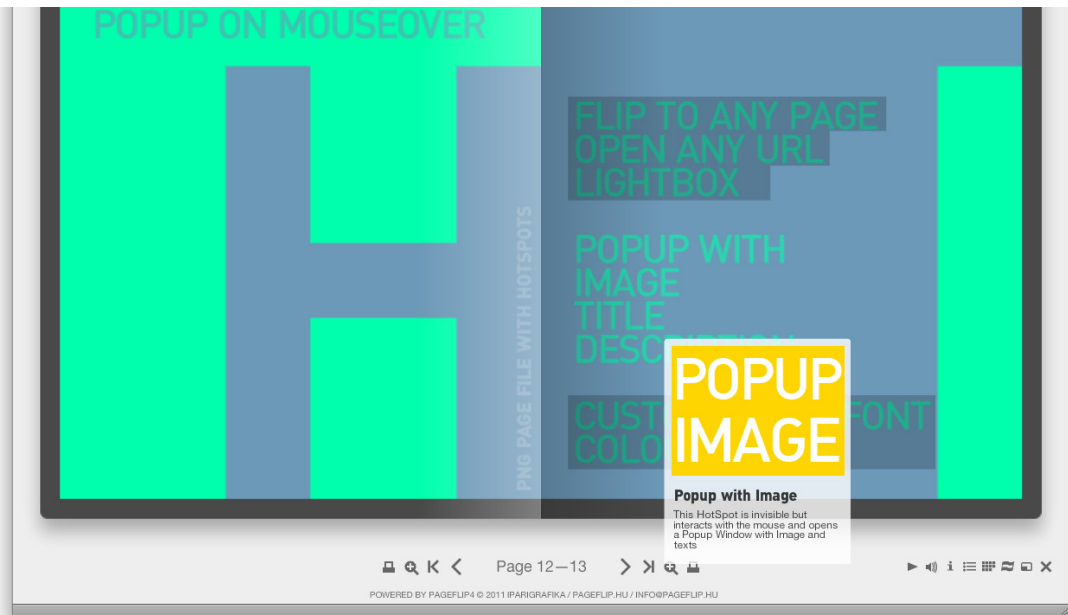


(Figure 2.1) The Book, with Preview, Table of Contents, ControlBar and Copyright message

2.2 THE BOOK

The Book includes all the pages of the document. The pages of the Book can be flipped with the help of the mouse, by drag'n'drop, or by clicking on the corner of the pages. User can navigate with the help of the ControlBar, or the Keyboard. Pages can be flipped horizontally, or vertically as a calendar. The pages can have soft or hard. The Cover pages can be larger than the pages inside.

We can create HotSpots on the pages with the help of XML configurations. HotSpots are areas created by the User on the screen that can open Popup windows if the mouse hovers over it, and send User to another page if User clicks on them. HotSpots can also open links or LightBox images.



(Figure 2.2) HotSpot hovered, and a Popup window opens with image, and texts

2.3 PREVIEW

Optionally, User can have a series of thumbnails beside the Book, that we can hover over with the mouse (if cursor is moved to the edges, it starts and accelerates), and the Book flips to the page on which we clicked.



(Figure 2.3) The Preview Bar

2.4 CONTROLBAR

By default, below the Book we find the ControlBar that can be divided into three areas: the loader status is visible optionally on the left side, the middle section includes the flip, zoom, print buttons and the current page number where the book is open, while the right side has all the other functions, namely AutoFlip, Sound, download, information, table of contents, thumbnails, language, full screen, exit.



(Figure 2.4) The ControlBar

The ControlBar can be disabled, if we want to view the Book only.

Each button of the control bar has a Key assigned, so if the given button is active, it can also be activated from the keyboard.

By clicking into the text box, the cursor appears and User can specify the page number or name* to jump to, and by pushing ENTER, the specified page opens if it exists.

If the mouse is over a button for more than 2 seconds, the assigned ToolTip appears (that can be configured in the language section**)

2.5 COPYRIGHT MESSAGE

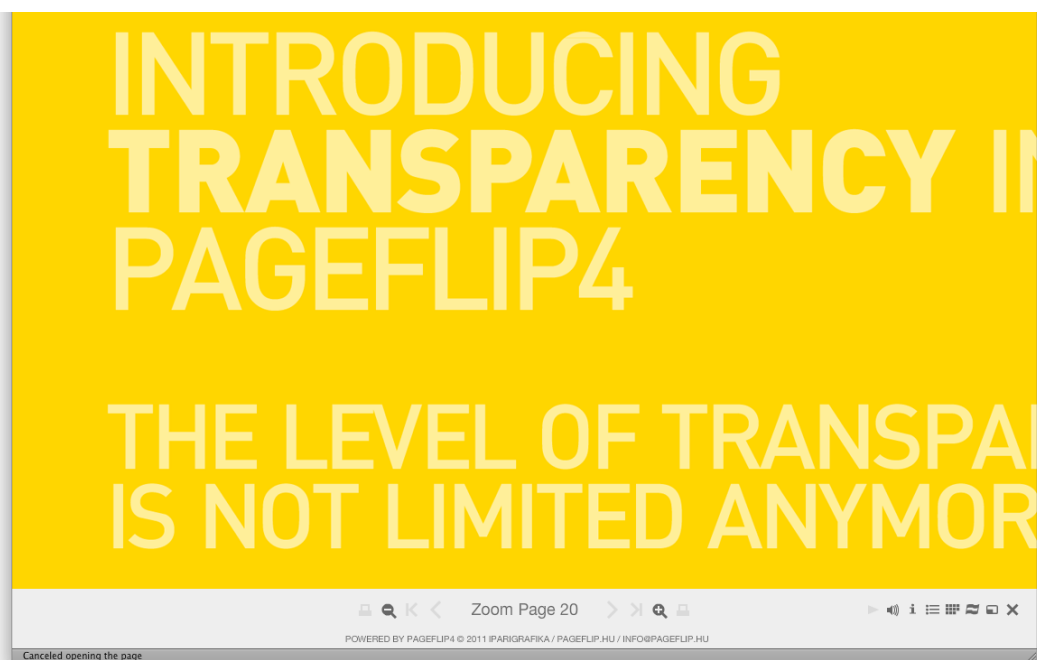
A one-line message in small print indicating the owners of the document.



(Figure 2.5) The Copyright Message

2.6 ZOOM VIEW

An enlarged area of a given page. More than one zooming level can be specified, depending on the configuration. A separate file belongs to all levels of zooms, which loads only if necessary (except if User enables the immediate loading of all sizes).



(Figure 2.6) The Zoom View

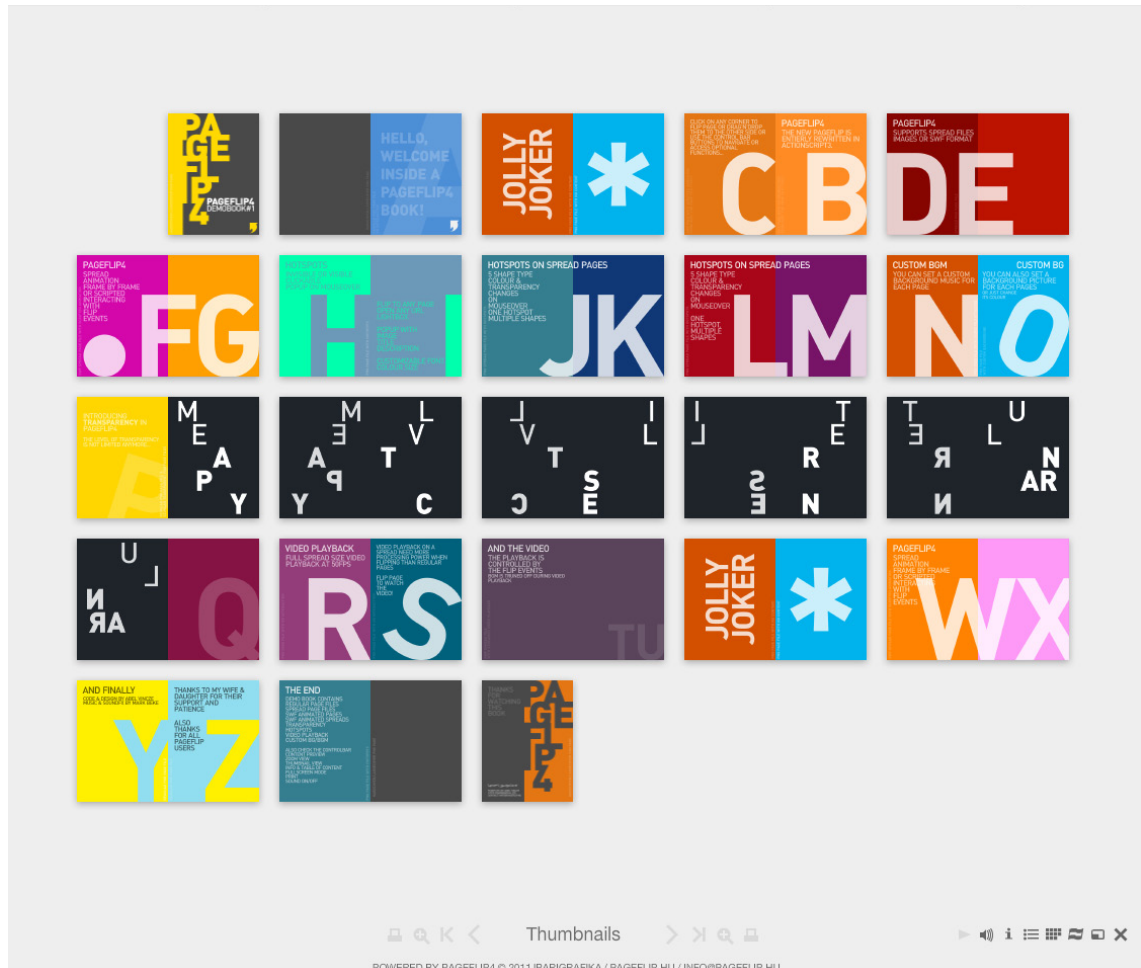
If User clicks on Zoom in Book view, the + sign turns to – with which the zoom view can be left, or the view can be decreased. User can further zoom in with the zoom button on the other side, if more than one zoom sizes have been prepared.

Scroll with the help of the mouse if the zoom does not fit the screen.

2.7 THUMBNAIL VIEW

In this view, PageFlip displays the small thumbnail images of the Book's pages in order on one or more pages, depending on the resolution of the screen.

If the cursor is moved over the pages, a ToolTip appears with the page number of the current spread, and the Book goes to the page by click.



(Figure 2.7) The Thumbnail View

2.8 PRINT

Each page of the Book – if enabled – can be printed, but only one-by-one.

2.9 SOUNDS

The Book has several flip sounds that User can change, or disable.

2.10 BACKGROUND MUSIC

User can specify one main background music track for *PageFlip4*, and sounds/music can be assigned to each page separately, which start to play only if the given page is visible. In this case, the main background music optionally fades out.

2.11 FILES

The *PageFlip4* package includes 3 main folders:

obj/	- Files of the <i>PageFlip4</i> and the Book contents
src/	- Source files (source files of the Page-files)
utils/	- location of the XML Validator
obj/css/	- Styles belonging to the HTML embedding file
obj/js/	- SWFobject and LightBox Javascript codes
obj/pageflipdata/	- External files of <i>PageFlip4</i>
obj/pageflipdata/backgrounds/	- Background images
obj/pageflipdata/BGM/	- Background music
obj/pageflipdata/hotspots/	- HotSpot description xml files
obj/pageflipdata/popupimages/	- Images loaded by popup windows
obj/pageflipdata/sounds/	- Flipping sound files
obj/pageflipdata/pages/	- Page-files
obj/pageflipdata/pages/thumbs/	- Thumbnail views of pages
obj/pageflipdata/pages/zoom/	- Enlarged views of the pages

PageFlip4 can be started with the index.html.

By default, all files loaded by PageFlip are located in one given folder: pageflipdata/. The configuration XML file is also saved in this folder (pageflipdata.xml).

3 CONFIGURATION OF *PAGEFLIP4*

3.1 CREATING THE TEST ENVIRONMENT

The downloaded *PageFlip4* is ready for testing immediately after extraction. Make sure to save the extracted folders with the *PageFlip4* files to the right location, and to place the later created contents within the original file-structure for easier handling.

3.2 MODIFICATION OF THE XML FILE PATH

the `pageflipdata.xml` (default file name) configuration file is in the `pageflipdata/` folder. Both the name and the location of the file can be modified optionally in FlashVars variables:

```
flashvars.XMLFileName = "pageflipdata.xml";  
flashvars.DataFolder = "pageflipdata/";
```

Edit the file with a simple Text Editor (for example TextEdit in OS X, or the Notepad in Windows) if no dedicated XML editor is available, but not with a word processor like MS Word. Never format fonts, colors, or sizes within the document, as they save formatting information in the file, and will break the XML structure and PageFlip will not be able to handle it.

When editing, current attribute values can be modified in the quotation marks after the equal sign. Regardless of the data type, quotation marks can never be left out.

Add new attributes to the XML with extra care. The simplest way is to push Enter after the existing attribute of the given category, and add the new one after that in the next line.

3.3 THE USE OF THE XML VALIDATOR

The XML Validator provides copy protection to our *PageFlip4*. Start the XML Validator before running *PageFlip4* online or offline. *For further information on the use of the XML Validator, refer to the Appendix A.*

3.4 SIZES

The preparation of the Book's pages depends on its final environment.

After specifying the resolution to optimize to, User can calculate the *Screen* area, including the Book, so that Preview (if enabled) and the ControlBar are also visible.

Consider the type of the Book (Horizontal/Vertical), if Large Cover is preferred, and the area designated for the Book, and then calculate the ideal page size.

Also consider that in case of Full Screen view, the graphic elements of the pages might need larger resolution. For this reason and because the system automatically scales the Book (default setting), use the larger, rather than smaller, page size for calculation.

If the content of the book allows no free scaling, choose a page size that fits full screen at the minimum resolution.

The following size settings can be found in the Settings node of the XML configuration file:

```
PageWidth="300"  
PageHeight="400"
```

The Width and Height of the pages. If scaling of the pages (Settings/PageScale="false") is disabled, the pages of the Book appear on the *screen* in this size.

If User loads content to a page that does not fit the specified page size, and automatic scaling (Settings/AutoSize="true") is enabled, then it will scale to this size.

When flipping Spreads, if animation during flipping is enabled ****, a Bitmap copy of the page is created in this size.

Coordinates of the loaded HotSpots have to match this page size, thus User should configure HotSpots when the Book is done.

```
MinStageWidth=""  
MinStageHeight=""
```

The minimum allowed size of the *screen*. At the scaling of the Flash Player or Browser window, *PageFlip4* continuously adjusts its content to have everything in its right place and size on the *Screen*. However, if User wants the window to be smaller than the minimum possible size, it is overruled, avoiding the overlaying of various contents.

If the Book cannot be scaled, then the minimum size should be enough to display the Book, the ControlBar and the Preview (if enabled).

```
LargeCover="true"  
LargeCoverHorizontalOversize="16"  
LargeCoverVerticalOversize="32"
```

If User wants the cover to be larger than the inside pages, LargeCover has to be enabled (Settings/LargeCover="true"). The difference in size has to be specified horizontally and vertically. If, for example, the Cover is larger around the edges by 16 pixels than the pages inside, then the cover page will be 16-pixel larger horizontally (Settings/LargeCoverHorizontalOversize="16"), while the difference vertically (margin is added on the top and the bottom) will be 32 pixels (Settings/LargeCoverVerticalOversize="32"). If User creates a vertical Book, then the horizontal size difference will be 32 pixels and the vertical will be 16 pixels.

Scaling, by default, concerns 4 pages, the external and inside pages of the front and back Covers. If User sets the Book to be open all the time (Settings/AlwaysOpened="true"), or set odd page number Book, then it is modified based on the variations of the settings.

3.5 CUSTOMIZING THE SCREEN AND ITS STRUCTURE

The *Screen* itself is an interface on which other elements of *PageFlip4* can be found. User can set a background, and specify the layout of the displayed content.

User can choose from three types of backgrounds:

No background

User can embed *PageFlip4* over html content, so that its background is transparent. Transparent background has to be enabled in the XML:

```
TransparentBackground="true"
```

and we have to enable the transparency of the Flash Movie at the embedding in the html:

```
params.wmode = "transparent";
```

If transparent background is disabled in the html, the default background is white, or the color set in the SWFObject parameter:

```
params.bgcolor = "#FFFFFF";
```

Background color

```
TransparentBackground="false"  
BackgroundColor="0x2A2A2A"
```

User can specify an optional background color (see *Appendix K: Color codes*)

Disable the transparency of the background (default setting), and set a background color.

Background image

Regardless of the background color, *PageFlip4* loads the image from the given URL and places it in the background. While the image is being loaded, the set or default background color is displayed.

```
TransparentBackground="false"  
BackgroundImageFile="backgrounds/demoBack2.jpg"
```

THE LAYOUT OF THE *PAGEFLIP4* ELEMENTS

The order of the Preview, the Book and the ControlBar can be set:

```
LayoutOrder="PREVIEW-CONTENT-CONTROLBAR"
```

By default, they appear from top to bottom in the following order:

- Preview
- Content
- ControlBar

User can freely modify the order, for example: "*CONTROLBAR-CONTENT-PREVIEW*"

The Preview has to be enabled, as it is turned off by default (not available in personal version):

```
ContentPreviewEnabled="true"
```

The Content can either be the Book, the zoom or the thumbnail view. Floating Windows appear on this content area too. When scaling the *Screen*, the size of the content part changes, the other features have fix sizes; for more information, see the configuration of the Preview and the ControlBar.

THUMBNAILS

`ThumbnailsEnabled="true"`

Enabling the use of the thumbnail view. Set the Thumbnail view button as follows (see: 3.6 Customizing the ControlBar):

<code>ThumbnailModeStart="false"</code>	- Start of the PageFlip in Thumbnail view
<code>ThumbnailWidth="120"</code>	- Size of pages in Thumbanil view
<code>ThumbnailHeight="120"</code>	
<code>ThumbnailDropShadow="true"</code>	- Enable DropShadow
<code>ThumbnailDropShadowSize="8"</code>	- The rate of blur in pixels
<code>ThumbnailDropShadowOffset="2"</code>	- Transparency of the shadow (0-100)

The settings of the Thumbnail view also concern the Preview.

CONTROLBAR

The ControlBar is enabled by default; it is visible and can be used. User should disable the function if it is not wanted:

`ControlBar="false"`

Read about the ControlBar customizing in *Appendix C. - Customizing the ControlBar*

COPYRIGHT MESSAGE

User can code the Copyright message with the XML Validator (see *Copyright Message setup in Appendix A*), and then User can customize it:

`CopyrightMessage="<encoded message >"`

If it was enabled previously in the XML Validator, User can disable the Copyright Message displaying:

`CopyrightMessageDisplay="false"`

FULLSCREEN

Enabling full screen view:

`FullScreenEnabled="true"`

Do not forget to enable it in the embed script in html:

`params.allowfullscreen = "true";`

3.5.1 DEFAULT FONT

User can modify the default font.

If the default Font is not set in the XML, the factory embedded font will be used.

Embedding custom font is only possible with the help of Adobe Flash CS5. *For further information, refer to Font Embedding in Appendix D.*

From this on, if no font attribute is set in the XML, the default font will be used. But User should only specify a value in a font attribute in case of custom embedded fonts.

3.5.2 POPUP AND TOOLTIP SETTINGS

The Popup and the ToolTip floating windows are basically the same, the only difference is that the ToolTip contains only a one-line text, and its width is adjusted to the length of the text.

<code>PopupBackgroundColor="0xFFFFFFFF"</code>	- Background color
<code>PopupBackgroundOpacity="80"</code>	- Transparency of the background (0-100)
<code>PopupBorderThickness="0"</code>	- Width of background border
<code>PopupBorderColor="0x000000"</code>	- Color of background border
<code>PopupRounded="true"</code>	- Enable rounded border
<code>PopupRoundedRadius="8"</code>	- Rate of rounded border
<code>PopupMargin="8"</code>	- Margin
<code>PopupTitleFont=""</code>	- Font of the title
<code>PopupTextFont=""</code>	- Font of the text
<code>PopupTitleColor="0x333333"</code>	- Color of the title
<code>PopupTextColor="0x333333"</code>	- Color of the text
<code>PopupSpace="8"</code>	- Space between image/title/text
<code>PopupMouseFollowSpeed="5"</code>	- Mouse follow speed

3.6 CUSTOMIZING THE CONTROLBAR

The ControlBar can be enabled/disabled. Enable it when customizing:

`ControlBar="true"`

`ControlBarHeight="55"`

- Height of the ControlBar area. Doesn't affects the size of the control bar icons.

ControlBar position can be vertically modified:

`ControlBarOffset="15"`

- Rate of downward offset in pixels

Further general settings:

`ControlBarScale="100"`

- Size of the ControlBar (%)

`ControlBarFullScreenScale="150"`

- Full screen size (%)

`ControlBarBackgroundColor="0x555555"`

- Background color

`ControlBarBackgroundAlpha="50"`

- Transparency of background (0-100)

If no color is specified, the ControlBar does not have a background.

Enabling the LoaderBar:

`ControlBarLoaderEnabled="true"`

Navigation buttons can be enabled/disabled based on their types:

`ButtonFirstLastEnabled="true"`

- First/Last page buttons

`ButtonLeftRightEnabled="true"`

- Previous/Next page buttons

`ButtonZoomEnabled="true"`

- Zoom buttons

`ButtonPrintEnabled="true"`

- Print buttons

Extra buttons can also be enabled/disabled:

`ButtonAutoFlipEnabled="false"`

- Toggle Autoflip button

`ButtonPDFLinkEnabled="false"`

- Download button

`ButtonLanguageEnabled="true"`

- Menu Language button

`ButtonThumbnailEnabled="true"`

- Thumbnail view button

`ButtonMuteEnabled="true"`

- Mute button

`ButtonInfoEnabled="true"`

- Info button

`ButtonIndexEnabled="true"`

- Table of Contents button

`ButtonFullScreenEnabled="true"`

- Full Screen button

`ButtonCloseEnabled="true"`

- Close button

Colors of the ControlBar:

`EnableButtonColoring="true"`

- Enable button coloring

`ButtonColor="0x666666"`

- Default color and transparency

`ButtonAlpha="100"`

`ButtonOverColor="0xF0B400"`

- MouseOver color and transparency

`ButtonOverAlpha="100"`

<code>ButtonPressColor="0x666666"</code>	- Press color and transparency
<code>ButtonPressAlpha="50"</code>	
<code>ButtonDisabledColor="0x666666"</code>	- Inactive color and transparency
<code>ButtonDisabledAlpha="15"</code>	

For information on customizing the graphic elements of the buttons, and/or the layout of the controlbar buttons, refer to Appendix C.

3.7 CUSTOMIZING THE BOOK

The Book settings determine the layout and the handling of the Book, and the layout of the pages. Either they directly affect the Book, or they concern all pages of the Book.

In order of significance:

(This section includes several attributes already explained earlier)

<code>PageWidth="300"</code>	- Width of the page
<code>PageHeight="400"</code>	- Height of the page
<code>AutoSize="true"</code>	- Automatic resizing of the content loaded to fill the Page
<code>PageScale="true"</code>	- Automatic scaling of the Book
<code>MinScale="50"</code>	- Minimum and maximum scaling in percent
<code>MaxScale="200"</code>	
<code>MinStageWidth="700"</code>	- Minimum <i>Screen</i> size
<code>MinStageHeight="620"</code>	
<code>ViewAngle="0"</code>	- View angle (-89 - 89)
<code>StartPage="1"</code>	- Start at the specified page
<code>AlwaysOpened="false"</code>	- Book always open Book
<code>ZeroPage="false"</code>	- 0. page content
<code>ZeroPageAlwaysVisible="false"</code>	- 0. page content always visible
<code>CenterSinglePage="false"</code>	- Single page in center
<code>RightToLeft="false"</code>	- Book written right to left
<code>VerticalMode="false"</code>	- Vertical flip mode
<code>OffsetX="0"</code>	- Offset of Book position
<code>OffsetY="0"</code>	
<code>HardCover="true"</code>	- Hard cover
<code>HardPage="false"</code>	- Hard pages
<code>EmbossedPages="true"</code>	- Embossing of pages with light-shadow effect (disabled for hard pages)
<code>LargeCover="true"</code>	- Larger cover
<code>LargeCoverVerticalOversize="32"</code>	- Vertical size difference
<code>LargeCoverHorizontalOversize="16"</code>	- Horizontal size difference
<code>DropShadow="true"</code>	- Enable DropShadow for Book
<code>DropShadowSize="20"</code>	- Rate of DropShadow in pixels

<code>DropShadowOffset="10"</code>	- Vertical offset of the DropShadow in pixels
<code>DropShadowAlpha="30"</code>	- Transparency of the shadow (0-100)
<code>TransparencyEnabled="true"</code>	- Enable transparency
<code>TransparencyAutoLevel="true"</code>	- Automatic handling of transparency levels
<code>TransparencyMaxLevel="16"</code>	- Maximum level of pages transparency
<code>FlippingSpreadAnim="true"</code>	- Enable animations on a Spread during flipping
<code>PageCache="5"</code>	- Number of preloaded/cached pages per direction
<code>UnloadPages="false"</code>	- Deleting pages from memory when not needed
<code>StartAutoFlip="false"</code>	- Starting automatic flipping after loading of <i>Page-Flip4</i>
<code>AutoFlipDefaultInterval="5"</code>	- Default flip interval in seconds
<code>AutoFlipLooping="true"</code>	- Flip looping at the end of the Book
<code>FPSControlEnabled="true"</code>	- Handling PageFlip frame rate
<code>ShowFPS="false"</code>	- Showing frame rate
<code>MaxFPS="60"</code>	- Set maximum frame rate

If the animations of the PageFlip consume too much CPU resources, decrease the frame rate down to 25-30 fps.

3.7.1 CUSTOMIZING SOUNDS

User has to enable Sounds, so that flipping is accompanied by sound effects (Settings node attribute):

```
SoundEnabled="true"
```

User can specify that PageFlip starts muted when loaded, which can be toggled later with a controlbar button:

```
StartMute="true"
```

If sounds are enabled, PageFlip will search for the list of sound files in the Sounds node:

```
<Sounds EffectVolume="10" >
  <HardPageClose SoundFile="sounds/sound1.mp3" />
  <PullOpenSoundFile="sounds/sound2.mp3" />
  <PullCloseSoundFile="sounds/sound3.mp3" />
  <PushOpenSoundFile="sounds/sound4.mp3" />
  <PushCloseSoundFile="sounds/sound5.mp3" />
</Sounds>
```

The global volume of the sound effects can be set in the Sounds/EffectVolume attribute from 0 to 100.

The various sound types are specified in separate nodes, as Sounds node child-nodes. User can specify more than one sound clips from each type of sound, so PageFlip can choose sounds randomly from the set.

The following 5 sound types exist:

HardPageClose	- Closing the Hard Cover
PullOpen	- Start of flip, type 1
PullClose	- End of flip, type 1
PushOpen	- Start of flip, type 2
PushClose	- End of flip, type 2

There are two types of flipping (Pull, when the page is “pulled”, and Push, when the page is “pushed”), to which we can assign different sounds.

3.7.2 ADDING AND CUSTOMIZING THE BACKGROUND MUSIC

We can customize Background music in the BackgroundMusic/MusicData node:

```
<BackgroundMusic>
  <MusicData StreamURL="BGM/mainBGM.mp3" Loop="999" Volume="40" />
</BackgroundMusic>
```

Attributes of the MusicData node:

StreamURL	- URL of the sound file to be played
Loop	- number of repeats
Volume	- volume level of playback (0-100)

The StartMute option and the mute button also apply to the background music. This Background music can “overwritten” at page level, see *section 3.10 Pages & Spreads*.

3.8 ORDER OF PAGES (PAGEORDER)

The order of Pages determines the visible content of the Book. User has to specify each page in the `<PageData>` nodes that load the book from the first to the last page.

Depending on the `AlwaysOpened` and the `ZeroPage` settings, the Book starts from page 0 or 1. The following figure shows the connection between the spreads and the possible settings (see *Figure 3.8*).

If `AlwaysOpened` is enabled, the Book starts open, and User cannot flip back to page 1. The first two pages defined in the `<PageData>` nodes will be the first two pages of the open book, laid out as spreads.

On the other hand, if our Book starts closed with page 1, then the first `<PageData>` node contains the single Cover page, and the next two will be the first spread in the Book.

If `ZeroPage` is enabled, then a Page-file is loaded to the empty area before the Cover, which looks as if blending in the background.

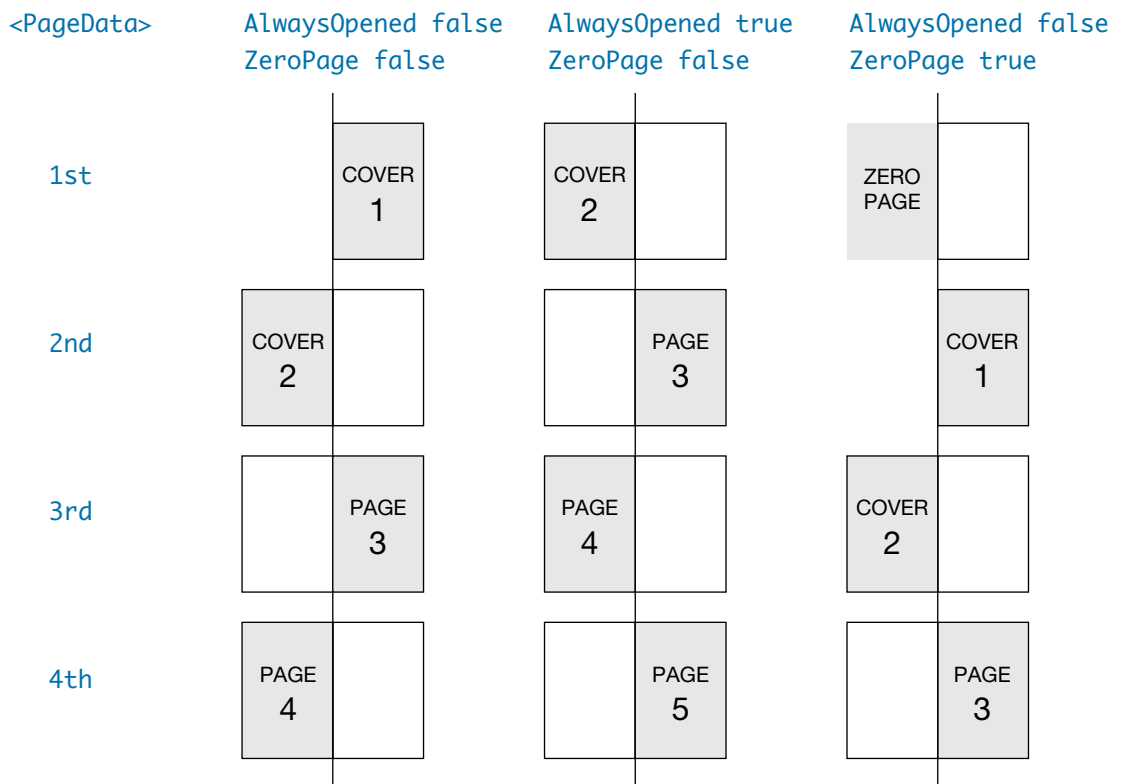


Figure 3.8

Always keep in mind spreads when defining pages. If defining a spread, it can be done only as the 1st page of the current spread; otherwise the Book cannot display it.

3.9 PREPARING THE GRAPHICS CONTENT

Save the files in the original structure, or in the same folder as pageFlip.swf, or one of its subfolder.

3.9.1 PAGE-FILES

Prepare the content of the pages now that requirements are known.

The size of the Book pages is clear, now User can create all the pages. The following file formats are supported:

- JPG, JPEG** – The best choice if no animations, interactions more complex than HotSpots, or transparency is used. It is the optimal choice from the aspects of rendering speed and file size. Flash Player cannot handle Progressive JPG.
(Tip: When saving JPG, do not save Preview in the file, use sRGB color profile, and at least 7/12 compression quality.)
- PNG** – It has similar characteristics as JPG, but the image quality is better (do to the lossless compression) and it also knows transparency (up to 8 bit alpha channel), but the file size is usually larger.
- GIF** – The system can load them, except for the AnimGIF, but it is worse in file size and quality than the other formats. Its use is not recommended, only if on purpose.
- SWF** – Use the Flash format when it is necessary for interaction or animation, or if User wants text selection on the page. It is important to know that too much text on the page can result in slower rendering.

Where User wants to use Spreads, the specified width has to be doubled, or the height where vertical mode is enabled (`Settings/VerticalMode="true"`).

Create all the Page-files in the format most suitable, in the size already specified in the XML. If LargeCover is enabled, do not forget to adjust their sizes!

3.9.2 THUMBNAIL FILES

Thumbnail view images files has to be created based on the sizes given in the `Settings/ThumbnailWidth` and `Settings/ThumbnailHeight` attributes (*PageFlip4* scales images with other sizes to the specified size, but for optimal quality, previews should be created in the correct size). In case preview is created to a Spread, User should double the correct size.

3.9.3 ZOOM FILES

User can zoom in two ways:

- Use the SWF file loaded on the page as source, as it is evident in case of vector-based content, and other file does not have to be loaded again, while the quality will be fine.
- User can load JPG/PNG/GIF file or files which PageFlip will display 1:1. The rate of zooming depends on the resolution of the source images.

In the second case, User enables zoom for every page, for which at least one scaled version of the page has to be created. If User wants multiple level of zooming on the page, the required number of versions is needed. User is recommended to refer to the % of scaling in the name of the file. The rate of the sides of the original and the enlarged pages' has to be the same; otherwise PageFlip will transform the page. User should also pay attention to the aspect-ratio of the page sides in case of Spreads.

3.9.4 BACKGROUND IMAGE

The resolution of the background image should be at least the resolution of the *Screen* by default; otherwise the blurred pixels due to the scaling can corrupt the overall look.

If the size of the *Screen* is smaller than that of the background image, it is displayed 1:1 (with no size reduction).

If the background image is smaller than the *Screen*, PageFlip will scale it, so it will fill the *Screen* keeping the original aspect-ratio.

3.9.5 OTHER CONTENTS

Images assigned to Popup windows, or LightBox gallery images require thorough preparation. (*refer to the Appendix J. Use of LightBox*)

3.10 PAGES & SPREADS

Definition of the pages is in the XML configuration's `<PageOrder>` node. All pages are described by a `<PageData>` node. A simple example:

```
<PageOrder>
  <PageData PageFile="pages/demo_largepage_316x432.png" />
  <PageData PageFile="pages/demo_largepage_316x432.png" />
  <PageData PageFile="pages/demopage_script1.swf" />
  <PageData PageFile="pages/demo_page4.png" />
</PageOrder>
```

The definition of Spreads is similar to that of single pages, but an attribute indicates that it is a spread, while User should also remember that a Spread is only one definition.

The following is a list of `<PageData>` attributes with value examples:

<code>Spread</code>	Its default value is <code>"false"</code> . It should only be set if defining a Spread: <code>Spread="true"</code>
<code>PageFile</code>	URL of the Page-file (name of the file with its path). It is a required attribute without which there is no content will be loaded on the given page: <code>PageFile="pages/pagefile.jpg"</code> or <code>PageFile="pages/pagefile.swf"</code>
<code>SWFPageFileAsZoom</code>	The use of SWF Page-file for the zoom view. Its default value is <code>"false"</code> . It should only be set if enabling the option: <code>SWFPageFileAsZoom="true"</code>
<code>ZoomFiles</code>	There are two methods for the User to load images to the zoom view of the page: - In case of a single zoom image to the page: <code>ZoomFiles="pages/zoom/zoomfile.jpg"</code> - More images can be loaded to the same page if User wants to use multiple zooming levels. The example shows three images in 200, 300 and 400 % zoom: <code>ZoomFiles="pages/zoom/zoomfile200.jpg[200%]pages/zoom/zoomfile300.jpg[300%]pages/zoom/zoomfile400.jpg[400%]"</code>
<code>ThumbnailFile</code>	They can only be image files (JPG, PNG, GIF). This image represents the page in the Preview or the Thumbnail view. <code>ThumbnailFile="pages/thumbs/thumbnailfile.jpg"</code>
<code>PrintFile</code>	User can enable printing of the page by specifying a print file that PageFlip loads and prints: <code>PrintFile="pages/print/printfile.jpg"</code> or <code>PrintFile="pages/print/printfile.swf"</code> Pages can only be printed separately.

HotSpotFile	User can assign HotSpots to all pages. <i>For further information on HotSpots, refer to section 3.11 Use of HotSpots.</i> <code>HotSpotFile="pages/hotspots/pagehotspotfile.xml"</code>
PageIsTransparent	If transparent page is created showing the page behind, User has to enable transparency, the default value is "false": <code>PageIsTransparent="true"</code>
HardPage	This attribute can override the Book configuration with similar name, concerning the whole book. <code>HardPage="true"</code>
BackgroundMusicFile	User can assign music files to each page, which <i>PageFlip4</i> starts to play when the given page is opened, and keeps it playing until User flips to another page. <code>BackgroundMusicFile="pages/bgm/pagebgm.mp3"</code>
BackgroundMusicLoop	If User wants to loop, or repeat the music file several times, the number of repeats can be specified by the following way: <code>BackgroundMusicLoop="100"</code>
PageNumber	User can customize the page numbering on any of the pages, and the numbering will continue on the next page in accordance: <code>PageNumber="6"</code>
PageLabel	Each page can have a Label, which User can refer to later, for example in the Table of Contents, or when creating HotSpots. It provides more flexibility than the use of page numbers, as it allows the reordering of the Pages. <code>PageLabel="YourLabel"</code>
PageWidth, PageHeight	Sometimes the automatic scaling of the Pages is not correct, or User wants to publish content on the page that cannot fit the frame. If User loads SWF file in which the content exceeds the page border (for example, invisible part of the text box, or the bleed of a graphics element), the size data of the Page within the Flash can differ from the specified data, so the User can modify them in the XML. Real width, height of the page and offset of page content horizontally, vertically: <code>PageWidth="300"</code> <code>PageHeight="400"</code> <code>PageOffsetX=""</code> <code>PageOffsetY=""</code>
BackgroundColor	If the default page-background color does not fit the page, User can modify the color: <code>BackgroundColor="0x000000"</code>
AutoFlipInterval	Set the Auto Flip Interval at page level in milliseconds. Default it is unset, the value defined in settings is used. <code>AutoFlipInterval="5000"</code>

3.11 USE OF HOTSPOTS

User can create HotSpots on a page with XML configurations. HotSpots are areas on the Page that can open Popup windows if the mouse hovers over it, and can send User to other pages, open new links, or a LightBox image, executes custom scripts, etc.

User can assign HotSpot XML files to each Page. Any number of HotSpots can be defined in each HotSpot XML file.

The HotSpot XML file structure is as follows:

```
<HotSpotData>
  <HotSpot>
    <Rectangle></Rectangle>
  </HotSpot>
  <HotSpot>
    <RoundRectangle></RoundRectangle>
  </HotSpot>
</HotSpotData>
```

Each HotSpot is described in a `<HotSpot>` node. Here, User can specify the content of the Popup if necessary, and what happens if clicked on. Moreover, there can be an optional number of shape definitions within this HotSpot node. These shapes are the areas on the Page that activate when mouse hovers them. The same HotSpot can contains multiple shape definitions

.

- Attributes of the HotSpot node

User has to specify at least one of the following three attributes to have a Popup window if mouse hovers over that HotSpot:

```
PopupTitle="Title"
PopupText="MultilineText"
PopupImageFile="page/popupimages/popupimage.jpg"
```

Example, with full HotSpot XML:

```
<HotSpotData>
  <HotSpot  PopupTitle="Title"
           PopupText="MultilineText"
           PopupImageFile="page/popupimages/popupimage.jpg" >
    <Rectangle>0,0,300,50</Rectangle>
  </HotSpot>
</HotSpotData>
```

- HotSpot Shapes

User can specify the location of the HotSpots (one or more) by defining the following shapes (The coordinates align to the specified page size). These nodes are sub-nodes of the `<HotSpot>` node as seen in the previous code example.

Rectangle

Indicates a rectangle:

```
<Rectangle>x,y,w,h</Rectangle>
```

Where:

x and y – x and y coordinates of the upper left corner of the rectangle

w and h – width and height of the rectangle.

RoundRectangle

Indicates a rounded rectangle:

```
<RoundRectangle>x,y,w,h,r</RoundRectangle>
```

Where:

x and y - x and y coordinates of the upper left corner of the rectangle

w and h - width and height of the rectangle.

r – radius of the rounding

Circle

Indicates a circle:

```
<Circle>x,y,r</Circle>
```

Where:

x and y - x and y coordinates of center of the circle

r – radius of the circle

Star

Indicates a star with optional number of arms:

```
<Star>x,y,p,r1,r2,phase</Star>
```

Where:

x and y - x and y coordinates of center of the star

p – number of tips

r1,r1 – radius of the tips and pits

phase – rate of rotation between 0-1, where 1 is a turn of one tip (in case of a 5-armed star $360/5 = 72$ degrees)

Shape

Indicates an optional shape with each point defined by User (the last point is automatically connected to the first one). Example:

```
<Shape>x1,y1/x2,y2/x3,y3/x4,y4</Shape>
```

The shape consists of 4 points in the example, the last point is automatically connected to the first to close the shape.

The shape of the HotSpot can be invisible, or it can have any color and transparency. User can define these with the following attributes of the `<HotSpot>` node:

<code>Color="0xFFFF00"</code>	- Color of the shape
<code>Opacity="20"</code>	- Rate of transparency (0-100)
<code>HoverOpacity="50"</code>	- Rate of transparency when mouse is over the HotSpot

- Events on clicking

User can choose from the following events to happen when clicking on a HotSpot. All of these are attributes of the `<HotSpot>` node like `PopupTitle`.

Jump to a Page based on a Label

The use of Labels is the most preferable reference mode. User can specify Labels for each page, which can be referred to in HotSpots. Attribute `Skip` determines that jumping to the page is direct (`"true"`), or flipping page-by-page (`"false"`). Example:

```
TargetLabel="PageLabel" Skip="true"
```

Jump to a Page by Page number

Page numbering – by default – starts with 1, and increases by one per page. If User modifies the page number on a page, `PageFlip` will reference to the modified page number. Information on the `Skip` attribute is given in section `Jump to a Page based on a Label`.

```
TargetPage="5" Skip="true"
```

Opening a URL in browser

URL, and target can be specified. Target can be `"_self"` as custom window, when the link opens in the `PageFlip` window, closing `PageFlip`, and `"_blank"`, when the specified URL opens in a new window (Tab). Example:

```
Link="http://pageflip.hu" Target="_blank"
```

Opening LightBox image

The following attributes can be used for images to be displayed in `LightBox` when clicking on the `HotSpot`:

```
LightBoxImage="pages/images/lightboximage.jpg"
```

```
LightBoxCaption="Caption for LightBox Image"
```

Opening LightBox image group

`LightBox Group`, with the required JavaScript configurations, can be opened with the following attributes:

```
LightBoxGroup="image1"
```

Where `"image1"` is the `id` of the image to show first in the group. An example how a group is defined in html:

```
<a id="image1" href="pageflipdata/pages/image1.png"
  rel="lightbox[Group1]" title="title1"></a>
<a id="image2" href="pageflipdata/pages/image2.png"
  rel="lightbox[Group1]" title="title2"></a>
<a id="image3" href="pageflipdata/pages/image3.png"
  rel="lightbox[Group1]" title="title3"></a>
```

Start/Stop/Toggle Autoflip*

User can control `Autoflip` feature with a `HotSpot` by adding the `AutoFlip` attribute and specifying one of the following values to it: `"start"`, `"stop"`, `"toggle"`. Example:

```
AutoFlip="start"
```

Executing custom functions

With this feature, by clicking on a HotSpot you can execute a function in a Page-file's script or in the External Script (*refer to the section Interactivity in Appendix B.*)

Custom function call in the page scripts can be set up by setting the following HotSpot attribute:

```
SWFfunction="test"
```

If this HotSpot is clicked, then the hotSpotCall function will be called in your Page-file script, and the String "test" will be sent to it:

```
function hotSpotCall( msg:String ):void {  
    if (msg=="test" ) {  
        // code here is executed  
    }  
}
```

The same way User can call the hotSpotCall function in the External Script using this HotSpot attribute:

```
ESfunction="test"
```

If this HotSpot is clicked, then the hotSpotCall function will be called in your External Script file. The String "test" and the internal page number of the Page containing that HotSpot will be sent to it:

```
function hotSpotCall( msg:String, pIPN:uint ):void {  
    if (msg=="test" ) {  
        trace("HotSpot clicked on page "+pIPN);  
        // code here is executed  
    }  
}
```

3.12 TABLE OF CONTENTS & INFO WINDOW

The structure of the two Floating Windows is very similar; the only difference is that the Info window does not contain links.

Structure of the Table of Contents XML:

```
<TableOfContents>
  <Title />
  <Description />
  <Link />
</TableOfContents>
```

Structure of the Info XML:

```
<Info>
  <Title />
  <Description />
</Info>
```

User can vary the number and order of the Title, Description, and Link nodes optionally (in case of Info, Link cannot be used),

- Title's attribute:

`Name="Table of Content"` - Text of the Title

- Description's attribute:

`Text="Description text"` - Text of the Description

- Link attributes:

`Name="Description text"` - Text of the Link
`Skip="true"` - Direct jump to page
`targetLabel="PageLabel"` - Jump to specified page by Label
`targetPage="5"` - Jump to specified page number

User can specify the configuration parameters of the Title, Description and Link nodes if they vary from the default:

`Font="FontName"` - Custom embedded Font
`Color="0x000000"` - Text color
`Size="12"` - Size
`Align="right"` - Align

User can use the following four alignments:

- left - Default value, align left
- right - Align right
- center - Align center
- justify - Justify

The length of the Table of Contents is not limited: if the content exceeds the size of the window, scrolling helps navigation in the window.

3.13 MENU LANGUAGE SETTINGS

User can specify the menu language in the `<Language>` node.

By default, the menu language is English, so User has to define the `<Language>` node only to enable other languages. If User enables more languages and wants to use English too, then it has to be defined in a `<Lang>` node in the XML.

The default values, and XML structure:

```
<Language>
  <Lang Name="English"
    Loading="LOADING"
    PagerPage="Page #"
    PagerThumbnailPopup="Page #"
    PagerThumbnails="Thumbnails"
    PagerThumbnailPages="Thumbs #"
    PagerZoomPage="Zoom Page #"

    PrintLeftTooltip="Print left side page"
    PrintRightTooltip="Print right side page"
    ZoomLeftTooltip="Zoom on left side page"
    ZoomRightTooltip="Zoom on right side page"
    ZoomInTooltip="Zoom In"
    ZoomOutTooltip="Zoom Out"
    FirstPageTooltip="First page"
    PreviousPageTooltip="Previous page"
    NextPageTooltip="Next page"
    LastPageTooltip="Last page"

    AutoFlipTooltip="Autoflip On/Off (A)"
    MuteTooltip="Mute (M)"
    InfoTooltip="About (Shift-I)"
    IndexTooltip="Table Of Contents (I)"
    LanguageTooltip="English"
    ThumbnailTooltip="Thumbnail View (T)"
    FullscreenTooltip="Fullscreen On/Off (Shift-F)"
    CloseTooltip="Quit PageFlip (Shift-Q)"
    PDFDownloadTooltip="Download PDF version"
  />
</Language>
```

To be able to use more than one language, User should create new `<Lang>` nodes within the `<Language>` node, and enable the Language switch button on the ControlBar (`ButtonLanguageEnabled="true"`).

(hash mark) within quotation marks indicates the page numbers to be inserted later to the given text. Example, book is opened at page 10:

```
PagerPage="Page #"      results: Page 10–11
PagerPage="#. oldal"    results: 10–11. oldal
```

3.14 PUBLICATION OF THE CREATED PAGEFLIP

If the created Book is successfully tested, User can save it to its final folder, which can be located on a web server, or offline.

If User followed the recommended file structure, the following should be copied to the final folder:

index.html	(the html file that embeds pageFlip.swf)
pageFlip.swf	
pageflipdata/	(folder with all the external files for PageFlip)
css/	(folder, html and LightBox styling)
js/	(folder, LightBox and SWFobject JavaScript library)

All of these files are inside the original obj/ folder.

APPENDIX

A. COPY PROTECTION

The *PageFlip4* protection prevents unauthorized use - copying in similar format, or reconfigured as a new publication - of the Book created and published by the License owner.

User can protect the publication by entering a valid code sequence, generated by the XML Validator, in the CopyrightMessage in the XML file. The true function of the CopyrightMessage is the configuration of the Copyright message displayed at the bottom of the *PageFlip4 Screen*, where License owner (publisher) can display personal data and contact information (the display of this line can be enabled/disabled with a valid code).

B. INTERACTIVITY

This section includes all the available functions and variables that the loaded SWF Page codes can access.

In order to control SWF page-file by PageFlip (or control PageFlip by an SWF Page-file) User have to register the page-file with the following short script (Copy the following code to the beginning of first frame's actionscript of that page-file):

```
import flash.display.MovieClip;
import flash.events.Event;

var Container:MovieClip;
addEventListener( Event.ADDED_TO_STAGE, onAdded );

function onAdded( e:Event ):void {
    Container = MovieClip(parent.parent.parent.parent);
    Container.registerPage( this );
    removeEventListener( Event.ADDED_TO_STAGE, onAdded );
}

function onAppear():void {}
function onDisappear():void {}
function onTop():void {}
function onNotOnTop():void {}
function onFlipEnd():void {}
function onFlipStart():void {}
function onPageUnload():void {}
function hotSpotCall( msg:String ):void {}
```

User can access *PageFlip4* services through the Container variable.

The first one is the registration of the Page, by which the User's Page informs *Page-Flip4* that the code running on the Page is able to communicate (This function call is

done automatically by the `ADDED_TO_STAGE` event):

```
Container.registerPage( this );
```

It is enough for User to define the event handler function to be used. This is a short description of the events, when they are executed, and what the code should do:

onAppear

when a Page is displayed on the *Screen* (even if covered)

If User wants an event (for example animation) to start on the page already when it is not completely visible (during flipping to the Page, or partial overlay of a transparent page), then it has to be authorized and launched here.

onDisappear

when a Page is removed from the *Screen*.

If User wants to keep the activity until the disappearance of the Page, then activity on the Page has to be disabled here.

If User works with a great number of coded pages in the book, it is very important that invisible pages do not overload the CPU.

onTop

when the page is completely visible, without overlay (it is the selected page). Called only if there is no flipping.

Activate the content only on the completely visible page after flipping is over. For example buttons, their event handlers, etc:

```
function onTop():void {  
    myButton.enabled = true;  
}
```

onNotOnTop

when the Page loses its `onTop` status, that is, it is overlaid due to a flip. Disable the activities enabled by `onTop`:

```
function onNotOnTop():void {  
    myButton.enabled = false;  
}
```

onFlipEnd

when the flip of the given Page is over. Both pages that was part of the flip animation receives this event.

onFlipStart

when the flip of the given Page starts. Both pages that makes part of the flip animation receives this event.

onPageUnload

when PageFlip removes the page from the memory.

User has to delete everything the code created during run-time (Display Objects, Events, etc.), so that Page content can be removed from the memory safely.

hotSpotCall

when a Hotspot is set to execute custom function on a page. Each Hotspot can send a unique message to this event (`msg:String`)

The Page code can access the following variables:

<code>Container.OnLeftSide: Boolean;</code>	- true, if the Page is on the left side of the Book
<code>Container.IPN: uint;</code>	- Internal Page Number of the Page
<code>Container.PageData: PFPageData;</code>	- All data/information about the Page
<code>Container.isReady: Boolean;</code>	- Page has been registered
<code>Container.SWF: Object;</code>	- Root of the loaded Page-file
<code>Container.Pair: PFPageContainer;</code>	- Container of the Page pair

For communication with the Spread, make sure that it is ready. The following example invites the `body()` function of the spread:

```
if(Container.Pair.isReady) {  
    Container.Pair.SWF.test();  
}
```

PageData Object data (quick overview):

<code>isCover: Boolean;</code>	- if given Page is a Cover
<code>renderWidth: Number;</code>	- Rendered size (without scaling the Book)
<code>renderHeight: Number;</code>	

The following variable values are all from the XML `<PageData>` node (these are Read-Only datas):

<code>PageName: String;</code>	- The name of the Page
<code>PageNumber: uint;</code>	- The displayed number of the Page
<code>PageLabel: String;</code>	- The Label of the Page
<code>PageWidth: uint;</code>	- Width and Height of the Page
<code>PageHeight: uint;</code>	
<code>PageOffsetX: int;</code>	- The display Offsets of the Page
<code>PageOffsetY: int;</code>	
<code>PageIsTransparent: Boolean;</code>	- if true the Page behind is also rendered
<code>HardPage: Boolean;</code>	- if true the Page will animate like a hard cover
<code>DisableEmbossing: Boolean;</code>	- if true, no shadows/highlights are applied
<code>BackgroundColor: uint;</code>	- The background color of the Page while loading
<code>PageFile: String;</code>	- The relative URL of the page
<code>SWFPageFileAsZoom: Boolean;</code>	- If true, the same swf Page-file is used for zoom
<code>ZoomFiles: String;</code>	- The relative URL of the Zoom image files
<code>ThumbnailFile: String;</code>	- The relative URL of the Thumbnail file
<code>PrintFile: String;</code>	- The relative URL of the Print file
<code>HotSpotFile: String;</code>	- The relative URL of the HotSpot XML file
<code>BackgroundMusicFile: String;</code>	- The relative URL of the streamed sound file
<code>BackgroundMusicLoop: uint;</code>	- The number of times the sound file is repeated
<code>Volume: Number;</code>	- Volume of the sound playback
<code>MainBGMVolume: Number;</code>	- The sound of the main Background Music
<code>Spread: Boolean;</code>	- true if the Page is a Spread (double page)

Before describing the available *PageFlip4* APIs, let's talk about ExternalScript, which will also use the same APIs than the page-files, and in addition of that ExternalScript and the Page-files can communicate with each other directly.

EXTERNALSCRIPT

PageFlip4 can load an external script file (AS3 swf file) which is created by the User, and the scripts in that file can call *PageFlip4* APIs, and also receives main flipping events. A template script file (obj/pageflipdata/externalScripts.swf) is part of the *PageFlip4* package, which source file (src/externalScripts fla) is also included.

To load an external script, just set its URL in the config XML's settings node:

```
ExternalScript="externalScripts.swf"
```

An empty externalscripts file for example, with automatic registering, and event handlers defined, like in the previously discussed page-file script:

```
import flash.display.MovieClip;
import flash.events.Event;

var Container:MovieClip;
addEventListener( Event.ADDED_TO_STAGE, onAdded );
function onAdded( e:Event ):void {
    Container = MovieClip(parent.parent);
    Container.registerExternalScript( this );
    removeEventListener( Event.ADDED_TO_STAGE, onAdded );
}

// Event handlers
function onAppear          ( pIPN:uint ):void {}
function onDisappear      ( pIPN:uint ):void {}
function onFlipEnd        ( pIPN:uint ):void {}
function onFlipStart      ( pIPN:uint ):void {}
function onTop            ( pIPN:uint ):void {}
function onNotOnTop       ( pIPN:uint ):void {}
function onPageUnload     ( pIPN:uint ):void {}

function onRegister       ():void {}
function onPageRegister( pIPN:uint, SWF:Object=null ):void {}
function hotSpotCall( msg:String, pIPN:uint ):void {}

// Custom button functions
function onCustomButtonExecute( pID:uint, pButton:MovieClip ):void {}
```

In this code example, there is four additional event handlers that is not available for the page-file scripts:

onRegister

when the external script file has successfully registered (this event is recieved only once, just after `Container.registerExternalScript(this);` is executed.

onPageRegister

when a Page is registered, the extrnal script also knows that. When this event is recieved, the internal page number (`pIPN:uint`), and a pointer (`SWF:Object`) to the registered page is also recieved.

hotSpotCall

when a Hotspot is set to execute custom script. Each Hotspot can send a unique message to this event (`msg:String`), and the internal page number is also recieved (`pIPN:uint`).

onCustomButtonExecute

when a custom button is pressed on the Screen. The serial of the custom button is also recieved: `pID:uint` (0-7)

By using these event handlers, User can set up a various custom functions depending on what's happening in *PageFlip4*.

PAGEFLIP4 API

PageFlip4 can be controlled by the following functions (each of them can be called from a Page script, or External Script):

```
gotoPage( pIPN:uint, pSkip:Boolean ):Boolean
gotoPageNumber( pPageNumber:String, pSkip:Boolean ):Boolean
gotoPageName( pPageName:String, pSkip:Boolean ):Boolean
gotoPageLabel( pPageLabel:String, pSkip:Boolean ):Boolean
openURL( pURL:String, pTargetWindow:String ):void
openLightBoxImage( pURL:String, pCaption:String ):void
openLightBoxGroup( pGroupName:String ):void
setMainBG( pBGImage:String, pBGColor:int ):void
resetMainBG():void
```

Only External Script can access the following functions (and an Array):

```
gotoFirst():void
gotoPrev():void
gotoNext():void
gotoLast():void
zoomLeft():void
zoomRight():void
toggleMute():void
toggleFullscreen():void
toggleThumbnailView():void
closeWindow():void
toggleAutoFlip():void
kill():void
getIPN():uint
getNumberByIPN( pIPN:uint ):uint
getNameByIPN( pIPN:uint ):String
getLabelByIPN( pIPN:uint ):String
getIPNByNumber( pPageNumber:uint ):int
getIPNByName( pPageName:String ):int
getIPNByLabel( pPageLabel:String ):int
makeAutoPreflip():void
page:Array
pageScript( pIPN:uint ):Object
setMouseControl( pMouseControl:Boolean ):void
```

Only Page Script can access the following function:

```
showZoom():void
```

Access External Script from a page by calling:

```
Container.ES.myExternalScriptFunction();
```

and access a Page Script from the External Script (where *pIPN* is the internal page number):

```
Container.pageScript( pIPN ).myPageScriptFunction();
```

C. CUSTOMIZING THE CONTROLBAR

PageFlip4 allows User to customize the buttons of the ControlBar by loading an external swf file. The buttons can be modified easily with the use of the `pageFlipControls.fla` given in the Package. User has to configure two attributes in the XML Settings node:

`CustomControlBarIcons="true"` - Enabling custom buttons
`CustomControlBarFile="pageFlipControls.swf"` - File containing custom buttons

To create custom Button graphics, User have to edit the included `pageflipcontrols.fla` file with Adobe Flash CS5.

PAGEFLIPCONTROLS.FLA FILE MODIFICATION:

The Customizable UI Elements folder in the library contains the elements that can be modified.

The first group is the Buttons with timeline, with several layers in certain cases (the structure of which cannot be modified):

`controlBarButton_LeftHorizontal`
`controlBarButton_LeftMostHorizontal`
`controlBarButton_RightHorizontal`
`controlBarButton_RightMostHorizontal`
`controlBarButton_Zoom`
`controlBarButton_Print`
`controlBarButton_AutoFlip`
`controlBarButton_Close`
`controlBarButton_FullScreen`
`controlBarButton_Index`
`controlBarButton_Info`
`controlBarButton_Language`
`controlBarButton_Mute`
`controlBarButton_PDFDownload`
`controlBarButton_Thumbnails`

The second group contains the purely graphic elements that make up the visible parts of the buttons:

(26 x 26 pixel buttons)

`Symbol_LeftHorizontal`
`Symbol_LeftMostHorizontal`
`Symbol_Zoom`
`Symbol_Zoom_In`
`Symbol_Zoom_Out`
`Symbol_Print`

(22 x 26 pixel buttons)

`Symbol_AutoFlip_Pause`
`Symbol_AutoFlip_Play`
`Symbol_Close`
`Symbol_Download`

Symbol_Fullscreen_Off
Symbol_Index
Symbol_Info
Symbol_Language
Symbol_Mute_AudioOff
Symbol_Mute_AudioOn
Symbol_Thumbnails

With the reconstruction of the Symbol elements, User can create the custom buttons. User has to generate the swf file, and save it to the right folder (by default it is the pageflipdata/ folder).

CUSTOM CONTROLBAR BUTTON LAYOUT:

User can also design a custom button layout, which means each buttons can be placed anywhere on the Screen, and an additional 8 custom buttons can be added to the screen which function can be defined in the External Script.

User can define the button graphics in pageFlipControls.fla, just like as it is described in the previous section

To enable Custom Button Layout set the following attribute in <settings> to true:

```
CustomControlBarLayout="true"
```

Default buttons keeps their place, until one is overwritten in the <CustomControlBarLayout> node. For example:

```
<CustomControlBarLayout>  
  <Button ID="0" Name="LOADERBAR"  
    Width="200" Height="26"  
    HitAreaOffsetX="0" HitAreaOffsetY="0"  
    HitAreaWidth="26" HitAreaHeight="26"  
    CustomX="-80" CustomY="0" Rotation="0"  
    OnTop="false" Align="BOTTOM-RIGHT" />  
  <Button ID="1" Name="LEFTPRINT"  
    Width="26" Height="26"  
    HitAreaOffsetX="0" HitAreaOffsetY="0"  
    HitAreaWidth="26" HitAreaHeight="26"  
    CustomX="-52" CustomY="0" Rotation="0"  
    OnTop="true" Align="TOP-CENTER" />  
</CustomControlBarLayout>
```

Description of the attributes:

ID and the Name are fixed (name is only used to identify the button by the User, it is not used by the script)

Width/Height are the size of the button (usually the clickable size, but it can be overwritten, see below)

HitAreaOffsetX/Y the position of the Hit-rectangle relative to the buttons top-left cor-

ner. Set only if values are not 0.

HitAreaWidth/Height set only if different from the buttons size set in Width/Height.

CustomX/Y are the coordinates of the button on the screen (depends on alignment)

Rotation set rotation for your button if needed

OnTop set true if the buttons have to be over the Book.

Align set the reference point of the buttons position:

alignment is set as follows: it is composed by verticalAlignment-horizontalAlignment, they can be: TOP, MIDDLE, BOTTOM and LEFT, CENTER, RIGHT. examples:

`Align="TOP-LEFT"`

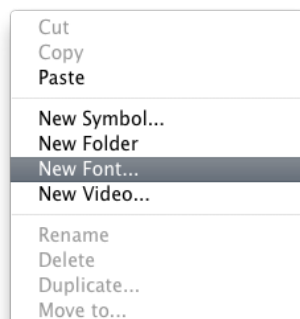
`Align="MIDDLE-RIGHT"`

User can make place for the buttons with the `BookMarginVertical` and `BookMarginHorizontal` attributes in config XML's <settings> node. It is applied to the book, and also to the thumbnails.

D. FONT EMBEDDING

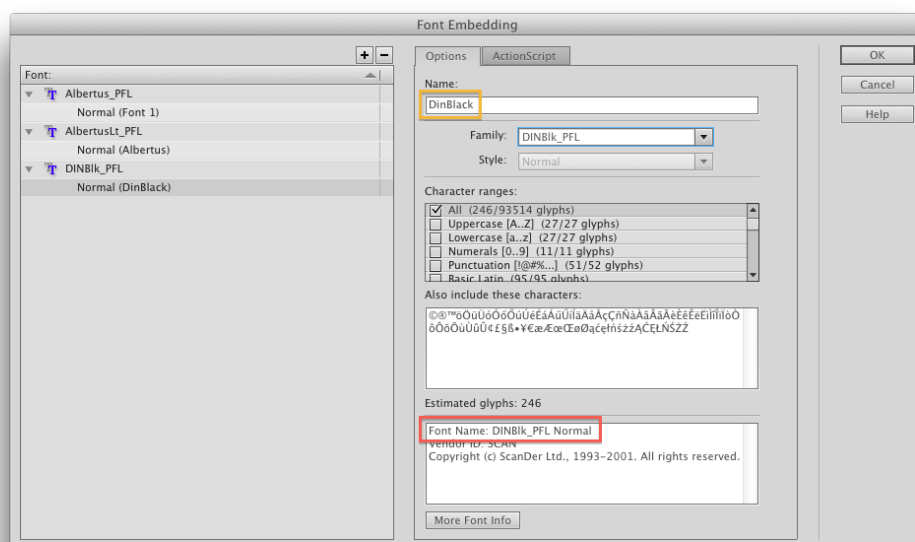
User can embed fonts with Adobe Flash CS5, by the following way:

- 1 - Open a new ActionScript3 FLA document
- 2 - Right click on the list of the Library window, then select New Font



(Figure D.1) Adding New Font to the Library

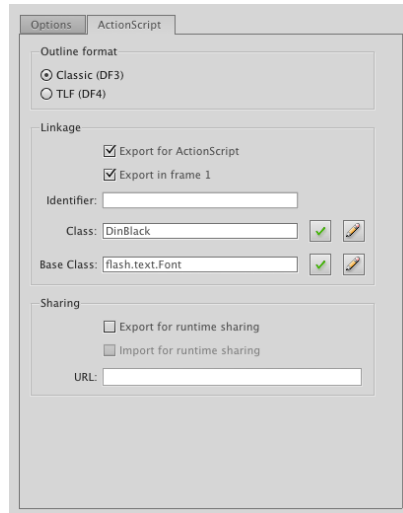
- 3 - The Options Tab is active (Figure D.1)



(Figure D.1) The Option Tab

- 4 - Add a custom name (Figure D.1 - highlighted with orange rectangle)
- 5 - Choose the Font from the Family drop-down menu
- 6 - Choose a Style if possible
- 7 - Set the Character Range so it contains all the necessary special characters too
- 8 - Note down the name of the Font following "Font Name: ", as it should be referenced in the XML (Figure D.1 - highlighted with red rectangle).

9 - Choose the Actionscript Tab (Figure D.2)



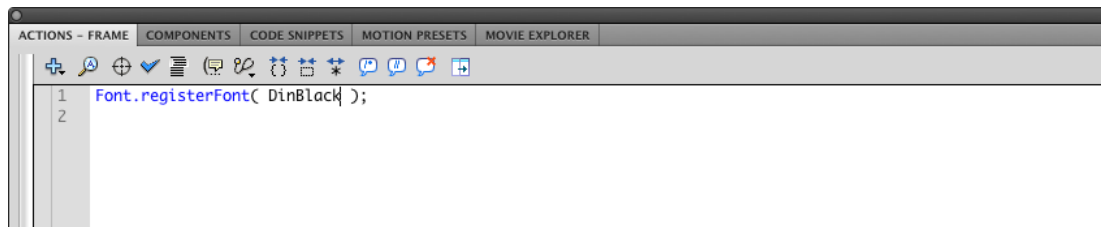
(Figure D.2) The Actionscript Tab

10 - Select Export for Actionscript option, and then press 'OK'

11 - Insert the following code on the first frame of the Flash timeline (Figure D.3):

```
Font.registerFont( FontName* );
```

* Where the FontName is the custom name specified in step 4 (Figure D.1 - highlighted with orange).



(Figure D.3) Registering new Font in ActionScript

To embed further fonts, repeat the process from step 2.

When ready, create an .swf file, and specify the URL of this .swf file in the XML [Settings/FontLibraryFile](#) attribute:

```
FontLibraryFile="pageFlipFonts.swf"
```

Then, User can refer to the newly embedded Fonts with their names noted down in step 8.

To specify a custom font as *PageFlip4* default Font, configure the [Settings/Default-Font](#) attribute. Example:

```
DefaultFont="DINBlk_PFL Normal"
```

E. XML SETTINGS REFERENCE

All the XML configurations - sorted in categories - are listed here, with default values indicated.

Settings node attributes of the XML configuration file contain the general settings (Attribute name="default value" (type) – short description):

<code>StartPage="1"</code>	(uint)	- Start PageFlip with this page opened
<code>StartAutoFlip="false"</code>	(Boolean)	- Enable automatic flip activation at start
<code>ThumbnailModeStart="false"</code>	(Boolean)	- Enable Thumbnail view when <i>PageFlip4</i> starts
<code>MouseControl="true"</code>	(Boolean)	- Enable flipping with mouse
<code>KeyboardShortcutsEnabled="true"</code>	(Boolean)	- Enable keyboard control
<code>CopyrightMessage=""</code>	(String)	- Copyright Message (encoded with XML Validator)
<code>PageWidth="300"</code>	(uint)	- Page width in pixels
<code>PageHeight="400"</code>	(uint)	- Page height in pixels
<code>AutoSize="true"</code>	(Boolean)	- Resize to fit loaded content on the <i>Page</i>
<code>PageScale="true"</code>	(Boolean)	- Scaling the Book to fit the <i>Screen</i>
<code>MinScale="50"</code>	(uint)	- Minimum scale in %
<code>MaxScale="200"</code>	(uint)	- Maximum scale in %
<code>ViewAngle="0"</code>	(int)	- Tilting Book
<code>MinStageWidth="632"</code>	(uint)	- Minimum width of <i>Screen</i> in pixels
<code>MinStageHeight="580"</code>	(uint)	- Minimum height of <i>Screen</i> in pixels
<code>BookMarginHorizontal="32"</code>	(uint)	- Horizontal margins of the Book
<code>BookMarginVertical="32"</code>	(uint)	- Vertical margins of the Book
<code>RightToLeft="false"</code>	(Boolean)	- Book written from right to left
<code>VerticalMode="false"</code>	(Boolean)	- Vertical flipping
<code>OffsetX="0"</code>	(uint)	- Horizontal offset of the Book position,
<code>OffsetY="0"</code>	(uint)	- and vertical in pixels
<code>CenterSinglePage="false"</code>	(boolean)	- Center Book if only one Page is visible
<code>AlwaysOpened="false"</code>	(Boolean)	- Book always open (can't close the covers)
<code>ZeroPage="false"</code>	(Boolean)	- Enable page 0 (a page in background)
<code>ZeroPageAlwaysVisible="false"</code>	(Boolean)	- Enable 0 page always visible
<code>HardCover="true"</code>	(Boolean)	- Hard cover Book
<code>HardPage="false"</code>	(Boolean)	- Hard Pages
<code>LargeCover="true"</code>	(Boolean)	- Cover Pages larger than inside Pages
<code>LargeCoverVerticalOversize="10"</code>	(uint)	- Vertical oversize in pixels
<code>LargeCoverHorizontalOversize="10"</code>	(uint)	- Horizontal oversize in pixels
<code>EmbossedPages="true"</code>	(Boolean)	- Embossed pages (with light/shadow effect)
<code>EmbossAlpha="100"</code>	(uint)	- Transparency of the emboss effect (0-100)
<code>ShadowsEnabled="true"</code>	(boolean)	- Enable shadow effect while flipping
<code>ShadowsAlpha="100"</code>	(uint)	- Transparency of shadow effect (0-100)

DropShadow="true"	(Boolean)	- Enable DropShadow of the Book
DropShadowSize="30"	(uint)	- Size of DropShadow blur in pixels
DropShadowOffset="10"	(int)	- Vertical offset of DropShadow in pixels
DropShadowAlpha="50"	(uint)	- Transparency of DropShadow (0-100)
PageLoaderBackColor="0x333333"	(uint)	- Background color of pages without content
TransparentBackground="false"	(Boolean)	- Enable Transparent <i>Screen</i> background
BackgroundColor="0xF0F0F0"	(uint)	- Background color of <i>Screen</i>
BackgroundImageFile=""	(String)	- URL of <i>Screen</i> background image
LayoutOrder="PREVIEW-CONTENT-CONTROLBAR"	(String)	- Order of layout elements
ThumbnailsEnabled="false"	(Boolean)	- Enable Thumbnail view
ContentPreviewEnabled="false"	(Boolean)	- Enable Preview
ControlBar="true"	(Boolean)	- Enable ControlBar
PageDisplayEnabled="true"	(Boolean)	- Enable pager text display
PageNameDisplay="false"	(Boolean)	- Display Page name instead of numbers
CopyrightMessageDisplay="true"	(Boolean)	- Displaying Copyright Message
ContentPreviewHeight="60"	(uint)	- Preview height in pixels
ControlBarHeight="55"	(uint)	- ControlBar height in pixels
ControlBarOffset="15"	(uint)	- Offset of ControlBar position in pixels
ControlBarMargin="16"	(uint)	- Horizontal margins of the Controlbar
ControlBarScale="100"	(Number)	- ControlBar scale in %
ControlBarFullScreenScale=""	(Number)	- Scale of ControlBar in % in Full Screen mode
ControlBarPagerWidth="150"	(uint)	- Width of ControlBar pager in pixels
ControlBarOnTop="false"	(Boolean)	- Controlbar over the Book
ControlBarBackgroundColor=""	(Number)	- Background color of ControlBar
ControlBarBackgroundAlpha=""	(Number)	- Transparency of ControlBar background (0-100)
CopyrightMessageBackgroundColor=""	(Number)	- Copyright Message background color, and
CopyrightMessageBackgroundAlpha=""	(Number)	- transparency of its background (0-100)
FontLibraryFile=""	(String)	- URL of file including embedded Fonts
DefaultFont=""	(String)	- Embedded font instead of default
DragArea="64"	(uint)	- Drag area width from outer side of the page
AutoFlipArea="56"	(uint)	- Width of rectangle indicating active corner
FlippingDuration="1000"	(uint)	- Duration of flipping animation in milliseconds
MouseFollowSpeed="5"	(uint)	- Mouse follow speed when flipping
ZoomFollowSpeed="5"	(uint)	- Mouse follow speed in zoom
ZoomOnClick="false"	(Boolean)	- Enable Zooming by mouse click
DragZoom="false"	(Boolean)	- Enable Zoom image dragging with mouse
ZoomHotspot="true"	(Boolean)	- Enable HotSpots in Zoom mode
DefaultTransition="1"	(uint)	- Default transition (NONE,FADE,ZOOM,BLUR)
DefaultTransitionDuration="250"	(uint)	- Duration of Default transition in milliseconds
FlippingSpreadActivity="false"	(Boolean)	- Enable animation on Spreads during flipping
AutoFlipDefaultInterval="5"	(uint)	- Interval of automatic flip in seconds
AutoFlipLooping="true"	(Boolean)	- Starts from beginning if reaching end (Loop)
TransparencyEnabled="true"	(Boolean)	- Enable transparent Pages

TransparencyAutoLevel="true"	(Boolean)	- Automatic adjustment of transparency level
TransparencyMaxLevel="1"	(uint)	- Maximum transparency level if not automatic
CustomControlBarIcons="false"	(Boolean)	- Enable custom ControlBar buttons
CustomControlBarFile=""	(String)	- URL of .swf file including custom buttons
CustomControlBarLayout="false"	(Boolean)	- Enable custom ControlBar layout
EnableButtonColoring="true"	(Boolean)	- Enable coloring of the ControlBar buttons
ButtonColor="0x333333"	(uint)	- Base color of button, and
ButtonAlpha="100"	(uint)	- transparency (0-100)
ButtonOverColor="0xF0B400"	(uint)	- Button color when mouse hovers over, and
ButtonOverAlpha="100"	(uint)	- transparency (0-100)
ButtonPressColor="0x333333"	(uint)	- Color of button when pressed, and
ButtonPressAlpha="100"	(uint)	- transparency (0-100)
ButtonDisabledColor="0xEEEEEE"	(uint)	- Color of button when inactive, and
ButtonDisabledAlpha="100"	(uint)	- transparency (0-100)
ControlBarLoaderEnabled="true"	(Boolean)	- Enable Loaderbar on ControlBar
ButtonFirstLastEnabled="true"	(Boolean)	- Enable buttons flipping to First/Last pages
ButtonLeftRightEnabled="true"	(Boolean)	- Enable buttons flipping Right/Left
ButtonZoomEnabled="true"	(Boolean)	- Enable Zoom buttons
ButtonPrintEnabled="false"	(Boolean)	- Enable Print buttons
ButtonAutoFlipEnabled="false"	(Boolean)	- Enable Automatic flipping button
ButtonPDFLinkEnabled="true"	(Boolean)	- Enable .pdf download button
ButtonThumbnailEnabled="true"	(Boolean)	- Enable Thumbnail button
ButtonLanguageEnabled="false"	(Boolean)	- Enable Language switch button
ButtonMuteEnabled="true"	(Boolean)	- Enable Mute button
ButtonInfoEnabled="false"	(Boolean)	- Enable Info window
ButtonIndexEnabled="false"	(Boolean)	- Enable Table of Contents button
ButtonFullScreenEnabled="true"	(Boolean)	- Enable Full Screen button
ButtonCloseEnabled="false"	(Boolean)	- Enable Close button
ButtonCustom0Enabled="false"	(Boolean)	- Enable Custom buttons
ButtonCustom1Enabled="false"	(Boolean)	
ButtonCustom2Enabled="false"	(Boolean)	
ButtonCustom3Enabled="false"	(Boolean)	
ButtonCustom4Enabled="false"	(Boolean)	
ButtonCustom5Enabled="false"	(Boolean)	
ButtonCustom6Enabled="false"	(Boolean)	
ButtonCustom7Enabled="false"	(Boolean)	
PDFLink=""	(String)	- .pdf download button downloads this file
ButtonToolTip="true"	(Boolean)	- Enable ToolTip if mouse hovers over button
ButtonToolTipDelay="60"	(uint)	- Tooltip display delay in frames
ThumbnailWidth="90"	(uint)	- Thumbnail view width, and
ThumbnailHeight="120"	(uint)	- height
ThumbnailDropShadow="true"	(Boolean)	- Enable DropShadow of Thumbnail view
ThumbnailDropShadowSize="8"	(uint)	- Size of DropShadow blur in pixels
ThumbnailDropShadowOffset="3"	(int)	- Vertical offset of DropShadow in pixels
ThumbnailDropShadowAlpha="50"	(uint)	- Transparency of DropShadow (0-100)
PopupMaxWidth="200"	(uint)	- Set the maximum width of popup window

PopupBackgroundColor="0x000000"	(uint)	- Background color of Popup window, and
PopupBackgroundOpacity="50"	(uint)	- transparency (0-100)
PopupBorderThickness="0"	(uint)	- Thickness of Popup border (0 = no border)
PopupBorderColor="0x000000"	(uint)	- Border color
PopupRounded="true"	(Boolean)	- Enable rounded border
PopupRoundedRadius="8"	(uint)	- Radius of Popup rounding
PopupMargin="8"	(uint)	- Margin of Popup content
PopupTitleFont=""	(String)	- Title font
PopupTextFont=""	(String)	- Text font
PopupTitleColor="0xFFFFFFFF"	(uint)	- Title color
PopupTextColor="0xFFFFFFFF"	(uint)	- Text color
PopupSpace="8"	(uint)	- Space between content elements in pixels
PopupMouseFollowSpeed="5"	(Number)	- Mouse follow speed
InfoWindowWidth="300"	(uint)	- Width of Info window content
FloatingWindowColumnWidth="200"	(uint)	- Width of floating window column
FloatingWindowMargin="10"	(uint)	- Margin of floating window content
FloatingWindowHorizontalSpace="16"	(int)	- Floating window horizontal space
FloatingWindowVerticalSpace="16"	(int)	- Floating window vertical space
FloatingWindowSeparator="true"	(Boolean)	- Enable floating window separators
FloatingWindowSeparatorColor="0xFFFFFFFF"	(uint)	- Color of floating window separators
FloatingWindowSeparatorAlpha="50"	(uint)	- Transparency of floating window separators
FloatingWindowSeparatorHeight="1"	(Number)	- Height of floating window separators
FloatingWindowSeparatorOffset="6"	(Number)	- Offset of floating window separators
FloatingWindowBackgroundColor="0x000000"	(uint)	- Background color of floating window
FloatingWindowBackgroundAlpha="50"	(uint)	- Background transparency of floating window
FloatingWindowBorderThickness="0"	(Number)	- Thickness of floating window border
FloatingWindowBorderColor="0xFFFFFFFF"	(uint)	- Color of floating window border
FloatingWindowBorderAlpha="50"	(uint)	- Transparency of floating window border
FloatingWindowCornerRadius="16"	(uint)	- Radius of floating window rounded corner
FloatingWindowFont=""	(String)	- Default font of floating window
FloatingWindowColor="0xFFFFFFFF"	(uint)	- Color of floating window content
FloatingWindowSize="11"	(Number)	- Size of floating window text
FloatingWindowAlign="LEFT"	(String)	- Align of floating window text
TOCTitleFont=""	(String)	- Font of Table of Contents Title
TOCTitleColor="0xFFFFFFFF"	(uint)	- Color of Table of Contents Title
TOCTitleSize="19"	(Number)	- Size of Table of Contents Title font
TOCTitleAlign="LEFT"	(String)	- Align of Table of Contents Title
TOCLinkFont=""	(String)	- Font of Table of Contents Link
TOCLinkColor="0xFFFFFFFF"	(uint)	- Color of Table of Contents Link
TOCLinkSize="12"	(Number)	- Size of Table of Contents Link font
TOCLinkAlign="LEFT"	(String)	- Align of Table of Contents Link
TOCLinkPageNumberFont=""	(String)	- Font of Table of Contents page number
TOCLinkPageNumberColor="0xFFFFFFFF"	(uint)	- Color of Table of Contents page number
TOCLinkPageNumberSize="22"	(Number)	- Size of Table of Contents page number font
TOCLinkPageNumberAlign="LEFT"	(String)	- Align of Table of Contents page number
TOCPageNumberFirst="false"	(Boolean)	- Page number first
TOCPageNumberWidth="32"	(uint)	- Width of page number in pixels
TOCLinkOverColor="0xF0B400"	(uint)	- Color of Table of Contents Link if mouse hovers over
TOCLinkPressColor="0xFFFFFFFF"	(uint)	- Color of Table of Contents Link when pressed

<code>TOCLinkBackgroundColor="0x333333"</code>	(uint)	- Color of Table of Contents background
<code>TOCLinkBackgroundAlpha="0"</code>	(uint)	- Transparency of Table of Contents Link background
<code>TOCLinkBackgroundOverColor="0x333333"</code>	(uint)	- Color of Table of Contents Link if mouse hovers over
<code>TOCLinkBackgroundOverAlpha="50"</code>	(uint)	- Transparency of Table of Contents Link background if mouse hovers over
<code>TOCDescriptionFont=""</code>	(String)	- Font of Table of Contents Description
<code>TOCDescriptionColor="0xFFFFFFFF"</code>	(uint)	- Color of Table of Contents Description
<code>TOCDescriptionSize="11"</code>	(Number)	- Font Size of Table of Contents Description
<code>TOCDescriptionAlign="LEFT"</code>	(String)	- Align of Table of Contents Description
<code>ExternalScript=""</code>	(String)	- URL of the External Script file
<code>FPSControlEnabled="true"</code>	(Boolean)	- Enable Frame Per Second Control
<code>ShowFPS="false"</code>	(Boolean)	- Show Frame Per Second on Screen
<code>MaxFPS="60"</code>	(Number)	- Maximum Frame Per Second setting
<code>StartMute="false"</code>	(Boolean)	- Activate mute at start
<code>SoundEnabled="false"</code>	(Boolean)	- Enable sounds/effects
Settings/Sounds node attribute:		
<code>EffectVolume="10"</code>	(uint)	- Volume of sound effects
Settings/Sounds/HardPageClose node attribute:		
<code>SoundFile=""</code>	(String)	- Sound of Hard Cover flipping over
Settings/Sounds/PullOpen node attribute:		
<code>SoundFile=""</code>	(String)	- Sound of PullFlip flipping start
Settings/Sounds/PullClose node attribute:		
<code>SoundFile=""</code>	(String)	- Sound of PullFlip flipping over
Settings/Sounds/PushOpen node attribute:		
<code>SoundFile=""</code>	(String)	- Sound of PushFlip flipping start
Settings/Sounds/PushClose node attribute:		
<code>SoundFile=""</code>	(String)	- Sound of PushFlip flipping over
Settings/Sounds/PageTear node attribute:		
<code>SoundFile=""</code>	(String)	- Sound of Page tear

And finally, there is a list of the new attributes added in version 4.1:

<code>CenterSinglePage="false"</code>	(Boolean)	- Center Book if only one Page is visible
<code>BookMarginHorizontal="32"</code>	(uint)	- Horizontal margins of the Book
<code>BookMarginVertical="32"</code>	(uint)	- Vertical margins of the Book
<code>EmbossAlpha="100"</code>	(uint)	- Transparency of the emboss effect (0-100)
<code>ShadowsEnabled="true"</code>	(Boolean)	- Enable shadow effect while flipping
<code>ShadowsAlpha="100"</code>	(uint)	- Transparency of shadow effect (0-100)
<code>PageNameDisplay="false"</code>	(Boolean)	- Display Page name instead of numbers
<code>ControlBarMargin="16"</code>	(uint)	- Horizontal margins of the Controlbar
<code>ControlBarPagerWidth="150"</code>	(uint)	- Width of ControlBar pager in pixels
<code>ControlBarOnTop="false"</code>	(Boolean)	- Controlbar over the Book
<code>KeyboardShortcutsEnabled="true"</code>	(Boolean)	- Enable keyboard control
<code>PageDisplayEnabled="true"</code>	(Boolean)	- Enable pager text display
<code>ButtonCustom0Enabled="false"</code>	(Boolean)	- Enable Custom buttons
<code>ButtonCustom1Enabled="false"</code>	(Boolean)	
<code>ButtonCustom2Enabled="false"</code>	(Boolean)	

<code>ButtonCustom3Enabled="false"</code>	(Boolean)	
<code>ButtonCustom4Enabled="false"</code>	(Boolean)	
<code>ButtonCustom5Enabled="false"</code>	(Boolean)	
<code>ButtonCustom6Enabled="false"</code>	(Boolean)	
<code>ButtonCustom7Enabled="false"</code>	(Boolean)	
<code>ButtonToolTipDelay="60"</code>	(uint)	- Tooltip display delay in frames
<code>ZoomOnClick="false"</code>	(Boolean)	- Enable Zooming by mouse click
<code>DragZoom="false"</code>	(Boolean)	- Enable Zoom image dragging with mouse
<code>ZoomHotspot="true"</code>	(Boolean)	- Enable HotSpots in Zoom mode
<code>PopupMaxWidth="200"</code>	(uint)	- Set the maximum width of popup window
<code>ExternalScript=""</code>	(String)	- URL of the External Script file

F. HOTSPOT XML REFERENCE

Structure of the HotSpot XML files: The root-node directly contains the HotSpot nodes that describe 1-1 active links:

A `<HotSpot>` node attribute:

<code>PopupImageFile=""</code>	(String)	- URL of Popup window image
<code>PopupTitle=""</code>	(String)	- Title of Popup window
<code>PopupText=""</code>	(String)	- Text of Popup window
(If no value is specified for the tree attributes above, the Popup window will not be displayed)		
<code>TargetLabel=""</code>	(String)	- Label of target page
<code>TargetName=""</code>	(String)	- Name of the target page
<code>TargetPage=""</code>	(String)	- Page number of target page
<code>Skip="false"</code>	(Boolean)	- Direct jump to target page
<code>Link=""</code>	(String)	- URL the User wants to open
<code>Target="_blank"</code>	(String)	- Link Target Window
<code>LightBoxImage=""</code>	(String)	- URL of image to be opened in LightBox, and
<code>LightBoxCaption=""</code>	(String)	- image description
<code>LightBoxGroup=""</code>	(String)	- ID of the Image in the LightBox Group
<code>ESfunctions=""</code>	(String)	- Executes an External Script's function
<code>SWFfunctions=""</code>	(String)	- Executes a Page Script's function
<code>AutoFlip=""</code>	(String)	- Controls AutoFlipping (" <code>start</code> ", " <code>stop</code> ", " <code>toggle</code> ")
<code>Color="0x000000"</code>	(uint)	- Color of the area
<code>Opacity="0"</code>	(uint)	- Transparency (0-100)
<code>HoverOpacity="0"</code>	(uint)	- Transparency (0-100) if mouse hovers over

User can configure Hotspot shapes with the following nodes (one or more for each HotSpot):

```
<Rectangle>x,y,w,h</Rectangle>
<RoundRectangle>x,y,w,h,r</RoundRectangle>
<Circle>x,y,r</Circle>
<Star>x,y,p,r1,r2,phase</Star>
<Shape>x1,y1/x2,y2/x3,y3/x4,y4</Shape>
```

For information on the latter, refer to section 3.11.3.

G. PAGEFLIP4 ERROR MESSAGES

Error messages appears at the startup of *PageFlip4*, centered on the empty *Screen*

INVALID LICENSE

Wrong code has been given to the *CopyrightMessage* attribute. See *XML Validator/Configuration of protection in Appendix A*.

INVALID URL

URL from where *PageFlip* has been loaded is unauthorized. See *URLs/Configuration of protection in Appendix A*.

TRIAL LICENSE EXPIRED

Time-limited trial License Key expired.

OFFLINE USAGE NOT ALLOWED

Offline run is unauthorized. See *URLs/Configuration of protection in Appendix A*.

XML ERROR

Error during the loading of the XML configuration file. It can be a wrong path to the file (pointing to a non-existent file), lack or loss of Internet connection, or other reasons due to which the file could not be loaded.

XML SYNTAX ERROR

Specified data cannot be interpreted at the processing of the XML file due to syntactic error.

H. TIPS FOR OPTIMIZATION

Optimization is important to maintain high fps. Follow the instructions to avoid difficulties with flipping the thoroughly prepared Book due to the accumulation of too complex contents:

- Make Pages the smallest possible, the content loaded to the Pages should be images the same size as the page. In case of vector-based content, avoid too complex graphics (similarly to text content which should be an image unless enabling selection).
- When transparency is used, make sure that the number of consecutive transparent pages is not too great (it increases the number of Pages PageFlip has to render simultaneously). In case it cannot be avoided, try to load simple content to the Pages (for example 32 bite png Page-files).
- To have all Pages rounded, transparency is unnecessary as the visible page hides the one behind it.
- Make sure to display videos on single Pages.
- If possible, try to avoid the use of Spreads, as they require more resources, especially when more Spreads follow each other,
- In case more Spreads follow each other, disable animation during flipping. ****
- Create interactive pages in a way that when using the event handlers the animations/scripts running on the page should stop until the page is displayed.
- If the Book consists of a great number of pages, or the Page-files are too big, enable Page Unloading, so that unwanted pages get removed from the memory (in case of pages running scripts, removal has to be separately disabled not to lose data)

I. EMBEDDING *PAGEFLIP4*

PageFlip4 can be embedded in our homepage or any other html file in several ways. To have all its functions fully working, attach the `swfobject.js` file to your html file, as in the case of the original `index.html`:

```
<head>
  <script type="text/javascript" src="js/swfobject.js"></script>
</head>
```

Make sure that the relative location of the html files, the `pageflip.swf`, and the `pageflipdata/` folders are correct in the folder structure. The file system root for the code running under *PageFlip4* is the folder containing the html file which embeds *PageFlip4*. `DataFolder`, which points at the `pageflipdata/` subfolder by default, is added to this, and the URLs in the XML connect to this as well. For example:

`http://pageflip.hu/pagefliptest/index.html` (let's assume that this is the html file embedding *PageFlip4*, and that `DatFolder` has not been modified)

If User specifies the following path to a file in the XML configuration:

```
pages/coverpage.jpg
```

it will be the following URL actually:

```
pageflipdata/pages/coverpage.jpg
```

Which, in this case (due to the `index.html` file situation), is similar to the following absolute URL:

```
http://pageflip.hu/pagefliptest/pageflipdata/pages/coverpage.jpg
```

If User wants to use absolute URLs to reach files, then the `DataFolder` should be deleted with the help of the flashvars:

```
flashvars.DataFolder = "";
```

Then, be aware that all relative URLs change, if User does not modify the location of the files, start with the path of the XML:

```
flashvars.XMLFileName = "pageflipdata/pageflipdata.xml";
```

The embedding Script is also in the `<head>` tag:

```
<head>
  <script type="text/javascript">
    var flashvars = {};
    flashvars.XMLFileName = "pageflipdata.xml";
    flashvars.DataFolder = "pageflipdata/";
    flashvars.StartPage = "1";
    flashvars.StartAutoFlip = "false";
    var params = {};
    params.scale = "noscale";
    params.salign = "TL";
    params.wmode = "transparent";
```

```
        params.allowscriptaccess = "always";
        params.allowfullscreen = "true";
        params.menu = "true";
        params.bgcolor = "#FFFFFF";
        var attributes = {};
        swfobject.embedSWF("pageFlip.swf", "pageflip", "100%", "100%",
            "10.0.0", false, flashvars, params,
            attributes);
    </script>
</head>
```

The `<div>` tag is responsible for the display of PageFlip is in the `<body>` tag:

```
<body>
    <div id="pageflip" style="margin: 0;"></div>
</body>
```

If PageFlip does not fit the browser completely, then User should specify the width and height in pixels instead of "100%" (swfobject.embedSWF).

The next section of the Appendix informs User about further preparations necessary for the support of LightBox functions.

J. USE OF LIGHTBOX

User has to attach the following .css, .js files and a short script to the embedding html file for a functioning LightBox:

```
<head>
  <link rel="stylesheet" href="css/lightbox.css" type="text/css"
        media="screen" />
  <script src="js/prototype.js" type="text/javascript"></script>
  <script src="js/scriptaculous.js?load=effects" type="text/javas-
        cript"></script>
  <script src="js/lightbox++.js" type="text/javascript"></script>
  <script type="text/javascript">
    function GroupDelegate(id) {
      var objLink = document.getElementById(id);
      Lightbox.prototype.start(objLink);
    }
    function LightboxDelegate(url,caption) {
      var objLink = document.createElement('a');
      objLink.setAttribute('href',url);
      objLink.setAttribute('rel','lightbox');
      objLink.setAttribute('title',caption);
      Lightbox.prototype.start(objLink);
    }
  </script>
</head>
```

When PageFlip displays an image with the help of LightBox, it only needs its file name (URL) and image description, and it automatically invites the LightboxDelegate() JS function with the modified URL.

If User wants to use the LightBox group (image groups), then the group has to be defined in the html first, while reference has to be made to the image id at display.

Definition of a group also in the embedding html:

```
<body>
  <a id="pic1" href="pageflipdata/pages/a.png" rel="lightbox[group1]"
    title="caption1"></a>
  <a id="pic2" href="pageflipdata/pages/b.png" rel="lightbox[group1]"
    title="caption2"></a>
  <a id="pic3" href="pageflipdata/pages/c.png" rel="lightbox[group1]"
    title="caption3"></a>
</body>
```

For more information on customizing LightBox, visit the developer's homepage: <http://www.huddletogether.com/projects/lightbox2/>

K. COLOR CODES

Color codes are specified in Hexadecimal RRGGBB code.

For example, the color white is: RGB 255, 255, 255 – in hexadecimal format: FF, FF, FF, which the User can specify as color attribute:

```
Color="0xFFFFFF"
```


L. PAGEFLIP4 FAQ

Can *PageFlip4* display PDF documents?

No, it can only read and display formats directly supported by Flash. PDF publications have to be converted page-by-page to the adequate format first, and Page-Flip can only load them afterwards.

Is there an admin or editor interface?

No, the contents of the pages have to be prepared by the User; the XML files have to be edited with text editor (TextEdit, TextWrangler, NotePad, Edit+, etc.).

How the Copyright message can be removed from the bottom of the Screen?

First you have to uncheck the "Copyright Message Always Visible" checkbox in XML Validator, then set the following attribute in <Settings> node:

```
CopyrightMessageDisplay = "false"
```

How to remove the "Powered by *PageFlip4*" context menu?

Disable Context Menu by the following <Settings> attribute:

```
ContextMenu = "false"
```