hexagonit.virtualgallery Documentation

Release 1.0.0.1

Hexagon IT

December 04, 2014

hexagonit.virtualgallery is a Plone add-on that renders a Flash-based 3D virtual gallery.

- Source code @ GitHub
- Releases @ PyPI
- Sphinx docs @ ReadTheDocs

CHAPTER 1

Installation

To install `hexagonit.virtualgallery` you simply add `hexagonit.virtualgallery` to the list of eggs in your buildout, run buildout and restart Plone. Then, install *hexagonit.virtualgallery* using the Add-ons control panel.

CHAPTER 2

Requirements

- Plone 4.1 or newer
- Flash player 10.2 or newer

CHAPTER 3

Basic usage

This package adds the *Virtual 3D gallery* display mode to Folder and Collection. So, go to any folder or collection that contains images and select `Virtual 3D gallery` from the display drop-down menu.

In the gallery each room has enough space for 17 images. If the image source (Folder or Collection) contains more images then additional rooms will be automatically created and doors allow the user to travel between the rooms.

The gallery can be navigated either by using the mouse or the keyboard. Use the arrow keys to move within the space (hold down shift to strafe). Clicking an image with the mouse will concentrate on that image and the image metadata will become visible when hovering over the image.

You can view each image in turn by clicking on the arrow buttons on the left and right side of the viewer. The viewer can also be put in fullscreen mode by clicking on the fullscreen button on the top right corner.

CHAPTER 4

Configuration

Each gallery may be configured to use a particular image scale (provided by plone.app.imaging) by choosing the appropriate scale on the `Virtual gallery settings` page. A link to the settings page will become visible when the context is using the virtual gallery display.

By default the gallery uses the original scale of the images but in case of large images this may slow down the gallery significantly. Using a smaller scale version of the images usually improves the loading time and performance.

You can also configure new image scales in `Site setup › Image handling`. For example, you could define a “HD” scale at 1200x1200 pixels which would give most current day users a good quality image even in the fullscreen mode.

CHAPTER 5

Advanced usage

The Flash viewer is independent of Plone and may be used in any web context, even another framework or a language environment. The viewer requires two distinct parts to work:

- The HTML code to embed the viewer and pass parameters
- JSON configuration which configures the viewer.

5.1 Embed code

The HTML code needs to embed the `Virtual3DGallery.swf` viewer and pass a single flashvars variable called `dataURL` to the viewer. This variable must contain the URL to the JSON configuration file which configures the viewer.

If you are embedding the viewer within a page that contains other content it is also recommended set the `wmode=window` flash parameter so that HTML elements can be positioned above the flash content.

By default, this package uses the SWFObject Javascript library to do the embedding.

5.2 JSON configuration

The Flash viewer is driven by the JSON configuration which contains the list of images to display in the gallery, associated image metadata (title, author, description) and translations for the UI textual elements, e.g.:

```
{"images": [
    {"url": "http://my.server.com/images/image1.png",
     "title": "Nice scenery",
     "description": "Lorem lipsum..",
     "author": "dokai"},
    {"url": "http://my.server.com/images/image2.png",
     "title": "Screenshot of Foo",
     "description": "Lorem lipsum..",
     "author": "dokai"}
    ],
 "ui": {
    "enterRoomToolTip": "Click to enter",
    "fullscreen": "Fullscreen",
    "loadingImg": "Loading image:",
    "enterRoom": "Entering room [x] of [y]",
    "anaglyph": "Anaglyph"
 },
 "settings": {
    "anaglyphModeEnabled": "false"
 }}
```

The viewer does not care where the images and associated metadata come from so you can implement any custom logic that puts together the list (custom catalog queries, structural hierarchies, etc) as long as the resulting JSON document is valid.

The package contains an associated JSON schema document which can be used to validate the JSON configuration, e.g.:

```
from hexagonit.virtualgallery.schema import GALLERY_DATA_SCHEMA
from validictory import validate
import json

try:
    # Assuming the 'my_custom_config' contains the Python
    # data structure with the image information.
    validate(my_custom_config, GALLERY_DATA_SCHEMA)
    json_config = json.dumps(my_custom_config)
except ValueError:
    # Validation failed, do something.
    pass
```

You might want to display the gallery somewhere else or possibly in a toolbarless new window. To keep all Plone stuff away from the virtual gallery use a URL like below to only get the title of the gallery and the Flash object that displays it:

```
http://<path>/<to>/<your>/<gallery>/<folder>/virtualgallery?ajax_load=1&ajax_include_head=1
```

CHAPTER 6

TODO

- cross-browser testing
- use in the wild

CHAPTER 7

Credits

- Idea, skeleton and sponsoring provided by Hexagon IT Oy.
- Flash part of the gallery developed by Michal Zwieruho, BlackMoon Design.
- Plone integration by Nejc Zupan, NiteoWeb Ltd.

CHAPTER 8

Changelog

8.1 1.0.0.1 (2012-11-20)

- Release for Plone-4.2.2. [taito]

8.2 1.0 (2011-08-26)

- Added support for configuring the image scale used in the gallery on per-context basis. [dokai]
- Updated documentation [dokai]
- Updated Finnish translations. [dokai]
- Support for translations added. [zupo]
- Fixed the Flash window mode so that Plone dropdown menus are visible on top of the Flash movie. [dokai]
- Updated the Flash movie to a new version. Highlights:
  - No more empty frames on walls.
  - The default position when entering a room shows the middle box with the room number better.
  - Redundant tooltips (prev, next, forward, backward, etc) removed.
  - Traversal through doors using mouse clicks.
  - Repositioned the prev/next and fullscreen buttons.
  - Fixed a bug with image info popups being sticky in fullscreen mode.
  - Fixed a bug with empty galleries.
  - Fixed a bug with tooltips not expanding correctly over different sized text.
  - Fixed a bug where in some cases clicking on an image would zoom-in on a wrong one.
  - Traversing through doors is possible with a single click.

  Removed also the now redundant entries in the JSON configuration.

  [dokai]

8.3 1.0b4 (2011-08-10)

- Really fix the packaging error. The setuptools `unpack_tarfile` function filters out symlinks when unpacking so they will not be present in the unpacked package. This, combined with some weirdness in either `tar` itself or the Python tar module which reverses the order of the link, caused the targets of the symlinks to be removed from the final unpacked package. [dokai]

8.4 1.0b3 (2011-08-10)

- Fixed packaging error in 1.0b2. [dokai]
- JSON schema validation. [zupo]

8.5 1.0b2 (2011-08-04)

- Code cleanups. [zupo]
- More comments and documentation. [zupo]
- More tests. [zupo]

8.6 1.0b1 (2011-08-04)

- Initial release. [zupo]

License

Translations

Rebuild POT:

```
$ i18ndude rebuild-pot --pot locales/hexagonit.virtualgallery.pot --merge locales/manual.pot --cr
```

Sync a translation file with POT:

```
$ find locales -name '*.po' -exec i18ndude sync --pot locales/hexagonit.virtualgallery.pot {} \;
```

CHAPTER 11

Indices and tables

- *genindex*
- *modindex*
- *search*