

YAG - Yet Another Gallery

Extension Key: yag

Language: en

Keywords: gallery, forEditors, forAdmins, forDevelopers

Copyright 2010-2012, Daniel Lienert, Michael Knoll, <info@yag-gallery.de>

For further information see <http://www.yag-gallery.de>

This document is published under the Open Content License
available from <http://www.opencontent.org/opl.shtml>

The content of this document is related to TYPO3
- a GNU/GPL CMS/Framework available from www.typo3.org

Table of Contents

YAG - Yet Another Gallery.....	1	Upgrading from YAG 1.x.x to 2.x.x	32
Some Credits.....	3	Administration.....	33
Who made this?.....	3	Setting up YAG for standalone usage.....	33
How can you help?.....	3	Setting up YAG for usage as content-element.....	34
Introduction.....	4	Setting up a random item list.....	35
What does it do?.....	4	Configuration possibilities in FlexForm.....	36
Screenshots.....	4	Configuration.....	40
Users manual.....	10	TypoScript Reference.....	40
Setting up a sysfolder for YAG.....	10	Use returnUrl with YAG to get speaking URLs.....	46
Setting up your first Gallery.....	11	Tutorial.....	47
Setting up your first Album.....	11	How to create your own Themes as a third-party	
Uploading Images into Albums.....	13	extension	47
Editing Image data inside albums.....	15	Developers Guide	54
Editing all images of an album at once.....	20	YAG Architecture.....	54
Sorting of images by attribute.....	20	Filesystem Structure.....	55
Editing Albums.....	21	Image Storage and Resolution File Cache.....	55
Editing Galleries.....	22	Themes and Templates.....	57
Editing all albums of a gallery at once.....	22	The YAG context object.....	59
Gallery Maintenance.....	23	YAG ViewHelpers.....	59
Setting up Frontend-Editing with YAG.....	24	Extending via Signal/Slot.....	62
Get feedback for your images.....	26	Known problems.....	63
Installation.....	27	Working with TYPO3 4.5.x.....	63
Dependent extensions.....	27	Open Issues for version 1.x.....	63
Importing YAG from TER.....	27	To-Do list.....	64
Installing additional themes.....	29	ChangeLog.....	65

Some Credits

Who made this?

This extension was developed by [Daniel Lienert](#) and [Michael Knoll](#) with a lot of help from many different people. We spent a lot of our free time, implementing all the stuff. As we could use quite some open source software like TYPO3, Extbase Extension Framework, FLUID, jQuery and pt_extlist, we would like to give something back to the community which we do by uploading this extension to TER.

Special thanks goes to the Extbase team – Jochen and Sebastian and now Bastian – you did and do a great job on this one! Both developers of yag work at [punkt.de](#) in Karlsruhe and could use quite some software that was developed there and now supports YAG. So another big thank you goes to Jürgen Egeling for letting us use punkt.de's extensions to make things a lot easier.

We hope that you enjoy our extension and would appreciate your feedback. We need you to find all the bugs in there! Please let us know, if you have any questions or problems: info@yag-gallery.de

How can you help?

There are several ways to contribute: First of all, you can install the extension and give us your feedback. If you like it and use it on your website, it would be great, if you would let us know the link for our reference page on www.yag-gallery.de. If you would like to help us with development, we can set up a git account for you and you can easily join the team. If YAG gallery helps you on a commercial website or you want to donate – feel free to contact us on info@yag-gallery.de

If you need any features to be implemented – yes we also work for money :-)

Introduction

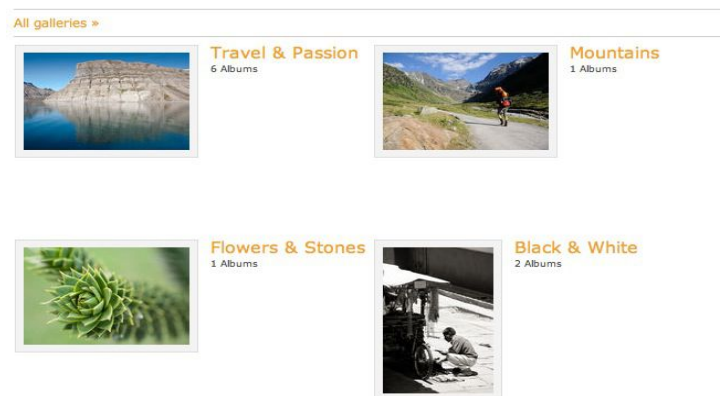
What does it do?

YAG is a gallery extension for Typo3. It offers many great features like gallery, album and image management as well as a full-featured backend-module for administration. See the screenshots below to get an impression on how it looks like.

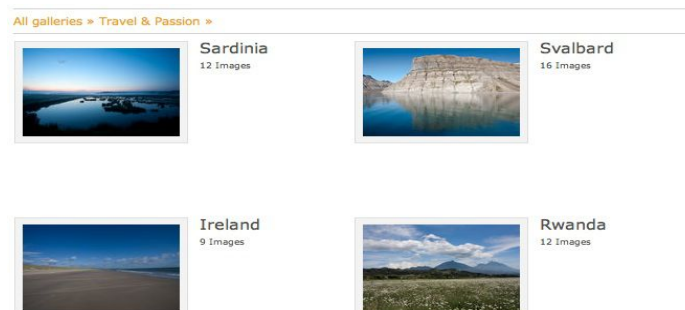
Screenshots

Frontend

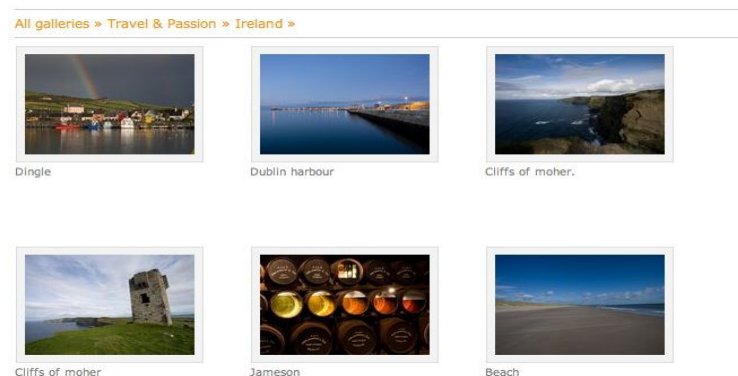
You can set up a list of galleries and use the as folders for your albums. The galleries can be sorted and paged as a list:



Within each gallery you can have as many albums as you like. These albums can be displayed as a list in the frontend:



Each album can be displayed as a list of images contained by this album:



An image can be presented in many different ways, using different templates and themes. Here is the standard template – showing the image with some useful information:

T3CON12 Asia Cambodia » Keynote

[Previous](#)

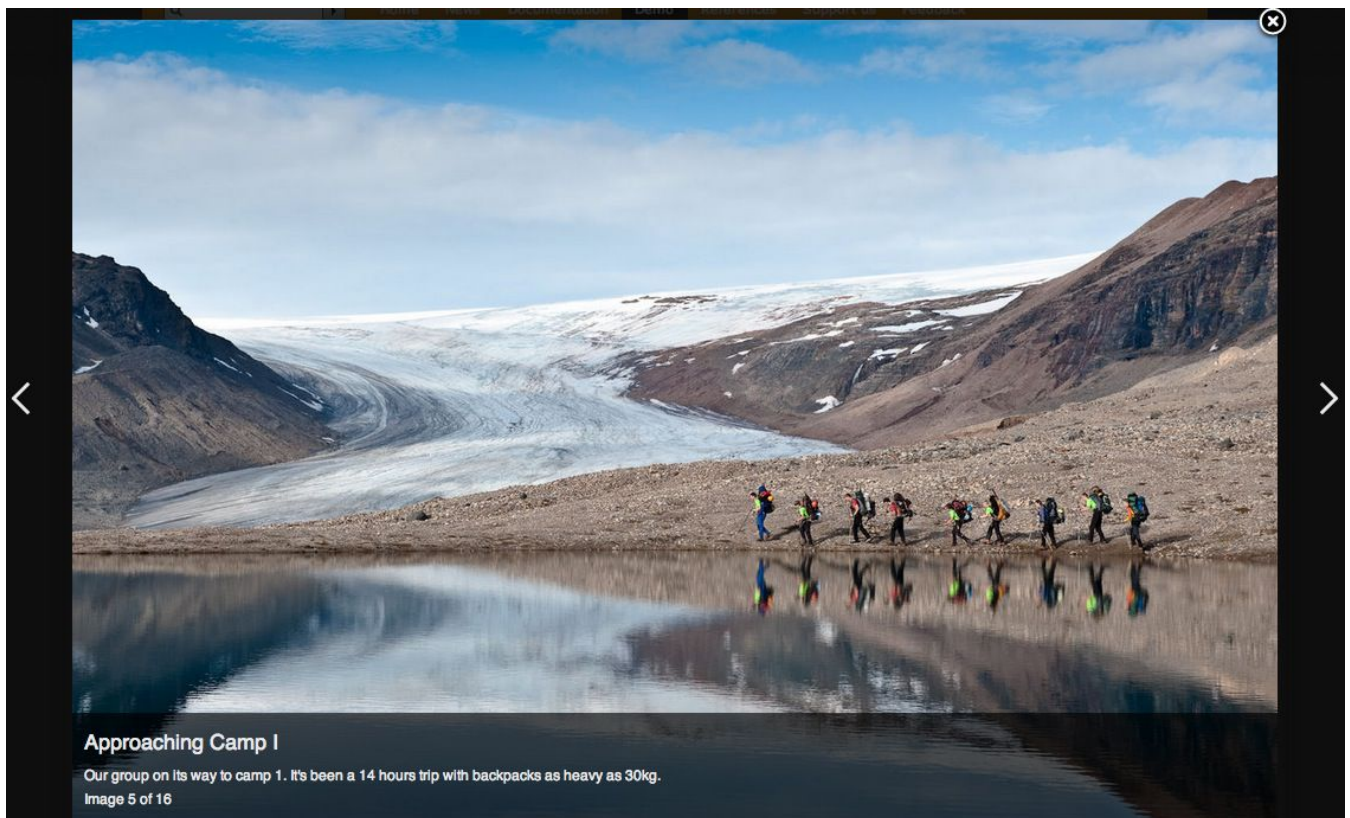
10 of 12

[Next](#)

Title:	Keynote	Camera Model:	Canon - Canon EOS 5D Mark II
Description:		Lens:	EF24-105mm f/4L IS USM
Artist:	Photographer: Daniel Lienert / d	Focal Length:	24 mm
Keywords:		Iso:	1000
Capture Date:	17.08.2012. 03:36	Shutter Speed:	1/60s
		Aperture:	f/4
		Flash:	16

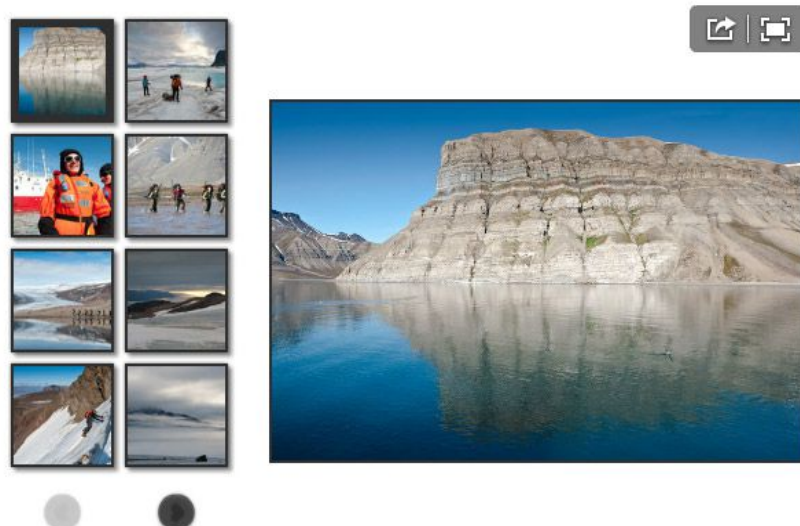
[Download full size image \(1000 x 562\)](#)

If you like, you can also use a lightbox like here:



With version 1.4.0 we added a „random photo“ widget which shows a random photo of a selected range of photos.

You can select from many different templates and plugins (like JavaScript Lightboxes or Flash albums) to actually render your albums on your site. It is one goal of YAG to enable you write your own templates easily. Here is another example of a rendered album using [SimpleViewer](#):



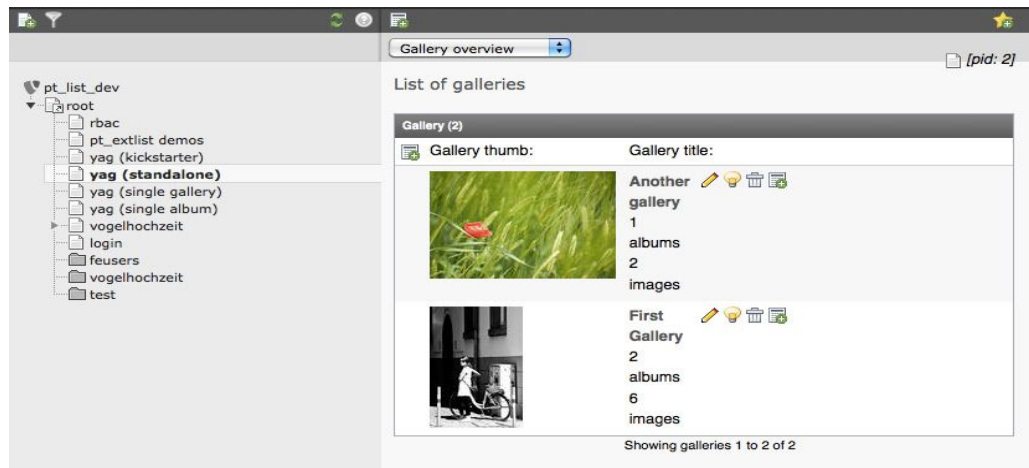
To get an overview of currently available themes, please refer to the demo section of our website: <http://www.yag-gallery.de/examples/>

Backend

YAG ships with a full-featured backend module that lets you administrate your gallery easily. After installing YAG, you get a new icon in the menu bar:



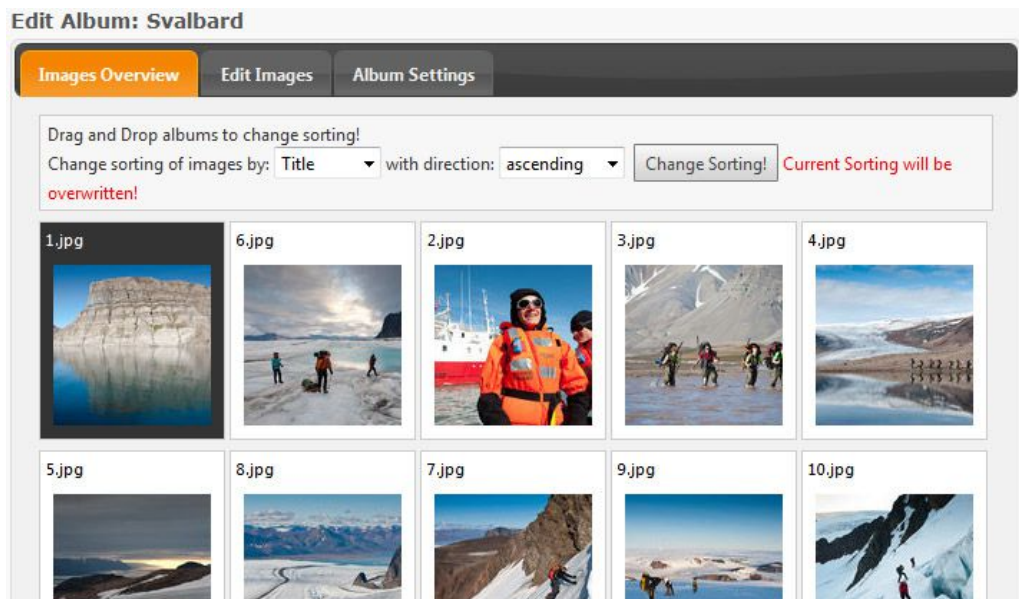
This will show an administration backend:



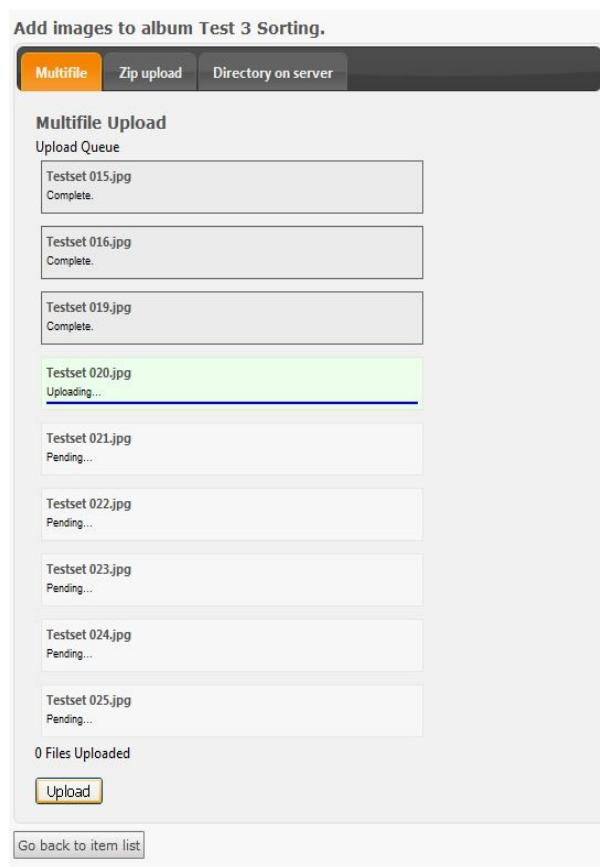
You can get a list of galleries, and for each gallery a list of albums inside this gallery:



There is a administration view that lets you easily manipulate your images in your albums. You can drag and drop images for sorting, change titles and descriptions and set thumbnails for albums:



Several upload possibilities including Flash-Multifile-Uploader and ZIP uploader make it easy to add images to your album:



Finally there is gallery maintenance module that gives you nice statistics on the images in your gallery and lets you maintain the resolution file cache:

Gallery maintenance

Statistics

Resolution File Cache

Gallery Info

Galleries: 6
Albums: 14
Images: 175
Total size: 98 M

14 instances of YAG content elements are included on pages.
Clear the cache entries of these pages.

Cache Info

Filecount: 1103
Cache size: 27 M

Clear resolution file cache

Gallery maintenance

Statistics

Resolution File Cache

Cache Info

Filecount: 1029
Cache size: 26 M

Clear resolution file cache

Build cache

Defined Themes:

- ☒ Default
- ☐ Lightbox
- ☐ GalleryView
- ☐ CrossSlide (Simple Crossfade)
- ☐ CrossSlide (Ken Burns Effect)
- ☐ Isotope
- ☐ SuperSized jQuery

Build resolution file cache

Inserting plugin via FlexForm

YAG comes with a very comfortable FlexForm that lets you select albums, galleries and images easily:

General Options

Source

Gallery

No selection

Flowers & Stones
1 Albums

Mountains
1 Albums

Travel & Passion
6 Albums

Black & White
2 Albums

YAG Documentation
3 Albums

Album

No selection

Sardinia
12 Images

Svalbard
16 Images

Ireland
9 Images

Russia
0 Images

Rwanda
12 Images

Image

No selection

sardinien-006.jpg

sardinien-015.jpg

sardinien-036.jpg

sardinien-040.jpg

sardinien-105.jpg

After you inserted the plugin, you get some information about your plugin in the page-module:

Page Content

Left

Normal

Right

Border

Default: Insert Plugin
Plugin: YAG - Yet Another Gallery
Plugin Mode: Specific Album
Album: First album
Theme: lightbox

Default: Insert Plugin
Plugin: YAG - Yet Another Gallery
Plugin Mode: Specific Album
Album: Here comes another album
Theme: default

Show hidden content elements

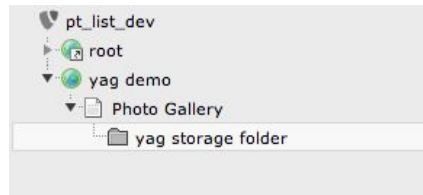
Users manual

We assume, that you have installed YAG according to the section "Installation". This chapter is about using the backend module. See chapter "Inserting YAG on your website" for setting up pages, templates and content element etc.

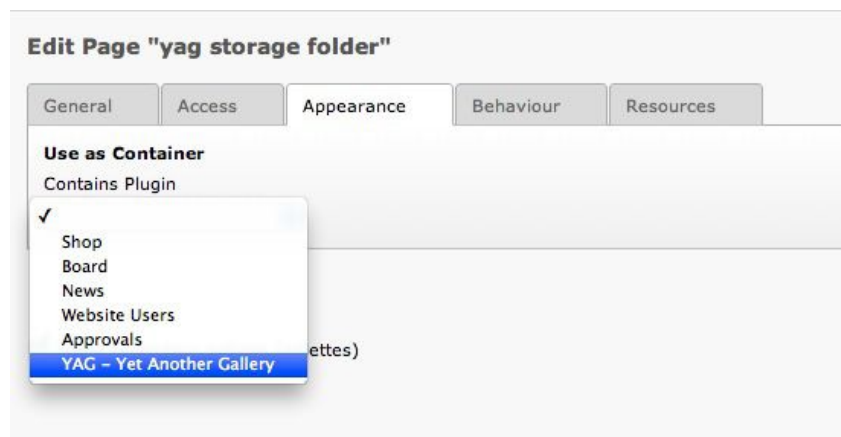
Setting up a sysfolder for YAG

Since version 2.0 of YAG, you have to set up a sysfolder that contains yag elements, e.g. galleries, albums and items. Without this sysfolder, you won't be able to insert YAG plugins on your pages later!

So first of all, create a new sysfolder and name it "yag sysfolder" for example:



Now go to the page settings of this folder and mark the folder to contain yag records:



After that, reload the page tree, the icon of the folder should change to YAG icon:



You can now create your first gallery within this storage folder following the instructions given below.

Whenever you select a page that is not marked as a YAG page, you get a message like the following one:

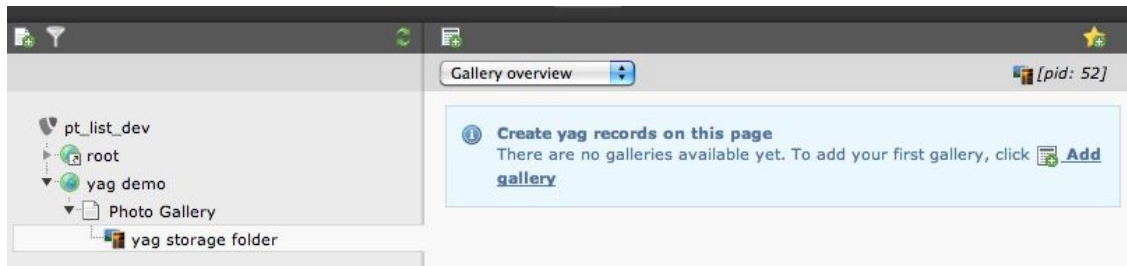


Now you have two possibilities: either you select a page which is already marked as a yag page using the select field or you use the link "Mark this page as a yag sysfolder" if you want to mark the page you currently opened as a yag page.

Without having a page marked as yag page, you won't be able to select the yag records in your content element plugins. If you want to upgrade from yag version 1.x.x where PIDs were not supported, refer to section "Upgrading from YAG 1.x.x to 2.x.x".

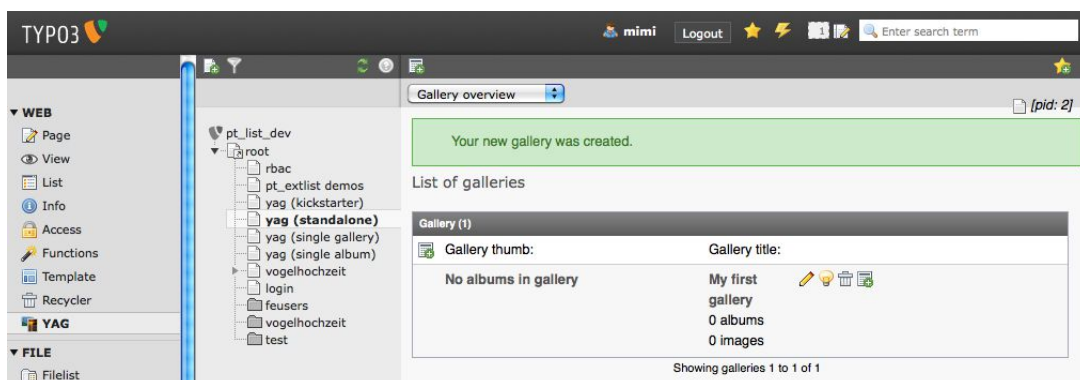
Setting up your first Gallery

After setting up a page for yag as described before, you will get a screen like this, if you open the page using the gallery module:



So let's get our first gallery created. Click on the 'Add gallery' link and you get a input form like this:

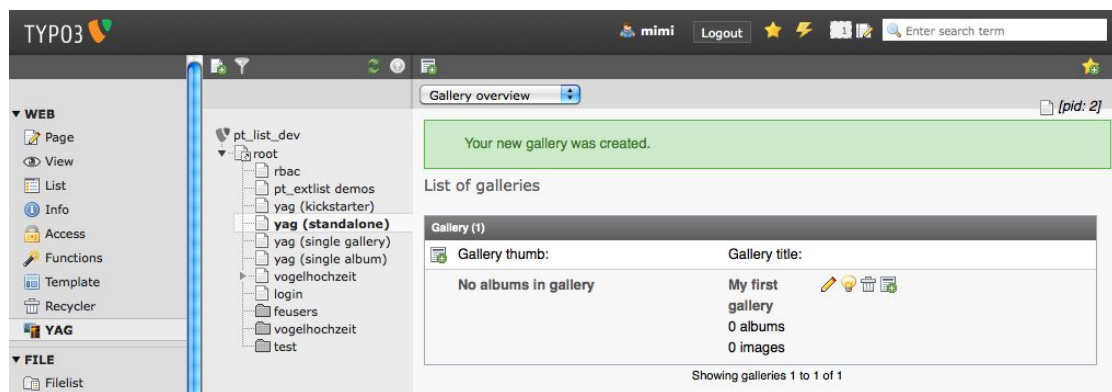
Fill in some text as name and description and press 'Save'. Now you should have a first gallery in your list:



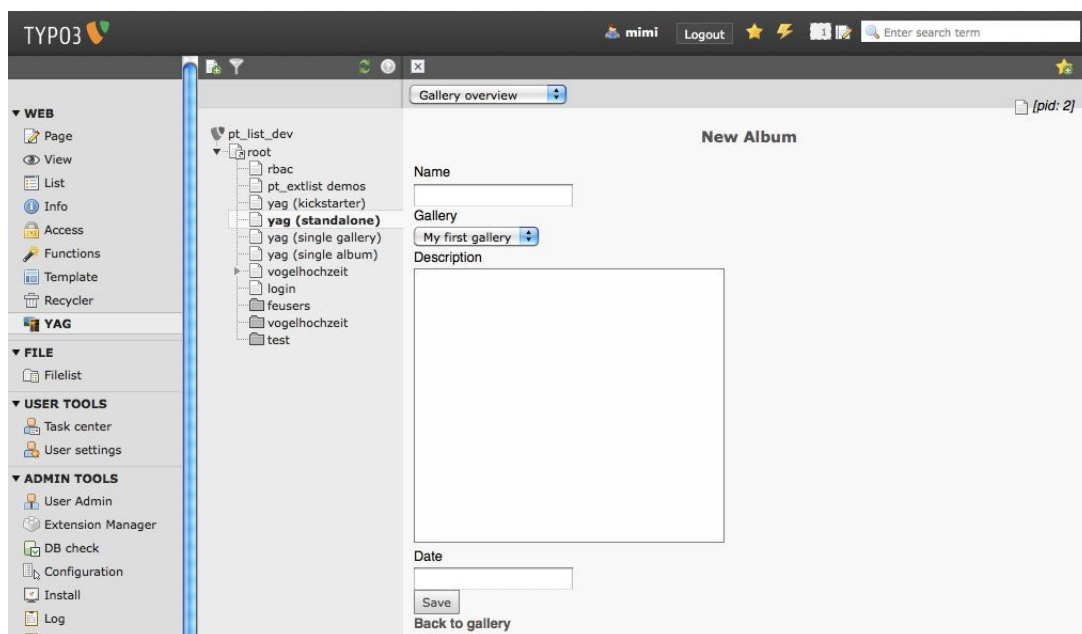
So that's it – you just created your first gallery. Let's go on and create some albums inside this gallery.

Setting up your first Album

After you successfully created a new gallery, you can now insert some albums. Therefore you can click on the green plus right beneath the garbage symbol in your gallery list:



You will get an empty form for creating a new Album. Fill in some text for Name and Description and click 'Save':



You will get an album list showing your newly created album:

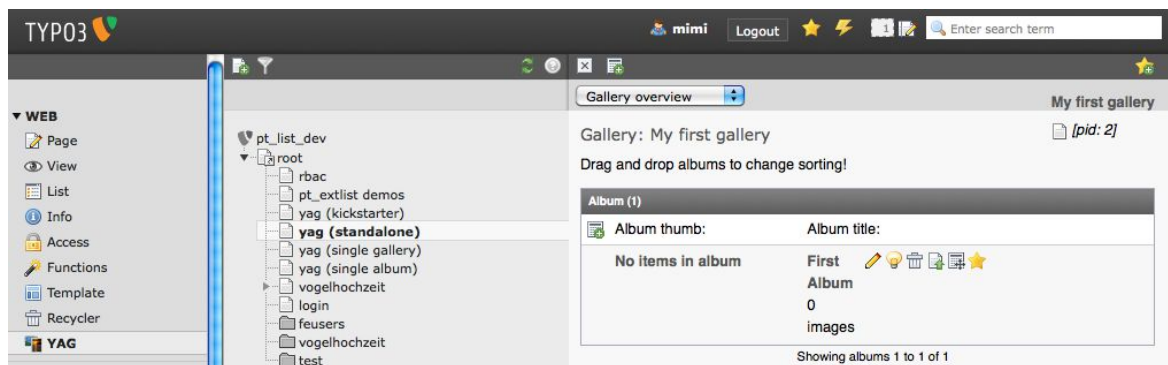
The screenshot shows a web form titled 'New Album' within a 'Gallery overview' window. The form contains the following fields and controls:

- Album name (required)**: A text input field.
- Gallery**: A dropdown menu with 'First Gallery' selected.
- Description**: A large text area.
- Date**: A date picker showing '7-10-2011'.
- Hide Album**: A checkbox.
- Buttons**: 'Back to gallery' and 'Save'.

Now we can start uploading some images into our album.

Uploading Images into Albums

There are several ways for uploading images. We show you uploading from album list. So make sure, you have selected a gallery and get a list of albums inside this gallery. Now click on the green arrow-up symbol next to the garbage-bin symbol:

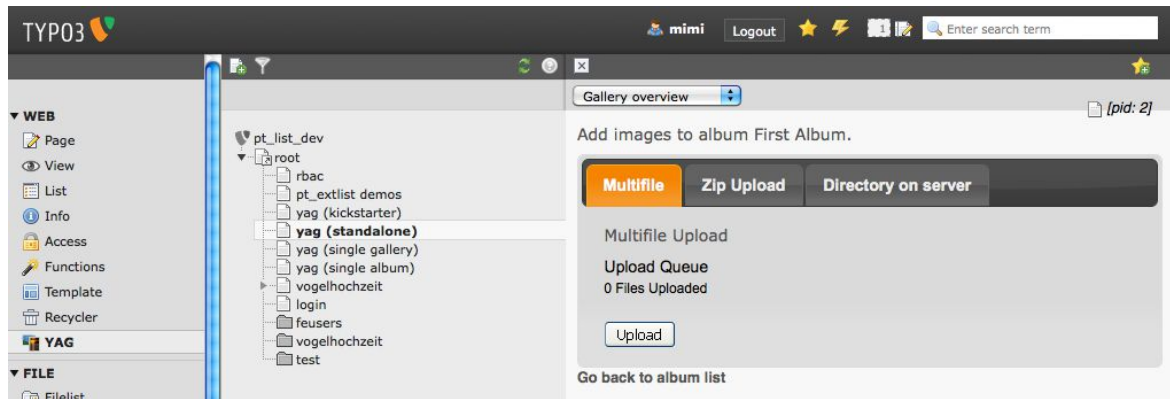


Drag & Drop Upload

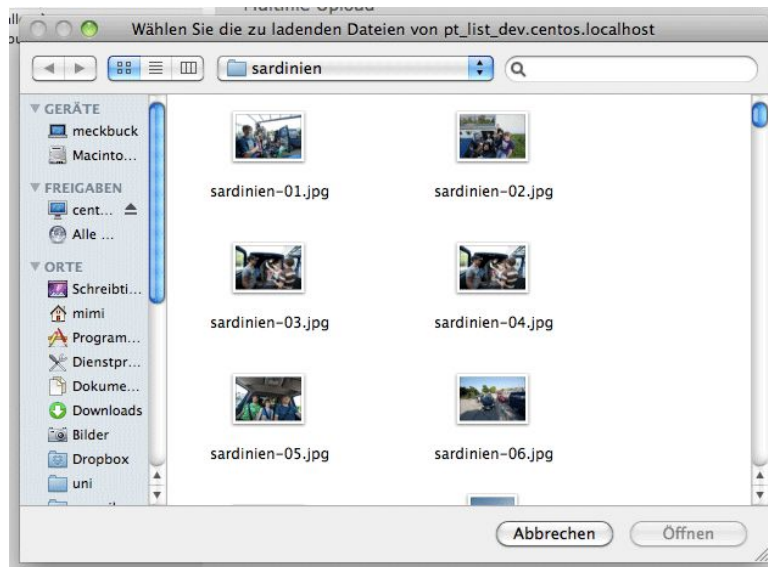
Using a HTML5 capable browser, you can use the Drag & Drop upload tab to upload you images by just dragging them from your filesystem into your browser.

Multifile Upload

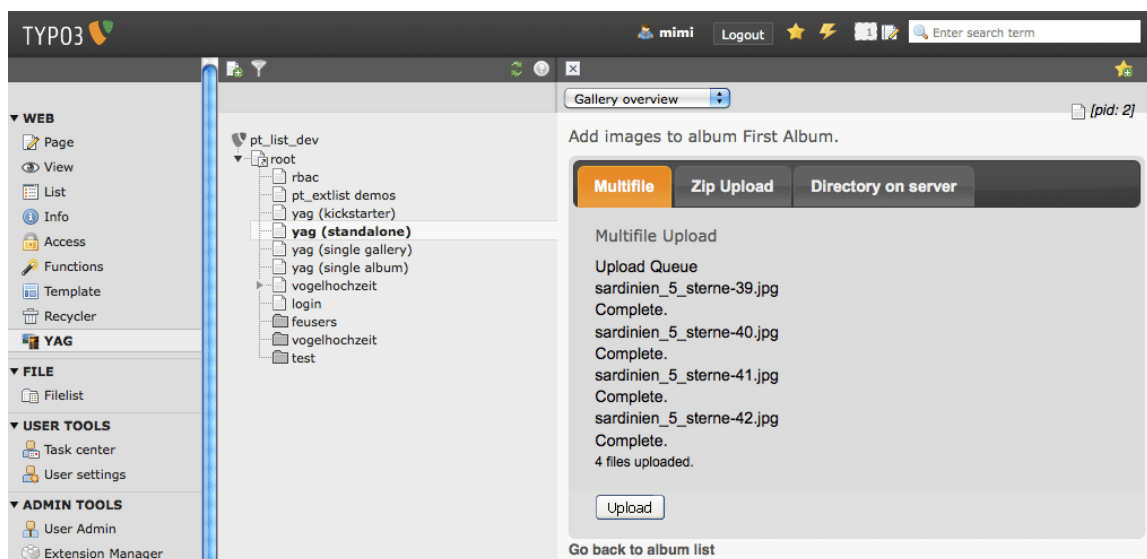
This will show up a form containing different possibilities for uploading images into your album. First of all we use the Multifile-Uploader:



Click on the 'Upload-Button' and you will get a file dialog opened in which you can select multiple files that should be uploaded to your gallery:



Select as many images as you like and click 'Open'. The uploading will start immediately. After you have finished uploading, click on 'Go back to album list':



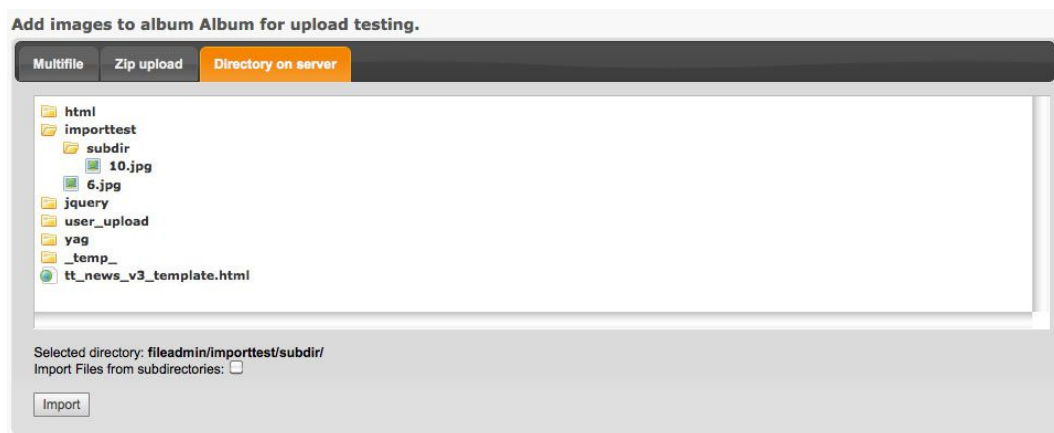
ZIP-File uploading

You can also upload images by putting them into a zip file and uploading this file. ZIP files may also contain folders. Any images in any folder of the ZIP file will be imported:



Importing images from directory on server

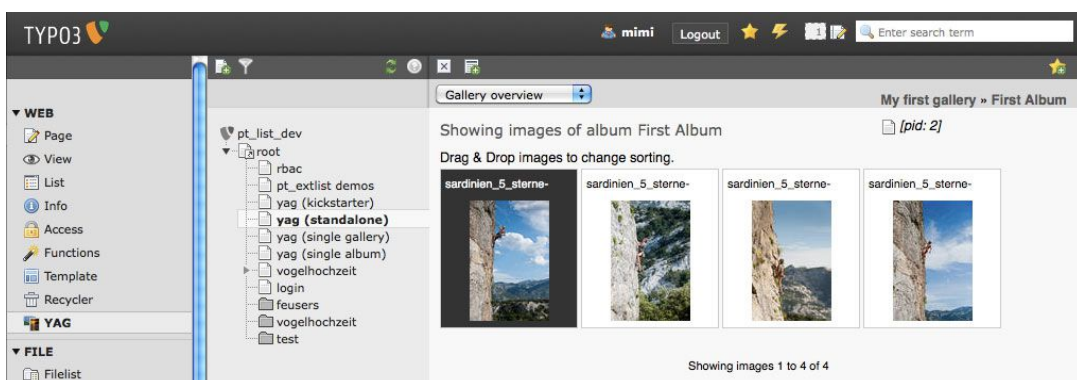
If you want to import images from a directory on your server, you can use the third import method:



The checkbox at the bottom let's you import files from subdirectories of the folder you chose.

Overview of imported images

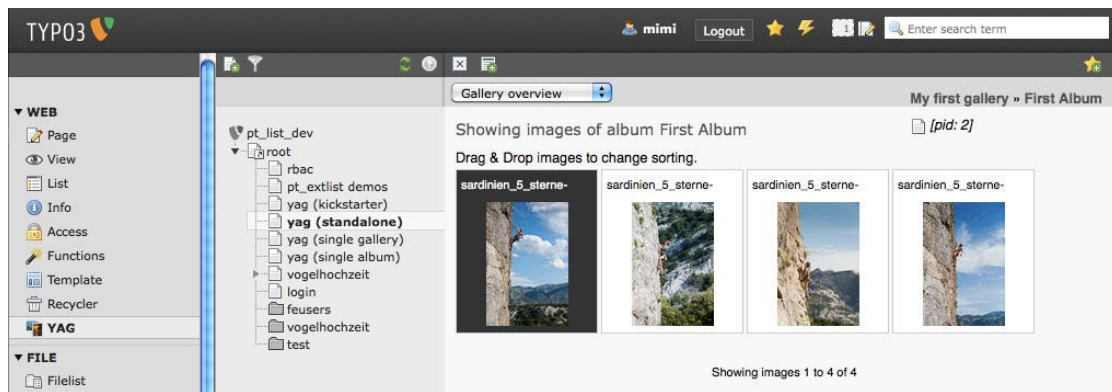
After importing items with any of the options a list will show up showing all the images just being uploaded to your album:



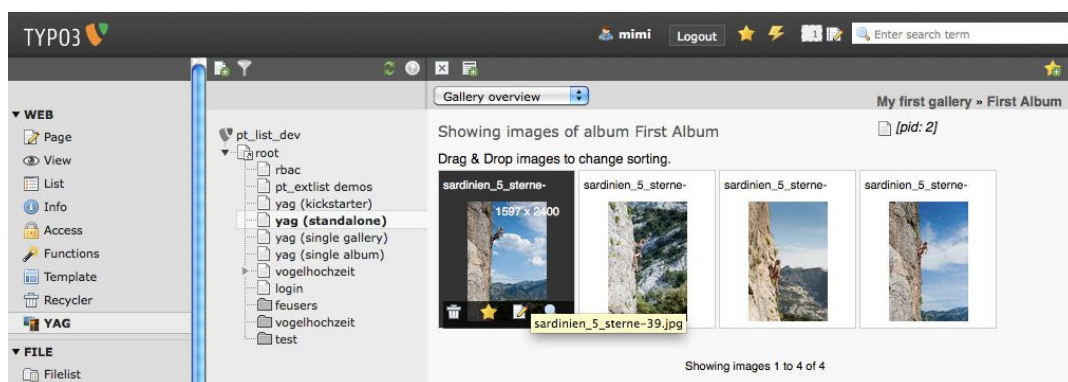
Congratulations – you just created your first album!

Editing Image data inside albums

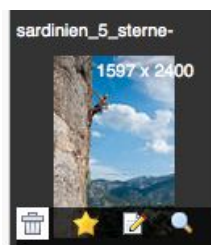
Open up the YAG module, select a gallery and an album within this gallery. You should now see something like that:



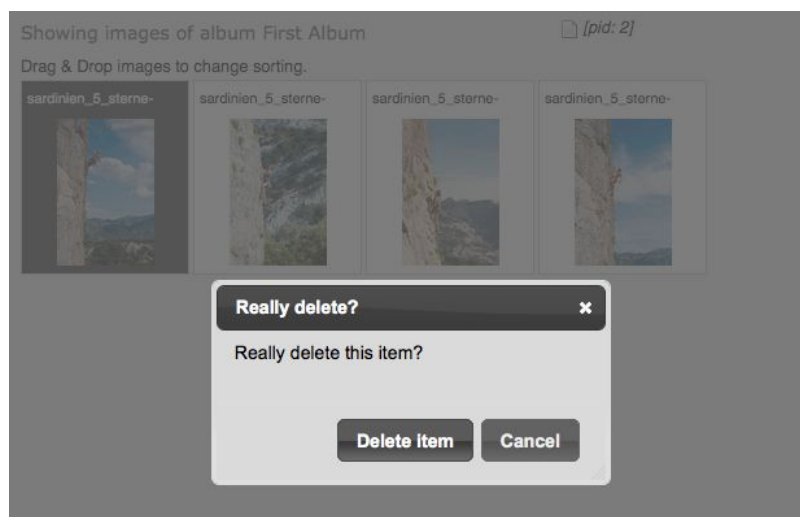
When you move your mouse over an image, a menu will fade in:



Besides the size of the image, you see some icons below your image, that let you manipulate it:

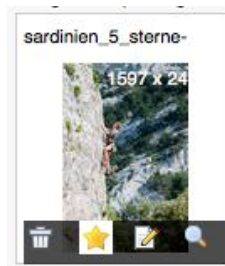


Using the garbage-bin, you can delete an image. Clicking on it, a dialog will appear and ask you again, whether you want to delete your image:

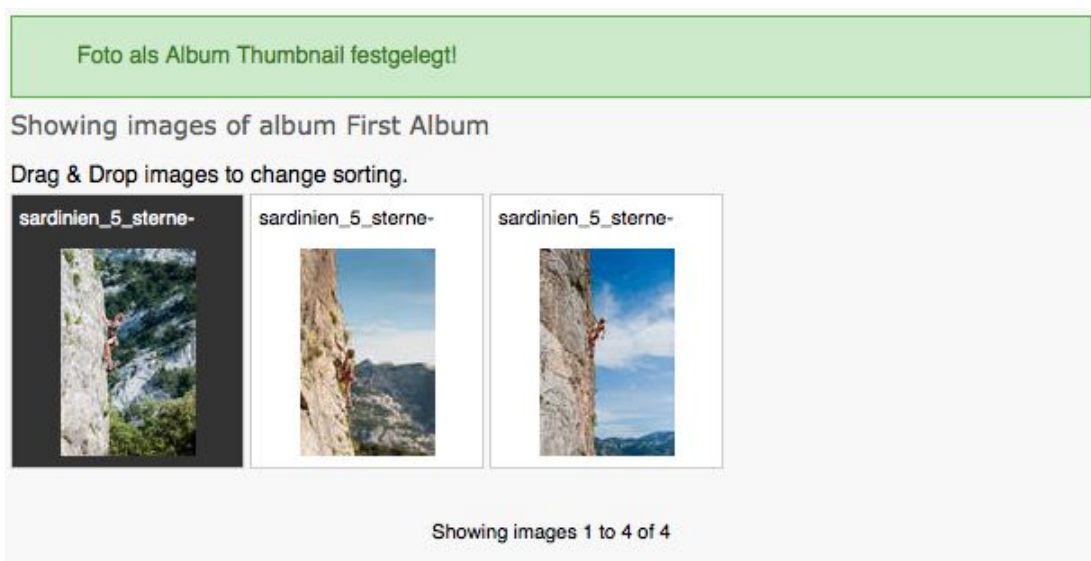


After clicking 'Delete item', the image will fade out and you will get a message, telling you what just has happened.

The yellow star sets your image as album thumb. Clicking on it will make the image the thumbnail image for you album:



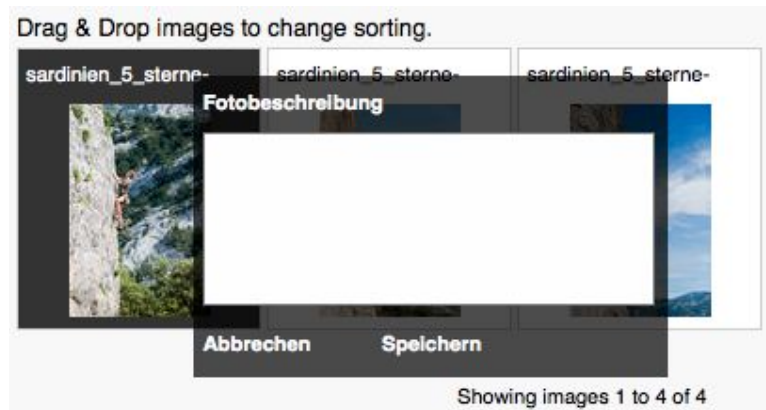
After clicking on it, your image will get a darker background which tells you, that this image is set as album thumbnail:



The edit icon lets you modify the description of your image:



Clicking on it will bring up a little form that lets you edit your image's description:



You can edit your image description and click 'Save' to save changes.

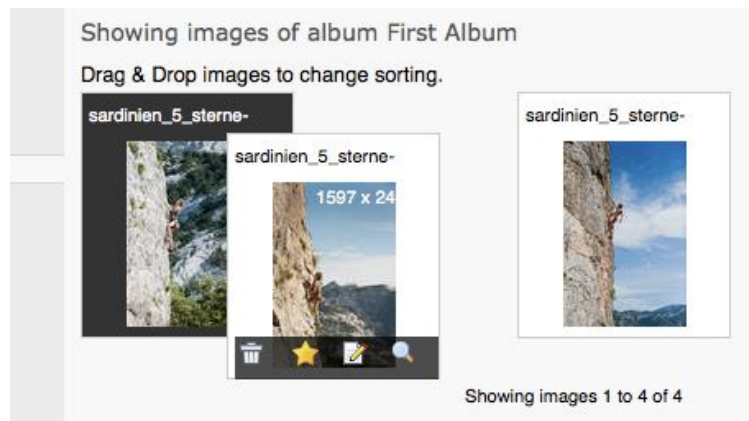
The last icon will open up a lightbox showing a bigger version of your image:



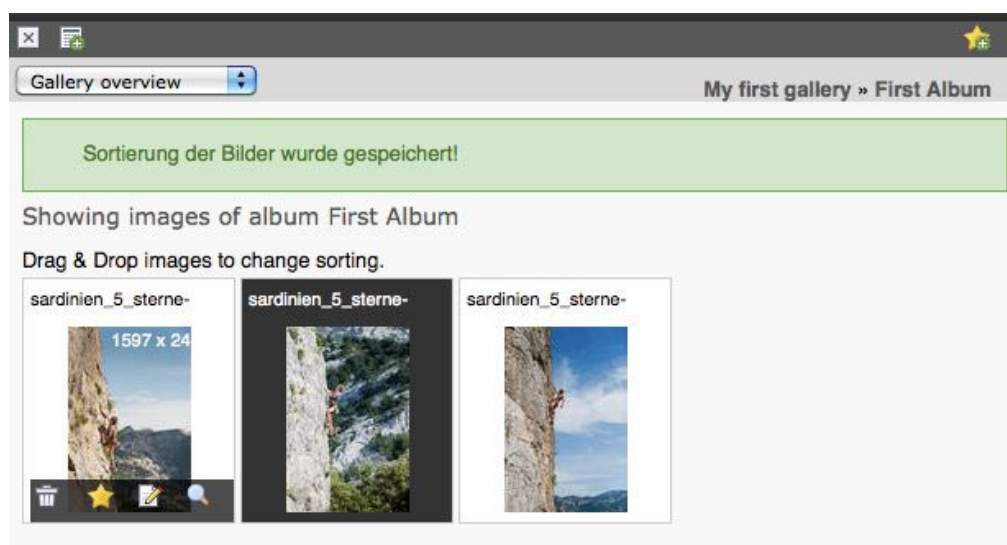
When moving over the heading of an image, the background will change to blue and by clicking the heading, you will be able to change the title of your image:



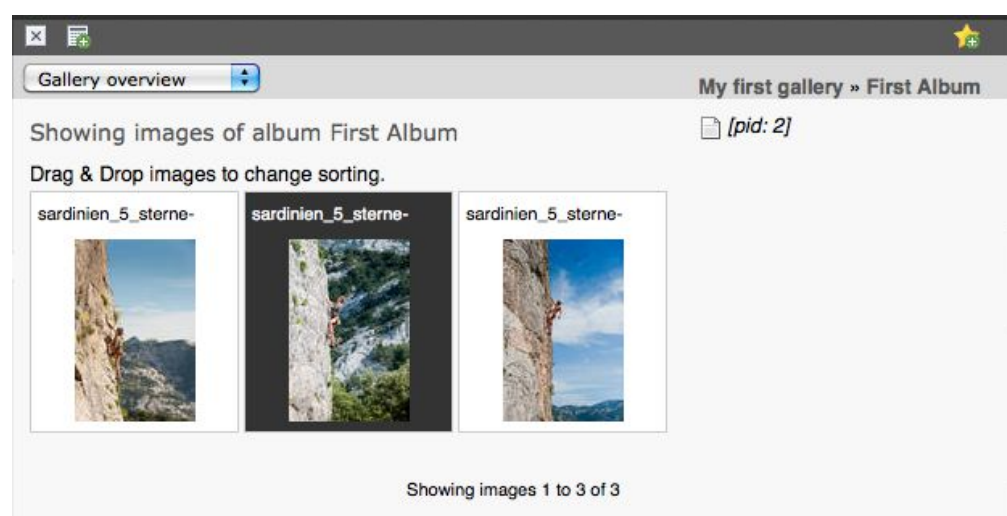
Last but not least, you can change the sorting of your images inside an album by dragging and dropping an image:



After dropping the image in your preferred position, the sorting will be saved:



Clicking on the 'x' Symbol in the upper left corner will bring you back to your album list:



Editing all images of an album at once




Within the tab „Edit Images“ you can edit all images of an album at once:

Edit Album: Another album

Images Overview **Edit Images** Album Settings

Edit all images of album Urlaub

Save

<p>Title: kletterhalle_bad_gross-010.jpg</p> <p>Description:</p> <p>Tags: baden-baden, kalle, kletterhalle</p> <p>Image size: 4256 x 2832 Filesize: 5.4 M Image path: fileadmin/yag/3/16.jpg</p>	 <p><input checked="" type="radio"/> Image is album thumb <input type="checkbox"/> Delete image Move to album: Another album</p>
<p>Title: kletterhalle_bad_gross-011.jpg</p> <p>Description:</p> <p>Tags: baden-baden, kalle, kletterhalle</p> <p>Image size: 3383 x 2251 Filesize: 3.0 M Image path: fileadmin/yag/3/17.jpg</p>	 <p><input type="radio"/> Image is album thumb <input type="checkbox"/> Delete image Move to album: Another album</p>
<p>Title: kletterhalle_bad_gross-012.jpg</p> <p>Description:</p>	


Sorting of images by attribute

You can change sorting of all images in an album by using sorting toolbar. Select a field by which you want to sort images:

Drag and Drop albums to change sorting!

Change sorting of images by: **Title** with direction: ascending Change Sorting! Current Sorting will be overwritten!

10.jpg 6.jpg kletterhalle_bad_gross- regenspaziergang- regenspaziergang- regenspaziergang-




Besides an attribute you can select a direction for sorting:

Drag and Drop albums to change sorting!

Change sorting of images by: Title with direction: **ascending** Change Sorting! Current Sorting will be overwritten!

10.jpg 6.jpg kletterhalle_bad_gross- regenspaziergang- regenspaziergang- regenspaziergang-



Remind that setting sorting by attribute will overwrite manual settings.

Editing Albums

Your album list comes with some icons for editing:



This lets you edit your albums data in a form:

Edit album First Album

Name

Gallery

Description

Date

[Back to Album](#)



The lamp will hide / unhide an album in the frontend. If you click on it, it will change to a switched-off lamp and the icon of the album will get lighter to mark it as hidden:



The Garbage-bin will delete your album.



The green arrow will open up an upload dialog that lets you upload some images to your album.



This will open up the image list that lets you change the sorting of your images.



The star sets your album as thumbnail for the gallery. **Please note that the background of your album entry will**

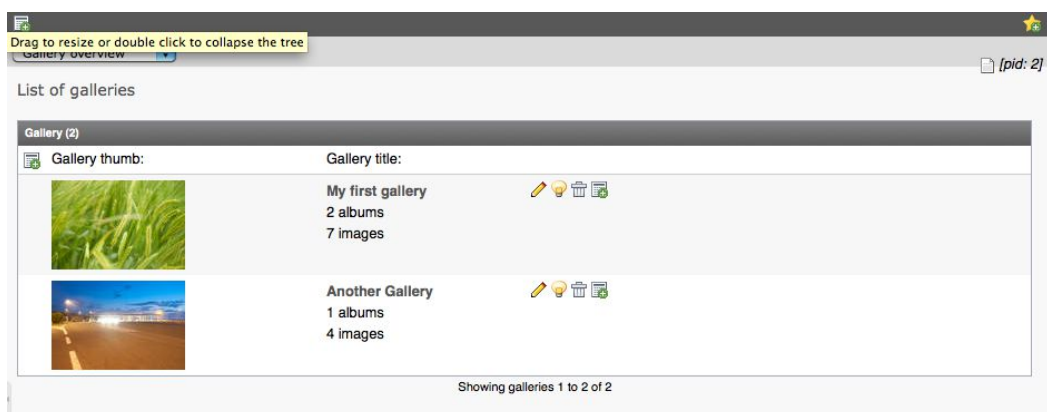
turn yellow, after you set it as gallery thumb.



You can add a new album to your gallery by using this button.

Editing Galleries

The first screen you see when starting the YAG backend module is a list of galleries, available on your site:



You can edit a gallery by clicking on the pencil icon. This will open a form that lets you edit your gallery's information:

Another Gallery

Name(required)

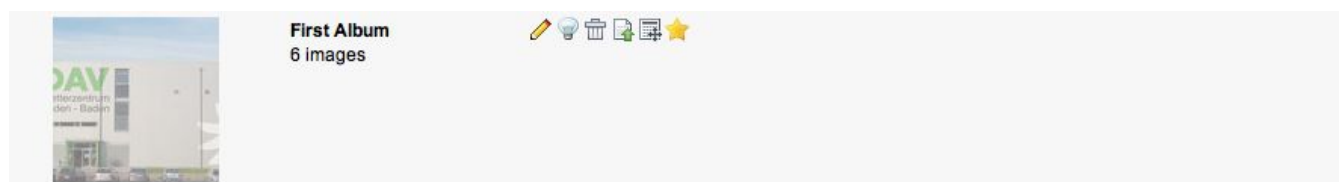
Another Gallery

Description

Save

Back to gallery

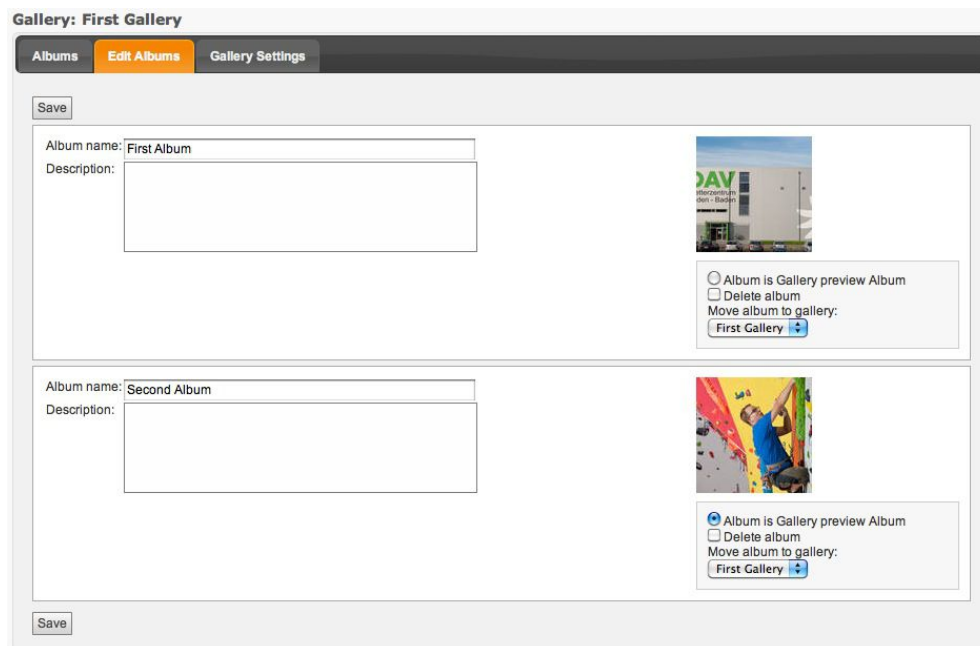
Again, the bulb-icon hides and unhides your gallery. A hidden gallery will no longer be displayed in frontend. Hidden galleries are displayed with a switched-off light bulb and a half-transparent thumbnail:



Use the garbage-bin to delete a gallery and the green plus to add a new album.

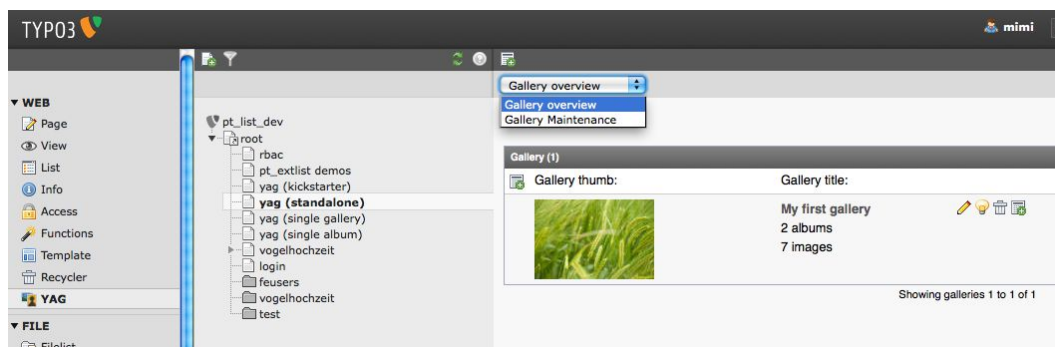
Editing all albums of a gallery at once

Within the tab „Edit Albums“ you can edit all albums of a gallery at once:

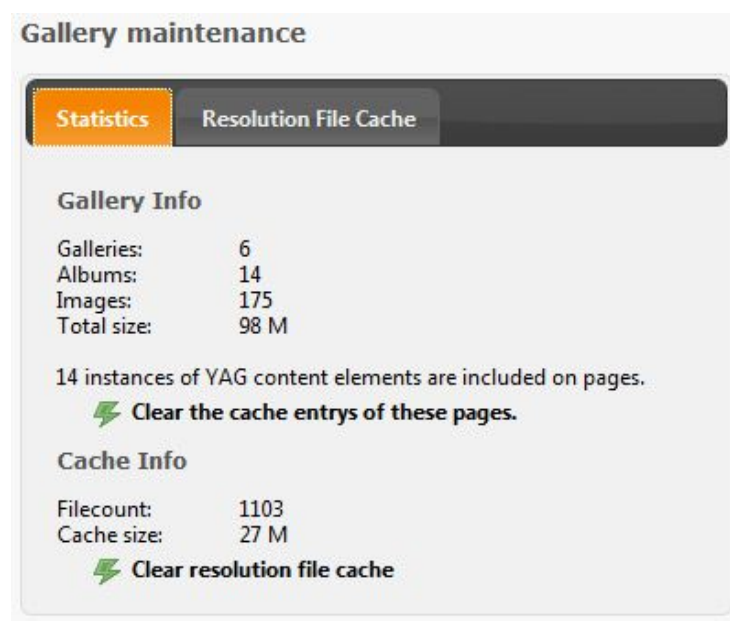


Gallery Maintenance

There is a section for gallery maintenance in the backend. You can open it via the dropdownlist in the YAG module:



Select 'Gallery Maintenance' there and you will see the following screen:

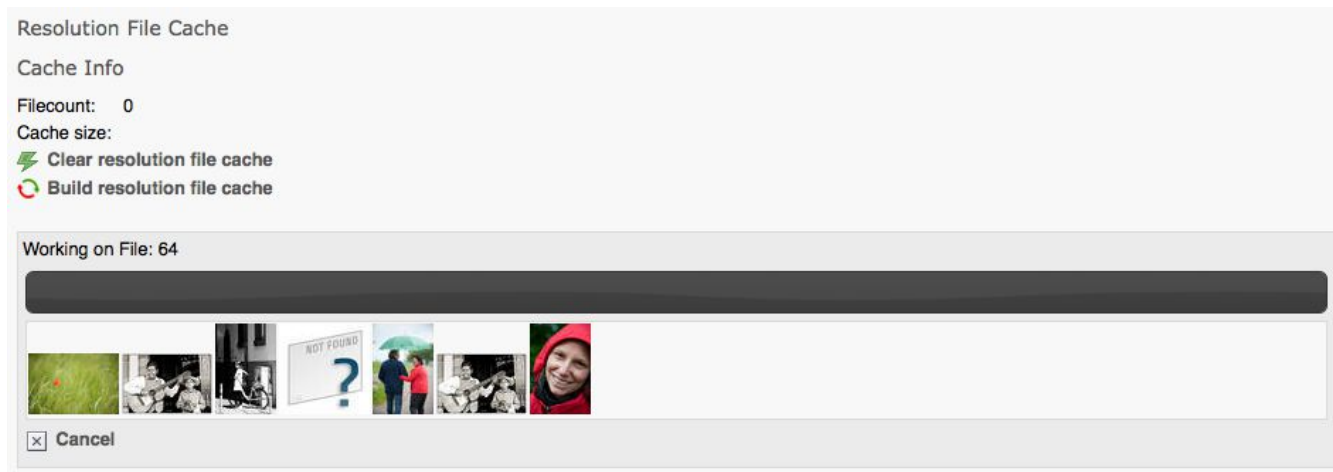


Besides a statistics of galleries, albums and images on your site, there is also some information about cached files and the amount of data used for image files on your harddisk.

The last two links of the site are used for maintaining the cached files. The cache is filled automatically if an image is requested in a specific size and configuration.

When clicking on 'Clear resolution file cache', a message informs you, that the cache has been deleted.

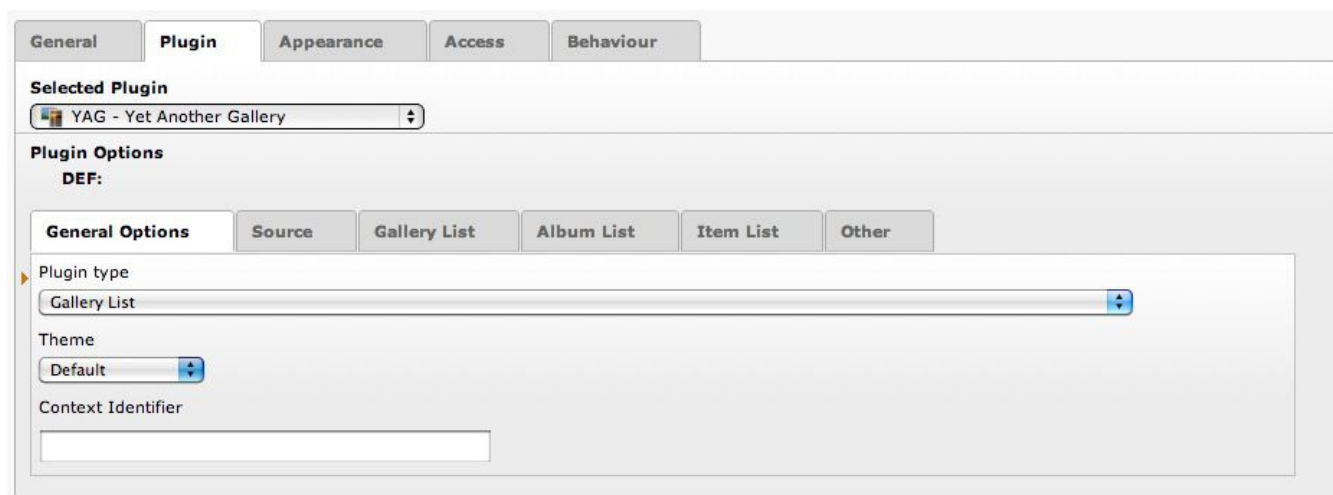
You can build all resolutions at once, this speeds up the page creation in frontend for the first user of your site. When you click on 'Build resolution file cache', there is a progress bar, that shows you the progress of rebuilding the cache:



Setting up Frontend-Editing with YAG

Since version 2.0 we have re-introduced frontend editing in yag. Here comes a short introduction on how to set up frontend editing.

First insert a plugin with type "Gallery List" with theme "Default":



After that you have to make some changes in the TypoScript settings for the page on which you want to enable FE editing. You will find the configuration for the role based access controll in EXT:yag/Configuration/TypoScript/Frontend/Security.ts. This is automatically included with the yag static template.

The next thing you have to do is create Frontend-User-Groups that will be enabled to edit your yag contents in the frontend. Create a group and set

```
plugin.tx_ptextbase.settings.rbac.extensions.yag.feGroups {
    # Do the group <=> role asignment here
    #<groupUId> {
    #     10 = <ROLENAME>
    #     20 = <AnotherRoleName>
```

```
#}

1 {
    10 = admin
}

}
```

In our example, we made the user group with UID 1 to be an admin on our site. If you want to get an impression on what you can do with the RBAC settings in the security.ts file, take a look at the file. It is self-explaining and well documented.

That's it. If you open up the frontend now, you will see some additional links that let you edit the gallery content:

Alle Galerien



Cambodia

1 Alben



Hamburg

1 Alben

Zeige Galerien 1 bis 2 von 2

[Galerie hinzufügen](#)

There are more links, once you select a gallery:

Alle Galerien » Cambodia



Phnom Penh

14 Bilder

Zeige Alben 1 bis 1 von 1

[Galerie bearbeiten](#)

[Album hinzufügen](#)

Once you opened an album, you have a link for uploading images:



cambodia_1-013.jpg



cambodia_1-014.jpg

Zeige Bilder 1 bis 14 von 14

Bilder hinzufügen

Album bearbeiten

Album löschen

RSS Feeds

YAG is capable to generate RSS a RSS feed from your image list. To include the feed into your template add the following configuration to your theme typoscript:

```
plugin.tx_yag.settings.themes.default.feed.active = 1
```

Visit the typoscript reference for a detail description of the rss configuration options.

Get feedback for your images

The default theme offers some easy to configure user-feedback and social network widgets.

Add social share buttons

YAG offers an easy to include lightweight privacy aware social widget. All you have to do to add some social share buttons beneath your images is to activate the widget using a line of typoscript:

```
plugin.tx_yag.settings.themes.default.item.interaction.socialSharePrivacy.show = 1
```



Additional social networks can be configured, including delicious, flattr, linkedin and many others. Visit the typoscript reference for details.

Comments with Disqus

With the Disqus comments widget, you can easily get feedback for your images. To activate the widget via typoscript:

```
plugin.tx_yag.settings.themes.default.item.interaction.disqus {
    settings.disqus_shortcode = YourDisqusInstanceName
    show = 1
}
```

Installation

This chapter will give you a step-by-step introduction on how to install the extension.

Dependent extensions

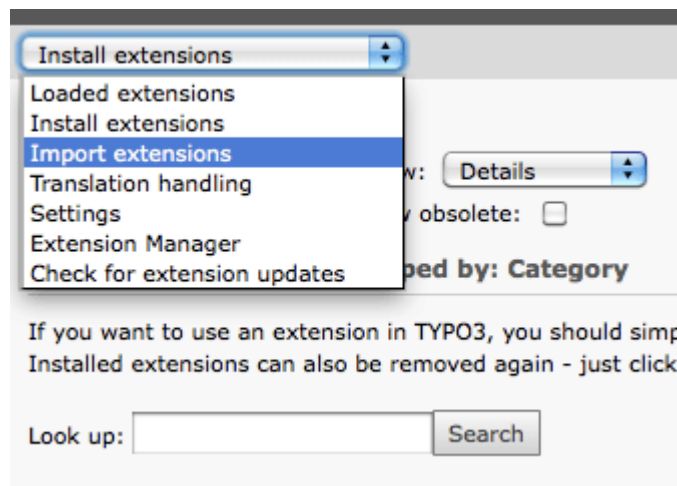
yag depends on two other extensions which you have to install first:

- pt_extbase – a collection of libraries used throughout this extension.
- pt_extlist – a list generator used to render all kinds of lists, pagers and filters.

In order to install YAG, you have to set up those two extensions in the order given above (pt_extbase, pt_extlist). After that, the installation of YAG is quite straight-forward.

Importing YAG from TER

Open up the Extension Manager and select 'Import Extension' from the menu:



Search for 'yag':





Extensions in the TYPO3 Extension Repository (online) - Grouped by: Category									
Look up extensions:									
yag <input type="button" value="Look up"/>									
	Title	Extension key:	Version	Upload date:	Author	Cur. Ver:	Cur. Type:	DL:	State
Frontend									
	Simple Page Header Selector - Extended	bs_headerselector_yags	0.0.1	18-05-08	Markus Wanjura			56/56	Beta
Frontend Plugins									
	YAG Theme Simpleviewer	yag_theme_simpleviewer	0.1.1	24-06-09	Markus Wanjura			31/30	Beta
	YAG Theme Simpleviewer	yag_theme_simpleviewer	0.1.1	07-03-11	Daniel Lienert			4/4	Beta
	Yet Another Gallery	yag	1.0.2	08-03-11	Michael Knoll, Daniel Lienert	1.0.2		16/8	Beta

Click on the install button:

Extensions in the TYPO3 Extension Repository (online) - Grouped by: Category

Look up extensions:

yag

	Title	Extension key:	Version	Upload date:	Author	Cur. Ver:	Cur. Type:	DL:	State
Frontend									
	Simple Page Header Selector - Extended	bs_headerselector_yags	0.0.1	18-05-08	Markus Wanjura			56/56	Beta
Frontend Plugins									
	jk_poll extended	yags_jk_poll_extended	0.1.1	24-06-09	Markus Wanjura			31/30	Beta
	YAG Theme Simpleviewer	yag_theme_simpleviewer	0.1.1	07-03-11	Daniel Lienert			4/4	Beta
	Yet Another Gallery	yag	1.0.2	08-03-11	Michael Knoll, Daniel Lienert	1.0.2		16/8	Beta

There will be a message informing you, that additional information has to be given:

Extension Manager


Extension:  Yet Another Gallery (yag)

 Installing  Yet Another Gallery: DATABASE NEEDS TO BE UPDATED


Before the extension can be installed the database needs to be updated with new tables or fields.
Please select which operations to perform:

 This extension provides additional configuration options which become available once it is installed.


Click on 'Make updates' and you will get a form you have to submit:

 The extension "yag" has been installed.

Extension Manager

Extension:  Yet Another Gallery (yag)

Current status:

The extension is installed (loaded and running)!
 Click here to remove the extension: 

Configuration:

(Notice: You may need to clear the cache after the configuration of the extension. This is required if the extension adds TypoScript depending on these settings.)

Path of directory where YAG should ... [hashFilesystemRoot]
 Path of directory where YAG should store all cached images generated for albums - relative to Typo3 base path. This path should not be changed unless you know what you do! (Image base path)

Path of directory where YAG stores ... [origFilesRoot]
 Path of directory where YAG stores all original files.


Importing TypoScript RBAC settings ... [updateMessage]
 Importing TypoScript RBAC settings into database
 RBAC data has been imported into database.

There are two settings you have to make:


- **Path of directory where YAG should ...[hashFilesystemRoot]:** YAG will create a hashed image for each image shown in different sizes. Those cached files need to be stored on your server. You have to determine here, which folder will be used to do that. Default is typo3temp/yag
- **Path of directory where YAG stores ...[origFilesRoot]:** YAG will store each original file on your server. You have to determine here, where YAG should store original files on your server. Default is fileadmin/yag

Click 'Update' to confirm those settings. A message will be shown informing you about the state of the installation:

Extension Manager

Extension:  Yet Another Gallery (yag)

Current status:

The extension is installed (loaded and running)!
 Click here to remove the extension: 

That's it. You have now successfully installed YAG on your server.

Installing additional themes

To shape the look-and-feel of your gallery, you can use different themes available in TER. Installation is straight forward. Go to the Extension Manager and search for 'yag'. There should be at least one theme:

Extensions in the TYPO3 Extension Repository (online) - Grouped by: Category

Look up extensions:

	Title	Extension key:	Version	Upload date:	Author	Cur. Ver:	Cur. Type:	DL:	State
Frontend									
	Simple Page Header Selector - Extended	bs_headerselector_yags	0.0.1	18-05-08	Markus Wanjura			56/56	Beta
Frontend Plugins									
	YAG Theme Simpleviewer	yag_theme_simpleviewer	0.1.1	07-03-11	Daniel Lienert			4/4	Beta
	Yet Another Gallery	yag	1.0.2	08-03-11	Michael Knoll, Daniel Lienert	1.0.2		16/8	Beta

Use the install button to import the extension on your server:

Extensions in the TYPO3 Extension Repository (online) - Grouped by: Category

Look up extensions:

	Title	Extension key:	Version	Upload date:	Author	Cur. Ver:	Cur. Type:	DL:	State
Frontend									
	Simple Page Header Selector - Extended	bs_headerselector_yags	0.0.1	18-05-08	Markus Wanjura			56/56	Beta
Frontend Plugins									
	YAG Theme Simpleviewer	yag_theme_simpleviewer	0.1.1	07-03-11	Daniel Lienert			4/4	Beta
	Yet Another Gallery	yag	1.0.2	08-03-11	Michael Knoll, Daniel Lienert	1.0.2		16/8	Beta

After import, a message will be shown and you can install the theme click on 'Install extension':

Extension Manager

Extension import results

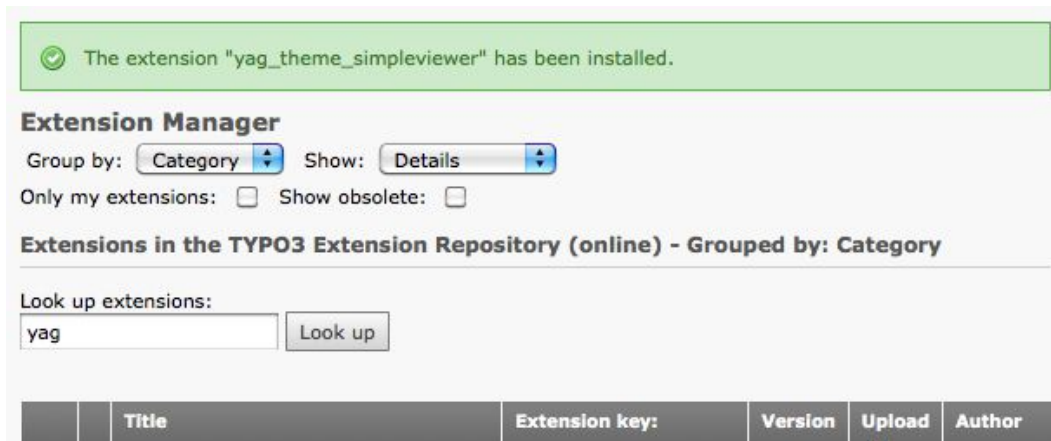
Extension imported

Folder created: /var/www/kunden/pt_list_dev.centos.localhost/typo3conf
 /ext/yag_theme_simpleviewer/
 ext_emconf.php: /var/www/kunden/pt_list_dev.centos.localhost/typo3conf
 /ext/yag_theme_simpleviewer/ext_emconf.php
 Installation Type: Local

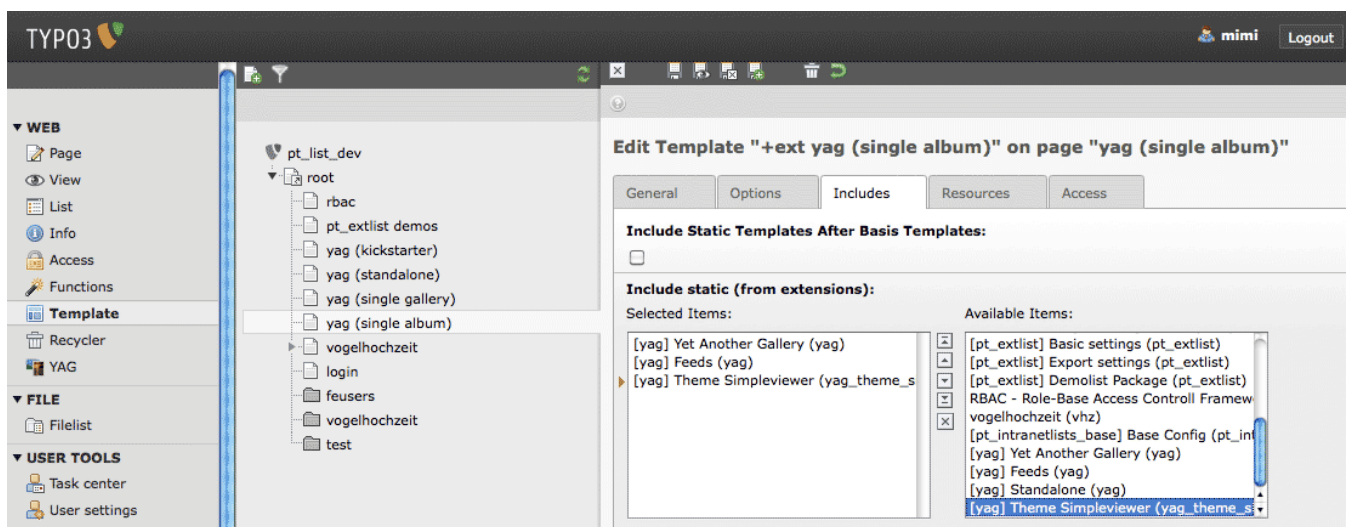
Install/Uninstall Extension:

Install extension

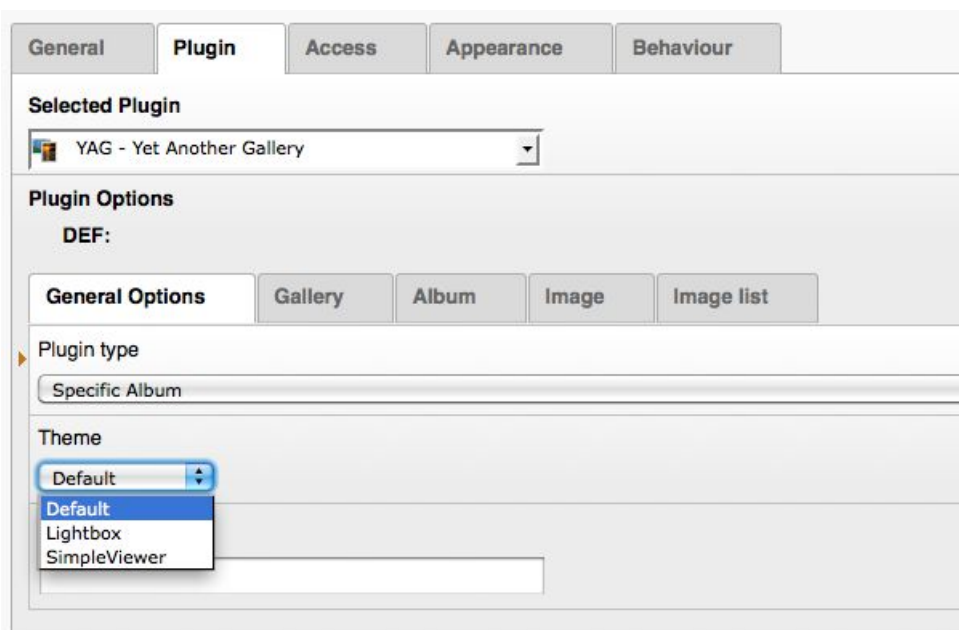
There should be a message, telling you that the extension has been installed:



Now you have to include the theme's static template on the page you would like to use the theme. This is done via Template-Module on the page you want to have the new theme:



Right after that, you should be able to select the theme from within your FlexForm of the content element that includes your gallery widget:



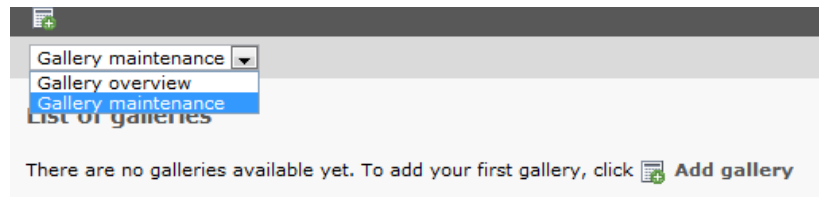
Upgrading from YAG 1.x.x to 2.x.x

YAG version 2.0 and above uses Sysfolders to store its records instead of storing the data on PID 0. For that reason you have to do some manual adjustments when upgrading from version 1.x to 2.x.

Before you proceed with the steps described below, make sure, you have created a sysfolder for yag as described in section "Setting up a sysfolder for YAG". Without this sysfolder, YAG won't work properly anymore!

After installing the current YAG, pt_extlist and pt_extbase you first have to create a Sysfolder to store the existing YAG records in. Refer to the section Installation for further details.

Select the "Gallery" Module and select the Sysfolder from the pagetree. The module shows the message "There are no galleries available yet.". Chnage to the Gallery Maintenance Module using the module selector:



The Maintenance Module now shows the database update wizzard:

YAG database update needed!

It seems you are currently running YAG version 2.0-dev but the YAG database is a version smaller or equal 1.5.0.

Target PID:
 In YAG prior version 2.0 all records are kept on PID 0. Starting with version 2.0 YAG, you are able to separate YAG galleries by sysfolders. Please specify the page Id of an existing sysfolder to move the existing records to.

The wizzard again explains what to do. Simply enter the PID of the created Sysfolder and all records are moved there.

The last step you have to do, is to manually open every YAG instance on your pages, and select the PID the records are now stored on (hopefully you have not too much ;)).

Administration

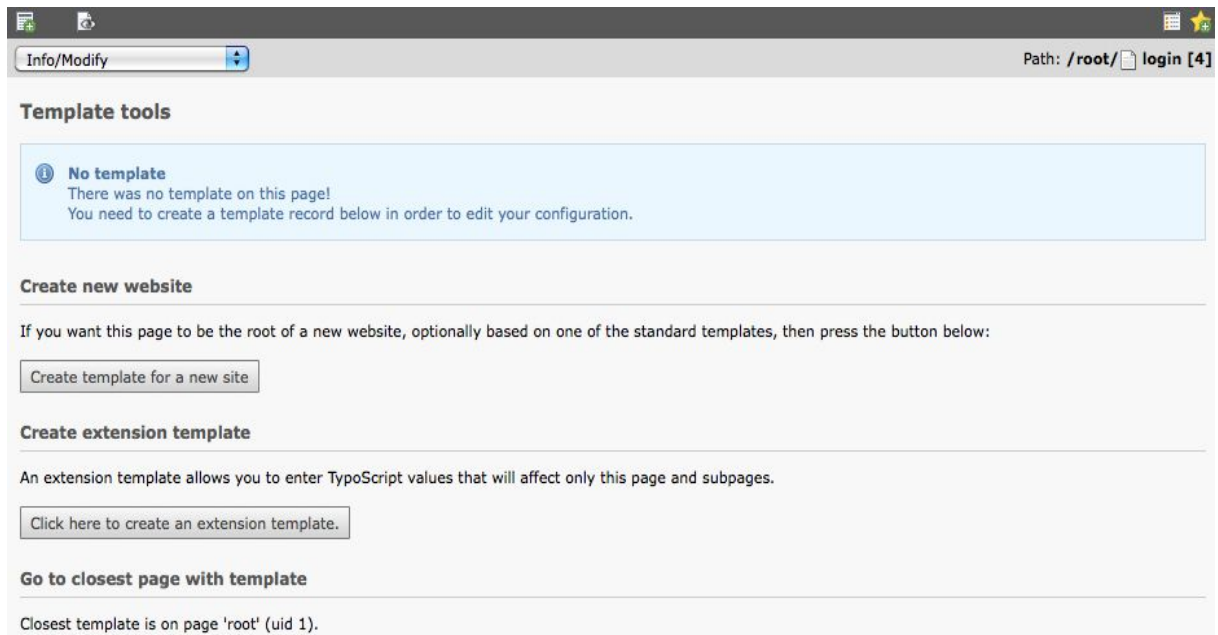
This chapter gives you a step-by-step introduction on how to set up a page hierarchy and templates to bring YAG on your site.

Setting up YAG for standalone usage

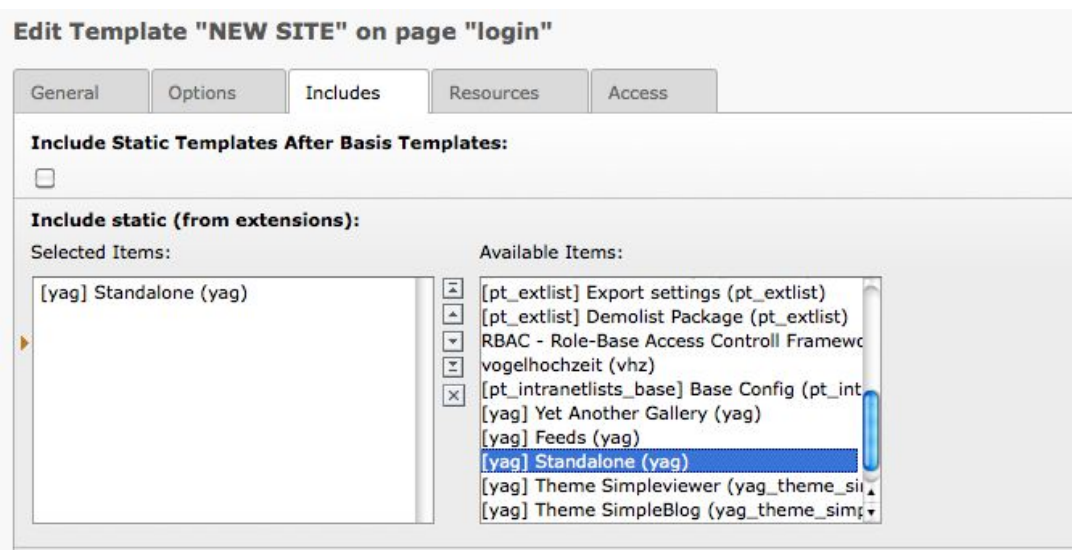
WARNING: The standalone theme of YAG is currently not maintained anymore – so use this with care!

YAG ships with a TypoScript Template that lets you install YAG standalone, without requiring a page hierarchy or anything. You only have to set up a single page in backend and include the standalone TS-Template. Here are the steps:

1. Create a new, empty page where you want to have YAG standalone installed.
2. Create a new TypoScript Template using the template module on this page. Click "Create template for a new site"



3. Within your new template, remove anything from "Setup" and open the tab "Includes". Select "Standalon (yag)" from Include statics:

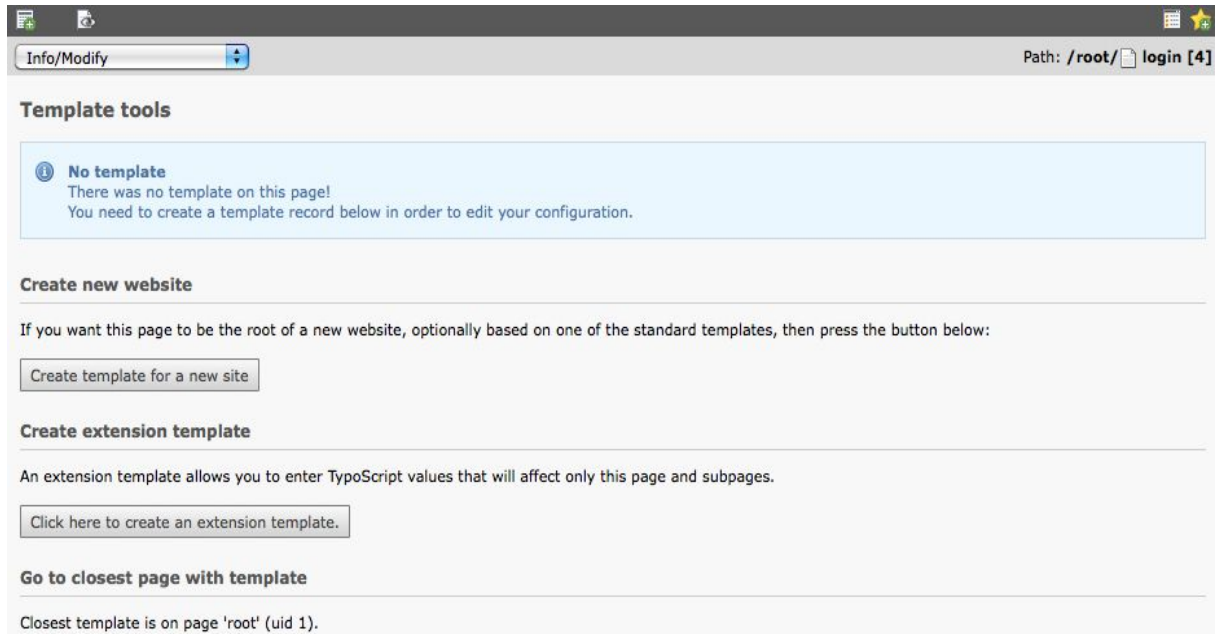


4. Save your TS template and you are done!

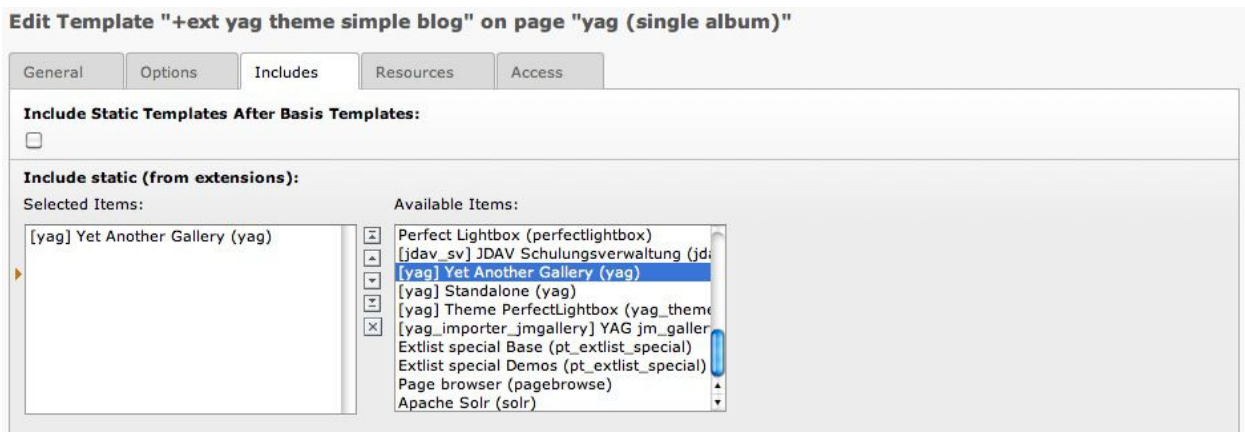
Setting up YAG for usage as content-element

If you want to use YAG as content elements, here are the steps, you have to follow:

1. Create a new page on which you want to include YAG.
2. Create an extension template using Template module:



3. Edit the extension template and switch to the "Includes" tab. Include "Yet Another Gallery (yag)":



4. Save your TS template.
5. Open the "Page" module and go to the page you just created. Insert a page content element. From the list of contents, chose "Plugins → YAG – Yet Another Gallery":



6. Switch to the "Plugin" tab and you can configure your content element:

Setting up a random item list

1. General Configuration

First you select the Plugin Mode "ImageList (Uncached)". Hence the name and unlike the other YAG modes, this mode is uncached and the HTML is rendered on every reload. This is slower, than using the TYPO3 Cache, but necessary to pick new random images on every reload.

For another new function described later in the article it is necessary to define a context. This can be every string you like, in this example I choose yag.

2. Configure The ItemList

The Itemlist configuration received some new options in version 2.3. For our purpose, the "Items per Page" option defines how many random images are displayed.

There is also a new "Filter" selection at the bottom. Here I added a filter to pick random images. Check this option.

That is all you have to do. With this configuration, a set of 4 images is randomly picked from your images and displayed. A click on the image opens a detail view of the image on the same page with the default theme or renders a lightbox with the lightbox theme.

A common use case is to use a list of random images as a teaser and then jump to the containing album when clicked on an image. To configure this behavior, you can use the new "Image Link" section. The default "Image Link Mode" is to use the configuration from the theme. When you set it to "Detail Page", the image links to a detail page which can be chosen with the next option.

The option "Plugin Mode on Target Page" defines which YAG mode should be triggered on the target page. You have the option to show the gallery or the album containing the image, or just show the detail view of this image on the target page.

Here comes the context identifier into play. To make this work, you have to set the context identifier of the YAG instance on the target page to the same as in the random list.

Configuration possibilities in FlexForm

Tab "General"

- **Plugin type** lets you chose what you want the plugin to do:
 - **Gallery** list shows a list of galleries
 - **Specific gallery** shows a single gallery as defined in „Source“ Tab
 - **Specific album** shows a single album as defined in „Source“ Tab
 - **Specific image** shows a single image as defined in „Source“ Tab
 - **AlbumList** shows a list of albums.
 - **ImageList** shows a list of images.
 - **RandomSingle** shows a single random image from sources defined in „Source“ Tab.

- **Theme** lets you chose the theme you want to use to style your gallery
- **Context Identifier** whenever you want to put more than one gallery plugin on a single page, you have to set up a context to make them work independently.

Tab "Source"

You can select a source for your content element here. Depending on whether you want to show gallery list, gallery, album or single image, you have to select gallery, album or image respectively. The first tab show all sysfolders / pages that are marked as YAG pages. If no pages are displayed in this column, refer to section "Setting up a sysfolder for YAG".

Tab „Gallery List“

You can set up gallery lists within this tab:

- **Items per page:** You can define, how many galleries should be shown on a single page. If you have more galleries, a pager will be used to limit the results.
- **Sort gallery list by:** You can set up which field you want to use for sorting galleries on a gallery list:

- **None (respect sorting from theme):** In this mode, the sorting defined in the current theme will be taken to sort galleries.
- **Custom sorting (Sorting of galleries in backend):** Will use the sorting defined in backend module for sorting galleries.
- **Title:** Will use the gallery title for sorting
- **Date:** Sort galleries by date gallery has been created in backend.
- **Description:** Sort galleries by its description field.

Tab „Album List“

The screenshot shows the 'Plugin' tab selected in the administration interface. Under 'Selected Plugin', 'YAG - Yet Another Gallery' is chosen. In the 'Plugin Options' section, the 'DEF:' sub-section has tabs for 'General Options', 'Source', 'Gallery List', 'Album List' (which is active), 'Item List', and 'Other'. Within the 'Album List' tab, there is a text input for 'Items per page', a dropdown for 'Sort album list by' set to 'None (Respect sorting of theme)', and a dropdown for 'Ascending'.

You can configure album lists in this tab. See section about gallery tab for details.

Tab „Item List“


The screenshot shows the 'Plugin' tab selected in the administration interface. Under 'Selected Plugin', 'YAG - Yet Another Gallery' is chosen. In the 'Plugin Options' section, the 'DEF:' sub-section has tabs for 'General Options', 'Source', 'Gallery List', 'Album List', 'Item List' (which is active), and 'Other'. Within the 'Item List' tab, there is a text input for 'Items per page', a dropdown for 'Sort image list by' set to 'None (Respect sorting of theme)', and a dropdown for 'Ascending'.

You can configure item lists in this tab. See section about gallery tab for details.

Tab „Other“

General	Plugin	Access	Appearance	Behaviour
---------	---------------	--------	------------	-----------

Selected Plugin


 YAG - Yet Another Gallery

Plugin Options

DEF:

General Options	Source	Gallery List	Album List	Item List	Other
-----------------	--------	--------------	------------	-----------	--------------

Page to jump to if random image is clicked

 Page Content [68]

The only setting currently available on this tab is a PID of a page to jump to if you are in random image mode and want to set a page on which you want to jump to, if a random image is clicked.

Configuration

As most TYPO3 extensions, YAG ships with a lot of TypoScript settings to be made. To get a first impression of what is configured via TypoScript, open up a page on which YAG static template is included using the Template module and open 'TypoScript Object Browser':



The following reference will give you a in depth explanation of all TypoScript settings used in YAG.

TypoScript Reference

plugin.tx_yag.settings

This is the main section of our extension. All non-framework-specific configuration goes here.

Property:	Data type:	Description:	Default:
crawler	array	Settings for the YAG file crawler used for directory import.	
accessDenied	array	<p>Holds a controller / action pair, that defines which controller and action is called whenever access is denied for an action.</p> <p>Example:</p> <pre>accessDenied { controller = Gallery action = list }</pre> <p>This will show list action of Gallery controller, whenever access is denied.</p>	
sysImages	array	Holds an array of paths for different images used throughout the extension.	
themes	array	Holds an array of themes.	
extlist	array	Holds settings for pt_extlist extension. Take a look at the pt_extlist documentation for further information.	
importer	array	Holds settings for import.	

Property:	Data type:	Description:	Default:
overwriteFlexForm	array	<p>Use this to overwrite settings made in flexform For example to force the same theme in alle instances of the plugin.</p> <pre> overwriteFlexForm { contextIdentifier = contextReset = theme = context { selectedPid = selectedGalleryUid = selectedAlbumUid = selectedItemUid = galleryList { itemsPerPage = sorting { field = direction = } } albumList { itemsPerPage = sorting { field = direction = } } itemList { itemsPerPage = sorting { field = direction = } } linkMode = linkTargetPageUid = linkTargetPluginMode = filter { random = } } } </pre>	

config.tx_yag.settings.upload.multifile

Configuration for the multifile uploader

Property:	Data type:	Description:	Default:
file_size_limit	string	Size limit in Mb	1000
file_upload_limit	int		1000
file_types	string		*.jpg;*.jpeg;*.JPG;*.JPEG
button_image_url	string		EXT:yag/Resources/Public/Icons/XPButtonUploadText_61x22.png
available	Int		1

config.tx_yag.settings.upload.dragNDrop

Property:	Data type:	Description:	Default:
maxFiles	Int	Size Limit in Mb	1000
maxFileSize	int		1000

Property:	Data type:	Description:	Default:
available	Int		1

config.tx_yag.settings.crawler

Configuration for file crawler

Property:	Data type:	Description:	Default:
fileTypes	string	Comma-separated list of file-endings that should be indexed by crawler. Example: <code>jpg, jpeg</code>	jpg, jpeg

config.tx_yag.settings.importer

Configuration for importers

Property:	Data type:	Description:	Default:
parseMetaData	bool	If set to 1, meta data of imported images is parsed and written to itemMeta table.	1
generateTagsFromMeta Data	bool	If set to 1, keywords from meta data are imported as tags in corresponding table.	1
generateResolutions	csv	Comma-separated list of themes for which resolutions are created, when image is imported.	backend
importFileMask	string	File mask (UNIX file mask like 666) which is used on UNIX systems for imported files.	660
titleFormat	array	Set the title of the uploaded image automatically from the images filename or its meta data. Example: titleFormat = TEXT titleFormat.dataWrap = {field:fileName} by {field:author} / {field:artistWebsite} Available fields are: - origFileName - the original filename of the import file - fileName - Formated filename (suffix removed) And the fields of the imported meta data: - author - copyright - artistMail - artistWebsite - description - cameraModel - lens - focalLength - shutterSpeed - aperture - flash - keywords - description - tags	titleFormat = TEXT titleFormat.dataWrap = {field:fileName}
descriptionFormat	array	Example: descriptionFormat = TEXT descriptionFormat.dataWrap = {field:description} Fields are the same as in titleFormat	descriptionFormat = TEXT descriptionFormat.dataWrap = {field:description}

plugin.tx_yag.settings.imageProcessor

Property:	Data type:	Description:	Default:
meaningfulTempFilePrefix	integer	MeaningfulTempFilePrefix specifies the length of the chunk of the original filename which is prefixed to the temp filename	config.meaningfulTempFilePrefix

plugin.tx_yag.settings.sysImages

Configuration for all kinds of images used for skinning.

Property:	Data type:	Description:	Default:
imageNotFound	Item file description	<p>Configures a path, title and description for an item.</p> <p>Example:</p> <pre>sysImages { imageNotFound { sourceUri = typo3conf/ext/yag/Resources/Public/Icons/imageNotFound.jpg title = No image found. description = No image found. } }</pre> <p>Mind that the sourceUri of the image must be relative to TYPO3 root.</p>	

plugin.tx_yag.settings.themes

Most of the configuration for YAG can be found in themes. We have a default theme, where you can find all the settings available in YAG. See section 'Themes and Templates' in the Developers' chapter for further information on how to extend themes and write your own themes.

Property:	Data type:	Description:	Default:
[your_theme_name]	array	You can define your own themes here. YAG ships with a default theme and a backend theme.	

plugin.tx_yag.settings.themes.default

In this section, you can find the settings for the default theme which acts as basis for all other themes. Best practice for developing your own themes is to extend this theme with your own theme like that:

```
plugin.tx_yag.themes.[your_theme_name] < plugin.tx_yag.themes.default
plugin.tx_yag.themes.[your_theme_name]{
    # ... your theme specific settings
}
```

Property:	Data type:	Description:	Default:
showBreadcrumbs	bool	If set to 1, breadcrumbs are shown as navigation.	1

Property:	Data type:	Description:	Default:
resolutionConfigs	array	<p>Configuration for image resolutions. You can define the resolutions of thumbnails, single images etc. here.</p> <pre> resolutionConfigs { thumb { width = 150c height = 150c } medium { maxW = 800 maxH = 600 } } </pre> <p>In the default theme, thumb for thumbnails and medium for medium sized images in single view are defined and used. For your own template, you can define any kind of resolutions with the name of your choice.</p> <p>A resolution configuration can consist of any parameter that the TYPO3 IMAGE type provides, including image manipulation via GIFBUILDER.</p>	
gallery	array	Gallery specific settings of your theme. See section below	
album	array	Album specific settings of your gallery. See section below	
extlist	array	This section configures pt_extlist specific settings for YAG. See pt_extlist documentaiton for further information.	
itemList	array	This section configures the list of images shown, when you click on an album. See section below for further information.	
item	array	This section configures single view of an item. See section below for further information.	
includeLibJS	CSV	<p>Comma-separated list of defined librarys from wich you want to include javascript files.</p> <p>Defined libraries are jQuery, jQueryUi, jQueryShadowBox</p>	
includeLibCSS	CSV	<p>Comma-separated list of defined librarys from wich you want to include CSS files.</p> <p>Defined libraries are jQuery, jQueryUi, jQueryShadowBox</p>	
includeJS	array	Define JS files which should be included in the page header. Same schema as in page.	
includeCSS	array	Define CSS files which should be included in the page header. Same schema as in page.	

plugin.tx_yag.settings.themes.default.feed

Property:	Data type:	Description:	Default:
Active	bool	Activate the feed	0
title	String	The feeds title	YAG Gallery Feed
description	String	The feeds description	Description
Author	String	The feeds author	The Photographer
Language	String	The feed language	de_de

plugin.tx_yag.settings.themes.default.galleryList

Gallery specific settings of your theme.

Property:	Data type:	Description:	Default:
columnCount	int	Number of columns used for rendering gallery overview.	2
GalleryThumbPartial	String	Pathand filename of the gallery thumb partial.	Gallery/GalleryThumb.html
pagerIdentifier	String	Pager Identifier default / delta	Default
pagerPartial	String	Path to Pagerpartial <ul style="list-style-type: none"> Pager/Default Pager/Delta 	Pager/Default

plugin.tx_yag.settings.themes.default.albumList

Album specific settings of your theme.

Property:	Data type:	Description:	Default:
itemsPerPage	int	Number of albums shown on album list	12
showBreadcrumbs	bool	If set to 1, breadcrumbs are shown on album page.	1
columnCount	int	Number of columns used for rendering album list.	2
AlbumThumbPartial	String	Pathand filename of the album thumb partial.	Album/AlbumThumb.html
pagerIdentifier	String	Pager Identifier default / delta	Default
pagerPartial	String	Path to Pagerpartial <ul style="list-style-type: none"> Pager/Default Pager/Delta 	Pager/Default

plugin.tx_yag.settings.themes.default.extlist

pt_extlist specific settings of your theme. See pt_extlist documentation for further information.

plugin.tx_yag.settings.themes.default.itemList

Configuration of image list of your theme.

Property:	Data type:	Description:	Default:
itemsPerPage	int	Number of images shown on a single page.	12
columnCount	int	Number of columns used to render images on image list.	4
showTitle	bool	If set to 1, album title is shown on overview page.	1
imageThumbPartial	path	Path to partial used to render an image in image list. This can be Extbase path (relative to EXT:yag/Resources/Private/Partials): Image/ImageThumb or common TS resource path to set offer paths: EXT:yag/Resources/Private/Partials/Image/ImageThumb.html	Image/ImageThumb
imageAdminThumbPartial	path	Not used at the moment.	

Property:	Data type:	Description:	Default:
pagerPartial	path	<p>Path to partial used to render a pager in image list. This can be Extbase path (relative to EXT:yag/Resources/Private/Partials):</p> <p>Pager</p> <p>or common TS resource path to set offer paths:</p> <p>EXT:yag/Resources/Private/Partials/Pager.html</p> <p>This is especially useful, if you want to add additional parameters to the links generated by the pager, as in the following example:</p> <pre><extlist:link.action addQueryString="true" controller="{controller}" action="{action}" arguments="{extlist:namespace.GPArray(object:'{pagerCollection}') arguments:'page:{i}'}">{pageNumber}</extlist:link.action></pre>	Pager
pagerIdentifier	String	Pager Identifier default / delta	Default
pagerPartial	String	Path to Pagerpartial <ul style="list-style-type: none"> • Pager/Default • Pager/Delta 	Pager/Default
linkMode	string	Link mode [show link]	show
linkTargetPageUid	integer	The page uid of the target page	
linkTargetPluginMode	string	The plugin mode on the target page	Album
Filter.random	Boolean	Activates the random uid filter	0

plugin.tx_yag.settings.themes.default.itemList.zipDownload

Property:	Data type:	Description:	Default:
active	int	Activate / Deactivate the zip download	0
fileNameFormat	array	Defines the zip file name. Currently available fields are gallery and album: fileNameFormat = TEXT fileNameFormat.dataWrap = {field:album}.zip	{field:album}.zip
resolution	String	Name of the resolution in which the images are packed.	original

plugin.tx_yag.settings.themes.default.item

Configuration of image single view of your theme.

Property:	Data type:	Description:	Default:
showItemMeta	bool	If set to 1, metadata of image will be shown in single view.	1
itemMetaPartial	path	<p>Path to partial used to render image meta data (EXIF etc.). This can be Extbase path (relative to EXT:yag/Resources/Private/Partials):</p> <p>Image/ImageMeta</p> <p>or common TS resource path to set offer paths:</p> <p>EXT:yag/Resources/Private/Partials/Image/ImageMeta.html</p>	Image/ImageMeta
showTitle	bool	Show the item title beneath the image	1
showDescription	bool	Show the item description beneath the image	1
showPager	Bool	Show the back / forward pager	1
showItemMeta	bool	Show Meta information for an item (including title and description)	1
showOriginalDownloadLink	bool	Show download link to original item	1

Property:	Data type:	Description:	Default:
pagerPartial	string	Path to pager partial	Pager/SingleItem
itemMetaPartial	string	Path to item meta partial	Image/ImageMeta

plugin.tx_yag.settings.themes.default.item.interaction

Configures optional visitor interaction services

Property:	Data type:	Description:	Default:
socialSharePrivacy	array	Configuration for the social share privacy widget:	
disqus.path	string	Path to the partial	Interaction/SocialSharePrivacy
socialSharePrivacy.show	bool	Activate the widget	0
socialSharePrivacy.settings	array	info_link = http://panzi.github.com/SocialSharePrivacy/ language = en services { buffer.status = false delicious.status = false disqus.status = false mail.status = false flattr.status = false linkedin.status = false pinterest.status = false reddit.status = false stumbleupon.status = false tumblr.status = false xing.status = false facebook.status = true twitter.status = true gplus.status = true }	
socialSharePrivacy.path	string	Path to the partial	Interaction/SocialSharePrivacy
disqus.show	bool	Activate the widget	0
disqus.settings	array	disqus_shortcode = YourDisQusName	

plugin.tx_yag.settings.themes.lightbox.fancybox

YAG uses the fancybox as a default lightbox. The keys in this path are used to configure the lightbox. See <http://fancybox.net/api> for a detailed description of the available configuration variables.

module.tx_yag.settings

Holds settings for the backend of YAG. The content of this setting is the same as plugin.tx_yag.settings.

Use realUrl with YAG to get speaking URLs

To be honest, it is not trivial to build a realURL config for YAG. But we have good news for you - we have done the job and created a realURL-hook to do the work.

So, to use YAG with speaking URLs, all you have to do is adding the following hooks to your realURL config:

```
$GLOBALS['TYPO3_CONF_VARS']['EXTCONF']['realurl'] = array (
    'encodeSpURL_postProc' => array(
        'yag' => 'EXT:yag/Classes/Hooks/RealUrlHook.php:user_Tx_Yag_Hooks_RealUrl->encodeSpURL_postProc',
```



```
),  
'decodeSpURL_preProc' => array(  
    'yag' => 'EXT:yag/Classes/Hooks/RealUrlHook.php:user_Tx_Yag_Hooks_RealUrl->decodeSpURL_preProc',  
),  
...  
)
```

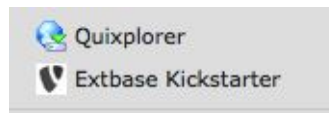
Tutorial

How to create your own Themes as a third-party extension

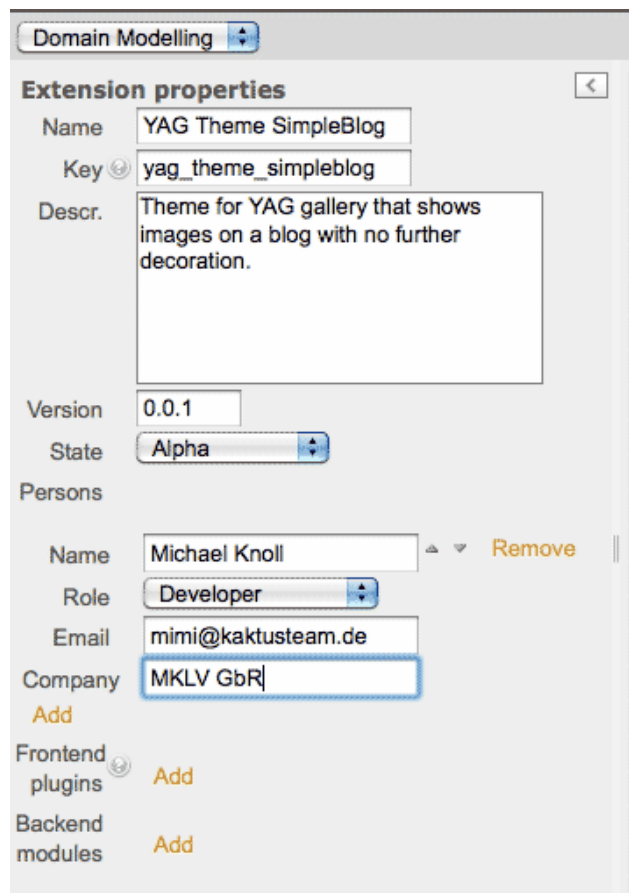
Although you can create your themes with TypoScript and some Fluid templates, we will show here how to create a theme as a third party extension. We use Extbase's Kickstarter to create a basic extension and then show how to bring up all the rest.

1. Creating a basic extension using FLUID kickstarter

Start Extbase kickstarter:



Open the properties pane and give your theme a descriptive name and some more information:



Domain Modelling

Extension properties

Name: YAG Theme SimpleBlog

Key: yag_theme_simpleblog

Descr.: Theme for YAG gallery that shows images on a blog with no further decoration.

Version: 0.0.1

State: Alpha

Persons

Name: Michael Knoll

Role: Developer

Email: mimi@kaktusteam.de

Company: MKLV GbR

Add

Frontend plugins: Add

Backend modules: Add

Save your extension. If everything worked fine, you should get the following message:



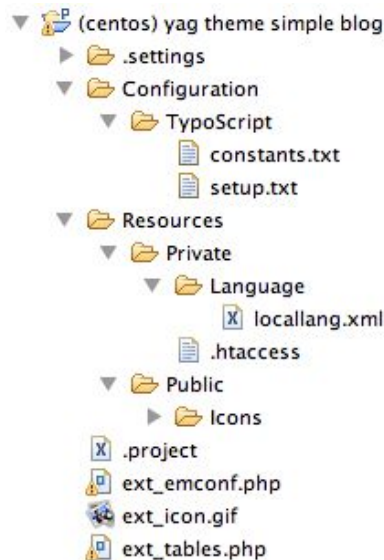
2. Open you extension in some Editor / Create a new project

Open your extension in any editor you like – e.g. ZendStudio and create a new project. First of all, you can delete some files

and folders, as we won't need them:

- /ext_localconf.php (we won't have any plugins configured and won't have any tables or TCA)
- /ext_tables.sql (as we won't have any tables, we don't need this file)
- /kickstarter.json (there is no need in keeping this file)
- /Resources/Private/Language/locallang_db.xml (we won't have any DB changes or fields)
- /Configuration/TCA (we don't need no TCA changes)

So after that, your folder structure looks something like that:



3. Modify ext_tables.php

We will later include a static template holding the configuration for our theme. This has to be included via ext_tables.php:

```
<?php
if (!defined ('TYPO3_MODE')) die ('Access denied.');
```



```
// Include static template for SimpleBlog Theme
t3lib_extMgm::addStaticFile($_EXTKEY, 'Configuration/TypeScript', '[yag] Theme SimpleBlog');
```

```
?>
```

Save your ext_tables.php and you're done with it.

4. Set up basic TypeScript Template

We need at least one TS-Template for registering our theme and copying all the basic settings from the default theme. Copying the settings from default prevents you from writing all configuration again. So first you copy everything and then you overwrite the settings you would like to change. Here is our first draft of the /Configuration/TypeScript/setup.txt:

```
#####
# YAG theme SimpleBlog
#
# @author Daniel Lienert <daniel@lienert.cc>
# @author Michael Knoll <knoll@punkt.de>
#####

# Copy default settings from default theme
plugin.tx_yag.settings.themes.simpleBlog < plugin.tx_yag.settings.themes.default
```

As long as we do not want to change any settings, that's all we have to do. We will come back to this file later.

5. Hands on FLUID Templates

Most of the look and feel of our themes is set via FLUID html templates. So let's create some. The only template we need for our theme is a simple image list, that shows the images from a single album.

So here's, what you have to do step-by-step:

1. Create a new folder in /Resources/Private and call it 'Templates'
2. Create a new folder in /Resources/Private/Templates and call it 'ItemList'
3. Create a new html file 'List.html' inside this folder (/Resources/Private/Templates/ItemList/List.html)
4. Remove all content from this html file your editor probably auto-generated
5. Open up your setup.txt again and add the following lines to overwrite the template used for the ItemListController and list action:

```
plugin.tx_yag.settings.themes.simpleBlog {
    controller {
        ItemList {
            list.template = EXT:yag_theme_simpleblog/Resources/Private/Templates/ItemList/List.html
        }
    }
}
```

6. Write some template code inside '/Resources/Private/Templates/ItemList/List.html':

```
{namespace yag=Tx_Yag_ViewHelpers}

<div class="tx-yag-theme-simpleblog-imagecontainer">
    <f:for each="{listData}" key="rowIndex" as="listRow">
        <f:render partial="{config.itemListConfig.imageThumbPartial}"
            arguments="{config: config, image: listRow.image.value,
                rowIndex: listRow.specialValues.absoluteRowIndex,
                pager: pager, pagerCollection: pagerCollection}" />
    </f:for>
</div>
```

The first line gives us access to some YAG viewhelpers we probably need later on. The rest is quite straight-forward: We open up a div container for all images. Then we iterate through the rows we get from pt_extlist as listData object. **Each row stands for a single image in our album.** Do not confuse row here with the visualized rows containing some images in a row. A row here is a record in a database table containing one record, which is an image here!

We then use a partial to render the thumbnail for an image. This partial will be created in the next step.

One line that should get your attention is '{config.itemListConfig.imageThumbPartial}'. 'config' holds a so called ConfigurationBuilder which gives you access to almost all the configuration of YAG within your templates and partials. We will come back to this later.

7. You are done with your template. Save it!

6. Hands on FLUID Partial

Although it is not really needed here to use partial, we will use them for practice. Again step-by-step what you have to do:

1. Create a new folder in '/Resources/Private' and call it 'Partials'
2. Create a new file in your new folder and call it 'ImageThumb.html'
3. Open this file and remove any content eventually created
4. Open your setup.txt again and add the following lines to make the partial known to yag:

```
plugin.tx_yag.settings.themes.simpleBlog {
    controller {
        ItemList {
            list.template = EXT:yag_theme_simpleblog/Resources/Private/Templates/ItemList/List.html
        }
    }

    itemList {
        imageThumbPartial = EXT:yag_theme_simpleblog/Resources/Private/Partials/ImageThumb.html
    }
}
```

```
}
```

- Write some code to make the partial work (for your own themes, you can copy and paste the content of the partial in YAG's default theme for a start):

```
{namespace yag=Tx_Yag_ViewHelpers}

<f:if condition="{image.width} > {image.height}">
    <div class="tx-yag-theme-simpleblog-thumb tx-yag-theme-simpleblog-thumb-landscape">
</f:if>
<f:if condition="{image.width} < {image.height}">
    <div class="tx-yag-theme-simpleblog-thumb tx-yag-theme-simpleblog-thumb-portrait">
</f:if>
<f:if condition="{image.width} == {image.height}">
    <div class="tx-yag-theme-simpleblog-thumb tx-yag-theme-simpleblog-thumb-square">
</f:if>
        <a href="{yag:resource.image(item: image, resolutionName: 'lightbox')}"
            rel="shadowbox[images_{image.album.uid}]" title="{image.title}">
            <yag:image item="{image}" resolutionName="thumb" alt="{image.title}"/>
        </a>
        <ul>
            <li class="tx-yag-theme-simpleblog-thumb-title">{image.title}</li>
        </ul>
    </div>
```

So this one's a little trickier. The outer div's class changes, depending on whether the image is a landscape image or a portrait image. Therefore we use an inline if-viewhelper.

For a single click, we use a link to a down-sized image whose URL we get using imageLink viewhelper. We later have to set up a resolution configuration with the name 'lightbox'.

- Add the required resolutions to setup.txt:

```
plugin.tx_yag.settings.themes.simpleBlog {
    resolutionConfigs {
        thumb {
            width = 150
            height =
            quality =
        }
        lightbox {
            width = 1200
            height = 800
        }
    }

    controller {
        ItemList {
            list.template = EXT:yag_theme_simpleblog/Resources/Private/Templates/ItemList/List.html
        }
    }

    itemList {
        imageThumbPartial = EXT:yag_theme_simpleblog/Resources/Private/Partials/ImageThumb.html
    }
}
```

- Save your partial.

7. Add some CSS

The last step in creating your theme is CSS styling. We therefore create a new folder '/Resources/Public/CSS' and in there a new file 'styles.css'.

Here is our CSS:

```
@CHARSET "UTF-8";

div.tx-yag-theme-simpleblog-imagecontainer:after {
    content: ".";
    display: block;
    height: 0;
    clear: both;
    visibility: hidden;
}
```

```
div.tx-yag-theme-simpleblog-thumb {
    float: left;
    height: 200px;
    width: 200px;
    border: 1px solid;
    border-color: #AAA #444 #444 #AAA;
    margin: 10px 10px;
    padding: 24px;
    -moz-border-radius: 3px 3px 3px 3px;
    -moz-box-shadow: 3px 3px 4px #aaa;
    -webkit-box-shadow: 3px 3px 4px #aaa;
    box-shadow: 3px 3px 4px #aaa;
    /* For IE 8 */
    -ms-filter: "progid:DXImageTransform.Microsoft.Shadow(Strength=4, Direction=135, Color='#aaaaaa')";
    /* For IE 5.5 - 7 */
    filter: progid:DXImageTransform.Microsoft.Shadow(Strength=4, Direction=135, Color='#aaaaaa');
}

div.tx-yag-theme-simpleblog-thumb ul {
    display: none;
}

div.tx-yag-theme-simpleblog-thumb img {
    border: 1px solid;
    border-color: #444 #AAA #AAA #444;
}

div.tx-yag-theme-simpleblog-thumb-landscape img {
    height: 133px;
    width: 200px;
    margin: 33px 0;
}

div.tx-yag-theme-simpleblog-thumb-portrait img {
    height: 200px;
    width: 133px;
    margin: 0 33px;
}
```

There is one thing left to do: Include CSS file via TypoScript so for a last time, open up setup.txt and bring it to its final version:

```
# Include CSS for this theme
page.includeCSS.yag_theme_simpleBlog = EXT:yag_theme_simpleblog/Resources/Public/CSS/styles.css

# Copy default settings from default theme
plugin.tx_yag.settings.themes.simpleBlog < plugin.tx_yag.settings.themes.default

# Some theme-specific settings
plugin.tx_yag.settings.themes.simpleBlog {

    resolutionConfigs {

        thumb {
            maxH = 200
            maxW = 200
        }

        lightbox {
            maxH = 1200
            maxW = 800
        }

    }

    controller {

        ItemList {
            list.template = EXT:yag_theme_simpleblog/Resources/Private/Templates/ItemList/List.html
        }

    }

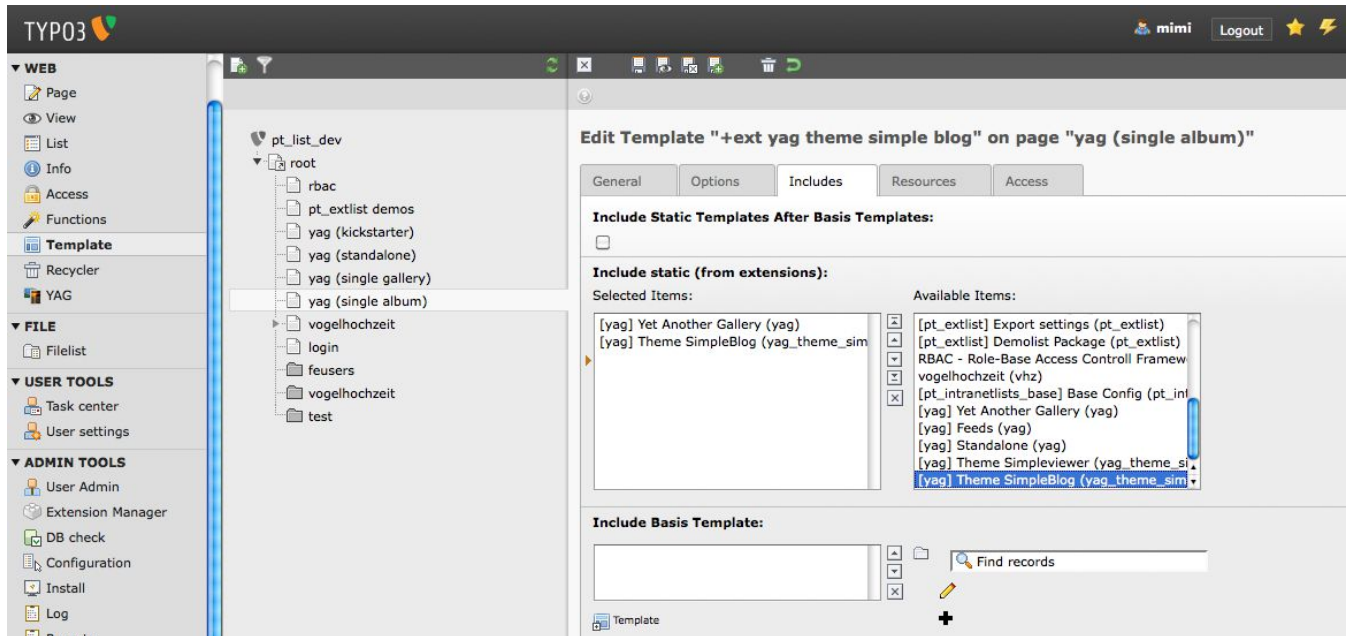
    itemList {
        imageThumbPartial = EXT:yag_theme_simpleblog/Resources/Private/Partials/ImageThumb.html
    }

}
```

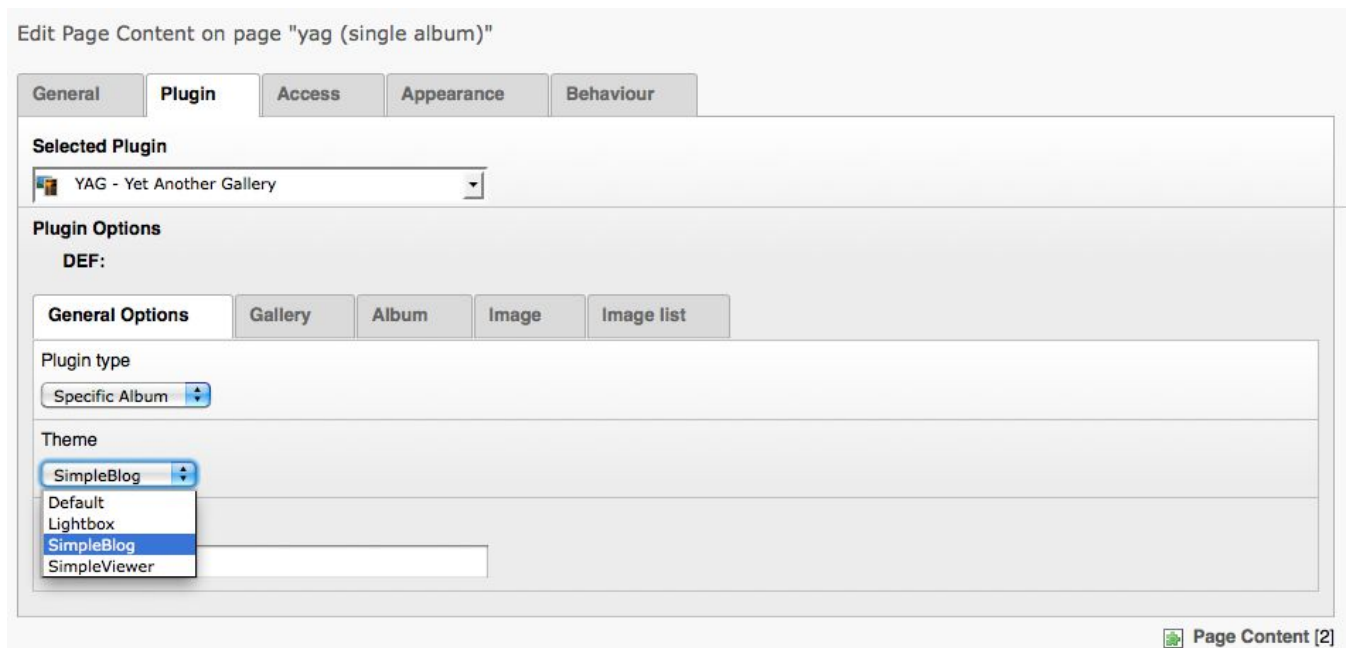

8. Installing your theme

You can now install your theme using the Extension Manager. Chose 'Install Extensions' from the dropdownlist and click on the gray brick to install your new theme.

After installing the extension, you can include the static template on the page you want to use the new theme. If you haven't yet done so, create a new extension template and add the static template you just created.



Now you can chose the theme from within your plugin's FlexForm on the page you want to include the album:



And basically that's it :-> You have just created your new theme. Depending on your settings, it could look something like this:



Developers Guide

We implemented YAG so that it is easily extendable by developers. Here are some basic things you need to know for writing your own extensions for YAG.

YAG Architecture

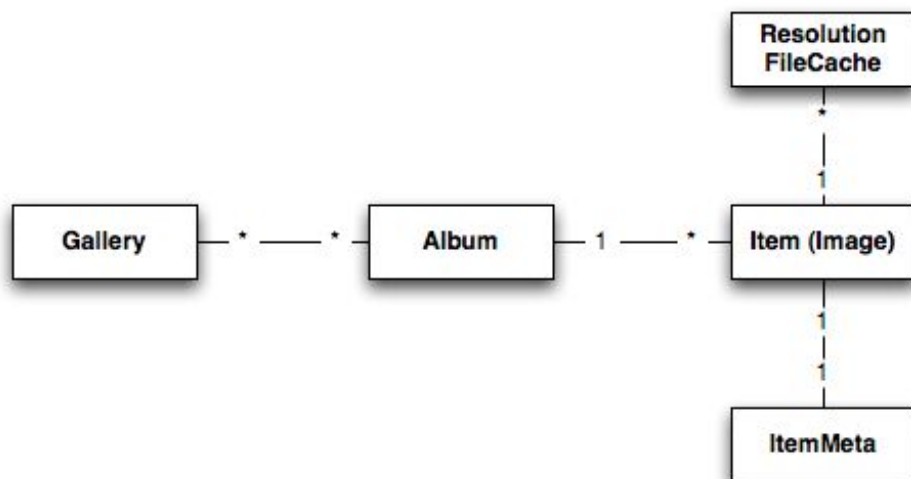
Usage of other extensions

We mainly use 3 dependent extensions:

- **Extbase** as basic TYPO3 Extension Framework
- **pt_extbase:** improvements to extbase. Custom controller, viewhelpers and utilities.
- **pt_extlist** for rendering all sorts of lists like lists of galleries, albums or images. pt_extlist does all the stuff like paging, filtering and sorting for us.

Domain Model

The domain model of YAG is quite simple:



We have the following domain objects:

- **Gallery** acts as a 'container' for everything. Inside a gallery, you can have multiple albums. Mind that an album can also belong to different galleries, what makes things a little more complicated...
- **Album** holds a set of images, which we call items in YAG (we call it items, because that could be other things except images, like videos etc.).
- **Item (Image)** holds all information of an image like its source path, filename and other data.
- **ItemMeta** holds meta-data information for an item. E.g. EXIF or IPTC.
- **ResolutionFileCache** holds caching information

We refer to the domain model of YAG as all objects that directly have to do with the domain of our extension (remember – it's a gallery) and have to be persisted in the database.

Further Domain Objects

Besides the model there are other domain objects used to handle gallery related stuff. Those objects are not persisted and are therefore not part of what we called the domain model above:

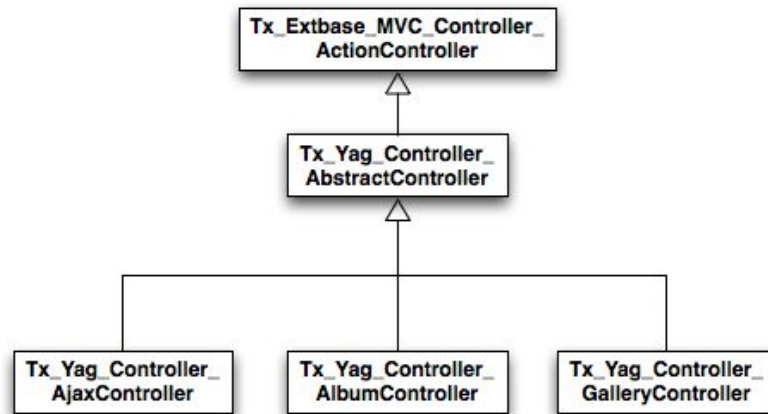
- **Configuration** keeps a set of configuration objects that implement an object oriented way to handle TypoScript configuration and validation.
- **Context** context is a container of other objects used to make it easier to set and access certain information. E.g. getting all images stored in an album is implemented via a list of images that is filtered by an album uid. Setting the album for the filter and accessing it afterwards is made quite simple using the context.
- **ImageProcessing** Where there are images, there has to be done different kinds of processing. E.g. resizing of

images is done by the objects kept in this part of the domain.

- **Import** holds all objects and classes required for importing images to YAG.

Controllers

As Extbase is implemented using the MVC paradigm, we have some controllers handling all the requests coming from Browsers or other applications from outside TYPO3.



Our controllers each extend an abstract controller that holds some functionality we require for YAG. This abstract controller extends Extbase's ActionController. For a detailed explanation of how controllers and actions are handled within TYPO3 and Extbase, refer to the Extbase documentation.

Filesystem Structure

The filesystem structure follows Extbase conventions. Each directory starts with an uppercase letter.

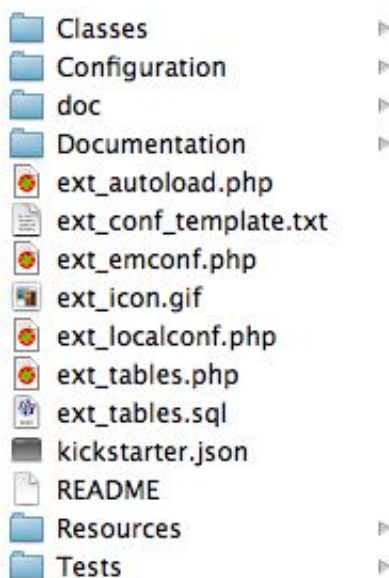


Image Storage and Resolution File Cache

A problem for every gallery application is the storage of original and resized images.

Storage of original images

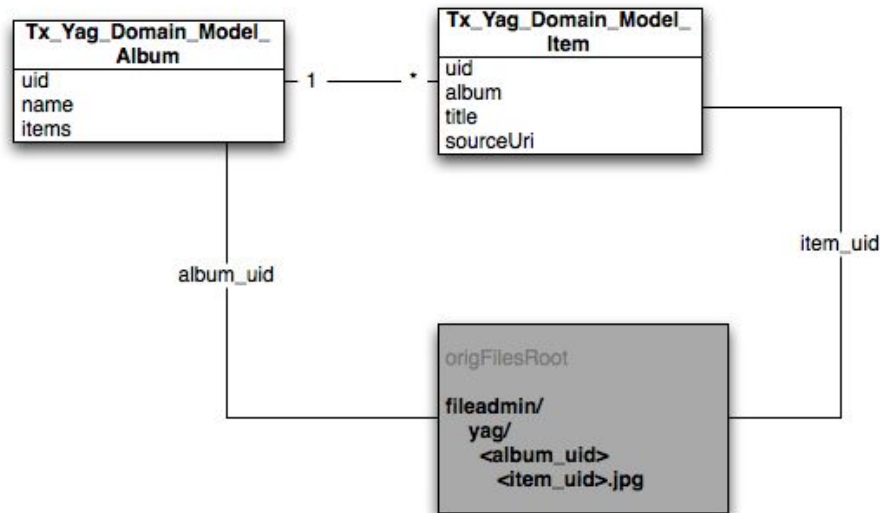
Each image in YAG is stored as original image. This image is never changed by YAG. You find those images in the folder you set in the Extension Manager as 'origFilesRoot':

Path of directory where YAG stores ... [origFilesRoot]

Path of directory where YAG stores all original files.

fileadmin/yag

Whenever you use an original image in an album, an item object is created. This item object has a UID and an albumUid as well as a sourceUri pointing to the file inside the origFilesRoot where the original file is stored. For each album there is an individual folder in the origFilesRoot named like the UID of the album. Each item (image) is stored, using its UID as filename:



Storage of resized (cached) images

Whenever you use an item in your albums, YAG will generate resized versions of this item according to the resolution settings in your theme. So for each item and for each resolution there will be a cached version on your server. YAG stores those images in a chaos-filesystem. Chaotic means, that YAG will automatically create a folder structure so that no more than a hundred files are located in one folder. Many files in the same folder will make the filesystem slow.

So here is how it works:

- Each item holds a **Tx_Yag_Domain_Model_ResolutionFileCache** object for each resolution set in TypoScript.
- A cached image is written to the server for each resolution. A hashed string is used for the filename to prevent traversing access to the files.
- The resolution file cache object holds the resolution and the path to the cached file on the server.

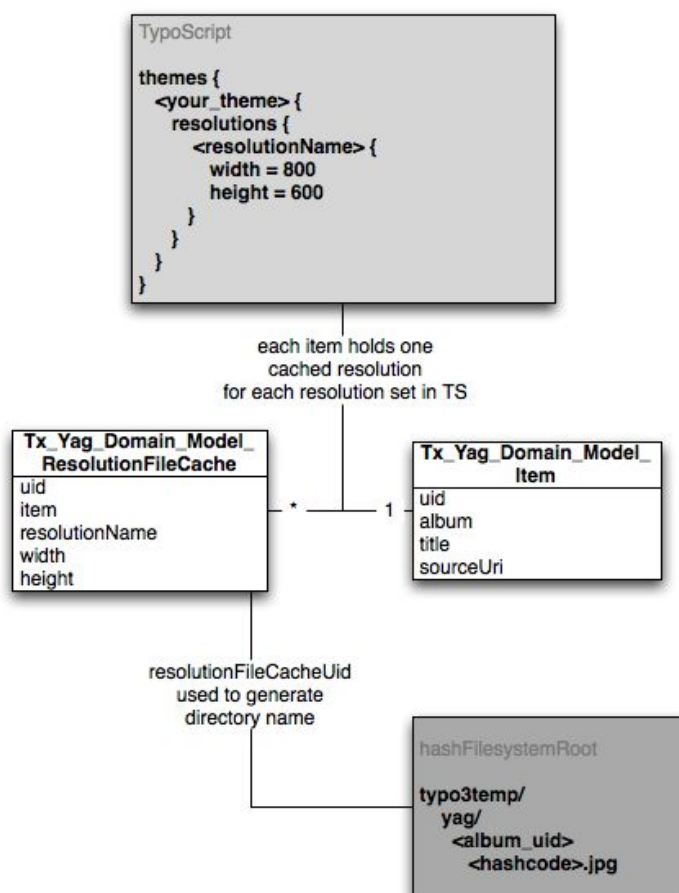
The root folder of the hash filesystem root is set in your Extension Manager:

Path of directory where YAG should ... [hashFilesystemRoot]

Path of directory where YAG should store all cached images generated for albums - relative to Typo3 base path. This path should not be changed unless you know what you do! (Image base path)

typo3temp/yag

The following diagram shows how the resolution file cache works:

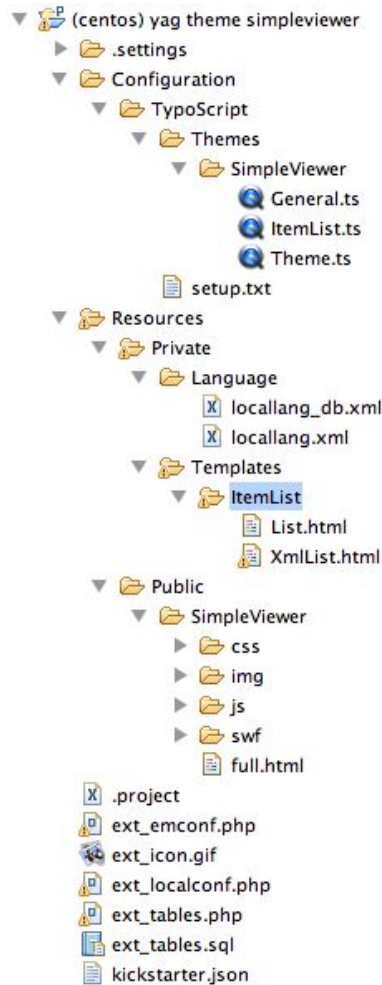


Themes and Templates

Themes are used to configure almost everything in YAG. This enables you to change almost everything using your own themes. The theme selector in your FlexForm lets you chose the theme you want to use for showing your gallery widgets.

What is included by a theme

A theme mainly consists of some TypoScript configuration and a set of FLUID templates. There also might be some FLUID partials and third party JavaScript libraries and CSS files. Take a look at the filestructure to get an impression of what are the contents of a theme:



How to overwrite templates used for Controller/Action pairs

We extended Extbase's default ActionController in such a way, that you can overwrite each template for each controller / action pair using TypoScript. Let's say, you have a controller called 'ItemList' and an action 'list'. Then you can overwrite the template used for this controller/action pair using the following TypoScript configuration:

```
##Overwriting template path for ItemList->listAction()
plugin.tx_yag.settings.controller.ItemList.list.template = EXT:<ext_key>/<path_to_template>

## For example:
plugin.tx_yag.settings.controller.ItemList.list.template = EXT:yag/Resources/Private/Template/ItemList/
list.html
```

This feature enables you to set templates for YAG controllers to html files outside the extension itself, which is currently not possible using Extbase.

So here is another example on how to set template paths for a theme to template files that ship with the theme's extension:

```
plugin.tx_yag.settings.themes.simpleViewer {
    controller {
        ItemList {
            list.template = EXT:yag_theme_simpleviewer/Resources/Private/Templates/ItemList/List.html
            xmlList.template = EXT:yag_theme_simpleviewer/Resources/Private/Templates/ItemList/XmlList.html
        }
    }
}
```

You can see, that you can set the template paths to files included by your theme-extension and not by YAG.

Which objects are available in your templates

After you know how to set templates for a controller / action pair, the next question should be how you can access data from within your templates. Therefore you have to know, how data is assigned to your template. The easiest way to find this out, is to take a look in your controller's code:

```
public function newAction(
    Tx_Yag_Domain_Model_Gallery $gallery=NULL,
    Tx_Yag_Domain_Model_Album $newAlbum=NULL) {

    $selectableGalleries = $this->objectManager->get(
        'Tx_Yag_Domain_Repository_GalleryRepository')->findAll();

    $this->view->assign('selectableGalleries', $selectableGalleries);
    $this->view->assign('selectedGallery', $gallery);
    $this->view->assign('newAlbum', $newAlbum);
}
```

So here you can see, that there are three objects passed to your template: 'selectableGalleries', 'selectedGallery' and 'newAlbum'. You can access those objects using FLUID's mechanism of accessing properties:

```
<f:layout name="Default" />
<f:section name="main">
<h1><f:translate key="tx_yag_controller_album_new.header" /></h1>

    <f:render partial="FormErrors" arguments="{for: 'newGallery'}" />
    <f:form method="post" controller="Album" action="create" name="newAlbum"
        object="{newAlbum}">
        <f:render partial="Album/FormFields" arguments="{album:album,
            selectableGalleries:selectableGalleries, selectedGallery:selectedGallery}" />
        <f:form.submit class="submit" value="{f:translate(key: 'general.save'
            default: 'Save')}" />
    </f:form>

    <f:link.action controller="Gallery" action="index">
    <f:translate key="tx_yag_controller_gallery.backToGallery" />
    </f:link.action>
</f:section>
```

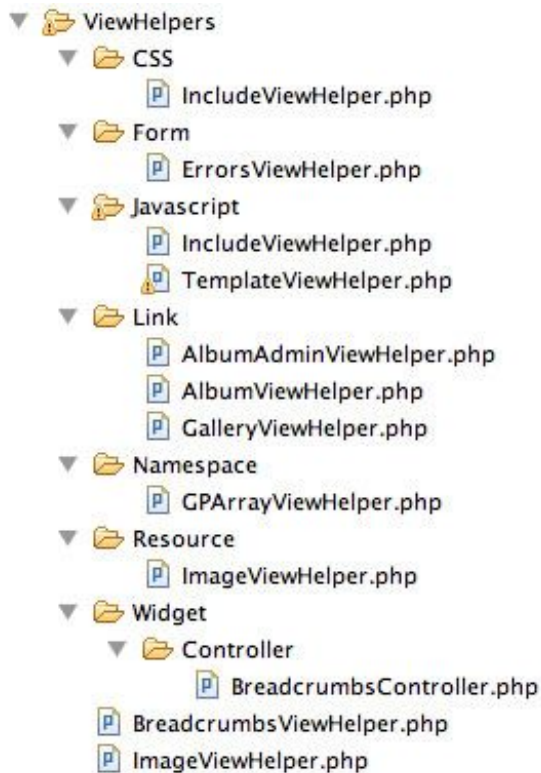
The YAG context object

Whenever YAG is used on a page (whether as content-element or in standalone mode), there are many dependencies to be resolved and a lot of environment to be set up. To make it easier for a developer to handle all this stuff, we implemented what we call a **context**. A context is a container giving you a nice way to set and get information gathered within a lifecycle of YAG. This might be configuration as well as currently set parameters. To get an impression of what is stored within the context container, here is a little diagram again:

TODO: finish me!

YAG ViewHelpers

Here is a list of viewhelpers available from within YAG:



If you want to use YAG's viewhelpers in your templates, you have to include them with the following line of code at the beginning of your template:

```
{namespace yag=Tx_Yag_ViewHelpers}
```

ViewHelper:	Parameters:	Description:
BreadcrumbsViewHelper	none	Renders a root path menu from gallery to image. <div> Travel & Passion » Sardinia » sardinien-015.jpg </div> Example: <yag:breadcrumbs />
ImageViewHelper	Item: item object resolutionName: The name of a defined resolution.	Renders an image in the given resolution. Example: <yag:image item="{image}" resolutionName="thumb" />

CSS

The followin viewhelpers are available for handling CSS related stuff:

ViewHelper:	Parameters:	Description:
IncludeViewHelper	Library: library name File: path to file	Includes CSS Files to the header section. If library is given, the css files defined in the library are included (see: Typoscript/BaseConfig/HeaderInclusion/) Example: <yag:CSS.Include library="jQueryShadowBox" />

Javascript

The followin viewhelpers are available for handling Javascript related stuff:

ViewHelper:	Parameters:	Description:
IncludeViewHelper	Library: library name File: path to file	<p>Includes JS Files to the header section. If library is given, the js files defined in the library are included (see: TypoScript/BaseConfig/HeaderInclusion/)</p> <p>Example:</p> <pre><yag:Javascript.Include library="jQuery" /></pre> <p>In order to make this work, you have to configure your libraries in TypoScript. You find a list of predefined libraries in Configuration/TypoScript/BaseConfig/HeaderInclusion/JQuery.ts:</p> <pre>plugin.tx_yag.settings.frontendLib { jQuery { include = {\$config.yag.addjQuery} includeJS.jQuery = EXT:yag/Resources/Public/Js/JQuery/jquery-1.5.1.min.js # includeCSS.jQuery = EXT:yag/Resources/Public/CSS/JQuery/base.css } }</pre>
TemplateViewHelper	TemplatePath: path to a jsTemplate Arguments: the arguments to replace in the template.	<p>This viewhelper is in some way a pragmatic approach to avoid the fluid restrictions with javascript inline markup in templates. All arguments given to the viewhelper are replaced in the Javascript template in the form <code>###argument###</code> with the given value.</p> <p>There are some implicit defined markers: extPath: relative path to the extension extKey: Extension Key pluginNamespace: Plugin Namespace for GET/POST parameters</p> <p>Example (usage of viewhelper):</p> <pre><yag:Javascript.Template templatePath="EXT:yag/Resources/Private/JSTemplates/ItemAdminList.js" arguments="{ajaxBaseURL : 'f:uri.action(controller:'Ajax')}'" /></pre> <p>Example (usage of template markers in JS templates – so it's JavaScript what you see here):</p> <pre>var del_url = '###ajaxBaseURL###' + '&###pluginNamespace###[action]=deleteItem';</pre>

Link

The following viewhelpers are available for rendering links:

ViewHelper:	Parameters:	Description:
AlbumViewHelper	Album: album object	Renders a link for an album
AlbumAdminViewHelper	Album: album object	Renders a link for administrating an album
GalleryViewHelper	Gallery: gallery object	Renders a link for a gallery

Namespace

The followin viewhelpers are available for using namespaces:

ViewHelper:	Parameters:	Description:
GPArrayViewHelper	###TODO daniel###	###TODO daniel###

Resource

The followin viewhelpers are available for getting URIs for resources:

ViewHelper:	Parameters:	Description:
ImageViewHelper	Item: item object resolutionName: The name of a defined resolution.	Renders URI for an image. Used in XML view for example. Example: <code><yag:resource.image item="{listRow.image.value}" resolutionName="thumb" /></code>

Extending via Signal/Slot

Provided Signals

Class	Signal Name	Description	Parameter
Tx_Yag_Domain_Import_Meta Data_ItemMetaFactory	processMetaData	Can be used to further adjust the meta data after it is processed by the metaDataFactory	metaData: Reference to the metaData array including the extracted raw meta Data. fileName: Path to the original image file name.

Known problems

Although we tried to make every version of YAG as stable as possible, there are still some open issues. For getting a complete overview of what's missing, take a look at:

<http://forge.typo3.org/projects/extension-yag/issues>

Working with TYPO3 4.5.x

We decided no longer to support TYPO3 4.5 with version 2.0. There will be a bugfix branch 1.5.x of yag that will fix bugs with TYPO3 4.5 but new features will no longer be added. As we cannot maintain two different versions of our extension in TER, there is a download page on our website, where you can get updated versions of 1.5.x and download them as .t3x files. The link is <http://www.yag-gallery.de/download/>. You will find further information there.

Open Issues for version 1.x

- Ajax-updating problems: There are some widgets like pagers in the backend, that are not updated when an Ajax request is handled. For example the number of items in an album is not updated, if an album gets deleted.
- Role-Management: Although we have a role-management implemented in YAG, there is no administration view to set up users etc. that are equipped with roles. This will come with Frontend-Editing in a future version.
- Categories / Subalbums: We are working on this and are still waiting for donation. We hope to be able to release categories with version 3.0

To-Do list

Find our To-Do list in forge.typo3.org:

<http://forge.typo3.org/projects/extension-yag/issues>

Feel free to add your bugs and wishes!

ChangeLog

For detailed change logs, visit <https://github.com/YAG-Gallery/yag/commits/master>

Version	Changes:
2.5.2	<p>[BUGFIX] Fix bug #48339: Albums lost after sorting with Dragn Drop</p> <p>[BUGFIX] #48160 Context identifier cannot be only numeric - prefix a "c" whenever the contextIdentifier is only numeric</p> <p>[BUGFIX] #48319 SqlErrorException after upgrade to YAG 2.5.1 fixed</p> <p>[BUGFIX] #48227 Original string not translated in Partials/Image/LightBoxThumb.html</p> <p>[TASK] Visible thumbs and pre / post list use the same partial now</p>
2.5.1	<p>[BUGFIX]: itemRepository:getRandomItemUIDs: pickRandomItems based on whitelist. Respect enableFields on album and gallery</p> <p>[BUGFIX] ZipPackingService adds file extension if not configured, checks if itemList is empty, cleans up the download filename.</p> <p>[BUGFIX] Fix Zip download link should only download images of current album. Should only appear if current list has images.</p> <p>[BUGFIX] Fixes random selection of images.</p>
2.5.0	<p>[FEATURE] ZipDownload for albums</p> <p>[FEATURE] Replaced the multfile flash uploader (swfupload) with uploadify.</p> <p>[FEATURE] Implemented import via "directory on server" for TYPO3 6.0+</p> <p>[BUGFIX] Fixed Album creation for 6.1 Property Manager</p> <p>[BUGFIX] Adjusted ResolutionFileCacheRepository for 6.1 repositories</p> <p>[BUGFIX] Creation of a new gallery in 6.1 was broken due to date conversion error</p> <p>[BUGFIX] Fixed warning in HeaderInclusion utility</p>

Version	Changes:
2.4.0	<p>[TASK] Refactored MetaData Factory</p> <p>[TASK] Huge refactoring towards object manger usage</p> <p>[FEATURE] YAG now includes a social share widget. OpenGraph infogrmmation is automatically added to the page if the facebook share is activated</p> <p>[FEATURE] Disqus commenting partial</p> <p>[FEATURE] Image-List can be rendered as RSS.</p> <p>[FEATURE] GPS Data are now parsed and available in the meta data</p> <p>[FEATURE] IPTC title added to the meta data</p> <p>[FEATURE] Image / Album / Gallery descriptions are now richtext fields</p> <p>[FEATURE] Javascript inclusion can now be configured by typoscript to header / footer and inline.</p> <p>[FEATURE] Using a checkbox in the YAG extension configuration, you can now configure YAG to flush its resolution file cache with the TYPO3 cache clear command.</p> <p>[FEATURE] The download link beneath single images now sends the file as download while protecting it from grabbing the whole database</p> <p>[FEATURE] Albumlist is sortable by date</p> <p>[FEATURE] MetaData encoding is recognized and metadata is automatically encoded to UTF8</p> <p>[FEATURE] Improved Plugin Information</p> <p>[BUGFIX] Deleted Pages are not longer seletced in Backend. #46702</p> <p>[BUGFIX] Breadcrumb not showin "All Albums" in Album List</p> <p>[BUGFIX] Album title is now also linked</p> <p>[BUGFIX] #45073 Fixed pid detector. TYPO3 caching was not able to handle comments in multi-line method calls (parameters spread over several lines with comments in each line).</p> <p>[BUGFIX] Fix album / gallery count in backend list</p> <p>[BUGFIX] Fix RealURL caching Bug</p> <p>... lots of other minor bugfixes ...</p>
2.3.0	<p>ADD: UncachedItemList as PluginMode</p> <p>ADD: Flexform configurable filter to pick random items from itemList (sponsored by viazenetti.de)</p> <p>ADD: Links of ImageList items can be configured via flexform to link to another page and trigger YAG actions there.</p> <p>ADD: A flag in flexform can be used to reset the context</p> <p>ADD: PagerType can be set via typoscript. Availabe are "default" and "delta"</p> <p>ADD: YAG now officially supports all image-Types supportet by TYPO3</p> <p>ADD: #44570 YAG respects meaningfulTempFilePrefix in resolution filenames</p> <p>CHG: Improved Flexform Structure</p> <p>Lots of code-refactoring and clean-up!</p> <p>FIX: XMP Parser</p> <p>FIX: Mimetype is now set correctly</p> <p>FIX: Bug #43846 Invalid character in TS configuration for T3 < 6.0</p> <p>FIX: Bug #44505 Cash fails with RealURL hook because of an error in the url hashing</p> <p>FIX: Bug #44517 RealURL hook won't work when plugin is inserted into root page</p> <p>FIX: Bug #44556 Frontend uploading: images are not saved on the server</p>
2.2.1	<p>Minor Bugfixes:</p> <ul style="list-style-type: none"> - Removed confusing ItemList / AlbumList - Fixed some Label Bugs - Removed Delete Link in default single image view.

Version	Changes:
2.2.0	YAG is now compatible to TYPO3 6.0 Implemented HTML5 Drag & Drop uploading.
2.1.0	The Backend Directory Importer now supports file mounts. Some minor changes. Fixed Bug: #42783, #43079
2.0.0	Major release, now supporting PIDs to store yag records. Make sure you read update section " Upgrading from yag 1.x.x to yag 2.x.x " CHG: Source selector in flexform now requires PID to be selected ADD: #32110 access rights for galleries and albums ADD: #34477 yag asks you to mark page as yag folder / select yag folder if you use module on a page that is not a yag folder yet. ADD: Updated documentation to match changes in v2.0.0 CHG: yag 2.0 depends on pt_extlist 1.0.0 and pt_extbase 1.0.0 ADD: Frontend-Editing has been re-introduced CHG: All backend TypoScript is included as extension TypoScript so no inclusion of TypoScript is necessary anymore to work in backend. By version 2.0 we skipped compatibility with TYPO3 version 4.5! Make sure to update your TYPO3 version to 4.6 at least!
1.5.4	FIX: #41589 Fixed dependency to wrong pt_extlist interface in 1.5.3
1.5.3	FIX: Fixed bug concerning deletion of albums due to missing dependency injection in domain models.
1.5.2	TER problems, no changes compared to 1.5.1
1.5.1	Fixed a lot of Bugs, thanks for your bug-reports and patches: #39211. Now missing directory is re-created if origis directory is deleted and file-not-found images can be created within this newly created directory. #37239 CSS does not align album/gallery description properly in frontend #39546 absRefPrefix not respected in Resource ViewHelper #34770: Problems with RealURL hook and defaultToHTMLsuffixOnPrev #35934: Random Single View tries to display not existent images. #39211: Better Error-Message if Original Images are moved #39540 Cyrillic letters are not properly saved in "Images Overview" #39006 Titles not editable in tab »edit images« #39466: Problem with result image creation in BE #38482 (Resolved): XMP-Parsing: Website is imported as Email
1.5.0	CHG: We now use jQuery fancybox as lightbox for the lightbox theme, wich is also way more configurable compared to the old lightbox. The lightbox theme now uses squared thumbnails. FIX BUG: #34483, #34478, #34222, #33003, #32979
1.4.5	FIX: BUG #34166, #33905, # 33902, #32601. Thx to the bug reporters!
1.4.4	FIX: BUG #32769 (thx to Steffen Gebert), #32634, #32622 (thx to Steffen Gebert), #32623 (thx to Steffen Gebert)
1.4.2	FIX: BUG #32097, #32129, #32137
1.4.1	ADD: Bootstrap class to easily integrate YAG in a third party extension. ADD: Typoscript Settings can now be retrieved from configurationBuilder in a Javascript compliant format

Version	Changes:
1.4.0	ADD: ItemsPerPage can now be set via FlexForm ADD: New widget „random image“ available ADD: Sorting of gallery list, album list and image list can now be set in FlexForm. FIX: Lightbox can now thumb through all images of an album not only paged items. FIX: Deletion of albums should now work again. RFT: Some code-refactoring.
1.3.3	FIX: Bug #31327, #31260, #31275 – made YAG compatible to V 4.6
1.3.2	FIX: Bug #30692, #30909
1.3.0/1.3.1	RFT: Removed unused controller actions from ext_localconf.php ADD: Feature bulk edit for images and albums ADD: MetaData is now processed correctly ADD: Tags are now imported from keywords ADD: Gallery uid filter for filtering certain galleries in gallery list FIX: Call-time pass-by-reference in returnUrl hook ADD: Russian translation, thanks to Sergey Alexandrov ADD: Images can now be sorted by different criteria in backend ADD: Resolutions can be rebuild for selected themes ADD: Status report now gives information about configuration and external libraries ADD: Newly imported images are now always added at the end of the album FIX: Sorting images in backend manually now works on each page individually FIX: Standalone template is working again DEL: Removed non-used import controller ADD: Filehash is now written to item on import. Prevention of duplicate import. FIX: Date can be set for gallery and album. RFT: Performance improvements in backend ADD: Added some styling to pager in backend FIX: Many minor and major bugfixes
1.2.4	FIX: It was not possible to delete images.
1.2.3	FIX: Fixed Bug #29187, #29393, #27964
1.2.1	CHG: Removed unused tabs from content element form FIX: Fixed Pager FIX: Removed warnings that showed up in different situations
1.2.0	RFT: Removed pt_tools. YAG now uses pt_extbase for external tools. FIX: Fixed Bug #27319, #27737, #27312, #27370 due to non existing original image file
1.1.9	ADD: Pager partial can now be set via TS CHG: Upload button in backend now looks like upload button
1.1.8	FIX: Removed some useless var_dump()
1.1.7	ADD: Resolutions for album thumb and gallery thumb can now be set individually
1.1.6	FIX: Bug #27172 – Umlaute are now correctly displayed in Front- and Backend.
1.1.5	FIX: Bug #26740 – Insert plugin in backend crashes under some circumstances. FIX: Bug #26111 - Fileadmin importer is not able to import folders with blanks
1.1.4	DEL: Removed RBAC installation routine FIX: Added some escaping for title and description RFT: Added some frontend styling CHG: Added .jpeg, .JPG and .JPEG as possible file endings for importers RFT: Removed unused gallery:album mm table from SQL definition FIX: Some minor bugfixes
1.1.3	CHG: Improvements in performance. Tested handling of up to 50k images. Seems to be quite fast now :-) CHG: Directory importer comes with directory picker now. CHG: ZIP import now can handle zipped folders. FIX: BUG #25454, fixed 1st level resolution file cache. ADD: Added some documentation.

Version	Changes:
1.1.2	<p>CHG: Changed TypoScript structure. Previously inserted plugins still remain functional, but if you edit the Plugin configuration, you have to select your gallery / album / item again.</p> <p>FIX: Paging in SpecificAlbum mode throws an exception. You have to edit your album and select the mode again.</p> <p>CHG: Plugins now displays mode / album / theme in the page content element overview</p> <p>CHG: Album / gallery description is displayed in the module</p>
1.1.1	<p>CHG: Galleries and Albums are now again sortable. (a change in the database was necessary!)</p> <p>CHG: Complete Extension is now translatable.</p> <p>ADD: Added german translation (Thanks to Matthias Kuchem).</p> <p>CHG: Add all parameters to the URL instead of using the stateHash</p> <p>CHG: Removed all tables from the list module. All data should be administrated by the YAG module.</p> <p>CHG: ReolutionFileCache-Files are now identified by parameter hash.</p> <p>FIX: Many more minor bugs.</p>
1.1.0	<p>RFT: RBAC is no longer a dependency. Features will be outsourced to yag_feedit extension</p> <p>FIX: German translations are removed from JS files</p> <p>FIX: Added lots of translations</p> <p>RFT: Removed lots of CSS and JavaScript to make Backend work better (thx to Matthias!)</p> <p>ADD: Page cache is cleared, if objects change</p> <p>FIX: Thumbs are now generated on Windows platforms</p> <p>FIX: Directory import now respects filetypes correctly</p> <p>RFT: Image processing now uses T3 standard libs and has many configurations now</p>
1.0.10	Bugfix release
1.0.9	Bugfix release
1.0.8	FIX: Fixed some bugs concerning contextIdentifier to enable tt_news integration
1.0.7	<p>FIX: Multiple instances of the plugin can now be positioned on the same page with different themes</p> <p>FIX: Bug #13820 – SWUploader not working without FE Session. Thanks to PETIT Yann</p> <p>FIX: Bug #13822 - No thumbnails are created on Windows servers. Thanks to PETIT Yann</p> <p>ADD: Caching has been refactored</p> <p>RFT: Image ViewHelper has been moved to another directory</p> <p>ADD: Implemented automatic cache cleaning, when objects change</p> <p>CHG: Added lazy loading for domain models</p> <p>ADD: Single image view now has Download-Link for full-res image</p> <p>ADD: Documentation</p>
1.0.6	<p>ADD: Implemented caching</p> <p>ADD: Documentation</p> <p>RFT: Reduced number of SQL queries in Domain Model</p>
1.0.5	Problems with TER upload – no changes
1.0.4	<p>ADD: Documentation</p> <p>FIX: Bug #13763 / display error message when static template is not included</p> <p>ADD: Breadcrumbs show "all galleries" when gallery list is shown</p> <p>ADD: Implemented pageCacheManager, clearAllPageCacheAction to Backend Controller</p> <p>FIX: #13775 Adding a new album to a gallery shows right gallery now</p> <p>FIX: #13776 After importing from directory on server, the album list is shown</p> <p>FIX: Fixed bug in directory crawler</p>
1.0.3	<p>ADD: Documentation</p> <p>ADD: Some translation</p> <p>FIX: Dependencies are set correctly in ext_emconf.php</p>
1.0.0	First release of this extension.

We are currently using GitHub.com for collaborative development. You can find all commit messages and an up-to-date trunk of this extension on:

<https://github.com/yag-gallery>

If you would like to join the team, send us an e-mail (info@yag-gallery.de)