

# **TYPO3 CMS 6.2 LTS – What's New**

Summary of the new features, changes and improvements

Created by:

Patrick Lobacher and Michael Schams

27/March/2014

Creative Commons BY-NC-SA 3.0



# TYPO3 CMS 6.2 LTS - What's New

---

## Chapter Overview

Introduction

Install Tool

Responsive Images

Backend Changes

TSconfig & TypoScript

Package Management

In-Depth Changes

Application Programming Interface

Extbase & Fluid

Upgrade to TYPO3 CMS 6.2 LTS

MythBuster

Sources and Authors

## Introduction (Quick Facts)

# Introduction

---

## TYPO3 CMS 6.2 LTS: The Facts

- Focus on:
  - Smooth Migration
  - Robust and Secure Foundation
  - User Happiness
  - Modern Technologies/Interoperability

- Release Manager:
  - Ernesto Baschny  
ernesto.baschny (at) typo3.org  
Twitter: @baschny

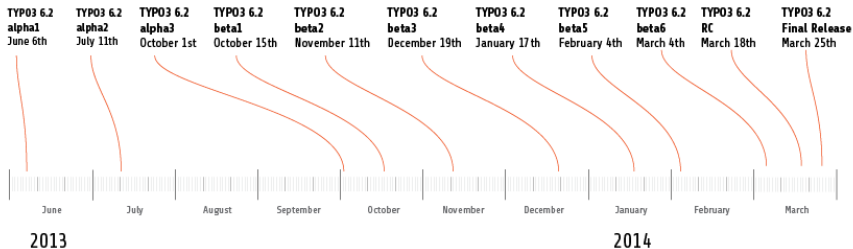


# Introduction

---

## TYPO3 CMS 6.2 LTS: The Facts

- Release date: 25 March 2014
- Development and release timeline:



# Introduction

---

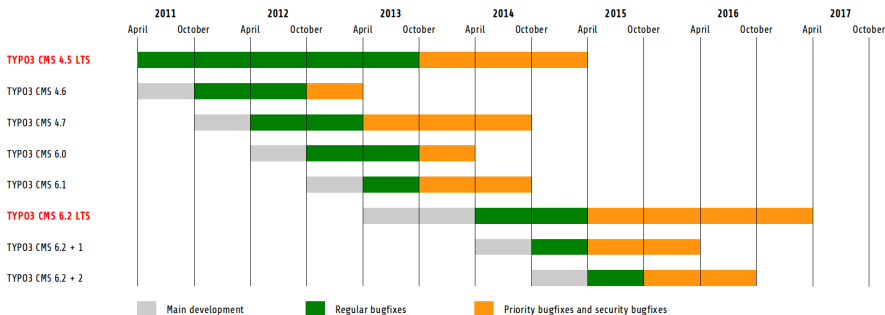
## TYPO3 CMS 6.2 LTS: The Facts

- System Requirements
  - PHP v5.3.7 - v5.5.x
  - MySQL v5.1.x - v5.6.x
- End of maintenance: March 2017
- TYPO3 CMS 6.2 is a **Long Term Support (LTS)** release (3 years support!)

# Introduction

## TYPO3 CMS 6.2 LTS: The Facts

### ■ TYPO3 CMS release agenda:



## Chapter 1: The Install Tool



# Install Tool

---

## Installation

- Only one package is required for an installation:  
`typo3_src-6.2.x.tar.gz` (file size: approx. 20MB)
- "Dummy" and "Blank" packages became obsolete
- Installation:
  - Extract source package into web root directory
  - Create symbolic links as required
  - Point web browser to your web root
  - TYPO3 Installer starts 1-2-3-4-5-steps wizard

# Install Tool

---

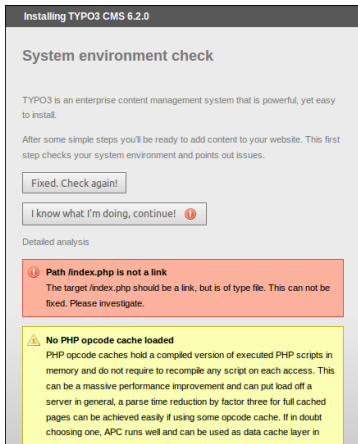
## Installation

- Installer ensures that all required files and directories are in place
- Files required for a custom setup will be created automatically
- The following symbolic links must exist:
  - `typo3_src` (points to TYPO3 source directory)
  - `typo3` (points to directory: `typo3_src/typo3`)
  - `index.php` (points to file: `typo3_src/index.php`)
- No further files/directories are required to install TYPO3!
- Directory `t3lib` removed
- Further details: TYPO3 Installation and Upgrade Guide  
<http://docs.typo3.org/typo3cms/InstallationGuide>

# Install Tool

## Re-Development

- Re-developed from scratch using Fluid
- First step tests system environment and reports issues
- Reported issues can be fixed (and re-tested) or ignored
- Invalid core setup (e.g. no symbolic links as recommended) is reported as an issue, too



Installing TYPO3 CMS 6.2.0

### System environment check

TYPO3 is an enterprise content management system that is powerful, yet easy to install.

After some simple steps you'll be ready to add content to your website. This first step checks your system environment and points out issues.

Fixed. Check again!

I know what I'm doing, continue! ⓘ

Detailed analysis

**ⓘ Path /index.php is not a link**  
The target /index.php should be a link, but is of type file. This can not be fixed. Please investigate.

**⚠ No PHP opcode cache loaded**  
PHP opcode caches hold a compiled version of executed PHP scripts in memory and do not require to recompile any script on each access. This can be a massive performance improvement and can put load off a server in general, a parse time reduction by factor three for full cached pages can be achieved easily if using some opcode cache. If in doubt choosing one, APC runs well and can be used as data cache layer in TYPO3 CMS.

# Install Tool

---

## Re-Development

- Second step allows users to enter database access details
- Connection types are selectable
  - TCP/IP based connection
  - Socket based connection
- MySQL alternatives are also possible

Installing TYPO3 CMS 6.2.0

### Connect to your database host

If you have not already created a username and password to access the database, please do so now. This can be done using tools provided by your host.

Username


Password

Type

Socket

---

TYPO3 CMS native database implementation is based on mysql. A database abstraction layer allows to run TYPO3 CMS on different database engines like postgres. This is used rather seldom and some core parts and extensions do not fully support this. Your TYPO3 CMS experience might suffer if you choose to install the system on anything different than mysql.



# Install Tool

---

## Re-Development

- Third step allows users to select/create the database (as in TYPO3 < 6.2)
- Fourth step allows users to set a password for the "admin" user (which is also the initial Install Tool password) and a site name

Installing TYPO3 CMS 6.2.0

### Create user and import base data

Import basic database structure and create a backend administrator user. The password can be used to log in to the install tool and to the TYPO3 CMS backend (default: with username "admin").

The table import will drop possibly existing tables!

Username

Password

Show password

---

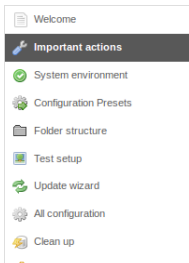
Set a site name

# Install Tool

---

## Clear All Cache

- New function under "Important actions" lets users clear all cache
- This also works, if cache contains invalid PHP code (which possibly locks TYPO3 CMS)
- Bypass a not-working TYPO3 instance by accessing the Install Tool directly: `http://example.com/typo3/install`



**Clear all cache**

This clear all cache function works similar to the cache clearing in the backend but follows a more straight ahead approach and the according backend hooks are not executed.

This method can throw a fatal error if some broken extension is loaded. If you get a white page or a PHP error message, check your system with the broken extension test below.

# Install Tool

---

## Clear All Cache

Sequence of actions when executing "Clear all cache":

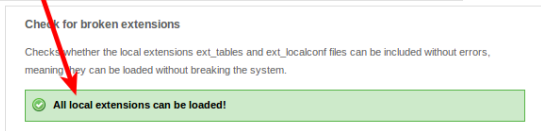
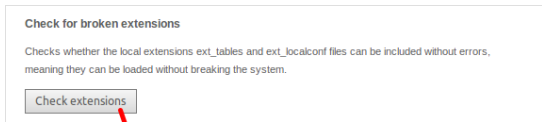
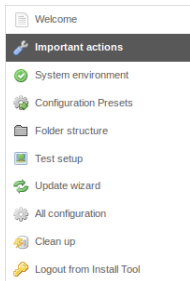
1. Content of directory `typo3temp/Cache` is deleted
2. Database tables `cf_*` are emptied
3. Files `ext_localconf.php` and `ext_tables.php` are loaded from extensions
4. `flushCaches()` are executed

# Install Tool

---

## Check For Broken Extensions

- New function under "Important actions" lets users check, if extensions can be loaded without breaking the system
- Very useful for an update from TYPO3 4.5 to 6.2



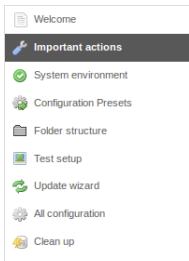


# Install Tool

---

## Salted Passwords

- When creating new backend administrator user via Install Tool, a **salted** password is used  
(requires installed, loaded and configured EXT:saltedpasswords)
- Install Tool password is a **salted** password as well  
(existing MD5 hashes are converted automatically at first login)



**Create backend administrator user**

You should use this function only if there are no admin users in the database, for instance if this is a blank database. After you've created the user, log in and add the rest of the user information, like email and real name.

Username:

Password:

Password again:

# Install Tool

---

## Application Context (1)

- TYPO3 >= 6.2 takes **Application Context** into account  
(known from TYPO3 Flow)
- Environment variable `TYPO3_CONTEXT` sets the context  
(default: Production, sub-context such as Production/Staging possible)

```
# File: .htaccess
```

```
# Rules to set Application Context based on hostname:
```

```
RewriteCond %{HTTP_HOST} ^dev\.example\.com$  
RewriteRule (.*) $1 [E=TYPO3_CONTEXT:Development]
```

```
RewriteCond %{HTTP_HOST} ^www\.example\.com$  
RewriteRule (.*) $1 [E=TYPO3_CONTEXT:Production]
```

```
# Sets an environment variable, which is then available to TYPO3 CMS:  
SetEnv TYPO3_CONTEXT Production
```

# Install Tool

## Presets of TYPO3\_CONF\_VAR Settings

- Certain TYPO3\_CONF\_VAR settings can be configured in Install Tool
- Settings such as debug output, deprecation log, devIPmask and other system logs and log levels
- Build-in contexts: "Production" and "Development" (custom configuration is also possible)

Development / Production settings

TYPO3 can be run in a specific application context by using one of the built-in contexts "Production" (default), "Development" or "Testing". This can be used to provide specific configuration sets for each context. The context can be defined with the environment variable "TYPO3\_CONTEXT" which is usually set through your webserver configuration (e.g. in htaccess).

However, if you don't set a context environment variable, you can still use the install tool to select a configuration preset for "Production" or "Development" context. The goal is to configure a production instance with maximum performance and no debug output that is possibly shown to users, while development instances should enable error output. The configuration preset for "Production" is set by default. As a third alternative, you can enter a custom configuration.

**Production (Active)**  
Production settings turn off debug output, deprecation logs and set logging to warnings and errors only.

**Development**  
Development settings enable debug output, deprecation logs and set logging to info level.

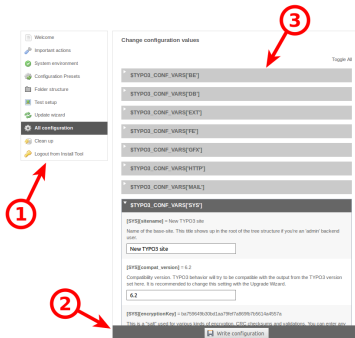
**Custom configuration**  
Custom configuration mixture if no other preset fits.

<input type="text"/>	BE/debug
<input type="text"/>	FE/debug
<input type="text"/>	SYS/devIPmask
<input type="text"/>	SYS/displayErrors
<input type="text"/>	SYSenableDeprecationLog
0	SYS/sqlDebug
2	SYS/systemLogLevel

# Install Tool

## Improved Usability

- Fixed position of menu left-hand-side when scrolling (1)
- Fixed position of button "Write configuration" at the bottom (2)
- Entries in "All Configuration" are grouped (unfold a section by click on headline) and sorted (3)



# Install Tool

---

## Human-Friendly Error Codes

- Meaningful keywords can be used for the following options:  
(TYPO3 < 6.2: numeric values only)

[SYS] [errorHandlerErrors]

[SYS] [exceptionalErrors]

[SYS] [syslogErrorReporting]

[SYS] [belogErrorReporting]

```
[SYS][errorHandlerErrors] = 30466
```

Integer: The E\_\* constant that will be handled by the errorhandler. Not all PHP error types can be handled! Default is E\_ALL & ~(E\_STRICT | E\_NOTICE | E\_COMPILE\_WARNING | E\_COMPILE\_ERROR | E\_CORE\_WARNING | E\_CORE\_ERROR | E\_PARSE | E\_ERROR).

Current PHP error code **30466** represents:

E\_WARNING | E\_USER\_ERROR | E\_USER\_WARNING | E\_USER\_NOTICE |  
E\_RECOVERABLE\_ERROR | E\_DEPRECATED | E\_USER\_DEPRECATED

- An Extbase ViewHelper **format.phpErrorCode** takes care of the conversion to PHP error codes

# Install Tool

## Errors In Folder Structure

- Errors under "Folder Structure" are listed as a badge (circled number)

- Welcome
- Important actions
- System environment
- Configuration Presets
- Folder structure** 3
- Test setup
- Update wizard
- All configuration
- Clean up
- Logout from Install Tool

### These files or folders have errors and may be automatically fixable:

- ❗ Path /index.php is not a link**  
The target /index.php should be a link, but is of type file. This can not be fixed. Please investigate.
- ❗ Path /typo3 is not a link**  
The target /typo3 should be a link, but is of type dir. This can not be fixed. Please investigate.
- ❗ /typo3\_src should be a link, but it does not exist**  
Links can not be fixed by this system
- ⚠ /typo3temp has wrong permission**  
Target permission are 2770 but current permission are 0777

# Install Tool

---

## Core Updates

- Update TYPO3 core to its latest minor version with a click of a button
- Environment variable `TYPO3_DISABLE_CORE_UPDATER=1` disables this feature

**Core update**

The install tool can automatically update the TYPO3 CMS core to its latest minor release if certain criteria are met.

Check for core updates

🌐 Fetching list of released versions from typo3.org

✅ Fetched list of released versions

ℹ Update to release 6.2.1 is available!

Update now

# Install Tool

---

## Miscellaneous

- All forms are CSRF (*cross-site request forgery*) protected
- Install Tool uses a simplified Fluid Standalone View
- Only essential TYPO3 functions are loaded  
(corrupt `ext_localconf.php` or `ext_tables.php` of extensions can not break the Install Tool any more)
- New starting point: `typo3/sysext/install/Start/Install.php`  
Before: `typo3/install/index.php`  
(redirect from old to new URL exists)
- Deactivated cache ensures that Install Tool remains usable, even if cache contains invalid PHP code



# Install Tool

---

## Miscellaneous

- Check if PHP option `xdebug.max_nesting_level` shows a value of 250 or higher (default value "100" possibly causes problems)
- "Relaxed permission check":  
If the web root folder does not have correct permissions set (e.g. "2770"), and this issue can not be fixed, e.g. because the directory does not belong to the system user who runs the Install Tool, the first step of the installation breaks. The option "targetPermissionRelaxed" lowers the severity if permissions are not ideal and allows for continuing installation as long as required sub folders can be created.

# Install Tool

---

## Miscellaneous

- Removed options (keys) from Install Tool  
(and therefore from file `LocalConfiguration.php`, too):

BE/loginLabels

BE/loginNews

BE/useOnContextMenuHandler

EXT/em\_mirrorListURL

EXT/em\_wsdlURL

EXT/extList

EXT/extList\_FE

EXT/noEdit

FE/defaultTypoScript\_editorcfg

FE/simulateStaticDocuments

GFX/noIconProc

GFX/TTFLocaleConv

SYS/additionalAllowedClassPrefixes

SYS/caching/cacheBackends

SYS/caching/cacheFrontends

SYS/extCache

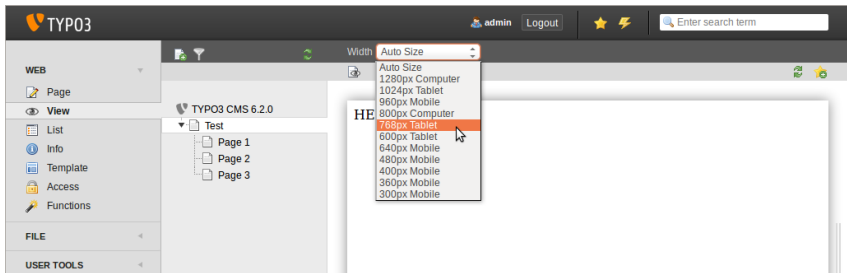
SYS/T3instID

## Chapter 2: Responsive Images

# Responsive Images

## Select Screen Size In Page Preview

- Editors can select various screen sizes in module "View" to test responsive sites



# Responsive Images

---

## Customize Available Screen Sizes

- Screen Sizes are configurable via PageTSconfig:

```
mod.web_view.previewFrameWidths {  
    1780.label = <any LLL or string>  
    1780.height = 145  
}
```

- Width is defined by key (here: 1780), height is optional

- Pre-defined sizes can be found in file:

`typo3/sysext/core/Configuration/DefaultConfiguration.php`

- Labels can be defined via PageTSconfig:

```
mod.web_view.previewFrameWidths {  
    1280.label = LLL:EXT:viewpage/Resources/Private/Language/locallang.xlf:computer  
    1024.label = LLL:EXT:viewpage/Resources/Private/Language/locallang.xlf:tablet  
}
```

# Responsive Images

---

## Responsive Image Galleries

- Additional attributes to implement responsive image galleries
- "CSS styled content" expanded to achieve this
- Example: HTML5 (requires `config.doctype = html5`)

TYPO3 CMS < 6.2:

```
<div class="csc-textpic-imagewrap">...</div>
```

TYPO3 CMS >= 6.2:

```
<div class="csc-textpic-imagewrap"  
  data-csc-images="{register:imageCount}"  
  data-csc-cols="{field:imagecols}">...</div>
```

# Responsive Images

---

## Responsive Image Rendering

- cObject IMAGE renders a so-called "sourceCollection" to support various screen dimensions
- Responsive image rendering for cObjects "text/image" and "image" requires two settings in Constant Editor:  
`styles.content.imgtext.responsive`  
`styles.content.imgtext.layoutKey`
- Valid ("out of the box") options are:
  - `default`: `default <img>-tag`
  - `srcset`: `<img>-tag with alternate sources as srcset-attribute`
  - `picture`: `<picture>-tag with source-child-tags`
  - `data`: `<img>-tag with alternate sources as data-attributes`

# Responsive Images

---

## Property: `layoutKey`

- `layoutKey` defines render layout (this is the HTML code, used for the `<img>`-tag)
- Each option shows unique behaviour for HTML rendering
- Option `default` renders the `<img>`-tag traditionally (this should be used, if frontend is not responsive)
- Implementing a responsive layout requires different image dimensions for various resolutions and screen sizes
- Depending on HTML framework, browser capabilities and JavaScript library (for progressive enhancement):
  - use one of the pre-defined layouts or
  - define your own custom layout



# Responsive Images

---

## Property: layout

```
layoutKey = {$styles.content.imgtext.layoutKey}
layout {
  default {
    element = 
  }
  srcset {
    element = 
    source = |*|###SRC### ###SRCSETCANDIDATE###,|*|###SRC### ###SRCSETCANDIDATE###
  }
  picture {
    element = <picture>###SOURCECOLLECTION###</picture>
    source = <source src="###SRC###" media="###MEDIAQUERY###"###SELCLOSINGTAGSLASH###>
  }
  data {
    element = 
    source = data-###DATAKEY###="###SRC###"
  }
}
```

# Responsive Images

---

## Property: `layout.[layoutKey].element`

- `###SRC###`

URL for attribute: `src`

- `###WIDTH###`

Image width (in pixel) for attribute: `width`

- `###HEIGHT###`

Image height (in pixel) for attribute: `height`

- `###PARAMS###`

Additional parameters as defined in cObject IMAGE

- `###ALTPARAMS###`

Additional alternative parameters as defined in cObject IMAGE

# Responsive Images

---

## Property: `layout.[layoutKey].element`

- `###BORDER###`

Border (in pixel) for attribute: `border`

- `###SELFCLOSINGTAGSLASH###`

Closing tag, e.g. `<img ... />` vs. `<img ... >`

(depends on `config.xhtmlDoctype` or `config.doctype`)

- `###SOURCECOLLECTION###`

Additional image sources, depends on usage of responsive web design.

Exact values are defined in key: `layout.[layoutKey].source`

# Responsive Images

---

## Property: sourceCollection.[dataKey]

- Default sourceCollection of EXT:css\_styled\_content
- Writing your own sourceCollection is highly recommended

```
sourceCollection {
  small {
    width = 200
    srcsetCandidate = 600w
    mediaQuery = (max-device-width: 600px)
    dataKey = small
  }
  smallRetina {
    if.directReturn = 1
    width = 200
    pixelDensity = 2
    srcsetCandidate = 600w 2x
    mediaQuery = (max-device-width: 600px) AND (min-resolution: 192dpi)
    dataKey = smallRetina
  }
}
```

# Responsive Images

---

## Further Resources

- Working code example:  
[http://wiki.typo3.org/Responsive\\_Image\\_Rendering](http://wiki.typo3.org/Responsive_Image_Rendering)
- Article by Sven Wolfermann on typo3.org:  
<http://typo3.org/news/article/responsive-image-rendering-in-typo3-cms-62/>
- W3C specification:  
<http://www.w3.org/html/wg/drafts/srcset/w3c-srcset/>  
<http://www.w3.org/TR/html-picture-element/>
- Working-Draft of the "Responsive Image Community Group":  
<http://responsiveimages.org>

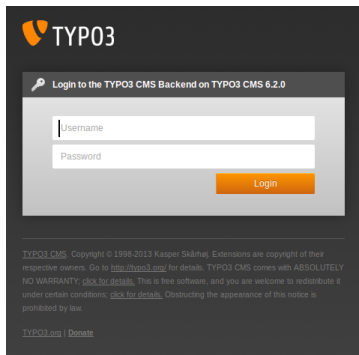
## Chapter 3: Backend Changes

# Backend Changes

---

## Backend Login

- Autofocus on username field in the backend login form (HTML5 attribute: `autofocus="autofocus"`)



The screenshot shows the TYPO3 CMS Backend Login interface. At the top left is the TYPO3 logo. Below it is a search icon and the text "Login to the TYPO3 CMS Backend on TYPO3 CMS 6.2.0". The main form area contains two input fields: "Username" and "Password". The "Username" field has a vertical cursor at the beginning, indicating it is the active field. Below the input fields is an orange "Login" button. At the bottom of the form, there is a small copyright notice: "TYPO3 CMS. Copyright © 1999-2013 Kasper Skårhøj. Extensions are copyright of their respective owners. Go to <http://typo3.org> for details. TYPO3 CMS comes with ABSOLUTELY NO WARRANTY; [click for details](#). This is free software, and you are welcome to redistribute it under certain conditions; [click for details](#). Obstructing the appearance of this notice is prohibited by law." Below the notice are links for "TYPO3.org" and "Donate".

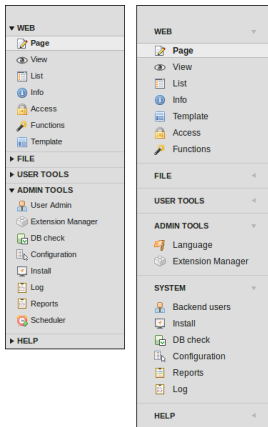
# Backend Changes

## Visual Appearance

- Improved usability by livening the layout up
- Margins between module items (left-hand-side column) increased
- Based on a 12px grid, which has been doubled

Left: TYPO3 4.5

Right: TYPO3 6.2



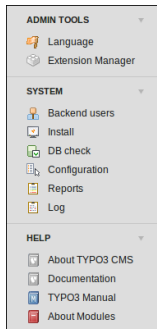


# Backend Changes

---

## Visual Appearance

- Modules in left-hand-side column restructured
- Module "ADMINTOOLS" divided into two parts:
  - **ADMINTOOLS** ("Languages" and "Extension Manager")
  - **SYSTEM** (low-level tools, which do not show the page tree column)
- Module "TypoScript Help" removed (obsolete)

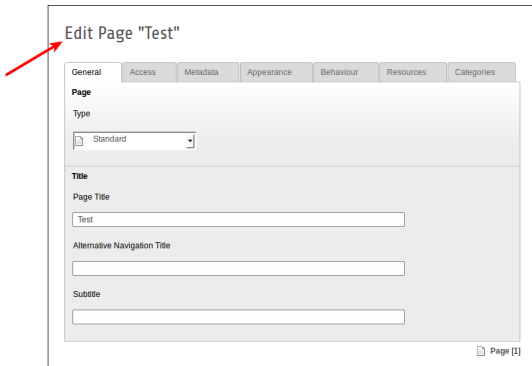


# Backend Changes

---

## Visual Appearance

- `<h1>`-headlines in main area use TYPO3 font "Share" consistently



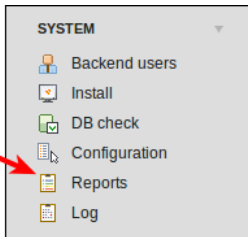
The screenshot shows the 'Edit Page' interface for a page titled 'Test'. The page title 'Test' is displayed in the 'Page Title' field, which is highlighted by a red arrow. The interface includes a navigation bar with tabs for 'General', 'Access', 'Metadata', 'Appearance', 'Behaviour', 'Resources', and 'Categories'. The 'Page' section contains a 'Type' dropdown menu set to 'Standard'. The 'Title' section contains three input fields: 'Page Title' (containing 'Test'), 'Alternative Navigation Title', and 'Subtitle'. The 'Page [1]' label is visible in the bottom right corner.

# Backend Changes

---

## Visual Appearance

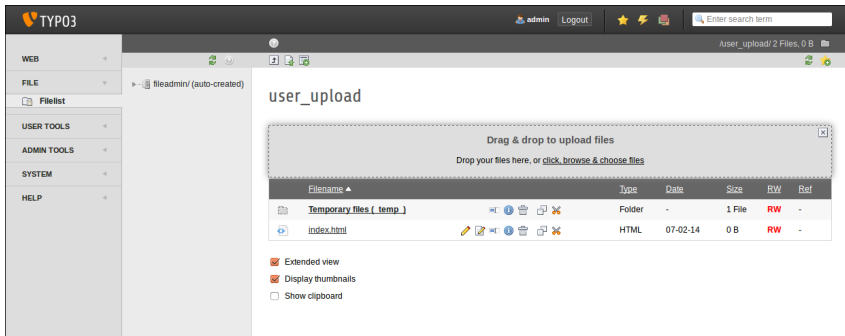
- Module "Reports" shows new icon



# Backend Changes

## Drag&Drop File Upload (1)

- HTML5 Drag&Drop file upload functionality implemented in filelist



The screenshot displays the TYPO3 CMS backend interface. The top navigation bar includes the TYPO3 logo, the user name 'admin', a 'Logout' button, and a search bar. The left sidebar contains a menu with categories: WEB, FILE, USER TOOLS, ADMIN TOOLS, SYSTEM, and HELP. The main content area shows the 'user\_upload' directory. A prominent grey box with a dashed border contains the text 'Drag & drop to upload files' and 'Drop your files here, or [click, browse & choose files](#)'. Below this is a table listing files and folders:

Filename	Type	Date	Size	RW	Ref
Temporary files ( temp )	Folder	-	1 File	RW	-
index.html	HTML	07-02-14	0 B	RW	-

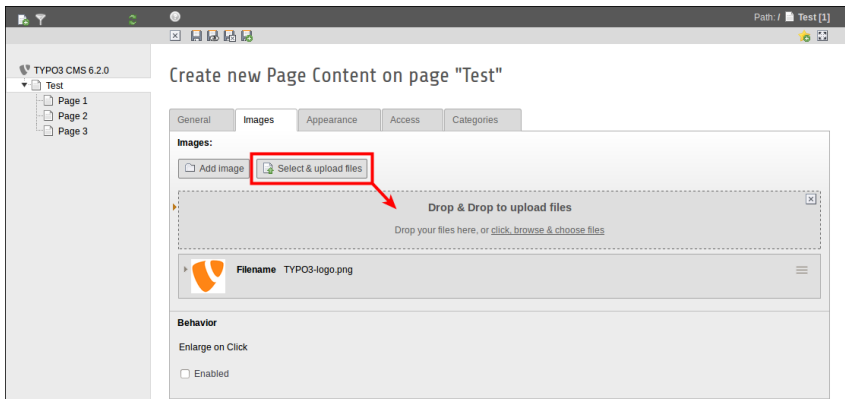
Below the table, there are three checkboxes for filelist settings:

- Extended view
- Display thumbnails
- Show clipboard

# Backend Changes

## Drag&Drop File Upload (2)

- ...and via content elements (button: "Select & upload files")























# Backend Changes

---

## Usability: Backend User List

- Username and real name is shown (first column in list view)
- Click on (user)name links to edit user record
- Delete button added to list view

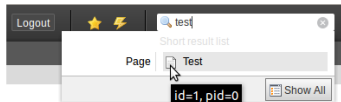
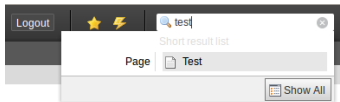
Username / Real Name				Last login
 <b>administrator</b> Firstname Lastname	 Compare	  		22-02-14 03:56
 <b>ajolie</b> Angelina Jolie	 Compare	     		Never
 <b>bpitt</b> Brad Pitt	 Compare	     		Never
3 Users				

# Backend Changes

---

## Live Search

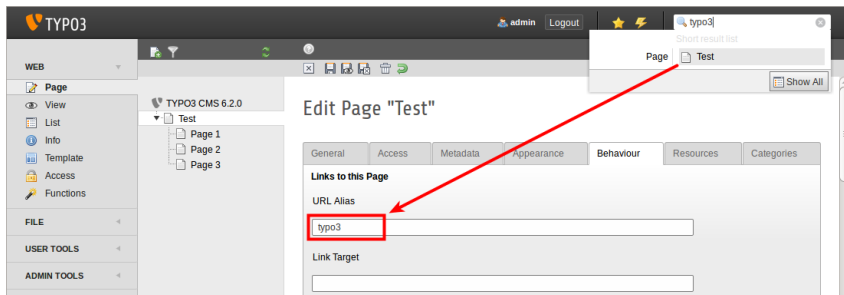
- Tooltip shows UID as well as PID in "livesearch"
- When, after a search, the edit form is closed again, the list view of the page is shown (not an empty page)



# Backend Changes

## Live Search

- In TYPO3 < 6.2, for pages, only database fields `title` and `uid` are taken into account
- In TYPO3 >= 6.2, field `alias` can be added to search (requires `UserTSconfig: options.pageTree.searchInAlias = 1`)



The screenshot displays the TYPO3 CMS 6.2.0 administration interface. On the left, a sidebar contains navigation menus for 'WEB', 'FILE', 'USER TOOLS', and 'ADMIN TOOLS'. The main content area shows the 'Edit Page "Test"' configuration. The 'Links to this Page' section is active, and the 'URL Alias' field is highlighted with a red box, containing the text 'typo3'. A red arrow points from a search result dropdown above to this field. The search dropdown shows a 'Page' result for 'Test' with a 'Show All' button. The top navigation bar includes the TYPO3 logo, user information ('admin'), and a 'Logout' button.

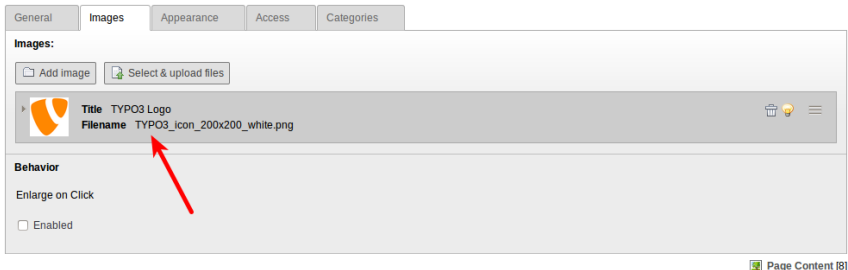


# Backend Changes

---

## File Abstraction Layer

- Title and filename are shown in FAL element header

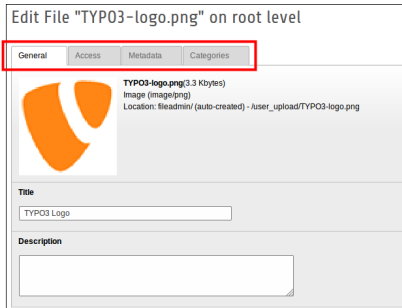
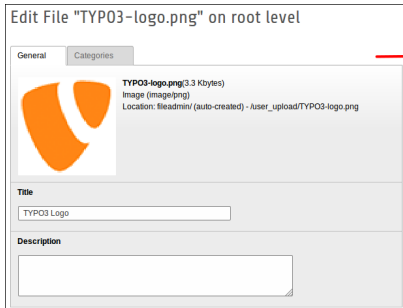


The screenshot displays the TYPO3 backend interface for managing images. At the top, there are tabs for 'General', 'Images', 'Appearance', 'Access', and 'Categories'. The 'Images' tab is selected. Below the tabs, the 'Images:' section contains two buttons: 'Add image' and 'Select & upload files'. A list of images is shown below, with the first entry selected. The entry has a small thumbnail of the TYPO3 logo on the left. To the right of the thumbnail, the 'Title' is 'TYPO3 Logo' and the 'Filename' is 'TYPO3\_icon\_200x200\_white.png'. A red arrow points to the 'Filename' text. To the right of the filename, there are three icons: a trash can, a lightbulb, and a hamburger menu. Below the image list, the 'Behavior' section is visible, with the 'Enlarge on Click' option checked and the 'Enabled' checkbox checked. At the bottom right of the interface, there is a 'Page Content [8]' indicator.

# Backend Changes

## File Abstraction Layer (EXT:filemetadata)

- System extension "filemetadata" add tabs to show meta data (extension is shipped with the core, but not installed by default)



# Backend Changes

---

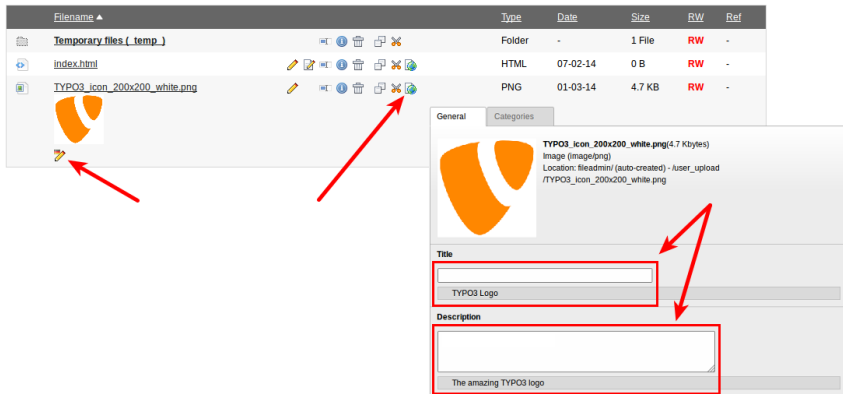
## File Abstraction Layer (EXT:filemetadata)

General	Access	Metadata	Categories
<b>Creator</b>			
<input type="text"/>			
Creator Tool	Publisher	Source	
<input type="text"/>	<input type="text"/>	<input type="text"/>	
<b>Geo Location</b>			
Country	Region	City	
<input type="text"/>	<input type="text"/>	<input type="text"/>	
Latitude	Longitude		
<input type="text" value="0.0000000000000000"/>	<input type="text" value="0.0000000000000000"/>		
<b>Metrics</b>			
Width	Height	Unit	Color Space
<input type="text" value="68"/>	<input type="text" value="68"/>	<input type="text"/>	<input type="text"/>

# Backend Changes

## File Abstraction Layer

- It is now possible to translate FAL meta data into frontend languages



The screenshot displays the TYPO3 file manager interface. On the left, a file list shows three items: a temporary folder, an index.html file, and a PNG image file named 'TYPO3\_icon\_200x200\_white.png'. A red arrow points to the edit icon for this file. On the right, a detailed view of the selected file is shown. The file name is 'TYPO3\_icon\_200x200\_white.png' (4.7 Kbytes). The location is '/User\_upload/TYPO3\_icon\_200x200\_white.png'. The 'Title' field contains 'TYPO3 Logo' and the 'Description' field contains 'The amazing TYPO3 logo'. Red arrows indicate the flow from the file list to the detailed view and from the edit icon to the title and description fields.

Filename	Type	Date	Size	RW	Ref
Temporary files (.temp_)	Folder	-	1 File	RW	-
index.html	HTML	07-02-14	0 B	RW	-
TYPO3_icon_200x200_white.png	PNG	01-03-14	4.7 KB	RW	-

**General** Categories

**TYPO3\_icon\_200x200\_white.png** (4.7 Kbytes)  
Image (image/png)  
Location: fileadmin/ (auto-created) - User\_upload  
/TYPO3\_icon\_200x200\_white.png

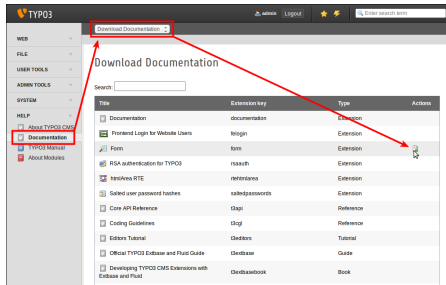
**Title**  
TYPO3 Logo

**Description**  
The amazing TYPO3 logo

# Backend Changes

## Module: Documentation

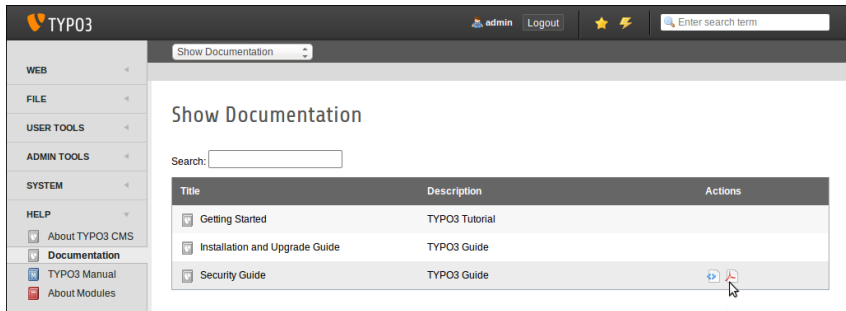
- Module "Documentation" allows BE users to download and view manuals
- New TYPO3 installations load this module by default
- Function "Download Documentation" downloads manuals (see illustration)
- Use the Extension Manager to load "Documentation" in an updated TYPO3 installation





# Backend Changes

## Module: Documentation

- Function "Show Documentation" displays downloaded manuals



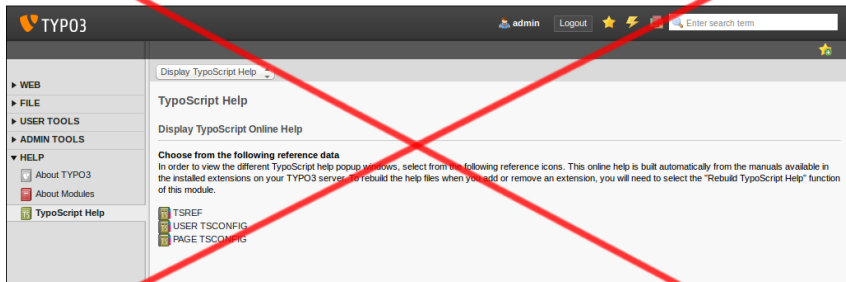
The screenshot displays the TYPO3 Backend interface for the 'Show Documentation' module. The top navigation bar includes the TYPO3 logo, user information (admin), a Logout button, and a search input field. The left sidebar contains a menu with categories: WEB, FILE, USER TOOLS, ADMIN TOOLS, SYSTEM, and HELP. The 'HELP' category is expanded, showing 'About TYPO3 CMS', 'Documentation', 'TYPO3 Manual', and 'About Modules'. The main content area is titled 'Show Documentation' and features a search input field. Below the search field is a table with the following structure:

Title	Description	Actions
Getting Started	TYPO3 Tutorial	
Installation and Upgrade Guide	TYPO3 Guide	
Security Guide	TYPO3 Guide	 

# Backend Changes

## Removed: TypoScript Help

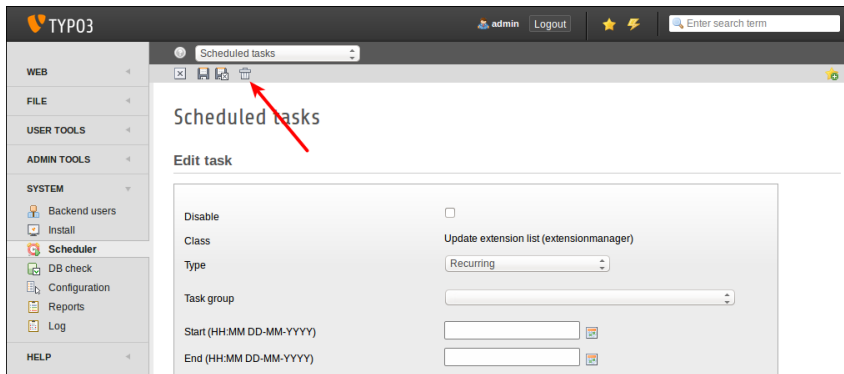
- EXT:tsconfig\_help ("TSconfig Quick Reference") removed (outdated information and not maintained since TYPO3 CMS 4.1)



# Backend Changes

## Scheduler

- Delete scheduler task in edit view  
(in TYPO3 < 6.2, delete function was available in the list view only)



The screenshot shows the TYPO3 Backend interface. The top navigation bar includes the TYPO3 logo, the user 'admin', a 'Logout' button, and a search bar. The left sidebar contains a menu with categories: WEB, FILE, USER TOOLS, ADMIN TOOLS, SYSTEM, and HELP. Under the SYSTEM category, the 'Scheduler' option is selected. The main content area displays the 'Scheduled tasks' section, with a sub-section titled 'Edit task'. A toolbar at the top of the 'Edit task' form contains icons for back, save, refresh, and delete (trash can). A red arrow points to the delete icon. The form fields include: 'Disable' (checkbox), 'Class' (text field with 'Update extension list (extensionmanager)' as a hint), 'Type' (dropdown menu set to 'Recurring'), 'Task group' (dropdown menu), 'Start (HH:MM DD-MM-YYYY)' (text field with a calendar icon), and 'End (HH:MM DD-MM-YYYY)' (text field with a calendar icon).



# Backend Changes


---

## Scheduler

- Description can be assigned to scheduler tasks and shown as subheaders in list view, or as tooltips (see next slide)

### Edit task

Disable	<input type="checkbox"/>
Class	Update extension list (extensionmanager)
Type	Recurring
Task group	
Start (HH:MM DD-MM-YYYY)	<input type="text"/>
End (HH:MM DD-MM-YYYY)	<input type="text"/>
Frequency (seconds or cron command)	86400
Allow Parallel Execution	<input type="checkbox"/>
Description	This task updates the extension list once a day



# Backend Changes

## Scheduler

- Task description as subheader  
(this feature needs to be activated in extension configuration)

	ID	Task	Type	Frequency	Parallel Execution	Last Execution	Next Execution
<input type="checkbox"/>	1	<b>Update extension list (extensionmanager)</b> This task updates the extension list once a day	Recurring	86400	No	-	24-02-14 00:00

Execute selected tasks

- Task description as tooltip ("hover")

	ID	Task	Type	Frequency	Parallel Execution	Last Execution	Next Execution
<input type="checkbox"/>	1	Update extension list (extensionmanager)	Recurring	86400	No	-	24-02-14 00:00

Execute selected tasks

This task updates the extension list once a day

# Backend Changes

## Scheduler

- It is now possible to group scheduler tasks
- Add "scheduler task group" records to root page (UID: 0) and select group in the task

The image displays two screenshots from the TYPO3 CMS 6.2.0 Scheduler interface. The left screenshot, titled "New record", shows a tree view of the Scheduler section. A red box highlights the "Scheduler" folder, which contains a "Scheduler task group" record. A red arrow points from this record to the "Edit task" screenshot on the right. In the "Edit task" screenshot, a red box highlights the "Task group" dropdown menu, which is currently set to "System Maintenance Tasks". The "Edit task" form includes fields for "Disable", "Class" (Update extension list (extensionmanager)), "Type" (Recurring), "Start" (00:00 01-01-2014), "End", "Frequency" (86400), "Allow Parallel Execution", and "Description" (This task updates the extension list once a day).

# Backend Changes

## System Extension: Form

- New post-processor for cObject  
FORM: **redirect**  
(redirect after form submission)
- Value is parsed by `typolink`  
(TypoScript function),  
which means, value can be a  
page ID or a URL

The screenshot shows the 'Form' configuration tab in the TYPO3 backend. The 'POST PROCESSORS' section is expanded, and the 'Redirect' option is selected. A red box highlights the 'POST PROCESSORS' section header and the 'Redirect' dropdown. Below this, the 'Send email' section is visible with fields for recipient, sender, and subject. At the bottom, the 'Redirect' configuration is shown with a 'Destination to' field containing the value '123'. A red box highlights the 'Redirect' section header and the 'Destination to' field. Two red arrows point to the 'POST PROCESSORS' and 'Redirect' section headers.

# Backend Changes

## List Module

- Additional columns "UID" and "PID" in list view for non-admins

The screenshot displays the TYPO3 Backend List Module interface. At the top, a header bar shows "Template (1)". Below it, a table lists the items. The first row is "NEW SITE". The table has columns for "Template Title", "[Ref]", "[uid]", and "[pid]". The values for "NEW SITE" are "-", "1", and "1". A red box highlights the "[uid]" and "[pid]" columns in the table. Below the table, a "Set fields" menu is open, showing a list of fields: "Include Static Templates After Basis Templates", "Setup", "Description", "Static Template Files from TYPO3 Extensions", "Versioning Label", "[uid]", "[pid]", "[tstamp]", "[crdate]", and "[cruser\_id]". A red arrow points to the "[uid]" field in this menu.

Template Title	[Ref]	[uid]	[pid]
NEW SITE	-	1	1

Set fields

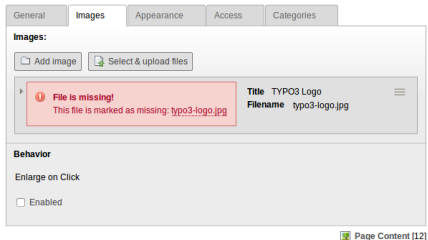
- Include Static Templates After Basis Templates
- Setup
- Description
- Static Template Files from TYPO3 Extensions
- Versioning Label
- [uid]
- [pid]
- [tstamp]
- [crdate]
- [cruser\_id]

# Backend Changes

---

## File Abstraction Layer

- If indexer detects a missing file, a message is shown and a flag in the database record is set
- Module "Reports" also lists this as an issue
- When file re-appears, message and flag are reset



The screenshot shows the TYPO3 CMS backend interface. At the top, there are tabs for "General", "Images", "Appearance", "Access", and "Categories". The "Images" tab is active. Below the tabs, there are two buttons: "Add image" and "Select & upload files". A red warning box is displayed, containing an information icon, the text "File is missing!", and "This file is marked as missing: typo3-logo.jpg". To the right of the warning box, the file details are shown: "Title TYPO3 Logo" and "Filename typo3-logo.jpg". Below the warning box, there is a "Behavior" section with the option "Enlarge on Click" and a checkbox labeled "Enabled". At the bottom right of the interface, there is a small icon and the text "Page Content [12]".

### ⚠ File Abstraction Layer

#### ⚠ Files flagged as missing

1 files

These files are flagged as missing. Restore the files and run the indexer to reset the missing flag.  
fileadmin/ (auto-created) user\_upload/typo3-logo.jpg

# Backend Changes

## Category-based Menus (1)

- Content element "Menu/Sitemap" can create a menu, based on categories

The screenshot displays the TYPO3 CMS backend interface. On the left, a sidebar shows various content elements under the 'Special elements' tab. The 'Special Menu' element is highlighted with a red box. A red arrow points from this box to a configuration window titled 'Menu and Sitemap'. This window contains the following settings:

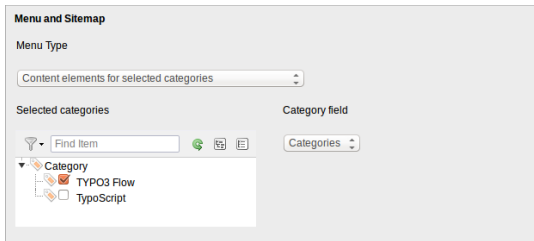
- Menu Type:** A dropdown menu set to 'Pages for selected categories'.
- Selected categories:** A search field with the text 'Find Item' and a 'Categories' dropdown menu.
- Category field:** A list of categories with checkboxes. 'TYPO3 Flow' is checked, and 'TypoScript' is unchecked.

# Backend Changes

---

## Category-based Menus (2)

- Another new type of menu: "Content elements for selected categories"





# Backend Changes

---

## Sorting Categories

- Categories can be sorted now  
(in TYPO3 < 6.2, categories are always sorted alphabetically)



The screenshot shows a table of categories in the TYPO3 backend. The table has a header row with a dropdown arrow, the text 'Category (2)', and a column header 'Title' with a reference icon and '[Ref]'. There are two data rows: 'TYPO3 Flow' and 'TypoScript'. Each row has a set of action icons: edit, info, list, create, sort, lightbulb, and delete. A red arrow points to the sort icon in the 'TypoScript' row.

▼ Category (2)	Title	[Ref]
	TYPO3 Flow	1
	TypoScript	-

# Backend Changes

---

## Category Visibility

- Visibility of categories can be restricted for BE users/groups

The screenshot shows the 'Mounts and Workspaces' configuration page in the TYPO3 Backend. At the top, there are five tabs: 'General', 'Access Rights', 'Mounts and Workspaces' (which is active), 'Options', and 'Access'. Below the tabs, the 'Workspace permissions:' section contains a single checked checkbox labeled 'Edit Live (Online)'. A vertical ellipsis (three dots) is positioned below this section. The 'Category Mounts:' section is a large, light gray area containing a tree view of category mounts. The tree view shows a root 'Category' folder with two sub-items: 'TYPO3 Flow' and 'TypeScript', each with a checkbox. In the bottom right corner of the page, there is a user icon and the text 'Backend user [2]'.

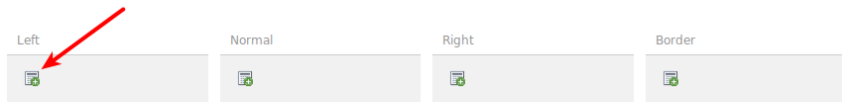
# Backend Changes

---

## Usability

- Icon "new content" is always visible if the column is empty (this helps editors to understand what they can do)

## Test Page



# Backend Changes




---

## Functions

- When creating multiple pages in module "functions", a new checkbox allows editors to hide these pages in menus (very useful, when creating a number of pages at a time)

### Create multiple pages

Create new pages:

Page 1:	<input type="text"/>	Type:  Standard
Page 2:	<input type="text"/>	Type:  Standard
Page 3:	<input type="text"/>	Type:  Standard

Place new pages after the existing subpages

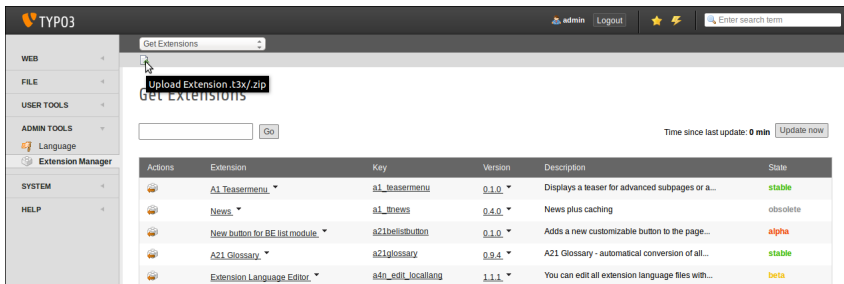
Hide new pages

Hide new pages in menus

# Backend Changes

## Extension Manager

- Upload an extension via the "Get Extensions" function



The screenshot shows the TYPO3 Backend interface. The top navigation bar includes the TYPO3 logo, user information (admin), and a search bar. The left sidebar contains a menu with categories like WEB, FILE, USER TOOLS, ADMIN TOOLS, SYSTEM, and HELP. The main content area is titled 'Get Extensions' and features a search input field and a 'Go' button. A tooltip is visible over the search input, indicating the option to 'Upload Extension (.t3x/.zip)'. Below the search area, there is a table listing installed extensions. The table has columns for Actions, Extension, Key, Version, Description, and State. The extensions listed are:

Actions	Extension	Key	Version	Description	State
	A1 Teasermenu	a1_teasermenu	0.1.0	Displays a teaser for advanced subpages or a...	stable
	News	a1_tnews	0.4.0	News plus caching	obsolete
	New button for BE list module	a21belistbutton	0.1.0	Adds a new customizable button to the page...	alpha
	A21 Glossary	a21glossary	0.9.4	A21 Glossary - automatical conversion of all...	stable
	Extension Language Editor	adn_edit_loclang	1.1.1	You can edit all extension language files with...	beta

# Backend Changes

## Recycler

- Recycler records can be sorted by time stamp  
(this helps users to decide whether to recover a specific record or not)

Recycler

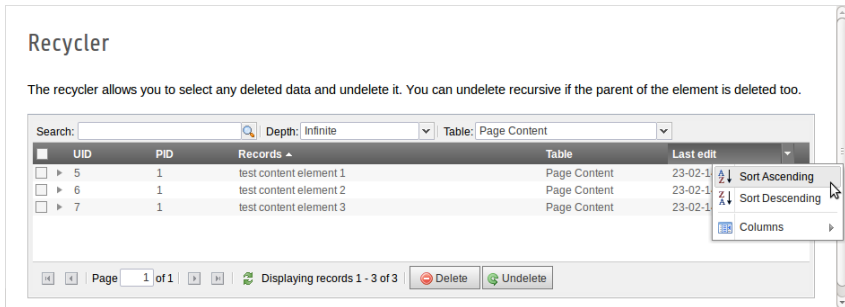
The recycler allows you to select any deleted data and undelete it. You can undelete recursive if the parent of the element is deleted too.

UID	PID	Records	Table	Last edit
<input type="checkbox"/> ▶ 5	1	test content element 1	Page Content	23-02-1
<input type="checkbox"/> ▶ 6	1	test content element 2	Page Content	23-02-1
<input type="checkbox"/> ▶ 7	1	test content element 3	Page Content	23-02-1

Search:  Depth: Infinite Table: Page Content

Sort Ascending  
Sort Descending  
Columns

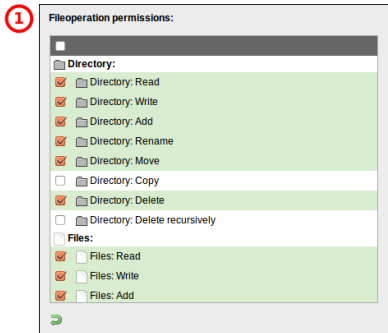
Page 1 of 1 | Displaying records 1 - 3 of 3 | Delete | Undelete



# Backend Changes

## File/Directory Permissions

- Much more granular file/directory permissions for BE users/groups (1)
- This is possible since TYPO3 6.0, but only via UserTSconfig (2)



# Backend Changes

## OpenID (1)

- OpenID for BE user authentication can be configured by using a wizard
- EXT:openid (system extension) is required for this feature

Edit Backend user "brad.pitt" on root level

General | Access Rights | Mounts and Workspaces | Options

**Disable:**

**Username:**

**OpenID Identifier:**

**Password:**

**OpenID registration**

Add an OpenID to your backend user. This OpenID can then be used to log in the TYPO3 backend.

OpenID Identifier:



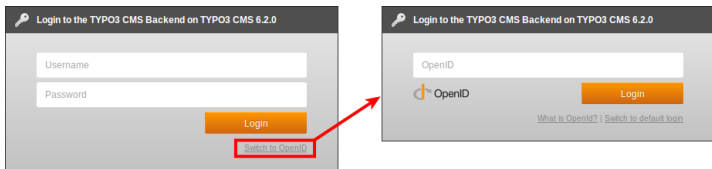


# Backend Changes

---

## OpenID (2)

- OpenID for BE user authentication can be configured by using a wizard
- EXT:openid (system extension) is required for this feature



- Further details about OpenID:  
<http://openid.net>

# Backend Changes

## Workspaces

- Editors/users can define who to notify, without limiting this on the system level
- Tab "All" is now visible to all users

Generate Workspace Preview Link

Path: /Test Page/ [pid: 1]

TYPO3 CMS 6.2.0

- Test Page
  - Page 1
  - Page 2
  - Page 3

LIVE workspace | DRAFT workspace | All

Infinite | all languages

Changed	Live-Title	Current Stage	Actions
▼ Path: /Test Page/ (2 Items)			
<input type="checkbox"/>	Headline 1	Editing	
<input type="checkbox"/>	Headline 2	Editing	

choose Action | choose Mass Action | Page 1 of 1

Legend: edited • moved • created • hidden • deleted

## Chapter 4: TScnfig & TypoScript

# Tsconfig & TypeScript

---

## Include TypeScript

- Include all TypeScript files from a directory (recursive)

```
<INCLUDE_TYPOSCRIPT: source="DIR:directory">
```

```
<INCLUDE_TYPOSCRIPT: source="DIR:EXT:myextension/res/setup">
```

- Order in which files are included:  
alphabetically, first files, then directories

- Limit files to be included by adding `extensions="..."`

```
<INCLUDE_TYPOSCRIPT: source="DIR:directory" extensions="ts">
```

- By default, only files with extensions `ts`, `t3`, `t3s`, `t3c`, `txt` can be included
- This list is configurable (Install Tool):  
`$TYPO3_CONF_VARS['SYS']['tsfile_ext']`

# TSconfig & TypoScript

---

## Include TypoScript

- Relative paths can be passed to INCLUDE\_TYPOSCRIPT, if the inclusion is called recursively from a file
- First include **must be** absolute
- ./ reflects the actual directory of the last include
- ../ reflects the parent directory of the last include
- Examples:

```
<INCLUDE_TYPOSCRIPT: source="FILE:directory/typoscript/setup.ts">
```

```
<INCLUDE_TYPOSCRIPT: source="FILE:./filename.ts">
```

```
<INCLUDE_TYPOSCRIPT: source="FILE:../filename.ts">
```

```
<INCLUDE_TYPOSCRIPT: source="FILE:../directory/filename.ts">
```

# TScnfig & TypoScript

---

## strPad

- Option stdWrap has been added to strPad properties

```
page = PAGE
page.10 = TEXT
page.10 {
    value = Hello World!
    strPad {
        length = 5
        length {
            current = 1
            setCurrent.data = TSFE:page|uid
            setCurrent.wrap = | + 80
            prioriCalc = 1
        }
        padWith = .
    }
}
```

# Tsconfig & TypeScript

---

## `_DEFAULT_PI_VARS`

- `stdWrap` has been added for `_DEFAULT_PI_VARS`
- `_DEFAULT_PI_VARS` are used to set default values for `piVars` (GET/POST variables for an extension)
- TYPO3 < 6.2

```
plugin.tt_news._DEFAULT_PI_VARS {  
    year = 2013  
}
```

- TYPO3 >= 6.2

```
plugin.tt_news._DEFAULT_PI_VARS {  
    year.stdWrap.data = date:Y  
}
```

# TSconfig & TypoScript

## Debug Output

- Debug output for register and page variables:

```
$GLOBALS['TSFE']->register
```

```
$GLOBALS['TSFE']->page
```

- Examples:

```
10 = LOAD_REGISTER
```

```
10.variable = value
```

```
20 = TEXT
```

```
20.data = debug:register
```

```
30 = TEXT
```

```
30.data = debug:page
```

SYS_LASTCHANGED	1376804898
variable	wert
uid	1
pid	0
t3ver_old	0
t3ver_id	0
t3ver_wsid	0
t3ver_label	
t3ver_state	0
t3ver_stage	0
t3ver_count	0
t3ver_tstamp	0
t3ver_move_id	0
t3_origuid	0
tstamp	1376804898
sorting	256
deleted	0



# TScnfig & TypoScript

---

## File Links

- File links offer a description, title text and alternative label text for each file. All three can be accessed via registers:
  - `register:description`
  - `register:titleText`
  - `register:altText`

- Example:

```
# filelinks
tt_content.uploads.20 {
    # link description instead of filename
    labelStdWrap.data = register:description
    # output alternative text
    itemRendering.20.data = register:titleText
}
```

# TSconfig & TypeScript

---

## stdWrap function: replacement (1)

- Option replace of stdWrap-function replacement supports optionSplit now
- Example 1:

```
10 = TEXT
10.value = TYPO3_inspires_people_to_share
10.replacement.10 {
    search = _
    replace = 1 || 2 || 3
    useOptionSplitReplace = 1
}
```

Output:

TYPO31inspires2people3to3share

# Tsconfig & TypeScript

---

## stdWrap function: replacement (2)

- Option replace of stdWrap-function replacement supports optionSplit now
- Example 2:

```
10 = TEXT
10.value = TYPO3 inspires people to share
10.replacement.10 {
  search = #(TYPO3|people|share)#i
  replace = ${1} CMS || all ${1} || collaborate and ${1}
  useOptionSplitReplace = 1
  useRegExp = 1
}
```

### Output:

TYPO3 CMS inspires all people to collaborate and share

# Tsconfig & TypeScript

---

## cObject FILE

- Two registers added to cObject FILES:  
FILE\_NUM\_CURRENT and FILES\_COUNT
- Example:

```
10 = FILES
10 {
  references {
    table = tt_news
    uid.field = uid
    fieldName = media
  }
  renderObj = COA
  renderObj {
    10 = TEXT
    10.value = Renders first file twice
    10.if.isFalse.data = register:FILE_NUM_CURRENT
    20 = TEXT
    20.value = file {register:FILE_NUM_CURRENT} of {register:FILES_COUNT}
    20.insertData = 1
  }
}
```

# TSconfig & TypoScript

---

## Category Menu

- Generate a menu of categories in TypoScript
- Example:

```
page.20 = HMENU
page.20 {
  special = categories
  special {
    # comma-separated list of categories
    value = 1
    # sort by title (stdWrap)
    sorting = title
    # sorting "asc" or "desc" (stdWrap)
    order = desc
    1 = TMENU
    1.NO {
      allWrap = <li> | </li>
    }
  }
}
```

# TScnfig & TypoScript

---

## Access Categories

- Property `categories` allows the access to categories for the `cObject RECORDS`
- Example:

```
# menu of categorized content elements
categorized_content = RECORDS
categorized_content {
    categories.field = selected_categories
    categories.relation.field = category_field
    tables = tt_content
    conf.tt_content = TEXT
    conf.tt_content {
        field = header
        typolink.parameter = {field:pid}#{field:uid}
        typolink.parameter.insertData = 1
        wrap = <li>|</li>
    }
    wrap = <ul>|</ul>
}
```

# Tsconfig & TypeScript

---

## CSS and JavaScript files

- `splitChar` can now be defined for the `allWrap` properties
- The wrap works like the standard `stdWrap.wrap` method now
- Default `splitChar`-character is the pipe symbol: `|`
- This change affects:
  - `includeCSS`
  - `includeJSlibs`
  - `includeJSFooterlibs`
  - `includeJS`
  - `includeJSFooter`

# Tsconfig & TypeScript

---

## Conditions

- Condition `userFunc` accepts multiple arguments now

- TYPO3 < 6.2

```
[userFunc = user_function(argument1)]
```

- TYPO3 >= 6.2

```
[userFunc = user_function(argument1, argument2, ...)]
```

- Example:

```
[userFunc = user_match(checkSubnet, 192.168)]
```

```
function user_match($command, $subnet) {  
    switch($command) {  
        case 'checkSubnet':  
            if (strstr(getenv('REMOTE_ADDR'), $subnet)) { ... }  
    }  
}
```



# Tsconfig & TypoScript

---

## Conditions

- Application context can be determined in conditions
- Wildcards "+" and "\*" and regular expressions are supported
- Examples:

```
[applicationContext = Development/Debugging, Development/Profiling]
  # TYPO3 site in development stage
[global]
```

```
[applicationContext = Production*]
  # TYPO3 site in production stage
  # for example "Production/Live" or "Production/Staging"
[global]
```

```
[applicationContext = /^TestServer\d+$/]
  # TYPO3 site on TestServer1 or TestServer2 or TestServer3, etc.
[global]
```

# TScnfig & TypoScript

---

## Conditions

- When using an IP condition, keyword `devIP` can be used to check if client's IP address matches with `devIpMask` setting in Install Tool
- Example:

```
[IP = devIP]
  page.10 = TEXT
  page.10.value = Hello Developer!
[global]
```

# Tsconfig & TypeScript

---

## Records Without Default Translation

- New option `includeRecordsWithoutDefaultTranslation` retrieves records without a localization parent (but with `languageField` matching the current language)
- Example:

```
pageContent = CONTENT
pageContent {
  table = tt_content
  select.includeRecordsWithoutDefaultTranslation = 1
  ...
}
```

# TSconfig & TypeScript

---

## cObject FILES

- cObject FILES supports begin and maxItems as properties now
- Example:

```
page.10 = FILES
page.10 {
  references {
    table = pages
    uid.data = page:uid
    fieldName = media
  }

  # retrieve up to 5 files, beginning at the first (0):
  begin = 0
  maxItems = 5

  renderObj = TEXT
  renderObj {
    data = file:current:size
    wrap = <p>File size:<strong>|</strong></p>
  }
}
```

# TSconfig & TypoScript

---

## Exclude doktypes From Page Tree

- Specific doktypes can be excluded from the page tree
- Configuration happens in UserTSconfig (therefore user or group specific)
- Examples:

```
# exclude "folder" pages  
options.pageTree.excludeDoktypes = 254
```

```
# exclude "folder" and "standard" pages  
options.pageTree.excludeDoktypes = 254,1
```

# TSconfig & TypoScript

---

## Hide Modules In Backend

- Modules can be hidden in backend
- This has not impact on the access to the module (use the ACL for BE users and groups for restricting access)
- Examples:

```
options.hideModules = file, help
```

```
options.hideModules.web := addToList(func,info)
```

```
options.hideModules.system = BelogLog
```

# TSconfig & TypoScript

---

## Preview Domain

- An alternative domain can be set for page/site previews in PageTS
- Useful for multidomain sites
- Example:

```
TCEMAIN.viewDomain = example.com
```

# TScnfig & TypoScript

---

## Conditions in Backend Layouts

- Backend layouts now support conditions
- Example:

```
backend_layout {
  colCount = 2
  rowCount = 1
  rows {
    1 {
      columns {
        1.name = Main
        1.colPos = 0
        2.name = Right
        2.colPos = 1
      }
    }
  }
}
```

```
[PIDupinRootline = 123]
# remove right column in branch of page ID 123
backend_layout.rows.1.columns.2 >
[global]
```



# TScnfig & TypoScript

---

## Miscellaneous

- Disable/enable "forgot password" link by option `showForgotPassword` (useful, if multiple login forms are included by EXT:felogin on one page)
- HTTP response includes header `Content-length` by default now
  - Speeds up rendering if pipelining is enabled in Apache
  - Can be configured by `config.enableContentLengthHeader`
- Result list of EXT:indexed\_search has `stdWrap-properties` (option: `plugin.tx_indexedsearch.resultlist_stdWrap`)

## Chapter 5: Package Management

# Package Management

---

## Package Manager

- TYPO3 Flow's **Package Manager** ported to TYPO3 CMS
- Development/exploration started during TYPO3 CMS 6.1 development
- This project aims to harmonize package formats
- Extensions in TYPO3 CMS are just a special type of "Packages"
- Main project goals:
  - Proper API for Package Management
  - Vendor Namespace Support
  - Composer Package Support
  - Flow Package Support
  - Autoloader Re-factoring

# Package Management

---

## Package Manager Integration

- Removal of `$TYPO3_CONF['EXT']['extListArray']` from file `typo3conf/LocalConfiguration.php`
- Old content of file `typo3conf/LocalConfiguration.php` copied to `typo3conf/LocalConfiguration.beforePackageStatesMigration.php`
- File `typo3conf/PackageStates.php` contains:
  - status of package (active/inactive)
  - extension location in filesystem
- Extensions in the following directories are automatically detected:
  - `typo3/sysextd/`
  - `typo3/ext/`
  - `typo3/contrib/`
  - `typo3conf/ext/`
  - `Packages/` (*recursive*)

# Package Management

---

## Package Manager Integration

- Two new (additional) files in extension's directory:
  - `composer.json`
  - `Classes/Package.php`
- If extension is required, a protected flag is to be set in file `composer.json`
- If file `PackageStates.php` is missing, it will be (re-)created, containing all extensions, which have the property above set to `TRUE`
- Autoloader receives its own caching backend
- Further information:  
<http://wiki.typo3.org/Blueprints/Packagemanager>

# Package Management

---

## Package Manager Integration

### Example: typo3conf/PackageManager.php

```
return array ('packages' =>
    array (
        'core' =>
            array (
                'manifestPath' => '',
                'composerName' => 'typo3/cms/core',
                'state' => 'active',
                'packagePath' => 'typo3/sysexst/core/',
                'classesPath' => 'Classes/',
            ),
        'workspaces' =>
            array (
                'manifestPath' => '',
                'composerName' => 'typo3/cms/workspaces',
                'state' => 'inactive',
                'packagePath' => 'typo3/sysexst/workspaces/',
                'classesPath' => 'Classes/',
            ),
        ...
    ),
    'version' => 4,
);
```

# Package Management

---

## Package Manager Integration

### Example: composer.json

```
{
  "name": "typo3/cms-indexed-search",
  "type": "typo3-cms-framework",
  "description": "TYPO3 Core",
  "homepage": "http://typo3.org",
  "license": ["GPL-2.0+"],
  "version": "6.2.0",
  "require": {
    "typo3/cms-core": "*"
  },
  "replace": {
    "indexed_search": "*"
  }
}
```

# Package Management

---

## Package Manager Integration

- Packages can also be activated at runtime by using the key:  
`$GLOBALS['TYPO3_CONF_VARS']['EXT']['runtimeActivatedPackages'] = array( packageKey );`
- This key is activated immediately after the initialisation of the Package Management



## Chapter 6: In-Depth Changes

# In-Depth Changes

---

## Normalize.css

- Backend user interface makes use of `normalize.css`, which makes browsers render all elements more consistently and in line with modern standards
- Modern, HTML5-ready, alternative to the traditional CSS reset
- Aims of `normalize.css` are:
  - Preserve useful browser defaults rather than erasing them
  - Normalize styles for a wide range of HTML elements
  - Correct bugs and common browser inconsistencies
  - Improve usability with subtle improvements
  - Explain the code using comments and detailed documentation

# In-Depth Changes

---

## TCA: displayCond Options BIT And !BIT

- Check with a multi-value field in displayCond (bitwise)  
BIT: bit is set, !BIT: bit is not set

Assuming this TCA:

```
'content' => array(
  'label' => '...',
  'config' => array(
    'type' => 'check',
    'items' => array(
      array('Content A', ''),
      array('Content B', ''),
      array('Content C', ''),
    ),
  ),
),
```

Examples:

```
'content_a' => array(
  'label' => '...',
  'displayCond' => 'FIELD:content:BIT:1',
  'config' => array(
    'type' => 'text',
  )
),

'content_b' => array(
  'label' => '...',
  'displayCond' => 'FIELD:content:!BIT:2',
  'config' => array(
    'type' => 'text',
  )
),
```

# In-Depth Changes

---

## Language Updates

- Extbase Command Controller allows language updates for extensions:

```
$GLOBALS['TYPO3_CONF_VARS']['SC_OPTIONS']['extbase']  
  ['commandControllers'][] =  
  'TYPO3\CMS\Lang\Command\LanguageCommandController';
```

- Example call:

```
typo3/cli_dispatch.phpsh extbase language:update de,en,fr
```

- Comma-separated list of locales (e.g. `de,en,fr`) limits the update to these languages
- Without this argument, all languages which are set in module "Languages" are updated

# In-Depth Changes

---

## System Extensions: ReST Manuals

- All system extension manuals are migrated to reStructuredText
- OpenOffice manuals are not longer used and have been removed
- ReST is an easy-to-read, what-you-see-is-what-you-get plaintext markup syntax and parser system
- ReST files of system extensions are stored in:  
`typo3/sysex<extensionkey>/Documentation/*`
- Further information:
  - <http://de.wikipedia.org/wiki/ReStructuredText>
  - <http://wiki.typo3.org/ReST>

# In-Depth Changes

---

## Custom Translation Servers

- Support of custom translation servers for extensions was implemented
- With the use of XLIFF and a new Signal/Slot, this becomes a no-brainer (see next slide for an example)
- A possible translation server solution: **Pootle**
  - online translation management tool with translation interface
  - written in the Python/Django
  - originally developed and released by [translate.org.za](https://translate.org.za)
  - GNU GPL license

# In-Depth Changes

---

## Custom Translation Servers

Example: EXT:myextension/localconf.php

```
/**
 * @var \TYPO3\CMS\Extbase\SignalSlot\Dispatcher $signalSlotDispatcher
 */
$signalSlotDispatcher =
    \TYPO3\CMS\Core\Utility\GeneralUtility::makeInstance(
        'TYPO3\CMS\Extbase\SignalSlot\Dispatcher');

$signalSlotDispatcher->connect(
    'TYPO3\CMS\Lang\Service\UpdateTranslationService',
    'postProcessMirrorUrl',
    'Company\Extension\Slots\CustomMirror',
    'postProcessMirrorUrl'
);
```

# In-Depth Changes

---

## Custom Translation Servers

### Example: EXT:myextension/Classes/Slots/CustomMirror.php

```
<?php
namespace Company\Extensions\Slots;
class CustomMirror {

    /**
     * @var string
     */
    protected static $extKey = 'myextension';

    public function postProcessMirrorUrl($extensionKey, &$mirrorUrl) {
        if ($extensionKey === self::$extKey) {
            $mirrorUrl = 'http://example.com/typo3-packages/';
        }
    }
}
```



# In-Depth Changes

---

## Custom Translation Servers

Expected file/directory structure on server:

```
http://example.com/typo3-packages/  
  '-- <first-letter-of-extension-key>  
    '-- <second-letter-of-extension-key>  
      '-- <extension-key>-l10n  
        |-- <extension-key>-l10n-de.zip  
        |-- <extension-key>-l10n-fr.zip  
        |-- <extension-key>-l10n-it.zip  
        '-- <extension-key>-l10n.xml
```

For example:

```
http://example.com/typo3-packages/m/y/myextension-l10n/myextension-l10n.xml
```

# In-Depth Changes

---

## Custom Translation Servers

Example: <extension-key>-l10n.xml

```
<?xml version="1.0" standalone="yes" ?>
<TERlanguagePackIndex>
  <meta>
    <timestamp>1374841386</timestamp>
    <date>2013-07-26 14:23:06</date>
  </meta>
  <languagePackIndex>
    <languagepack language="de">
      <md5>1cc7046c3b624ba1fb1ef565343b84a1</md5>
    </languagepack>
    <languagepack language="fr">
      <md5>f00f73ae5c43cb68392e6c508b65de7a</md5>
    </languagepack>
    <languagepack language="it">
      <md5>cd59530ce1ee0a38e6309544be6bcb3d</md5>
    </languagepack>
  </languagePackIndex>
</TERlanguagePackIndex>
```

# In-Depth Changes

---

## Automatic t3d Import

- Extensions can now import initial **t3d packages** automatically upon extension installation
- t3d files contain things such as data, relations, files, etc.
- The t3d file has to be named `data.t3d` and located in:  
`EXT:myextension/Initialisation/`
- Import happens once only  
(even if the extension is re-installed later)

# In-Depth Changes

---

## Automatic File Import

- Extensions can now import initial **files** automatically upon extension installation
- Files have to be located in:  
`EXT:myextension/Initialisation/Files/...`
- Files are copied to:  
`fileadmin/<extensionkey>/`
- Import happens once only  
(even if the extension is re-installed later)

# In-Depth Changes

---

## Use An Extension As Repository

- Sometimes extensions depend on customized versions of other extensions or on extensions which have not been released to the official TYPO3 Extension Repository (TER)
- To address this issue, extensions can now be shipped with "other" extensions
- These have to be located in (unpacked):  
`EXT:myextension/Initialisation/Extensions/...`
- Upon extension installation, they are copied to:  
`typo3conf/ext/`
- After that, extension dependencies are resolved

# In-Depth Changes

---

## Install/uninstall extensions via CLI

- Install and uninstall extensions by command line interface (CLI)

- Examples:

```
typo3/cli_dispatch.phpsh extbase extension:install <extensionkey>
```

```
typo3/cli_dispatch.phpsh extbase extension:uninstall <extensionkey>
```

- Note: a backend user **\_cli\_lowlevel** is required for that

# In-Depth Changes

---

## Cascading Deletion Of Child Elements

- TCA now features a setting to enable/disable cascading deletion of child elements
- Relation must be of type "**inline**"
- Default value is TRUE (deletion of inline child records is enabled)
- Example (disable deletion of inline child records):

```
...  
'type' => 'inline',  
'foreign_table' => ...,  
  'behaviour' => array(  
    'enableCascadingDelete' => 0  
  )  
  ...  
)  
...
```

# In-Depth Changes

---

## Multiple Category Fields Per Table

- In TYPO3 < 6.2, it is only possible to do one `makeCategorizable()` call per table (multiple calls would overwrite previous category field declarations)
- Since TYPO3  $\geq$  6.2, multiple category fields per table are possible
- Example:

```
\TYPO3\CMS\Core\Utility\ExtensionManagementUtility::makeCategorizable(  
    $extensionKey,  
    $tableName,  
    $fieldName = 'categories',  
    $options = array(  
        'label' => 'my category'  
    )  
);
```

- Custom labels for each category field can be set in array `$options`



# In-Depth Changes

---

## Backend Layout Data Providers

- In TYPO3 < 6.2, backend layouts are stored in the DB as regular records
- Since TYPO3 >= 6.2, so-called *data providers* can be defined (for example to enable extensions to ship their own backend layout definitions from static files)

- Data providers have to implement the interface:

```
TYPO3\CMS\Backend\View\BackendLayout\DataProviderInterface
```

- and can be registered by:

```
$GLOBALS['TYPO3_CONF_VARS']['SC_OPTIONS']  
    ['BackendLayoutDataProvider'][$_EXTKEY] = 'Classname';
```

# In-Depth Changes

---

## Backend Layout Data Providers

- New API functions for handling of backend layout data providers:

```
'itemsProcFunc' => 'TYPO3\CMS\Backend\View\  
BackendLayoutView->addBackendLayoutItems'
```

```
getBackendLayoutView()->getSelectedCombinedIdentifier($id);  
getBackendLayoutView()->getSelectedBackendLayout();
```

- New PageTSconfig option to exclude backend layouts:

```
options.backendLayout.exclude = default_1, my_extension__headerLayout
```

# In-Depth Changes

---

## Multiple Value Selector (1)

- Filter available items in a multi-select element (by TCA settings)
- For example: enable a text field for individual word filter and pre-define search words a user can select from a drop down box
- To use this new feature, adjust TCA accordingly (e.g. in file `typo3conf/extTables.php`):

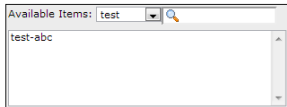
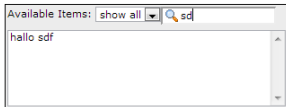
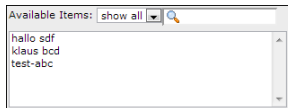
```
$GLOBALS['TCA']['fe_users']['columns']['usergroup']['config']  
    ['enableMultiSelectFilterTextfield'] = TRUE;  
  
$GLOBALS['TCA']['fe_users']['columns']['usergroup']['config']  
    ['multiSelectFilterItems'] = array(  
  
    array('', 'show all'), // no filter  
    array('test', 'test'), // first value: filter, second value: label  
  
    array(  
        'TYPO3',  
        'LLL:EXT:myext/Resources/Private/Language/locallang_db.xlf:tx_myext.label.typo3'  
    ),  
);
```

# In-Depth Changes

---

## Multiple Value Selector (2)

- Two options are available:
  - Select pre-defined values from dropdown box
  - Enter search/filter keyword in an input field
- The result could look like:

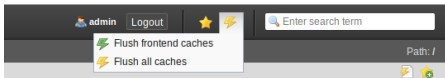


# In-Depth Changes

---

## Cache Groups (1)

- TYPO3 core uses two types of caches:
  - **system-related caches**: class loading cache, configuration cache, l10n\_cache, extbase\_object, extbase\_reflection etc.
  - **frontend-related caches**: cHash cache, page cache, page section cache
- In TYPO3 < 6.2, *clear all caches* empties all caches, which is not ideal
- In TYPO3 >= 6.2, the core uses two cache groups:  
"pages" with all page-related caches and "**system**", which is used for compile-time and configuration caches



# In-Depth Changes

---

## Cache Groups (2)

- Relevant configuration option:

(in files: LocalConfiguration.php/DefaultConfiguration.php)

```
'cache_hash' => array(  
    'frontend' => 'TYPO3\CMS\Core\Cache\Frontend\VariableFrontend',  
    'backend' => 'TYPO3\CMS\Core\Cache\Backend\Typo3DatabaseBackend',  
    'options' => array(),  
    'groups' => array('pages', 'all')  
),
```

- "*Flush all caches*" command does not flush system-related caches any more (only "Clear Configuration Cache" or the Install Tool empty these caches)
- A new userTSconfig option enables non-admins to clear system caches:  
`options.clearCache.system = 1`

**THIS IS A BREAKING CHANGE!**

# In-Depth Changes

---

## TCA: Number of Ticked Checkboxes

- TCA allows validation of number of ticked checkboxes
  - `maximumRecordsChecked`:  
limit number of records system-wide
  - `maximumRecordsCheckedInPid`:  
limit number of records PID-wide (parent ID)
- If a BE user exceeds the max number, the additional tick gets reverted until another record is unchecked
- Example:

```
$tcaConfiguration = array(  
    'type' => 'check',  
    'eval' => 'maximumRecordsChecked',  
    'validation' => array(  
        'maximumRecordsChecked' => 5  
    )  
);
```

# In-Depth Changes

---

## TCA: MM\_oppositeUsage property

- On copying a `sys_category` record, a new MM reference is created, but without setting the "fieldname"
- This value is basically defined from the opposite entity with `MM_match_fields`, but cannot be accessed
- To address this issue, a new property `MM_oppositeUsage` has been introduced for the TCA:

```
'config' => array(  
    'allowed' => '*',  
    'MM' => 'tx_myextension_first_second_mm',  
    'MM_oppositeUsage' => array(  
        'tt_content' => array('somefield'),  
        'tx_myextension_domain_model' => array('some_property'),  
    ),  
)
```



# In-Depth Changes

---

## Miscellaneous

- **Custom record list:**

A custom record list instance can be used in the element browser to override the default element browser record list

- **More subgroups:**

Attribute subgroup in DB table `be_groups` changed from `varchar(250)` to `text`, which allows for much more subgroups (backend users/groups)

- **Extensions TS/Template merged:**

Technically, "WEB > Template" was spread among several extensions (`tstemplate_ceditor`, `tstemplate_info`, `tstemplate_objbrowser` and `tstemplate_analyzer`). Those extensions are now merged into one single extension: "tstemplate"

# In-Depth Changes

---

## Miscellaneous

- **label\_userFunc\_options:**

Support of `label_userFunc_options` added to `BackendUtility`

- **Extension filename:**

When downloading an extension in the Extension Manager, filename contains timestamp (year, month, day and time):

```
<extensionKey>_<version>_<timestamp>.zip  
myextension_1.0.0_201312102359.zip
```

- **EXT:saltedpasswords:**

Extension `EXT:saltedpasswords` is a required system extension and enabled by default now. This forces salted hashes for backend authentication. The Install Tool checks settings and adapts them if required.

# In-Depth Changes

---

## Miscellaneous

- **SignalSlots to modify arguments:**

Arguments passed to SignalSlots dispatcher can be modified now and dispatcher returns the (modified) arguments as it received them in order to keep chaining intact.

- **Workspace preview:**

Query parameters are passed to workspace preview now. This was a problem in TYPO3 < 6.2, where extensions passing custom parameters do not work properly.

- **TCEforms Placeholder feature:**

Introduced in TYPO3 CMS 4.7, the Placeholder features of TCEforms works recursively now (e.g. `__row|uid_foreign|field`).

# In-Depth Changes

---

## Miscellaneous

- **Double-resolution icons:**

SpriteManager supports high resolution icons now: it generates a second sprite with double sized icons (a second file with "@x2.png" suffix). CSS3 ensures, that the high-res file is loaded on devices which support this (this does not affect performance on other devices).

- **Proxy NTLM authentication:**

Support for proxy NTLM authentication (**NT LAN Manager**: a suite of Microsoft security protocols) added. This feature can be activated in the Install Tool:

```
$GLOBALS['TYPO3_CONF_VARS']['SYS']['curlProxyNTLM']
```

*(by the way: this feature was requested more than 8 years ago :-)*

# In-Depth Changes

---

## Miscellaneous

- **cookieHttpOnly by default:**

In order to make the session cookie only accessible through the HTTP protocol, `cookieHttpOnly` is enabled by default now.

This means, cookies "fe\_typo\_user" and "be\_typo\_user" will not be accessible by scripting languages (e.g. JavaScript), which hardens the protection against XSS attacks (*cross site scripting*). Although, some older browsers do not support this technique.

- **Clean-up Database Table:**

Following attributes removed from DB table `tt_content` (not used since TYPO3 4.0): `text_align`, `text_face`, `text_size`, `text_color`, `text_properties`.

# In-Depth Changes

---

## Miscellaneous

- **HTML Tidy removed:**

The *HTML Tidy* functionality has been removed from the TYPO3 core. It can easily be re-instituted by installing EXT:tidy from the TER.

- **dontSetCookie removed:**

Due to the fact that the cookie "fe\_typo\_user" is only set if required (and not always), the Install Tool option `dontSetCookie` became irrelevant and has been removed.

- **"Wizard" scripts removed:**

Removal of the following "wizard" scripts: `typo3/wizard_add.php`, `typo3/wizard_colorpicker.php`, `typo3/wizard_edit.php`, `typo3/wizard_forms.php`, `typo3/wizard_list.php`, `typo3/wizard_rte.php`, `typo3/wizard_table.php`

## Chapter 7: Application Programming Interface (API)

# Application Programming Interface

---

## Hook: `tsfe::checkEnableFields`

- In TYPO3 < 6.2, "extend to subpages" can not be used in own extensions that provide additional rules for page visibility (list of fields to check is hard-coded in `tsfe::checkEnableFields()`)
- In TYPO3 >= 6.2, a new hook allows extensions to provide additional rules for page visibility when parent pages have "extend to subpages" activated.

- Class:

```
\TYPO3\CMS\Frontend\Controller\TypoScriptFrontendController
$GLOBALS['TYPO3_CONF_VARS']['SC_OPTIONS']
    ['tslib/class.tslib_fe.php']['hook_checkEnableFields']
```



# Application Programming Interface

---

## Hook: `checkFlexFormValue` in `DataHandler`

- In TYPO3 < 6.2, when updating Flexform values, there is no check if an existing value in the database has actually been deleted
- This became a problem, e.g. when saving switchable controller actions (Extbase) in the Flexform: old actions that may not be present any longer have to be removed manually
- In TYPO3 >= 6.2, a new hook allows to adjust the old Flexform data right before it is merged with the new one

- Class:

```
\TYPO3\CMS\Core\DataHandling\DataHandler  
$GLOBALS['TYPO3_CONF_VARS']['SC_OPTIONS']  
    ['t3lib/class.t3lib_tcemain.php']['checkFlexFormValue']
```

- Method:

```
checkFlexFormValue_beforeMerge()
```

# Application Programming Interface

---

## Hook to customize header

- In TYPO3 >= 6.2, a new hook allows modifying the header of a page in the page module (Module: "Web > Page")
- This hook is called before the content of the page is rendered

- **Class:**

```
\TYPO3\CMS\Backend\Controller\PageLayoutController
```

```
$GLOBALS['TYPO3_CONF_VARS']['SC_OPTIONS']
```

```
['cms/layout/db_layout.php']['drawHeaderHook']
```

- **Method:**

```
callUserFunction()
```

# Application Programming Interface

---

## IRRE: default values for created records

- New TCA option allows to configure "inline" fields
- Key `foreign_record_defaults` allows to set (default) values in new created records

```
'config' => array(  
    'type' => 'inline',  
    'foreign_table' => 'tt_content',  
    'foreign_record_defaults' => array(  
        'CType' => 'image'  
    ),  
)
```

Example above: `tt_content` elements that are created for this IRRE field will be **image content elements** by default. Editor can set this to another type before saving.

# Application Programming Interface

---

## Workspaces (1)

- In TYPO3 < 6.2, module "Workspaces" can be extended by overriding PHP and JavaScript components only
- In TYPO3 >= 6.2, it is now possible to extend the definition and behaviour of displayed columns in the module
- Some examples on the following slides...

# Application Programming Interface

---

## Workspaces (2)

Example (file `ext_localconf.php`):

```
$GLOBALS['TYPO3_CONF_VARS']['SC_OPTIONS']  
    ['t3lib/class.t3lib_tcemain.php']['processCmdmapClass']['workspaces_logger'] =  
    'Vendor\\WorkspacesLogger\\Hook\\DataHandlerHook';
```

Example (file `ext_tables.php`):

```
\TYPO3\CMS\Workspaces\Service\AdditionalColumnService::getInstance()->register(  
    'WorkspacesLogger_StageChange',  
    'Vendor\\WorkspacesLogger\\DataProvider'  
);  
  
\TYPO3\CMS\Workspaces\Service\AdditionalResourceService::getInstance()->addJavaScriptResource(  
    'WorkspacesLogger',  
    'EXT:myextension/Resources/Public/JavaScript/StageChange.js'  
);
```

# Application Programming Interface

---

## Workspaces (3)

### Example (file Vendor\WorkspacesLogger\Hook\DataHandlerHook):

```
<?php
namespace Vendor\WorkspacesLogger\Hook;
use TYPO3\CMS\Core\SingletonInterface;

class DataHandlerHook implements SingletonInterface {

    const TABLE_Name = 'tx_workspaceslogger_event';
    const EVENT_SetStage = 91;

    /**
     * hook that is called when no prepared command was found
     */
    public function processCmdmap($command, $table, $id, $value, &$commandIsProcessed,
        \TYPO3\CMS\Core\DataHandling\DataHandler $tcemainObj) {
        ...
        $action = (string) $value['action'];
        if ($command === 'version' && $action === 'setStage' && $commandIsProcessed) {
            ...
        }
    }
}
```

# Application Programming Interface

---

## PSR-3 compatible Logger

- The TYPO3 CMS 6.2 logging API is now PSR-3 compatible
- PSR-3 aims to set a standard for logging in PHP (standard of the PHP Framework Interop Group)
- The main goal of PSR-3 is *"to allow libraries to receive a LoggerInterface object and write logs to it in a simple and universal way."*
- Logger interface contains shorthand log methods such as `debug()`, `warning()`, `notice()`, `alert()`, `error()`, etc.
- Further resources:  
<http://www.php-fig.org/psr/3/>

# Application Programming Interface

---

## CSRF Protected Ajax Calls

- Ajax calls in the TYPO3 backend can be protected against CSRF (*cross-site request forgery*) by registering their handlers

```
\TYPO3\CMS\Core\Utility\ExtensionManagementUtility::registerAjaxHandler(  
    'TxMyExt::process',  
    '\Vendor\MyExt\AjaxHandler->process'  
);
```

- URL for a given Ajax ID contains a CSRF protection token, which will be checked in the `ajax.php` dispatcher

```
$ajaxUrl = \TYPO3\CMS\Core\Utility\BackendUtility::getAjaxUrl('TxMyExt::process');
```

- These settings can then be accessed in the JavaScript context of the page

```
var ajaxUrl = TYPO3.settings.MyExt.ajaxUrl;
```



# Application Programming Interface

---

## Miscellaneous

- New method `canBeInterpretedAsFloat()` in class: `MathUtility`  
(This is an analogue of: `canBeInterpretedAsInteger()`)
- New enumeration type (without a relation to 3rd party PHP modules):  
`\TYPO3\CMS\Core\Type\Enumeration`

For example used in:

```
\TYPO3\CMS\Core\Versioning\VersionState
```

...and then as:

```
new VersionState(VersionState::DEFAULT_STATE);
```

## Chapter 8: Extbase & Fluid

# Extbase & Fluid

---

## ObjectManager->getScope()

- Method ObjectManager->getScope() determines, if a class is of type **prototype** or **singleton**

```
/**
 * @var \TYPO3\CMS\Extbase\Object\ObjectManagerInterface
 * @inject
 */
protected $objectManager;

$this->objectManager->getScope($propertyTargetClassName) === \TYPO3\CMS
\Extbase\Object\Container\Container::SCOPE_PROTOTYPE

$this->objectManager->getScope($propertyTargetClassName) === \TYPO3\CMS
\Extbase\Object\Container\Container::SCOPE_SINGLETON
```

# Extbase & Fluid

---

## Page Type For URIs

- Custom page type attribute is not longer required in links, when rendering a special format

### TYPO3 < 6.2:

```
<f:link.action arguments="{blog: blog}" pageType="{settings.plaintextPageType}"
  format="txt">[plaintext]</f:link.action></li>
```

- New TypoScript option `formatToPageTypeMapping` allows for a global mapping:

```
plugin.tx_myextension {
    view.formatToPageTypeMapping {
        txt = 99
        pdf = 123
    }
}
```

### TYPO3 >= 6.2:

```
<f:link.action arguments="{blog: blog}"
  format="txt">[plaintext]</f:link.action></li>
```

# Extbase & Fluid

---

## Object Type Converter (1)

- Maps array sources to non-persistent objects
- Useful if you need transitional objects built from request arguments
- Some examples on the following slides...

# Extbase & Fluid

---

## Object Type Converter (2)

### GET request

```
http://example.com/index.php?id=299
&tx_myextension[action]=list
&tx_myextension[controller]=Entity
&tx_myextension[demand][title]=foo
&tx_myextension[demand][relation]=1
```

### Entity controller: initializeListAction()

```
use [Vendor]\myextension\Domain\Dto\Demand;
public function initializeListAction() {
    /**
     * @var PropertyMappingConfiguration $demandConfiguration
     */
    $demandConfiguration = $this->arguments['demand']->getPropertyMappingConfiguration();
    $demandConfiguration->allowAllProperties()->forProperty('relation')->allowAllProperties()->
        setTypeConverterOption(
            'TYPO3\CMS\Extbase\Property\TypeConverter\PersistentObjectConverter',
            PersistentObjectConverter::CONFIGURATION_CREATION_ALLOWED,
            TRUE
        );
}
```

# Extbase & Fluid

---

## Object Type Converter (3)

### Entity controller: listAction()

```
use [Vendor]\myextension\Domain\Dto\Demand;
/**
 * @var PropertyMappingConfiguration $demandConfiguration
 */
public function listAction(Demand $demand = NULL) {
    $entities = $this->entityRepository->findAll();
    $this->view->assign('entities', $entities);
}
```

### Model: [Vendor]\myextension\Domain\Dto\Demand.php

```
namespace [Vendor]\myextension\Domain\Dto;
use [Vendor]\myextension\Domain\Model\Relation;
class Demand {
    protected $relation;
    /**
     * @param \TYPO3Friends\MapperExample\Domain\Model\Relation $relation
     */
    public function setRelation($relation) {
        $this->relation = $relation;
    }
}
```

## Chaining Of set\* Functions

- set\* handling methods can now be *chained* within QuerySettings API
- Includes new options introduced with TYPO3 CMS 6.0:  
setIncludeDeleted and setIgnoreEnableFields

```
$query->getQuerySettings()  
->setRespectStoragePage(FALSE)  
->setRespectSysLanguage(FALSE)  
->setIgnoreEnableFields(TRUE)  
->setIncludeDeleted(TRUE);
```



# Extbase & Fluid

---

## returnRawQueryResult As Argument

- Raw query result not longer as a central method, but as an argument in method: `execute()`

### **TYPO3 < 6.2:**

```
$query->getQuerySettings()->setReturnRawQueryResult(TRUE);
```

### **TYPO3 >= 6.2:**

```
$query->execute(TRUE);
```

# Extbase & Fluid

---

## Recursive Validation

- Extbase uses recursive validation now (as known from TYPO3 Flow)
- This means, when nested objects are created by the Property-Mapper, objects inside a property, as well as the outer object are validated (in TYPO3 CMS < 6.2, only the outer object has been validated)
- Additionally, validators allow empty values now

**THIS IS A BREAKING CHANGE!**

In order to make a property required, you have to add the **NotEmptyValidator** explicitly!

# Extbase & Fluid

---

## Application Context

- Access current Application Context in Extbase  
(set as environment variable TYPO3\_CONTEXT or in Install Tool)

```
\TYPO3\CMS\Core\Core\Bootstrap::getInstance()->getContext();
```

```
\TYPO3\CMS\Core\Utility\GeneralUtility::getContext();
```

# Extbase & Fluid

---

## ViewHelper: image

- Fluid ViewHelper **image** with optional `title` attribute

### Example:

```
<f:image src="background.jpg" alt="Text" />
```

### TYPO3 < 6.2:

```

```

### TYPO3 >= 6.2:

```

```

# Extbase & Fluid

---

## ViewHelpers: textfield and textarea

- Arguments autofocus and placeholder (valid HTML5 argument) for Fluid ViewHelpers **form.textarea** and **form.textfield**

### Example ("placeholder"):

```
<f:form.textfield
  id="powermail_field_{field.marker}"
  ...
  placeholder="{field.title -> vh:string.RawAndRemoveXss()}"
  ...
  name="field[{field.uid}]"
  required="{field.mandatory}" />
```

# Extbase & Fluid

---

## ViewHelper: switch

- New Fluid ViewHelper **switch** renders content depending on a given value or expression
- Behaves similar to the `switch()` statement in PHP

```
<f:switch expression="{person.gender}">
  <f:case value="male">Mr.</f:case>
  <f:case value="female">Mrs.</f:case>
  <f:case default="TRUE">Mrs. or Mr.</f:case>
</f:switch>
```

- **Note:** excessive usage of this ViewHelper is an indicator of a bad design! Example above could also be achieved by using the partials "title.male.html" and "title.female.html" and the following:

```
<f:render partial="title.{person.gender}" />
```

# Extbase & Fluid

---

## ViewHelper: fileSize

- Converts a file size (integer) to a human readable string

**Example 1** (fileSize = 1263616):

```
fileSize -> f:format.bytes()
```

Output: "1234 KB"

**Example 2** (fileSize = 1263616):

```
fileSize -> f:format.bytes(  
    decimals: 2,  
    decimalSeparator: '.',  
    thousandsSeparator: ',',  
)
```

Output: "1,234.00 KB"

# Extbase & Fluid

---

## ViewHelper: `format.date`

- Default value of ViewHelper **format.date** is the value configured in the Install Tool  
`$GLOBALS['TYPO3_CONF_VARS']['SYS']['ddmmyy']`
- If this value is not set, "Y-m-d" is used (year, month, day)



# Extbase & Fluid

---

## ViewHelper: Backend Container

- Fluid ViewHelper backend container (`be.container`) reworked:  
`typo3/sysext/fluid/Classes/ViewHelpers/Be/ContainerViewHelper.php`

### Deprecated:

- `$addCssFile` (use `$includeCssFiles` instead)
- `$addJsFile` (use `$includeJsFiles` instead)

### New:

- `$loadJQuery`
- `$includeCssFiles`
- `$includeJsFiles`
- `$addJsInlineLabels`

# Extbase & Fluid

---

## ViewHelper: button.icon

- Fluid ViewHelper **button.icon** finalised (was "experimental")
- Creates a button icon (optionally with a link)

```
<f:be.buttons.icon uri="{f:uri.action(action:'new')}"  
    icon="actions-document-new" title="Create new Foo" />
```

```
<f:be.buttons.icon  
    icon="actions-document-new" title="Create new Foo" />
```

- Attribute `icon` accepts more than 310 values!

Search for:

```
$GLOBALS['TBE_STYLES']['spriteIconApi']['coreSpriteImageNames']
```

...in file:

```
typo3/systext/core/ext_tables.php
```

# Extbase & Fluid

---

## Option `addQueryStringMethod`

- Option `addQueryString` supports **GET**-arguments only (which are then added to the generated link)
- **POST**-arguments (used by Widgets) do not work with this option
- New option `addQueryStringMethod` addresses this issue and allows to define, which methods should be taken into account: GET (default), POST, GET/POST or POST/GET
- Several Fluid ViewHelpers support this new option:
  - `link.action`
  - `link.page`
  - `uri.action`
  - `uri.page`
  - `widget.link`
  - `widget.uri`
  - `widget.paginate`

# Extbase & Fluid

---

## Fluid: Fallback Path For Templates

- Fluid supports "fallback" paths for templates, partials and layouts now: `templateRootPaths`, `partialRootPaths`, `layoutRootPaths`
- Highest index first, then iterate through lower indexes, until template is found

```
plugin.tx_myextension {
    view {
        templateRootPath = EXT:myextension/Resources/Private/Templates/
    }
}
```

```
plugin.tx_myextension {
    view {
        templateRootPath >
        templateRootPaths {
            10 = fileadmin/myextension/Templates/
            20 = EXT:myextension/Resources/Private/Templates/
        }
    }
}
```

## Chapter 9: Upgrade to TYPO3 CMS 6.2 LTS

# Upgrade to TYPO3 CMS 6.2 LTS

---

## General Upgrade Instructions

- Upgrade instructions:

  - [http://wiki.typo3.org/Upgrade#Upgrading\\_to\\_6.2](http://wiki.typo3.org/Upgrade#Upgrading_to_6.2)

- Official TYPO3 guide "TYPO3 Installation and Upgrading":

  - <http://docs.typo3.org/typo3cms/InstallationGuide>

- General approach:

  - Check if system meets minimum requirements (PHP, MySQL, etc.)
  - Review **deprecation\_\*.log** in old TYPO3 instance
  - Update all extensions to the latest version (check TYPO3 6.2 compatibility)
  - See chapter "Install Tool" in this presentation

# Upgrade to TYPO3 CMS 6.2 LTS

---

## Upgrade from TYPO3 CMS 4.5 LTS

- Many TYPO3 sites will go from LTS to the next LTS version
- Smooth Migration project:
  - Aims to make a migration from 4.5 to 6.2 as smooth as possible
  - Documentation, identification of issues in extensions, etc.
  - <http://forge.typo3.org/projects/typo3cms-smoothmigration>
- EXT:typo3-upgradereport:
  - Mainly developed by Steffen Ritter
  - Install in a TYPO3 CMS 4.5 LTS instance and run the tests
  - Participate in the development
  - <https://github.com/nxpthx/typo3-upgradereport>

# Upgrade to TYPO3 CMS 6.2 LTS

---

## What's New for Editors

- Summarises the main changes between TYPO3 CMS 4.5 and 6.2
- Target audience: predominantly editors (non or less technical users)
- Also aims to help agencies:
  - in preparation of responding to support requests
  - conducting workshops, seminars, trainings, etc.
- Download the document "**What's New for Editors**":  
<http://typo3.org/download/release-notes/whats-new>



## Chapter 10: TYPO3 CMS 6.2 LTS – MythBuster

# MythBuster

---

## Myths About TYPO3 CMS 6.2

- TYPO3 CMS 6.2 LTS will be the last TYPO3 CMS release → **not true!**  
Truth is, that despite the release of [TYPO3 Neos](#), the development of TYPO3 CMS will continue and we will see further releases in the future.
- The TYPO3 core was completely rewritten in 6.x → **not true!**  
Truth is, that we introduced the concept of PHP namespaces with TYPO3 CMS 6.0, which results in new class names. However, a compatibility layer ensures, developers can still use the old class names in their extensions.
- Extensions developed for 4.5 will not work on 6.2 → **not true!**  
Truth is, that the core API has not changed completely and features backwards compatibility, if in accordance with our [deprecation strategy](#). The core of TYPO3 CMS 6.2 still supports most extensions that were written for 4.5 with no or little modifications.

## Myths About TYPO3 CMS 6.2

- **TemplaVoila cannot be used in TYPO3 6.2 anymore** → **not true!**  
Truth is, the community is working on a compatible version, which will enable you to use TemplaVoila in TYPO3 CMS 6.2. However, TemplaVoila will not be developed further and integrators are encouraged to investigate alternatives for future projects.
- **tslib\_pibase-based extensions do not work** → **not true!**  
Truth is, class tslib\_pibase still exists in 6.2, but has a new name due to namespace conventions: `\TYPO3\CMS\Frontend\Plugin\AbstractPlugin`.  
A class alias ensures, the old name still works (compatibility layer).
- **There is no way to migrate DAM records to 6.2 with FAL** → **not true!**  
Fact is, DAM does not work with TYPO3 6.x. However, FAL is meant to provide an API that makes it possible to recreate whatever was possible with DAM. There is also a [DAM-to-FAL-migration extension](#) available.

# MythBuster

---

## Myths About TYPO3 CMS 6.2

- You can upgrade 4.5 to 6.2 with an upgrade wizard → **not true!**  
Rumors say, that the "Smooth Migration" project provides a big upgrade wizard which automatically upgrades TYPO3 4.5 to 6.2. Truth is, that the project aims to provide information, documentation, detect incompatibilities, etc. to support integrators in the migration process.
- TYPO3 6.2 requires much better hardware → **not true!**  
Rumors say, that 6.2 is 10 times slower than 4.5. Truth is, that in most cases the performance is similar to previous versions. The [minimum requirements](#) for running TYPO3 have not changed. However, due to the nature of the architectural changes and new modern technologies, system administrators should consider to hardware upgrade (keep in mind: TYPO3 4.5 was released in January 2011, almost 3 years ago).

## Chapter 11: Sources and Authors

# Sources and Authors

---

## Sources (1)

### TYPO3 News:

- <http://typo3.org/news>

### Release Notes:

- [http://wiki.typo3.org/TYPO3\\_6.2](http://wiki.typo3.org/TYPO3_6.2)
- <http://typo3.org/download/release-notes/typo3-6-2-release-notes/>
- NEWS.txt and ChangeLog

### TYPO3 Git Repositories:

- <https://git.typo3.org/TYPO3v4/Core.git>
- <https://git.typo3.org/TYPO3v4/CoreProjects/MVC/extbase.git>
- <https://git.typo3.org/TYPO3v4/CoreProjects/MVC/fluid.git>

# Sources and Authors

---

## Sources (2)

### TYPO3 Bug-/Issuetracker:

- <http://forge.typo3.org/projects/typo3v4-core/issues>

### Other Resources:

- Responsive Image Community Group  
<http://responsiveimages.org>
- Package Manager (Blueprint)  
<http://wiki.typo3.org/Blueprints/Packagemanager>
- Normalize.css  
<http://nicolas.github.io/normalize.css/>
- PHP Logging standard PSR-3  
<http://www.php-fig.org/psr/psr-3/>
- "LTS Smooth Migration" project  
<http://forge.typo3.org/projects/typo3cms-smoothmigration>
- TYPO3 CMS 4.5 to 6.2 upgrade report  
<https://github.com/nxpthx/typo3-upgradereport>

# Sources and Authors

---

## **TYPO3 CMS What's New Slides:**

Patrick Lobacher (Research and Information Gathering)

Michael Schams (English/German version and Project Leader)

## **Translations by:**

Andrey Aksenov, Paul Blondiaux, Sergio Catala,

Philippe Hérault, Sinisa Mitrovic, Michel Mix,

Roberto Torresani, Ric van Westhreenen, Christiaan Wiesenekker

<http://typo3.org/download/release-notes/whats-new>

Licensed under Creative Commons BY-NC-SA 3.0

