

Project Description: FORM



FORM

Project Leader:
Patrick Broens

Overseen by:
François Suter
TYPO3 v4 Sponsorship Manager

Table of Contents

FORM.....	1
Table of Contents.....	2
Summary.....	3
Goals.....	3
Future goals.....	3
Background.....	3
Teams.....	5
Project Team.....	5
Review Team.....	5
Description.....	6
History.....	6
Form content object.....	6
Form Wizard.....	6
Technical specifications.....	9
Milestones.....	10
Milestone 1 – Ground work, MVC and Form objects.....	10
Milestone 2 – Filtering and validating incoming data.....	10
Milestone 3 – Handling of incoming data.....	10
Milestone 4 – Form wizard GUI.....	10
Milestone 5 – Generating Typoscript from wizard.....	11
Milestone 6 – Fine tuning, additions, tester's feedback.....	11
Workload and timeline.....	12

Project Description: FORM

Summary

The project consists of two parts; rewriting and enhancing the Form cObj, which makes it possible to generate the forms, and secondly the Form Wizard, which makes it easy for editors to construct forms.

This project will add a new extension to TYPO3 which will take a lot of functionality from various form extensions, like server side validation, build in captcha functionality, and more validation methods which can be extended.

The Form Wizard for editors will be very intuitive and uses drag & drop and sorting methods to add and replace form elements. While working in the wizard the editor will exactly see how the form will be displayed in the frontend, the website.

Goals

- Rewriting the Form cObj to make it Object Oriented based. This will make it easier for future enhancements (see future goals). Also community extensions can profit from this library as well.
- Output must be right for accessible websites.
- Various validation methods of the submitted values, with choice to do validation server or client side. User defined methods can be added. Possibility to add multiple validation rules to one field or have validation rules which spans more than one element.
- Possibility to group elements in the form with a fieldset. Nested fieldsets will also be possible.
- (X)HTML layout completely based on CSS.
- Intuitive drag and drop system with inline editing for editors in the backend to construct the form. The form will be displayed exactly the way it will be displayed in the frontend.
- Separation of markup and code. This will make it possible to render the forms as XFORMS or PDF Forms using TYPO3 services.
- Cover all possible form elements, attributes and behaviors and add special form fields like date or time fields.
- Configurable confirmation screen after submit.
- Handling of file uploads
- Built-in captcha method
- Insert regular TYPO3 content elements in the form.

Future goals

More features will be added after the release of TYPO3 4.3. Future goals are:

- Add more possibilities for handling the submitted data instead of only emailing, like storing data in the database.
- Multipage forms
- Redirecting mails to different addresses depending on submitted data
- AJAX validation. This makes it possible to show an error immediately after filling a single field without submitting the whole form.

Background

The FORM Content Object is around for some time. Although TYPO3 has evolved to a Content Management System on Enterprise level, the FORM Content Object stayed behind. Some improvements were made in the previous years, but it still lacks functionality, which is reflected by the numerous Form extensions which are available for TYPO3 which all have their advantages and disadvantages.

The project consists of two parts; rewriting and enhancing the Form cObj, which makes it possible to generate the forms, and secondly the Form Wizard, which makes it easy for editors to construct forms.

Both are currently available, but lack a lot of functionality. Complex forms are not possible right now and it's not possible to store data. Validation must be enhanced a lot, first because it is only done client side and can be manipulated, which makes validation worthless, secondly there are only some simple validation rules for single fields.

Project Description: FORM

The Form Wizard for editors will be much more intuitive and uses drag & drop and sorting methods to add and replace form elements. While working in the wizard the editor will exactly see how the form will be displayed in the frontend, the website.

Project Description: FORM

Teams

Project Team

Project Leader: Patrick Broens (Core Team Member)

Email: patrick@patrickbroens.nl

Backup Project Leader: (to be filled)

Email: xxx@xxx.com

François Suter

Email: francois@typo3.org

Review Team

Ernesto Baschny (Core Team Member)

Email: ernst@cron-it.de

Peter Foerger

Email: pfoerger@googlemail.com

Description

History

This project started with the aim at only the Form Wizard, which became pretty old and was not very intuitive for editors. Pretty soon it became clear that not only the current Form Wizard was outdated and not easy to use, but also the Form content object, which translates the form Typoscript or configuration made by the Form wizard into a displayable form, needed to get revamped as well. The Form wizard was set aside for a while and the main focus was on rewriting the form cObj first. This work has already started. Also a mockup for the Form Wizard has been written.

In the past there were some attempts to rewrite the Form cObj, but most of the times it stopped because it was pretty hard to make it backwards compatible with the old Form cObj and wizard. That's why we decided to make a separate extension first. This makes it possible to leave the old form configuration as it is, when this is necessary. Especially for new websites it will be useful to install the new extension. The extension will take over the Typoscript Form configuration and users can start writing Typoscript with the new features.

Form content object

New Typoscript

The first goal is to write a completely new Typoscript configuration which makes it possible to add all the new features. The new Form Typoscript will be more like the regular Typoscript and will add form content objects, like input fields, fieldsets, checkboxes, radiobuttons and option groups. This can be used in combination with regular content objects like text, text with image etc. Validation and filter rules will be added to the new form Typoscript, as well as methods to store data, a configurable confirmation screen, captcha methods and handling of file uploads. In the future this can be enhanced with multiple page forms.

Form Library

Secondly the Form library will be written, together with the unit tests it needs. The library will be object oriented based and written with the Model-View-Controller (MVC) in mind. Writing the library using MVC makes it possible to separate the data handling and the rendering of the form. A big advantage is that the form will not be bound to (X)HTML output only, but using a service makes it possible to render the form as PDF or, in the future, as XFORM, which is a much more enhanced way of displaying forms and handling data. The model of the library will parse the Typoscript of the form into separate objects. First you have the form object, which contains the form fields objects. All objects have their own configuration, embedded in these objects. The model also controls the validation and filter rules.

Frontend output

Writing the view will be the third phase of this project. The view will be written as a service, so it can be changed with a different one. The main focus of the view service will be (X)HTML output. This output is mainly used in TYPO3. The view controls how data will be rendered as (X)HTML and outputted to the frontend, the website. It controls how the markup of the form will be, like the form fieldsets and fields, the error messages and confirmation screen. This (X)HTML markup can be styled with CSS, so there are almost no boundaries how to display the form.

At this stage it's possible to render a form using the new Form extension. But this will only be possible using Typoscript, which is only useful for experienced backend users.

Form Wizard

For regular editors a new Form Wizard will be made. This wizard will have a very easy to use interface, using drag and drop methods to add or move form elements like input fields. Editing the form will be as easy as using a text editor.

Project Description: FORM

Predefined forms

Already predefined forms can be chosen from a selector or the editor can start with a blank form.

Drag & drop

From a menu an editor can select a form element and drag it into the form on the place where the editor wants it to be.

Editing made easy

By clicking on a form element it will show a delete, a move, and a edit button.

By clicking on the delete button, the element will be removed from the form.

Clicking and dragging the move button will move the form element to a different position in the form.

The edit button is meant to show the configuration of the selected element in a overlaying box.

In this box you can edit the label, validation and filter rules and styling of the element. This configuration box will be adapt itself for the different form elements. The label can also be edited by clicking the label in the form itself. By clicking this label it will change into an editable field, so you can change the text inline.

WYSIWYG

A main feature of the wizard is that it can be styled exactly the way it will be shown in the frontend, so the editor already sees how it will be presented to the website visitors. This can be done by adding a CSS stylesheet to the form wizard by a backend administrator. This is a feature which is also available in the text editor of TYPO3, rteHtmlArea.

Storage method

The configuration of the form, made with the wizard will also be stored as Typoscript, but should not be visible to a regular editor. It will however be configurable to show this configuration to a more advanced backend user, so he/she can change this configuration as needed. Not all available Typoscript configuration will be available in the Form Wizard, but it will be possible to add most of the needed elements and features.

Mockup

A mockup of the Form Wizard, which is not styled at all, is available at http://www.patrickbroens.nl/fileadmin/user_upload/formwizard/wizard_forms.html. This will show how the drag and drop, the sorting of elements and inline editing will work. Please keep in mind that this is only a very premature mockup version of the wizard.

Project Description: FORM

Untitled Form

Put the elements here

Personal information

First name

Last name

Gender

Male Female ✖

Date of birth (dd-mm-yyyy)

Address

Form elements

Textbox

Textarea

Dropdown

Fieldset

Checkbox

Radiobutton

File upload

Password

Reset button

Submit button

Element properties

Label This is the new one

Required No

Max. size 20

Validation No

Mockup of the Form Wizard in a very early stage

Technical specifications

The Form Wizard will use the javascript framework combination Prototype and Scriptaculous, which are external frameworks but supported by the TYPO3 Core.

Project Description: FORM

Milestones

Milestone 1 – Ground work, MVC and Form objects

The ground work consists of building the first objects using the architectural pattern MVC, which isolates business logic from user interface considerations.

This milestone will have a very limited feature set, like Typoscript handling and displaying a very basic form.

The Form cObj will make use of Typoscript intensively, and therefore the incoming Typoscript needs to be separated into chunks of variables which will be assigned to the various form element objects in the Model.

Secondly this data needs to be presented in the frontend, which will be handled by the View.

Milestone 2 – Filtering and validating incoming data

The purpose of this milestone is to take the incoming data and compare this incoming data with rulesets. Two rulesets will be used:

Filtering

Some of the incoming data needs to be filtered, for instance for security reasons. Other possibilities are that a submitted value needs to be converted to upper or lower case or a floating point. Basic rules will be added including a hook to add your own filtering rules.

Validation

In order to get the requested information, submitted values need to be validated. A field could be required, needs a minimum or maximum length of characters or needs to be compared with the value of another field. Single field as well as multiple field validation will be added at this point, including a hook to add own validation rules. Validation will be done server side.

Milestone 3 – Handling of incoming data

The main purpose of a form is to gather information from the visitors, store it or send it by email. In this milestone we will add the handling of incoming data.

Confirmation

After submitting the form the user can be presented a confirmation screen, with all the submitted data. The user can check this data and, if he/she made an error, go back to the form to change the data.

Email

Mail the submitted data to one or multiple addresses.

After this milestone it is possible to build a working form using Typoscript only.

Milestone 4 – Form wizard GUI

At this point the Form Wizard will be added to the form extension. Features in detail:

Drag & drop, sorting methods

Needed form elements can be selected in a menu and dragged into the form to the position where it is required. Elements which are already in the form can be dragged to a different location if needed.

Project Description: FORM

In-place-editing

Labels and legend values can be changed by clicking the text and edit it directly

Element configuration

By clicking an element some buttons will show for this particular element, like move, delete or edit. By clicking the edit button an overlaying box will show the configuration of this element, which a user can edit. Here you can change for instance the value(s) of the element, validation or filter rules etc.

Milestone 5 – Generating Typoscript from wizard

The Form wizard will generate Typoscript after saving the form. This will be partly done by Javascript which serializes the form DOM to a XML string including all extra properties. This XML string will be send to a php method which will transform the XML to Typoscript and store it in the database

Milestone 6 – Fine tuning, additions, tester's feedback

This milestone will be used mostly for testing and feedback from early bird users.

Workload and timeline

Milestone	Workload
1 Ground work, MVC and Form objects	4 MD
2 Filtering and validating incoming data	4 MD
3 Handling of incoming data	2 MD
4 Form wizard GUI	8 MD
5 Generating TypoScript from wizard	5 MD
6 Fine tuning, additions, tester's feedback	2 MD
Total development time	25 MD
Review process	2.5 MD
Daily rate	400 Euro
Reviewer's meeting expenses	400 Euro
Total amount:	11,400 Euro