Alkacon Software GmbH

An der Wachsfabrik 13
DE - 50996 Köln (Cologne)

Geschäftsführer / CEO
Alexander Kandzior

Amtsgericht Köln
HRB 54613

Tel: +49 (0)2236 3826 - 0
Fax: +49 (0)2236 3826 - 20

http://www.alkacon.com
http://www.opencms.org

Alkacon Software GmbH

Technote

Alkacon OAMP Webform Module

Version: 1.1

Date: Friday, January 4, 2008

1 Table of Content

2 Abstract

This document describes the installation, configuration and usage of the Alkacon OpenCms Add-On Module Package Webform. With the Webform module, it is possible to create highly configurable online input forms without knowledge of HTML.

Once created, configured and published, a Webform may be filled out by website visitors. The data entered may be sent to an email account and / or stored in the OpenCms database.

3 General purpose of the Alkacon OAMP Webform Module

The module extends a basic OpenCms installation with the capability to create highly configurable online input forms. It provides the following features:

- Create Webforms by mouse clicks and simple keyboard input. A Webform is a structured XML content which offers a comfortable user interface.
- Complete configuration of a Webform is done in one file.
- Input fields, their labels, default values and options are freely configurable.
- For each input field it is possible to pick from 9 different field types: Text input, Text area, Checkbox, Privacy Checkbox [a checkbox with a popup link to a privacy statement], Radio buttons, Hidden fields, File upload fields, Email Field and Empty field.
- For each input field it is possible to define a default value, control if it is mandatory, define a regular expression for validation and an error message to be shown in case validation was not successful.
- To avoid that spiders use your Webforms (e.g. in order to spread spam to forums or guestbooks) a CAPTCHA field may be configured. This is an image containing distorted text that has to be entered in a text input to verify that a human is filling out the form.
- The collected data of a submission may be sent to a configurable email account and / or stored in the OpenCms database in separate exclusive tables.

4 Installation

Note: To use the Alkacon OAMP Webform module, you need at least OpenCms version 7.0.3. The module is not compatible with older OpenCms versions.

Note: The OpenCms module `org.opencms.frontend.templateone.form` which is the predecessor and a "lighter" version of this module (no support for database persistence) should be deinstalled before installation of this module. Old email forms that have been created with this module might still work except for the captcha image verification. The used captcha library JCaptcha of this module is a newer version and it is required that the older captcha libraries are deleted from the OpenCms web application classpath.

Step by step installation procedure:

1. Go to the OpenCms Administration view
2. Click "Module Management" and select either "Import Module from Server" if the module

was placed in the **`WEB-INF/packages/modules/`** folder of your OpenCms installation, or select "Import Module with HTTP" to upload the module from your local file system

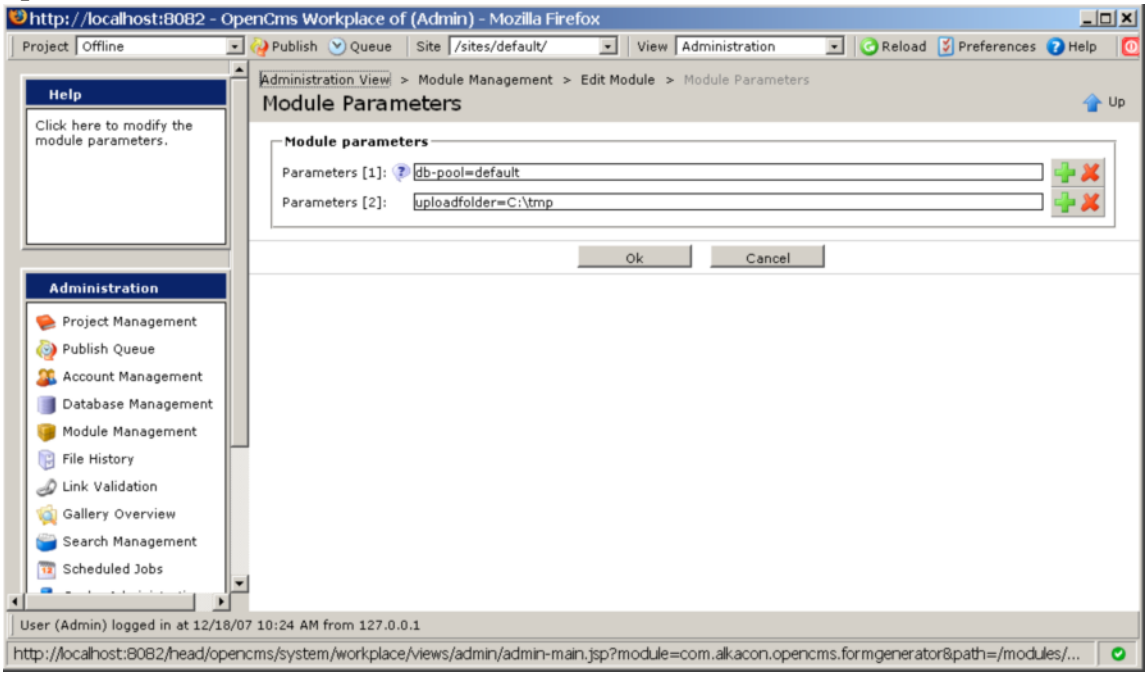
3. Select the Alkacon OAMP Webform module zip file **`com.alkacon.opencms.formgenerator_1.0.x.zip`** to import
4. Check if the jar file **`com.alkacon.opencms.formgenerator.jar`** has been deployed in the **`WEB-INF/lib/`** folder after installation
5. In case you find the libraries
   `ehcache-1.0.jar`
   `jcaptcha-all-1.0-RC2.0.1.jar`
   in the **`WEB-INF/lib/`** folder please delete them.
6. Restart your servlet container afterwards

4.1 Configuration

Configuration is necessary to define the database in which the Webform submissions will optionally be stored in. Furthermore the location in the real file system has to be configured where file uploads should be stored in case a Webform is set up that stores it's data in the database and has a file upload field.

4.1.1 Database

1. Log in to the OpenCms Workplace.
2. Switch to the Administration View.
3. Go to the Module Management page.
4. In the list of the modules click on the module name of the module **`com.alkacon.opencms.formgenerator`**.
5. Click on Module Parameters. Adapt the settings for the module parameter **`db-pool`** and **`uploadfolder`**.



The **db-pool** parameter value references a database pool configured in **<tomcat-home>/webapps/<webappname>/WEB-INF/config/opencms.properties**. The value "**default**" should be OK for most applications. If you need to collect your Webform data in a dedicated database you may configure another pool in the **opencms.properties** and refer to it in this module parameter.
The parameter **uploadfolder** is needed whenever you set up a Webform that has to store submissions in the database and contains a file upload field. These uploads are not stored in the database but in a folder that has to be set here. Please select a valid directory and check the log file when setting up and testing such a form.

5 Module usage

After successful installation and configuration of the Webform module, it is ready to use. Webforms can be set up by creating a new resource of the type "**Alkacon Webform**".

5.1 Creating an Alkacon OpenCms Webform

To create a new Webform the "new" dialog in OpenCms has to be used:

1. Click "New" in the Top Bar of the OpenCms Workplace.
2. In the following dialog click "Structured content" and then "Continue".
3. Then select "Alkacon Webform" and "Continue".
4. In the following dialog enter the name of the new file and more properties if desired.

→ A new file of type "alkacon-webform" has been created and is visible in the OpenCms Explorer.

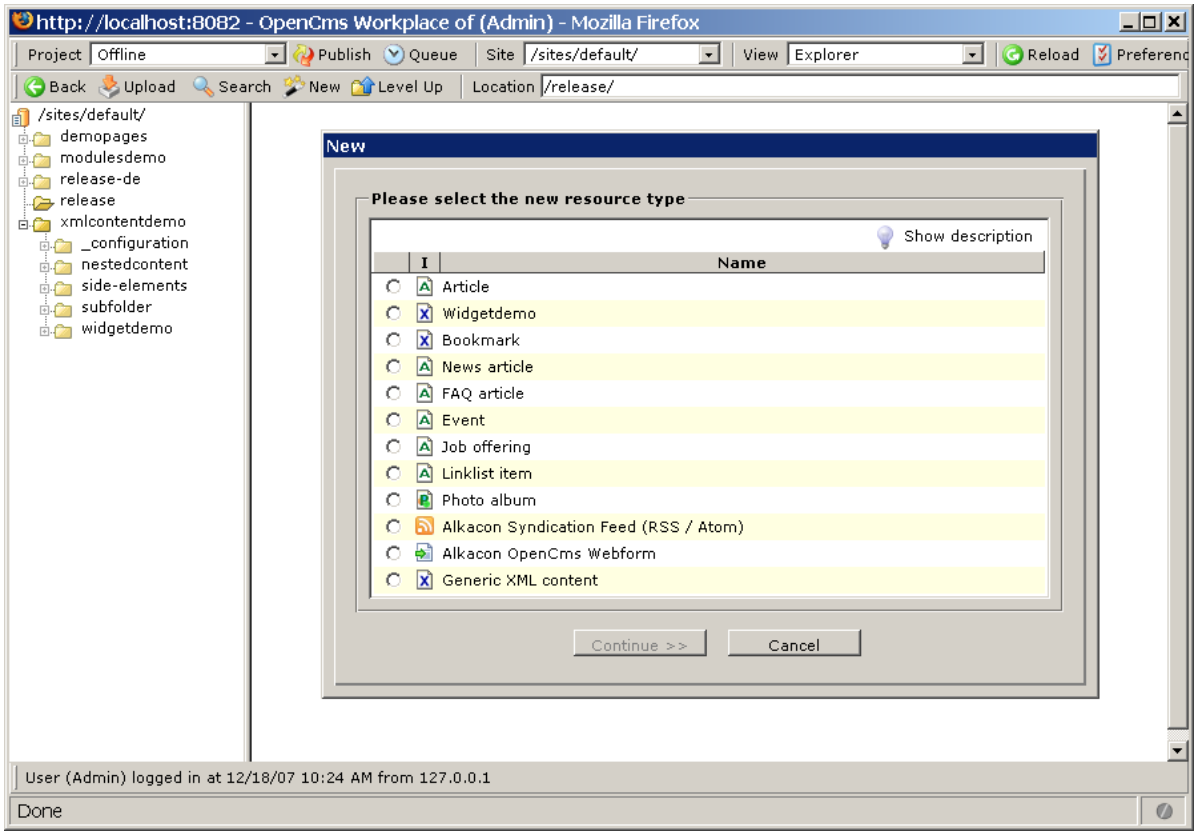


Figure 1: The 2nd stage of the new dialog to create an Alkacon OpenCms Webform.

5.2 Editing an Alkacon OpenCms Webform

Webforms are edited by clicking their file symbol in the OpenCms explorer with the right mouse button and choose "edit" from the popup – menu.

The XML content editor for Webforms appears. One can add as many labelled fields as desired to collect data.

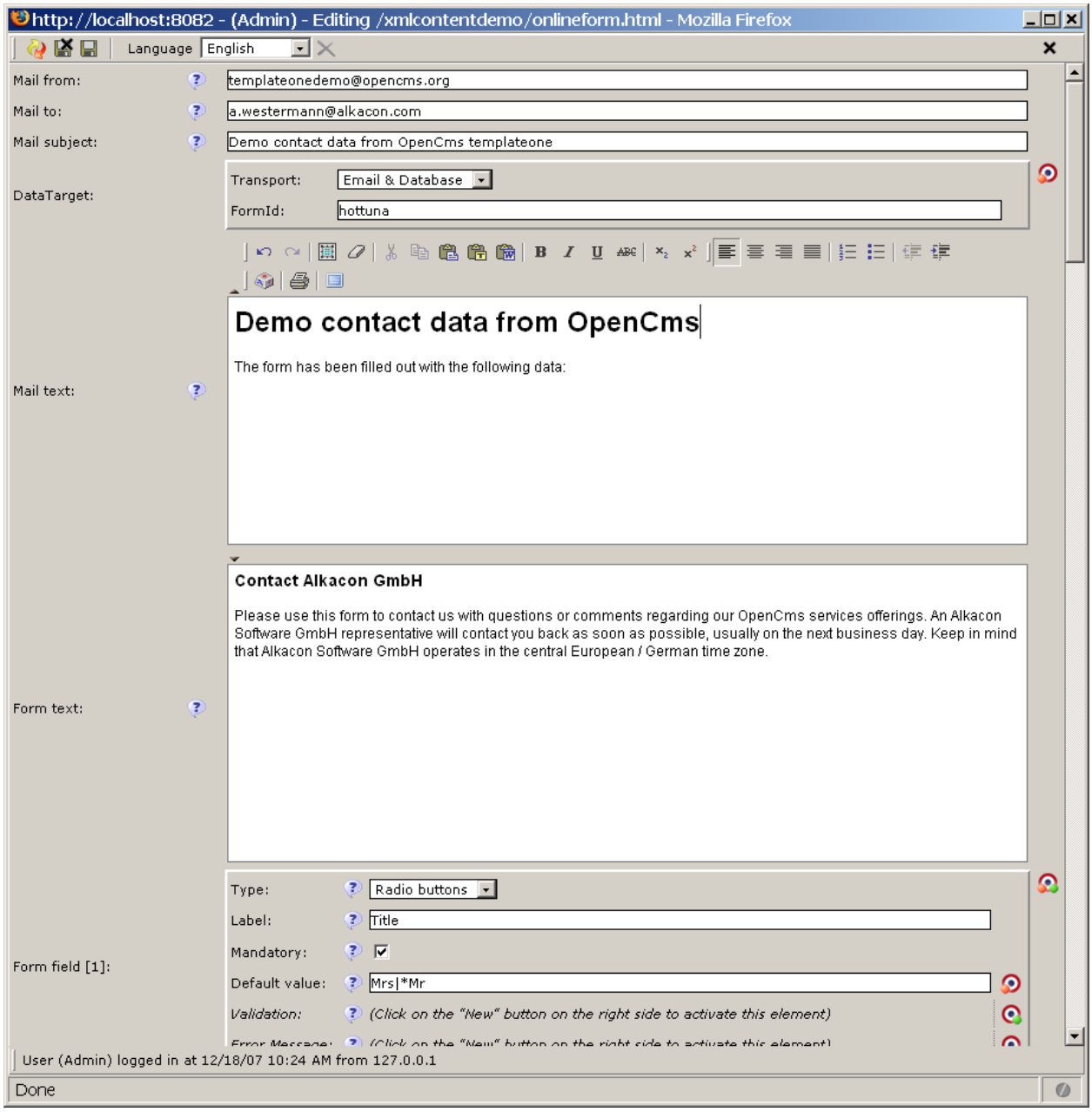


Figure 2: Alkacon OpenCms Webform editor.

There are **some mandatory** fields that have to be configured to make the resulting Webform work:

- **Mail from**: This has to be a valid email address that should be accepted from the mail server you use.[1]
- **Mail to**: This has to be a valid email address the collected data will be sent to.
- **Mail subject**: The Subject of the Email if email transportation is configured.
- **Form text**: This is the text that will appear on the web page showing the Email form.

The following fields are optional:

[1] The mail server that is used is configured in the OpenCms configuration (file: `opencms-system.xml`). If sending of emails fails please contact your OpenCms System administrator to check the configuration.

- **DataTarget:** This nested box allows configuring how the data is processed. By default (if this box is not added with the plus button) the submitted data will be sent as an email to the configured "Mail to" – address.
    - **Transport**: Controls, which mode of transportation is used. In case "Email & Database" or "Database" is used the 2nd field "FormId" is required for database persistence.
    - **FormId**: In case submitted data is stored in the database this ID will identify which form was submitted. Later on, if OpenCms workplace users want to download the data of a form, they see a "Download Data" Link on the frontend page in the offline project which will produce an excel csv file with all data matching this ID. With this solution it is possible to have several siblings of the same page in different language folders all sharing the same `FormId` thus being exported all into a single csv file.
- **Mail text**: This is some additional text that is sent in the optionally generated Emails. This most often can be left out or some lines (e.g.: "New contact request made on http://www.alkacon.com made!") are sufficient as many mails in this form will be received.
- **Form field**: Many form fields may be added. They have a type selector that affects, which widget will be used in the resulting webpage for the data to enter.
    - **Label**: Should be entered. It will be displayed to the left of the input field and also in the generated email to identify the value entered by the website visitor.
    - **Mandatory**: If the mandatory flag is selected empty fields will not be accepted.
    - **Default value**: This field may be used to preset values for text fields or text area fields. **Especially for checkbox fields, radio button fields or select box fields the default value has to be used to specify the items available for selection**: A special syntax:
      "i1:Item 1|i2:Item2|i3:Item3" allows to specify each item separated by pipe symbol ("|"), where you may differ between the internal name for the selection that will be used in the email (here: i1,i2,i3,…) and the visible selection on the webpage (here: Item 1, Item 2, Item 3). You can just specify one value for each item: "Spaghetti|Coca Cola|Cerveza" if you don't require internal names. The sole purpose is to allow different localized versions that have translated visible items for each language but use the same value in the generated Email (in case a computer program reads those emails).
      Preselections (default items) may be marked with an asterisk like Mr in "Mrs|*Mr".
    - **Validation**: This field allows specifying a regular expression that has to match the entered value. If the user enters something that does not match such an expression submission will fail with a error message.
    - **Error Message**: Allows configuring a custom error message if validation fails.
- **Confirmation Text**: This text is shown if the website visitor successfully submitted the Email form.
- **Target Uri**: This most often should be blank. If configured after a successful submission the user's Web browser will be redirected to that URL.

- **Captcha field**: This is a powerful way to avoid robots / spiders [2] successfully send you Emails by showing an image and a text field below where the user has to type in the same characters as seen in the image.

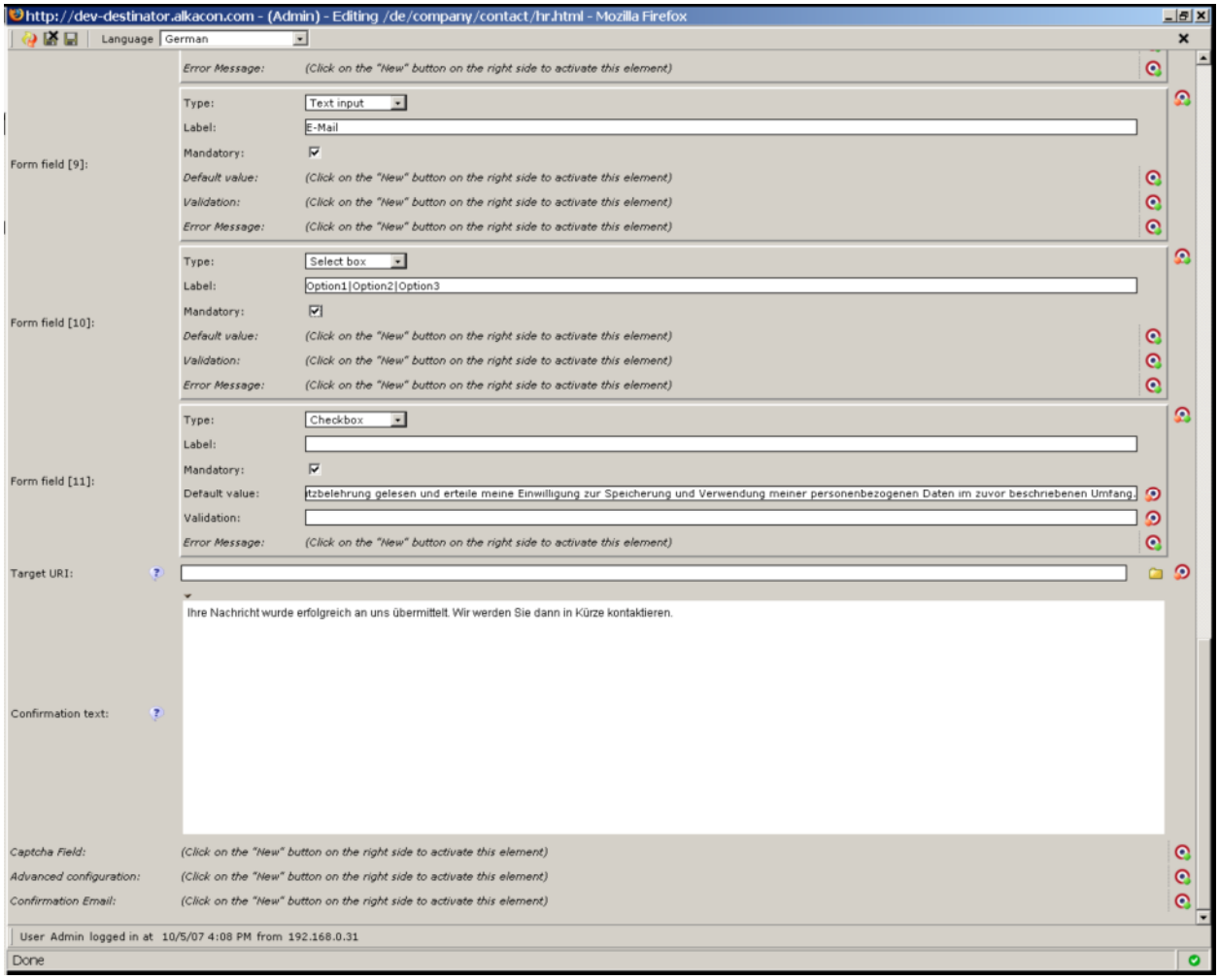


Figure 3: Configurations for Checkbox field and Select box field.

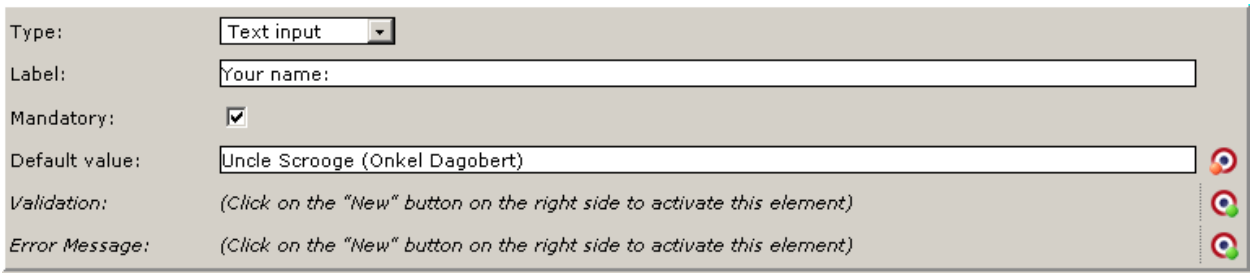
5.3 Webform Field Types

Many different types of fields for Webforms are available. They differ in functionality and in the resulting UI controls shown on the frontend of the Webform.

5.3.1 Text input

A normal single line input field. This is the best choice for short data to collect like "City". A validation value of "\d*" may e.g. be used to force digits.

[2] Computer programs that want to send you ads / spam Emails.



5.3.2 Text area

Offers more lines for more textual input. Most often used as "message" field.

5.3.3 Checkbox

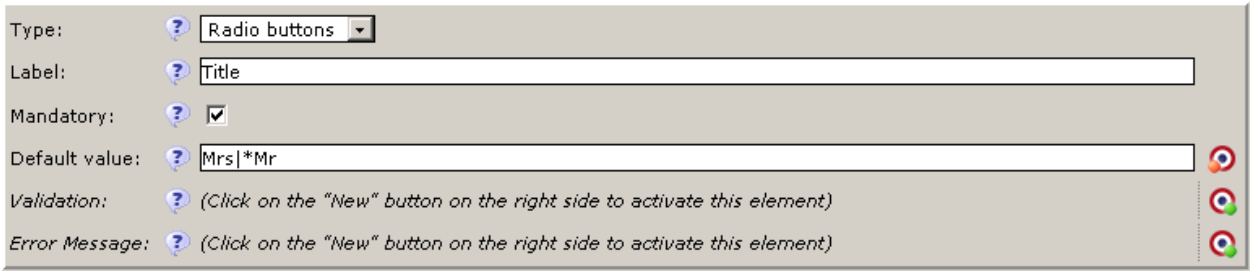
Will show up as a group of checkboxes on the frontend webpage. Each **`value:displayname`** pair like "**`value1:Lorem Ipsum`**" in the default value configuration will result in a single checkbox.

5.3.4 Privacy

A privacy field displays a checkbox and a link at the right side of it. It is very special. It may be used to let the website visitor confirm that they have read the "privacy statements" or "accept to the license" while the link will lead to a more detailed explanation. The link has to be site – relative like "/de/index.html" The link text is specified first, a separation colon follows and then the link has to be written in the Default value box: "Linktext:/de/privacy/index.html".

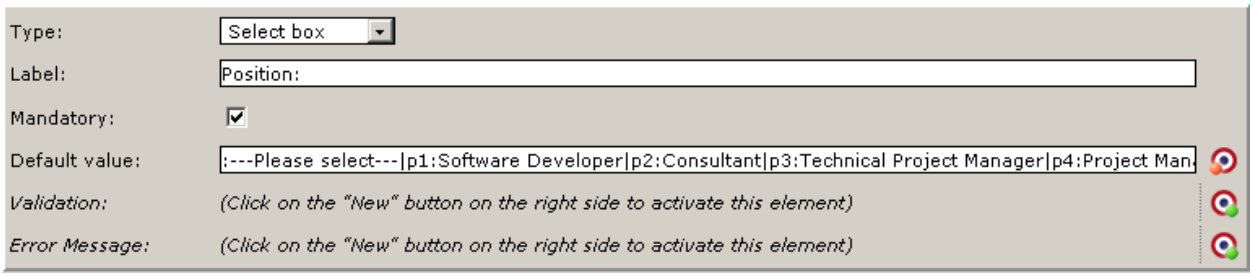
5.3.5 Radio Buttons

Will show a group of selectable items, where only one item may be selected.



5.3.6 Select Box

Displays as a select box with several options. This widget has the same function as radio buttons: Select a single item out of a limited amount of items. But it will use less space if many items have to be offered. Most often it is good to use radio buttons for only two or three alternatives and select boxes for more items.



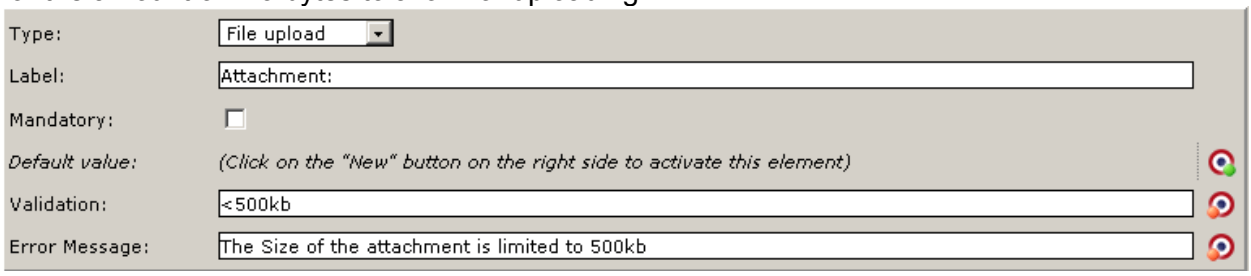
5.3.7 Hidden Field

Hidden fields may be used to let a certain value be in the submission but not be visible or editable by the website visitors. This value (the Default value) will be transmitted in the Email and / or into the database. This might be used to differ between several forms that have all other fields equal. One could see from which form the submission was made even if all other submitted value names are the same.

Some spiders might crawl the web to find forms to fill out in order to enter their spam messages to forums or guestbooks and fill out even the hidden fields. As humans cannot access these fields a change of value of the hidden field is a proof that a program has submitted the form.

5.3.8 File Upload

File upload fields offer to select a file on the local hard drive which will be uploaded to the OpenCms server. A speciality is the possibility to use the validation field for configuration of the maximum upload size in the form "**<xkb**" where "**<**" and "**kb**" are fixed and **x** is an integer value for the amount of kilo bytes to allow for uploading.



In case of email transport the upload will be attached to the mail. If the submissions are stored to the database the uploaded temporary files will be copied according to the module parameter "uploadfolder" and their full path will be stored in the database.

5.3.9 Email Field

Email fields are a comfortable way of validating the input as a valid email address. They are nothing more than a Text input with an internal additional regular expression.

5.3.10 Empty Field

Empty fields may be used to have additional space between form rows. Also their Default value will appear in the column together with the other input field: This can be used to show input field group – headlines. No data is produced / submitted by empty fields.

6 The front end view

Once created an Email field should seamlessly integrate with your web page.

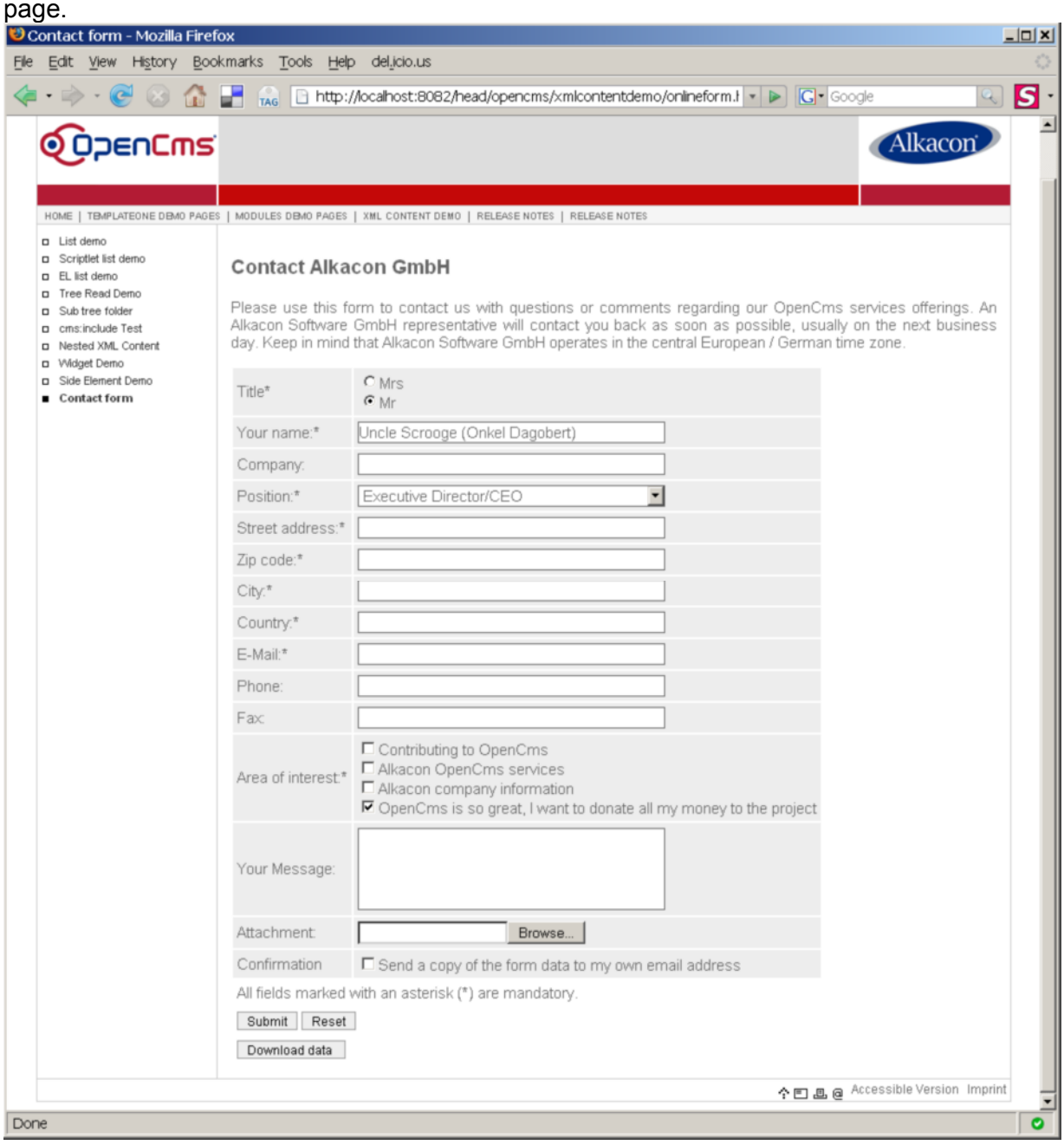


Figure 4: Demo frontend view of a Webform

When looking at a web form in the offline project of OpenCms[3] a "Download data" button is offered that will lead to a page where it is possible to export a Microsoft Excel CSV file with the exported data of the submissions of this form. It is also possible to export the data only in a selected time range.

[3] You view pages in the offline mode when you are logged in to the OpenCms Workplace and have the project slider set to „offline“.

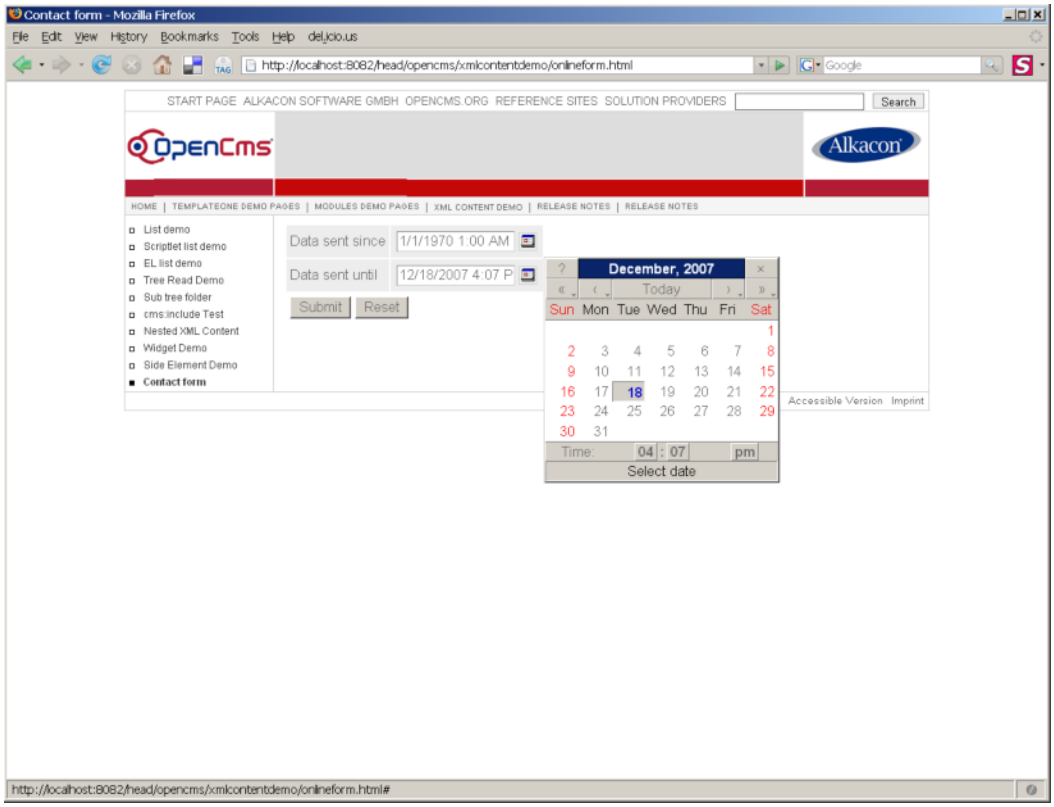


Figure 5: The data download page.

6.1 Customization of the frontend HTML

In case the layout is not OK you can configure the module parameter "**`message`**" of the OpenCms module **`com.alkacon.opencms.formgenerator`** to point at a properties file that contains the HTML snippets used for showing the webform in variations. The default file with these HTML snippets is located in the VFS: **`/system/modules/com.alkacon.opencms.formgenerator/classes/com/alkacon/opencms/formgenerator/workplace.properties`**. If a properties file with these HTML snippets and other localizations has been changed it is required to publish it and re-initialize the workplace in the OpenCms Administration.

7 Using the module API

All classes used to generate and configure the webforms are part of the package **`com.alkacon.opencms.formgenerator`**, **`com.alkacon.opencms.formgenerator.database`** and **`com.alkacon.opencms.formgenerator.database.export`**.

Package **`com.alkacon.opencms.formgenerator`** contains the implementation of the webform XML content data access along with the logic to display the webform and process the submission. The following classes are used:

- **I_CmsFormField**: Defines the methods required for form fields.

- **A_CmsFormField**: The abstract base class of field implementations.
- **CmsCaptchaEngine**: A JCaptcha implementation that allows configuring the captcha deformation of images with the XML contents in the VFS folder **`/system/workplace/admin/captcha/`**.
- **CmsCaptchaField**: A field implementation for generating captcha images and collection of the response phrase along with validation.
- **CmsCaptchaService**: A JCaptcha implementation to set up the captcha service that uses the **`CmsCaptchaEngine`**.
- **CmsCaptchaServiceCache**: A cache that avoids reading the captcha engine deformation configuration XML contents from database per request if possible.
- **CmsCaptchaSettings**: Bean for accessing captcha setting XML contents along with routines for reading from the XML contents and from requests.
- **CmsEmailField**: Text field implementation with integrated email format validation.
- **CmsCheckboxField**: Form field implementation for checkboxes.
- **CmsEmptyField**: Form field implementation for empty fields.
- **CmsFieldFactory**: A factory to create form field instances for a given type name.
- **CmsFieldItem**: Represents a single item of field implementations with items like **`CmsCheckboxField`**, **`CmsRadioButtonField`** and **`CmsSelectionField`**.
- **CmsFieldValue**: Represents a single input field value of a submitted form.
- **CmsFileUploadField**: Form field implementation for file upload fields.
- **CmsForm**: Bean for accessing webform XML contents along with routines for reading from the XML contents and from requests.
- **CmsFormHandler**: Contains the logic to process the webform (**`CmsForm`**): which page to show (confirmation, initial,…), where to send the data, error and validation processing,….
- **CmsHiddenField**: Form field implementation for hidden fields.
- **CmsPrivacyField**: Form field implementation for privacy fields: a check box with a link to a configurable target.
- **CmsRadioButtonField**: Form field implementation for radio buttons.
- **CmsSelectionField**: Form field implementation for select boxes.
- **CmsSelectWidgetXmlcontentType**: A highly configurable select widget used in the XML content editor for webforms to offer the different captcha preset XML contents found in the VFS.
- **CmsTextareaField**: Form field implementation for text areas.
- **CmsTextField**: Form field implementation for text fields.
- **Messages**: convenience class to access the localized messages of the webform package.

Package **`com.alkacon.opencms.formgenerator.database`** contains the database access layer for storing and reading submissions of webforms. The following classes are used:

- **I_CmsFormDataAccess**: The interface for the database access layer.

- **CmsFileUtil**: Utility class for RFS file access with support for logging and localized exceptions.
- **CmsFormDataAccess**: Implementation of the database layer access.
- **CmsFormDatabaseModuleAction**: Module action that ensures that the required tables for webform persistence are created if they do not exist in OpenCms startup process.
- **CmsFormDataBean**: Bean that stores the data of a single submission of a webform.
- **CmsFormDataEntry**: Represents a single field name – field value pair of a webform submission.
- **Messages**: convenience class to access the localized messages of the webform database subpackage.

Package **`com.alkacon.opencms.formgenerator.database.export`** contains a plain forward implementation for exporting webform submissions within a time – range to Microsoft Excel CSV files. The following classes are used:

- CmsCvsExportBean: Implementation of the database export that acts as a façade towards the database access layer