UNIVERSITÄT BERN

TPF
Technologieplattform

Fabrikstrasse 8
Büro A209
3012 Bern

031 6315513

SelfHelp WebApp - Create Web Applications Serving as a Platform for Research Experiments

Tutorial

Simon Maurer - simon.maurer@humdek.unibe.ch
4th October 2018

Contents

Chapter 1

Introduction

This documentation refers to the TPF project SLP-selfhelp. The project aims at providing a tool that allows to create a web application that serves as a platform for research experiments. The basic concept is as follows:

Pages are organised as a collection of sections that are rendered on the page, one below the other. Sections have different styles which define the appearance of the sections. Depending on the style of a section, the section has different fields which define the content of the section. The value of a field can be a simple plaintext or a collection of child sections which have their own styles and children.

Currently available styles include, but are not limited to, alert boxes, buttons, card containers, forms, media elements, tabs, lists, and support for markdown texts.

The app is designed in such a way that it can be extended with new styles or custom components without having to modify existing code.

The main purpose of this document is to provide a step-by-step tutorial of how to work with the admin section of the SelfHelp WebApp.

For an overview of existing styles and their fields refer to this demo page[1]. Instructions for configuring the web server can be found on this page[2]. For the documentation of the source code refer to this doxygen page[3]. Finally, this page[4] provides some information about possible extension and customization options of the WebApp.

In the following it is assumed that a server is up and running and an instance of this WebApp is made available.

[1] https://selfhelp.psy.unibe.ch/demo/styles
[2] https://selfhelp.psy.unibe.ch/demo/configs
[3] https://selfhelp.psy.unibe.ch/demo/doc/doxygen/html/index.html
[4] https://selfhelp.psy.unibe.ch/demo/extend

Chapter 2

Creating a New Page

In order to crate a new page the `select` and `insert` access right of **Page Management** are required. By default users in the group `admin` and `experimenter` are granted the mentioned access rights.

Once logged in, navigate to the *Content Management System (CMS)* in the menu *Admin* and click the button *Create New Page* in the top left corner. This opens the *Create New Page* form where properties of the new page can be entered:

Keyword This must be a unique name which is used to identify a page internally (e.g. `demo`). This is also the name that will be used in the CMS when a link to this page must be shown. Note that this field is not visible for a user without access to the CMS (e.g. a subject).

Header Position Enabled this field to make the page appear as a menu-point in the navigation bar of the WebApp. Drag and drop the item to the location you want it to appear. Note that the page will only appear as a menu item once the `title` field is set (this must be done after the page creation process).

Page Type The page type specified how the page content will be assembled. Choose type **Sections**.

User Input This field specifies whether the page will hold forms for user input. Leave this disabled.

Protocol The protocol specifies how a page is accessed. Enable **GET** and leave everything else disabled.

Url Pattern This field describes the url path of how the page will be accessible. By default the keyword is used.

Click the button **Create** to create the page and on success the button **To the new Page** to return back to the CMS with the new page selected.

2.1 Add Content to the Page

In order to add content to a page the `update` access right of **Page Management** is required. By default users in the group `admin` and `experimenter` are granted the mentioned access rights.

In a first step let's define the title of the page `demo`. The title is used as label for the menu item in the navigation bar of the WebApp and will appear in the browser tab. To set the page title select the pen symbol in the top right corner of the *Page Properties* card. Doing so will change the card colour to yellow and provide an input field where the title can be entered. Click **Submit Changes** to make the change permanent.

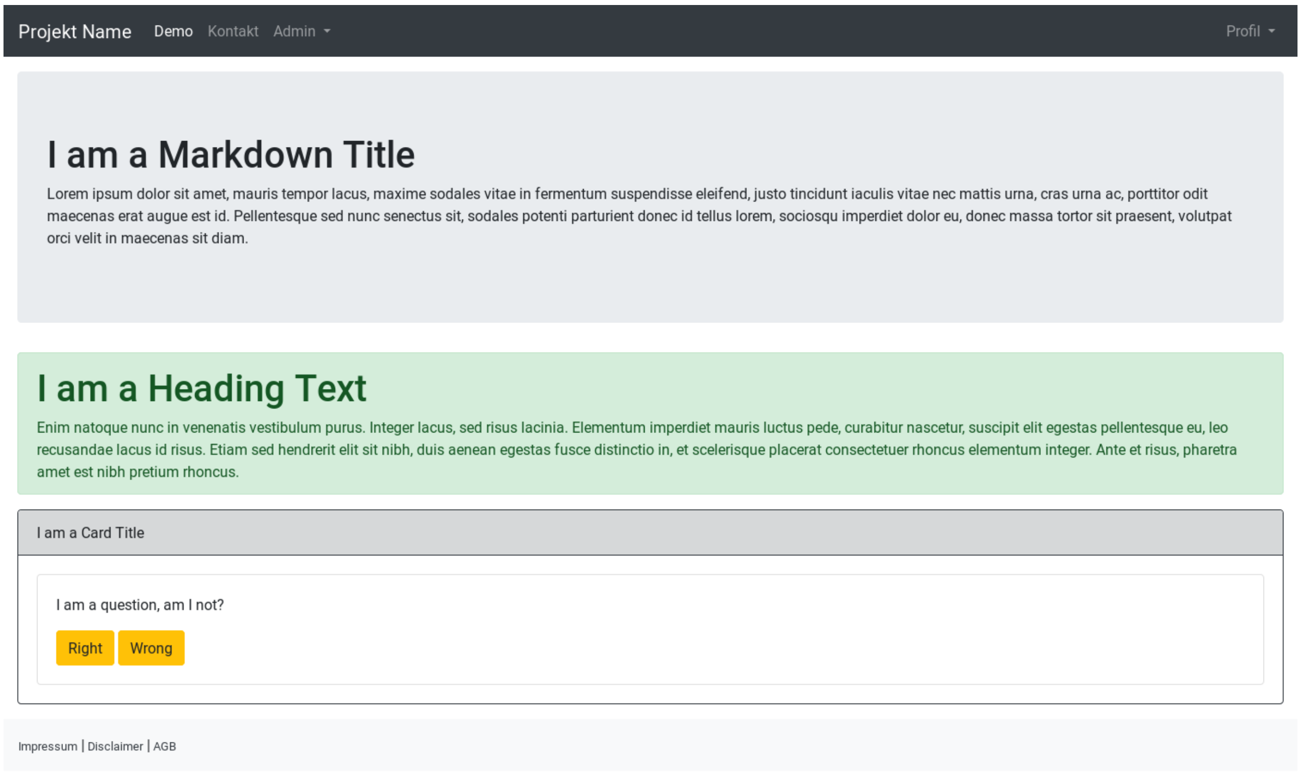
Adding content to the page is done by creating and adding sections to the page. This can be done by clicking on the button **Add** below the field `sections` in the *Page Properties* card. This opens the *Add Section* form where the name and the style of the new section can be defined. The name serves only to let you identify the section at a later point and can be any string. It is, however, recommended to use a logic structure in naming sections (it is a good idea to include the page name into the section name). Note that the style is always appended to the section name. The style defines how the section will look like and what type of content can be assigned to it. Let's choose the style `container` which is used as wrapper for other styles to be added to the page. This helps to manage the spacing at the edges of the page. Click **Add Section** to create the new section which will return you to the CMS with the new section selected in edit mode (the card *Section Properties* is yellow).

The next step is to edit the properties of the section. A container section only has the two fields `css` and `is_fluid`. Enable the field `is_fluid` to make the container span over the whole page. The field `css` is special in the sense that every style has this filed. It allows to specify css classes (sperated by spaces) that will be added to the root html tag of the style. Add the bootstrap class `my-3` to the field to add some spacing at the top and the bottom of the container (refer to the Bootstrap documentation from more details).

Now continue to add sections to the container you just created. For example:

- A section of style `jumbotron` which, itself, holds a section of style `markdown`. In the field `text_md` of the markdown section use markdown syntax to add a title and a paragraph of text.
- A section of style `alert` which has a child section of style `heading` and another child section of style `plaintext`.
- A section of style `card` which has a child section of style `quiz`.

Doing all of this and tweaking the available fields of the different styles should net you with something resembling Figure 2.1



I am a Markdown Title

Lorem ipsum dolor sit amet, mauris tempor lacus, maxime sodales vitae in fermentum suspendisse eleifend, justo tincidunt iaculis vitae nec mattis urna, cras urna ac, porttitor odit maecenas erat augue est id. Pellentesque sed nunc senectus sit, sodales potenti parturient donec id tellus lorem, sociosqu imperdiet dolor eu, donec massa tortor sit praesent, volutpat orci velit in maecenas sit diam.

I am a Heading Text

Enim natoque nunc in venenatis vestibulum purus. Integer lacus, sed risus lacinia. Elementum imperdiet mauris luctus pede, curabitur nascetur, suscipit elit egestas pellentesque eu, leo recusandae lacus id risus. Etiam sed hendrerit elit sit nibh, duis aenean egestas fusce distinctio in, et scelerisque placerat consectetuer rhoncus elementum integer. Ante et risus, pharetra amet est nibh pretium rhoncus.

I am a Card Title

I am a question, am I not?

Figure 2.1: A demo page of type **Sections**.

2.2 Create a Menu of Pages

Up to this point we created a section page and added the link to this page to the navigation bar. It is also possible to create a menu item with multiple page links. It is, however, not possible to change the position or hierarchy of a page once it is created. A workaround is the remove the page and create a new one that fits those requirements. Note that sections remain untouched when a page is deleted, hence, recreating a page with its initial content is not a big deal.

In order to delete a page the `delete` access right of **Page Management** is required. By default users in the group `admin` and `experimenter` are granted the mentioned access rights.

As a demonstration let's create a dropdown menu in the navigation bar and add the content we created before as a child element:

First, the recently created page `demo` has to be removed. To do this select the page in the *Page Index* of the CMS, open the card *Delete Page* (red), and click the button **Deplete Page**. This opens the *Delete Page* from where the keyword of the page has to be entered into the input field to confirm the deleting process. Click **Delete Page** and on success click **Back to CMS** to return to the CMS.

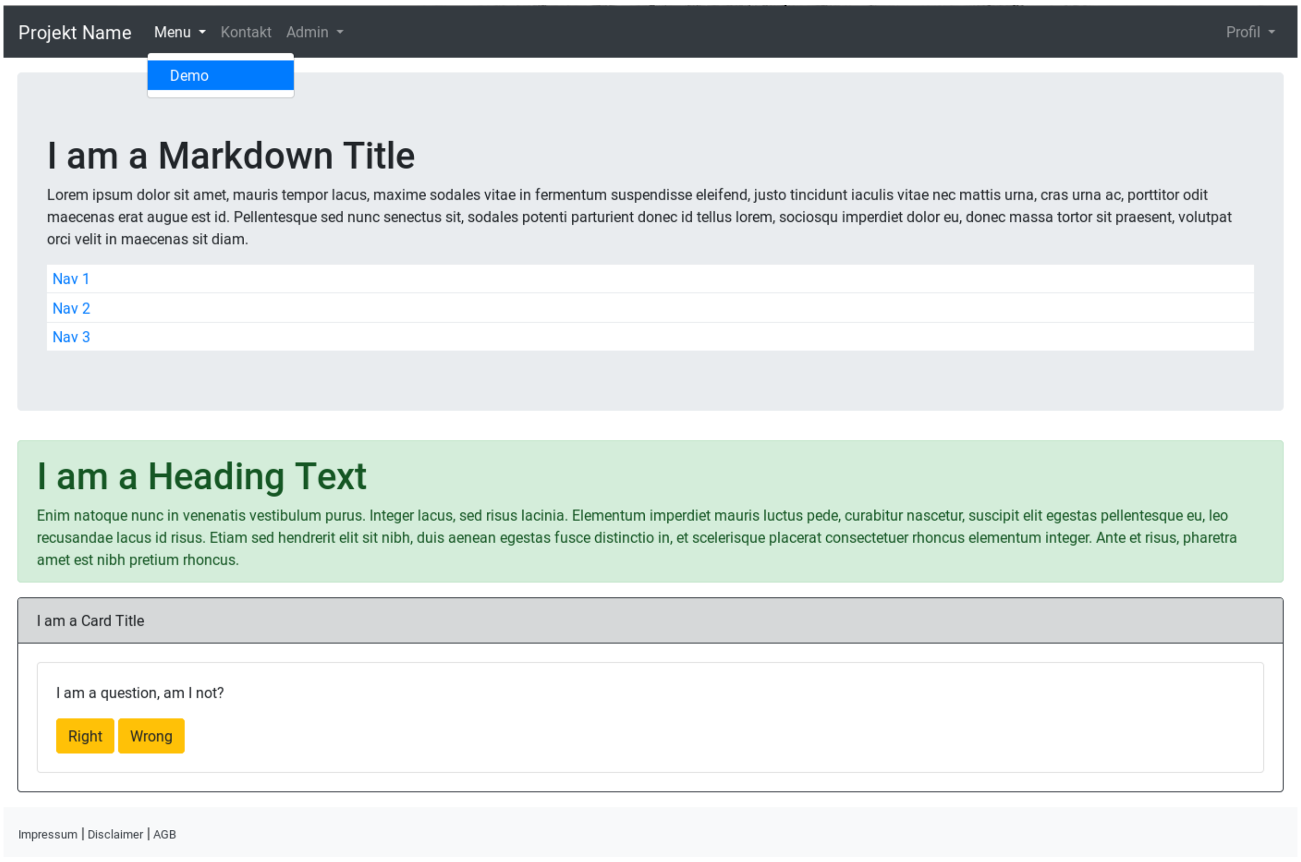
Now, create a new page which will only serve as the root menu item in the navigation bar. For this purpose I propose to append the string `-link` to the keyword of the new page (e.g `demo-link`). Enable the field `Header Position` and choose the appropriate position. Leave the remaining fields untouched and click **Create**.

Back in the CMS define the title of the page and then click the button **Create New Child Page** in the card *Create New Child Page*. Once again the *Create New Page* form is opened where we can recreate the page `demo` from the beginning of this chapter. Note that when enabling `Header Position` only the new page is shown. This is because this page is the first child page in this menu. Back in the CMS define the title of the page and click the button **Add** below the field `sections` to add the sections we created before.

Once the *Add Section* form opens you will notice that on the right in the card *Unassigned Sections* the container section we create before is listed. Select the section and click **Add Section** which will restore the page as it is represented in Figure 2.1 with the exception that now, the page is reachable through a dropdown menu in the navigation bar.

2.3 Navigation Page

Often in might be interesting to navigate from page to page by clicking on buttons **Next** and **Back** or by selecting the corresponding page in a list. To achieve this the page type `Navigation` can be used. A navigation page is a special type of section page where only one section is rendered at a time. A set of root sections, or navigation sections, each of style `navigationContainer` is assigned to a navigation page. An id that is postfixed to the url indicates which section is rendered. To demonstrate this, let us create such a navigation page which we can reach from our page `demo` which we created earlier.



I am a Markdown Title

Lorem ipsum dolor sit amet, mauris tempor lacus, maxime sodales vitae in fermentum suspendisse eleifend, justo tincidunt iaculis vitae nec mattis urna, cras urna ac, porttitor odit maecenas erat augue est id. Pellentesque sed nunc senectus sit, sodales potenti parturient donec id tellus lorem, sociosqu imperdiet dolor eu, donec massa tortor sit praesent, volutpat orci velit in maecenas sit diam.

I am a Heading Text

Enim natoque nunc in venenatis vestibulum purus. Integer lacus, sed risus lacinia. Elementum imperdiet mauris luctus pede, curabitur nascetur, suscipit elit egestas pellentesque eu, leo recusandae lacus id risus. Etiam sed hendrerit elit sit nibh, duis aenean egestas fusce distinctio in, et scelerisque placerat consectetuer rhoncus elementum integer. Ante et risus, pharetra amet est nibh pretium rhoncus.

I am a Card Title

I am a question, am I not?

Figure 2.2: A demo page of type **Sections** with a list of navigation items added to the jumbatron.

We start by creating a new page of type `Navigation`. Let this page be a child of the page `demo-link`. There is no real importance to this but it helps to keep things grouped together in the CMS. So, select the page `demo-link` and click the button **Create New Child Page**. Set a keyword (e.g. `demo-nav`) and set the type to `Navigation`. Do not enable the field `Header Position` and let all other fields as is. Note that when selection the type `Navigation` the pattern `[i:nav]` as appended to the `Url Pattern`. This pattern will be replaced by the id of the selected navigation section to be displayed on the naviagtion page.

Back in the CMS the page fields can be edited. As with section pages, the field `title` defines what will be displayed in the browser tab. The fields `label_back` and `label_next` will be the labels of the buttons **Next** and **Back**, respectively that were mentioned before. These buttons will only be displayed if also the field `has_navigation_buttons` is enabled. The field `is_fluid` defines whether the content is stretched to the whole of the page or not. For more information about the rest of the available fields refer to the `nestedList` style documentation.

Note that a navigation page has the field `sections` (which we already know from section pages) that allows to add section, one below the other, to the navigation page. In addition to this, the field `navigation` allows to add navigation sections from which only one is displayed at a time. Which section to display is defined by the trailing id in the url.

Let's add some navigation sections to see what this does: Click the **Add** button below the field `navigation` which will open the *Add Section* form that we already know. Enter a name (e.g. `demo-nav1`), select the style `navigationContainer`, and click **Add Section**. The style `navigationContainer` is used to wrap the content of a navigation section. The field `title` serves as title for the navigation list item representing the navigation section wrapped by this container. The field `text_md` is rendered at the top of the container. Note that the special string `@title` can be used to render the field `title` within the field `text_md`.

As an example set the field `title` to "`Nav 1`" and the field `text_md` to "`# Demo @title`". Select again the page `demo-nav` and add two more naviagtion sections "`Nav 2`" and "`Nav 3`".

Finally we need to link to this navigation page and all its navigation sections. We can achieve this by adding a new section to the page `demo` we created in the beginning: Select the page `demo` and add a new child section of style `nestedList` to the jumbatron for example. Back in the CMS only fill out the field `items` with the content `{"nav_page": "demo-nav"}`. This is a special json string which tells the nested list to go and fetch all navigation sections of the navigation page `demo-nav`.

As a result you should get something similar to what is represented in Figure 2.2.

The navigation page `demo-nav` is reached by clicking on any of the items displayed in the nested list. The result should look similar to Figure 2.3.

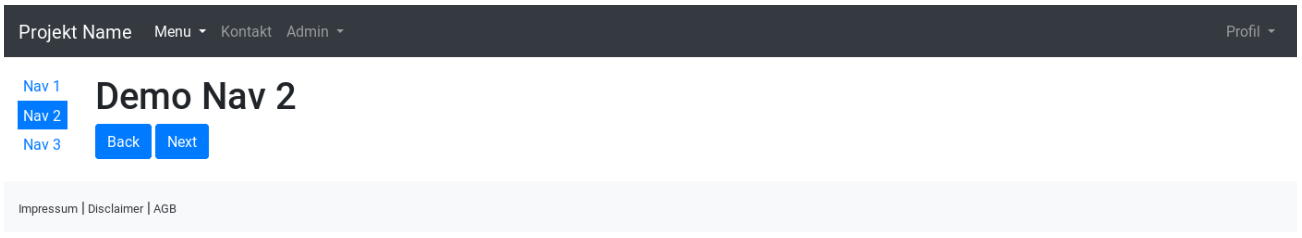


Figure 2.3: A demo page of type **Navigation** with a list of navigation items displayed on the left and navigation buttons displayed at the bottom of the page.

Note that sections that are added to the navigation page `demo-nav` through the filed `sections` will appear on top of each navigation section as demonstrated in Figure 2.4.

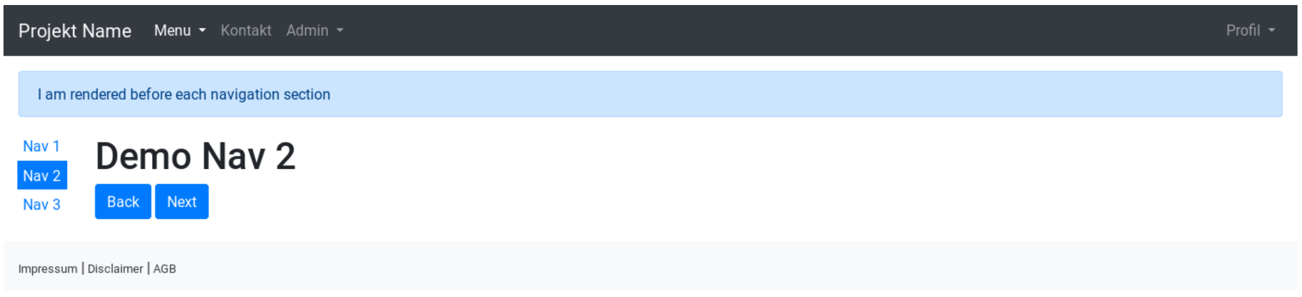

Projekt Name
Menu
Kontakt
Admin
Profil
I am rendered before each navigation section
Nav 1
Nav 2
Nav 3
Demo Nav 2
Back
Next
Impressum | Disclaimer | AGB

Figure 2.4: A demo page of type **Navigation** where before each navigation section the same section is displayed (the blue alert box).

Note that navigation sections can be arranged hierarchically.

Chapter 3

User Management

In order to view, create, modify, and delete users and groups the `select`, `insert`, `updatei`, and `delete` access rights of **User Management** is required. By default users in the group `admin` are granted all access rights while the users in the group `experimenter` are granted all access rights safe `delete`.

3.1 Users

To manage users navigate to *Users* in the menu *Admin*. All registered users are listed in a card on the left. A user can be either `active` (can log in and visit all accessible pages), `inactive` (cannot login as long as the account is not verified), or `blocked` (cannot login until the blocked status is reversed). The cards on the left hand side provide an interfaces to block, unblock, or delete a user and allow to manage the groups a user belongs to. A user inherits all access rights of the groups the user belongs to.

3.2 Groups

To manage groups navigate to *Groups* in the menu *Admin*. All groups are listed in a card on the left. Manage the access right of each group by checking or unchecking the appropriate checkbox in the card *Group Access Rights*.

Chapter 4

Extending the WebApp

The code base is designed in a modular way such that it is easy to extend the functionality of a WebApp. Please refer to the documentation on the demo page[1] for more information on extending the source code:

Workflow provides a short overview on how to manage the source code and how to report back the changes made.

The Concept Behind the Code provides an overview on the code structure and introduces the vocabulary used within the code.

Evaluate User Input explains in very broad terms how user inputs can be handled.

Customize the Theme provides several approaches to change the look and feel of the web app without changing to much of the source code.

Create a Custom Style

Create a Custom Component

Create a Custom Page

[1] https://selfhelp.psy.unibe.ch/demo/extend