Web Technologies - Lab 2

Mohamed Sellami, ISEP

Contents

1 HTML, CSS and JavaScript

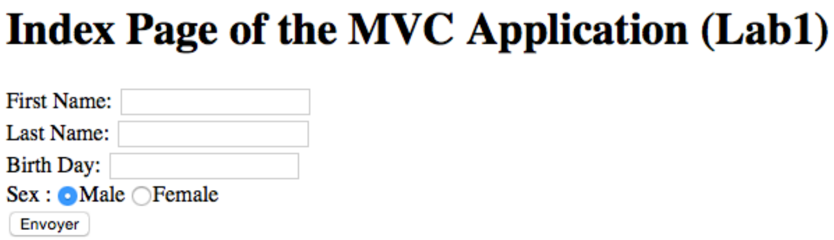
In this part we will define a style for the HTML page we developped during the first lab for the MVC application (`index.html` in Section 3.4). We will also define some basic validation in JavaScript.

Here is a sample source for this Web page (you can also use the one you defined during our last practical Lab):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html><head>
<title>Index MVC application</title>
</head>
<body>
```

```
<h1>Index Page of the MVC Application (Lab1)</h1>
<form>
        First Name: <input type="text" name="fName" /><br>
        Last Name: <input type="text" name="lName" /><br>
        Birth Day: <input type="text" name="bDay" /><br>
        Sex :<input type="radio" name="sex" value="male" checked>Male
        <input type="radio" name="sex" value="female">Female
        <br><input type="submit" />
</form></body></html>
```

The result is:



Index Page of the MVC Application (Lab1)

First Name:

Last Name:

Birth Day:

Sex : ● Male ○ Female

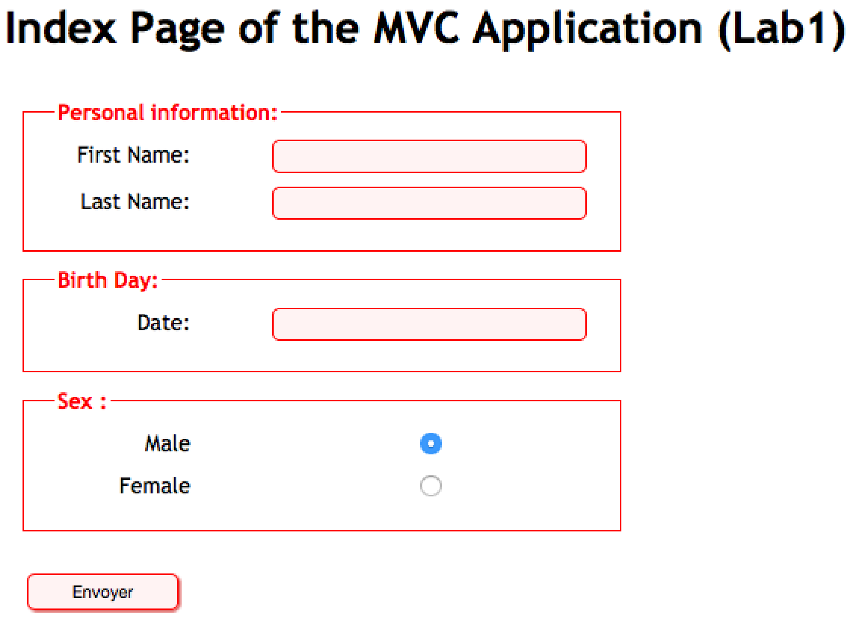
Figure 1: Output of the sample HTML form (no style)

1.1 Part 1: CSS style sheet

Update the HTML code, define a CSS style sheet (`myFirstStyle.css`) and apply it to `index.html`. The result must look like in Figure 2.

Hints:

- use the HTML `<fieldset>` tag.
- add the HTML `<label>` tag to the `<input>` elements.
- Displays `<label>` element as block element using the CSS display property (`display:block;`)
- The `background-color` property of `<input>` element is #FFF3F3.
- When the focus is on an input element the `background-color` becomes white.



Index Page of the MVC Application (Lab1)

Personal information:

First Name:

Last Name:

Birth Day:

Date:

Sex :

Male ●

Female ○

Envoyer

Figure 2: Output of the sample HTML form whith a CSS style sheet

1.2 Part 2: Basic form validation (JavaScript)

Define a JavaScript function (`validateMyForm()`) that will be executed when the user ckicks on the submit button. This function (1) checks that the different input elements are filled and (2) displays an alert box with the missing fields if any, else the alert box contains a summary of the form content.

Define the JavaScript function in a seperate file (`myFirstScript.js`).

1.3 Part 3: Dynamically change the style and the DOM

Define another JavaScript function (`toogleFM()`) that will be executed when the user clicks on the male or female radio buttons. This function:

- changes the color of the fieldset border and its legend to blue if female is clicked, red if male is clicked.
- changes the border color of the inputs (text and button) to blue if female is clicked, red if male is clicked.

- changes “Index Page of the MVC Application (Lab1)” in the `<h1>` element into “Hello Madam” if female is clicked and into “Hello Sir” if male is clicked.

In addition to the JavaScript function, you might need to bring some changes in the css style sheet and the HTML source.

2 Simple scripts with Jquery

Create an HTML page with the following code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>My Title</title>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js">
</script>
<script src="myJqueryScript.js" type="text/javascript"></script>
</head>
<body>
<h1 id="myTitle"> Default Tile</h1>

<div id="div1"> Some text</div>

<ol>
<li><input type=button value=’Change <h1> Title’ onClick=’f1();’>
: Changes the text in &lt;h1&gt;.
<li><input type=button value=’Change the text in <div>’ onClick=’f2();’>
: Changes the text in &lt;div&gt;.
<li><input type=button value=’Insert some text after the text in <div>’
onClick=’f3();’>
: Insert some text after the text in &lt;div&gt;.
<li><input type=button value=’Change the <title>’ onClick=’f4();’>
: Changes the title (&lt;title&gt;) of the Web page.
<li><input type=button value=’Toggle’ onClick=’f5();’>
: toggles between hide() and show() the text in &lt;div&gt;.
</ol>
</body>
</html>
```

- Use jQuery to create a file `myJqueryScript.js` defining the functions `f1(),f2(),f3(),f4() and f5()`.

3 Login/Password check using AJAX

In this part, we will use AJAX to check whether or not a login and a password provided in a form are correct prior to submitting the entire form to the server.

1. Create an HTML page with a simple form containing two text input elements (login and password) and a submit button.
2. When clicking on the button, ask the server using AJAX with jQuery, whether or not the <login,password> are correct. To ensure this, create a Java Servlet on the server side that will check the <login,password> combination: returns 1 if they are correct (<login="abc",password="123">), 0 if not.
3. display an alert box with the result.

Hint:

- to stop the default action of the submit button click event, use the method preventDefault. Details at: http://api.jquery.com/event.preventdefault/
- Update your code to display the result in the HTML page instead of an alert box.

4 Autocopmlete

4.1 Using the jQuery UI API and a predefined cities list

Create a simple HTML page with one input field to get a city name. Use the jQuery UI autocomplete function (see http://jqueryui.com/autocomplete/) to add an auto-complete functionnality to this input element.

4.2 Using AJAX and a cities API

In the previous version, cities were stored at the client side (if you used the jQueryUI example). Create another version using AJAX and the free AJAX cities JsonP API (http://www.geobytes.com/free-ajax-cities-jsonp-api.htm). Just follow the example provided in the previous link.