Developing Apps for the BlackBerry Smartphone

Lab # 1

Getting Started with HTML and CSS

The objective of this lab is to review some of the concepts in HTML and CSS for creating WebWorks application for the BlackBerry Smartphone. In this lab, we'll be dealing with the interface of the application. Before attempting this lab, please be sure to download and install the BlackBerry WebWorks SDK for Smartphones available at https://bdsc.webapps.blackberry.com/html5/download/sdk. Make sure you that you have Java Software Development Kit installed and configured on your computer. Please also install the Ripple Emulator available at: https://bdsc.webapps.blackberry.com/html5/download/ripple. The Ripple emulator is a mobile environment emulator that is custom-tailored for mobile HTML5 application development and testing. You do not need to worry about signing your application as we'll be testing it on the emulator. Instructions on how to setup and start using the emulator can be found here: https://bdsc.webapps.blackberry.com/html5/documentation/ww_getting_started/getting_started_with_ripple_1866966_11.html. This lab also assumes that you have knowledge of basic HTML and CSS.

Creating the HTML interface

- Open Lab1.zip extract it to a folder of your choice, then open index.html in your favorite HTML editor.
- We'll use DIV tags as containers on this page. Start by viewing our root DIV element with the id attribute set to "container" which already exists on the HTML page. We'll be making other containers within this root element in order to create the interface.
- Start by creating a header container with a DIV tag with the id "header"
    - `<div id="header">`
- Within that header DIV tag, create another container with the id "logo" and "headertext". These two containers will be used to place the title of our application on the HTML page.
- Inside the "headertext" container, insert the following:
    - `<center>Google Scholar CMER Lab</center>`
- The HTML code inside your DIV with the id "container" should now look like the following:

```
<div id="header">
    <div id="logo"></div>
        <div id="headertext">
                <center>Google Scholar CMER Lab</center>
        </div>
</div>
```

- If you open index.html in your web browser now, you'll notice that the title is simply placed in the center, and does not look very fancy at all. Don't worry; we'll add some CSS code later on in the lab to make the page look nicer.
- We'll now add containers for some buttons that will be placed on the application. Underneath the "header" container, create two more containers with the id "visitcmer" and "aboutbutton".
- Inside the "visitcmer" container, create a HTML button with the text "Visit CMER website" on the button.
    - `<input type='button' value='Visit the CMER website'/>`
- Inside the "aboutbutton" container, create another HTML button that says "About the application."

- <input type='button' value='About the application'/>

- If you open the page in your browser now, you'll see things aren't nicely lined up, so let's start working on making the interface look better now. The first thing to do is change the background color. This is done in the CSS file with the following:
  - background-color: #aeaeae;
- Create a body tag in CSS with the following text:

```
body
{
    background-color: #aeaeae;
    margin: 0;
    padding: 0;
}
```

- Now we'll add styles to make the title look more presentable. Add the following CSS code to create the header background. We'll use a picture included in the images folder in the lab, and repeat it horizontally. We'll also remove any border, and set the height to 100px.

```
#header
{
    height: 100px;
    background-image:url('../images/topbackground.png');
    border: 0;
    background-repeat:repeat-x;
}
```

- The text color and logo also needs to be set. The logo can be anything you want, but for the purposes of this lab, make sure the height of the image you want to use for the logo is 100px. In the lab files, the CMER logo is included to make things easier. Add the following code to the CSS file to add the logo to the interface

```
#logo
{
    background-image:url('../images/cmerlogo.jpg');
    background-repeat:no-repeat;
    height: 100px;
    width: 200px;
    float: left;

}
```

- In this case, the image will not repeat and we define the width and height directly in the code. The image also floats to the left.
- The text in the interface should stand out. CSS can be used to change the color, height, style, etc. of the text in the header. Include the code below in your CSS to change the appearance of the text in the header. In this case, we pad the text 30px from the top of the container it is in, to center it more vertically.

```
#headertext
{
    height: 45px;
    font-family: Geneva, Helvetica, sans-serif, bold;
    color: white;
    font-size: 20px;
```

```
    padding-top:30px;
}
```

- To adjust the location of the buttons in the application, CSS is also needed. Z-index is used in this CSS code to give the elements a different stack order. In this case, the buttons will be 1 stack above the rest of the elements in the page, which has a default z-index of 1. This allows us to change the location of the element without affecting other items on the page.

```
#visitcmer
{
    padding-left: 205px;
    padding-top: 40px;
    position:absolute;
    z-index:2;
}

#aboutbutton
{
    padding-left: 5px;
    padding-top: 40px;
    position:absolute;
    z-index:2;

}
```

- Your page should now look something close to what is shown below

Blackberry Bold 9900 and Blackberry Torch 9810

- Next, the search input-field should be added to allow users to input a name to search for using Google Scholar. First, some code needs to be added to the index.html file to create a search

box. Create a DIV container with the id “scholarsearchbox”, and place it underneath the “aboutbutton” DIV container.

- Within that container, create a HTML input field with the following attributes: type="text" and id="searchbox".
- Also inside the “scholarsearchbox” container, create another DIV container with the id “searchbutton”. Inside that container, use the HTML button tag to create the search button. The text in between the open and close button tag should be “Search”. The “scholarsearchbox” container should look something like the following:

```
<div id="scholarsearchbox">
        <input type="text" id="searchbox" />
        <div id="searchbutton">
            <button>Search</button>
        </div>
</div>
```

- You’ll notice we use the button tag this time instead of the input HTML tag. This is used simply to inform you that there is more than one way to make a button in HTML.
- If you open the HTML page now, you’ll see that the input box and search button are behind the “About the application” button. This is due to the z-index CSS attribute we used earlier (The “About the application” button is now in the second stack, while the input box and search button are in the first stack).

Blackberry Bold 9700

- To fix this interface issue, we’ll use CSS to make everything look nicer.
- First, center the input box by using CSS to edit the position of the “scholarsearchbox” container. You should add the following attributes to the “scholarsearchbox” element using CSS. You should know how to do this based on previous steps we’ve done in the lab.
    - Top padding of 100px
    - Left padding of 10px
    - Bottom padding of 20px
- The reason we use bottom padding is so that anything underneath this element will have some space before it appears on the page. This helps to make things less cluttered.

- Next, adjust the input element within the "scholarsearchbox" using CSS. This is done with the following code.

```
#scholarsearchbox input
{
    ..code to edit style goes in here..
}
```

- The attributes within this element should allow the input box to:
    - Have a white background
    - Have black text when typing
    - Have a solid black border with a width of 2px.
    - Have a width of 300px
    - Have a height of 40px
    - Have a font size of 22px
    - Should have absolute positioning
    - Have a Z-index of 2
- If you're unsure how to do any of the above, check the CSS online documentation here.
- If you load the page in your browser, you may notice the search button is missing. This is because it is behind the search box. CSS should also be used to re-position and change the appearance of this element.
- Use CSS code to do the following to the "searchbutton" element inside the CSS file. The code should be placed in:
    - Padding should be 320px to the left
    - The positioning should be absolute
    - The z-index value should be 1

```
#scholarsearchbox #searchbutton
{
        ..code to edit style goes in here..
}
```

- The style of the actual button should also be changed to make it bigger and stand out more. Add the code to make the following changes to the button in the CSS file.
    - Make the button width 120px
    - Make the button height 45px
    - Change the font size of the button to 22px

```
#searchbutton button
{
        ..code to edit style goes in here..
}
```

Your HTML page should now look like:

Blackberry Torch 9810

- Next, create a DIV container to tell the user how the use the application. Give this DIV the id of "abouttheapp". This container should go under the "scholarsearchbox" container.
- Underneath that, create two more DIV containers for search results and search information. Give these an id of "results" and "searchinfo" respectively.
- In between the DIV container with the id "abouttheapp", type. This application allows you to use Google Scholar to obtain statistics about any published Author.
- What you have added should look like something similar to:

```
<div id="abouttheapp">
        This application allows you to use Google Scholar to obtain statistics about any published Author.
</div>
<div id="results">
</div>
<div id="searchinfo">
</div>
```

- Next, create a DIV container for the loading icon. This DIV should be placed under the 'searchinfo' container. The div should be hidden by default and should be visible only when the user clicks the search button. This allows the user to know that something is happening, and they should wait for the results. In order to have this DIV hidden by default, style="display:*none*" should be added inside the tag. The id of this tag should be 'loading'.
- Inside the opening and closing of the div, place an image inside there with the following attributes.
    - Height should be 200px
    - Width should be 200px
    - The location of the image source is "images/loading.gif"
- The HTML code added should look similar to:

```
<div id="loading" style="display:none">
        <img src="images/loading.gif" height="200px" width="200px"/>
</div>
```

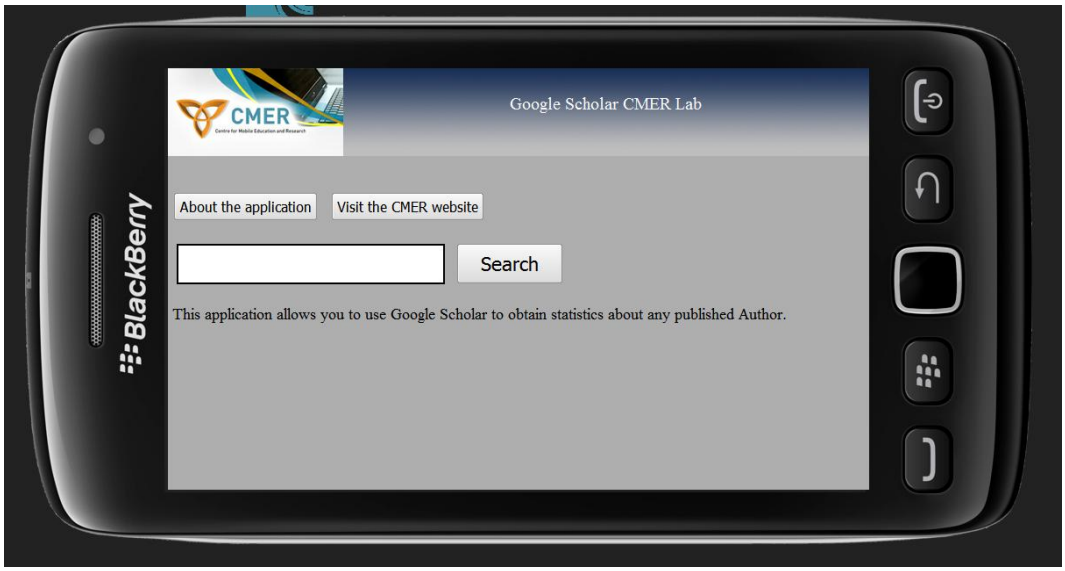
- Next, CSS needs to be used to correctly position the elements you have added. For the container with the id "abouttheapp", it should have the following attributes.
    - Padding on all sides of 5px
    - Top padding of 50px
    - The text should be aligned in the left
    - The font size should be 18px
    - The text color should be black

- For the “results” container, the following attributes should be applied.
    - Top padding of 10px
    - Font size of 20px
    - The text color should be black
    - The text should be bold
    - The padding on the left should be 22px

Within the results, there will also be a button that needs to be centered. The ID of this element will be “viewpapersbutton”. Add the CSS code to add 100px of padding on the left to this element only if it is in the results container. The CSS code to do this is similar to the following

```
#results #viewpapersbutton
{
    padding-left: 100px;
}
```

- For the “searchinfo” container, the following attributes should be applied.
    - The text should be aligned in the center
    - The font size should be 20px
    - The color should be black
    - The font should be bold
    - The padding on the top should be 100px
- That’s it! The HTML interface is now finished. Your final product should look similar to:



Blackberry Torch 9850

Testing the Application

- Now test your app using the Ripple Emulator.
- The instructions for using the Ripple emulator are found at https://bdsc.webapps.blackberry.com/html5/documentation/ww_getting_started/getting_started_with_ripple_1866966_11.html and it will guide you through the installation and testing

procedure. If you do not have a web server to host your files on, follow the instructions here and Ripple will host them for you.

- Next we will test in the BlackBerry simulator. The simulator can be downloaded from: http://us.blackberry.com/developers/resources/simulators.jsp Choose the Bold 9900 and OS 7.0. This provides an experience more akin to what you will find in the Ripple Emulator. The emulator is useful for quick testing, however the simulator should still be used for final applications tests.
- The simulator is installed into the following directory C:\Program Files\Research In Motion\BlackBerry Smartphone Simulators 7.0.0 on most systems
- Run the Simulator using the <model number> and .bat
- Save the directory of the Lab as zip file called "Lab1.zip"
- At a command prompt, navigate to the installation folder for the BlackBerry WebWorks Packager. The file path may vary based on where you installed the BlackBerry WebWorks Packager:

```
cd C:\Program Files\Research In Motion\BlackBerry WebWorks SDK <version>
```

Compile the application by using the following syntax:

```
bbwp <directory>Lab1.zip –o <output>
```

Load Blackberry application by going to File\Load Blackberry Application or Theme… and select the Lab1.cod located in <directory>\bin\StandardInstall\Lab1.cod in the simulator.

Blackberry Bold 9900

- Ensure your application loads successfully and looks similar to the above application. If not, look over your code for errors. Once completed, move on to Lab 2 for the next steps.