**blackCSIS210 - Data Structures**

WWeb App. Dev.

LLaboratory 2

**Lab 1**

# Names Roeldi Staro, Mark Smith, Kate Frisch

#

# General Lab Procedures

* You should create a directory (folder) in your home account called csis390. At the beginning of each lab, create a new sub-directory called labX, where X is the lab number.
* Files used in the lab can be found on the course canvas webpages.
* Turn in this lab sheet stapled to print outs of the code you produce as needed in each assigned section from the laboratory manual. These sheets should be in order. One lab submission is sufficient for each group.
* You can find documentation for the HTML 5 specification and the CSS reference respectively at

<https://html.spec.whatwg.org/>

<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>

**Lab Objectives**

* Create a bank check application, using form tags from HTML5.
* Create a webpage with various types of css styling
* Use winSCP to move files from a local machine to a server.

## Lab 1

#### Part 1 – HTML Forms

In this part of the lab, you will create an HTML webpage that mimics a signed bank check. You will be using forms that will represent the fields needed to perform this task.



* Above is an example of a bank check. It will have six areas for form input.
	+ Date
	+ Pay to the order of:
	+ $ (dollar amount in numerical form)
	+ DOLLARS (dollar amount in text form)
	+ MEMO
	+ SIGNATURE
* Create an HTML page with BANK CHECK as the title.
* Within the body, create a form tag with the following attributes.
	+ id = “checkinput”
	+ method=”post”
	+ action=”noAction.exe”
* Within the form, create a fieldset element for your entire check.
	+ According to the HTML5 whatwg specification, what does a fieldset do for your form inputs?

 **Groups widgets together**

* + Add a legend tag, using an appropriate bank name, within the fieldset tags. What does that do to your form?

 **Adds a caption to the fieldset box**

* Following the legend, add an input tag that will represent the date.
	+ Using an input tag to enter the date information
		- type = “date”
		- name = “checkDate”
	+ Add a text label to the input form on the left that says **Date:**
	+ Right justify your form. (HINT: Use a DIV TAG with an ALIGN attribute)
* Next, add a text input with a left label **Pay to the order of:**
	+ name=”checkReceiver”
* Next, add a dollar input with a left label **$**
	+ Use an input with a type=”number”
	+ name=”checkAmount”
* Both of these labels should be on the same line, with the text input left justified, and the dollar input right justified.
* Next, add a text input with a right text label Dollars
	+ name=”checkAmountText”
* Next, add a text input for the Memo line and a text input for the signature. Use appropriate name attributes, and be sure that these two inputs appear on the same line, with the Memo left justified, and the text input right justified.
* Finally, spruce up your check with a background image.
* Once finished, upload your webpage to your servers and write down the urls below.

Kate: http://sienasellbacks.com/kj26fris/lab2/bankCheck.html

* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

#### Part 2 – Practice using CSS

In this part of the lab, you will modify an existing website’s style to conform to new specifications using CSS.

* Download from the Canvas main course page resources the files **selectors.css** and **selectors.html**.
* Modify the CSS code in selectors.css using Notepad++, or any other editor of your choice.
* Look at the #container element in selectors.css. What is the significance of the # sign? **It is an id selector and only applies to the specific id**
* What is the significance of the @media?

 **Changes the style depending on what media/device is viewing the webpage**

* Open selector.html in Chrome and refresh the web page each time you save your CSS changes.
* Alter the CSS code to make the website title and the main navigation more compact:



* + Add your css at the end of selectors.css to override the existing CSS.
	+ Change the text-align of the header so it aligns left instead of center.
	+ Make the h1 and nav inside the header display as inline-block
	+ Make the ul inside the header have a margin of zero all around so it is as compact as possible
	+ Make the h1 inside the header have a left margin of 30px and zero on the other three sides
* Move the side menu and content to the left of the main content as show below.



* + Read about the CSS float and the CSS clear properties.
	+ Make .side\_content elements float left and clear left, and set the left margin to zero.
	+ Make the #main\_content element float right.
	+ Make the footer clear both, so it will always be below the elements above it that are floating left or right.
* After following the instructions, your page will look like this:



* The #main\_content is positioned below the side menu. This is because the #main\_content element is defined after the two .side\_content elements. When we "clear" the .side\_content **section**, the #main\_content **div** moves down even though there is space above it. When using float, content defined below an element cannot not be displayed above that element.
* While we could use absolute positioning to fix this, a better solution is to define the .side\_content **section** after the #main\_content div in the HTML document.
* Open selectors.html in an editor and move the side content section so that it is defined below the #main\_content
* In general, an HTML document should define content in order of importance from top to bottom. In this case, the **side menu** might link to other pages, so consider it more important than the #main\_content. But the .side\_content **section** is less important than the #main\_content, so it should be defined afterwards in the HTML document.
* Use CSS to create the 3-column layout shown below but only if the browser's width is 1000px or greater:



* + Use the existing @media code as an example. Copy and paste it to create a new media query that does the following
		- If the min-width is 1000px, then...
		- Make the #container width 950px
		- Make .side\_content float left and set clear to none
		- Make #main\_content float left
		- Give the section.side\_content left margin of 30px, so it is not close to the #main\_content. Note that you do not want to add left margin to all .side\_content elements just the side content section, i.e., section.side\_content. Note that the selector does not have a space.
* By defining the side menu, main content and side content in order of importance, we can float these elements to the left to create a 3-column layout where the most important elements are on the left and the least important are on the right. Alternatively, we can float these elements to the right to reverse the horizontal ordering.
* Notice that the conclusion section and footer touch each other even though the footer has top margin. This is because the #main\_content floats left and floating elements can occupy the margin of block elements such as headers and footers.
* To fix this, add 30px of bottom margin to #main\_content. Block elements will "respect" the margin of floating elements.
* Notice that we use a "special" class and a "unique" id to identify list items. These can be used to apply style to specific elements.
* Use CSS to make the "special" class have bold text no matter where it is used.
* Use CSS to make the "special" class have underlined text but only if the element itself is a **paragraph**, i.e., p tag. Hint: the selector p .special has a space in it, so it is really specifying two elements, a p and an element with the class name special. If you leave out the space, i.e. p.special, you are specifying one element, i.e., a paragraph with class name special.
* Use CSS to make the "special" class have red text but only if the element is inside of a section element.
* Use CSS to make the "unique" id have a text-shadow that is gray with a blur radius of 5px and a vertical/horizontal position of 3px 3px.
* The website should now look like this:



* Verify that your CSS code, i.e., selectors.css validates at: [jigsaw.w3.org/css-validator](https://jigsaw.w3.org/css-validator/#validate_by_upload)
Then, show your instructor your completed valid document. Note that the website should have a responsive three column design. When the browser is 1000px wide or greater, there should be three columns. When the browser is less than 800px wide, there should be one column. Otherwise, there should be two columns.
* Once completed, show your webpage for your instructor and have him initial here. If you do not finish during the lab period, then show your webpage by the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

