1 Introduction

This document serves two purposes. The first section provides a high level overview of how the different pieces of technology in web applications relate to each other, and how they relate to the specific framework we will use, the Google Web Toolkit (GWT). The second provides some common suggestions, of the sort you might receive from a team leader or senior developer on your team in industry, and they are designed to make you more productive, and coding less tedious and frustrating.

2 Web Architecture

The original purpose of the world wide web was to store documents in a way that could be accessed from anywhere. These documents sat on a server at a publicly accessible location, and often linked to other documents. HTTP, the Hyper-text Transport Protocol allowed the users' browser to make a request for a document, and then retrieve it. The document format was HTML, The Hyper-text Markup Language and is a plain-text encoding of a document, that includes additional formatting and links to other documents.

As an example if you browse to: http://www.ubc.ca/index.html, you browser first connects to the server and asks for the file: **index.html**. The server responds with the contents of the file, and then looks for the content it needs. In Illustration 1, the next thing it needs is **header.png**.

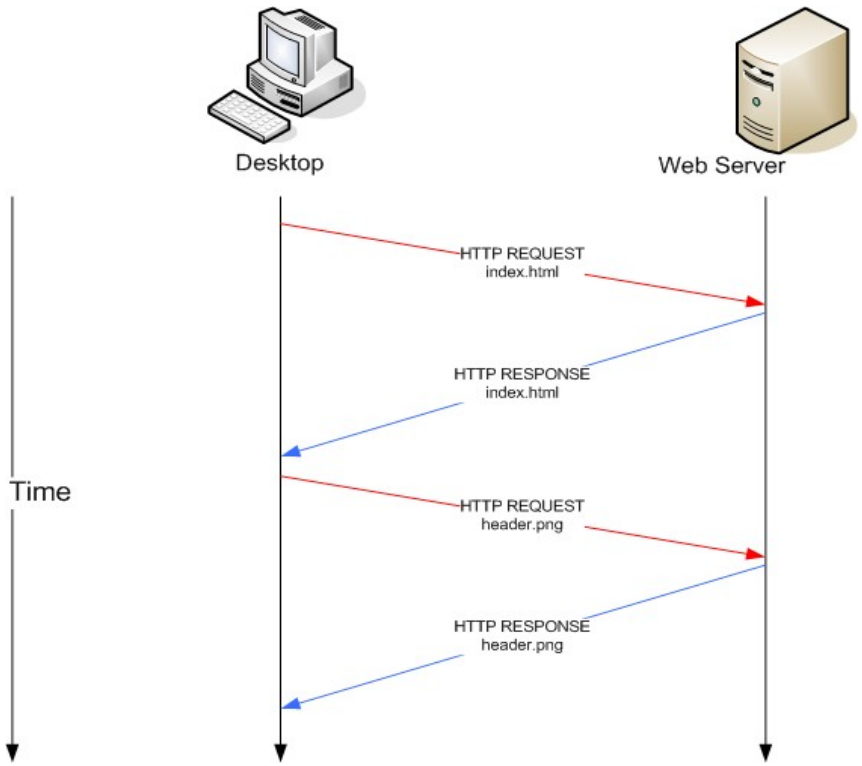

Illustration 1: Retrieving Data from the Server

As the world wide web continued to be developed eventually static documents were replaced by programmatically generated HTML documents whose content depended on the request and other

external state information. These dynamically generated HTML pages allowed for applications to be written that used the browser as a user interface. For instance the UBC Computer Science Groups (https://www.cs.ubc.ca/groups) use this technique for the most part.

Making a request to the server and retrieving an entire HTML page as a response is not very responsive, and so a programming language called JavaScript was developed that allowed for programs to run within the browser. The naming of JavaScript is unfortunate, as it is entirely unrelated to Java[1]. A JavaScript process could manipulate the HTML document directly which allowed for an improved user experience. For instance the user could be notified without contacting the server, that their password did not have the minimum number of characters required.

Still JavaScript applications were limited in that anything that required a server interaction required requesting a new document, effectively killing the previous application and limiting the complexity of JavaScript programs as they needed to be fast as they would only run for maybe a few seconds. For instance early web based chat rooms, operated by automatically refreshing the page every few seconds.

In the early 2000's this changed with the advent of AJAX (purportedly standing for Asynchronous Java and XML). With minor additions to JavaScript functionality offered by browser, JavaScript programs could now directly request data to the server, and process the results. As pages no longer needed to be refreshed, JavaScript applications could offer a much richer user experience.

3 Web Application Development

Web application development is significantly different than traditional applications you may have developed in previous applications. Web development makes some things greatly simpler, for instance formatting something nice is easier in HTML than it is with Swing. But some aspects are also more complex. Web applications follow a client-server model, where the application is broken up into several pieces each of which has it's own state and communicates with the other as needed.

Traditional web application development often suffers from another problem from software engineering perspective, code duplication. Application logic, for instance the length of a password, is often coded twice, once in JavaScript and once in whatever programming language is running on the server. This code duplication is a problem, because it makes the code brittle, and can introduce inconsistencies that are hard to detect. As JavaScript applications have become richer the amount of code duplication has only increased.

The Google Web Toolkit (GWT), the toolkit we will use for the project addresses this by allowing all application logic to be coded once. It then takes part of your application and compiles it into javascript, as the client and this part runs in the users browser. The other part of the application runs on the server, and they communicate by using AJAX.

An additional complication with web development is concurrency, the part of the application that runs on the server, is the same program for all the individual browsers using it at the same time. Managing state in the face of concurrent actions is difficult, but the use of a database allows most web applications to ignore these problems and treat the individual requests as individually without reasoning about others.

---

1 It is often said that Java is to JavaScript, as Ham is to Hamster.

4 Technology Examples

While GWT does a good job of hiding many of the details of the underlying environment the translation isn't perfect and GWT only lessens the burden on the developer to know the underlying environment, it is not quite the same as how you can develop in C and never need to be exposed to Assembly.

HTML:

The Hyper-Text Markup Language is a document format (like PDF, DOC, or ODT) for encoding a document. Unlike most other document formats you are use to, it is designed to be human readable (you can edit the file without any special utilities like Microsoft Word).

The **ex1/** folder contains a simple html document (**index.html**) that you can open and play with in your browser. It is very bare and spartan (see Illustration 2), but contains the logical structure of the document.

For Your Project:

- When you create a GWT project, it will have an associated HTML file. You can edit this file to add more content as needed.
- If you are trying to do something fancy with the User Interface, it is often easier to prototype in HTML than it is to write the code that generates it directly.

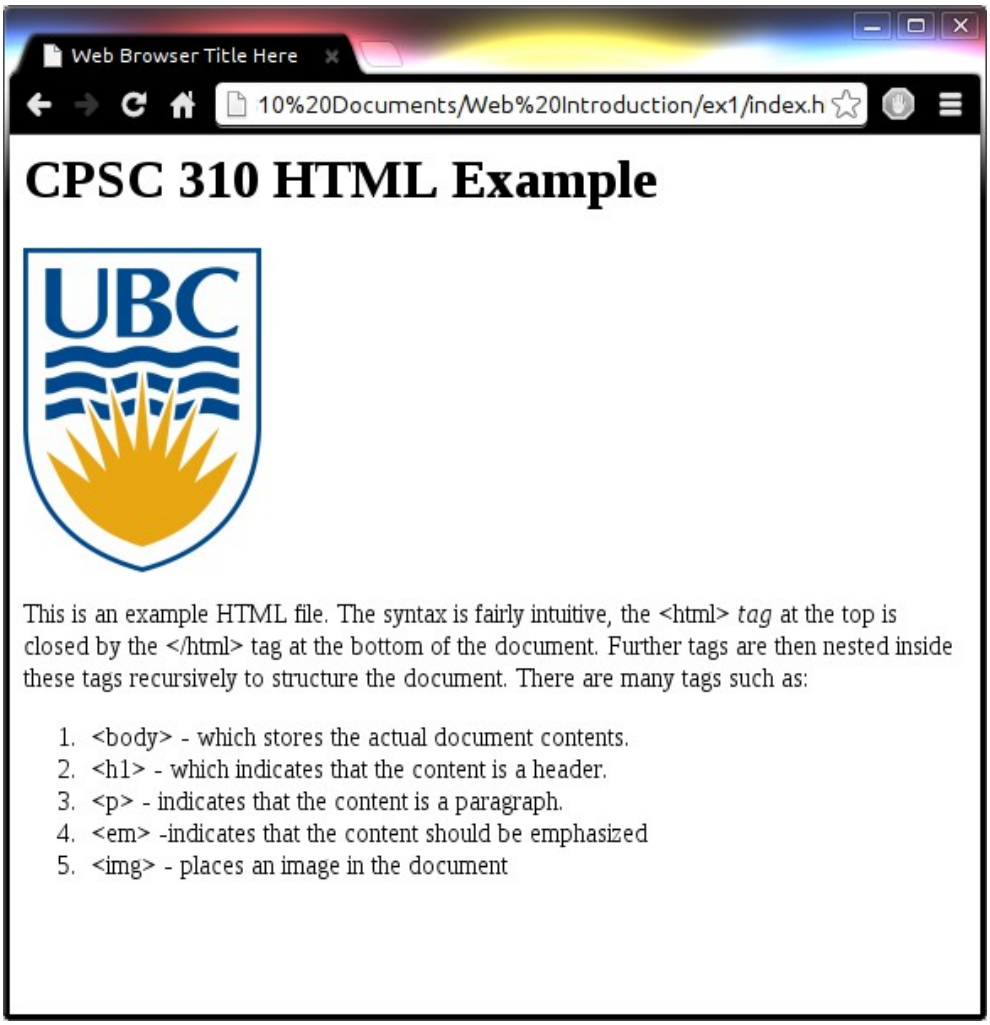

Illustration 2: Bare HTML Document in Browser

Further Reference:

1) W3C Schools HTML Tutorial (http://www.w3schools.com/html/default.asp)

2) Head First HTML with CSS & XHTML (Book)

CSS:

A Cascading Style Sheets (CSS) file is a file that contains visual or presentation information about a document. In the **ex2/** folder our web-page has been modified to reference another file **example.css**. When this file is linked, the look of the page has changed significantly but if you look closely we haven't really changed anything in the HTML file from the previous example.

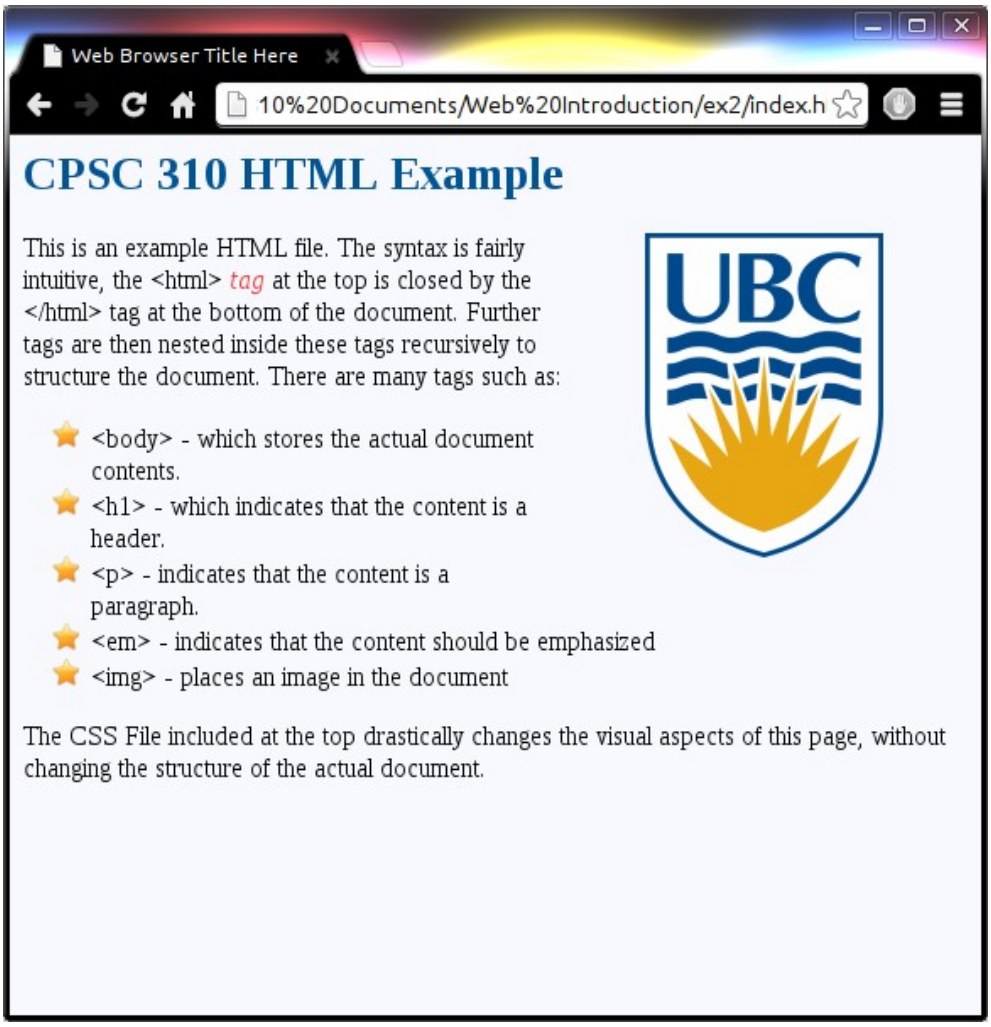

Illustration 3: CSS Styling

Keep In Mind:

CSS support varies between different browsers and within the same browser across versions. A layout that works in one browser may be very broken in another. Quirksmode (http://www.quirksmode.org/css/contents.html) lists the difference between browsers and may prove helpful if you are running into issues with CSS not working in your browser.

For Your Project:

- You can for the most part manipulate the visual element of your site fairly easily in GWT directly using the built in widgets.
- Editing the CSS file of your project is certainly cleaner but by no means required in this project.
- You can create many rules in your CSS file, and then alter the style of your widgets (see the third reference), to create more complicated effects like animations.

Further Reference:

1) W3C Schools CSS Tutorial (http://www.w3schools.com/css/).

2) Head First HTML with CSS & XHTML (Book)

3) Developer's Guide – CSS Style (https://developers.google.com/web-toolkit/doc/latest/DevGuideUiCss)

JavaScript:

JavaScript is the predominant programming language supported in browsers for client side programming, and **ex3**/ contains an example of such a program in the **example.js** file. Now included at the bottom is a text box where you can add elements to the list. JavaScript is very powerful and can build the entire web page programmatically (this is what GWT does for the most part).

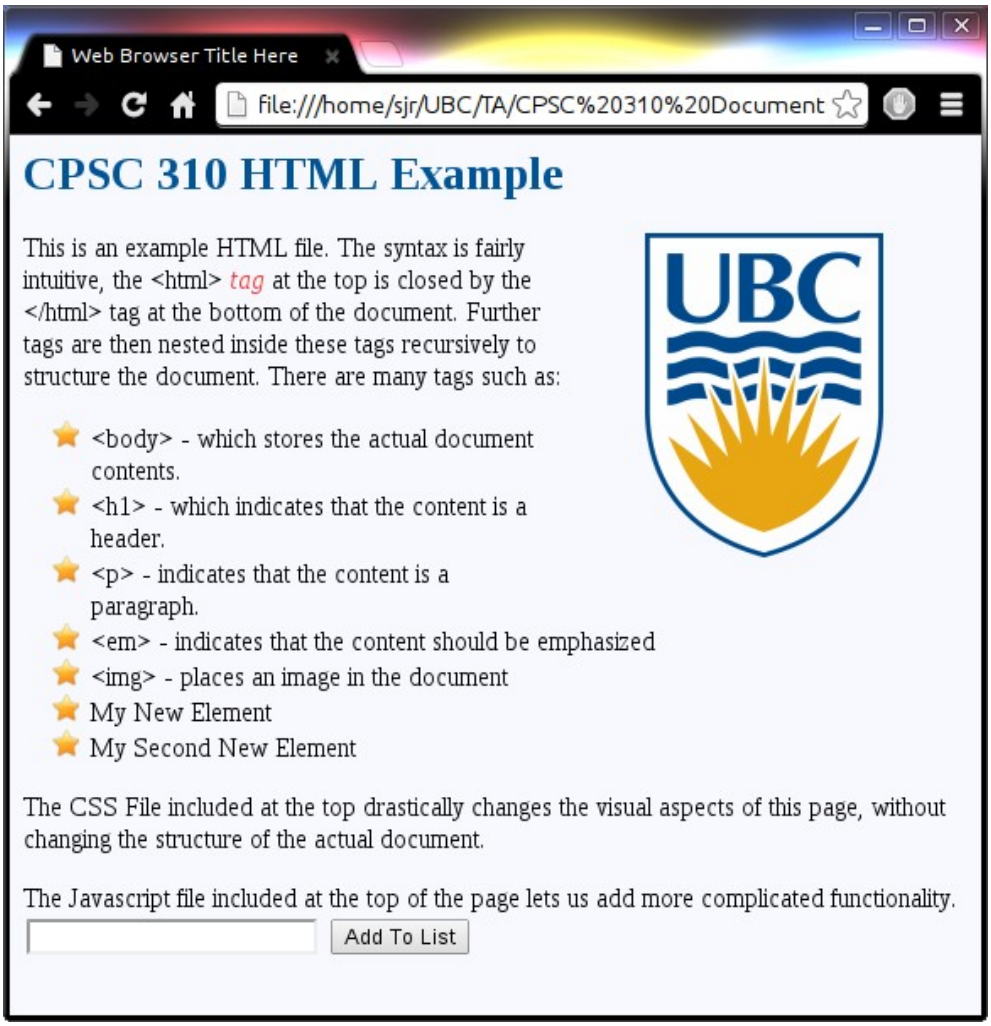

Illustration 4: JavaScript Functionality

Keep In Mind:

Javascript is much better supported across browsers, but one aspect that is weak is support for events like detecting a mouse wheel or the key press event. Quirksmode (http://www.quirksmode.org/dom/events/index.html) lists the differences between browsers and may prove helpful if you are running into issues with some event based code working in your browser.

For Your Project:

Of the three things we have seen so far, JavaScript is the thing you are least likely to interact with directly in your project. It is however the one that influences your project the most.

- When your project is running, it is running as JavaScript and the restrictions on Java support listed here (https://developers.google.com/web-toolkit/doc/latest/DevGuideCodingBasicsCompatibility), are a direct consequence of this.
- JavaScript applications are very limited in what external things they can access. They cannot open files, communicate with other servers, talk to databases, etc...
- Occasionally you may have better luck consulting the Javascript documentation or bug reports for a library such as Google Maps, and then figuring out how to apply them to GWT.
- If you are a already proficient with JavaScript, GWT supports the JSNI which allows invoking JavaScript functions from GWT.

Further Reference:

1) W3C Schools JavaScript Tutorial (http://www.w3schools.com/js/default.asp).

Other Technologies

There are a few other technologies we are using, that you should be familiar with when you see the terms.

Servlet

Loosely speaking a Servlet is a single Java object that accepts requests typically over HTTP. These are handled by Web Servers called Servlet Containers. GWT ships with one when running in Eclipse, and when deployed on the Google App Engine, it runs in a different one.

NOTE: Every Servlet has exactly one instance and all member variables of a Servlet are shared between all users of your web application.

AJAX

Asynchronous JavaScript and XML is the technique that allows JavaScript applications to communicate with the server in the background. In GWT, this is handled by the service and asynchronous service implementations.

NOTE: These Service interfaces are the boundary between the part of your application that runs in an individual users browser and the part of the application that runs on the server and is shared with all clients. The Service implementation is in fact a Servlet.

DOM

Document Object Model refers to the representation of an HTML or XML document in memory as a tree. It is what was used in **ex3/** to add items to the list. In GWT manipulating the HTML document is done through the DOM. The DOM class in GWT (http://google-web-toolkit.googlecode.com/svn/javadoc/2.5/com/google/gwt/user/client/DOM.html), it roughly corresponds to the HTML Document Object Model (http://www.w3schools.com/htmldom/default.asp) and gives you the same access as the JavaScript one http://www.w3schools.com/js/js_ex_dom.asp.