Web Science

Quiz 1: March 6, 2014

100 points max

1. **HTML 5** (25 points): One of the first things we covered was HTML 5
   1. (5) How do we let the browser know that we are using HTML5?

The DTD tells the browser what kind of markup language we are using in the document being presented, so for html5 we use the <!DOCTYPE html> DTD to let the browser know the following code is HTML5

* 1. (5) How does the browser know how to decode the tags properly?

The DTD helps tell the browser which xml file to go to in order to properly decode the html5 tags. So the DOCTYPE associates the html5 standard to the DTD.

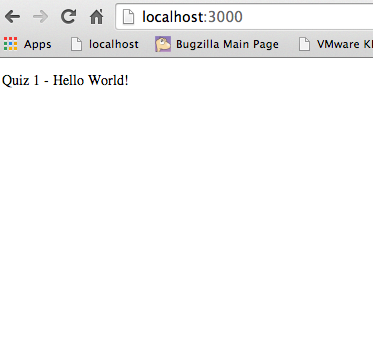
* 1. (10) How would we represent a thematic grouping of content in an HTML document?

Specifically we could contain the content within a specific element or tag for the document, for example if we are writing an article of new for the web we can contain the data within a <article> tag to let the browser know that the content is being rendered as an article. It helps to add semantic data to the document.

* 1. (5) Describe the difference between cookies and local storage

The difference between cookies and local storage is that local storage is primarily serving client-side uses and cookies are normally used for storing server-side information. An example of this is storing information like user login password and user ID’s, in this case a cookie would be used to store the content. Local storage would be used to store things like images, or some content of the web application to help increase load times.

1. **node.js** : (25 points) Create a webserver in node.js which will output a simple html page when a request is received on http://localhost:port/



//NOTE the beginning of each line consists of a capital letter, this is Word’s

//problem not mine, sorry.

Var http = require(‘http’);

Var fs = require(‘fs’);

Fs.readFile(‘index.html’, function(error, html){

If (error) {

Throw error;

}

Else {

http.createServer(function(request, response){

response.writeHead(200, { ‘Content-type’: ‘text/html’});

response.write(html);

response.end();

}).listen(3000);

}

});

//to show the console that we are in fact serving the content on the server:

Console.log(‘server running on port 3000’);

1. the **MVC model** (25 points)
   1. (5) What is it?

A description of web applications built into 3 layers, one for the model, pertaining the business logics, one pertaining the view containing the presentation of the web application, and on pertaining to the controller, containing the user requests and controlling the business objects to fulfill these requests.

* 1. (5) Why do we use it?

We use it for several key purposes:

Separation of data from presentation layer

Ability to have multiple views of data

Ability to have multiple interfaces of the same application

It is less painful to change the data layer

* 1. (10) Describe, and give an example of the View layer?

HTML is a perfect example of the view layer, it encodes the data we want to represent in a simple fashion that browsers can then present to clients.

* 1. (5) Name an MVC framework other than Struts that we covered in class.

Ruby on Rails is another example of an MVC framework because it controls the three components view, model and controller. So when the browser wants to connect to a web application run through ruby on rails, it posts a request to a router, the router discovers the controller, and executes the Rails code the controller then calls the model to get the output the controller then sends the output to the view application layer and the view layer renders the user interface, this is all driven on the ruby on rails code.

1. APIs (10 points)
   1. (5) What are APIs and why do we use them?

API’s are application programming interfaces, basically it allows other applications to access information and data generated or held by another web application. We use them because we as developers do not want to re-develop entire applications just to generate some data that another platform has already done. It helps to develop a platform of apps that can all communicate together using these structures.

* 1. (5) Give an example of a Web APi

An example of a web API is Twitter’s API version 1.1, or the Facebook Graph API, these both allow other applications access to data and information contained on both Twitter and Facebook. They both are ReSTfull API’s in the sense that they follow the same 6 constraints for representation State Transfer, which is the architecture for essentially all web API’s.

1. (15) Explain *in detail* what the following code does;



The above code will first tell node.js that the below lines of code will require a certain package, the net package, and loads that package into a variable called net. Then an array called sockets is initialized. The next section first creates a server on the net package and pushes the socket object (i.e., the location of the person who is accessing the server) to the sockets array (add the location of each person who joins the server to an array). The server then accepts an argument “data”, aka whatever the terminal on the socket types in and writes that data to all the sockets on the net package server. This will essentially create a very basic chat application. The last segment of code will remove the socket that terminates its connection with the net server from the array of sockets so that the net server will not have to worry about trying to write the data to a user that does not exist on the chat network server anymore. The final line actually starts the server itself, starting a net package server on localhost:8000.