Part One Understanding the Client Server relationship

One constant in web development is the relationship between the client which is usually the web browser you are using to go to your favorite sites, and the server, or a computer running a program that listens for incoming connections, and serves web pages, and all their resources like images, javascript, and css back to the client.

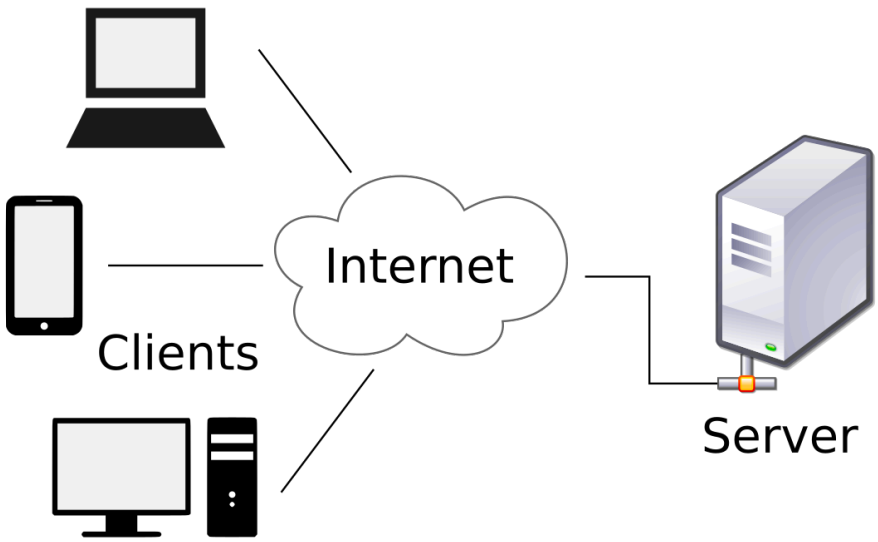
Now there are different protocols that can be used to communicate between the server, and the client, but the most commonly used for websites, and the one we will explore today is HTTP or hyper text transfer protocol. This is how modern web browsers usually communicate with servers, and all it is is a standard for how to encode messages in text.

These messages consist of two main parts, a header, and a body.

The header has metadata like a time stamp, how large the message being sent is, but more importantly for us it also contains the path requested, for instance if you go to http://sprite-creator.com/web your browser is actually adding that "/web" to the header in a field that allows us to check what page on our website was actually requested so we can send a different page for each path, as well as some information about the type of the http request which fall under four categories:

1. Get requests - These are usually sent from the client to the server to request a web page, but can also be used to ask a web server for data in other forms like JSON, or XML, as well as sending data unencrypted to the web server.
2. Post requests - These requests are used to send some information from the client to the server usually, and if you use TLS also called HTTPS then the body of the message where you send all the data for a post request will be encrypted this makes it ideal for sending sensitive information
3. Put requests - These requests are much like post requests, but are supposed to support updating information that already exists instead of just sending new information, frequently used to update databases
4. Delete requests - These requests are used to request that some data stored on the web server, or that the web server has access to like a database be deleted

In addition there are also status codes sent in the HTTP header, if you've ever seen error 404 not found or 500 internal server error you already have some experience with two of the most important status codes that let you know when something has gone wrong. Perhaps the other two most important codes are 200 for success, and 304 which is a request the server will send back to a web browser to say this file hasn't been changed so you should be able to use the previous version I already sent you.



What we will be doing in this workshop is taking a library that handles the hyper text transfer protocol for us, and helps us to extract information from requests sent by the client, and dynamically generate responses using our C++ code, but in order to do this we must learn how to use HTML or hyper text markup language to setup a user interface for the user to submit data back to the server, and for that we will look at the HTML form.

Below you will see an example of a simple HTML form:

```
<form action="/option" method="GET">
  <label for="Company">Company:</label><br>
  <input type="text" id="Company" name="Company" value="">
  <input type="submit" value="Submit">
</form>
```

In the first line of the above code we can see two very important fields of the form being set.

First is the action. This is the path on the server the request will be sent when submitted by the user.

Second is the method that will be used to send the request, this is where we specify the type of request we wish to send to the server, as you can see the method is set equal to the string “GET” which means we will be sending a get request to the server. This will be important later while building our web server and setting it up to respond dynamically.

The next portion to analyze are the different items inside the form.
First we have a label as specified by the following code:

```
  <label for=“Company">Company:</label><br>
```

This will simply be used to add a caption for the next field. We put the caption text in between the > < signs.

The next element is an input, and we can see it has a type of “text”, and id, and a name of Company. This is a textbox the contents of which will be sent to the server when the user submits the form, and the server will be able to find the data the user typed into the textbox by the field’s name, in this case “Company”.

```
  <input type="text" id="Company" name="Company" value=“">
```

The final input element has the type “submit” and this specifies it as the button that will send a request to the server with the other input data when pressed by the user.

This is our primary means of communicating between the client, and the server allowing the user to send some data to the server to process, and respond dynamically.

When a request is sent from a form the server is responsible for taking the data sent, and using it to dynamically generate a new page, then send that page to the client making the request. In the next part we will see how this is done.

You can see the final project here:
https://replit.com/@AndrewRubinstei/CSC211FinalProject-Final#main.cpp