The Server

Client <-> Server communication

Week 6

Setting up your environment

- Install node.js
- Make sure you can run a server

Hello Node.Js

- Start server with node server.js
- locahost:3000

Mental Models

- Client or Server programs
  - Server abstracts behavior multiple clients want to utilize
  - i.e. FTP Server + client
  - Client-Server Model Illustration
  - Dev on remote server means longer feedback loop
  - Dev on local server dev reduces feedback loop

Practicalities

- List of dev applications
  - The browser (e.g. Chrome)
  - The text editor (e.g. Sublime)
  - Git
  - Usually a few terminal windows (e.g. iTerm)

Hello HTTP!

- Look at server.js, what is it doing?

```
var http = require("http"),
   server;
server = http.createServer(function (req, res) {
   res.writeHead(200, {"Content-Type": "text/plain"});
   res.end("Hello World!\n");
});
server.listen(3000);
console.log("Server running on port 3000");
```

Modules and Express

- Install express.js http://expressjs.com/
- Create simple express server
- Go through expressjs guide

Examples

- Go through example applications

Homework

Let's take what we have learned and implement a simple counter application. The requirements are as follows:

- As a user
- I want to be able to increment the counter, retrieve its value, and reset it back to 0
- So that I can manage a counter

This will involve implementing the following routes

- POST /counter
  - Should increment a number variable in your webapp by 1
  - Should return a 204 success code
- GET /counter
  - Should return the current value of the number variable
  - Should return a 200 success code
- DELETE /counter
  - Should reset the current value of the number variable to 0
  - Should return a 204 success code