Server-Side

But you said this is more of a front-end class?!
Well... but all the cool kids use NodeJS and I want you to see just how powerful it can be.

Class 5 - 9/28/2016

What's a web server?

We usually think of a web server as a huge computer somewhere, sending pages back as we type an address on the browser. But it can also simply refer to the software serving the pages.

We can have more than one server running in the same machine (on different ports) — we did that with:

```
$ python -m SimpleHTTPServer 8000
$ python -m SimpleHTTPServer 3000
$ python -m SimpleHTTPServer 1234
```

The HTTP Protocol

History

Though implemented by Tim-Berners Lee at CERN, the term *hypertext* was coined by Ted Nelson in his Project Xanadu

Specs

- The HTTP takes a REQUEST and sends back a RESPONSE
- The REQUEST usually includes information such as:
    - HTTP methods: GET, POST (+ PUT and DELETE)
    - url
    - User-agent

So what is Node.js?

Basically, it's a server-side Javascript!

Runs on Google's super powerful V8 JS Engine that compiles the Javascript into low-level code/machine code.

(If you don't know what that means, just understand that it's SUPER FAST.)

A typical procedural program:

1. Initialize database connection. ***wait til that's ready.***
2. Run a query for some rows. ***wait til it actually comes back with some.***
3. With each row, show it. ***wait til each is done before moving on to the next.***
4. End of script, spit out the result to the user!

Basically line 2 cannot be executed until line 1 finishes its task.

Also known as 'synchronous/blocking' world. It's great for total integrity, BUT it doesn't scale well. It's slow.

This is where Node comes in. It's the complete opposite. It's 'asynchronous/non-blocking'. Instead of procedural, it's event-driven.

Wait, it's blocking/non-blocking what exactly?!
I don't understand..

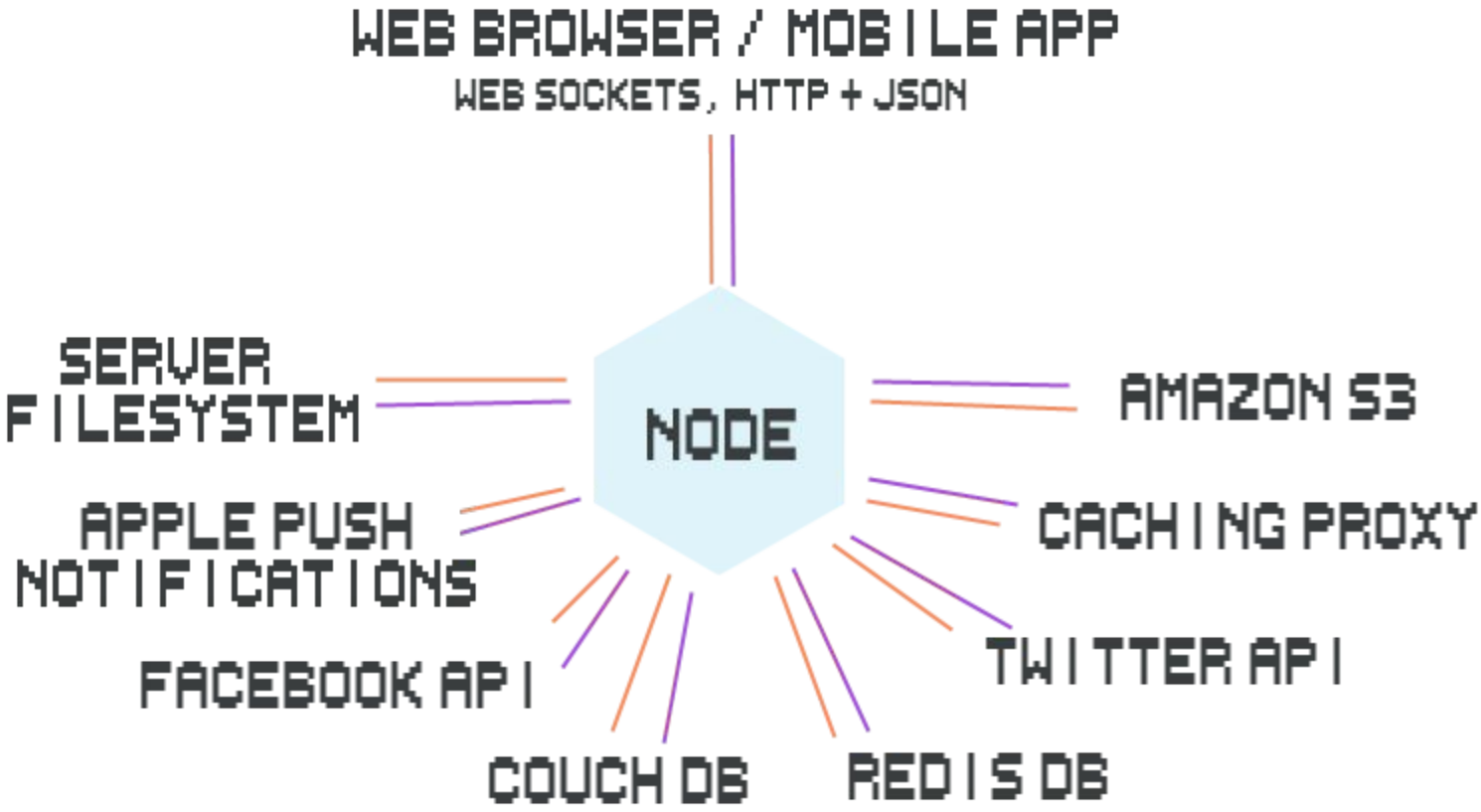
In NodeJS...

- Line 2 cannot depend on line 1, because line 1 might be taking awhile.
- And all the program wants to do is get to the end of the file!

The program would be idle most of the time, waiting for things to happen, before receiving an event and continuing what it needs to do.

What can we do with NodeJS

- Build simple to complex web servers and API servers
- Communicate with databases - CRUD files
- Communicate with device's low-level hardware - sound card, GPU, network configurations, etc
    - Node.js + PComp = IOT!
- Implement websocket protocols
    - Build an IRC chat bots
    - Make collaborative drawing tool
- Scrape websites (for data visualization)


WEB BROWSER / MOBILE APP
WEB SOCKETS, HTTP + JSON
SERVER FILESYSTEM
NODE
AMAZON S3
APPLE PUSH NOTIFICATIONS
CACHING PROXY
FACEBOOK API
TWITTER API
COUCH DB
REDIS DB

What Node is NOT:

- A programming language. (We the write the same exact JS!)
- A web framework.

How it works

Node.js **require**s modules, as Processing's **import** or C++ (oF)'s **include**.

- Node.js

```
var http = require('http');
```

- Processing

```
import http.requests.*;
```

- C++ (oF)

```
#include "ofxHttpUtils.h"
```

Glossary for this week & next week

- NodeJS is ... everything that we just talked about.
- **npm** is the package manager for Javascript. It makes it easy for JavaScript developers to share and reuse code.
- **Express** is a framework for building web applications on top of Node.js. It simplifies the server creation process that is already available in Node.

Glossary for this week & next week

- **MongoDB** is a database. This is the place where you store information for your web apps.
- **CRUD** is an acronym for Create, Read, Update and Delete. It is a set of operations we get servers to execute (POST, GET, PUT and DELETE respectively). This is what each operation does:
  - Create (POST) - Make something
  - Read (GET)_- Get something
  - Update (PUT) - Change something
  - Delete (DELETE)- Remove something

Node.js 101 --- Starting a New Project

- Create a new folder for your Node project
- Open terminal and:
  - `cd your/project/folder`
  - `npm init`

    // Initializes a project, creating a `package.json` file that will store our project information and list its dependencies
  - `npm install some-node-module another-module --save`

    // You might need "`sudo`" before those npm, depending on your permissions

    // `--save` is specifying that the modules should be registered as dependencies in you

    // After this command, you should be able to see a `node_modules` folder

    Example: `npm install express --save`

Node.js 101 --- Keep it compact on Github

If you're using git, you should **gitignore** the **node_modules** folder!!!

— you can Google that info

Node.js 101 --- Running a Project

- Assuming that you have an app.js (or whatever name you choose) file, type `node app.js`
- CTRL+C to stop a running server
- If you don't want to stop and run the server again everytime you make a change, use nodemon
- After installing, you'll simply type `nodemon app.js` — instead of `node app.js`

Node.js 101 --- Running a Project You Downloaded From Someone's Github

- Clone the repository first or simply download the .zip
- Open Terminal and navigate to the project directory - `cd some/project/folder`
- Type `npm install`
  // npm will read the dependencies listed on **package.json** and
  // automatically install them for you
- Run `node app.js`

Random npm

- Cleverbot https://cleverbot.io/
- Node oxford emotion https://www.npmjs.com/package/node-oxford-emotion
- Speed Test https://github.com/sindresorhus/speed-test

- https://node.cool/

What is web scraping?

The process of crawling onto websites, to extract information.

Why scrape the web?

- Get real time data from the source
- Many sites don’t offer API
- APIs can be difficult to work with
  - Can be expensive (both time and effort)
  - Generally lack of support and documentation
  - Dealing with rate limits...

Scraping with Node.js

- PhantomJS or CasperJS
- Cheerio or Request
- X-ray

Resources

- http://nodeschool.io/
- https://node.cool

- The art of Node https://github.com/maxogden/art-of-node/#the-art-of-node