Welcome to Server Side Development!

Re:Coded - Server Side Development Workshop

A bit about me...


PETZL

STEVE

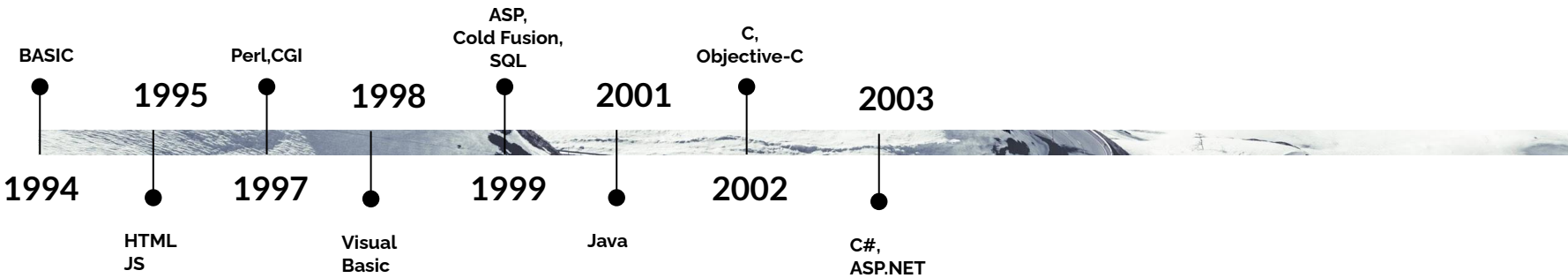
Wright

Google

LAWN

GOOGLER

My Programming Journey



A bit about the workshop...

Schedule

- Day 1:
  - Introductions
  - General web concepts
  - Node.js and Express
- Day 2:
  - Templating Engines
  - AJAX & Web API Design
- Day 3:
  - Authentication
  - Intro to CRUD Operations in SQL
- Day 4:
  - Complex SQL
- Day 5
  - Review
  - Follow-Up Project Discussion
  - Group Kickoffs

A quick survey

Questions for me?

Getting Started

- GitHub:
  - If you do not have a Github account, please create one!
- Checkout Repository:
  - Private message me your github username and I will add you as a collaborator on:
  - https://github.com/StevenEWright/recoded-sept-2020
  - Once that's done, please clone the repository locally.
  - git clone https://github.com/StevenEWright/recoded-sept-2020
- Node.js:
  - Please install Node.js from https://nodejs.dev/
- DB Browser for SQLite
  - Please install DB Browser for SQLite from https://sqlitebrowser.org/dl/

Day 1:
Hello Node.js & Express!

Re:Coded - Server Side Development Workshop

What happens when you click a link?

CRASH COURSE COMPUTER SCIENCE

29

THE INTERNET

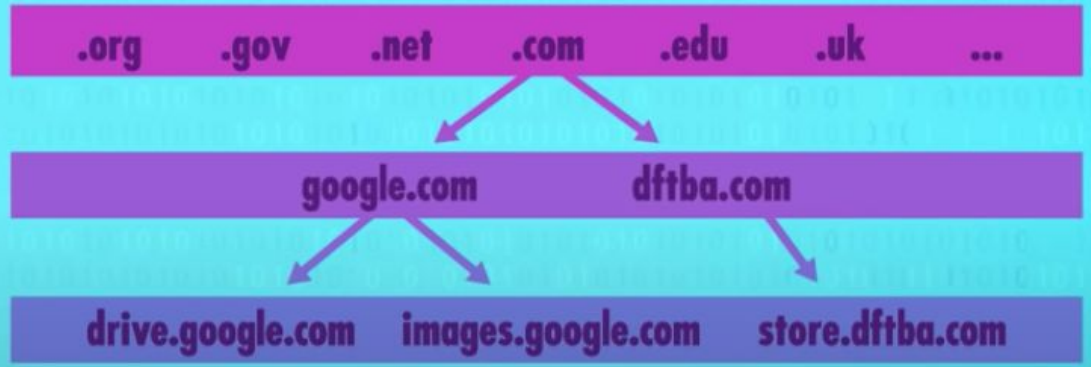
COMPUTER NETWORKS

DOMAIN STRUCTURE

TOP LEVEL DOMAINS

SECOND LEVEL DOMAINS

SUB-DOMAIN OF PARENT



LAYERS OF THE OSI MODEL

(OPEN SYSTEM INTERCONNECTION)

Common TCP/IP Ports

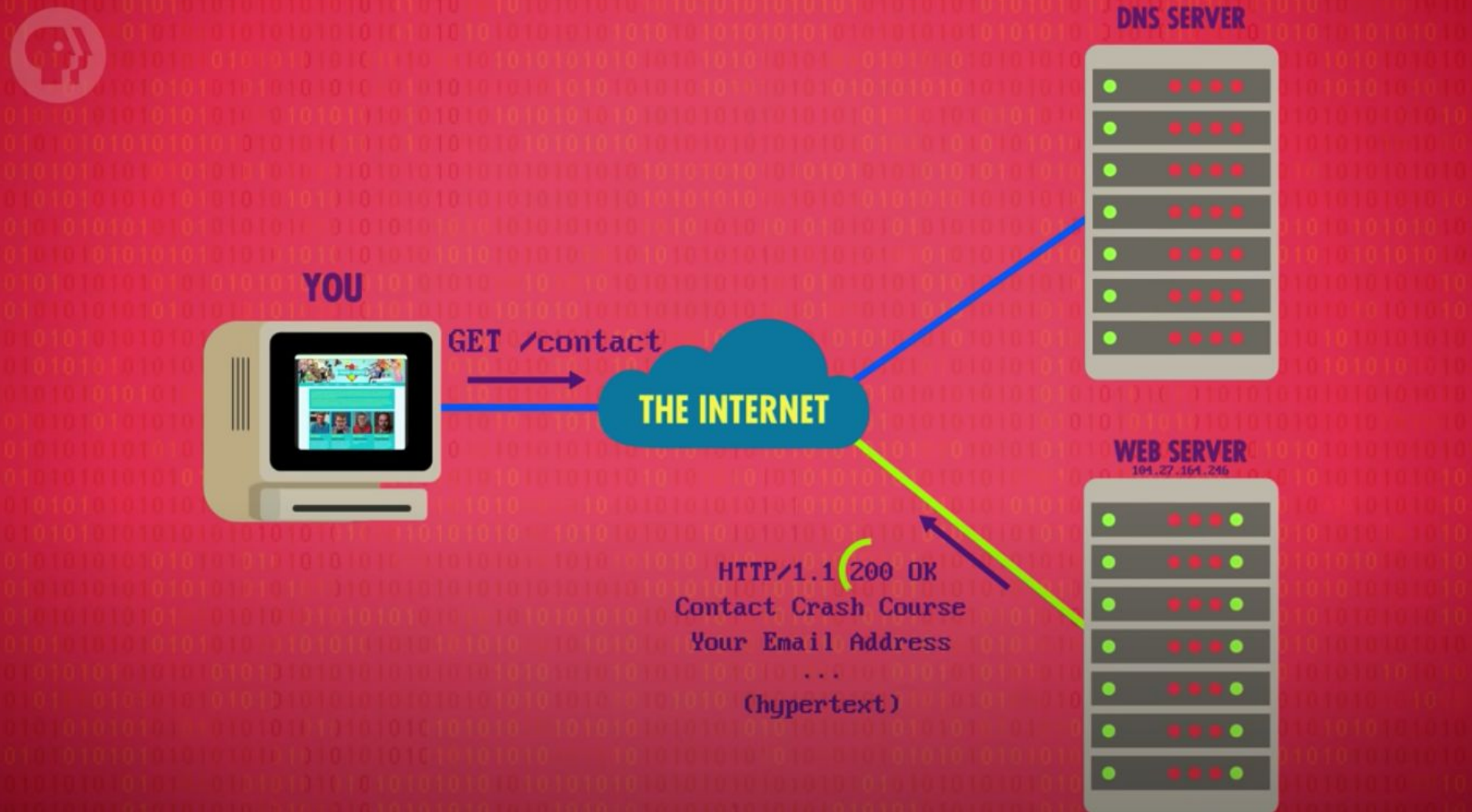
- 80: HTTP
- 443: HTTPS
- 53: DNS
- 25: SMTP
- 20/21: FTP

CRASH COURSE COMPUTER SCIENCE

30

WORLD WIDE WEB

THE WORLD WIDE WEB


DNS SERVER
YOU
GET /contact
THE INTERNET
WEB SERVER
104.27.164.246
HTTP/1.1 200 OK
Contact Crash Course
Your Email Address
...
(hypertext)

Anatomy of a Uniform Resource Locator

https://www.google.com:443/feeling/lucky?q=easter+eggs&r=1#first

A Uniform Resource Locator

Anatomy of a Uniform Resource Locator

https://www.google.com:443/feeling/lucky?q=easter+eggs&r=1#first

Scheme

Host

Port

Path

Query

Fragment

A Uniform Resource Locator

Anatomy of an HTTP Request

```
POST / HTTP/2
Content-Type: Application/JSON

{ "awesome": "yes" }
```

A Uniform Resource Locator

Anatomy of an HTTP Request

```
POST / HTTP/2
Content-Type: Application/JSON

{ "awesome": "yes" }
```

HTTP Verb

Request Path

HTTP Version

Header

Body

A Uniform Resource Locator

```
GET / HTTP/2
Host: www.google.com
User-Agent: curl/7.64.1
Accept: */*
```

```
POST /posts/upvotes HTTP/1.1
Host: localhost:3000
User-Agent: curl/7.64.1
Accept: */*
Content-Type: application/json
Content-Length: 18

{"upvoted":"true"}
```

HTTP Status Codes

- 1xx informational response
- 2xx success
- 3xx redirection
- 4xx client errors
- 5xx server errors

More details: https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

Common HTTP Headers

- Accept
- Content-Type
- Content-Length
- Cookie
- Cache-Control

Common Content Types

- application/x-www-form-urlencoded
- multipart/form-data
- text/plain
- application/octet-stream
- text/html
- text/javascript
- image/jpeg

Why do we need software to run on servers? Why not do everything in JS?

What's the difference between a Web Server and a Web Application?

Break

Hello World in Node.js

What is Node.js?

What's the difference between a Javascript Engine and a Javascript Runtime Environment?

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

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Hello World in Node.js

```
node debug app.js
```

Starting our application with the debugger

```
about:inspect
```


Connecting with Chrome DevTools

Where do we go from here?

Hello World in Express

Express - Hello World

```
const express = require('express')
const app = express()
const port = 3000

app.get('/', (req, res) => {
  res.send('Hello World!')
})

app.listen(port, () => {
  console.log(`Example app listening at http://localhost:${port}`)
})
```

```
{
  "name": "express_hello_world",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.1"
  }
}
```

Package.json

```
npm install
npm start
```


Starting our app

```
node debug ./bin/www
```


Starting our application with the debugger

Static Content

Express - Site Generator

```
npx express-generator -ejs bulletin_board
```

Where do we go from here?

See you tomorrow!

Stay for Q&A!