GITHUB AND CONTINUOUS INTEGRATION

THE NEED FOR CI

HTML / CSS / JS

PHP / MYSQL

NODE

THE NEED FOR CI

Risks Within Software Development

1. Fixing Bugs late can be costly
2. Lack of Team Cohesion
    1. “Your changes are incompatible with mine, how do we merge...”
    2. When did we decide to upgrade to v2?
    3. I thought that bug was fixed
3. Poor code quality
    1. We have 3 implementations of the same thing
4. Lack of project visibility
    1. What do you mean all the tests are failing
    2. Whats is v1.2.3 of the code?
5. Lack of deployable software
    1. It worked on my machine
    2. The customer is coming...we need a demo now!

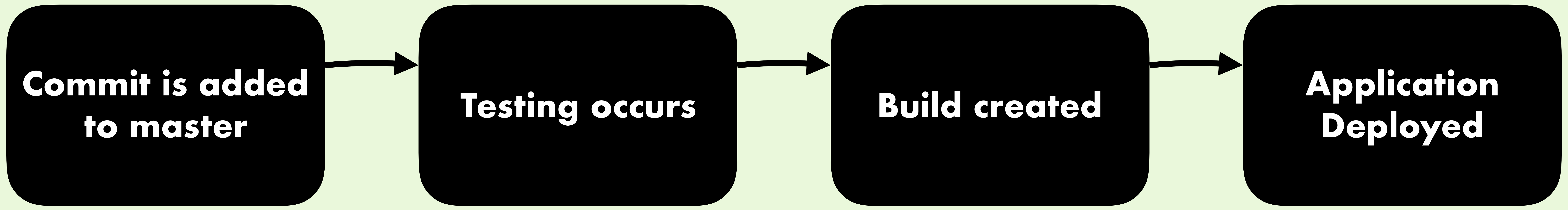
Better, Faster, Cheaper

- Better…
    - Build quality
    - Testing that happens early and often
    - Code from best practice and code standard
- Faster…
    - Test in parallel, not just at the end
    - Builds are a non-event
- Cheaper
    - Identify defects earlier
    - Fix when least costly

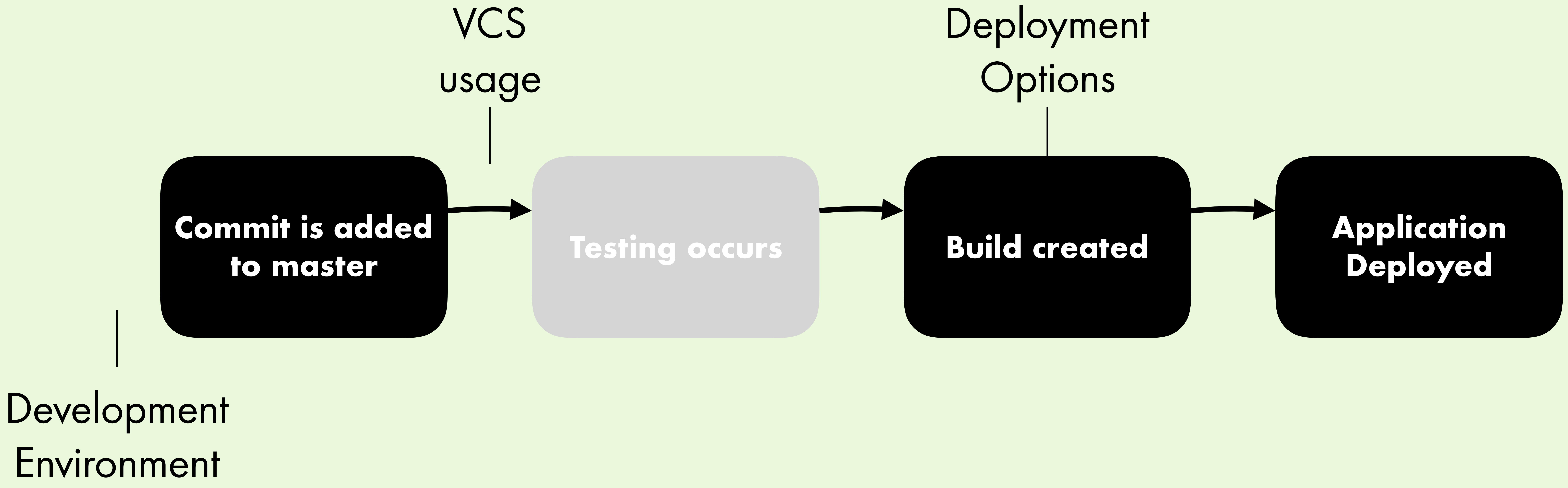
7 Steps to using CI

1. Commit early, and commit often
2. Never commit broken code
3. Fix build failures immediately
4. Fail fast
5. Act on metrics
6. Build in EVERY target environment
7. Create artefacts from ever build

Full CI Pathway



What we are covering today



Development Environment

- Personal computer setup
- Used as a base for where you will be creating code
- Should mimic real world deployment as close as possible
- Gives developer tools to carry out code creation

VCS Usage

- Covered last week
- Focus on scoping project through gitHub issues
- Use **project** and **automated KanBan** as a way to monitor project progress
- Never commit directly to master!
  - Every time you do one of Santas elves suffers.
- Focus on collaboration!

Deployment Options

- Many exist, and are very easy to move between
- Option you choose really depends on the technologies that you are using
- Most have free tiers (especially for students!)
- If something is “free”...don’t put your credit card details in!

HTML / CSS / JS

Development Environment

- Many different **IDEs** exist that you can use
  - Choice of these is really up to you too, but explore before picking one

- **Development Environment** for creating HTML / CSS / JS applications is very straight forward.
  - Key challenge is in running local js files for testing
  - You'll need a local HTTP server to do this

- Find a way to manage VCS integration
  - Some IDEs deal with this
  - Others require command line

Development Environment
IDE Choices

- Atom
  - I use this quite a lot, very simple structure but has a lot of expandable modules
- Visual Studio Code
  - Haven't used this but have heard very good thing about it. Built on the same platform as Atom (electron) and also has good gitHub integration
  - If you are beginning I would recommend this over Atom (Microsoft owns GitHub which owns Atom...)
- Web Storm
  - Commercial application (similar to IntelliJ), very powerful and with lots of features

Development Environment

Development Environment

- The HTTP Server
  - There is a HTTP server package within npm that can be used
  - https://www.npmjs.com/package/http-server
  - Requires node to be installed

localhost:8080
localhost:8080
Serving up static files like they were turtles strapped to rockets.
V8

Development Environment

Development Environment

- Jekyll Environment
    - Jekyll lets you create a static blog website and has a number of benefits.
    - Has a local server that can be used
        - https://jekyllrb.com/

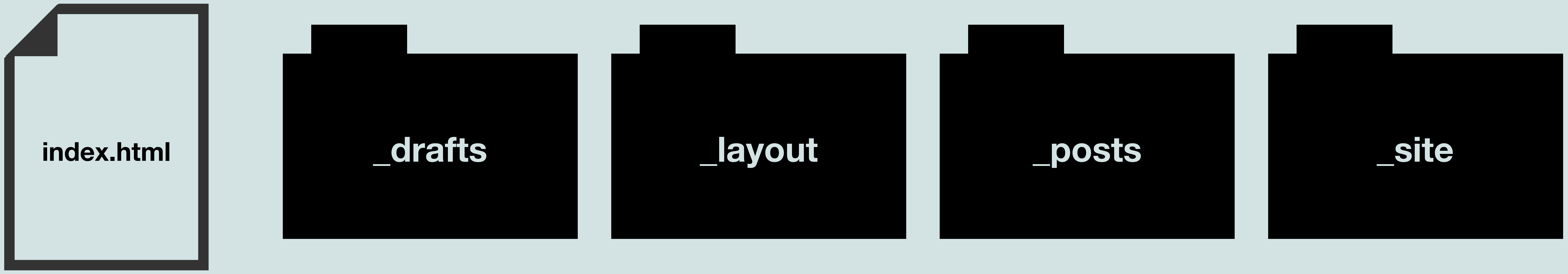
Development Environment

Deployment Options

- GitHub Pages is the main deployment option here
- Takes a repository and turns it into a webpage
- Can be expanded though Jekyll/Liquid to create page templates
    - GitHub Pages renders jekyll/liquid automatically

Development Environment

Jekyll Project Structure



STORES UNPUBLISHED PAGES (OR BLOG POSTS)

PAGE TEMPLATES

CONTAINS PUBLISHED POSTS (FOR BLOGS…)

WILL CONTAIN YOUR WEBSITE ONCE IT HAS BEEN RENDERED (CAN IGNORE THIS)

Development Environment

Jekyll Project Structure

_includes

CAN BE USED TO CREATE MODULAR COMPONENTS THAT YOU CAN THEN USE (E.G. HEADER).

```
{% include header.html %}
```

_sass

IF YOU WANT TO DO MORE ADVENTUROUS CSS THINGS THEN YOU CAN PLACE .SCSS FILES IN HERE TO BE COMPILED TO .CSS FILES. MORE ON SASS IS AVAILABLE HERE:

https://sass-lang.com/guide

Development Environment

Jekyll Project Structure

```
---
layout: default
title: "Home"
fulltitle: "UX-D Website"
type: "Main"
---

<main>
<h1> Hello World </h1>
<p>This is the
{{page.title}} Page </p>
</main>
```

index.html

- You'll have something like this at the top of every page
    - Stores variables that you can access (both in this page and in other places)
    - Sets what template you want to use
- Main page content goes underneath

Development Environment

Jekyll Project Structure

```
<!DOCTYPE html>

<!-- header goes here. -->

{{ content }}

<!-- footer goes here. -->
```

_layout > default.html

- This is what a standard layout page might look like
    - ...obviously you would have something a bit more adventurous than just commenting

Development Environment

Jekyll Project Structure

- Can also do more adventurous things, this creates a list of all pages in your website and puts them in a <ul>
  - Great for automatically generated menu structure

```
<ul class="navbar-nav ml-auto”>
{% assign sorted_pages = site.pages | sort:"order" %}
{% for node in sorted_pages %}
     <li class="nav-item {% if page.url == node.url %} active {% endif %}”>
          <a class="nav-link" href=“{{node.url}}">{{node.title}}</a>
     </li>
{% endfor %}
</ul>
```

Development Environment

Jekyll Project Structure

- Locally, jekyll will render your webpages and show you the resultant files within the _site directory
- When uploaded, they will be rendered and help by gitHub pages
  - You shouldn’t see the resultant files.

demo

Jekyll Site Structure

https://www.ux-d.co.uk/

PHP / MYSQL

Development Environment

- Similar to before…many different IDEs exist that you can use
  - Choice of these is really up to you too, but explore before picking one

- **Development Environment** for creating PHP / mySQL applications takes some work to get set up
  - Think about replicating your server setup
    - Are you using apache or ngineX?
    - What version of mySQL?

Development Environment

IDE Choices

- phpStorm
  - Very powerful editor, lots of nice features but can be a bit bloated
  - Difficult to get all features working in local environment with shared desktop (ie here)
- Atom
  - Again, I use this pretty much exclusively
  - Can download plugins to assist with php code completion and syntax highlighting

Development Environment

Local Development Environment

- A local development environment is a MUST. Most OS don’t support PHP out of the box

- MAMP
  - Application to allow development of php/mySQL applications in Mac environment
  - MAMP (for windows) also exists
  - https://www.mamp.info/en/
- XAMPP
  - Another one that is used often
  - https://www.apachefriends.org/index.html

Development Environment

Deployment Options

- Microsoft Azure has an App Service called Web App + MySQL
- An option inside of this hooks into a gitHub repository and automatically updates the web app based on contents of a particular branch
- mySQL is delivered through an inApp option that can be accessed through phpMyAdmin.

- Database connection strings will be different for local development and final deployment
    - Use .gitignore to specify not to copy your dbconnect.php script
    - Code for connection script for WebApp + MySQL is on next page

```
// MYSQL IN APP apache_get_version

$connectstr_dbhost = '';
$connectstr_dbname = '';
$connectstr_dbusername = '';
$connectstr_dbpassword = '';

foreach ($_SERVER as $key => $value) {
    if (strpos($key, "MYSQLCONNSTR_localdb") !== 0) {
        continue;
    }

    $connectstr_dbhost = preg_replace("/^.*Data Source=(.+?);.*$/", "\\1", $value);
    $connectstr_dbname = preg_replace("/^.*Database=(.+?);.*$/", "\\1", $value);
    $connectstr_dbusername = preg_replace("/^.*User Id=(.+?);.*$/", "\\1", $value);
    $connectstr_dbpassword = preg_replace("/^.*Password=(.+?)$/", "\\1", $value);
}

$db = mysqli_connect($connectstr_dbhost, $connectstr_dbusername, $connectstr_dbpassword,$connectstr_dbname);

if (!$db) {
    echo "Error: Unable to connect to MySQL." . PHP_EOL;
    echo "Debugging errno: " . mysqli_connect_errno() . PHP_EOL;
    echo "Debugging error: " . mysqli_connect_error() . PHP_EOL;
    exit;
}
```

https://www.php.net/manual/en/function.preg-replace.php

Development Environment

Deployment Options

- Most of the time that I am using PHP + mySQL is to create a web API that I can then interact with using javascript

- Many different options for how to do this, my preferred way is:
  - Redirect every page to the root index.html page
  - Parse the address bar and `include’ the correct page based on what the user is attempting to do

- The challenge here is in the redirect:
  - With apache this requires a .htaccess file
  - With ngineX this requires a web.config file
- MAMP is apache by default, Azure is nginеX by default

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <system.webServer>
        <rewrite>
        <rules>

            <rule name="allToIndex">
                <match url="^((?!css).)*$"/>
                <action type="Rewrite" url="index.php" appendQueryString="true"/>
            </rule>
        </rules>
        </rewrite>
    </system.webServer>
</configuration>
```

web.config

```
RewriteEngine On
RewriteRule ^inc/.*$ index.php
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ index.php [QSA,L]
```

.htaccess

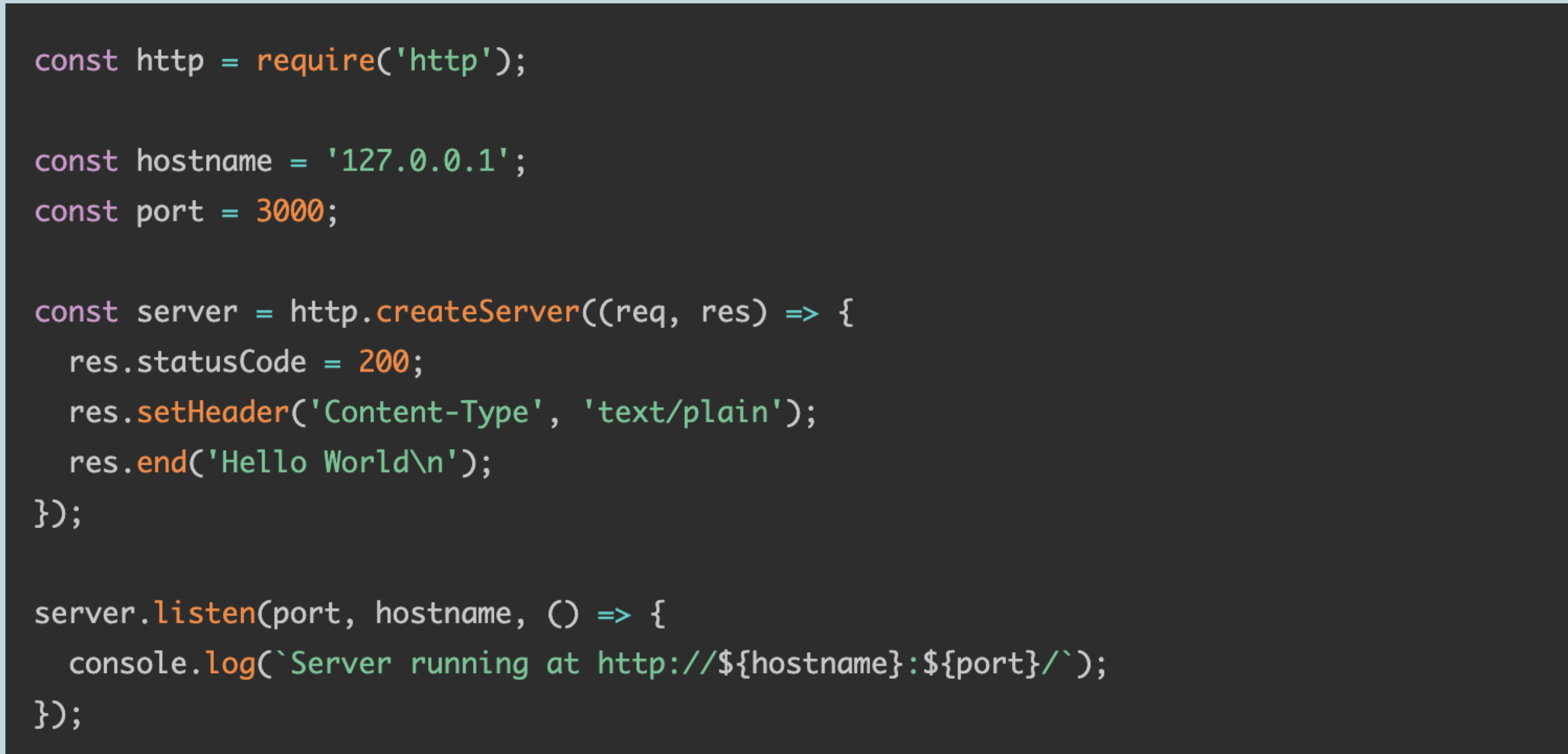
demo

PHP + mySQL

NODE

What is Node?

- As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.
- In the following "hello world" example, many connections can be handled concurrently.
- Upon each connection, the callback is fired, but if there is no work to be done, Node.js will sleep.



```
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\n');
});

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

What is Node?

- Many web applications are now built on top of node and use a number of additional javascript libraries to assist in development

- Common structure is the MEAN, MERN, and MEVN stacks
  - MEAN = Mongo / Express / Angular / Node
  - MERN = Mongo / Express / React / Node
  - MEVN = Mongo / Express / Vue / Node

Development Environment

Local Development Environment

- First thing that has to be installed is Node
    - https://nodejs.org/en/

- Node applications are configured within a file called package.json. You will need a package.json file for each project you create.
    - This file is where you configure the name of your project, versions, repository, author, and the all important dependencies.

```
{
  "name": “introduction-to-node“,
  "main": “server.js”
}
```

```json
{
  "name": “introduction-to-node“,
  "version": "1.0.0",
  "description": “Code repository for intro to node",
  "main": "server.js",
  "repository": {
    "type": "git",
    "url": “https://github.com/blargyblarg/blarg“
  },
  "dependencies": {
    "express": “latest",
    "mongoose": "latest"
  },
  "author": “Frodo Baggins",
  "license": "MIT",
  "homepage": “https://github.com/shireCoders/shirehome“
}
```

Development Environment

Local Development Environment

- ...continued in the demo...

Development Environment

Deployment Options

- Each web application created with node will have a number of packages attached to it
    - As few as 1 or 2
    - As many as 30…40…onwards?
- Our package.json is the file that stores references to all of these so that it can be deployed on different platforms

- Heroku is a great application that can deploy node applications
    - Also has free options for students…

demo

Node Application

THE NEED FOR CI

HTML / CSS / JS

PHP / MYSQL

NODE

GITHUB AND CONTINUOUS INTEGRATION