Code your own site

Derek Bell
Lucid Web Design

http://lucid.com.au

What we will do:

Frontend

- Start with an HTML5 boilerplate
- Set up version control using git
- Use NPM to set up package management
  - GulpJS
  - Yarn
- Create responsive HTML5 page

What we will do:

Backend

- Convert that to a WordPress theme

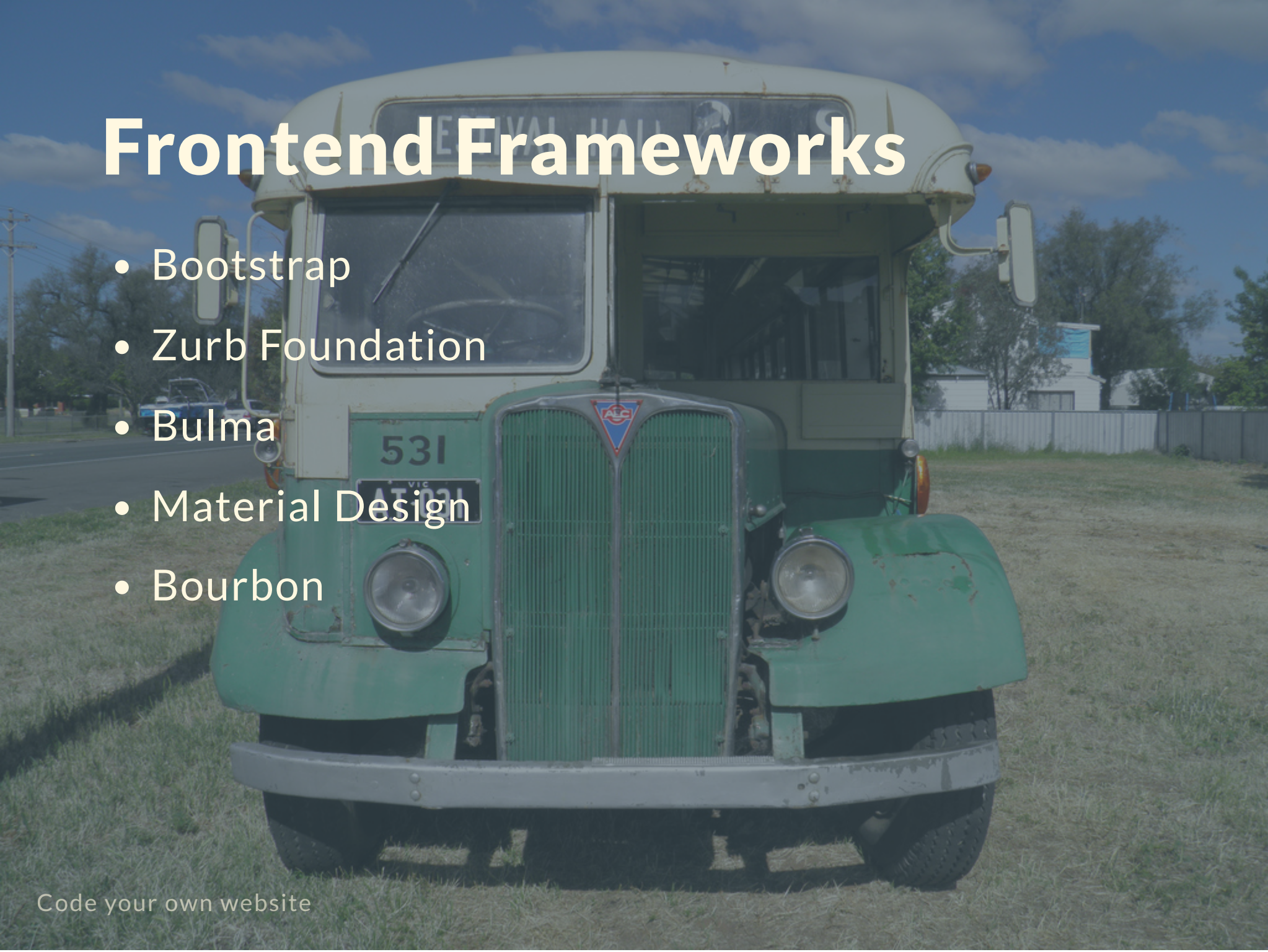
Although you use any other CMS or build your own application.

HTML5 Boilerplate

https://html5boilerplate.com/

Good starter for:

- Websites
- Frontend Apps



Frontend Frameworks

- Bootstrap
- Zurb Foundation
- Bulma
- Material Design
- Bourbon


Do frameworks suck?

Excercise

Dowload HTML 5 Boilerplate and extract to your folder.

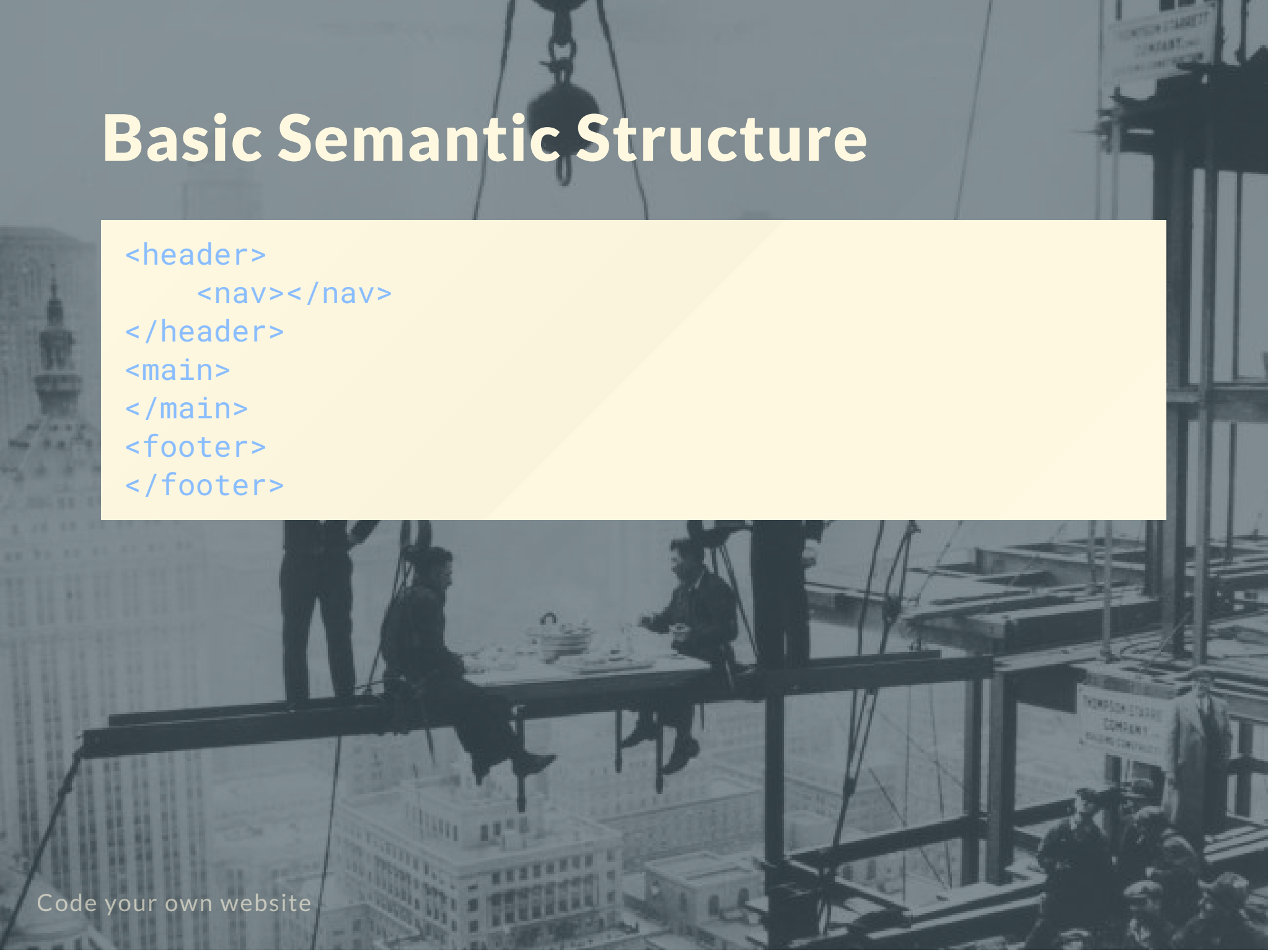
Building the HTML

Code your own website

Describe what you see

Basic Semantic Structure

```
<header>
    <nav></nav>
</header>
<main>
</main>
<footer>
</footer>
```

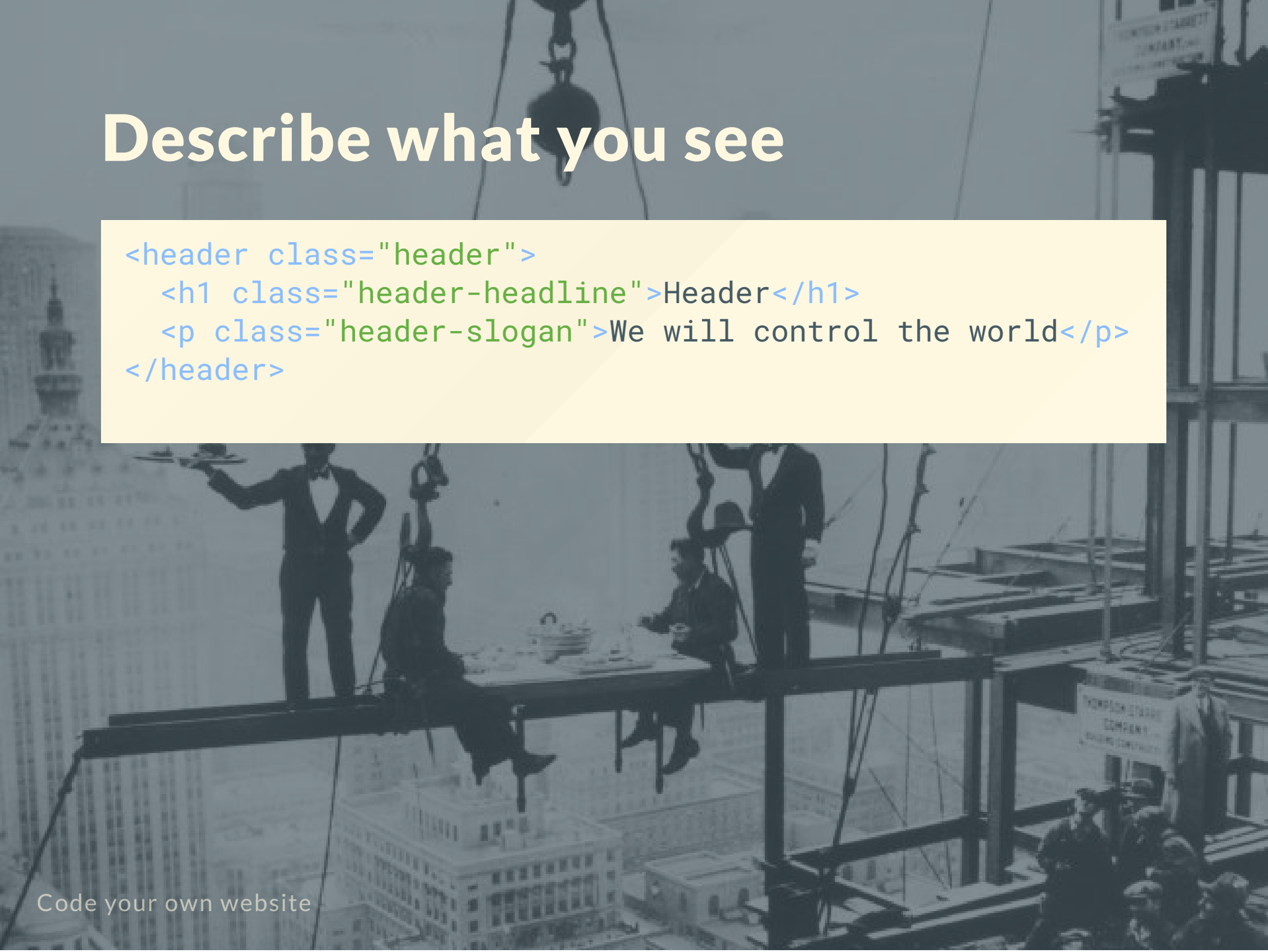


Describe what you see

```
<nav class="topnav">
  <li class = "topnav-menu">
    <a href="#">Home</a>
    <a href="#">About</a>
    <a href="#">Products</a>
    <a href="#">Contact</a>
  </li>
</nav>
```

Describe what you see

```
<header class="header">
  <h1 class="header-headline">Header</h1>
  <p class="header-slogan">We will control the world</p>
</header>
```



Skeleton HTML : Head

```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="x-ua-compatible" content="ie=edg
        <title></title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width,

        <link rel="apple-touch-icon" href="apple-touch-ico
        <!-- Place favicon.ico in the root directory -->
        <link rel="stylesheet" href="css/style.css">
            <style>
                        div {border: 1px #ccc solid;}
                </style>
    </head>
```

Skeleton HTML : Body & Nav

```
<body>
   <nav>
          <div>
               <li><a href="#">About</a></li>
               <li><a href="#">Services</a></li>
               <li><a href="#">Contact</a></li>
           </ul>
         </div>
      </nav>
```

Skeleton HTML : Header

```
<header>
        <div>
                <h1 class="header-headline">Header</h1>
                <p class="header-slogan">We will control t
        </div>
</header>
```

Skeleton HTML : Main

```
<main>
  <div>
    <div></div>
    <div></div>
    <div></div>
  </div>

</main>
```

Skeleton HTML : Footer

```
        <footer>
          <div>
            <div></div>
            <div></div>
          </div>
        </footer>

    </body>
</html>
```

Comments

Add Comments : Head

```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="x-ua-compatible" content="ie=edg
        <title></title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width,

        <link rel="apple-touch-icon" href="apple-touch-ico
        <!-- Place favicon.ico in the root directory -->
        <link rel="stylesheet" href="css/style.css">
    </head>
```

Add Comments : Body & Nav

```
<body>
    <!-- navbar -->
    <nav>
        <div">
            <ul>
                <li><a href="#">Home</a></li>
                <li><a href="#">About</a></li>
                <li><a href="#">Services</a></li>
                <li><a href="#">Contact</a></li>
            </ul>
        </div>
    </nav>
```

Add Comments : Header

```
<!-- header -->
<header>
    <div>
        <h1>Title</h1>
        <p>Lorem ipsum dolor sit amet, consectetur adip
    </div>
</header>
```

Add Comments : Main

```
        <!-- main -->
    <main>
        <!-- top section -->
        <div>
            <div></div>
        </div>

        <!-- middle section -->
        <div>
            <div></div>
        </div>

        <!-- bottom section -->
        <div>
            <div></div>
        </div>
    </main>
```

Add Comments : Footer

```
        <!-- footer -->
        <footer>
          <div>
            <div> </div>
            <div></div>
          </div>
        </footer>

    </body>
</html>
```

CSS Classes

CSS : Head

```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="x-ua-compatible" content="ie=edg
        <title></title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width,

        <link rel="apple-touch-icon" href="apple-touch-ico
        <!-- Place favicon.ico in the root directory -->
        <link rel="stylesheet" href="css/style.css">
    </head>
```

CSS : Body & Nav

```
<body>
    <!-- navbar -->
    <nav class="navbar">
        <div class="container">
            <ul class="navbar-menu">
                <li class="navbar-item"><a href="#">Home</a
                <li class="navbar-item"><a href="#">About</
                <li class="navbar-item"><a href="#">Service
                <li class="navbar-item"><a href="#">Contact
            </ul>
        </div>
    </nav>
```

CSS : Header

```
<!-- header -->
<header>
    <div class="container">
        <h1 class="header-headline">Title</h1>
        <p class="header-slogan">Lorem ipsum dolor sit
    </div>
</header>
```

CSS : Main

```
        <!-- main -->
    <main class="main">
        <!-- top section -->
        <div class="main-top">
            <div class="container">
            </div>
        </div>

        <!-- middle section -->
        <div class="main-middle">
            <div class="container">
            </div>
        </div>

        <!-- bottom section -->
        <div class="main-bottom">
            <div class="container">
            </div>
```

CSS : Footer

```
        <!-- footer -->
        <footer class="footer">
            <div class="footer-top">
                <div class="container">
                </div>
            </div>
            <div class="footer-bottom">
                <div class="container">
                </div>
            </div>
        </footer>

    </body>
</html>
```

Let's fill in the content

```
        <!-- main -->
    <main class="main">
        <!-- top section -->
        <div class="main-top">
          <h2>Main Top</h2>
          p>lorem5
        </div>
```

Press tab after `p>lorem5` becomes...

```
<main>
    <div>
        <h2>Main Top</h2>
        <p>Lorem ipsum dolor sit amet.</p>
    </div>
```

Main Middle

```
<!-- middle section -->
<div class="main-middle">

    <div class="third">
        <div>Icon</div>
        <h3>Item 1</h3>
        <p>Lorem ipsum dolor sit amet, consectetur
        <div class="text-centre">
            <a href="#" class="button">Read More</
        </div>
    </div>
```

SASS

What is SASS?

It's like CSS only crunchy

Why use SASS?

Variables

Assign variables to colours and fonts and it's dead easy to make changes.

```
$brand-color: #fc3;
a {
color: $brand-color;
}
nav {
        background-color: $brand-color;
}
```

SASS is reusable

```
@mixin default-type {
        margin-bottom: 20px;
        font-size: 14px;
        line-height: 1.5;
}

p {
        @include default-type;
}

footer {
        @include default-type;
}
```

SASS Functions

SASS has functions that can:

- Deal with colours
- Number funtions
- Variables and arrays
- Loop structures

http://sass-lang.com/documentation/Sass/Script/Functions.html


Mix SASS with Gulp and gain super powers!
Code your own website

Organising SASS

- Small Chunks
- Keep it modular
- Each file serves one purpose
- Use `@import` to pull everything together

How organise SASS files

- The SASS way http://thesassway.com/beginner/how-to-structure-a-sass-project
- SMACSS https://smacss.com/
- Atomic Design http://atomicdesign.bradfrost.com/

The SASS Way

```
stylesheets/
|
|-- modules/                # Common modules
|   |-- _all.scss           # Include to get all modules
|   |-- _colors.scss        # Etc...
|   ...
|
|-- partials/               # Partials
|   |-- _base.sass          # imports for all mixins + globa
|   ...
|
|-- vendor/                 # CSS or Sass from other project
|   |-- _colorpicker.scss
|   ...
|
`-- main.scss             # primary Sass file
```

SMACSS

```
scss/
|- _base/
|  |- _config.scss
|  |- _presets.scss
|- _layouts/
|  |-base.scss
|  |-grid.scss
|- _modules/
|  |-buttons.scss
|  |-tabs.scss
|- _states/
|  |-buttons.scss
|  |-tabs.scss
|- application.scss
```

Atomic Design

```
stylesheets/
|- utilities/
|    |- _variables.sass, reset
|    |- ...
|- atoms/
|    |- _headings.sass, buttons
|    |- ...
|- molecules/
|    |- _media.sass, search forms
|    |- ...
|- organisms/
|    |- _sign_in_form.sass
|    |- ...
|- templates/
|    |- _default_layout.sass
|    |- ...
style.scss
```

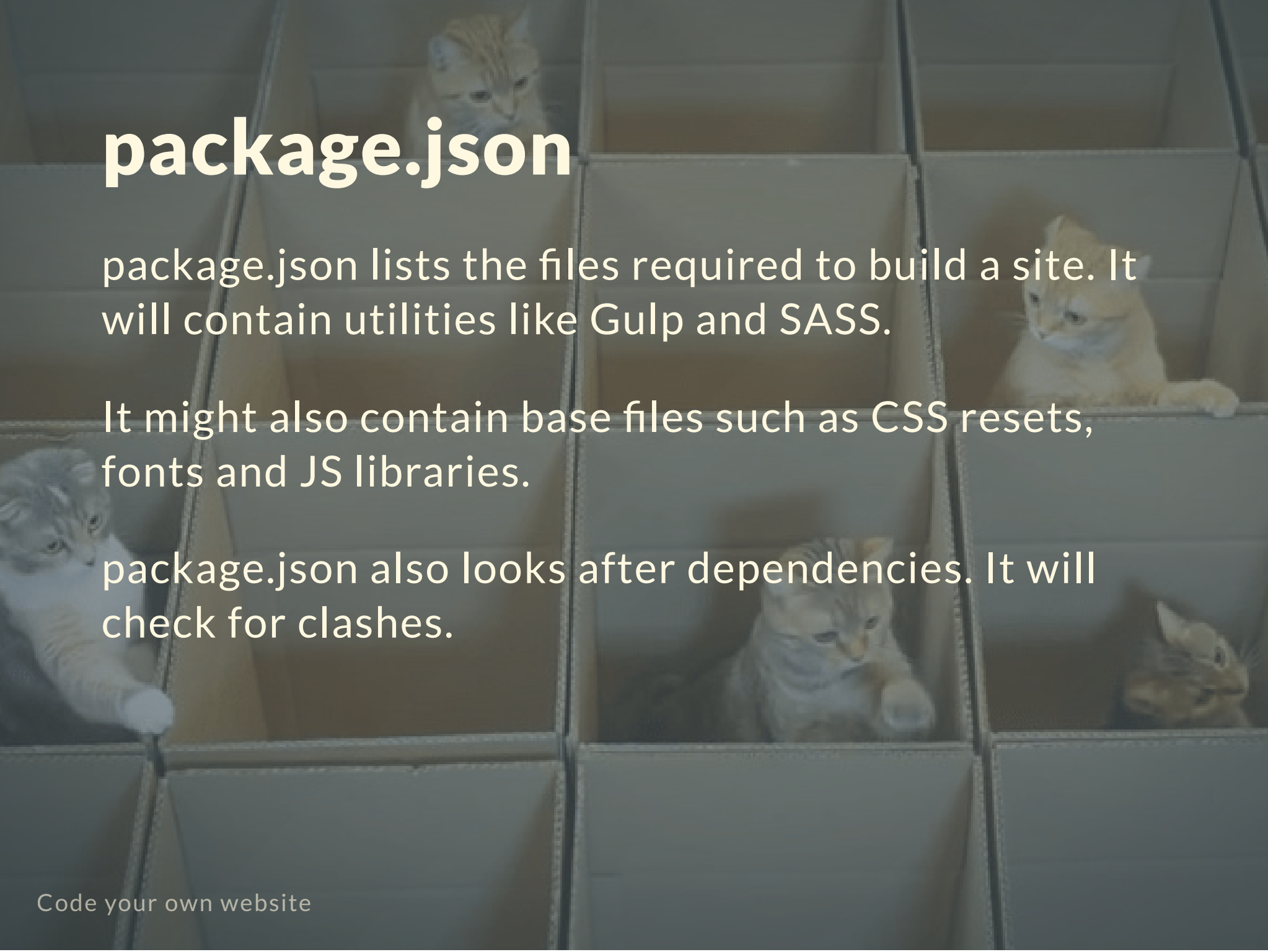

Which way is the right way?


We will use SMACSS

Get your Base SASS Files here

https://github.com/llebkered/codeyoursite/blob/master/sass.zip

Go over the SASS file structure

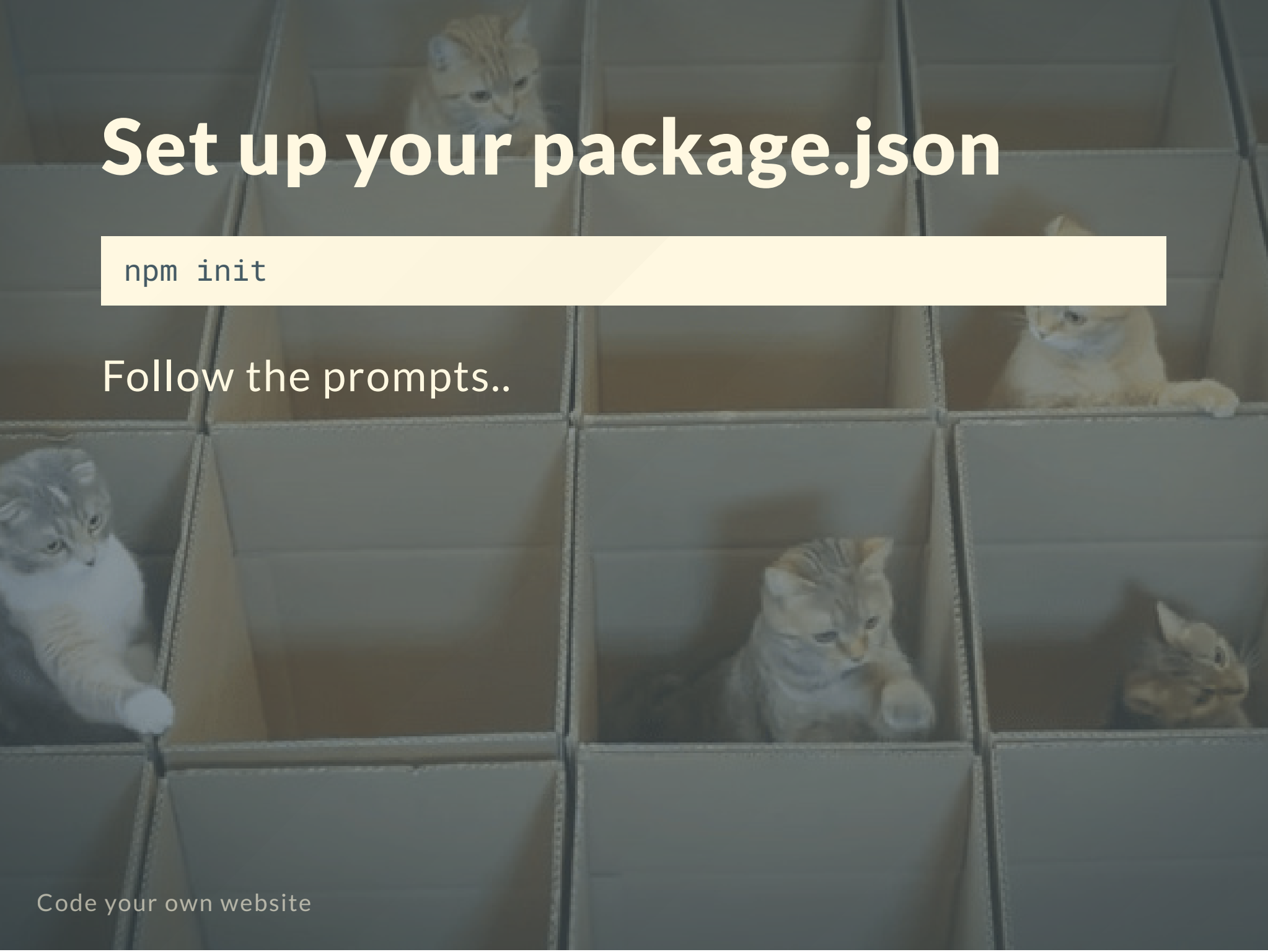


package.json

package.json lists the files required to build a site. It will contain utilities like Gulp and SASS.

It might also contain base files such as CSS resets, fonts and JS libraries.

package.json also looks after dependencies. It will check for clashes.



Set up your package.json

```
npm init
```

Follow the prompts..

```
{
  "name": "codeyoursite",
  "version": "1.0.0",
  "description": "This repository contains the files used
  "main": "gulpfile.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/llebkered/codeyoursite.
  },
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/llebkered/codeyoursite/issu
  },
  "homepage": "https://github.com/llebkered/codeyoursite#r
  "dependencies": {
```

Gulp.js

gulp is a toolkit for automating painful or time-consuming tasks in your development workflow, so you can stop messing around and build something.

Gulp JS

Why bother? It seems like a lot of work?

4 Uses for Gulp JS

- Spin up a web server to test your site.
- Reloading the browser automatically whenever a file is saved
- Compiles Sass into CSS
- Optimise assets like CSS, JavaScript, and images

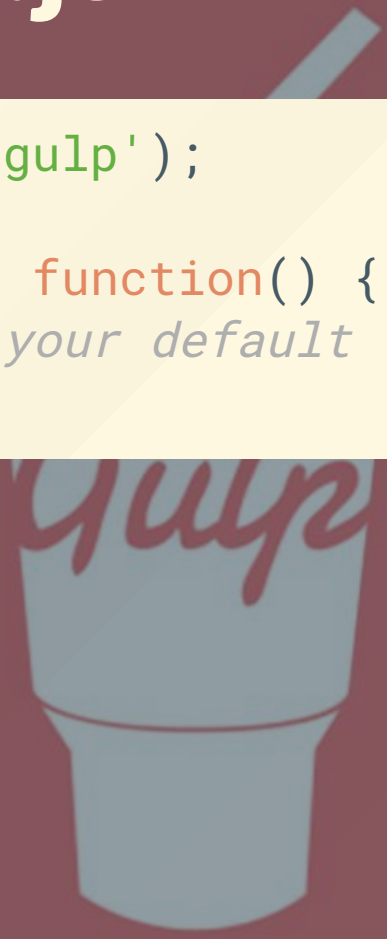
Useful site

https://css-tricks.com/gulp-for-beginners/

Basic Gulp.js

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

gulp.task('default', function() {
  // place code for your default task here
});
```



```
// Gulp & utilities
var gulp = require('gulp');
var gutil = require('gulp-util');
var watch = require('gulp-watch');

// CSS
var sass = require('gulp-sass');
var autoprefixer = require('gulp-autoprefixer');
var csso = require('gulp-csso');


// Browser Sync
var browserSync = require('browser-sync').create();
```

```
/* ================= */
/* CSS Files */
// Compile SASS and Autoprefix.
gulp.task('sass', function() {
  gulp.src('sass/**/*.scss')
    .pipe(sass().on('error', sass.logError))
    .pipe(sass({
      outputStyle: 'expanded'
    }))
    .pipe(autoprefixer(['last 15 versions', '> 1%', 'ie 8'
      cascade: true
    }))
    // Minify CSS
    .pipe(csso())
    .pipe(gulp.dest('css'))
    .pipe(browserSync.stream());
});
```

```
// Browser Sync
// see https://www.browsersync.io/docs/gulp/

// Dynamic server
//gulp.task('browser-sync', function() {
//    browserSync.init({
//        proxy: "yourlocal.dev"
//    });
// });

// Static server
gulp.task('browser-sync', function() {
    browserSync.init({
        server: {
            baseDir: "./"
        }
    });
});
```

```
/* ================== */
/* Gulp Watch */

gulp.task('watch', function() {
  // watch scss files
  gulp.watch('sass/**/*.scss', ['sass']);

});

/*gulp.task('default', function() {
  // place code for your default task here
});
*/
gulp.task('default', ['sass', 'browser-sync','watch']);
```

Download at:

https://raw.githubusercontent.com/llebkered/codeyoursite/master/gulpfile.js

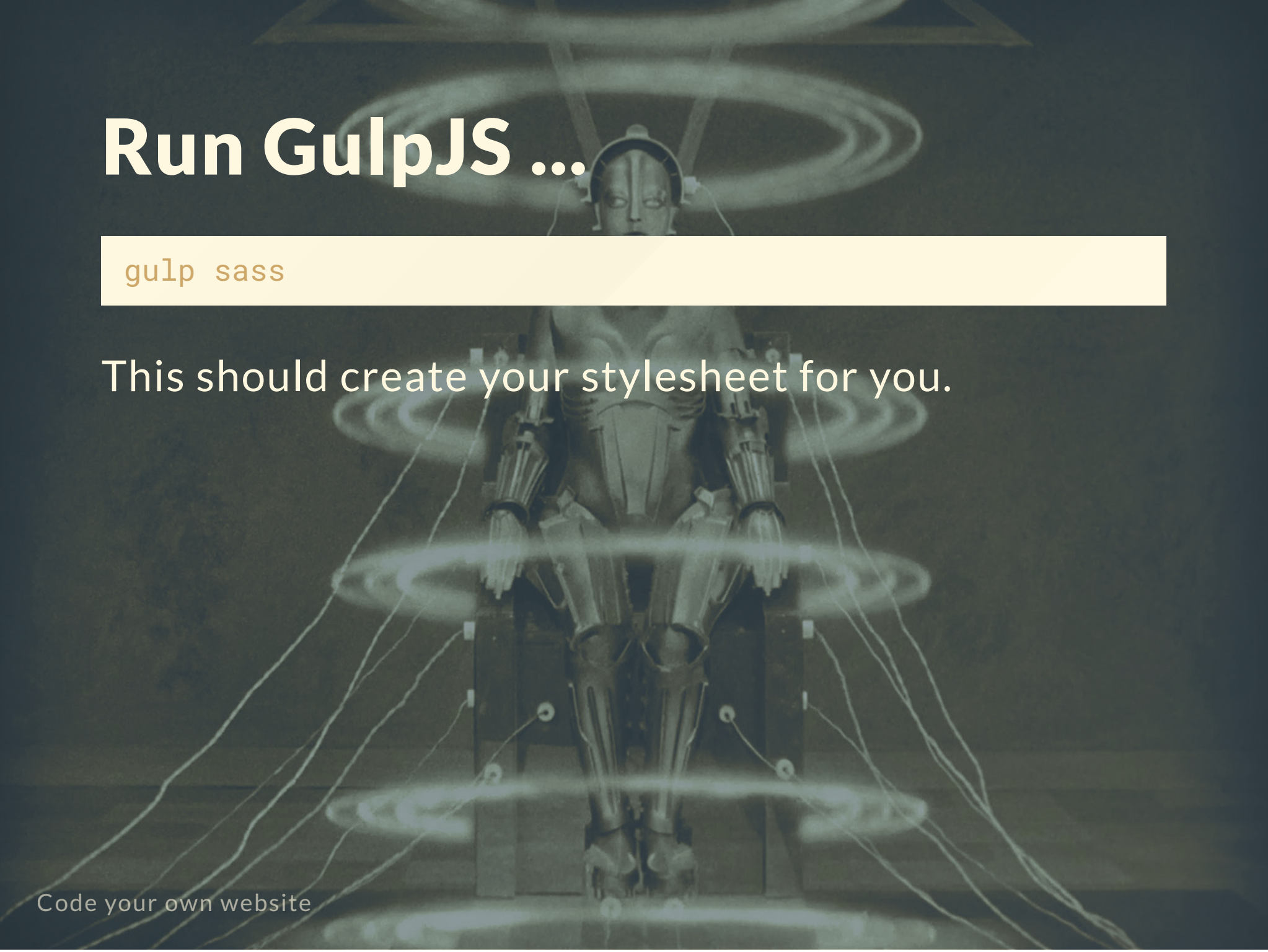
Let's install the packages ...

```
yarn add gulp gulp-sass gulp-autoprefixer gulp-csso
gulp-watch browser-sync
```

Install Gulp command line

```
npm install gulp-cli -g
```

Note: you only need to do this once.



Run GulpJS ...

```
gulp sass
```

This should create your stylesheet for you.

Layout

Create skeleton sass files for layout in the sass/layout folder

1. _index.scss

Add the following code:

```
body {}

.header {}

.navbar {}

.main {}

.footer {}

.container {}
```

Save and run `gulp sass`

Let's create the container

```
.container {
        max-width: 1200px;
    margin-left: auto;
    margin-right: auto;
}
```

Save and run `gulp sass`

What happened?

Let's add some padding

```
.container {

    max-width: 1200px;

    margin-left: auto;
    margin-right: auto;

    padding-left: 20px;
    padding-right: 20px;
}
```

Save and run `gulp sass`

What happened?

Modules

sass/modules folder will contain most of your work. Let's set up the starter files. Create the following file:

`sass/modules/_page.scss` (note the underscore)

add the code

```
body {}
```

Import the code

Now we can import it. Open `sass/modules/_index.scss`

Add the following statement on line 19

```
@import 'page';
```

Do the same for each section:

- navbar
- header
- main
- footer

File `sass/modules/_navbar.scss`

Code:

```
.navbar {}
```

Now we can import it. Open `sass/modules/_index.scss`

Add the following statement on line 20

`@import 'navbar';`

File `sass/modules/_header.scss`

Code:

```
.header {}
```

Now we can import it. Open `sass/modules/_index.scss`

Add the following statement on line 21

`@import 'header';`

File `sass/modules/_main.scss`

Code:

```
.main {}
```

Now we can import it. Open `sass/modules/_index.scss`

Add the following statement on line 22

`@import 'main';`

Code your own website

File `sass/modules/_footer.scss`

Code:

```
.footer {}
```

Now we can import it. Open `sass/modules/_index.scss`

Add the following statement on line 23

`@import 'footer';`

Version Control

- Tracks changes
- Reverse any bad changes
- Teams
- Set up branches of code

Git

Git is a version control system (VCS) for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source code management in software development, but it can be used to keep track of changes in any set of files. As a distributed revision control system it is aimed at speed, data integrity, and support for distributed, non-linear workflows.

Some Basic Local Git Commands

`git init` Initialise a repository

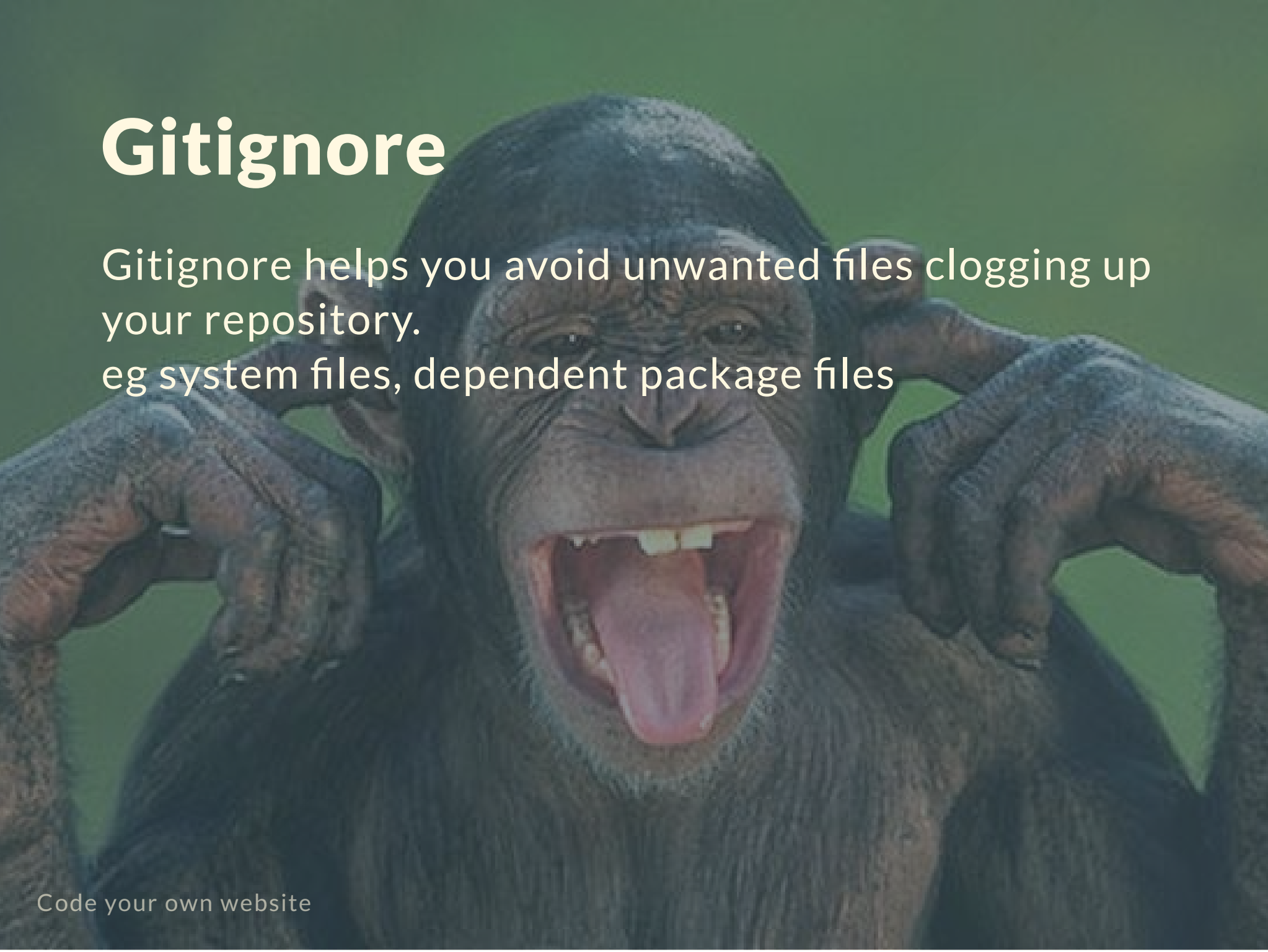
`git add --all` Adds files to the repository

`git commit -m 'commit message'` Commits your changes with a status comment

`git status`

Gitignore

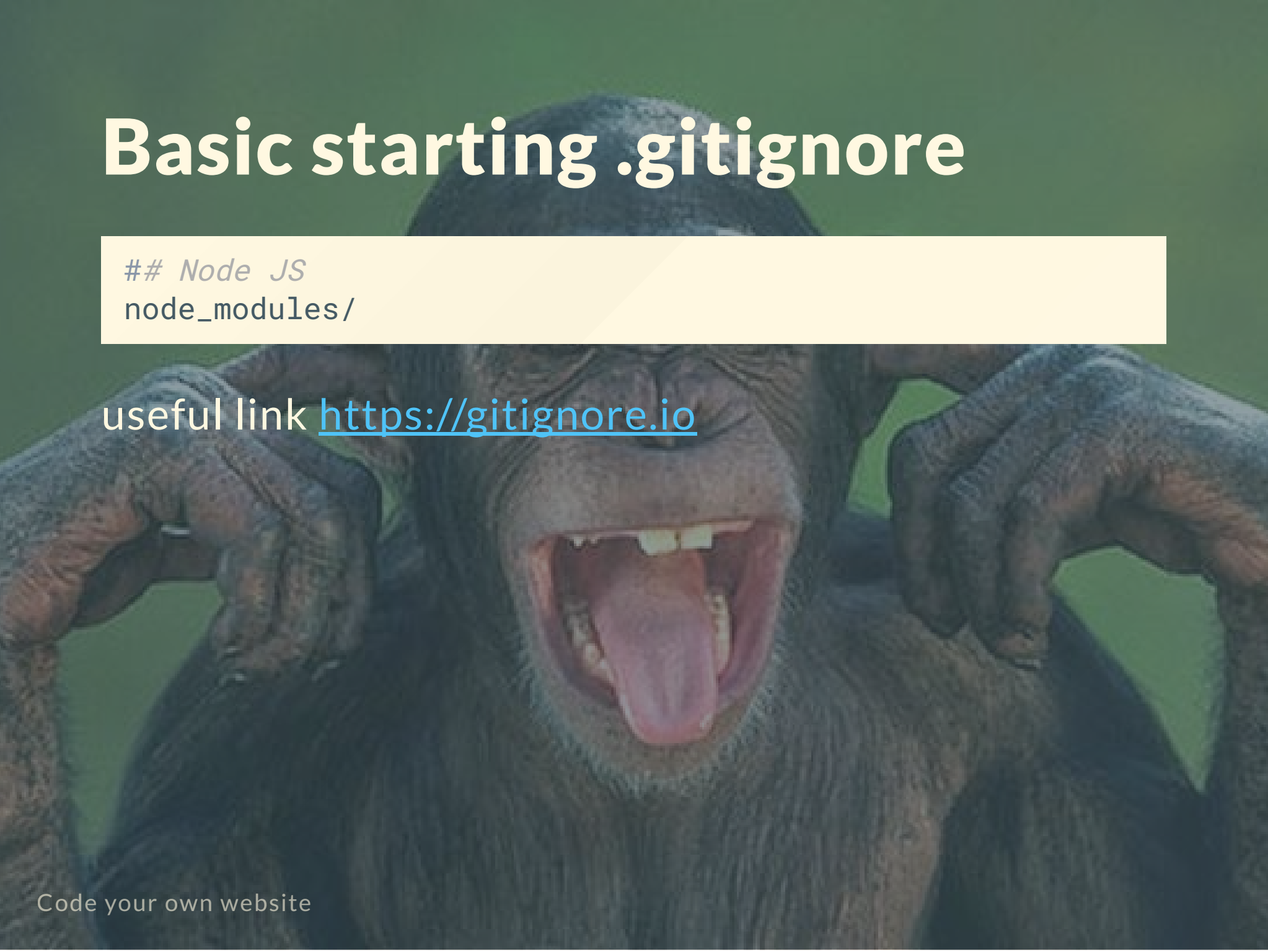
Gitignore helps you avoid unwanted files clogging up your repository.
eg system files, dependent package files



Basic starting .gitignore

```
## Node JS
node_modules/
```

useful link https://gitignore.io



SASS Variables

Think of variables as a way to store information that you want to reuse throughout your stylesheet. You can store things like colors, font stacks, or any CSS value you think you'll want to reuse.
Sass uses the $ symbol to make something a variable.

```scss
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

produces ...

```scss
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

Using Variables 1

open `sass/base/_variables.scss`

add a new line

```
// Breakpoints
$breakpoint-large: 1200px;
```

Using Variables 2

now open `sass/layout/_index.scss`

change `max-width` from 1200px to $breakpoint-large

```
.container {
        max-width: 1200px;
    margin-left: auto;
    margin-right: auto;
}
```

Save and run `gulp sass`

What happened?

Nested SASS

When writing HTML you've probably noticed that it has a clear nested and visual hierarchy. CSS, on the other hand, doesn't.

Sass will let you nest your CSS selectors in a way that follows the same visual hierarchy of your HTML. Be aware that overly nested rules will result in over-qualified CSS that could prove hard to maintain and is generally considered bad practice.

Nested SASS

```
nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

Gives us

```
nav ul {
  margin: 0;
  padding: 0;
  list-style: none;
}

nav li {
  display: inline-block;
}

nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none;
}
```

Don't go more than 3 levels deep

(The Inception Rule)