INFO3180 - LECTURE 7

FRONT END LIBRARIES AND TOOLS AND VUEJS

FRONTEND TOOLS

CSS FRAMEWORKS

CSS Frameworks are meant to help you to create layouts and build more standards-compliant websites/web apps faster and easier using HTML and CSS.

BENEFITS OF A CSS FRAMEWORK

- Can help you to build websites/prototypes quickly.
- Encourage a grid based design.
- Helps with Responsive Web Design
- Don't Repeat Yourself (DRY) since they take care of some common components and design patterns used for websites.
- Cross-browser compatibility
- They can help you to learn CSS

DRAWBACKS OF A CSS FRAMEWORK

- They can be bloated, as you may not need all the functionality that it gives you. And some don't allow you to remove what you don't need from the package.
- You can get stuck doing things the way the framework wants you to do it. And if you try to break out of that, then you end up losing the time trying to change the way the framework does things.

CSS FRAMEWORKS

- Bootstrap - getbootstrap.com
- Tailwind CSS - https://tailwindcss.com/
- Foundation - http://foundation.zurb.com/
- Bulma - http://bulma.io/
- UI Kit - https://getuikit.com/
- and there are many others

Bootstrap

The one we will focus on in this course is Bootstrap. It was originally designed and developed by some members of the Twitter design team.

Bootstrap is built to be mobile friendly from the start and includes a responsive grid system.

Bootstrap's grid system uses a series of containers, rows, and columns to layout and align content. It's built with flexbox and is fully responsive.

https://getbootstrap.com/docs/5.3/layout/grid/

ents

Site Name

Utility Nav Link   Utility Nav Link

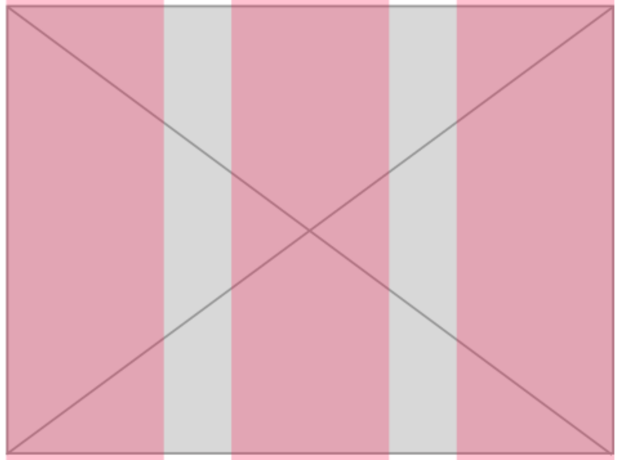
Services ▾   Work ▾   About ▾   Blog   Contact

Home / Page Title

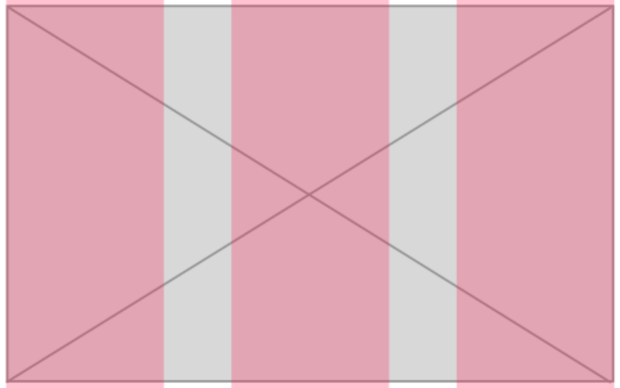
Page Title

Some large introductory text that can provide an overview of what this page is about. This can be useful for readers that are skimming (everyone).

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla tristique quam in purus eleifend maximus. Nullam sed magna eget leo consequat faucibus in vitae magna. Quisque ultricies, enim at tempor ornare, lorem orci congue lectus, ut viverra ante lectus a quam. Pellentesque elementum luctus risus, vel pulvinar magna facilisis at. Vivamus sit amet pretium diam. Suspendisse in neque ac orci auctor rutrum.

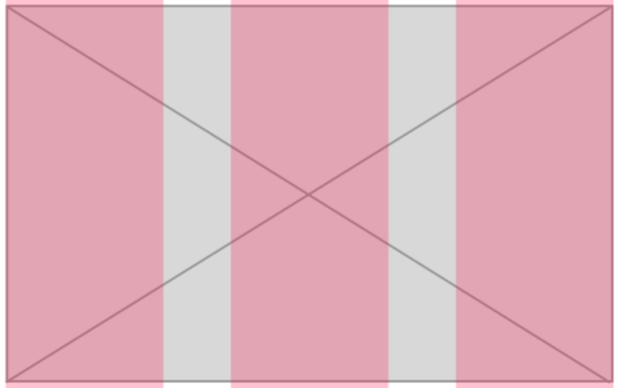


Four-column Layout



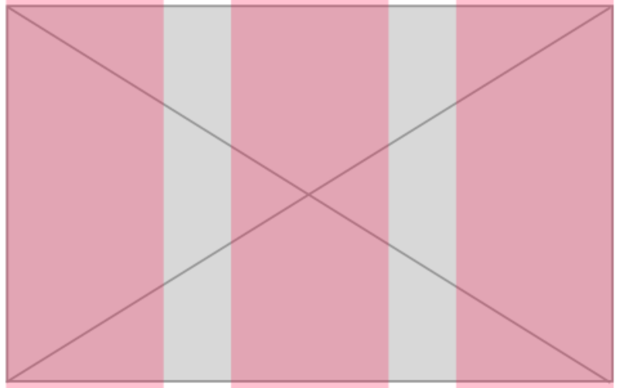
Title of this column goes here

Nunc quis massa vitae neque tempus eleifend. Etiam at nisi rhoncus, rutrum ligula vitae, aliquet elit. Nunc tellus sem, congue eu laoreet ut.



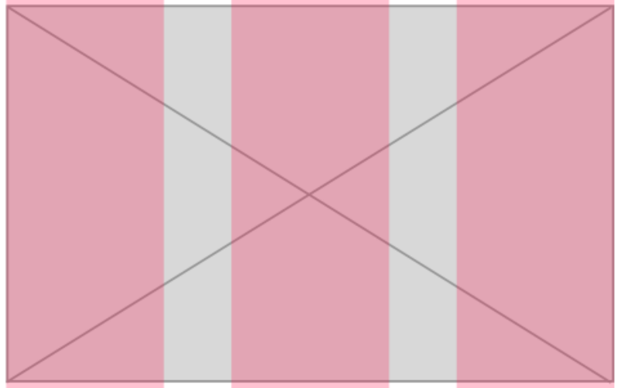
Title of this column goes here

Nunc quis massa vitae neque tempus eleifend. Etiam at nisi rhoncus, rutrum ligula vitae, aliquet elit. Nunc tellus sem, congue eu laoreet ut.



Title of this column goes here

Nunc quis massa vitae neque tempus eleifend. Etiam at nisi rhoncus, rutrum ligula vitae, aliquet elit. Nunc tellus sem, congue eu laoreet ut.



Title of this column goes here

Nunc quis massa vitae neque tempus eleifend. Etiam at nisi rhoncus, rutrum ligula vitae, aliquet elit. Nunc tellus sem, congue eu laoreet ut.

Bootstrap uses a 12-column grid and each column uses a CSS class to determine how wide. e.g. **`col-*, col-sm-*, col-md-*, col-lg-*`**, *etc.*

.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1
.col-md-8								.col-md-4			
.col-md-4			.col-md-4			.col-md-4					
.col-md-6					.col-md-6						

https://getbootstrap.com/docs/5.3/layout/grid/

```
<div class="container">
  <div class="row">
    <div class="col-md-4">
      One of three columns
    </div>
    <div class="col-md-4">
      One of three columns
    </div>
    <div class="col-md-4">
      One of three columns
    </div>
  </div>
</div>
```

Bootstrap also provides default styling for headings, paragraphs, lists, forms, buttons, etc.

It even takes it a step further by providing styling for some common components such as navigation bars, alerts, breadcrumbs, pagination, cards, slideshows and others.

ALERT COMPONENT STYLES

This is a primary alert—check it out!

This is a secondary alert—check it out!

This is a success alert—check it out!

This is a danger alert—check it out!

This is a warning alert—check it out!

This is a info alert—check it out!

This is a light alert—check it out!

This is a dark alert—check it out!

```
<div class="alert alert-primary" role="alert">
  A simple primary alert—check it out!
</div>
<div class="alert alert-secondary" role="alert">
  A simple secondary alert—check it out!
</div>
<div class="alert alert-success" role="alert">
  A simple success alert—check it out!
</div>
<div class="alert alert-danger" role="alert">
  A simple danger alert—check it out!
</div>
```

CARD COMPONENT

Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

Go somewhere

```
<div class="card" style="width: 18rem;">
  <img src="..." class="card-img-top" alt="...">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to
build on the card title and make up the bulk of the
card's content.</p>
    <a href="#" class="btn btn-primary">Go
somewhere</a>
  </div>
</div>
```

BUTTON COMPONENT STYLES

```
<button type="button" class="btn btn-primary">
  Primary
</button>
<button type="button" class="btn btn-secondary">
  Secondary
</button>
<button type="button" class="btn btn-success">
  Success
</button>
<button type="button" class="btn btn-danger">
  Danger
</button>
```

Take a look at the Bootstrap documentation for more information on these and other components.

http://getbootstrap.com

CSS PREPROCESSORS

As our stylesheets get larger and more complex, they also become harder to maintain.

***CSS Preprocessors extend CSS** by giving you extra features not currently available in CSS such as **variables**, **nesting**, **mixins**, **inheritance** and even **mathematical calculations**. These are then compiled into regular CSS syntax.*

SOME ADVANTAGES

- You can write cleaner code with reusable pieces (DRY) by allowing us to create variables or inherit properties from another selector from reusable CSS properties and even allows nesting CSS selectors.
- More flexibility to do things on the fly (e.g. calculations)
- You can import snippets (with partials) or other libraries (with imports)
- Cross browser compatibility (with the help of mixins)

CSS PREPROCESSORS

- Syntactically Awesome Stylesheets (SASS/SCSS) - http://sass-lang.com/
- LESS - http://lesscss.org/
- Stylus - http://stylus-lang.com/
- and there are others...

The one we will focus on today is SASS.

EXAMPLE OF VARIABLES

```sass
/* SASS Code */

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

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


/* Output when compiled to CSS */
body {
  font: 100% Helvetica, sans-serif;
  color: #333;
}
```

EXAMPLE OF INHERITANCE

```scss
/* SASS Code */
%message-shared {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}

.message {
  @extend %message-shared;
}

.success {
  @extend %message-shared;
  border-color: green;
}

.error {
  @extend %message-shared;
  border-color: red;
}
```

```scss
/* output when compiled to CSS */
.message, .success, .error {
  border: 1px solid #cccccc;
  padding: 10px;
  color: #333;
}

.success {
  border-color: green;
}

.error {
  border-color: red;
}
```

EXAMPLE OF NESTING

```scss
/* SASS code */
nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

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

```css
/* output when compiled to CSS */
nav ul {
  margin: 0;
  padding: 0;
  list-style: none;
}

nav li {
  display: inline-block;
}

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

EXAMPLE OF MIXINS

```scss
/* SASS Code */
@mixin border-radius($radius) {
  -webkit-border-radius: $radius;
     -moz-border-radius: $radius;
      -ms-border-radius: $radius;
          border-radius: $radius;
}

.box {
  @include border-radius(10px);
}
```

```scss
/* output when compiled
to CSS */
.box {
  -webkit-border-radius: 10px;
     -moz-border-radius: 10px;
      -ms-border-radius: 10px;
          border-radius: 10px;
}
```

EXAMPLE OF PARTIALS AND IMPORTS

```scss
/* A SASS Partial called
_somepartial.scss */
html,
body,
ul,
ol {
  margin: 0;
  padding: 0;
}
```

```scss
/* Another file called
base.scss */
@import 'somepartial';

body {
  font: 100% sans-serif;
  background-color: #efefef;
}
```

SASS files have a .scss (or a .sass) extension and are compiled back into .css files. From the command line you would run the following to compile the sass file.

```
$ sass mysassfile.scss:mycssfile.css

or

$ sass --watch mysassfile.scss:mycssfile.css
```

Take a look at the SASS documentation for more examples and features.

http://sass-lang.com/

TYPESCRIPT

TypeScript is an open-source language which builds on JavaScript, one of the world's most used tools, by adding static type definitions.

typescriptlang.org

Types provide a way to describe the shape of an object, providing better documentation, and allowing TypeScript to validate that your code is working correctly.

typescriptlang.org

BASIC TYPES

- string
- number
- boolean
- arrays (e.g. `number[]` or `string[]`)
- any
- union types (e.g. `number | string`)

DEFINING A TYPE

```
let name: string = "Lauren";

let idNumber: number = "620099999";

let id: number | string;
```

PARAMETER TYPE ANNOTATIONS

```
// Parameter type annotation
function greet(name: string) {
    console.log("Hello, " + name.toUpperCase()
+ "!!");
}
```

RETURN TYPE ANNOTATIONS

```
function getFavoriteNumber(): number {
    return 7;
}
```

OBJECT TYPES

```ts
// The parameter's type annotation is an object type
function printCoord(pt: { x: number; y: number }) {
  console.log("The coordinate's x value is " + pt.x);
  console.log("The coordinate's y value is " + pt.y);
}

printCoord({ x: 3, y: 7 });
```

OPTIONAL PROPERTIES

```
function printName(obj: { first: string; last?:
string }) {
// ...
}
// Both OK
printName({ first: "Bob" });
printName({ first: "Alice", last: "Alisson" });
```

INTERFACES

```ts
interface Point {
  x: number;
  y: number;
}

function printCoord(pt: Point) {
  console.log("The coordinate's x value is " + pt.x);
  console.log("The coordinate's y value is " + pt.y);
}

printCoord({ x: 100, y: 100 });
```

Learn more about TypeScript at

https://www.typescriptlang.org

PACKAGE MANAGERS, TASK RUNNERS AND BUNDLERS

As developers we spend most of our time coding, but there are often some basic tasks that we have to do over and over that can take up a lot of our time.

SOME COMMON TASKS THAT FRONTEND DEVELOPERS NEED TO DO

- Compressing new and modified images
- Compiling SASS to CSS code
- Removing console and debugger statements from scripts
- Transpiling ES6 to cross-browser-compatible ES5 code
- Code linting and validation
- Concatenating and minifying CSS and JavaScript files
- Deploying files to development, staging and production servers.

***Package Managers** are tools that can help you to find, download and install frontend libraries for your web applications.*

***Task Runners** are tools that can help you to automate the frontend tasks we mentioned earlier.*

***Bundlers** gather all your dependencies/ modules (not just code, but other assets as well) and generate a dependency graph. It then packages and optimizes all those dependencies/modules into one or more bundles.*

PACKAGE MANAGERS, TASK RUNNERS AND BUNDLERS

- Node.js/npm (Package Manager) - https://nodejs.org and https://www.npmjs.com/
- Yarn (Package Manager) - https://yarnpkg.com/en/
- Grunt (Task Runner) - https://gruntjs.com/
- Gulp (Task Runner) - http://gulpjs.com/
- Webpack (Bundler) - https://webpack.js.org/
- Browserify (Bundler) - http://browserify.org/
- Parcel (Bundler) - https://parceljs.org
- Vite (Bundler) - https://vitejs.dev

The ones we will focus on today are Node.js, npm and Gulp.

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. It is an open source, cross-platform runtime environment for developing server-side and networking applications.

SIMPLE NODE SCRIPT TO START A WEB SERVER

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

http.createServer((request, response) => {
  response.statusCode = 200;
  response.setHeader('Content-Type', 'text/plain');
  response.end('Hello World!!');
}).listen(8888, () => {
  console.log('Server listening on port 8888');
});
```

TO RUN YOUR NODE SCRIPT

```
$ node app.js
```

npm is the package manager for JavaScript and comes with Node.js.

With npm you can install a package using the command:

```
$ npm install -g <packagename>
$ npm install --save-dev <packagename>
```

These packages can be found at npmjs.com

These packages will get stored in a node_modules folder within your application.

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

http://gulpjs.com/

Gulp tasks and configurations are stored in a `gulpfile.js` file.

```
const { dest, src, watch, series } = require('gulp');
const rename = require('gulp-rename');
const uglify = require('gulp-uglify');
const sass = require('gulp-sass')(require('sass'));


function minify() {
  return src('src/js/*.js')
    .pipe(uglify())
    .pipe(rename({ extname: '.min.js' }))
    .pipe(dest('build/'));
}

function scss() {
  return src('src/scss/*.scss')
    .pipe(sass.sync().on('error', sass.logError))
    .pipe(dest('build/css'));
}

exports.default = series(minify, scss);
```

We can then simply run `gulp` from the command line and it will take our `app.js` JavaScript file and create a minified version of it. It will also take any of our SASS files and compile them to CSS files.

RESOURCES

- Bootstrap - https://getbootstrap.com/
- TypeScript - https://www.typescriptlang.org/
- SASS - https://sass-lang.com
- Preprocessors - http://learn.shayhowe.com/advanced-html-css/preprocessors/
- GulpJS - https://gulpjs.com
- NodeJS - https://nodejs.org
- NPM - https://npmjs.com

DEMO