```
@import "theme/contact-form7"; // Contact Form 7 - Bootstrap 4 support
@import "theme/events_slider";
@import "theme/footer";
@import "theme/megamenu";
@import "theme/navmenus";
@import "theme/frontrow";
@import "theme/whitenews";
@import "theme/maincontent";
@import "theme/instit
@import "theme/theme          things, need their own files

/* Font Awesome 5 */

// We need to know the URL of the fonts at compile time, therefore we
// can't use get_theme_relative_uri() and friends. See
// ../inc/serve-assets.php to find out how this magical URL is being
// served.
$fa-font-path: 'theme-epfl-sti/node_modules/@fortawesome/fontawesome-free-webfon

@import "../node_modules/@fortawesome/fontawesome-free-webfonts/scss/fortaweso
@import "../node_modules/@fortawesome/fontawesome-free-webfonts/scss/fa-regular
@import "../node_modules/@fortawesome/fontawesome-free-webfonts/scss/fa-solid
@import "../node_modules/@fortawesome/fontawesome-free-webfonts/scss/fa-brands
```

WEBSTI

— DEVELOPING THE THEME —

PURPOSE

The purpose of this presentation is to present the tools we use (or can use) in the development process for the new School of Engineering website's WordPress theme.

SIDE NOTE

This document has been made with and can be displayed as (reveal.js) slides here: https://hackmd.io/p/H1yB28iEM#/

LIST OF TOOLS

- npm - https://www.npmjs.com/
- gulp - https://gulpjs.com/
- sass - http://sass-lang.com/
- browsersync - https://browsersync.io/
- bootstrap 4 - https://v4-alpha.getbootstrap.com/

NPM

BUILD AMAZING THINGS

- npm is the package manager for JavaScript.
- Use npm to install, share, and distribute code; manage dependencies in your projects; and share & receive feedback with others.

WHERE DO WE USE NPM ? (1/2)

- We are using `node_modules` for a lot of JS related stuff. For example, we use it for importing bootstrap, fontawesome, jquery, cookieconsent, etc.
- The full list stand in `package.json` in the `epfl-sti` theme folder.

WHERE DO WE USE NPM ? (2/2)

- When we run `npm i` (or `npm install`), all the package listed in `package.json` are downloaded and installed into the `node_modules` folder.

GULP

AUTOMATE AND ENHANCE YOUR WORKFLOW

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

WHERE DO WE USE GULP ?

- When we run `npm`, the `postinstall` command in `package.json` call `gulp` and execute the content of `gulp.js`.
- When called, it:
  - process the `sass` files;
  - unify and unglify `css` and `js` files;
  - create minified images.

SASS

CSS WITH SUPERPOWERS

- Sass is the most mature, stable, and powerful professional grade CSS extension language in the world.

SASS MAKES CSS FUN AGAIN

- Sass is an extension of CSS, adding nested rules, variables, mixins, selector inheritance, and more.

- The main syntax (as of Sass 3) is known as "SCSS" (for "Sassy CSS"), and is a superset of CSS's syntax. This means that every valid CSS stylesheet is valid SCSS as well. SCSS files use the extension `.scss`.

WHERE DO WE USE SASS ?

- When we run `npm`, the `postinstall` command in `package.json` call `gulp` and execute the content of `gulp.js` which compiles the theme's SCSS files (located in the `sass` folder) to CSS files.

SASS EXAMPLE (VARIABLES)

```
body {
  font: 100% Helvetica, sans-serif;
  color: #333;
}
```

is the same as:

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

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

SASS EXAMPLE (NESTING)

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

nav li {
  display: inline-block;
}

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

is the same as:

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

  li { display: inline-block; }

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

SASS EXAMPLE (IMPORT)

CSS has an import option but each time you use @import in CSS it creates another HTTP request. Sass builds on top of the current CSS @import but instead of requiring an HTTP request, Sass will take the file that you want to import and combine it with the file you're importing into so you can serve a single CSS file to the web browser.

```
// base.scss

@import 'color_variables';
```

Note the missing `.scss` file extentions

SASS EXAMPLE (EXTENDS)

```scss
.message, .success, .error, .warning {
  border: 1px solid #cccccc;
  padding: 10px;
  color: #333;
}
.error {
  border-color: red;
}
...
```

```scss
%message-shared {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}
.error {
  @extend %message-shared;
  border-color: red;
}
...
```

Note the missing `.scss` file extentions

BROWSERSYNC

TIME-SAVING SYNCHRONISED BROWSER TESTING.

- From live reloads to URL pushing, form replication to click mirroring, Browsersync cuts out repetitive manual tasks. It's like an extra pair of hands.

WHEN TO USE BOWERSYNC ?

- When developing the theme, use `npm start` to let `BrowserSync` refresh your browser when `gulp` has finished its works.

BOOTSTRAP

Showcase: *https://loichu.github.io/Bootstrap4-Showcase/*

WHY BOOTSTRAP

- Fast development
- Flexibility
- Big community

BASICS

- Grid part and raw components part
- Containers and rows
- Latest css features embeded

http://getbootstrap.com/docs/4.0/layout/overview/

GRID LAYOUT

- **12**, the golden number
- Responsive design
- Nesting grids
- Offsets

http://getbootstrap.com/docs/4.0/layout/grid/#how-it-works

A row made of one column at 50% of the screen surrounded by two automatic width columns.

```
<div class="container">
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col-6">
      2 of 3 (wider)
    </div>
    <div class="col">
      3 of 3
    </div>
  </div>
</div>
```

RESPONSIVE BREAKPOINTS

There are five grid tiers, one for each responsive breakpoint:

- all breakpoints (extra small) `xs`,
- small `sm`,
- medium `md`,
- large `lg`,
- and extra large `xl`.

The first and third columns will disappear on small screens and the second one will take the full width. Thanks to the display property wrapper (d-).

```html
<div class="container">
  <div class="row">
    <div class="col d-sm-none d-md-block">
      1 of 3
    </div>
    <div class="col-md-8 col-sm-12">
      2 of 3 (wider)
    </div>
    <div class="col d-sm-none d-md-block">
      3 of 3
    </div>
  </div>
</div>
```
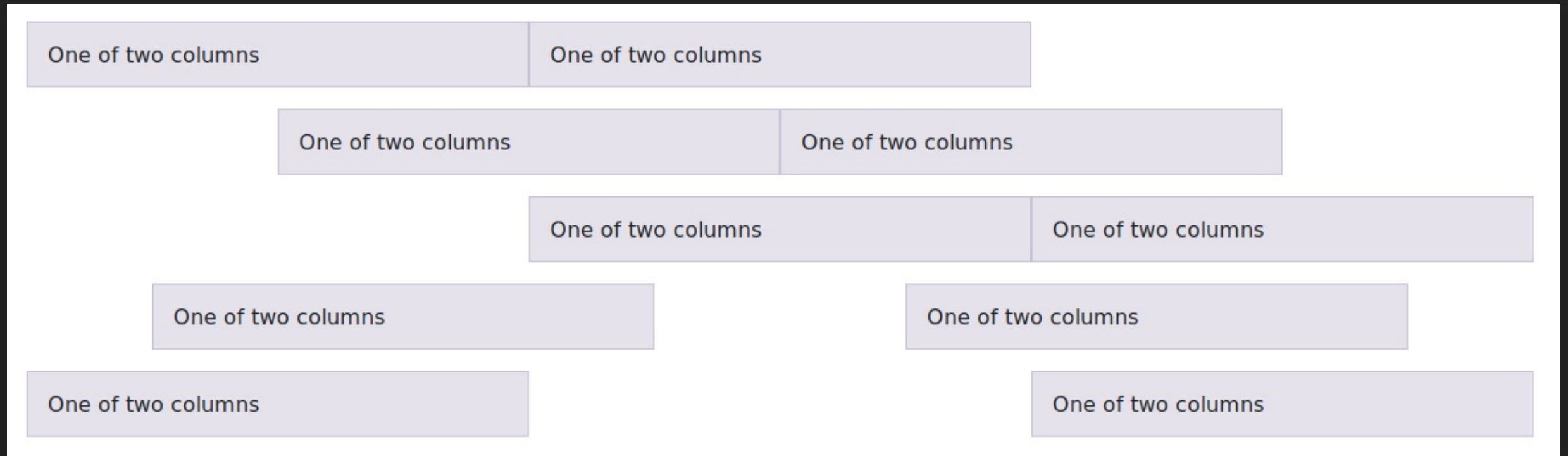
https://getbootstrap.com/docs/4.0/utilities/display/

We can make a smaller column for small screens with an offset to keep it centered. Or better: horizontal alignment (next slide…)

```
<div class="container">
  <div class="row">
    <div class="col d-sm-none d-md-block">
      1 of 3
    </div>
    <div class="col-md-8 col-sm-10 offset-sm-1 offset-md-0">
      2 of 3 (wider)
    </div>
    <div class="col d-sm-none d-md-block">
      3 of 3
    </div>
  </div>
</div>
```

http://getbootstrap.com/docs/4.0/layout/grid/#offsetting columns

HORIZONTAL ALIGNMENT

http://getbootstrap.com/docs/4.0/layout/grid/#horizontal alignment

```
<div class="container">
  <div class="row justify-content-start">
    <!-- 2 div .col-4 -->
  </div>
  <div class="row justify-content-center">
    <!-- 2 div .col-4 -->
  </div>
  <div class="row justify-content-end">
    <!-- 2 div .col-4 -->
  </div>
  <div class="row justify-content-around">
    <!-- 2 div .col-4 -->
  </div>
  <div class="row justify-content-between">
    <!-- 2 div .col-4 -->
```

VERTICAL ALIGNMENT

One of three columns	One of three columns	One of three columns

One of three columns	One of three columns	One of three columns

One of three columns	One of three columns	One of three columns

```
<div class="container">
  <div class="row align-items-start">
    <!-- 3 div .col -->
  </div>
  <div class="row align-items-center">
    <!-- 3 div .col -->
  </div>
  <div class="row align-items-end">
    <!-- 3 div .col -->
  </div>
</div>
```

ADVANCED USAGE

With sass it is easy to reuse bootstrap css in our own stylesheet

- Variables
- Functions (mixins)

CHANGE BOOTSTRAP'S BEHAVIOR

- Responsive breakpoints
- Number of columns
- Gutter width
- Container max width
- Colors

http://getbootstrap.com/docs/4.0/layout/grid/#variables

```scss
$grid-columns:      12;
$grid-gutter-width: 30px;
```

```scss
$grid-breakpoints: (
  // Extra small screen / phone
  xs: 0,
  // Small screen / phone
  sm: 576px,
  // Medium screen / tablet
  md: 768px,
  // Large screen / desktop
  lg: 992px,
  // Extra large screen / wide desktop
  xl: 1200px
);
```

```scss
$container-max-widths: (
  sm: 540px,
  md: 720px,
  lg: 960px,
  xl: 1140px
);
```

CUSTOM CSS CLASSES

Make your custom classes using `bootstrap-sass` functions (`mixins`).

http://getbootstrap.com/docs/4.0/layout/grid/#mixins

```
.example-content-main {
  @include make-col-ready();

  @include media-breakpoint-up(sm) {
    @include make-col(6);
  }
  @include media-breakpoint-up(lg) {
    @include make-col(8);
  }
}
```

```
<div class="example-content-main">
    <!-- content here -->
</div>
```

is the same as:

```
<div class="container"> <!-- .container set a width -->
    <div class="col-sm-6 col-lg-8">
        <!-- content here -->
    </div>
</div>
```

UTILITIES

Almost all CSS features are wrapped into bootstrap.

https://getbootstrap.com/docs/4.0/utilities

They provide:

- Mostly CSS classes
- Some SASS variables
- Some SASS mixins

BOOTSTRAP JS

- PopperJS and JQuery
- Mainly for components
- Per component documentation

CONCLUSION

- the wheel is not reinvented
- save time
- allows to have a workflow which would be difficult to set up manually
- bootstrap impose itself as a standard for grid system