Moden Frontend Development:

Mini project - 8 September 2017

Cristian C. Gorrin

Class: opbwu17fint

Project files: https://github.com/CristianGorrin/MFD-Mini-Project

Live demo: http://mfd.gorrin.dk

Table of Contents

Problem statement

This assignment will describe the technologies and tools used to create the e-commerce website prototype. The website scope will include the following pages: home-, collection-, product- and contact-page. I will also detailed my implementation process of various parts of the prototype.

Technologies and tools used

NPM

Npm is an abbreviation of Node package manager. Npm has large number of projects in it's repository this also includes frameworks, tools, components and more. Before installing anything whit Npm, run the command "npm init" to create package.json (this is the dependency list of the project). When using Npm to install a project in to yours, you will need to use the command "npm install". To uninstall a package replace the install with uninstall in the previous command.

Usage:

npm [install|uninstall] <*flag*> <*package*>

flag (optional): if -g is used the install will be in the global path and not the current directory.

package: path to the package.

For further information visit Npm documentations website[1].

Angular 4

Angular 4 is a frameworks for creating website. In this project the angular components was written in Typescript, but Javascript is also an opinion. Woking with Angular you will need to install Angular CLI. If you have Npm installed it will be a trivial task to get the CLI using the command "npm install -g @angular/cli". Then to create a Angular project run the command "ng new <projectname>" this will initialize and download the necessary files. The CLI has a build-in development server, that can be started by the command "ng serve". The build-in development server will automatic rebuild the web page, when a file is update. To get the production build, use the command "ng build –prod" (the result is save in ./dist/).

To start creating content for your web page in Angular, it will benefit you to know the various blueprints. The blueprints are used in the generate command "ng g <blueprint> <name>" (the command "ng g -h" to see available blueprints). The component blueprint creates a directory based on the name parameter containing the files HTML, CSS (this assignment is SASS), component.ts and component.spec.ts (unit testing). Furthermore registering the component in app.module.ts

1 https://docs.npmjs.com/

making it available in the project. There are many more blueprint available, these can be found in the wiki website[2].

Usage:

ng new <*project_name*> <*flag*>

project_name: name of the new Angular project.

flag (optional): if –style=sass then the project will use SASS instead of CSS files.

ng g <*blueprint*> <*name*>

blueprint [component|service|…]: selects the blueprint use.

name: target name.

ng serve

ng build <*flag*>

flag (optional): if –prod is used, then the build will be for production level.

For further information visit Angular documentations website[3].

Version control – Git

Git is a tool to keep track of changes in the code base. This is achieved by creating commits and branches to ideally going from one stable state to another. Version control in general is great for working in teams. Theres are CLI and GUI version of git, but the one used in this assignment is the CLI version from one of the Ubuntu's ppa[4].

To initialize git in your project, run the command "git init". This will create the directory ./.git/ where git keeps the version control history. Optionally you can add the file ./.gitignore[5] to let git ignore files and directorys in you're project based on patterns. To add a remote origin run "git remote set-url origin <url>" . This will allow for push from a local repository to a remote with the command "git push origin master".

I didn't use the branches functionality of git in this assignment. Well technically origin/master and local/master is two different branches. But they will contain the same code when push command is executed. So in this assignment the git work flow was staging the files, then committing them and afterwards pushing them to origin.

2 https://github.com/angular/angular-cli/wiki/generate
3 https://angular.io/docs
4 https://help.ubuntu.com/community/Repositories/Ubuntu
5 https://git-scm.com/docs/gitignore

Usage:

git add <*file*>

file: the file being add to the staging

git commit -m "<*message*>"

message: the decision for current commit

git push <*from*> <*to*>

from: the branch push from

to: the branch push to

For further information visit git-scm documentations website[6].

SASS

SASS is an extension that adds functionality to CSS. To use SASS your will need a preprocessing, that converts SASS into CSS. Some of the improvements in SASS is variables, nesting (the CSS selectors), partials, import and mixins. SASS allows for splitting the code in partials for reuse-ability by import where needed. This is done by using this code "@import '<path_to_partials> '". Mixins are really useful tools, they are basically function for CSS.

For further information visit SASS guide website[7].

CSS: Flexbox

CSS is a tool to describe how the html should be displayed. There are much that could be covered in CSS, but the focus here will be Flexbox. Flexbox is a display property, that gives a lot of control over the layout of the elements within it. For example justify-content controls the main axis (horizontal), and align-items controls cross axis (vertical). There is more property to affect how Flexbox will behave, a great article on this is "A Complete Guide to Flexbox"[8].

6 https://git-scm.com/docs
7 http://sass-lang.com/guide
8 https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Reset CSS from meyerweb[9]

Rest CSS is a way to have more even browser experience on a web side. It set some default values from the CSS. This tryes to make the starting point for the deferent browser the same. This will in theory have the web page looking the same on more browsers.

BEM

BEM is naming convention for CSS. BEM has three parts: blocks, elements and modifiers, this is used in a pattern like so “<block>__<element>--<modifier>”. Let’s say you want to select the email input in a sign up form, then the BEM name can be “signup-form__email-input”. Then the block is called signup-form, and the element is email-input. Now we also need an invalid state, when the user doesn’t give a valid e-mail. For this we can use the modifier part of BEM, then the selector will be “signup-form__email-input—invalid”.

Font awesome[10]

Font awesome is a collection of icons and animated assets. To start using font awesome, you need to download it, and then add a stylesheet link tag in to the html. After this you can use this html to add a microchip icon “”.

Implementation process

I started the project by initialize git and afterwards copy all assets from fronter. Then I got the web_artboards.psd exported as png files. I used collection, homepage and product png files to decide which components I need for the project. After a while I decided on some of the components, and additionally a naming convention for them. The naming convention is rather simple, and only has one rule, a prefix for the components. So the component with the prefix of page, is used in the routing as the starting component. The page component are build up by including the ones with the prefix part. The last prefix is partial, they are intended to be the support components for the others ones.

After the initial planing I create a new angular project, and set the style to SASS. Then I developed one component at a time starting with html structure and adding the SASS to style it. After it looked fine I add the bindings and used the directives such as ngfor. Then I add the functionality in the component class using the bindings. The last part of creating a component in my case, was making it responsive. I achieved it by adding media queries to the SASS based on the width.

I create a service (product.service.ts) to emulate getting the product information from the server. But at the moment it only uses hard-coded values. So when using the function GetProductInfo the result is one of three products based on the id parameter. It is done by dividing the id with the total number of products and getting the remainder of this operation. Then the result is used in a switch statement to determinant, which product information to return.

9 http://meyerweb.com/eric/tools/css/reset/index.html
10 http://fontawesome.io/

The slider for the homepage has a loop, that fires every second. The loop counts to twenty, and reset it self. At the first second it starts the progress bar. At the last second of the loop, it sets the next image and reset it self. Besides that, the loop has an override that can reset the counter and progress bar, as well set the next or previous image.

Creating the tabs for the product page, I used property binding to select the class in the html based on show object from the component. This hides the text not selected, and makes the current tab be active. To change the selected tab, you have to update the show objects boolean values. This is done by a click event binding on the tab, that it will result in it’s update.

Conclusion

Angular is a great framework to be working with. It has an extensive and organized documentation on it’s web side. The way an angular project is build, means that everyone can get op to speed quickly, assuming they have knowledge of angular. I also had a pleasant experience working Npm, it made getting started quite quickly and easy. This is likewise true for the Angular CLI, it has all the tool needed build in.