WEBPACK ES6 on Clients & Angular 1.x with ES6

1)

Create an empty project, and complete the tutorial given below, to get a basic understanding of:

- How to use webpack and to get the starting point for the following exercise.

https://scotch.io/tutorials/getting-started-with-webpack-module-bundling-magic

Hint: This tutorial misses a script-include in `index.html`. Add this line:

```
<script src="bundle.js"></script>
```
or it won't work.

2)

Dom Manipulation, using es2005 features and node-modules

*In this exercise we will import an external package **loadash** and use it for or js-client code. We will also include the bootstrap framework via npm, include it in our main js-file, and package all of it with webpack.*

Getting Started:

We will continue with the project made in part-1. In this project do the following:

Execute these npm-includes (observe and understand the **-d** in the first, but not in the second):

- `npm i file-loader --save-d`
- `npm file-loader i bootstrap –save`

Add this to your webpack.config.js file:

```
{
  test: /\.(png|jpg|jpeg|gif|svg|woff|woff2|ttf|eot)$/ ,
  loader: "file-loader"
}
```

And (important) remove the `exclude: /node_modules/` part from the css-loader section, since bootstrap now lives here.

Task:

Given this function (what do we call a function like this?):

```
function Person(fn,ln,s){
  this.firstName = fn;
  this.lastName = ln;
  this.favoriteSport = s;
}
```

And these data:

```
const persons = [
  new Person("Kurt", "Wonnegut","Socker"),
  new Person("Jan", "Peterson","Hockey"),
  new Person("Jane", "Peterson","Skating"),
  new Person("John", "Hansen","Socker"),
]
```

You should create a function, which given the data, will create (a bootstrap styled) table that should render like this:

First Name	Last Name	Favorite Sport
Kurt	Wonnegut	Socker
Jan	Peterson	Hockey
Jane	Peterson	Skating
John	Hansen	Socker

It should be a completely generic function that will take any kind of table including objects, use the object-property-keys to create the headers, and the data to populate the table. If object keys are camel-cased, they must rendered as sketched in the figure above

This might sound complicated at first glance, but we have already seen how we can use an arrays map-function to create a similar array, but now having all values encapsulated as a table row.

We will use the library loadash to help us with this exercise. Include it like this: `npm i loadash --save`

These two loadash-methods will be useful for this exercise:

- https://lodash.com/docs#keys
- https://lodash.com/docs#startCase

1. Implement the function (in `main.js`)
2. Add a `<div> with id="my-table"` to *index.html*
3. Test/use the function like this:

```
const table = makeTable(persons);
document.getElementById('my-table').innerHTML = table;
```

4. Extend the function with an extra argument to include only the sport provided in the argument

3) Using Angular 1.x with Webpack, and ES6 features

This exercise will give you a realistic example of how to develop modern SPA's with Webpack and es 2015.

We will use this seed (workflow) for the exercise: https://github.com/preboot/angular-webpack

I recommend you use this seed for all Angular 1.x you do for the rest of this course.

The seed has a supplementary exercise which is the exercise you should complete:

http://angular-tips.com/blog/2015/06/using-angular-1-dot-x-with-es6-and-webpack/