Lab 3. Using Webpack-dev-server

In this lab you will install all required prerequisites and
test the setup of webpack-dev-server.

> duration: 30 minutes

Step 1. Install the requirements

Create or navigate to the directory named 'webpack_labs'.
Install NodeJS from it's website and install webpack-dev-server using

```
$ sudo npm install webpack-dev-server -g
$ sudo npm install webpack -g
```

Run the webpack-dev-server and test to see if all went well.

Step 2. Change modules while running the server

The webpack-dev-server watches for changes of the files and reloads them
when changed.

Copy the starterfiles from the _starterfiles folder into the root of
the directory.

Start webpack-dev-server and navigate to
http://localhost:8080/webpack-dev-server

Open the chrome devtools and inspect the output. Check that the console logs
are written.

Go back to your code editor and change the test.js message to

```
var test2 = require('./test2.js')
console.log("testing 1 2 3");
console.log(test2);
```

Note that just the logmessage is changed here. Save your changes and inspect the
ouput. The content should not be automatically appear.

Change the content back to the previous state:

```
var test2 = require('./test2.js')
console.log("test");
console.log(test2);
```

Now navigate to http://localhost:8080/webpack-dev-server and click the link to the magic html bundle.
Not that there is a navigation bar in the top of the screen.

Inspect the dev-tools from Chrome to see the log output.

Now change the message as we did earlier and go back to the browser. Note that the messages now do appear, the javascripts are rebuilt and reloaded upon save.

Inline the sync

Stop and restart the webpack-dev-server using the --inline command line switch.
Navigate to the root of the appication using:
http://localhost:8080/

Change the scripts to show another message and test to see if automatic reloading and refreshing also works in the 'normal' page.

-= End of lab =-