Lab 4. Webpack merge

In this lab you will use different setups to split up the configuration into multiple settings distributed over multiple files

duration: 30 minutes

prerequisites

You need to have a package.json file in the root of the target folder and you need to have webpack installed globally for these labs to succeed. To ensure the best possible results, delete all previously generated folders for dev and production bundles.

Step 1. Install the requirements

Navigate into the earlier created directory 'webpack_labs'.
Install webpack-merge, if not done already, using

```
$ npm install webpack-merge
```

create a file 'test.js' inside your current folder and write the following code, only if it does not already exists.

```
console.log('test');
```

Save the results.

Step 1. Use multiple files for multiple setups

Navigate into the earlier created directory 'webpack_labs' and create or open the file 'webpack.config.js'
inside your current folder. Write the following code into this file:

```
module.exports = require('./config/webpack.dev.js');
```

Save the file.

Create a new folder named 'config' and add the file 'webpack.dev.js' to this folder. Write the following code into this file:

```
var webpackMerge = require('webpack-merge');
var commonConfig = require('./webpack.common.js');

module.exports = webpackMerge(commonConfig, {
  output: {
    path: process.cwd() + '/dev',
    filename: 'bundle.js'
  },
});
```

Save the file.

In the same folder, create a file called 'webpack.common.js' and fill in the following code

```
module.exports =  {
        context: process.cwd(),
        entry: "./test.js",
        output: {
            filename: "bundle.js"
        }
    }
```

Save the file.

Lastly, create a file called 'webpack.prod.js' in the same folder and give it the following contents:

```
var webpack = require('webpack');
var webpackMerge = require('webpack-merge');
var commonConfig = require('./webpack.common.js');

module.exports = webpackMerge(commonConfig, {
        output: {
            path: process.cwd() + "/prod",
            filename: "bundle.min.js"
        },
        plugins: [
            new webpack.optimize.UglifyJsPlugin({minimize: true})
        ]
    });
```

Save the file.

Check to see if all files are saved and from the command line, execute the following command:

```
webpack
```

Check if the bundle.js is generated inside the dev folder.

Step 2. Switch from environment

Open the 'webpack.config.js' file in the root of the folder and change the current line to:

```
modue.exports = require('./config/webpack.prod.js');
```

Save the file and rerun the webpack command.
Check if there is a generated minified bundle in the prod folder of your project.