



OKTOBER 2024



webpack

MANUAL

Guide til at installere, og konfigurere Webpack. I denne guide lærer du også at transpile SASS til CSS, lave bundles med PostCSS og sørger for at der bliver lavet sourcemaps



WEBUDVIKLING

Af Clara & Heva

Installation og konfiguration

Webpack bruges til at samle, transformere og optimere filer, ved at bundle projektets ressourcer så som css, javascript og billeder. Det er især nyttigt i store projekter.

For at installere webpack, er det allerførste man skal tage stilling til om man har node.js installeret til sit vs code. Derefter kan man starte med at installere webpack ved at oprette en lokal mappe som projektet skal være i og sørge for at webpack er aktiveret i mappen (i kontrollen). Kontrollen finder man ved at trykke på "terminal - new terminal" og så skrive `npm init -y`. Det skulle gerne se sådan ud i terminalen:


```
qlara@Claras-MacBook-Air my-webpack-project % npm init -y
```

Dette vil tilføje denne information i terminalen:

```
Wrote to /Users/qlara/Desktop/coding shit/automatisering/webpack-tutorial-webpack-tutorial-article/my-web
-project/package.json:

{
  "name": "my-webpack-project",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

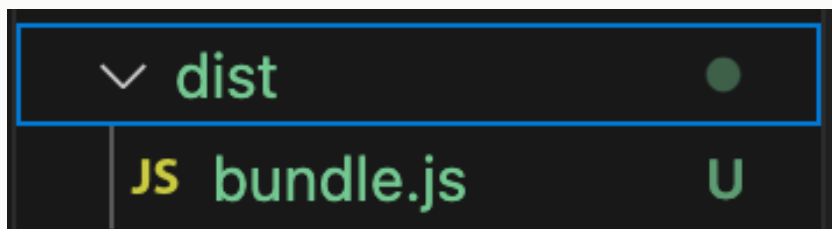
Nu skal vi tilføje noget konfiguration. Det gør du ved at tilføje nogle filer og noget kode. Lav en `index.html` fil og tilføj en sådan fil:

 **webpack.config.js**

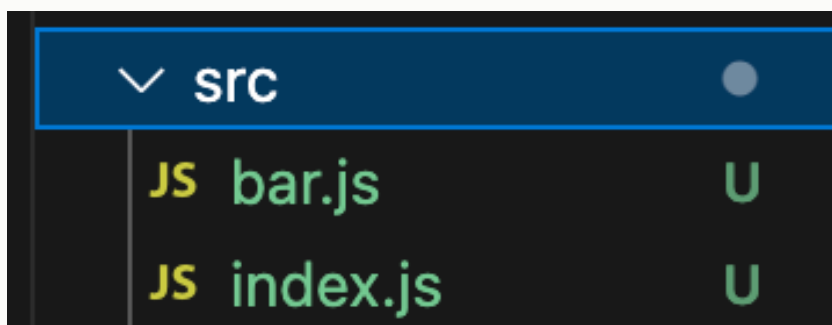
Inde i denne fil, tilføj denne kode:

```
webpack.config.js > ...
1  const path = require('path');
2
3  module.exports = {
4    mode: 'development',
5    entry: './src/index.js',
6    output: {
7      filename: 'bundle.js',
8      path: path.resolve(__dirname, 'dist'),
9    },
10 };
11
```

Nu skal der lige tilføjes et output directory som webpack skal bruge til de bundlede filer. Det ser sådan her ud:



Så skal du tilføje endnu en mappe med js filer i. Det skal se sådan ud:



Så tilføj lidt kode i bar.js:

```
1  export default function bar() {  
2    //  
3  }
```

Og lidt i index.js:

```
src > JS index.js  
1  import bar from './bar.js';  
2  
3  bar();
```

Tillykke, nu er du klar til at installere webpack! Gå tilbage til terminalen og kørs dette:

```
% npm install --save-dev webpack webpack-cli
```

Hvis webpack er installeret, ser det sådan ud:

```
asset bundle.js 4.05 KiB [compared for emit] (name: main)  
runtime modules 670 bytes 3 modules  
cacheable modules 77 bytes  
  ./src/index.js 35 bytes [built] [code generated]  
  ./src/bar.js 42 bytes [built] [code generated]  
webpack 5.95.0 compiled successfully in 46 ms  
qlara@Claras-MacBook-Air webpack-tutorial-webpack-tutorial-article %
```

Transpiling

Nu skal du lære hvordan man transpiler SASS til CSS. Dette bliver brugt til at oversætte SASS til CSS, uden at bruge watch funktionen.

For at installere det skal man bruge en af webpacks funktioner som er "loadere" der kan forbehandle filer og der er forskellige loaders inden for templates, frameworks, styling mm. For at transpile SASS til CSS skal vi bruge styling loaders. For at gøre dette skal vi installere nogle flere pakker, så derfor skal vi tilbage til terminalen og skrive dette:

```
qlara@Claras-MacBook-Air webpack-tutorial-webpack-tutorial-article % npm install --save-dev sass-loader sass css-loader style-loader
```

Hvis det blev installeret, skulle der gerne stå dette i terminalen:

```
added 32 packages, and audited 150 packages in 2s

26 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Nu skal loaderne tilføjes til den konfiguration som du lavede tidligere i webpack.config.js. Det kunne se sådan ud:

```
2   const path = require('path');
3
4   module.exports = {
5     mode: 'development',
6     entry: './src/index.js',
7     output: {
8       filename: 'bundle.js',
9       path: path.resolve(__dirname, 'dist'),
10    },
11    module: {
12      rules: [
13        {
14          test: /\.scss$/,
15          use: [
16            'style-loader',
17            'css-loader',
18            'sass-loader',
19          ],
20        },
21      ],
22    },
23  };
```

Nu skal du tilføje noget SASS til dit projekt for at se om det virker (husk at linke din css mappe i index). Lav en SCSS mappe som matcher test konfigurationen. Det vil sige vi har kaldt mappen scss i konfiguration så det skal vi også kalde den i direktoratet.

```
scss > style.scss > body
1  $body-color: red;
2
3  body {
4    color: $body-color;
5  }
```

Så skal vi lige huske at importere SASS filen også inde i index.js:

```
src > JS index.js
1  import bar from './bar.js';
2
3  bar();
4
5  import './scss/style.scss';
```

Og så køre webpage i terminalen og så skulle den være der.

npx webpack

PostCSS Bundling

Nu skal vi lave bundles med PostCSS.

PostCSS bundling bruges til at behandle og optimere CSS ved hjælp af et bredt udvalg af plugins. PostCSS er et værktøj, der giver dig mulighed for at transformere CSS med JavaScript-baserede plugins, hvilket gør det muligt at tilføje nye funktioner, optimere, og forbedre din CSS-kode under byggeprocessen.

il det skal vi igen bruge en style loader, denne gang postcss loaderen. Det gør vi på en lignende måde som de andre gange. Tilbage til terminalen!

```
qlara@Claras-MacBook-Air webpack-tutorial-webpack-tutorial-article % npm install --save-dev postcss-loader postcss

added 25 packages, and audited 175 packages in 2s

30 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
qlara@Claras-MacBook-Air webpack-tutorial-webpack-tutorial-article %
```

Og igen skal vi tilføje noget kode til konfigurationen (webpack.config.js). Det skulle gerne se sådan ud:

```
10  module: {
11    rules: [
12      {
13        test: /\.scss$/,
14        use: [
15          'style-loader',
16          'css-loader',
17          'sass-loader',
18        ],
19      },
20      {
21        test: /\.css$/i,
22        use: [
23          'style-loader',
24          'css-loader',
25          {
26            loader: 'postcss-loader',
27            options: {
28              postcssOptions: {
29                plugins: [
30                  [
31                    'postcss-preset-env',
32                    {
33                      // Options
34                    },
35                  ],
36                ],
37              },
38            },
39          ],
40        ],
41      },
42    ],
43  },
44  };
```



Og så kører du igen webpack

```
npx webpack
```

Sourcemaps

Til sidst skal vi lære hvordan man sørger for at der bliver lavet sourcemaps. Sourcemaps giver en måde at forbinde den optimerede kode, som browseren kører, med den oprindelige kildekode man har skrevet. Dette gør det lettere at finde fejl og fejlrette kode, der er blevet ændret af bundling- eller komprimeringsprocesser. Det er især smart når man bruger et eksternt bibliotek.

For at installere det, skal vi igen installere en loader - denne gang sourcemaps loaderen. Det foregår i terminalen og ser sådan ud:

```
qlara@Claras-MacBook-Air webpack-tutorial-webpack-tutorial-article % npm i -D source-map-loader
added 3 packages, and audited 178 packages in 855ms

31 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
qlara@Claras-MacBook-Air webpack-tutorial-webpack-tutorial-article %
```

Ligesom de andre gange skal vi nu tilføje kode til konfigurationen, men denne gang er der lidt ekstra at tilføje. Først tilføj dette:

```
1  const path = require('path');
2
3  module.exports = {
4    mode: 'development',
5    entry: './src/index.js',
6    output: {
7      filename: 'bundle.js',
8      path: path.resolve(__dirname, 'dist'),
9    },
10   devtool: 'source-map',
11   module: {
12     rules: [
13       {
14         test: /\.scss$/,
15         use: [
16           'style-loader',
17           'css-loader',
18           'sass-loader',
19         ],
20       },
21     ],
22   },
23 }
```



Og så tilføj dette:

```
43     {  
44       test: /\.js$/,  
45       enforce: 'pre',  
46       use: ['source-map-loader'],  
47     },  
48   ],  
49 },  
50 };  
51
```

Til sidst kør webpack og du er klar.

```
npx webpack
```