Web Application Development for Industrial Engineers

การพัฒนาเวปแอปพลิเคชันสำหรับวิศวกรอุตสาหการ

Module bundler

What is a module bundler?

- Module bundlers are tools to bundle JavaScript modules into a single JavaScript files that can be executed in the browser.

Bundling?

- In a large application, you usually split JavaScript code into multiple files.
  - Using a `module` system.
  - This makes code more maintainable.
- A bad way is to add JavaScript files into html via script tags.

```
<head>
  <script src="/src/this.js"></script>
  <script src="/src/is.js"></script>
  <script src="/src/a.js"></script>
  <script src="/src/bad.js"></script>
  <script src="/src/way.js"></script>
</head>
```

Why is this bad?

- Each file requires a separate http requests.
  - 5 round trip requests in order to get your application started.
- You have to make sure the order is correct.
- You have to prevent naming conflicts between "files".
- How do we get rid of unused files?

Other problems

- How do we maintain the correct links to assets likes images, fonts, css files?
- How can we make sure our code run in most browsers?
- How can we minify the code?
- We can benefit from a tool that analyze the code.

Module bunlders solve these kinds of problems.

Popular bundlers

- Webpack
  - We will use this.
- Rollup
- Parcel

Webpack installation

```
mkdir webpack-demo
cd webpack-demo
npm init -y
npm install webpack webpack-cli --save-dev
```

Entry and output

- Create `src` folder and `index.js` file
- Create `dist` folder and `index.html` file

```
📦root
 ┣ 📂dist
 ┃ ┗ 📜index.html
 ┗ 📂src
 ┃ ┗ 📜index.js
```

Entry and output

- In `index.html`, add to the header

```
<script src="main.js" defer></script>
```

Webpack configuration

- Create `webpack.config.js`

```
📦root
 ┣ 📂dist
 ┃ ┗ 📜index.html
 ┣ 📂src
 ┃ ┗ 📜index.js
 ┗ 📜webpack.config.js 👈👈👈👈👈👈👈👈
```

- `webpack.config.js`

```
const path = require('path');

module.exports = {
  entry: './src/index.js',
  output: {
    filename: 'main.js',
    path: path.resolve(__dirname, 'dist'),
  },
  devtool: 'eval-source-map',
  mode: 'development',
};
```

VSCode configuration

- create `jsconfig.json`

```
📦root
 ┣ 📂dist
 ┃ ┣ 📜index.html
 ┃ ┗ 📜main.js
 ┣ 📂src
 ┃ ┗ 📜index.js
 ┣ 📜jsconfig.json  👈👈👈👈👈👈👈👈
 ┗ 📜webpack.config.js
```

- `jsconfig.json`

```
{
  "compilerOptions": {
    "baseUrl": "."
  },
  "include": ["src"],
  "exclude": ["node_modules"]
}
```

Write some code

- `index.js`

```
const div = document.createElement('div');
const p1 = document.createElement('p');
p1.textContent = 'Hello from Webpack.';
div.appendChild(p1);
document.body.append(div);
```

Building code

- In `terminal`

```
npx webpack --config webpack.config.js
```

- or

```
npx webpack
```

- If a webpack.config.js is present, the webpack command picks it up by default.
- Inspect the `dist` folder to see the bundled code.

NPM scripts

- `package.json`

```
{
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack"
  }
}
```

- Don't forget the extra `,`.

NPM scripts

- Now we can type in the `terminal`.

```
npm run build
```

Import

- Create a file `utilities.js`

```
📦root
 ┣ 📂dist
 ┃ ┣ 📜index.html
 ┃ ┗ 📜main.js
 ┣ 📂src
 ┃ ┣ 📂lib
 ┃ ┃ ┗ 📜utilities.js 👈👈👈👈👈👈👈👈
 ┃ ┗ 📜index.js
 ┣ 📜jsconfig.json
 ┗ 📜webpack.config.js
```

- Add the following code to `utilities.js`.

```
function getMessage() {
  return 'Module is working, nice!';
}

const data = {
  name: 'Tim',
  age: 20,
};

export { getMessage, data };
```

- `index.js`

```
import { getMessage } from './lib/utilities';
import { data } from './lib/utilities';

const div = document.createElement('div');
const p1 = document.createElement('p');
p1.textContent = 'Hello from Webpack.';
div.appendChild(p1);

const p2 = document.createElement('p');
p2.textContent = getMessage();
div.appendChild(p2);

console.log(data);

document.body.append(div);
```

- Check the page and console.

Loading CSS

- In order to import a CSS file from within a JavaScript module, you need to install and add the `style-loader` and `css-loader` to your module configuration:

```
npm install --save-dev style-loader css-loader
```

- Modify `webpack.config.js`

```
module.exports = {
  // ...
  module: {
    rules: [
      {
        test: /\.css$/i,
        use: ['style-loader', 'css-loader'],
      },
    ],
  },
};
```

- Create `./src/style.css`

```
📦root
 ┣ 📂dist
 ┃ ┣ 📜index.html
 ┃ ┗ 📜main.js
 ┣ 📂src
 ┃ ┣ 📂lib
 ┃ ┃ ┗ 📜utilities.js
 ┃ ┣ 📜index.js
 ┃ ┗ 📜style.css 👈👈👈👈👈👈👈
 ┣ 📜jsconfig.json
 ┗ 📜webpack.config.js
```

- Add in `style.css`

```css
.main {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 1.5rem;
}
```

- Modify `index.js`

```
import { getMessage } from './lib/utilities';
import { data } from './lib/utilities';
import './style.css';

// ...

div.classList.add('main');

document.body.append(div);
```

- Build by running `npm run build` in the `terminal`.
- To see what webpack did, inspect the page and look at the page's head tags.

Loading Images

- Modify `webpack.config.js`

```
module.exports = {
  module: {
    rules: [
      {
        test: /\.css$/i,
        use: ['style-loader', 'css-loader'],
      },
      // Add from here ----->
      {
        test: /\.(png|svg|jpg|jpeg|gif)$/i,
        type: 'asset/resource',
      },
    ],
  },
};
```

- Add images.

```
📦root
 ┣ 📂dist
 ┃ ┣ 📜index.html
 ┃ ┗ 📜main.js
 ┣ 📂src
 ┃ ┣ 📂assets
 ┃ ┃ ┣ 📜bg.jpg 👈👈👈👈👈👈👈
 ┃ ┃ ┗ 📜logo.png 👈👈👈👈👈👈👈
 ┃ ┣ 📂lib
 ┃ ┃ ┗ 📜utilities.js
 ┃ ┣ 📜index.js
 ┃ ┗ 📜style.css
 ┣ 📜jsconfig.json
 ┗ 📜webpack.config.js
```

- modify `style.css`

```css
.main {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 1.5rem;
}

/* Add here */
.main {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  height: 100vh;
  background: url('./assets/bg.jpg');
  background-repeat: no-repeat;
  background-size: cover;
}
```

- Modify `index.js`

```
import { getMessage } from './lib/utilities';
import { data } from './lib/utilities';
import './style.css';
import logoPath from './assets/logo.png';

// ...

const logo = new Image(200);
logo.src = logoPath;
div.appendChild(logo);

document.body.append(div);
```

- Build by running `npm run build` in the `terminal`.
- You can see that we can now use images in both `CSS` and `JavaScript` files.

Watch mode

- Rebuild automatically when there is a file change.

```
{
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack",
    "watch": "webpack --watch"
  }
}
```

Using webpack-dev-server

- We can start the server by installing the package.

```
npm install --save-dev webpack-dev-server
```

- Modify `webpack.config.js`

```
module.exports = {
  // ...
  devServer: {
    static: './dist',
    hot: true,
  },
};
```

- Modify `package.json`

```
{
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack",
    "watch": "webpack --watch",
    "start": "webpack serve --open"
  }
}
```

- And simply run:

```
npm start
```