

Webpack

Подготовка проекта и установка webpack

1. В каталог проектов копируем каталог `05_jsa`
В этом каталоге открываем `GitBash`

2. Установка webpack глобально (делается один раз – если этих пакетов еще нет на машине)

```
npm i -g yarn webpack webpack-cli webpack-dev-server cross-env
```

3. Создаем проект. В `GitBash` набираем
`yarn init -y`

В каталоге проекта будет создан файл `package.json`

Затем выполняем команду

```
yarn add --dev webpack webpack-cli webpack-dev-server  
cross-env
```

4. В `GitBash` набрать
`webpack`

Будет создан каталог `dist/` и в нем файл `main.js`

Если нас не устраивает название скомпилированного файла
`main.js`

то при компиляции надо указать параметры
`webpack -output dist/bundle.js`

Также в сообщениях при компиляции мы увидим

```
WARNING in configuration
  The 'mode' option ...
```

Это означает что при компиляции нужно указывать режим работы

```
webpack --mode development --output dist/bundle.js
```

mode может быть `development` или `production`

5. Если теперь внести изменения в файл `src/index.js` то для того чтобы файл `dist/bundle.js` перекомпилировался надо опять запустить команду в п.4

Чтобы файл `dist/bundle.js` перекомпилировался при изменении файла `src/index.js` - запускаем компиляцию с отслеживанием изменений

```
webpack --mode development --watch --output dist/bundle.js
```

6. Чтобы не набирать длинных команд в `Bash` в файл `package.json` добавляем секцию `scripts`

`package.json`

```
"scripts" : {  
  "watch": "webpack --mode development --watch  
            --output dist/bundle.js "  
}
```

В `Bash` набираем команду

```
yarn watch
```

или `(npm run watch)`

7. В корне проекта создаем файл конфигурации `webpack.config.js`
<https://webpack.js.org/concepts/#entry>

`webpack.config.js`

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

module.exports = {
  entry: './src/index.js',
  output: {
    path: path.resolve(__dirname, 'dist'),
    filename: 'bundle.js'
  }
};
```

Корректируем файл `package.json`

`package.json`

```
"scripts" : {
} "build": "./node_modules/.bin/webpack --mode production",
  "watch": " ./node_modules/.bin/webpack --mode development
            --watch --output dist/bundle.js"
```

Bash -> `yarn watch`

8. Для разработки лучше использовать **webpack-dev-server**
<https://webpack.js.org/configuration/dev-server>

Для этого в файл **webpack.config.js** добавляем секцию

webpack.config.js

```
...  
module.exports = {  
  ...  
  devServer: {  
    contentBase: path.join(__dirname, "dist"),  
    watchContentBase: true,  
    open: true  
  }  
};
```

Корректируем файл **package.json**

package.json

```
"scripts" : {  
  ...  
  "watch": "./node_modules/.bin/webpack-dev-server  
            --mode development --open"
```

Bash -> **yarn watch**

- устанавливаем плагин

<https://github.com/webpack-contrib/copy-webpack-plugin>

Bash -> `yarn add -D copy-webpack-plugin`

- ВНОСИМ добавления в webpack.config.js

webpack.config.js

```
...  
const CopyWebpackPlugin = require('copy-webpack-plugin');  
  
module.exports = {  
  ....  
  plugins: [  
    new CopyWebpackPlugin([  
      {from: 'src/index.html'}  
    ])  
  ]  
};
```

Bash -> yarn watch

9. Начинаем работать с модулями ES6

В файл `src/components/App.js` добавляем код

`src/components/App.js`

```
export default class App {  
  constructor(greet) {  
    this.greet = greet;  
  }  
  
  showGreet(el) {  
    document.querySelector(el)  
      .innerHTML = `

# ${this.greet}</h1>`; } }


```

`src/index.js`

```
import App from './components/app';  
const app = new App("Hello webpack");  
app.showGreet('.container');
```

10. Нам для webpack нужны некоторые плагины
<http://babeljs.io/docs/setup/#installation>

Bash -> `yarn add -D babel-loader babel-core babel-preset-env`

```
module.exports = {  
  ....  
  module: {  
    rules: [  
      {  
        test: /\.js$/,  
        exclude: /node_modules/,  
        loader: "babel-loader"  
      }  
    ]  
  }  
  ....  
};
```

webpack.config.js

Создаем в корне проекта файл `.babelrc`

```
{  
  "presets": ["env"]  
}
```

`.babelrc`

Bash -> `yarn watch`

11. Нам нужно скомпилировать `src/scss/style.scss` и перенести его

в `dist/css/style.css`

Это делается с помощью плагинов

<https://webpack.js.org/guides/asset-management/#loading-css>

<https://webpack.js.org/loaders/style-loader>

<https://webpack.js.org/loaders/sass-loader>

<https://github.com/webpack-contrib/extract-text-webpack-plugin>

<https://github.com/webpack-contrib/mini-css-extract-plugin>

Использование плагинов смотри в файле

`read.txt` -> секция `STYLES`

—

—

—

—

—

—

webpack.config.js

```
...  
module.exports = {  
  ...  
};
```