Цель урока:

1. Вы узнаете, что такое Webpack и для чего он используется в JavaScript-проектах
2. Поймете на сколько важен Webpack для работы с современными приложениями
3. Научитесь подключать Webpack в проект и запускать ваш веб-сайт на локальном сервере

Содержание урока:

1. Сборщики проектов. Установка NodeJS и NPM
2. Rollup. Установка rollup и базовая конфигурация проекта
3. Rollup. Установка дополнительных пакетов для сборки Rollup
4. Webpack. Установка и конфигурация Webpack

Дополнительные материалы:

1. Webpack - Полный курс за 3 часа [2020] - https://youtu.be/eSaF8NXeNsA
2. Документация Webpack: https://webpack.js.org/guides/...
3. Документация Rollup: https://rollupjs.org/guide/en/

Обратите внимание!

На 12:43 в конфигурации webpack-dev-server вместо параметра **contentBase** необходимо использовать **static** (документация)

Если вы используете current версию Node.js и у вас появляются ошибки при использовании webpack-dev-server, то попробуйте переустановить Node.js на версию LTS, удалить папку node_modules и еще раз установить зависимости (npm install)

Задание #1

Вас посадили на проект, где вы являетесь главным разработчиком. Код еще никто не писал и поэтому ваша задача состоит в том, чтобы с пустой папки сделать базовую конфигурацию проекта.

Для этого используйте **rollup.js**. Теперь в папке проекта создайте 3 файла: index.html, index.css и index.js. Добавьте в **index.html** базовую разметку html. После создайте файл **package.json** при помощи команды **npm init -y**. Также в index.js добавьте строчку кода **console.log('Hello World!');**

Теперь руководствуясь документацией rollup.js, видео с уроком, установите **rollup** как пакет и создайте конфигурационный файл **rollup.config.js**. В **rollup.config.js** добавьте

базовую конфигурацию для запуска проекта. Для этого хватит свойств input, output и watch. Никаких плагинов пока не добавляйте.

После в файле package.json создайте 2 команды "**build"** и **“start”**.

```
"build": "rollup -c",
"start": "rollup -c -w"
```

Затем запустите команду **“start”**. У вас должен сгенерировать JS-файл, название которого вы указывали в свойстве **“output**” в файле **rollup.config.js**. Если файл не сгенерировался, то, возможно, вы сделали что-то не так и у вас возникла ошибка. В данном случае вам нужно найти ее и исправить.

Задание #2

Вы молодец! Вы инициализировали проект. Но это еще не все, так как нужно еще учесть следующие вещи:

1. Подключение стилей;
2. Работа с картинками;
3. Поддержка кода другими браузерами;
4. Запуск проект на локальном сервере (localhost).

Поэтому сейчас вы этим и займетесь. **Необходимо установить** несколько плагинов как **devDependencies**.

1. Для подключения стилей используйте “rollup-plugin-styles”: https://www.npmjs.com/package/rollup-plugin-styles
2. Для работы с картинками “rollup-plugin-img”: https://www.npmjs.com/package/rollup-plugin-img
3. Для того, чтобы переводить JavaScript в код, который поддерживают даже старые браузеры, используйте “ @rollup/plugin-babel”: https://www.npmjs.com/package/@rollup/plugin-babel
4. Для запуска проекта на локальном сервере вам понадобятся плагины “rollup-plugin-serve” (https://www.npmjs.com/package/rollup-plugin-serve) и “rollup-plugin-livereload” (https://www.npmjs.com/package/rollup-plugin-livereload).

После создайте папку **assets** и добавьте в нее любую картинку с изображением языка программирования JavaScript. Теперь в **index.js** файле вам необходимо добавить в DOM два элемента:

1. h1. В данном элементе должна быть надпись “I love JavaScript”;
2. img. Данный элемент должен отображать картинку из папки assets.

Кроме этого в **index.css** добавьте любые стили на ваш вкус. Главное, чтобы данный файл не был пустым.

После запустите команду **“start”** и при помощи тега script подключите сгенерированный пакетом **rollup.js** js-файл в **index.html**.

В итоге у вас должна отобразиться картинка, примениться все стили и должен открыться локальный сервер. Если что-то пошло не так то, возможно, у вас возникла ошибка. В данном случае вам нужно найти ее и исправить.

Задание #3

С инициализацией проекта на **rollup.js** вы справились успешно. Вся команда разработки довольна вами, потому что все теперь могут начать реализовывать функционал.

Ваш начальник понял, что вы достаточно хороший разработчик и поэтому просит вас помочь инициализировать еще один проект. Только дело в том, что требование у заказчика стоит следующее: необходимо использовать только **webpack.js**.

Сейчас ваша задача состоит в том, чтобы из пустой папки, инициализировать проект при помощи **webpack.js**.

В папке проекта создайте 3 файла: index.html, index.css и index.js. **Добавьте** в **index.html** базовую разметку html. После создайте файл package.json при помощи команды npm init -y. Также в **index.js** добавьте строчку кода **console.log('Hello World!');**

Теперь руководствуясь документацией webpack.js (https://webpack.js.org/guides/installation/), видео с уроком, установите webpack как пакет и создайте конфигурационный файл **webpack.config.js**.

В **webpack.config.js** добавьте базовую конфигурацию для запуска проекта. Для этого хватит свойств entry, output и mode. Никаких плагинов пока не добавляйте.

После в файл package.json добавьте команду **"start"**.

```
"start": "webpack --config webpack.config.js"
```

Затем запустите команду **"start"**. У вас должен сгенерировать JS-файл, название которого вы указывали в свойстве **"output"** в файле **webpack.config.js**. Если файл не сгенерировался, то, возможно, вы сделали что-то не так и у вас возникла ошибка. В данном случае вам нужно найти ее и исправить.

Задание #4

Вы молодец! Даже webpack.js вам по плечу! Сейчас же нужно разобраться с теми же пунктами, которые были при использовании rollup.js:

1. Подключение стилей;
2. Работа с картинками;
3. Поддержка кода другими браузерами;
4. Запуск проекта на локальном сервере (localhost).

Необходимо установить несколько плагинов как **devDependencies** и добавить немного кода в конфигурационный файл.

1. Для подключения стилей используйте "css-loader" (https://www.npmjs.com/package/css-loader) и "style-loader" (https://www.npmjs.com/package/style-loader).

2. Для работы с картинками добавьте в **webpack.config.js** данный код:

```
module: {
  rules: [
    {
      test: /\.(?:ico|gif|png|jpg|jpeg)$/i,
      type: 'asset/resource',
    },
    {
      test: /\.(woff(2)?|eot|ttf|otf|svg|)$/,
      type: 'asset/inline',
    },
  ],
},
```

3. Для того, чтобы переводить JavaScript в код, который поддерживают даже старые браузеры, используйте **“babel-loader”** (https://www.npmjs.com/package/babel-loader).
4. Для запуска проекта на локальном сервере вам понадобится плагин **“webpack-dev-server”** (https://www.npmjs.com/package/webpack-dev-server). Для его конфигурации добавьте данный код в webpack.config.js:

```
devServer: {
  open: true,
  compress: true,
  port: 4000,
},
```

Также, для того, чтобы webpack обрабатывал HTML и автоматически подключал в него JavaScript, необходимо установить **html-webpack-plugin** (https://www.npmjs.com/package/html-webpack-plugin). Добавьте данный код в webpack.config.js для конфигурации данного пакета:

```
plugins: [
  new HtmlWebpackPlugin({
    template: 'index.html'
  }),
]
```

После в файл **index.js** добавьте код, который вы писали при использовании **rollup.js**. Также не забудьте добавить картинку в **assets** и скопировать файл со стилями **index.css**.

Кроме этого замените команды в **package.json** на следующие:

```
"start": "webpack serve",
"prod": "webpack"
```

Затем запустите команду **“start”**. Если вы все сделали правильно и следовали инструкциям в ссылках по установке плагинов, то у вас должен открыться локальный сервер. Также у вас должна отобразиться картинка из **assets** и примениться все стили. Если что-то пошло не так то, возможно, у вас возникла ошибка. В данном случае вам нужно найти ее и исправить.