AURIMAS VILČINSKAS

@Aurimas4

aurimas4

paysera

ASSETS

Today lot of code goes to client side

Projects has lof of assets

Assets depends on others assets

PRODUCTS.HTML

```html
<!doctype html>
<html>
  <head>
    <link rel="stylesheet" href="/style/dashboard.css"/>
    <link rel="stylesheet" href="/style/product.css"/>
    <link rel="stylesheet" href="/style/products.css"/>
  </head>
  <body>
    <!-- ... -->
    <script type="text/javascript" src="/js/main.js"></script>
    <script type="text/javascript" src="/js/products.js"></scrip
    <script type="text/javascript" src="/js/product.js"></script
  </body>
</html>
```

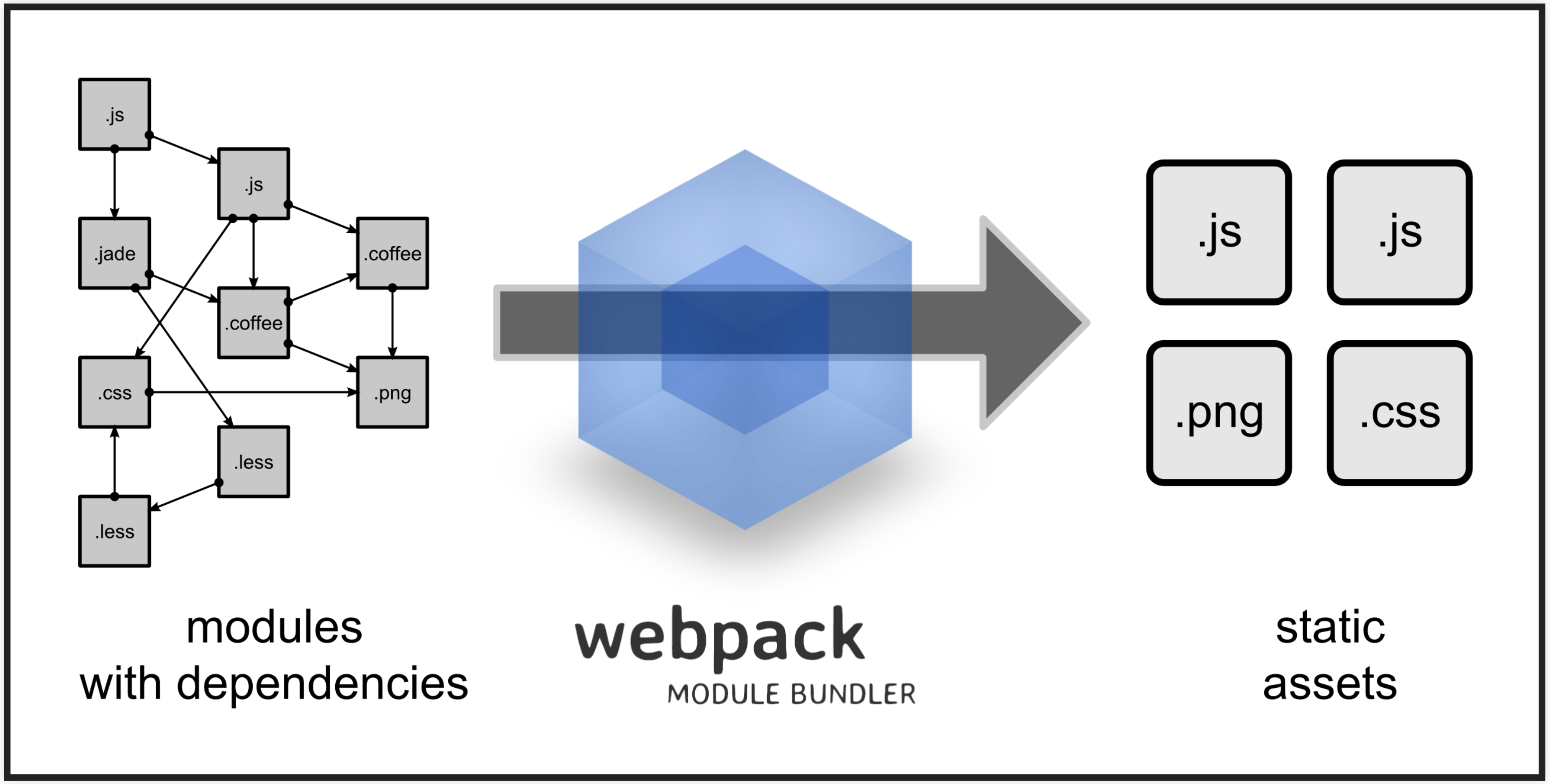
ORDERS.HTML

```html
<!doctype html>
<html>
  <head>
    <link rel="stylesheet" href="/style/dashboard.css"/>
    <link rel="stylesheet" href="/style/product.css"/>
    <link rel="stylesheet" href="/style/orders.css"/>
  </head>
  <body>
    <!-- ... -->
    <script type="text/javascript" src="/js/main.js"></script>
    <script type="text/javascript" src="/js/orders.js"></script
    <script type="text/javascript" src="/js/product.js"></scrip
  </body>
</html>
```

Assets needs to be organized

And solution is...

WHAT IS WEBPACK



Webpack is a module bundler

It takes modules with dependencies and generates static assets representing those modules

Not only for client side

CONCEPT

Entry point

Output

Loaders

Plugins

CODE SPLITTING

WEBPACK CONFIGURATION

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

module.exports = {
  entry: {
    products: './src/products.js',
    orders: './src/orders.js'
  },
  output: {
    filename: '[name].js',
    path: path.resolve(__dirname, 'dist')
  },
  // ....
};
```

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

SOME LOADERS

raw-loader

```
const htmlRule = {
  test: /\.html$/i,
  use: ['raw-loader']
};
```

file-loader

```
const imageRule = {
  test: /\.(png|jpg|jpeg|gif)$/i,
  use: [{
    loader: 'file-loader',
    options: {
      name: '[name]-[hash].[ext]',
    },
  }],
};
```

babel-loader

```
const jsRule = {
  test: /\.js$/i,
  use: [{
    loader: 'babel-loader',
    exclude: /node_modules/,
    options: {
      presets: [
        ['env', { targets: {
          browsers: ['last 2 versions', 'ie >= 8']
        } }],
        'stage-3'
      ]
    },
  }],
};
```

css-loader

```
const cssRule = {
  test: /\.css$/i,
  use: ['css-loader']
};
```

style-loader

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

SOME PLUGINS

DefinePlugin

```
const webpack = require('webpack');
module.exports = {
  plugins: [
    new webpack.DefinePlugin({
      SUM: '1+1',
      ENVIRONMENT: '"production"'
    }),
  ],
};

// main.js
console.log(SUM); // 2
console.log(ENVIRONMENT); // production
```

UglifyjsWebpackPlugin

```
const UglifyJsPlugin = require('uglifyjs-webpack-plugin');

module.exports = {
  plugins: [
    new UglifyJsPlugin()
  ],
};
```

ExtractTextWebpackPlugin

```
const ExtractTextPlugin =
  require("extract-text-webpack-plugin");

module.exports = {
  module: {
    rules: [{
      test: /\.css$/,
      use: ExtractTextPlugin.extract({
        use: 'css-loader'
      })
    }]
  },
  plugins: [new ExtractTextPlugin('[name].css')]
};
```

ProvidePlugin

```
const webpack = require('webpack');
module.exports = {
  plugins: [
    new webpack.ProvidePlugin({
      $: 'jquery',
      jQuery: 'jquery'
    }),
  ],
};

// main.js
$('#item');
jQuery('#item');
```

HtmlWebpackPlugin

```
const HtmlWebpackPlugin = require('html-webpack-plugin');

module.exports = {
  plugins: [new HtmlWebpackPlugin({
    template: require('html-webpack-template'),
  })]
};
```

DEV TOOLS

WEBPACK-DEV-SERVER

Provides live simple web server

```
npm i -D webpack-dev-server
```

```
module.exports = {
  devServer: {
    contentBase: path.join(__dirname, 'public')
  }
};
```

```
/* package.json */
{
  "scripts": {
    "start": "webpack-dev-server --open"
  }
}
```

HOT MODULE REPLACEMENT

Exchanges, adds, or removes modules while an application is running, without a full reload

HMR CONFIG

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

module.exports = {
  devServer: {
    hot: true,
  },
  plugins: [
    new webpack.HotModuleReplacementPlugin()
  ]
};
```

REFERENCES

- https://webpack.js.org/

QUESTIONS?

THANK YOU