**Webpack**

webpack é um empacotador de módulo JavaScript de código aberto. Ele é feito principalmente para JavaScript, mas pode transformar ativos de front-end, como HTML, CSS e imagens se os carregadores correspondentes forem incluídos. Webpack pega módulos com dependências e gera ativos estáticos que representam esses módulos.

Dentro da pasta principal do projeto, você precisa instalar o pacote “webpack” e “webpack-cli” com o npm, e o arquivo padrão do webpack é o arquivo “webpack.config,js”, dentro do arquivo “package.json”, você configura o script “start” como: “start”: “webpack”, para que a hora que você rodar o comando “npm start” no seu terminal/cmd dentro da pasta do projeto, ele execute o arquivo de configuração do webpack, e dentro do arquivo de configuração do webpack você deve exportar as configurações dentro de um objeto especificando o modo(development(desenvolvimento) ou production(produção)) e a entrada(o arquivo que será executada ao rodar o webpack), exemplo:

const webpack = require('webpack');

module.exports = {

mode: 'development',

entry: './src/principal.js'

}

A saída padrão do webpack é o arquivo “main.js” dentro da pasta “build”, caso queira mudar a saída do webpack, basta colocar nas configurações dele:

const webpack = require('webpack');

module.exports = {

mode: 'development',

entry: './src/principal.js',

output: {

filename: 'principal.js',

path: \_\_dirname + '/public'

}

}

Onde “filename” vai ser o nome do arquivo retornado, e “path” o caminho da pasta onde o arquivo será retornado. Para carregar arquivos “CSS” pelo webpack, primeiramente você precisa instalar os pacotes “style-loader” e “css-loader” com o npm, e após ter instalado os pacotes, basta colocar os modulos de CSS no arquivo de configuração do webpack:

const webpack = require('webpack');

module.exports = {

mode: 'development',

entry: './src/principal.js',

output: {

filename: 'principal.js',

path: \_\_dirname + '/public'

},

module: {

rules: [{

test: /\.css$/,

use: [

'style-loader', *// Adiciona CSS a DOM injetando a tag <style>*

'css-loader' *// Interpretar @import, url()...*

]

}]

}

}

Onde “style-loader” vai adicionar o CSS ao DOM da página, para quando ser criado um arquivo “.html” e linkar o arquivo “.js” criado pelo webpack(que no exemplo é o “principal.js”), todas as propriedades CSS já vão estar no arquivo, e serão adicionados na página, e o “css-loader” serve para interpretar @import, url e outras características do CSS. Caso queira criar um arquivo “.css” externo, basta instalar o pacote “mini-css-extract-plugin” com o npm, e dentro das configurações do webpack colocar:

const webpack = require('webpack');

const MiniCssExtractPlugin = require('mini-css-extract-plugin');

module.exports = {

mode: 'development',

entry: './src/principal.js',

output: {

filename: 'principal.js',

path: \_\_dirname + '/public'

},

plugins: [

new MiniCssExtractPlugin({

filename: 'estilo.css'

})

],

module: {

rules: [{

test: /\.css$/,

use: [

MiniCssExtractPlugin.loader,

'css-loader' *// Interpretar @import, url()...*

]

}]

}

}

Após ter ter configurado o plugin do “mini-css-extract-plugin” colocando o nome do arquivo que será gerado junto da extensão(.css), na parte de modules tem que remover o “style-loader” ou comentar ele para não ter conflito, e após isso basta rodar o “npm start” e linkar o arquivo “.css” no HTML e estará sempre sendo atualizado quando o script do webpack for executado pelo “npm start”. Para que você possa pegar um arquivo “SASS”,”SCSS” ou “CSS” e gerar um arquivo “.css”, basta instalar os pacotes “node-sass” e “sass-loader”, e depois colocar nas configurações do weback:

const webpack = require('webpack');

const MiniCssExtractPlugin = require('mini-css-extract-plugin');

module.exports = {

mode: 'development',

entry: './src/principal.js',

output: {

filename: 'principal.js',

path: \_\_dirname + '/public'

},

plugins: [

new MiniCssExtractPlugin({

filename: 'estilo.css'

})

],

module: {

rules: [{

test: /\.s?[ac]ss$/,

use: [

MiniCssExtractPlugin.loader,

'css-loader'*, // Interpretar @import, url()...*

’sass-loader'

]

}]

}

E agora ele vai procurar por todos os arquivos que terminem com “SS” como “CSS”, “SASS”, “SCSS”, etc.