### Создаем репозиторий на GitHub с двумя файлами

* 1. **.gitignore**
  2. **MIT License**

### Клонируем репозиторий проекта на локальный компьютер, командой:

* 1. **git clone *url\_repo***

в консоли Git Bash, открытой в отведенной папке или диске на вашем компьютере

### Заходим внутрь клонированного репозитория, командой:

* 1. **cd *repo\_name***

в консоли Git Bash

### Открываем папку проекта в редакторе кода VSCode, командой:

* 1. **code .**

### Открываем TERMINAL в редакторе кода VSCode (внизу окна редактора)

### Инициализируем (создаем) файл package.json в нашем проекте, командой:

* 1. **npm init -y**

в TERMINAL - слева в структуре проекта должен появится файл **package.json**

### Устанавливаем пакеты Webpack через TERMINAL, командой:

* 1. npm install --save-dev webpack webpack-cli

*должна произойти загрузка пакетов* ***webpack***  *в появившуюся в структуре проекта папку* ***node\_modules*** *и создается запись о подключении пакетов в файле* ***package.json*** *раздел* ***"devDependencies"***

* 1. npm install --save-dev webpack-dev-server
  2. npm install --save-dev babel-loader @babel/core @babel/preset-env @babel/plugin-proposal-class-properties
  3. npm install --save-dev html-loader
  4. npm install --save-dev style-loader css-loader postcss-loader postcss autoprefixer
  5. npm install -D file-loader url-loader
  6. npm install --save-dev html-webpack-plugin
  7. npm install --save-dev mini-css-extract-plugin
  8. npm install --save-dev clean-webpack-plugin
  9. npm install --save-dev webpack-merge
  10. npm install --save-dev friendly-errors-webpack-plugin
  11. npm install -D webpackbar
  12. npm install -D optimize-css-assets-webpack-plugin
  13. npm install -D handlebars handlebars-loader
  14. npm install --save-dev gh-pages

### Указываем скрипты в файле **package.json** раздел **"scripts":**

**"scripts": {**

**"start": "webpack-dev-server --env.mode development",**

**"build": "webpack --env.mode production",**

**"predeploy": "npm run build",**

**"deploy": "gh-pages -d dist"**

**}**

### Создаем структуру проекта

* 1. папка src
  2. внутри папки src файл index.js

### Создаем файл конфигураций Webpack в корне проекта

* 1. **webpack.config.js**

### Устанавливаем

* 1. Файлы настроек
  2. **.babelrc** [**https://babeljs.io/docs/en/configuration**](https://babeljs.io/docs/en/configuration)

**{**

**"presets": ["@babel/preset-env"],**

**"plugins": ["@babel/plugin-proposal-class-properties"]**

**}**

* 1. **postcss.config.js** [**https://github.com/postcss/postcss-loader**](https://github.com/postcss/postcss-loader)

**module.exports = {**

**plugins: [require("autoprefixer")],**

**};**

* 1. **.prettierrc** [**https://prettier.io/docs/en/options.html**](https://prettier.io/docs/en/options.html)

{

"printWidth": 80,

"tabWidth": 2,

"useTabs": false,

"semi": true,

"stringleQuote": true,

"trailingComma": "all",

"bracketSpacing": true,

"jsxBracketSameLine": false,

"proseWrap": "always"

}

## webpack.config.js (в корне проекта)

**const path = require("path");**

**const { CleanWebpackPlugin } = require("clean-webpack-plugin");**

**const { merge } = require("webpack-merge");**

**const FriendlyErrorsWebpackPlugin = require("friendly-errors-webpack-plugin");**

**const WebpackBar = require("webpackbar");**

**// console.log("DIRNAME", \_\_dirname); // глобальная переменная, содержащая абсолютный путь к файлу**

**const loadModeConfig = (env) =>**

**require(`./build-utils/${env.mode}.config`)(env);**

**// экспорт объекта настроек**

**module.exports = (env) =>**

**merge(**

**{**

**mode: env.mode,**

**context: path.resolve(\_\_dirname, "src"),**

**// 1. точка входа - откуда строить дерево зависимостей**

**entry: "./index.js",**

**// 2. куда положить результирующий бандл**

**output: {**

**path: path.resolve(\_\_dirname, "dist"),**

**filename: "[name].bundle.js",**

**},**

**// 3. загрузчики (loaders)**

**module: {**

**rules: [**

**{**

**test: /\.js$/, // регулярное выражение**

**exclude: /node\_modules/, // через указ папку свойства не прогонять**

**use: ["babel-loader"],**

**},**

**{**

**test: /\.(gif|png|jpe?g|svg)$/,**

**use: [**

**{**

**loader: "url-loader",**

**options: {**

**name: "[path]/[name].[ext]",**

**limit: 5000,**

**},**

**},**

**],**

**},**

**{**

**test: /\.html$/,**

**use: ["html-loader"],**

**},**

**{**

**test: /\.hbs$/,**

**use: ["handlebars-loader"],**

**},**

**],**

**// плагины применяются к результирующему бандлу**

**},**

**plugins: [**

**new CleanWebpackPlugin(),**

**new FriendlyErrorsWebpackPlugin(),**

**new WebpackBar(),**

**],**

**},**

**loadModeConfig(env)**

**);**

## development.config.js (в папке build-utils)

**const path = require("path");**

**const HtmlWebpackPlugin = require("html-webpack-plugin");**

**module.exports = (env) => ({**

**devtool: "cheap-eval-source-map",**

**module: {**

**rules: [**

**{**

**test: /\.css$/,**

**use: ["style-loader", "css-loader", "postcss-loader"],**

**},**

**],**

**},**

**plugins: [new HtmlWebpackPlugin({ template: "./index.html" })],**

**devServer: {**

**contentBase: path.join(\_\_dirname, "dist"),**

**historyApiFallback: true,**

**compress: true,**

**port: 4141,**

**noInfo: true,**

**quiet: true,**

**clientLogLevel: "warning",**

**stats: "errors-only",**

**open: true,**

**},**

**});**

## production.config.js (в папке build-utils)

**const HtmlWebpackPlugin = require("html-webpack-plugin");**

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

**const OptimizeCssAssetsPlugin = require("optimize-css-assets-webpack-plugin");**

**module.exports = (env) => ({**

**devtool: "source-map",**

**module: {**

**rules: [**

**{**

**test: /\.css$/,**

**use: [**

**MiniCssExtractPlugin.loader, // вытянет из js**

**"css-loader", // добавит все в js**

**"postcss-loader", // добавляет автопрефиксы**

**],**

**},**

**],**

**},**

**plugins: [**

**new HtmlWebpackPlugin({**

**template: "./index.html",**

**minify: {**

**collapseWhitespace: true,**

**removeComments: true,**

**removeRedundantAttributes: true,**

**removeScriptTypeAttributes: true,**

**removeStyleLinkTypeAttributes: true,**

**useShortDoctype: true,**

**},**

**}),**

**new MiniCssExtractPlugin({filename: "styles.css"}),**

**new OptimizeCssAssetsPlugin(),**

**],**

**});**