1. **安装NPM 和node.js并且配置好环境**：

安装（<https://www.cnblogs.com/zhuzhubaoya/p/7874486.html>）

配置环境（<https://blog.csdn.net/zb_ctrl/article/details/80552704>）

2、**在根目录中创建新的文件夹并转到文件夹目录**

mkdir myProject

Cd myProject

1. **新建src文件夹（这是我们源码放置的目录）并在该目录下创建app.vue文件（用于编写关于项目的业务代码）**
2. **在src目录下新建一个index.js作为入口文件**

import Vue from 'vue'

import App from './app.vue'

const root = document.createElement('div')

document.body.appendChild(root)

new Vue({

render: (h) => h(App)

}).$mount(root)

1. **在项目根目录下新建一个webpack.config.js文件**

const path = require('path')

module.exports = {

entry: path.join(\_\_dirname, 'src/index.js'),

output: {

filename: 'bundle.js',

path: path.join(\_\_dirname, 'dist')

}

}

1. **然后通过webpack把所有的文件打包成一个bundle.js文件，并且是能在浏览器里面直接运行的代码。现在我们可以在package.json 文件里的scripts对象里面添加一个脚本：**

"build": "webpack --config webpack.config.js"

1. **运行打包命令会生成dist文件夹和bundle.js文件：**

**Npm run build**

1. **在项目的根目录下新建一个文件夹叫build，把webpack的文件单独放到这个文件夹里面。因为我们项目中会用到很多不同的相关文件的配置，接下来先新建一个 webpack.config.base.js 文件，我们把webpack里面需要用到的共同的配置放到这个base的文件里面。比如开发环境和正式环境，以及后期我们要提到的服务端渲染的环境。我们都依赖于base这个配置。**

const path = require('path')

const createVueLoaderOptions = require('./vue-loader.config')

const isDev = process.env.NODE\_ENV === 'development'

const config = {

target: 'web',

entry: path.join(\_\_dirname, '../client/index.js'),

output: {

filename: 'bundle.[hash:8].js',

path: path.join(\_\_dirname, '../dist')

},

module: {

rules: [

{

test: /\.(vue|js|jsx)$/,

loader: 'eslint-loader',

exclude: /node\_modules/,

enforce: 'pre'

},

{

test: /\.vue$/,

loader: 'vue-loader',

options: createVueLoaderOptions(isDev)

},

{

test: /\.jsx$/,

loader: 'babel-loader'

},

{

test: /\.js$/,

loader: 'babel-loader',

exclude: /node\_modules/

},

{

test: /\.(gif|jpg|jpeg|png|svg)$/,

use: [

{

loader: 'url-loader',

options: {

limit: 1024,

name: 'resources/[path][name].[hash:8].[ext]'

}

}

]

}

]

}

}

module.exports = config

1. **新建一个 webpack.config.client.js ，这个client文件依赖于base文件，在此基础上扩展一些其他配置。**

const path = require('path')

const HTMLPlugin = require('html-webpack-plugin')

const webpack = require('webpack')

const merge = require('webpack-merge')

const ExtractPlugin = require('extract-text-webpack-plugin')

const baseConfig = require('./webpack.config.base')

const isDev = process.env.NODE\_ENV === 'development'

const defaultPlugins = [

new webpack.DefinePlugin({

'process.env': {

NODE\_ENV: isDev ? '"development"' : '"production"'

}

}),

new HTMLPlugin()

]

const devServer = {

port: 8000,

host: '0.0.0.0',

overlay: {

errors: true

},

hot: true

}

let config

if (isDev) {

// 开发环境的配置

config = merge(baseConfig, {

devtool: '#cheap-module-eval-source-map',

module: {

rules: [

{

test: /\.styl/,

use: [

'vue-style-loader',

'css-loader',

// {

// loader: 'css-loader',

// options: {

// module: true,

// localIdentName: isDev ? '[path]-[name]-[hash:base64:5]' : '[hash:base64:5]'

// }

// },

{

loader: 'postcss-loader',

options: {

sourceMap: true

}

},

'stylus-loader'

]

}

]

},

devServer,

plugins: defaultPlugins.concat([

new webpack.HotModuleReplacementPlugin(),

new webpack.NoEmitOnErrorsPlugin()

])

})

} else {

// 正式环境的配置

config = merge(baseConfig, {

entry: {

app: path.join(\_\_dirname, '../client/index.js'),

vendor: ['vue']

},

output: {

filename: '[name].[chunkhash:8].js'

},

module: {

rules: [

{

test: /\.styl/,

use: ExtractPlugin.extract({

fallback: 'vue-style-loader',

use: [

'css-loader',

{

loader: 'postcss-loader',

options: {

sourceMap: true

}

},

'stylus-loader'

]

})

}

]

},

plugins: defaultPlugins.concat([

new ExtractPlugin('styles.[contentHash:8].css'),

new webpack.optimize.CommonsChunkPlugin({

name: 'vendor'

}),

new webpack.optimize.CommonsChunkPlugin({

name: 'runtime'

})

])

})

}

module.exports = config

1. **命令行安装下 npm i webpack-merge -D**

参考文档：（<https://segmentfault.com/a/1190000014324125>）

部分webpack配置参考（<https://www.jianshu.com/p/cb0db034cb17>）

使用webpage工具，在编写VUE代码的时候经常报空格错误（<https://blog.csdn.net/qq_37674347/article/details/81049547>）