Webpack

2 相比其他构建工具，webpack的特点在于webpack能把任何类型的资源模块转换为.js代码块实现与其他模块的合并和加载。

3 webpack通过[loader](https://webpack.github.io/docs/loaders.html" \l "introduction" \t "https://segmentfault.com/a/_blank)来实现上述的转换。通过loader来实现转换其他资源到JavaScript代码。

4 webpack plugins插件，plugin的存在可以看做是实现那些loader实现不了或者不适合在loader中实现的功能。

5 webpack.config.js 配置



6 其他，webpack-dev-server使用

[WebPack + Vue打包流程理解](https://segmentfault.com/a/1190000010018042)

1.基于webpack构建vue项目，在不使用laoder或者plugins的情况下，使用webpack命令可以打包js

2.js引用了vue文件，使用了es6语法，webpack需要通过相应的laoder的帮助。

3.js引用关系完成之后，需要自动插入到相应的html中，使用对应的插件HtmlWebpackPlugin

4.避免文件变得臃肿，提取公共模块CommonsChunkPlugin

5.把css独立一个文件ExtractTextPlugin

6.热更新，node的express开启服务，通过中间件webpack-hot-middleware／webpack-dev-middleware

#### **使用webpack构建本地服务器**

想不想让你的浏览器监测你都代码的修改，并自动刷新修改后的结果，其实Webpack提供一个可选的本地开发服务器，这个本地服务器基于node.js构建，可以实现你想要的这些功能，不过它是一个单独的组件，在webpack中进行配置之前需要单独安装它作为项目的依赖

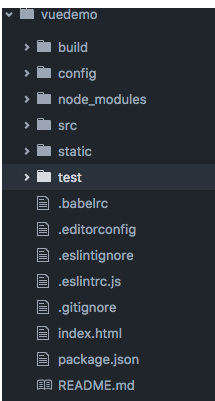
npm install --save-dev webpack-dev-server

#### **HtmlWebpackPlugin**

这个插件的作用是依据一个简单的模板，帮你生成最终的Html5文件，这个文件中自动引用了你打包后的JS文件。每次编译都在文件名中插入一个不同的哈希值。

ES6的模块加载功能：export和import，在组件中使用export，可以把组件定义为一个模块，import则可以把已定义的组件，这样就能方便处理模块间的依赖关系。

目录



| **目录/文件** | **说明** |
| --- | --- |
| build | 这个是我们最终发布的时候会把代码发布在这里，在开发阶段，我们基本不用管。 |
| config | 配置目录，默认配置没有问题，所以我们也不用管 |
| node\_modules | 这个目录是存放我们项目开发依赖的一些模块，这里面有很多很多内容，不过高兴的是，我们也不用管 |
| src | 我们的开发目录，基本上绝大多数工作都是在这里开展的 |
| static | 资源目录，我们可以把一些图片啊，字体啊，放在这里。 |
| test | 初始测试目录，没用，删除即可 |
| .xxxx文件 | 这些是一些配置文件，包括语法配置，git配置等。基本不用管，放着就是了 |
| index.html | 首页入口文件，基本不用管，如果是开发移动端项目，可以在head区域加上你合适的meta头 |
| package.json | 项目配置文件。前期基本不用管，但是你可以找一下相关的资料，学习一下里面的各项配置。至少，要知道分别是干嘛的。初期就不管了。 |
| README.md | 不用管 |

App.vue是项目入口文件

main.js这是项目的核心文件。全局的配置都在这个文件里面配置