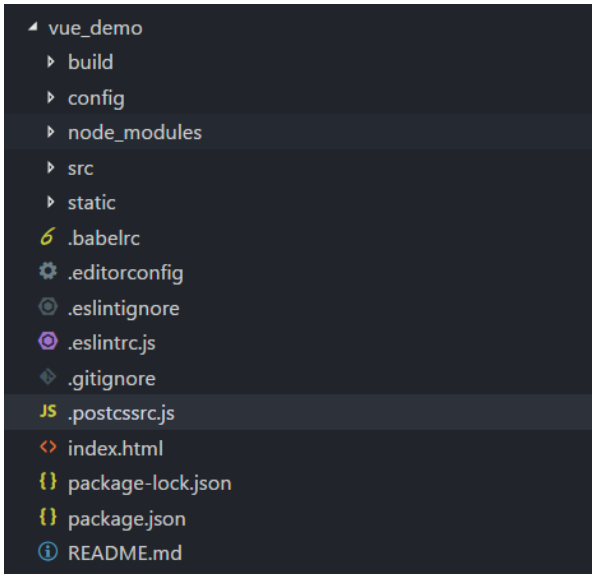
Vue 项目构建到打包体系

整体结构：



配置文件夹 build/config

config 中

node_modules 依赖

src 源码目录

static 静态目录，放全局资源

babelrc：ES6 转 ES5/JSX 转 JS，rc：runtime control（babel 的配置文件）

---

src 目录下的 main.js 文件是程序入口文件，配置选项在 webpack.base.conf.js（同样在其中配置了 index.html，不可随意改动）

最终整个项目在 index.html 中打包生成了一个

<script type="text/javascript" src="/app.js"></script>

Main.js 中的 template: '<App/>' （插件）会插入到 el 所匹配的页面上，此时渲染的根组件（根标签）就是 App。（PS:组件就是一个局部的功能模块，可复用）

Src 中目录中构建：

1. 构建入口 main.js
2. 构建根组件 App.vue

App.vue 中的结构：

```
<template>
    <div></div>
</template>
<script>
    export default {
        向外默认暴露一个配置对象
    }
</script>

<style>
</style>
```

3．构建子组件（eg.HelloWorld 组件）

```
<template>
    <div></div>
</template>
<script>
    export default {
        配置对象必须写成函数
}
</script>
<style>
</style>
```

构建好子组件（HelloWorld），根组件来引用使用子组件（App.vue，三步：1.引入组件，2.映射组件标签，3.使用组件标签）,最后入口文件 main.js 引入使用根组件 App.vue，创建 Vue 实例，（import Vue from ‘vue’ 注意大小写），el 中的元素与 index.html 中的标签 id 对应，最重要将 App.vue 中的内容渲染到主界面中，先引入 App.vue，再在 components 中将其映射成标签，最后使用组件标签。

template: ‘<App/>’

Vue 项目的打包发布：

1．打包：在文件目录下：npm run build

会生成一个打包文件夹 dist

2. 发布：