webpack

- webpack
  - 2009年初，commonJS规范还未出来，此时前端开发人员编写的代码都是非模块化的，
    - 那个时候开发人员经常需要十分留意文件加载顺序带来的依赖问题
  - 与此同时nodejs开启了js全栈大门，而requirejs在国外也带动着前端逐步实现模块化
    - 同时国内seajs也进行了大力推广
    - AMD规范，具体实现是requirejs define('模块id',[模块依赖1，模块依赖2]，function（){return ;}),ajax请求文件并加载。
    - Commonjs||CMD规范seajs淘宝玉伯
      - commonjs和cmd非常相似的
        - cmd require/module.exports
      - commonjs是JS在后端语言的规范：模块、文件操作、操作系统底层
      - CMD仅仅是模块定义
    - UMD通用模块定义，一种既能兼容AMD也能兼容commonjs也能兼容浏览器环境运行的万能代码
  - npm/bower集中包管理的方式备受青睐，12年browserify/webpack诞生
    - npm是可以下载前后端代码的js代码475000个包
    - bower只能下载前端的js代码，bower在下载bootstrap的时候会自动的下载jquery
    - browserify解决让require可以运行在浏览器，分析require的关系，组装代码
    - webpack打包工具，占市场主流
      - (function(root,factory){
      - if(typeof exports==='object'){
      - module.exports=factory();//commonjs环境下能拿到返回值
      - }else if(typeof define==='function'){
      - define(factory);//define(function(){return 'a'}) AMD
      - }else{
      - window.eventUtil=factory();
      - }
      - })(this function(){

- //module
- return {
- //具体模块代码
- addEvent:function(el,type,handle){
- //……
- },
- removeEvent:function(el,type,handle){
- }
- }
- })

- webpack打包模块的源码
  - 1把所有模块的代码放到函数中，用一个数组保存起来
  - 2根据require时传入的数组索引，能知道需要哪一段代码
  - 3从数组中，根据索引取出包含我们代码的函数
  - 4执行该函数，传入一个对象module.exports
  - 5我们的代码，按照约定，正好是用module.exports='xxxx'进行赋值
  - 6调用函数结束后，module.exports存原来的空对象中，就有值了
  - 7最终return module.exports;作为require函数的返回值
- scripts:{ "dev":"webpack ./main.js ./build.js" //依据入口文件main.js生成build.js }
  - scripts:{ "dev":"webpack ./main.js .build.js --watch" }//时刻监听，修改文件后会生成新的build.js
  - 使用webpack.config.js
    - scripts:{"dev":"webpack --config ./webpack.dev.config.js", "prod":"webpack --config ./webpack.prod.config.js" } //两个文件分别是用于测试和服务器发布，用于发布不需要检测watch
- webpack.config.js文件配置
  - entry是一个对象，程序的入口
    - key:随意写
    - value:入口文件
  - output是一个对象，产出的资源
    - key:filename
    - value:生成的build.js
  - module模块（对象）
    - loaders:[]
      - 存一些loader {test:正则，loader:'style-loader!css-loader'}

- 处理css
  - npm i css-loader -S
  - import ('./xxx.css');
  - npm i url-loader file-loader -S
  - import img‘1.jpg’
    - 加载本地图片的方式 靠谱的在非HTML页面中，加载本地资源的方式
      - import img from './a.jpg'
      - export default{
        - template:`
        - `,
        - data(){
        - }
      - }
- 处理less
  - loader:'style-loader!css-loader!less-loader'
- 处理ES6
  -