第一章

1.1学习背景

前端框架演变

js库->mvc->mvvm

js前端库：yui，jquery，dojo，prototype，kissy，mootools（对js方法的一些补充）

mvc时代：backbone.js，underscore,js,jquery

mvvm时代：angular ，react， vue

为什么需要构建?

开发复杂化

框架去中心化（npm install）

语言编译化

模块化

为什么需要webpack？

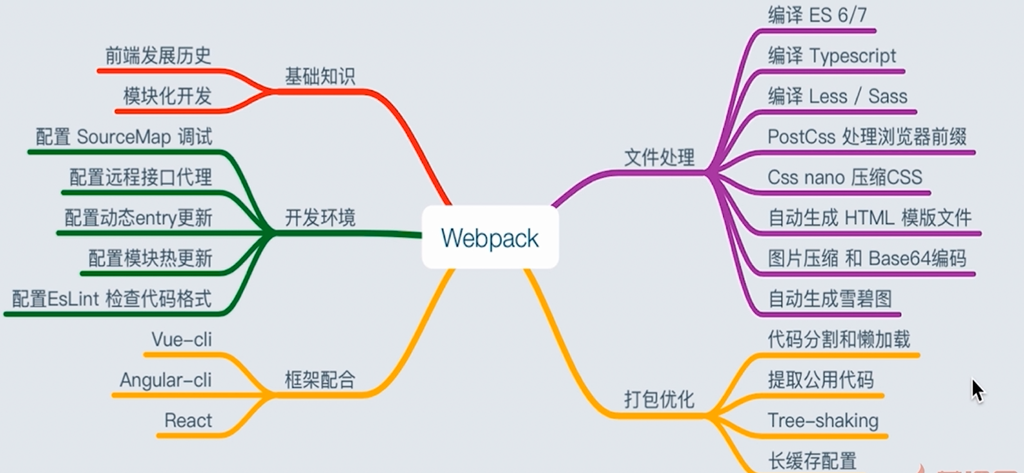
简单来说就是时代的需要 vue angular react都是基于webpack 为了不被时代抛弃

vue-cli、react-starter、angular-cli

代码分割（code-splitting）

天生的模块化

1.2 导学

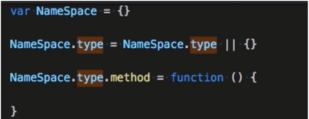


第二章

2.1

一.

js模块化

1 命名空间：库名。类名。方法名

2commomjs （服务端）

只有通过module.exports暴露模块接口

通过require引入模块

同步执行

3AMD、CMD、UMD

(1)AMD:Async Module Definition

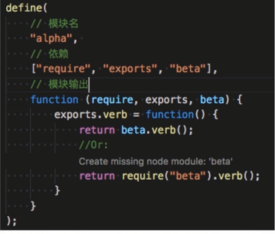
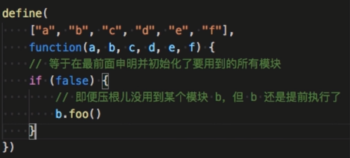
使用define定义模块

使用require加载模块

requireJs

依赖前置，提前执行

主要分为三部分：模块名，依赖 输出（模块名可以省略）

（2）CMD：Common Module Definition

一个文件为一个模块

使用define定义模块

使用require加载模块

SeaJS

尽可能懒执行

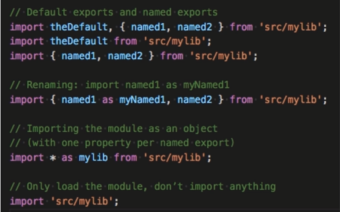


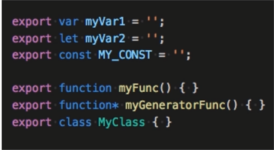
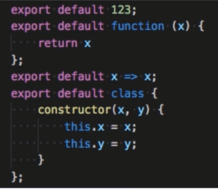
AMD和CMD之间的区别：执行方式不同，AMD引入就会全部执行，CMD引入以后不执行，只有当代码调用的时候才会执行

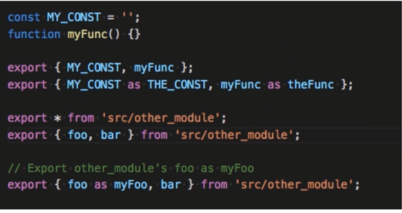
4.es6 module

暴露模块：export/export default

引入模块 ：import





二.webpack支持的

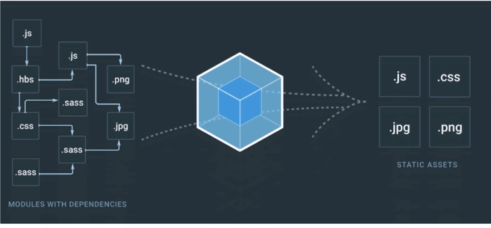
AMD（requireJS）

ES6Module（推荐）

commonJS

2.3.环境准备 node，npm

2.4webpack简介

1.概述：

2.5核心概念

（1）entry（打包人口）

代码的入口

打包的路口

单个或多个入口









（2）Output（出口）

打包成文件（bundle）

一个或多个

自定义规则（name，hash）



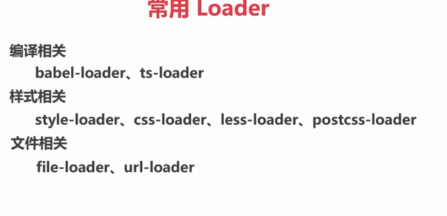


（3）loaders（处理除了js意外的其它类型的资源文件）

处理文件

转化为模块

常用的Loader





（4）Plugins（压缩代码，代码分割）

参与打包的整个过程

打包优化和压缩

配置编译时的变量

混淆压缩代码



优化相关

CommonsChunkPlugin（提取相同代码）

UglifyjsWebpackPlugin（混淆压缩代码）

功能相关

ExtracTextWebpackPlugin（css提取出来做一个单独文件）

HtmlWebpackPlugin（生成html）

HotModuleReplacementPlugin（模块热更新）

CopyWebpackPlugin（帮助拷贝文件）

第三章

3.1使用webpack

webpack命令

webpack –h （help 帮助）

webpack –v (版本)

webpack <entry>【entry】<output>

npm install webpack-cli –g

webpack-cli(知道就好)

交互式的初始化一个项目

迁移项目V1 -> V2

webpack配置

脚手架

vue-cli ，angular-cli ，react-sta

3.2打包js

两种方式



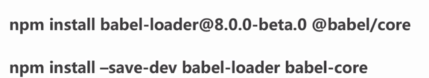
例子：webpack app.js bundle.js

参考文件夹

注意：AMD的规范是异步的 ，所以会出现两个bundle.js

3.3编译es6/7

babel-loader

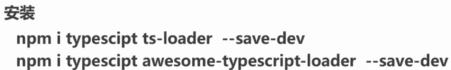


过程

3.4 解析typeScript

js的超集，来自微软

typeScripe-loader



会配置两个地方



tsconfig.json //ts的一些配置

过程 npm init



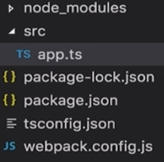
新建tsconfig.json



新建webpack.config.js



目录结构

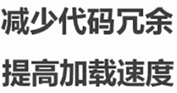


app.ts



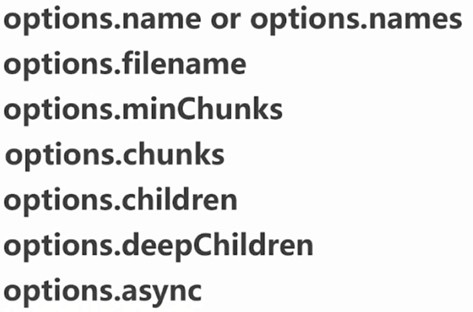
3-5提起公共代码

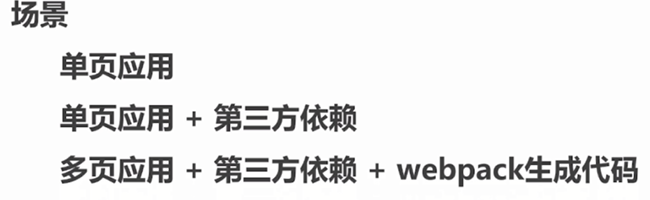
好处：





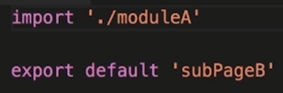




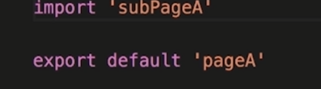




moduleA.js

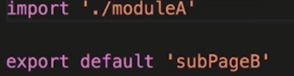


pageA.js

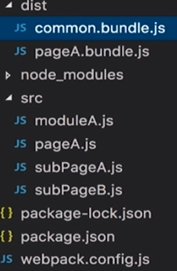


subpageA.js

subpageB.js



目录结构



webpack.connfig.js

