Webpack 学习笔记

是一个打包软件

同时还进行了翻译的工作（loader）

不仅强大而且灵活

webpack其实并不简单

课程重点

> 理解前端模块化
>
> 理解Webpack打包核心思路
>
> 理解webpack中的"关键人物"

误区规定

> 不要执着Api和命令

学习路径

> 作用域
>
> 命名空间
>
> 模块化

面向模块编程

作用域

文件多了容易发生命名冲突

加上命名空间

```
var Susan = {
  name: 'Susan',
  sex: '女孩',
  // 自我介绍方法
  tell: function() {
    console.log('我的名字是', this.name)
    console.log('我的性别是', this.sex)
  }
}
```

对象苏珊就是一个命名空间

闭包

> 函数内部的函数
>
> 外部可访问
>
> 始终保存在内存中(可能造成内存泄漏)

模块作用域

保护该保存的,

返回可访问的

```
(function(window) {
  var name = 'Susan'
  var sex = '女孩'
  function tell() {
    console.log('我的名字是', name)
    console.log('我的性别是', sex)
  }

  window.susanModule = {tell}
})(window)
```

模块化的有点

> 作用于封装

> 重用性

> 解除耦合

模块化的进化史

AMD(异步模块定义)

```
// 求和模块的定义
define('getSum', ['math'], function (math) {
  return function (a, b) {
    console.log('sum: ' + math.sum(a, b));
  }
});
```

COMMONJS

```
// 通过 require 函数来引入
const math = require('./math');

// 通过 exports 将其导出
exports.getSum = function (a, b) {
  return a + b;
}
```

ES6 MODULE

```
// import导入
import math from './math';

// export导出
export function sum(a, b) {
  return a + b;
}
```

webpack的打包机制

- 从入口文件开始，分析整个应用的依赖树
- 将每个依赖模块包装起来，放到一个数组中等待调用
- 实现模块加载的方法，并把它放到模块执行的环境中，确保模块间可以互相调用
- 把执行入口文件的逻辑放在一个函数表达式中，并立即执行这个函数

npm包管理器

可以直接把包安装下来

package.json

npm --save (将下载下来的包信心保存咋在packa.json中)

--dev(开发环境)

- ^version：中版本和小版本。
  ^1.0.1 -> 1.x.x
- ~version： 小版本
  ~1.0.1 -> 1.0.x
- version: 特定版本

npm install的过程

> 寻找包版本信息文件(package.json),依照她来进行安装
>
> 查package.json中的依赖,并检查项目中的其他版本信息文件
>
> 如果发现了新包,就更新版本信息文件

webpack的核心特性

学习目标

- 使用webpack构建简单的工程
- 了解webpack配置文件
- 掌握“一切皆模块与loaders”的思想
- 理解webpack中的“关键人物”

webpack.config.js可以指定入口文件,不一定非要是index.js

entry:''//工程资源的入口

output:{}//对打包的结果进行配置

devServe可配置项目访问端口来预览项目

loader

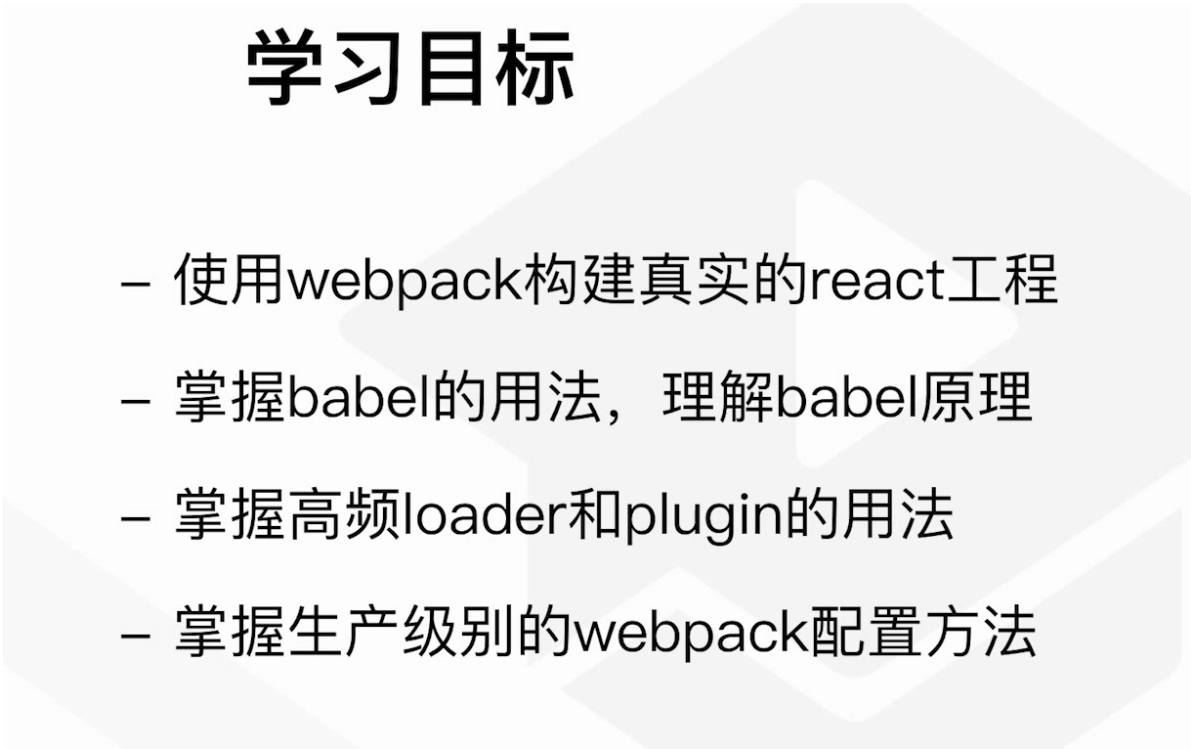
css loader的配置顺序和加载顺序相反

```
// 文件加载器-loader
module: {
  rules: [
    {
      test: /\.css$/,
      use: [
        'style-loader',
        'css-loader'
      ]
    }
  ]
}
}
```

plugins

uglifyjsplugins:丑化代码以进行压缩打包文件

webpack构建工程



babel

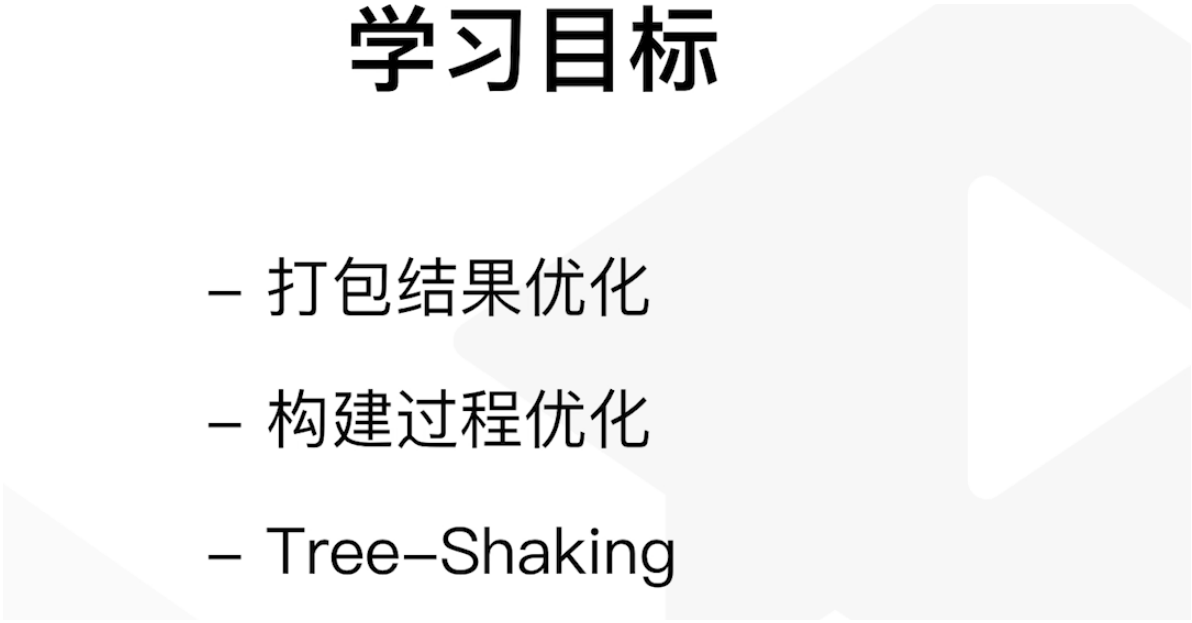
可以将高版本的代码编译成低版本的代码

比如将ES6代码编译成ES5代码: **npm install @babel/preset-env**

先找babelrc文件,没有就向外遍历

HMR(热刷新)

Webpack性能调优



体积优化

uglifyplugin

WerserPlugin是uglify的一个分支

bundleAnalyZerPlugin分析器可以分析打包中各部分的大小

webpack单线程

HappyPack可以分成多个子任务加速代码的构建.创建线程池

threead-loader

tree-shake

webpack是什么

前端发展的产物

模块化打包方案

工程化方案