Webpack优化方法

**提高开发环境下构建速度**

1.NoParse 忽略较大的库，要求是非模块化的库（不能含有require,define和import）

2.DllPlugin 把一些经常使用的库提取出来，变成一种引用的模式，避免打包时对不变的库进行构建。 动态数据库引用

在构建config里面使用DllReferencePlugin引用manifest描述文件。

主要是在开发环境使用。

**代码拆分**

把大的bundle文件拆分成若干个小的bundle文件，这样可以缩短首屏加载的时间。

方法1：手工定义入口，因为webpack会对入口的js文件进行递归式的依赖查询，所以可以手工拆分代码，抽离公共部分，避免重复代码

方法2：使用splitChunks插件进行公共代码提取。

在webpack配置文件里面,可以定义提取的公共代码

**动态加载，主要解决过大资源请求问题。**

Webpack原生 Import.then()

React 使用lazy引用组件， Suspense组件实现异步加载。

Vue使用promise来引用组件。在resolve函数回调里面传入需要异步加载的组件。

**Webpack资源压缩**

Terser压缩JS （生产模式自动启用）

Mini-css-extract-plugin 进行CSS的提取，提取到一个单独的文件里

Optimize-css-Assets-Plugin 进行CSS的压缩

HtmlWebpackPlugin 生产模式下会默认启用压缩功能。 类似于去掉空白，删除注释，删掉冗余属性。使用短描述方式html.

Webpack的资源持久化缓存

要保证html,css,js更新之后，用户本地缓存不会影响服务器文件的加载。

每个打包的资源文件名字里都包含有唯一的hash值。

使用content-hash进行hash值的计算。

Webpack进行性能检测和分析

Stats 分析与可视化图

Webpack –profile –json >stats.json

Webpack-bundle-analyzer进行体积分析

Speed-measure-webpack-plugin 进行速度温习