问题一：如何减少 Webpack 打包后的文件体积

按需加载

想必大家在开发 SPA 项目的时候，项目中都会存在十几甚至更多的路由页面。如果我们将这些页面全部打包进一个 JS 文件的话，虽然将多个请求合并了，但是同样也加载了很多并不需要的代码，耗费了更长的时间。那么为了首页能更快地呈现给用户，我们肯定是希望首页能加载的文件体积越小越好，**这时候我们就可以使用按需加载，将每个路由页面单独打包为一个文件。**当然不仅仅路由可以按需加载，对于`loadash`这种大型类库同样可以使用这个功能。

按需加载的代码实现这里就不详细展开了，因为鉴于用的框架不同，实现起来都是不一样的。当然了，虽然他们的用法可能不同，但是底层的机制都是一样的。都是当使用的时候再去下载对应文件，返回一个`Promise`，当`Promise`成功以后去执行回调。

Scope Hoisting

Scope Hoisting 会分析出模块之间的依赖关系，尽可能的把打包出来的模块合并到一个函数中去。

比如我们希望打包两个文件

```
// test.js
export const a = 1
// index.js
import { a } from './test.js'
```

对于这种情况，我们打包出来的代码会类似这样

```
[
  /* 0 */
  function (module, exports, require) {
    //...
  },
  /* 1 */
  function (module, exports, require) {
    //...
  }
]
```

但是如果我们使用 Scope Hoisting 的话，代码就会尽可能的合并到一个函数中去，也就变成了这样的类似代码

```
[
  /* 0 */
  function (module, exports, require) {
    //...
  }
]
```

这样的打包方式生成的代码明显比之前的少多了。如果在 Webpack4 中你希望开启这个功能，只需要启用`optimization.concatenateModules`就可以了。

```
module.exports = {
  optimization: {
    concatenateModules: true
  }
}
```

Tree Shaking

Tree Shaking 可以实现删除项目中未被引用的代码，比如

```
// test.js
export const a = 1
export const b = 2
// index.js
import { a } from './test.js'
```

对于以上情况，test 文件中的变量 b 如果没有在项目中使用到的话，就不会被打包到文件中。

如果你使用 Webpack 4 的话，开启生产环境就会自动启动这个优化功能。