https://blog.csdn.net/jlin991/article/details/61925347

https://blog.csdn.net/qq\_26708777/article/details/79210549

var webpack=require('webpack');

var path=require('path'),

node\_modules=path.resolve(\_\_dirname,'node\_modules');

//var pathToReact=path.resolve(node\_modules,'react/dist/react.min.js');

//public path

const ASSET\_PATH=process.env.ASSET\_PATH||'/public';

var pluginLoader=new webpack.LoaderOptionsPlugin({

minimize: true,

debug: false,

options: {

context: \_\_dirname

}

})

module.exports={

entry:{

index:'./src/entry.js'

}/\*,

resolve:{

alias:{

'react':pathToReact

}

}\*/,

output:{

path:path.join(\_\_dirname,"dist"),

filename:"[name].bundle.js",

chunkFilename:"[id].chunk.js"

},

devtool:'inline-source-map',

devServer:{

//hot:true,

contentBase:path.resolve(\_\_dirname,'dist'),

inline:true

},

module:{

loaders:[

{

test:/\.css$/,

loader:'style-loader!css-loader',

},

{

test:/\.scss$/,

use:[{loader:"style-loader"}, //create style node from js strings

{loader:"css-loader"}, //translate css into Commonjs

{loader:"sass-loader"} //compile Sass to css

]

},

{

test:/\.jsx?$/,

loader:'babel-loader', //loader babel ,

query:{

presets:['react','env']

}

},

{

test:/\.(png|jpg)$/,

use:'url-loader?limit=40000'

}

],

},

plugins:[commonsPlugin,

new webpack.DefinePlugin({

'process.env.ASSET\_PATH':JSON.stringify(ASSET\_PATH)

}),

new webpack.DefinePlugin({

'process.env.NODE\_ENV':JSON.stringify('production')

})

]

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73

**发现罪魁祸首是这个**

devtool:'inline-source-map',

设置了source-map之后的大小是 2.88MB   
而没有设置的时候是212KB   
所以说开发环境的时候可以开sourcemap   
但是生产环境就一定要把sourcemap注释掉！

**提取第三方库**

像 react 这个库的核心代码就有 627 KB，这样和我们的源代码放在一起打包，体积肯定会很大。所以可以在 webpack 中设置还有可以加入这个

var commonsPlugin=new webpack.optimize.CommonsChunkPlugin({

filename:"common.js",

name:"commons"

});

* 1
* 2
* 3
* 4

在index.html

<script src="common.js" defer async></script>

<script src="index.bundle.js" defer async></script>

* 1
* 2

**实际上使用webpack -p**

webpack -p

* 1

这是一个自动化的方法来构建生产环境下的项目。   
它会 自动使用UglifyJsPlugin进行最小化   
还会运行使用LoaderOptionsPlugin。   
等价于下面这条语句：

webpack --optimize-minimize --define process.env.NODE\_ENV="'production'"

以下是我的一些观点

以上是开启压缩的方式将文件体积减少

我现在是开发环境，能不能直接使用es6来调试或者转义为es5之后，有单个文件引用的方式

如果使用webpack这样来调试太累了

既然有srouceMap这个东西，那么如何启动srouceMap这样的工具呢

### 对于开发环境

以下选项非常适合开发环境：

1、每个模块都使用 eval() 执行，并且都有 //@ sourceURL。此选项会非常快地构建。主要缺点是，由于会映射到转换后的代码，而不是映射到原始代码（没有从 loader 中获取 source map），所以不能正确的显示行数。

eval-source-map - 每个模块使用 eval() 执行，并且 source map 转换为 DataUrl 后添加到 eval() 中。初始化 source map 时比较慢，但是会在重新构建时提供比较快的速度，并且生成实际的文件。行数能够正确映射，因为会映射到原始代码中。它会生成用于开发环境的最佳品质的 source map。

cheap-eval-source-map - 类似 eval-source-map，每个模块使用 eval() 执行。这是 "cheap(低开销)" 的 source map，因为它没有生成列映射(column mapping)，只是映射行数。它会忽略源自 loader 的 source map，并且仅显示转译后的代码，就像 eval devtool。

cheap-module-eval-source-map - 类似 cheap-eval-source-map，并且，在这种情况下，源自 loader 的 source map 会得到更好的处理结果。然而，loader source map 会被简化为每行一个映射(mapping)。



还有就是可不可以将import中的文件分类合并

这样可以看到哪些是需要分类合并哪些不需要

es6如何提供全局的js，还是说es6不需要import全局的js

对于大文件lint校验会死掉