React js

是

01介绍react：

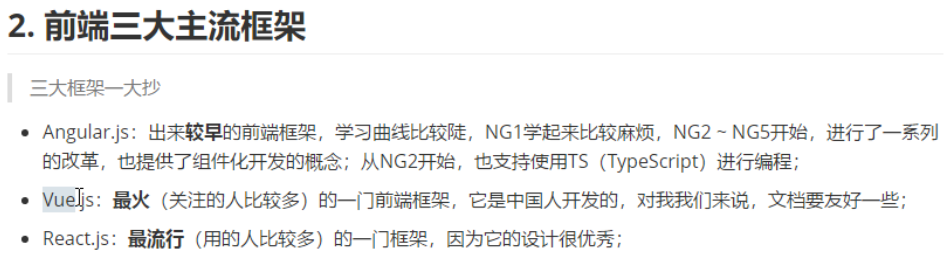




库的切换，比方从web端转到了app端，只需要修改少量的代码。

框架之间切换就不容易了。

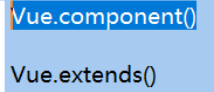
02.三大框架的现状：



03.从组件化方面对比React和Vue：



用于创建组件：



大部分时候用的是模板文件





React中，结构，行为，样式都是通过js来体现的。所以，js的功底一定要好。

Es6要很熟，es7部分。

04.从其它角度对比React和Vue：



05.为什么要学习React：



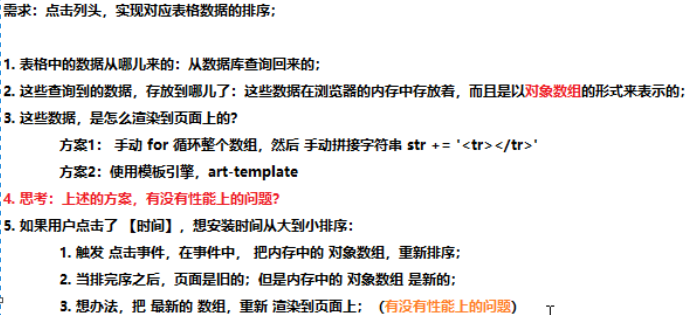
06.介绍DOM和虚拟DOM的概念：



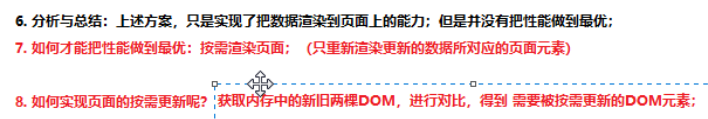
概念占50%，语法占50%

虚拟Dom中的api是由框架提供，而框架是程序员写的，其实也就是程序员提供的。

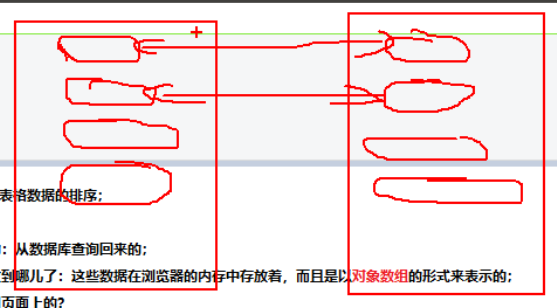
07.虚拟DOM的本质和目的

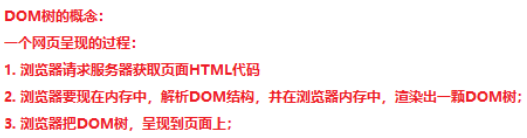


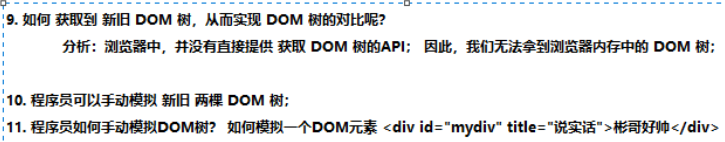
性能的问题，点击重新排序，需要排序的去排序，但位置不变的，就不需要重新渲染了。

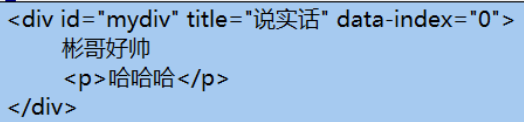


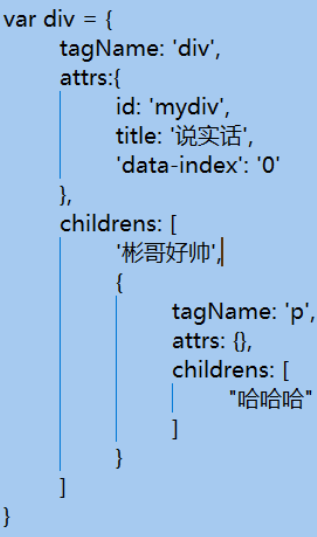
可以去对比新旧dom树的对应关系，如果需要更新，就标记为需要更新，不需要的，就不需要标记了。





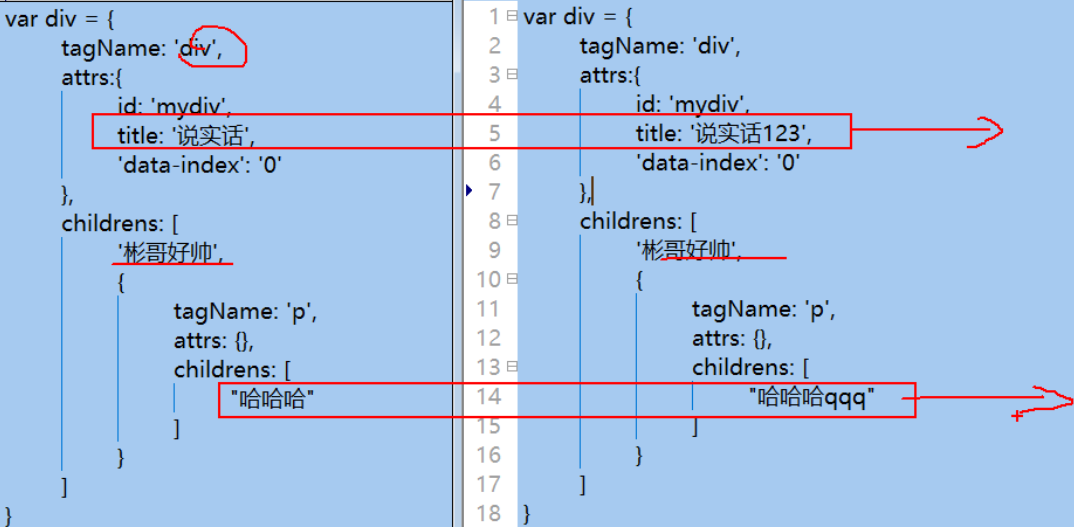




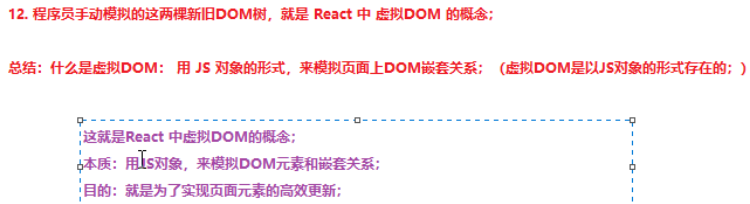


这就是用js对象模拟了dom树。

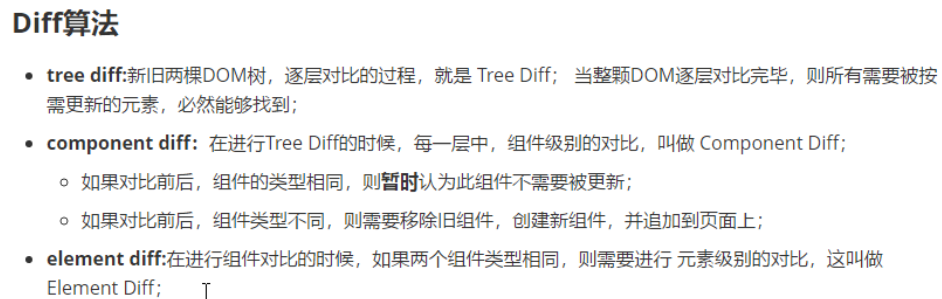
属性里面带-号得用单引号引起来，不然会解析成#。会出问题。

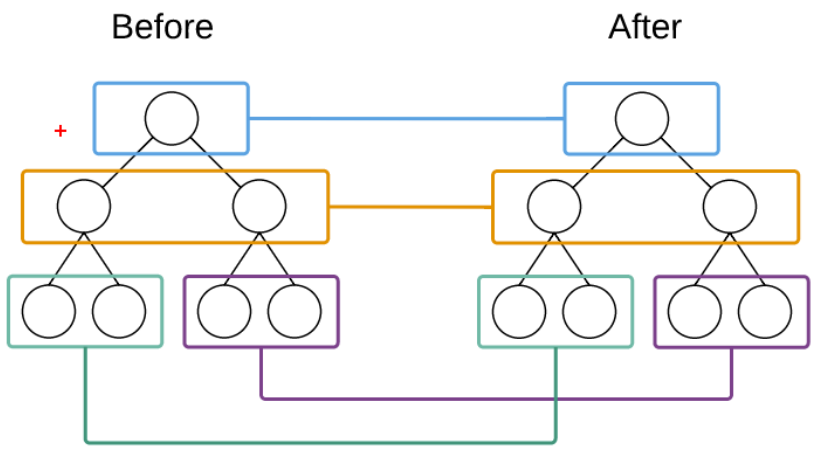


左边是旧的，右边是新的。按需更新，一一对比，改变了的，修改就是。

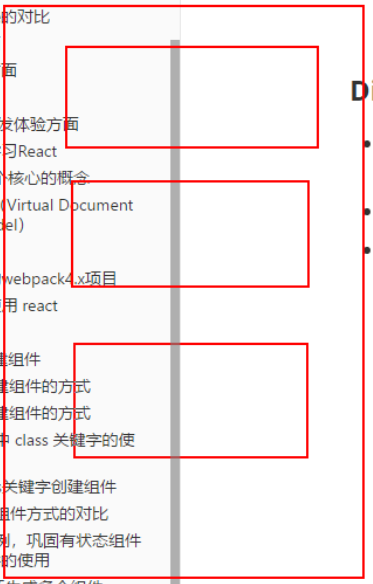


08.介绍Diff算法的概念：





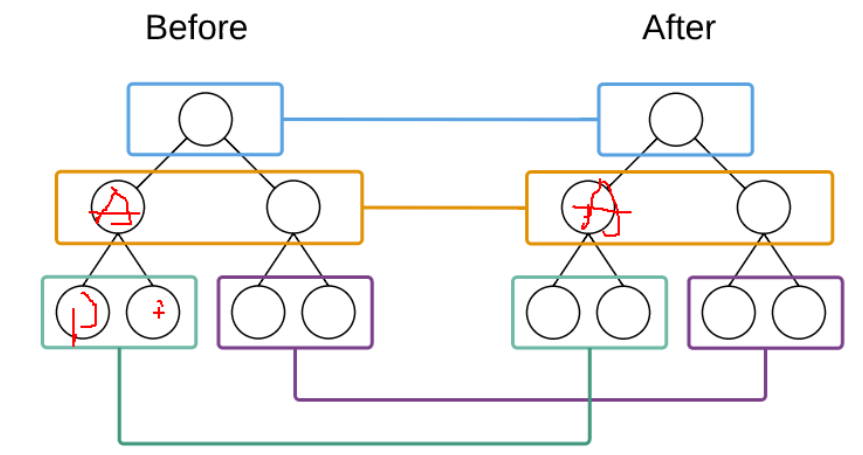
tree diff是逐层对比。每一个层都是一个组件component。



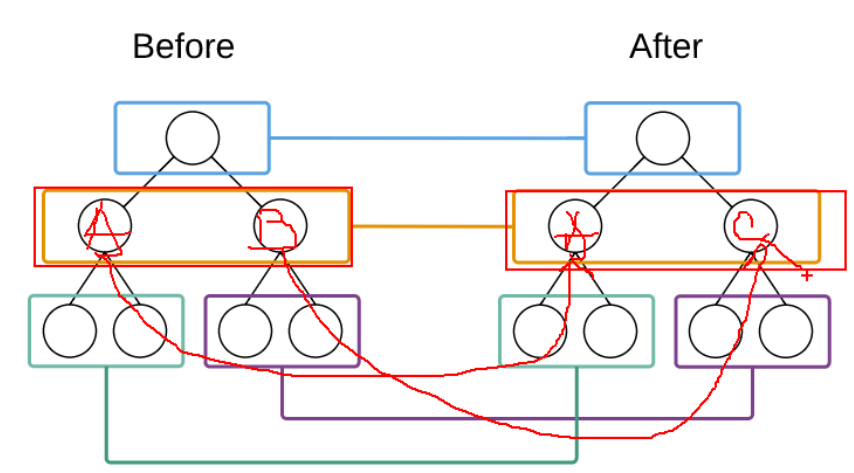
一个页面会有多个组件。比方从上自下有什么列表，什么什么列表，什么新闻，这个三个不同组件。这每个组件之间也要进行对比。

组件中，由一个又一个的ui元素组成。

下面图中新旧都是A组件，就暂且定为不修改。接下里还要看元素的具体对应。

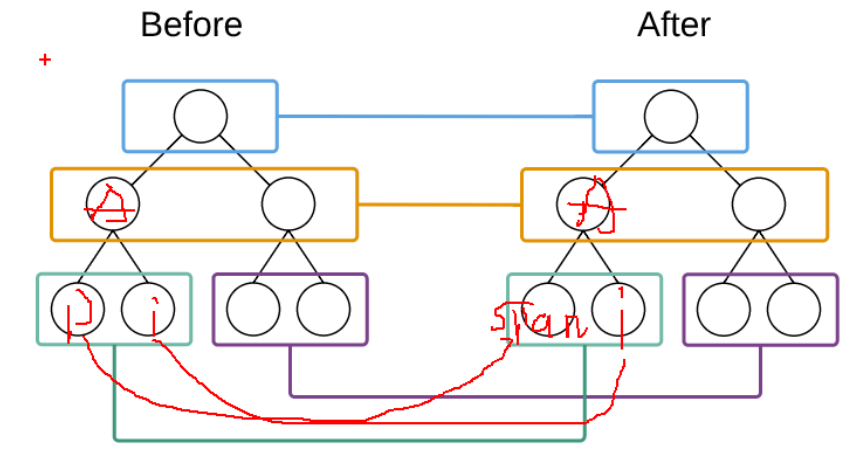


而这里，A组件暂时待定，C组件就是update，改成新组件了。



element diff。关于元素，比方一个层，也就是一个组件中，有两个元素，之前是p标签和i标签，现在变成了span标签和i标签。

这组建中，每一个一一对应的元素，我们也要进行对比。



总结下就是：虚拟Dom提供了这两棵新旧的Dom树。Diff算法提供了这种新旧Dom数对比的最优方案。

09.webpack4.x最基本的使用步骤：

新建文件夹：



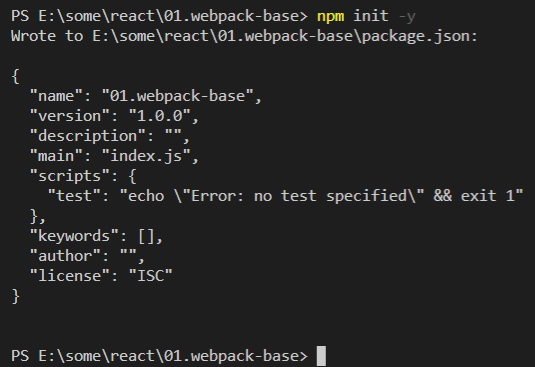
再用visual studio code打开

然后切换终端





输入npm init -y快速初始化一个项目。



输入之后发生的变化：



新建src目录，所有的源代码都往里面放。

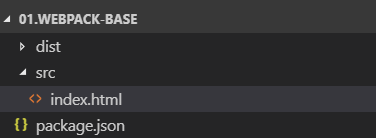
再建一个dist文件夹。

项目做好，最后要发布到dist当中去。（老师表示之前也有用bin的，这里老师是按标准的项目结构进行创建的，因为用jquery去引用的时候，都要用/dist/jquery.min.js。表示项目做成产品，要发布，最终都要发布到dist当中去。一些开源项目都是这么干的。）

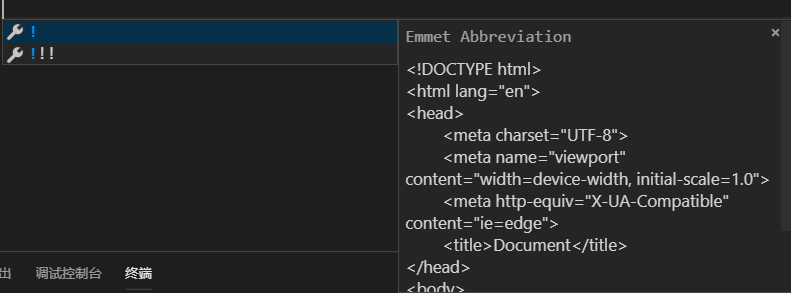
公司项目也是dist。但没看到src



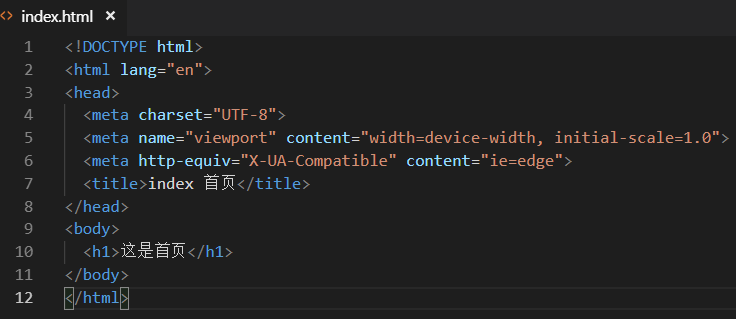
src中新建文件——index.html



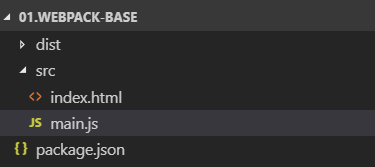
!会自动跳出整个页面代码，很好用。



稍微加点代码：



再新建一个main.js文件。



webpack打包的入口文件。

要想实现打包的话，从main.js开始打包，打包之后自动引入到index.html就可以了。

输入cls：



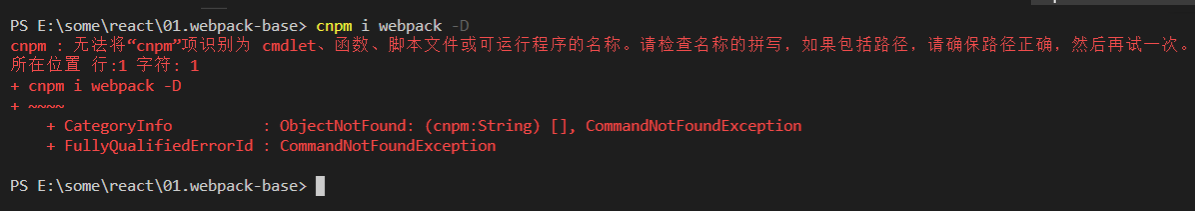
老师比较喜欢用cnpm，同学似乎都只用过npm

c代码china，中国的npm

因为用npm，默认是从国外的npm网站下载包的，回慢。



i应该就是install的意思。

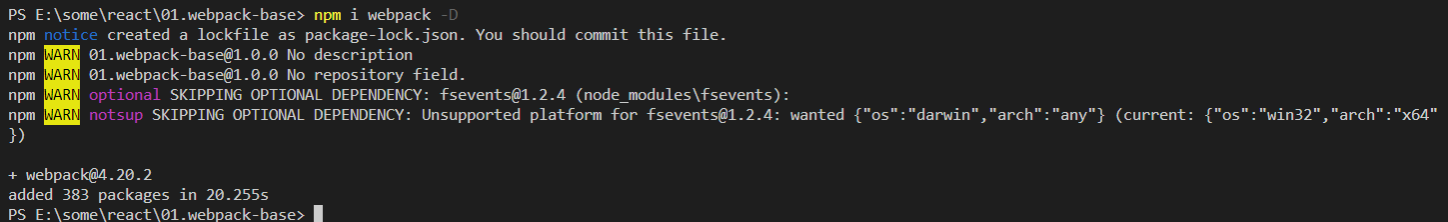


我用cnpm好像报错了。

用了npm就ok了。



结果：



PS E:\some\react\01.webpack-base> npm i webpack -D

npm notice created a lockfile as package-lock.json. You should commit this file.

npm WARN 01.webpack-base@1.0.0 No description

npm WARN 01.webpack-base@1.0.0 No repository field.

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node\_modules\fsevents):

npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

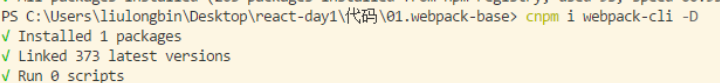
+ webpack@4.20.2

added 383 packages in 20.255s

PS E:\some\react\01.webpack-base>

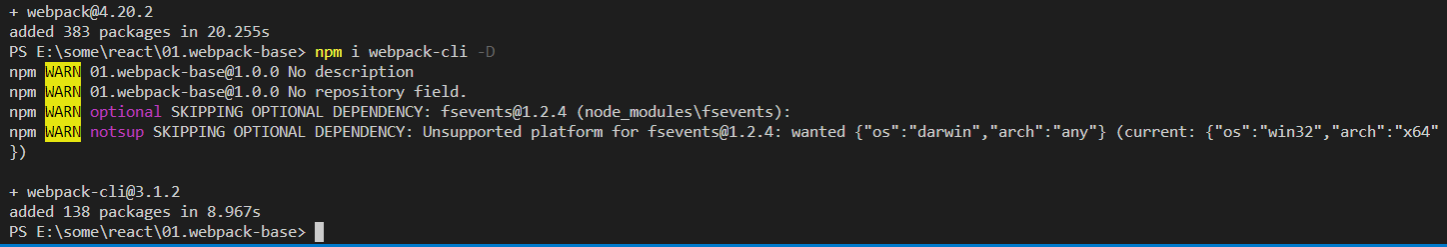
应该是没安装cnpm的缘故。

接着还要输入：



以前是直接打webpack就自动打包了。现在需要cli来进行提供webpack-cli





PS E:\some\react\01.webpack-base> npm i webpack-cli -D

npm WARN 01.webpack-base@1.0.0 No description

npm WARN 01.webpack-base@1.0.0 No repository field.

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node\_modules\fsevents):

npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ webpack-cli@3.1.2

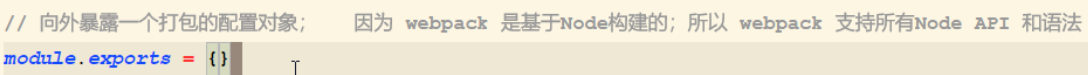
added 138 packages in 8.967s

这是之前用的方式。



webpack有一个配置文件：webpack.config。这里可以写任何合法的js文件。

向外暴露一个配置对象：

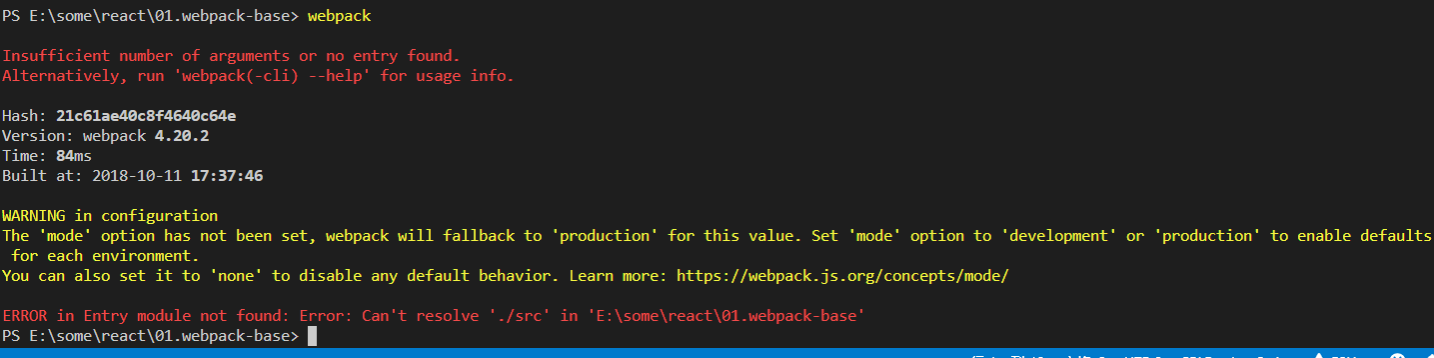


项目中最后一句话就是：

module.exports = config;

将来我们在运行webpack这个命令的时候，它会去项目的根目录找这个webpack.config.js这个文件，然后它会执行这个文件，它一执行，必然就执行到了module.exports{}这段代码，也就向外暴露了这个配置对象。

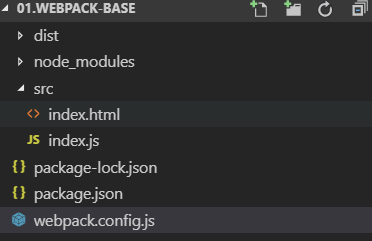
当为空的时候，果然有问题。



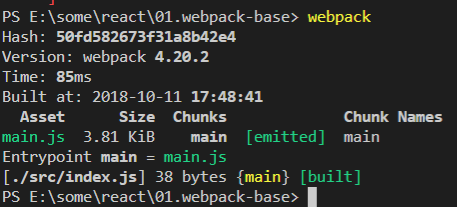
development是开发环境。production是产品环境。



约定大于配置，默认的打包入口路径是src-->index.js，所以main.js就不需要了，而改名为index.js。



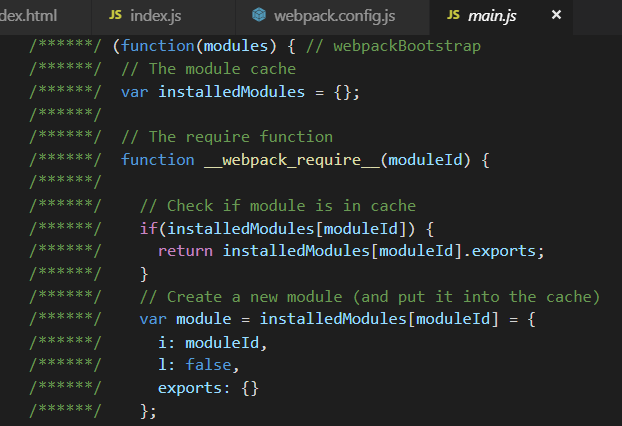
这样就成功了。



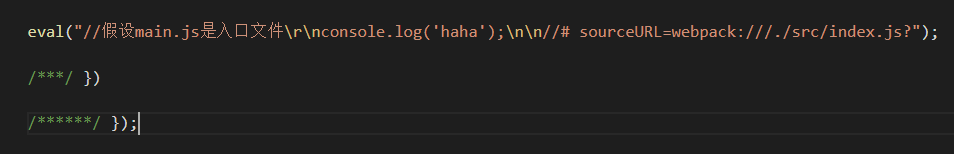
dist中默认就有了main.js



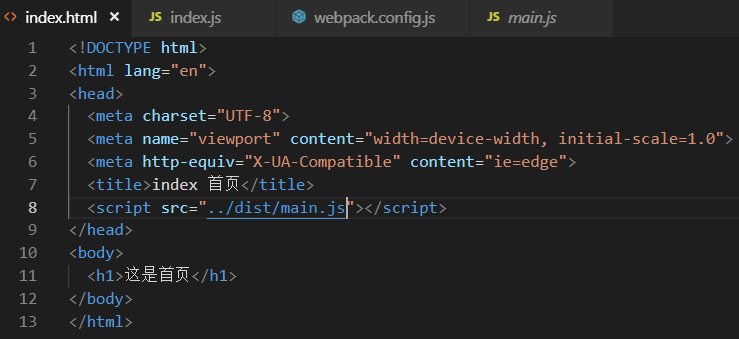
摘录其中部分代码：



里面有我们自己写入index.js中的代码。

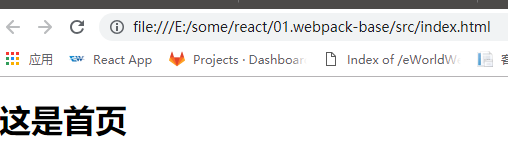


然后，index.html中引用src





打开index.html后：

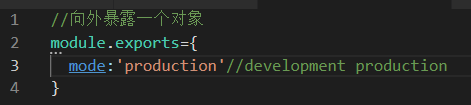


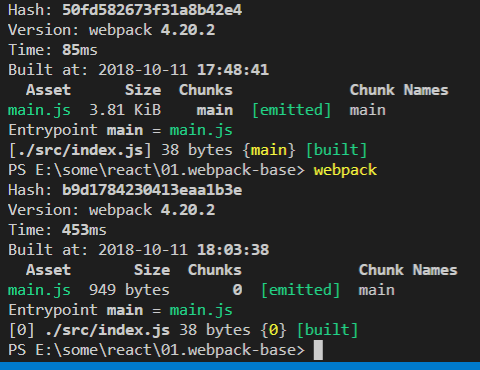
控制台也有想要的内容。

因为是开发模式，文件并没有被压缩：

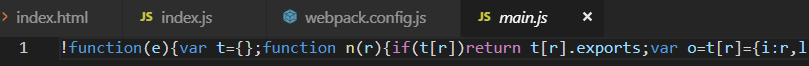


换成产品后，迅速压缩了。从3.81kb变成了949byte。





打开后发现被压缩了：



难怪我在项目中看到的都是这样的代码。

如果仍旧想自己配置，仍旧是可以的。entry+一个路径，比方src下的main.js，照样是可以的。如果我们自己提供了，就是把默认配置给覆盖了。



项目中也找到了：



其中代码：

var path = require('path');

var webpack = require('webpack');

var ExtractTextPlugin = require('extract-text-webpack-plugin');

// var HtmlWebpackPlugin = require('html-webpack-plugin');

var precss = require('precss');

var autoprefixer = require('autoprefixer');

var externalDir = path.join(\_\_dirname, '/app/scripts/external');

var jpegPath = externalDir + '/jpeg.js';

var jpxImagePath = externalDir + '/jpx.js';

var lossLessPath = externalDir + '/lossless.js';

var html2Canvas = externalDir + '/html2canvas.js';

var messengerPath = externalDir + '/messenger.js';

var config = {

cache: true,

entry: {

imageWebViewer: './app/scripts/image.jsx',

ecgWebViewer: './app/scripts/ecg.jsx',

pathologyWebViewer: './app/scripts/pathology.jsx',

},

output: {

filename: '[name].js',

},

module: {

loaders: [

{

test: /\.jsx?$/,

exclude: /(node\_modules|bower\_components)/,

loader: 'babel',

query: { compact: false, },

},

{

test: /\.js?$/,

exclude: /(node\_modules|bower\_components)/,

loader: 'babel',

query: { compact: false, },

},

{

test: /\.less$/,

loader: ExtractTextPlugin.extract([ 'css', 'postcss', 'less', ]),

},

{

test: /\.css$/,

loader: ExtractTextPlugin.extract([ 'css', 'postcss', ]),

},

{

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

loader: 'url-loader?limit=10240&mimetype=image/png&name=../images/[name].[ext]',

},

{

test: /\.(ico|jpg|jpeg|gif|eot|otf|webp|svg|ttf|woff|woff2)(\?.\*)?$/,

exclude: /\/favicon.ico$/,

loader: 'file',

query: {

name: '../fonts/[name].[ext]',

},

},

// {

// test: /worker\.js$/,

// loader: 'worker',

// },

],

noParse: [

lossLessPath,

]

},

postcss: function() {

return [ precss, autoprefixer, ];

},

plugins: [

new ExtractTextPlugin('[name].css'),

new webpack.ProvidePlugin({

'fetch': 'imports?this=>global!exports?global.fetch!whatwg-fetch',

'JpegImage': 'imports?this=>global!exports?global.JpegImage!' + jpegPath,

'JpxImage': 'imports?this=>global!exports?global.JpxImage!' + jpxImagePath,

'html2canvas': 'imports?this=>global!exports?global.html2canvas!' + html2Canvas,

'Messenger': 'imports?this=>global!exports?global.Messenger!' + messengerPath,

}),

new webpack.optimize.CommonsChunkPlugin({

name: 'conmmon',

filename: 'commom.js',

}),

]

};

module.exports = config;