链滴

react->antdesign 学习之路（2）：学习 webpack 工具

作者：wuhongxu

原文链接：https://ld246.com/article/1478773862860

来源网站：链滴

认识webpack

安装

cnpm install webpack -g

-g就是全局安装的意思，安装了之后我们就可以再项目中进行本地安装了。
在项目根目录下，执行cnpm install webpack --save-dev

使用

初识打包

新建静态页面index.html
加入下列代码

```
<html>
   <head>
     <meta charset="utf-8">
   </head>
   <body>
     <script src="bundle.js"></script>
   </body>
</html>
```

再新建一个index.js

```
   document.write('hello ');
```

然后编译index.js并打包到bundle.js

```
webpack index.js bundle.js
```

编译完成之后就可以直接打开index.html，查看效果

打包模块

添加一个module.js

```
module.exports = 'world,!'
```

新建一个index[downline]2.js

```
document.write('hello ')
document.write(require('./module.js')) // 添加模块
```

重新打包编译

```
webpack index_2.js bundle.js
```

刷新后就能看到hello world！字样

引用官方的话

Webpack 会分析入口文件，解析包含依赖关系的各个文件。

这些文件（模块）都打包到 bundle.js 。Webpack 会给每个模块分配一个唯一的 id 并通过这个 id 引和访问模块。

在页面启动时，会先执行 index.js 中的代码，其它模块会在运行 require 的时候再执行。

loader

认识

Webpack 本身只能处理 JavaScript 模块，如果要处理其他类型的文件，就需要使用 loader 进行转
。

Loader 可以理解为是模块和资源的转换器，它本身是一个函数，接受源文件作为参数，返回转换的果。这样，我们就可以通过 require 来加载任何类型的模块或文件，比如 CoffeeScript、 JSX、 LESS 或图片。

先来看看 loader 有哪些特性?

- Loader 可以通过管道方式链式调用，每个 loader 可以把资源转换成任意格式并传递给下一个 load r ，但是最后一个 loader 必须返回 JavaScript。
- Loader 可以同步或异步执行。
- Loader 运行在 node.js 环境中，所以可以做任何可能的事情。
- Loader 可以接受参数，以此来传递配置项给 loader。
- Loader 可以通过文件扩展名（或正则表达式）绑定给不同类型的文件。
- Loader 可以通过 npm 发布和安装。
- 除了通过 package.json 的 main 指定，通常的模块也可以导出一个 loader 来使用。
- Loader 可以访问配置。
- 插件可以让 loader 拥有更多特性。
- Loader 可以分发出附加的任意文件。

使用

我们首先要讲css文件也看做成一个模块

然后我们使用css-loader来读取

新建一个style.css文件

```
body { background: yellow; }
```

将index2.js修改一下

```
require("!style!css!./style.css") // 载入 style.css
document.write('hello ')
```

```
document.write(require('./module.js'))
```

安装loader

```
cnpm install css-loader style-loader
```

然后重新编译一次，就发现我们已经使用了loader了

如果每次 require CSS 文件的时候都要写 loader 前缀，是一件很繁琐的事情。我们可以根据模块类（扩展名）来自动绑定需要的 loader。

将 index2.js 中的 require("!style!css!./style.css") 修改为 require("./style.css")，然后执行：

```
webpack entry.js bundle.js --module-bind 'css=style!css'
# 有些环境下可能需要使用双引号,我就是用的双引号，否则会报错（windows平台，不知道是不是个原因）
$ webpack entry.js bundle.js --module-bind "css=style!css"
```

这里插一句

idea中不支持require是因为没有开启node.js的原因

1. 在Node中开启Node.js Core library

settings->Languages & Frameworks->Node.js Core library设置成enable

2. 在JS中开启Node.js Core library

settings->Languages & Frameworks->JavaScript->Libraries,将Node.js Core前的enable选中.

学习使用配置文件简化操作

创建一个webpack.config.js文件

```
var webpack = require('webpack');

module.exports={
    entry:'./index2.js',
    output:{
        path:[downline][downline]dirname,
        filename:'bundle.js'
    },
    module:{
        loaders:[
            {test:/.css$/,loader:'style!css'}
        ]
    }
}
```

现在我们就可以直接使用webpack进行打包，而不用跟任何的参数了，大大简化了我们反复编译带来成本了

插件使用

插件可以完成更多 loader 不能完成的功能。

插件的使用一般是在 webpack 的配置信息 plugins 选项中指定。

Webpack 本身内置了一些常用的插件，还可以通过 npm 安装第三方插件。

接下来，我们利用一个最简单的 BannerPlugin 内置插件来实践插件的配置和运行，这个插件的作用给输出的文件头部添加注释信息。

修改 webpack.config.js，添加 plugins：

```
var webpack = require('webpack')

module.exports = {
  entry: './entry.js',
  output: {
    path: [downline][downline]dirname,
    filename: 'bundle.js'
  },
  module: {
    loaders: [
      {test: /\.css$/, loader: 'style!css'}
    ]
  },
  plugins: [
    new webpack.BannerPlugin('this is a banner')
  ]
}
```

直接运行（我的没出来什么鬼QAQ，我看了源码都有BannerPlugin,结果显示不出来，是为什么，求点啊， @Vanessa ）