链滴

webpack 初体验

作者：flowaters

原文链接：https://ld246.com/article/1542428771097

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

背景

webpack是一个模块打包器，用于编译 JavaScript 模块。

初体验

安装

```
mkdir webpack-demo && cd webpack-demo
npm init -y
npm install --save-dev webpack webpack-cli
```

查看版本: 本文中版本为 4.25.1

```
$ node_modules/.bin/webpack -v
4.25.1
```

说明：

安装一个要打包到生产环境的安装包时，你应该使用 npm install --save

安装一个用于开发环境的安装包（例如，linter, 测试库等），你应该使用 npm install --save-dev

无webpack时的页面文件

两个文件

```
index.html
src
└── src/index.js
```

index.html

index.html，内容如下：

```
<!doctype html>
<html>
  <head>
    <title>起步</title>
    <script src="https://unpkg.com/lodash@4.16.6"></script>
  </head>
  <body>
    <script src="./src/index.js"></script>
  </body>
</html>
```

src/index.js

src/index.js内容如下：

```
function component() {
```

```
  var element = document.createElement('div');

  // Lodash（目前通过一个 script 脚本引入）对于执行这一行是必需的
  element.innerHTML = _.join(['Hello', 'webpack'], ' ');

  return element;
}

document.body.appendChild(component());
```

效果验证

用浏览器打开 index.html，效果如下：

Hello webpack

适配到webpack上

安装 lodash

上面是通过script脚本引入的外部lodash库。

这里我们希望把lodash打包到本地的js文件中，不再依赖外部js脚本。

因此先安装 lodash 库。

```
$ npm install --save lodash
```

因为 lodash 要发布，所以用为--save。如果只用于开发而不发布的话，那就用 --save-dev

dist/index.html

将 index.html 移动到 dist 目录下，内容如下：

```
<!doctype html>
<html>
   <head>
      <title>起步</title>
   </head>
   <body>
      <script src="main.js"></script>
   </body>
</html>
```

src/index.js

src/index.js，内容如下：

```
import _ from 'lodash';

function component() {
```

```
    var element = document.createElement('div');

    // Lodash, now imported by this script
    element.innerHTML = _.join(['Hello', 'webpack'], ' ');

    return element;
}

document.body.appendChild(component());
```

打包

```
$ npx webpack
Hash: b75da8f7583b35825ca3
Version: webpack 4.25.1
Time: 2828ms
Built at: 11/17/2018 11:57:02 AM
  Asset      Size  Chunks             Chunk Names
main.js  70.5 KiB       0  [emitted]  main
Entrypoint main = main.js
[1] ./src/index.js 264 bytes {0} [built]
[2] (webpack)/buildin/global.js 489 bytes {0} [built]
[3] (webpack)/buildin/module.js 497 bytes {0} [built]
    + 1 hidden module
```

可见生成了 dist/main.js文件

效果验证

用浏览器打开 index.html，效果如下：

```
Hello webpack
```

同时，通过检查，可以看到没有对外部js文件的请求。

原理

执行npx webpack打包里，会将入口起点下的脚本，打包到输出目录下。

如果不使用默认值的话，我们可以通过配置文件来指定其具体的值。

定制配置

如果我们要输出到 bundle.js 文件，而不是 main.js 文件，则可以这样做：

编辑文件 webpack.config.js，内容如下：

```
const path = require('path');

module.exports = {
    entry: './src/index.js',
    output: {
```

```
        filename: 'bundle.js',
        path: path.resolve(__dirname, 'dist')
    }
};
```

执行打包

```
npx webpack --config webpack.config.js
Hash: 33fcbd13f7bbedec8155
Version: webpack 4.25.1
Time: 2723ms
Built at: 11/17/2018 12:14:50 PM
    Asset      Size  Chunks             Chunk Names
bundle.js  70.5 KiB       0  [emitted]  main
Entrypoint main = bundle.js
[1] ./src/index.js 264 bytes {0} [built]
[2] (webpack)/buildin/global.js 489 bytes {0} [built]
[3] (webpack)/buildin/module.js 497 bytes {0} [built]
    + 1 hidden module
```

可见生成了 dist/bundle.js文件。

使用npm脚本打包

除了使用这样cli方式进行打包外，还可以更方便的使用npm脚本进行打包。

编辑 package.json 文件，增加 script.build一行，如下：

```
{
  "name": "webpack-demo",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "webpack": "^4.25.1",
    "webpack-cli": "^3.1.2"
  },
  "dependencies": {
    "lodash": "^4.17.11"
  }
}
```

打包：

```
npm run build

> webpack-demo@1.0.0 build /Users/note/web/webpack-demo
```

```
> webpack

Hash: 33fcbd13f7bbedec8155
Version: webpack 4.25.1
Time: 368ms
Built at: 11/17/2018 12:24:04 PM
    Asset      Size  Chunks             Chunk Names
bundle.js  70.5 KiB       0  [emitted]  main
Entrypoint main = bundle.js
[1] ./src/index.js 264 bytes {0} [built]
[2] (webpack)/buildin/global.js 489 bytes {0} [built]
[3] (webpack)/buildin/module.js 497 bytes {0} [built]
    + 1 hidden module
```

可见同时使用了webpack.config.js文件，生成了bundle.js文件。

下一步

更详细的配置，请参见原文webpack 中文文档

参考

- webpack对比
- webpack起步
- Webpack掉坑之路（1）——HelloWorld（基于npm,react）: 还没看，准备看
- 入门 Webpack，看这篇就够了: 还没看，准备看