**CSS Module**

它不是将css改造成编程语言，而是功能很单纯，加入了局部作用域和模块依赖，这恰恰是网页组件最急需的功能。

规则少同时又非常有用，保证某个组件的样式，不会影响到其他组件。

一、局部作用域

Css的规则都是全局的，任何一个组件的样式规则，都对整个页面有效。

产生局部作用域的唯一方法，就是使用一个独一无二的class的名字，不会与其他选择器重名。这就是CSS module的做法。

App.js

import React from 'react';

import style from './App.css';

export default () => {

return (

<h1 className={style.title}>

Hello World

</h1>

);

};

引入css文件App.css输入到style对象，然后引用style.title代表一个 class。

.title {

color: red;

}

构建工具会将类名style.title编译成一个哈希字符串。

<h1 class="\_3zyde4l1yATCOkgn-DBWEL">

Hello World

</h1>

//App.css

.\_3zyde4l1yATCOkgn-DBWEL {

color: red;

}

这样一来，这个类名就独一无二了，只对该App组件有效。

Webpack对其支持最好。

module.exports = {

entry: \_\_dirname + '/index.js',

output: {

publicPath: '/',

filename: './bundle.js'

},

module: {

loaders: [

{

test: /\.jsx?$/,

exclude: /node\_modules/,

loader: 'babel',

query: {

presets: ['es2015', 'stage-0', 'react']

}

},

{

test: /\.css$/,

loader: "style-loader!css-loader?modules"

},

]

}

};

该配置文件中最关键一行"style-loader!css-loader?modules"在css-loader后面加了一个查询参数modules,表示打开CSS Modules功能。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*不重要

局部作用域显式写法:local(.className)等同于.className

二、全局作用域

使用:global(.className)的语法来声明一个全局规则。凡是被这样声明的class,都不会被编译成哈希字符串。

.title {

color: red;

}

:global(.title) {

color: green;

}

此时使用普通的class的写法，就会引用全局class

import React from 'react';

import styles from './App.css';

export default () => {

return (

<h1 className="title">

Hello World

</h1>

);

};

Class的组合

一个选择器可以继承另一个选择器的规则，称为组合(composition)

.className {

background-color: blue;

}

.title {

composes: className;

color: red;

}

在App.css中，让title继承.className

输入其他模块

选择器当然也可以继承其他css文件里面的规则

//another.css

.className {

background-color: blue;

}

//App.css可以继承another.css里面的规则

.title {

composes: className from './another.css';

color: red;

}