**1.CommonJS和ES中的模块系统**

模块：Module，用于封装特定的数据和操作，防止污染全局对象；

JavaScript历史上曾经出现过如下四个模块标准：  
 ①CommonJS模块标准：module.exports + require()，支持该标准的有Node.js环境

②ES6模块标准：export + import，目前浏览器还不能直接支持ES6模块标准  
 ③AMD模块标准：define()，支持该标准的有RequireJS框架  
 ④CMD模块标准：define()+require()，支持该标准的有SeaJS框架

下面具体来比较一下最常用的CommonJS规范和ES6规范中模块的导出和导入语法：

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CommonJS规范** | **ES6规范** | |
| 导出内容 | //x.js  module.exports = {  age: 10,  print: function(){}  }  } | //x.js  export var age=10;  export function print(){} | //x.js  export default {  age: 10,  print: function(){}  } |
| 导入内容 | //y.js  const obj = require('./x')  obj.age  obj.print( ) | //y.js  import {print,age} from './x'  age  print( ) | //y.js  import obj from './x'  obj.age  obj.print( ) |
| 导出数量 | 只能导出一个对象 | 可以导出多个变量或方法 | 默认导出只能有一个 |
| 使用限制 | 可以是最顶层或内部代码 | 只能是最顶层代码 | |
| 支持情况 | 服务器端技术：可以被Node.js解释器直接理解 | 客户端技术：目前还不能被各大浏览器直接支持，只能使用编译器进行预处理转换为标准JS代码 | |

注意：ES6中的模块必需使用编译器（例如Webpack）转换为ES5中代码，才能间接在浏览器中使用

**2.Webpack和前端工程化**

**手册：**https://www.webpackjs.com/

**概念**：Webpack 是一个现代 JavaScript 应用程序的静态模块打包器。当 webpack 处理应用程序时，它会递归地构建一个依赖关系图，其中包含应用程序需要的每个模块(可能是JS/CSS/图片等)，然后将所有这些模块打包成一个资源包/资源簇(bundle)文件。

**用途**：资源打包后，再供客户端一次性加载，减少请求次数和请求数据总大小，从而提高访问效率。

**注意**：Webpack本身用于打包静态资源，通常的使用场景是：项目初步开发完成后，运行Webpack(是一个NPM扩展包)，把需要处理的静态资源打包，把得到的打包文件和HTML文件一起发布到WEB服务器(可以是Node / PHP / Java等)上供客户端访问。

**核心概念：**

(0)mode：运行模式，可以指定为production或者development

(1)entry：入口，指定资源包的入口文件

(2)output：输出，指定最终输出的资源包文件

(3)loader：加载器，让Webpack可以处理非JS文件，如CSS/图片等  
 (4)plugin：插件，可以进一步扩展Webpack本身的功能

**使用过程**：

(1)创建客户端项目，编写HTML/CSS/JS/图片等文件

(2)创建项目的描述文件：package.json

npm init

(3)下载webpack及其依赖的模块，并添加为当前项目的“开发阶段依赖模块”

npm i webpack --save-dev

npm i webpack-cli --save-dev

(4)创建webpack主配置文件： webpack.config.js

module.exports = {

mode: 'production',

entry: './src/index.js',

output: {

path: \_\_dirname + '/dist',

filename: 'bundle.js'

},

module: { },

plugins: [ ]

}

(5)运行webpack，根据主配置文件对静态资源进行打包

node ./node\_modules/webpack/bin/webpack.js

或者配置为scripts： "build": "webpack"，用 npm run build方式运行

课后任务：

①查看Webpack官网手册，掌握“图片文件”打包所必需的加载器

②查看Webpack官网手册，掌握常见插件的使用方法