12-9

AMD 规范全称是 Asynchronous Module Definition，即异步模块加载机制。

AMD 是 RequireJS 在推广过程中对模块定义的规范化产出。

AMD 规范则是非同步加载模块，允许指定回调函数。

AMD 推荐的风格通过返回一个对象做为模块对象。

AMD 推崇依赖前置，CMD 推崇依赖就近。

AMD 的 API 默认是一个当多个用，

CMD 的全称是 Common Module Definition，即通用模块定义，其提供了模块定义和按需加载执行模块。该规范明确了模块的基本书写格式和基本的交互规则。

CMD 是 SeaJS 在推广过程中对模块定义的规范化产出。

CMD 规范中，一个文件就是一个模块，使用 define 来进行模块。

CMD 直接使用 require 加载属于是同步加载，require 提供了 async 方法来在模块内部进行也不加载模块，并在加载完成以后执行指定的回调函数。

注意：require 是同步往下执行，require.async 则是异步回调执行。require.async 一般用来加载可延迟异步加载的模块。

CMD 的 API 严格区分，推崇职责单一。

虽然 AMD 也支持 CMD 的写法，同时还支持将 require 作为依赖项传递，但 RequireJS 的作者默认是最喜欢上面的写法，也是官方文档里默认的模块定义写法。

AMD 里，require 分全局 require 和局部 require，都叫 require。CMD 里，没有全局 require，而是根据模块系统的完备性，提供 seajs.use 来实现模块系统的加载启动。

同样Seajs也是预加载依赖js跟AMD的规范在预加载这一点上是相同的，明显不同的地方是调用，和声明依赖的地方。AMD和CMD都是用difine和require，但是CMD标准倾向于在使用过程中提出依赖，就是不管代码写到哪突然发现需要依赖另一个模块，那就在当前代码用require引入就可以了，规范会帮你搞定预加载，你随便写就可以了。但是AMD标准让你必须提前在头部依赖参数部分写好（没有写好？ 倒回去写好咯）。这就是最明显的区别。

方案	优势	**劣势**	**特点**
AMD	速度快	会浪费资源	预先加载所有的依赖，直到使用的时候才执行
CMD	只有真正需要才加载依赖	性能较差	直到使用的时候才定义依赖

CommonJS 是服务器端模块的规范，Node.js 采用了这个规范。

CommonJS 规范加载模块是同步的，也就是说，只有加载完成，才能执行后面的操作。

CommonJS 模块的特点有所有代码都运行在模块作用域，不会污染全局作用域。

CommonJS 的风格通过对 module.exports 或 exports 的属性赋值来达到暴露模块对象的目的。

CommonJS 模块可以多次加载，但是只会在第一次加载时运行一次，然后运行结果就被缓存了，以后再加载，就直接读取缓存结果。要想让模块再次运行，必须清除缓存。

CommonJS 模块加载的顺序，按照其在代码中出现的顺序。

了解即可

```
// CMD
define(function(require, exports, module) {
  var a = require("./a");
  a.doSomething();
  // 此处略去 100 行
  var b = require("./b"); // 依赖可以就近书写
  b.doSomething();
  // ...
});
```

```
// AMD 默认推荐的是
define(['./a', './b'], function(a, b) { // 依赖必须一开始就写好
a.doSomething()
// 此处略去 100 行
b.doSomething()
...
})
```

比如 AMD 里，require 分全局 require 和局部 require，都叫 require。CMD 里，没有全局 require，而是根据模块系统的完备性，提供 seajs.use 来实现模块系统的加载启动。CMD 里，每个 API 都简单纯粹。

Vue 使用的是 AMD 规范 源码 第 8 行 define.amd