### 模块化编程

模块化是一种将系统分离成独立功能部分的方法

优点：   
1、减少全局变量及命名冲突

2、防止牵一发而动全身的现象发生   
3、方便单个模块功能调试、升级   
4、多人协作互不干扰

5、方便代码迁移和改造

6、代码重用  
7、方便单元测试

……  
  
缺点： 性能损耗 

var moduleAAttr1, moduleBAttr1;

function moduleAMethod1(){

}

function moduleAMethod2(){

}

function moduleBMethod1(){

}

var moduleA = {

attr1:"",

method1:function(){

},

method2:function(){

}

};

var moduleB = {

attr1:"",

method1: function(){

}

}

var moduleA = (function($, \_){

var attr1 = "";

function method1(){

}

return {

attr2: "",

method2: function(){

}

}

}(jquery, underscore));

var moduleA = (function(){

var attr1 = "";

function method1(){

}

function method2(){

}

return {

method2: method2

}

}());

var moduleA = (function(){

var privateAttr1 = "";

function privateMethod1(){

method2();

}

function publicMethod2(){

console.log("hello");

}

function publicMethod3(){

method1();

}

return {

method2: publicMethod2,

method3: publicMethod3

}

}());

moduleA.method2 = function(){

console.log("world");

}

var moduleA = function(){

var attr1 = "";

function method1(){

method2();

}

function method2(){

console.log("hello");

}

function method3(){

method1();

}

return {

method2: method2,

method3: method3

}

};

var moduleA = (function(){

var attr1 = "";

function method1(){

result.method2();

}

var result = {

method2: function(){

console.log("hello");

},

method3: function(){

method1();

}

}

return result;

}());

// 库的导入

var moduleA = (function($, \_){

var attr1 = "";

function method1(){

}

return {

attr2: "",

method2: function(){

}

}

}(jquery, underscore));

// 模块扩展

var moduleA = (function(m){

m.method3 = function(){

}

return m;

}(moduleA));

// 放大扩展

var moduleA = (function(m){

m.method3 = function(){

}

return m;

}(moduleA || {}));

模块的重用度

1. 无重用

2. 不同时出现

3. 可同时出现

模块独立性的两个定性标准

耦合性也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。

耦合性分类(低――高):

1 无直接耦合:

2 数据耦合: 指两个模块之间有调用关系，传递的是简单的数据值，相当于高级语言的值传递;

3 标记耦合: 指两个模块之间传递的是数据结构，如高级语言中的数组名、记录名、文件名等这些名字即标记，其实传递的是这个数据结构的地址;

4 控制耦合: 指一个模块调用另一个模块时，传递的是控制变量（如开关、标志等），被调模块通过该控制变量的值有选择地执行块内某一功能;

5 公共耦合: 指通过一个公共数据环境相互作用的那些模块间的耦合。公共耦合的复杂程序随耦合模块的个数增加而增加。

6 内容耦合: 这是最高程度的耦合，也是最差的耦合。当一个模块直接使用另一个模块的内部数据，或通过非正常入口而转入另一个模块内部。

内聚性又称块内联系。指模块的功能强度的度量，即一个模块内部各个元素彼此结合的紧密程度的度量。

内聚性匪类(低――高):

1 偶然内聚: 指一个模块内的各处理元素之间没有任何联系。

2 逻辑内聚: 指模块内执行几个逻辑上相似的功能，通过参数确定该模块完成哪一个功能。

3 时间内聚: 把需要同时执行的动作组合在一起形成的模块为时间内聚模块。

4 通信内聚: 指模块内所有处理元素都在同一个数据结构上操作（有时称之为信息内聚），或者指各处理使用相同的输入数据或者产生相同的输出数据。

5 顺序内聚: 指一个模块中各个处理元素都密切相关于同一功能且必须顺序执行，前一功能元素输出就是下一功能元素的输入。

6 功能内聚: 这是最强的内聚，指模块内所有元素共同完成一个功能，缺一不可。与其他模块的耦合是最弱的。