预先加载

Layui的模块加载采用核心的 *layui.use(mods, callback)*方法，当你的JS 需要用到Layui模块的时候，我们更推荐你采用预先加载，因为这样可以避免到处写layui.use的麻烦。你应该在最外层如此定义：

### demo.js[layui.code](http://www.layui.com/doc/modules/code.html" \t "https://www.layui.com/doc/base/_blank)

1. /\*
2. Demo1.js
3. 使用Layui的form和upload模块
4. \*/
5. layui.use(['form', 'upload'], function(){ //如果只加载一个模块，可以不填数组。如：*layui.use('form')*
6. var form = layui.form //获取form模块
7. ,upload = layui.upload; //获取upload模块
9. //监听提交按钮
10. form.on('submit(test)', function(data){
11. console.log(data);
12. });
14. //实例化一个上传控件
15. upload({
16. url: '上传接口url'
17. ,success: function(data){
18. console.log(data);
19. }
20. })
21. });

按需加载（不推荐）

如果你有强迫症，你对网站的性能有极致的要求，你并不想预先加载所需要的模块，而是在触发一个动作的时候，才去加载模块，那么，这是允许的。

### demo.js[layui.code](http://www.layui.com/doc/modules/code.html" \t "https://www.layui.com/doc/base/_blank)

1. button.addEventListener('click', function(){
2. layui.use('laytpl', function(laytpl){ //温馨提示：多次调用use并不会重复加载laytpl.js，Layui内部有做模块cache处理。
3. var html = laytpl('').render({});
4. console.log(html);
5. });
6. });

模块命名空间

layui 的模块接口会绑定在 layui 对象下，内部由 layui.define() 方法来完成。

调用模块可通过 layui.use 方法来实现，然后再通过 layui 对象获得模块接口。如：

### code[layui.code](http://www.layui.com/doc/modules/code.html" \t "https://www.layui.com/doc/base/_blank)

1. layui.use(['layer', 'laypage', 'laydate'], function(){
2. var layer = layui.layer //获得layer模块
3. ,laypage = layui.laypage //获得laypage模块
4. ,laydate = layui.laydate; //获得laydate模块
6. //使用模块
7. });

我们推荐你将所有的业务代码都写在一个大的 use 回调中，而不是将模块接口暴露给全局，比如下面的方式我们是极不推荐的：

### code[layui.code](http://www.layui.com/doc/modules/code.html" \t "https://www.layui.com/doc/base/_blank)

1. //强烈不推荐下面的做法
2. var laypage, laydate;
3. layui.use(['laypage', 'laydate'], function(){
4. laypage = layui.laypage;
5. laydate = layui.laydate;
6. });

扩展一个 layui 模块

第一步：确认模块名，假设为：*mymod*，然后新建一个*mymod.js* 文件放入项目任意目录下（注意：不用放入layui目录）

第二步：编写test.js 如下：

### code[layui.code](http://www.layui.com/doc/modules/code.html" \t "https://www.layui.com/doc/base/_blank)

1. /\*\*
2. 扩展一个test模块
3. \*\*/
5. layui.define(function(exports){
6. //提示：模块也可以依赖其它模块，如：layui.define('layer', callback);
7. var obj = {
8. hello: function(str){
9. alert('Hello '+ (str||'mymod'));
10. }
11. };
13. //输出test接口
14. exports('mymod', obj);
15. });

第三步：设定扩展模块所在的目录，然后就可以在别的JS文件中使用了

1. //config的设置是全局的
2. layui.config({
3. base: '/res/js/' //假设这是你存放拓展模块的根目录
4. }).extend({ //设定模块别名
5. mymod: 'mymod' //如果 mymod.js 是在根目录，也可以不用设定别名
6. ,mod1: 'admin/mod1' //相对于上述 base 目录的子目录
7. });
8. //你也可以忽略 base 设定的根目录，直接在 extend 指定路径（主要：该功能为 layui 2.2.0 新增）
9. layui.extend({
10. mod2: '{/}http://cdn.xxx.com/lib/mod2' // {/}的意思即代表采用自有路径，即不跟随 base 路径
11. })
12. //使用拓展模块
13. layui.use(['mymod', 'mod1'], function(){
14. var mymod = layui.mymod
15. ,mod1 = layui.mod1
16. ,mod2 = layui.mod2;
17. mymod.hello('World!'); //弹出 Hello World!
18. });