一、JavaScript自动加载

①在文本中用onload： 当页面中所有内容（包括图片）加载完后再执行onload，如下：

**[html]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. **<body** onload="alert(1)"**></body>**                    <!-- 当有一个onload -->
2. **<body** onload="alert(2);alert(3);alert(4)"**></body>**  <!-- 当有多个onload用分号隔开,依次弹出 2 3 4 -->

②在脚本中用window.onload： 当页面中的所有内容（包括图片）加载完后再执行window.onload，如下：

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. window.onload = **function**(){...}; //正确写法，这是匿名函数
3. //------------↓多个window.onload的错误写法-------------
4. window.onload = **function**(){alert("text1");}; //不执行
5. window.onload = **function**(){alert("text2");}; //执行
6. //------------↑---------------------------------------
8. //------------↓多个window.onload的正确写法---------------------------
9. window.attachEvent("onload",**function**(){alert('a')});
10. window.attachEvent("onload",**function**(){alert('b')});
11. window.attachEvent("onload",**function**(){alert('c')});
12. //重点提示：在IE浏览器下用 (window.attachEvent)，会弹出 c b a
13. //重点提示：其他浏览器下用 (window.addEventListener)，会弹出 a b c
14. //------------↑-----------------------------------------------------

二、jQuery自动加载  
①当页面中DOM结构（不包含图片）加载完后再执行（可能DOM元素关联的东西并没有加载完），有三种写法，如下：

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. $(document).ready(**function**(){...});//写法1，全称
2. $(**function**(){...});                //写法2，简写
3. jQuery(**function**($){...});          //写法3，简写

②页面中所有元素（包括图片）加载完成才执行，如下。

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. $(window).load(**function**() {...});  //等于JavaScript的写法，如window.onload = function(){...};

③立即执行匿名函数。 当一个匿名函数被括起来，然后再在后面加一个括号，这个匿名函数就能立即运行起来，有两种写法，如下：

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. (**function**(){...})();        //写法1，不加参数
2. (**function**($){...})(jQuery); //写法2，加参数，避免与其他变量发生冲突

解释：

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. **function**(arg){...}; //定义一个匿名函数，参数为arg

调用此函数时，在函数后面写上括号和实参，由于操作符的优先级，函数本身也需要用括号，如下：

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. (**function**(arg){...})(param);//arg是形参，param是实参

相当于定义了一个参数为arg的匿名函数，并且将param作为参数来调用这个匿名函数。 所以下面的写法总结为：只在形参使用$，是为了不与其他库冲突，所以实参用jQuery，如下。

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. (**function**($){...})(jQuery);

说白了，如下：

**[javascript]** [view plain](https://blog.csdn.net/namejs/article/details/50885922) [copy](https://blog.csdn.net/namejs/article/details/50885922)

1. funtion show(s){...};       //相当于这种写法
2. show(jQuery);
4. //-------------------------------------------
6. **var** show=**function**(s){...};  //或者相当于这种写法
7. show(jQuery);