(function(){

}(typeof window !== "undefined" ? window : this, function( window, noGlobal ) {

} ) );

(function(){

}(参数1,参数2));

a = typeof window !== “underfined” ? window : this

b = function(window,noGlobal){…….}

(function(global,factory){

……..

}(a ,b);

形参global == 实参a

形参factory == 实参b

if ( typeof module === "object" && typeof module.exports === "object" ) { //为了兼容node.js作的判断，把JQ功能用node.js创建方式创建

module.exports = global.document ? AA : BB;//判断当前环境是否支持window.document属性,支持执行AA，不支持执行BB

} else {

factory( global ); //条件为假直接运行

}

AA = factory( global, true )

BB = function( w ) {

if ( !w.document ) {

throw new Error( "jQuery requires a window with a document" );

}

return factory( w );

}

jQuery框架的核心：从HTML该当中匹配元素并对其执行操作

命名空间：JS中没有命名空间，但可以通过闭包实现不污染全局变量

JQuery使用一个匿名函数形成一个闭包，然后该匿名函数自调用，所有的代码都在这个闭包中完成，但是无法在其他地方访问该闭包内的变量，所以将window对象传入该匿名函数，将jQuery对象进行暴露，同时给jQuery对象赋了一个别名$，这样就可以通过暴露的JQuery或者$访问闭包内的其他变量

(function(window,undefined){

if( typeof window === “object” && typeof window.document === “object”){

window.jQuery = window.$ = jQuery;

}

}(window);

自调用时传入window对象，这两个原因：

1、使用window由全局变量变为局部变量，当在jQuery代码块中访问window对象时，不需要将作用域链回退观察家顶层作用域

2、将window作为参数传入，可以在压缩代码时进行优化

undefined

1、避免undefined被赋值，确保在匿名函数的作用域中，它是未定义的

2、压缩代码时可节省空间

变量的声明

\_jQuery = window.jQuery = \_$ = window.$

申明变量时，jQuery先将window对象下的jQuery和$存储到临时变量\_jQuery和\_$，为了防止命名冲突，防止在导入其他库的时候命名出现重复导致一个库无法被调用。

deletedIds,class2tpe,support，将这三种变量类型常用的方法进行存储，再利用call/apply进行方法的借用。

jQuery = function(selector,context){ //构造函数

return new jQuery.fn.init(selector,context,rootjQuery);

}

可以看到jQuery对象相当于一个对jQuery.fn.init对象进行实例化的工厂，每次调用jQuery时，就返回一个init对象的实例，这样就实现了免new调用jQuery对象

jQuery.fn = jQuery.prototype = { //jQuery.fn（没什么意义）只是指向jQuery对象的原型

init:function(selector,context.rootjQuery){

当用jQuery.prototype.init()调用时，this指向对象jQuery.prototype本身

return this; //返回的是jQuery.prototype对象

}

}

jQuery.fn.init.prototype = jQuery.fn //让init继承了jQuery对象上的方法。所以调用jQuery时，返回的init的实例就相当于继承了jQuery原型上的所有的方法。

如果让jQuery返回jQuery的话就会形成一个死循环，其次如果返回init实例，init是jQuery原型下的一个方法，init中的this就是jQuery对象，这样就能轻松的实现链式调用，只要在每个方法中返回this,也就是返回了jQuery对象。