李南江亲授-jQuery+Ajax从放弃到知根知底

<https://www.bilibili.com/video/av22807707?p=1>

1. 原生JS会等DOM元素和图片都加载完才执行。jQuery会等到DOM元素加载完，但不会等待图片加载完就执行，也就是无法获取图片的参数（但新版本中可以获取）。
2. 原生JS如果有多个入口函数，后面的会覆盖前面的。jQuery如果有多个入口函数，则不会覆盖，且先执行前面的，再执行后面的。
3. 入口函数（ready）的4中写法：
   1. $(document).ready(function(){ });
   2. jQuery(document).ready(function(){ });
   3. $(function(){ }); 推荐。
   4. jQuery(function(){ });

ready函数默认在DOM元素加载完毕后执行。

1. 当 $ 与某些因素冲突时，在使用jQuery前用 jQuery. noConflict(); 语句来释放 $ 的使用权，然后在jQuery中要用到 $ 的地方全都用 jQuery 代替。

也可以自定义一个符号来代替 $ ，例如用czh来代替：var czh = jQuery. noConflict();

1. $( ); 调用jQuery的核心函数。
2. 接收一个函数。$(function () { alert(“1”); })
3. 接收一个字符串，返回一个jQuery对象

1.字符串选择器；var aa = $(“.box1”); 对象中保存了找到的DOM元素

2.代码片段。var bb = $(“<p>我是段落</p>”); 对象中保存了创建的DOM元素

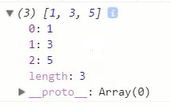
1. 接收一个原生的DOM元素，被包装成一个jQuery对象返回

var span = document.getElementsByTagName(“span”)[0];

var cc = $(span);

1. jQuery对象是一个伪数组（有0 ~ length-1的属性，并且由length属性）。

以数组[1, 3, 5]为例。



1. 原生JS的forEach方法只能遍历数组，不能遍历伪数组。$.each(arr, function(index, value){ … }) 可以遍历数组、伪数组。注意：arr是数组/伪数组名，function中第一个参数是遍历到的索引值index，第二个参数是遍历到的元素值value；而原生的forEach第一个参数是value，第二个参数是index。默认的返回值是进行遍历的数组/伪数组，不支持在回调函数中通过return对遍历的数组/伪数组进行处理。
2. 原生JS的map方法只能遍历数组，不能遍历伪数组。$.map(arr, function(value, index){ … })可以遍历数组、伪数组。注意：arr是数组/伪数组名，function中第一个参数是遍历到的元素值value，第二个参数是遍历到的索引值index。默认的返回值是一个空数组，可以在回调函数中通过return对遍历的数组/伪数组进行处理，生成一个新的数组返回。
3. var result = $.trim(str); 来去除字符串中两端的空格。注意：str为字符串，$.trim()需要用result接收，返回值为新字符串。
4. var result = $.isWindow(str); 判断传入的对象是否是window对象，返回值为true/false。
5. $.isArray(); 用于判断是否是数组，用法与上同。
6. $.isFunction(); 用于判断是否是方法，用法与上同。
7. 控制jQuery代码块的执行。在ready方法前用 $.holdReady(true); 用来暂停ready方法的执行。在其他地方用$.holdReady(false); 恢复ready方法执行。
8. :empty --> var $div = $.(“div:empty”) 找到既没有内容又没有子元素的指定元素。

:parent --> var $div = $.(“div:parent”) 找到有内容或有子元素的指定元素。

:contains() --> var $div = $.(“div:contains(‘xxx’)”) 找到包含文本内容是’xxx’的div元素。

:has() --> var $div = $.(“div:has(‘span’)”) 找到包含子元素span的div元素。

1. attr(name|pro|key, val|fn) 获取或设置属性节点的值，例：$(“span”).attr(“class”, “box”);

一个参数：获取属性节点的值，无论找到多少个元素，都只会返回第一个元素指定的属性节点的值。

二个参数：设置属性节点的值，找到多少个，就设置多少个；如果设置的属性节点不存在，则添加。

removeAttr() 删除属性节点，多个值之间用空格隔开，会删除所有找到的元素的属性节点，例：$(“span”).removeAttr(“class name”);

1. prop() 方法，用法与attr() 一致。 removeProp() 方法，用法与removeAttr() 一致。

prop() 方法不仅能操作属性，还能操作属性节点。官方推荐，在操作具有true和false两个属性的属性节点时，使用prop()，其他则使用attr()。

1. addClass() 方法，添加类，多个值之间用空格隔开，例：$(“div" ). addClass("class1 class2");

removeClass() 方法，删除类，用法与addClass() 相同。

toggleClass方法，切换类，有则删除，无则添加。

1. html()方法，与原生JS的innerHTML一样。

text()方法，与原生JS的innerText一样。

val()方法，获取value属性。

1. css()方法， CSS样式，例：

$(“div”).css(“width”); 获取属性的值

$(“div”).css(“width”, “50px”); 单个设置样式

$(“div”).css(“width”, “50px”).css(“height”, “100px”); 链式设置

$(“div”).css({ width: “50px”, height: “100px”, … }); 用对象批量设置

1. .width() 获取宽度，.width(“50px”) 设置宽度，其他属性同理。

.offset().left/right/… 获取元素距离窗口的偏移位

.offset({

left: 10 设置

})

.position().left/right/… 只能获取元素距离定位元素的偏移位，不能设置。

1. .scrollTop() 获取滚动的距离。

获取距离浏览器窗口的距离：$(“body”).scrollTop() + $(“html”).scrollTop()【兼容ie】

.scrollTop(300) 设置滚动的距离，值为数字。

设置：$(“body, html”).scrollTop(300)【兼容ie】

1. 两种绑定事件的方式：

* eventName(fn); 例：$(“button”).click(function() { });

编码效率高，但部分JS事件没有实现，不能添加。

* on(“eventName”, fn); 例：$(“button”).on(“click”, function() { });

编码效率低，但所有JS事件都已实现。

* 以上两种方式都可以添加多个相同或者不同类型的事件，不会覆盖。

1. off() 解除事件绑定

* 不传参，例：$(“button”).off() ，指定目标解除所有绑定的事件
* 一个参数，例：$(“button”).off(“click”)，解除所有指定类型的事件
* 二个参数，例：$(“button”).off(“click”, test1)，解除指定类型的指定事件

1. 阻止冒泡：①return false（在子元素中）；②传入参数event，event. stopPropagation() ;

阻止默认行为：①return false（在子元素中）；②传入参数event，event. preventDefault();

1. trigger() 自动触发事件，例：$(".father"). trigger("click"); 前提是之前已经定义click事件。会触发事件冒泡、默认行为。

triggerHandler() 用法与trigger() 相同，但不会触发事件冒泡、默认行为。

例如，如果想要a标签触发默认行为，但不会冒泡，那就用span包住a，然后a用triggerHandler()，span用trigger()。

1. 自定义事件，必须满足2个条件：①通过on绑定；②通过trigger/ triggerHandler触发
2. 事件命名空间，同一个事件可以通过命名为例如 click.czh 、click.abc来进行不同的命名，①必须通过on绑定；②通过trigger/ triggerHandler触发事件。

利用trigger触发子元素带命名空间的事件，父元素带相同命名空间的事件也会被触发，而父元素没有命名空间的事件不会被触发。

利用trigger触发子元素不带命名空间的事件，子元素所有相同类型的事件和父元素所有相同类型的事件都会被触发。

1. 事件委托：在入口函数执行之前就有的元素来监听后续动态添加的元素的某些事件，动态创建的元素不能一开始就监听。

例：$(“ul”).delegate(“li”, “click”, function(){ //其中，ul是委托的对象，li是委托人

console.log($(this.).html());

})

1. mouseover(fn) / mouseout(fn) ：鼠标移入/移出。子元素被移入/移出也会触发父元素的事件（移入子元素 = 移出父元素）。

mouseenter(fn) / mouseleave(fn) ：鼠标移入/移出。但子元素被移入/移出不会触发父元素的事件（移入子元素 ≠ 移出父元素）。

hover(fn, fn) 同时监听移入/移出，第一个函数是移入，第二个函数是移出，但子元素被移入/移出不会触发父元素的事件（移入子元素 ≠ 移出父元素）。

hover(fn) 同时监听移入/移出，移入/移出触发同一个方法。

1. siblings() 排他，选中当前元素的兄弟元素。
2. scroll() 方法监听滚动，可以是$(window).scroll(function(){ });
3. eq(index) 方法，返回带有被选元素的指定索引号的元素，index为元素的索引。
4. show() 、hide() 方法设置动画显示、隐藏，两种方法用法相同，以show()方法为例：

show(1000); 动画效果的时长为1000ms。

show(1000, function () { }); 动画显示完毕后调用函数。

toggle() 方法切换，有则隐藏，无则显示。

1. slideDown() 、slideUp() 、slideToggle()：以从上到下滑动的方式显示、隐藏、切换元素。使用方法与show() 、hide() 方法相同。
2. stop() 方法，停止当前正在运行的动画。在jQuery中如果需要执行动画，建议在执行动画之前先调用stop方法，然后再执行动画。
3. children(filter) 方法返回被选元素的所有直接子元素。filter为选择器表达式。
4. fadeIn()、fadeOut()、fadeToggle方法，淡入、淡出、切换。动画显示完毕后调用函数。

用法以fadeIn()为例：$(“div”).fadeIn(1000, function(){ }); //1000为时间。

fadeTo() 淡入到，指定淡的程度。

用法为：$(“div”).fadeIn(1000, 0.5, function(){ }); //第二个参数为程度。

1. animate() 方法，自定义动画。

$(“…”).animate({ width: 500, height: 500 }, 1000, function(){ })//第一个参数为对象，其中改变后的值，可以是多个属性，同时变化，也可以只有一个属性；第二个参数为动画时长；第三个参数为动画结束后执行的方法。

在动画时长后可添加参数动画节奏 “linear”、”swing”，表示动画为 匀速、先慢后快再慢，默认为”swing”。【该参数在其他动画效果中也可以指定】

第一个参数中可以使用累加属性，例如：{ width: “+=100” }

第一个参数中可以使用特殊值，例如：{ width: “hide” }、{ width: “toggle” }

1. delay() 方法，链式动画效果中进行延迟，例：delay(1000)
2. stop() 方法中可以传入参数，第一个参数为是否完成后续动画，默认为false，值为true时停止后续动画；第二个参数为是否立即完成当前动画，默认为false，值为true时完成当前动画。
3. jQuery.fx.off = true; 关闭页面上所有的动画。
4. jQuery.fx.interval = 100; 设置动画的帧数。
5. 添加结点：

* 内部插入

1. append() 将元素添加到指定元素内部（ul内）的最后

var $li = $(“<li>新增的li</li>”);

$(“ul”).append($li);

1. prepend() 将元素添加到指定元素内部的最前面

用法与append() 相同

1. appendTo() 功能与append() 相同，只是使用方法不同

var $li = $(“<li>新增的li</li>”);

$li.appendTo(“ul”);

1. prepend() 同理

* 外部插入
  + after() 将元素添加到指定元素外部（ul外）的后面

var $li = $(“<li>新增的li</li>”);

$(“ul”).after($li);

* + before() 将元素添加到指定元素外部（ul外）的前面

用法与after() 同

* + insertAfter() 功能与after() 相同，只是使用方法不同

var $li = $(“<li>新增的li</li>”);

$li.insertAfter(“ul”);

* + insertBefore() 同理

1. 删除结点：
   * remove() 方法，删除指定元素，例：$(“div”).remove(); //删除整个div
   * empty() 方法，删除指定元素的内容子元素，$(“div”).remove(); //只剩<div></div>
2. 替换结点：

* replaceWith() 将所选元素替换为指定的元素，例：

var $6 = $(“<h6>我是h6</h6>”);

$(“h1”).replaceWith($6)；

* replaceAll()

var $6 = $(“<h6>我是h6</h6>”);

$6.replaceAll(“h1”)；

1. 复制结点

clone() 方法，clone(false)为浅复制，clone(true)为深复制。

浅复制只会复制元素，深复制同时复制元素和事件。

例：var $li = $("li:first" ).clone(true);

1. change(fn) 方法，fn会在元素失去焦点时调用。
2. 监听内容的实时输入，需要用事件委托。
3. jQuery的本质是一个闭包
4. jQuery为什么要使用匿名立即执行函数？将变量定义在函数作用域内，防止多个框架可能的冲突。
5. jQuery如何让外界访问内部定义的局部变量？window.XXX = XXX;
6. jQuery为什么要给自己传递一个window参数？为了方便后期压缩代码后知道该参数是什么意思，比如压成后参数为w，如果没有window参数就不知道w是什么；自己提供window参数而不需要用别人传进来的，提升查找的效率。
7. jQuery为什么要给自己传递一个undefined参数？IE9以下的浏览器undefined可以被修改，为了保证内部使用的undefined不被修改，所以需要接收一个正确的undefined。
8. 1.传入 ’ ‘ null undefined NaN 0 false，返回空的jQuery对象。

2.字符串：

代码片段:会将创建好的DOM元素存储到jQuery对象中返回。

选择器:会将找到的所有元素存储到jQuery对象中返回。

3.数组：会将数组中存储的元素依次存储到jQuery对象中立返回。

4.除上述类型以外的：会将传入的基本数据类型存储到jQuery对象中返回。

1. Ajax的使用：

var oBtn = document. querySelector(' button" ) ;

oBtn.onclick = function (ev1) {

$. ajax({

type:"GET,

url: 09-ajax jquery.php",

data: "userName=lnj&userPwd=654321”,

success: function (msg) {

alert(msg);

},

error: function (xhr) {

alert(xhr.status);

}

});

}

原理见