//链式编程

对象.方法().方法().方法()

end() 恢复断链之前的状态 （有断链的情况，则不推荐使用链式编程）

jQ对象与原生js的dom的转换

1、原生js转成jQ对象

原生box ------> $(box)

this ------> $(this)

2、jQ转成原生js对象

$(".div") -----> $(".div")[0]

$(".div") -----> $(".div").get(0)

==========

jQuery

DOM获取的是 DOM对象

jQuery获取的是 jQuery对象

两者不相等

---------

选择器

并集选择器

$(".box, .cls, .demo")

交集选择器

$("p.active")

$("#box~p") #box后面的所有兄弟元素p

$(".dv").prev() .dv的上一个兄弟元素

$(".dv").prev("p") .dv的上一个兄弟元素p

$(".dv").prevAll()

$(".dv").prevAll("div")

$(".dv").next() .dv的下一个兄弟元素

$(".dv").next("p") .dv的下一个兄弟元素p

$(".dv").nextAll()

$(".dv").nextAll("div")

$(".dv").siblings() .dv的所有兄弟元素

$(".dv").siblings("li")

$("#dv>p:odd") 奇数

$("#dv>p:even") 偶数

应用：隔行变色

$("#dv>li:eq(索引值)")

$("p").eq(1); //选取第二个 <p> 元素（索引号 1）

$("p").filter(".url"); //返回带有类名 "url" 的所有 <p> 元素

$("p").not(".url"); //返回不带有类名 "url" 的所有 <p> 元素

$("#dv>li:first") #dv里的第一个li元素

$("#dv>li:last")

获取索引值 index()

$("#dv>li").index();

--------

多个js库共存的问题

var xy = $.onConflict();

//把$的权利给力xy了，$就可以作为其他的用法出现在代码中，避免出现冲突

----------

隐私迭代（内部遍历）

each()

$("#dv > li").each(function(index, ele) {

var opacity = (index+1)/10;

$(ele).css("opacity", opacity);

})

-----------

事件

//DOM //jQuery

页面加载事件 onload load()

ready()

jQuery() $()

鼠标进入事件 onmouseenter mouseenter()

鼠标离开事件 onmouseleave mouseleave()

鼠标点击 onclick click()

滚动事件 onscroll scroll()

按下鼠标按钮 onmousedown mousedown()

按下键盘按键 onkeydown keydown()

浏览器窗口大小变化 onresize resize()

input、select、

textarea输入内容改

变时 onchange change()

input输入框输入

内容时 oninput

// DOM中的页面加载事件

window.onload = function() {

console.log("你好");

}

window.onload = function() {

console.log("你好11");

}

注：后面的相同事件会覆盖前面的事件

// jQuer中的加载事件（高版本jQuery可能会不支持）

// 第一种

$(window).load(function() {

consolo.log("nihao");

})

注：jQuery中的相同事件会按顺序执行，不会覆盖

// 第二种

$(document).ready(function() {

console.log("第二种加载事件");

})

注：第二种比第一种快（页面中的基本元素加载后就触发）

// 第三种

jQuery(function() {

console.log("页面加载事件三");

})

//jQuery简写成$

$(function() {

console.log("页面加载事件三");

})

注：第三种和第二种一样块

//鼠标进入事件 （DOM的是 onmouseenter）

$("#box > p").mouseenter(function() {

$(this).css("backgroundColor", "red");

})

//鼠标离开事件 （DOM的是 onmouseleave）

$("#box > p").mouseleave(function() {

$(this).css("backgroundColor", "");

})

//链式编程

$("#box > p").mouseenter(function() {

$(this).css("backgroundColor", "red");

}).mouseleave(function() {

$(this).css("backgroundColor", "");

})

//bind方法绑定事件

$("#box > p").bind("mouseenter", function() {

$(this).css("backgroundColor", "red");

}).bind("mouseleave", function() {

$(this).css("backgroundColor", "");

})

$("#box > p").bind({

"mouseenter": function() {

$(this).css("backgroundColor", "red");

},

"mouseleave": function() {

$(this).css("backgroundColor", "");

}

})

这种方式绑定多个相同事件时会有覆盖现象（后面覆盖前面）

//父元素调用方法，为子元素绑定事件，delegate调用的就是下面的on方法

$("#box").delegate("p","click",function() {

alert("什么，我被点击了");

})

//也是父元素调用方法，为子元素绑定事件

$("#box").on("click","p",function() {

alert("什么，我被点击了");

});

$("#box > p").on("click",function() {

alert("什么，我被点击了");

});

$("#box").undelegate(); //移除p的所有事件

$("#box").undelegate("p","click");

// 移除p的点击事件

$("#box").undelegate("p","click mouseenter");

//移除p的多个事件

$("#box").unbind();

//所有的事件全部移除

$("#box").unbind("click");

//所有的点击事件全部移除

$("#box").unbind("click mouseenter");

//同时解绑多个事件

$("#box").off();

//父级元素和子级元素的所有事件全部解绑

$("#box").off("click");

//父级元素和子级元素的点击事件解绑

$("#box").off("click mouseenter");

//父级元素和子级元素的多个事件解绑

$("#box").off("click", "p");

//p的点击事件解绑

$("#box").off("click mouseenter", "p");

//p的多个事件解绑

$("#box").off("", "p");

//p的所有事件解绑

$("#box").off("click", "\*\*");

//所有的子元素的点击事件都解绑

---------

阻止事件冒泡

$("#dv").click(function() {

console.log($(this).attr("id"));

return false; //阻止冒泡

})

-------

事件参数对象

$("#dv").click(function(e) {

console.log(e);

console.log(e.button); //鼠标的按键值

});

$(document).mousedown(function(e) {

if(e.altKey) {

console.log("按下alt键和鼠标按键");

} else if(e.shiftKey) {

console.log("按下shift键和鼠标按键");

} else {

console.log("按下了鼠标按键");

}

});

$(document).keydown(function(e) {

switch(e.keyCode) {

case 65: $("#dv").css("backgroundColor","orange"); break;

case 66: $("#dv").css("backgroundColor","deeppink"); break;

case 67: $("#dv").css("backgroundColor","hotpink"); break;

}

});

trigger() 方法触发被选元素的指定事件类型

$(window).on("resize", function () { ... } ).trigger("resize");

表示屏幕大小变动之前就执行一次事件

----------------

获取(和设置)css样式

//DOM (属性) //jQuery (方法)

style.width / style.backgroundColor css()

width()

height()

//jQuery

$("#box").css("color","blue");

//键值对的写法

$("#box").css({

"width": "200px",

"height": "300px"

})

//链式编程

$("#box").css("width":"200px").css("height":"300px").css("backgroundColor","blue");

--------

添加(和移除)类（class）

hasClass() 判断元素是否应用了这个类

$(".dv").hasClass("cls");

$(".dv").hasClass("cls cls2");

添加类

//DOM //jQuery

classList.add() addClass()

classList.toggle();

setAttribute()

//DOM

box.classList.add("cls");

box.classList.add("cls1, cls2");

box.setAttribute("class","cls");

//jQuery

$("#dv").addClass("cls");

$("#dv").addClass("cls cls2");

$("#dv").addClass("cls").addClass("cls2");

移除类

//DOM //jQuery

classList.remove() removeClass()

setAttribute()

//DOM

dv.classList.remove("cls");

dv.setAttribute("class","");

//jQuery

$("#dv").removeClass("cls");

$("#dv").removeClass("cls cls2");

$("#dv").removeClass("cls").removeClass("cls2");

toggleClass() 却换类样式

$(".dv").toggleClass("cls");

---------

获取(或设置)元素内容

//DOM （属性） //jQuery （方法）

innerHTML html()

innerText text()

value val()

text() 获取(或设置)元素的文本内容

$(".box").text();

$(".box").text("设置.box元素的文本内容");

-----------

获取(和设置)元素 原有的 及 自定义的(主要) 属性

//DOM //jQuery

getAttribute / setAttribute attr()

//jquery

$("#dv").attr("index");

$("#dv").attr("index",2);

--------

移除元素的属性

//DOM //jQuery

setAttribute() removeAttr()

//DOM

box.setAttribute("id","");

box.setAttribute("index","");

//jQuery

$("#dv").removeAttr("id");

移除元素的id属性

$("#dv").removeAttr("index");

移除元素的自定义属性index

--------

获取单复选框是否选中

prop()

$("#ck").prop("checked"); //false

$("#ck").prop("checked",true);

------------

offset

//DOM //jQuery

offsetLeft offset()

offsetTop

//DOM

box.offsetLeft;

元素自身的左边框外部到offsetParent左边框内部的距离

//jQuery

$("#dv").offset(); // {top:230, left:100}

$("#dv").offset().left; // 230

offset().left = left + margin

----------

scroll

//DOM //jQuery

scrollTop scrollTop()

scrollLeft scrollLeft()

//DOM

box.offsetTop

//jQuery

$("#dv").scrollTop(); //div盒子向上卷曲出去的距离

$(document.body).scrollTop()

$(window).scrollTop();

整个页面向上卷曲出去的值

---------

//DOM //jQuery

width() height()

clientWidth clientHeight innerWidth() innerHeight()

offsetWidth offsetHeight outerWidth() outerHeight()

outerwidth(true) outerheight(true)

//DOM

clientWidth clientHeight 宽高 （center + padding）

offsetWidth offsetHeight 宽高 （center + padding + border）

//jQuery

width() height() 获取当前元素的内容的宽度（center）

innerWidth() innerHeight() 宽高 （center + padding）

outerWidth() outerHeight() 宽高 （center + padding + border）

outerwidth(true) outerheight(true) (center + padding + border + margin)

---------

元素创建

//DOM //jQuery

appendChild() append()

appendTo()

html()

prepend()

prependTo()

after()

//DOM

var ospan = document.body.appendChild(span);

document.body.appendChild(ospan);

box.appendChild(ospan);

追加的节点ospan可以是新创建的，也可以是已存在的

如果ospan是已存在的，追加节点的方式是 剪切粘贴

//jQuery

父级元素.append(子级元素);

$("#dv").append($("<a href=''>百度</a>"));

$("#dv").append("<a href=''>百度</a>");

子级元素.appendTo(父级元素);

$("<span>紫金山</span>").appendTo($("#dv"));

都是添加到末尾

$("#dv").html("<input type='text'>");

以字符串的方式创建，只能创建一个，因相同的会覆盖

prepend()

prependTo()

添加到最前面

after() 添加到目标节点之后，作为下一个兄弟

before() 添加到目标节点之qian前，作为上一个兄弟

---------

元素移除

//DOM //jQuery

removeChild() remove()

html()

empty()

//DOM

box.removeChild(p1);

移除box的子元素p1

//jQuery

remove()

$("#dv").remove(""); //自杀，删除自己

$("#dv").remove(".italic"); //删除clas=".italic"的所有dv元素

html()

$("#dv").html("")

移除所有的子元素

empty()

$("#dv").empty(); 清空

---------

复制元素

//DOM //jQuery

cloneNode() clone()

//DOM

box.cloneNode(); //克隆节点（不包括子节点）

box.cloneNode(true); //包括所有子节点

//jQuery

$("#dv").clone(); //克隆dv的子元素（不包含事件）

$("#dv").clone(true); //包含事件

---------

父元素

//DOM //jQuery

parentNode

parentElement parent()

parents()

parentsUntil()

//DOM

demo.parentNode; //父节点

demo.parentElement; //父元素

//jQuery

$("span").parent(); //返回每个<span>元素的直接父元素

$("span").parents(); //返回所有<span>元素的所有祖先元素

$("span").parents("ul"); //返回所有<span>元素的所有为ul的祖先元素

$("span").parentsUntil("div");

//返回介于 <span> 与 <div> 元素之间的所有祖先元素：

---------

子元素

//DOM //jQuery

find()

childNodes

children children()

firstChild （第一个节点）

firstElementChild （第一个子元素节点） first()

lastChild （最后一个节点）

lastElementChild last()

//DOM

box.childNodes; //box的子节点（包含元素、文本、换行、空格等）

box.children; //box的子元素节点（只包含元素）

//jQuery

$("#dv").find("p");

//返回属于 dv 后代的所有 p 元素(一路向下直到最后一个后代)

$("#dv").find("\*"); //返回 dv 的所有后代

$("#dv").children(); //返回每个 dv 元素的所有直接子元素

$("#dv").children("p"); //返回每个 dv 元素的所有直接子元素p

$("div p").first(); //选取首个 <div> 元素内部的第一个 <p> 元素

$("div p").last(); //选择最后一个 <div> 元素中的最后一个 <p> 元素

注意：上面两个的使用方式

---------

兄弟元素

//DOM //jQuery

siblings()

previousSibling

previousElementSibling prev()

prevAll()

prevUntil()

nextSibling

nextElementSibling next()

nextAll()

nextUntil()

//DOM

box.previousSibling //box的上一个兄弟节点

box.previousElementSibling //box的上一个兄弟元素

box.nextSibling //box的下一个兄弟节点

box.nextElementSibling //box的下一个兄弟元素

//jQuery

$(".demo").siblings(); //返回 .demo 的所有同胞元素

$(".demo").siblings("p"); //返回 .demo 的所有同胞元素p

$("h2").next(); //返回 <h2> 的下一个同胞元素

$("h2").nextAll(); //返回 <h2> 的往下所有跟随的同胞元素

$("h2").nextUntil("h6");

//返回介于 <h2> 与 <h6> 元素之间的所有同胞元素

-----------

动画

show() 显示

hide() 隐藏

toggle()

$("#dv").show();

可以设置参数： 参数1: 时间 参数2：回调函数

$("#dv").show(1000);

$("#dv").show(1000, function() {

console.log("动画执行完成后，再执行回调函数");

});

slideDown() 下拉

slideUp() 上拉

slideToggle()

也有参数，和上面一样

fadeIn() 淡入

fadeOut() 淡出

fadeToggle()

也有参数，和上面一样

fadeTo(时间，结束透明值)

时间参数也可写成 fast、slow、normal

fadeTo(fast，结束透明值)

animate()

参数1：键值对

参数2：时间

参数3：回调函数

$("#dv").animate({"width":"300px","height":"300px","left":"100px"}, 5000);

$("#dv").animate({"width":"300px","height":"300px","left":"100px"}, 5000).animate({"width":"30px","height":"30px","left":"10px"}, 300);

--------

jQuery插件

已经写好的功能，如 轮播图