21.4.13学习记录

一、内容选择器

```
//:empty作用：找到既没有文本内容和子元素的指定元素
// var $div = $("div:empty");
// console.log($div);

//:parent作用：找到有文本或者子元素的指定元素
// var $div = $("div:parent");
// console.log($div);

//:contains(text):找到包含指定文本内容的指定元素
// var $div = $("div:contains('1')");
// console.log($div);

//has:找到包含指定子元素的指定元素
// let $div = $("div:has('span')")
// console.log($div)
```

二、操作属性节点

```
//属性:对象身上保存的变量
//如何操作属性：
    // 对象.属性名称 = 值;
    // 对象.属性名称;
    // 对象["属性名称"] = 值;
    // 对象["属性名称"];
//什么是属性节点
    // 在编写html代码时，在html标签中添加的属性就是属性节点
//如何操作属性节点
    //DOM元素.setAttribute("属性名称","值");
    //DOM元素.getAttrbute("属性名称")
//属性和属性节点有什么区别
```

attr()和removeAttr()

```
//attr(name|pro|key,val|fn)
// 作用：获取或设置属性节点的值
// 传一个参数，代表获取属性节点的值
// 传两个参数，代表设置属性节点的值
//注意：只会返回找到的第一个元素指定的属性节点
//但是设置元素的话就是全部，而且如果设置的属性节点不存在，系统会自动新增

//removeAttr(name)删除属性节点
//删除所有找到元素指定的属性节点,可以同时删除多个属性节点
```

另外两个方法：

```
//prop()
//特点和attr方法一致，还能操作属性节点
//操作有true/false属性的属性节点时，建议用prop
//removeProp()
//特点和removeAttr方法一致
```

三、操作CSS类

```
//addClass:添加类
//removeClass:删除类
//toggleCLass:切换类
$(() => {
    let btns = document.getElementsByTagName("button");
    btns[0].onclick = function () {
        $("div").addClass("class1 class2");
    }
    btns[1].onclick = function () {
        $("div").removeClass("class2 class1")
    }
    btns[2].onclick = function () {
        $("div").toggleClass("class1 class2")
    }
});
```

四、操作css样式

```
//逐个设置
// $("div").css("width", "100px");
// $("div").css("height", "100px");
// $("div").css("height", "100px");

//链式操作
// $("div").css("width", "100px").css("height", "100px").css("backgroundColor", "red");

//批量设置
// $("div").css({
//     width: "100px",
//     height: "100px",
//     backgroundColor: "red"
// })
//获取css
// console.log($("div").css("width"));
```

位置尺寸

```
$(() => {
    let btns = document.getElementsByTagName("button");
    btns[0].onclick = function () {
        // 获取元素距离窗口的偏移量
        // console.log($(".son").offset().left);
        //position方法只能获取不能设置
        //获取元素距离定位元素的偏移量
        console.log($(".son").position().left);
    }
    btns[1].onclick = function () {
        //设置元素距离窗口的偏移量
        $(".son").offset({ left: 10 });
    }
});
```

五、操作HTML文本值

```
let btns = document.getElementsByTagName("button");
btns[0].onclick = function () {
    $("div").html("<p>我是段落<span>我是span</span></p>");
}
btns[1].onclick = function () {
    console.log($("div").html());
}
btns[2].onclick = function () {
    $("div").text("<p>我是段落<span>我是span</span></p>");
}
btns[3].onclick = function () {
    console.log($("div").text());
}
btns[4].onclick = function () {
    $("input").val("请输入内容");
}
btns[5].onclick = function () {
    console.log($("input").val());
}
```

滚动偏移量

```
$(() => {
    let btns = document.getElementsByTagName("button");
    btns[0].onclick = function () {
        //获取滚动的偏移量
        console.log($(".scroll").scrollTop());
        //为了浏览器兼容，获取网页滚动偏移量
        // $("body").scrollTop()+$("html").scrollTop();
    }
    btns[1].onclick = function () {
        //设置滚动的偏移量
        $(".scroll").scrollTop(300);
        //设置网页滚动偏移量,浏览器兼容
        // $("html,body").scrollTop(val);
    }
});
```

事件绑定：

```
//jQ中有两种事件绑定方式
    //eventName(fn)
//部分事件JQ未实现
    //on(eventName,fn)
//所有js事件都可以
//可以给一个节点添加多个相同事件不会覆盖
//还可以同时给一个节点添加多个不同事件
```

事件移除：

```
//off():不传参会移除所有事件
//传递一个参数，会移除所有指定类型的事件
// 两个参数，移除所有指定类型的指定事件
// $("button").off();
$("button").off("click", test1);
```

事件冒泡：

```
//阻止事件冒泡
// return false;
// event.stopPropagation();
```

默认行为：比如a标签链接跳转和submit提交

```
//阻止链接跳转
return false;
event.preventDefault();
```

```
//阻止提交跳转
return false;
e.preventDefault();
```

事件自动触发

```
//事件自动触发
//trigger：会触发事件冒泡,会触发默认行为
//triggerHandler:不会触发冒泡,不会触发默认行为
//但是a标签无论用哪个方法都不会触发默认行为，
//所以若想用此方法自动触发a标签的默认行为，可以在a标签
//内加一个span标签，然后去触发a中的内容
// $(".father").trigger("click");
//$(".father").triggerHandler("click");
```

自定义事件

```
//自定义事件,使用on定义，再使用trigger触发
$(() => {
    $(".son").on("myClick", () => {
        alert("son");
    })
    $(".son").trigger("myClick");
});
```

事件命名空间

```
//事件命名空间，需用on绑定，再用trigger触发
$(() => {
    $(".son").on("click.zs", () => {
        alert("click1")
    })
    $(".son").on("click.ls", () => {
        alert("click2")
    })
    $(".son").trigger("click.zs");
});
```

事件委托

```
//事件委托就是请别人帮忙做事情，然后将做完的结果反馈给我们
$(() => {
    $("button").click(() => {
        $("ul").append("<li>我是新增的li</li>");
    })
    //在JQ中，如果通过核心函数找到的元素不止一个，那么在添加事件的时候
    //JQ会遍历所有的元素，给所有找到的元素添加事件
    // $("ul>li").click(() => {
    //     console.log($(this).html());
    // })
    $("ul").delegate("li", "click", function () {
        console.log($(this).html());
    })
});
```

移入移出事件

```
// $(".father").mouseover(() => {
//     console.log("移入");
// })
// $(".father").mouseout(() => {
//     console.log("移出");
// })

// $(".father").mouseenter(function ()
//     console.log("移入");
// })
// $(".father").mouseleave(function ()
//     console.log("移出");
// })

// $(".father").hover(function () {
//     console.log("father被移入了");
// }, function () {
//     console.log("father被移出了");
// })

$(".father").hover(function () {
    console.log("father被移入移出了");
})
```

显示、隐藏、切换

```
$(() => {
    $("button").eq(0).click(() => {
        // $("div").css("display", "block");
        $("div").show(1000, () => {
            alert("显示完毕")
        });

    })
    $("button").eq(1).click(() => {
        // $("div").css("display", "none");
        $("div").hide(1000, () => {
            alert('隐藏完毕')
        });
    })
    $("button").eq(2).click(() => {
        $("div").toggle(1000, () => {
            alert("切换完毕");
        });
    })
})
```

展开、收起、切换

```
$(() => {
    $("button").eq(0).click(() => {
        $("div").slideDown(1000, () => {
            console.log("展开完毕")
        })
    })
    $("button").eq(1).click(() => {
        $("div").slideUp(1000, () => {
            console.log("收起完毕")
        })
    })
    $("button").eq(2).click(() => {
        $("div").slideToggle(1000, () => {
            console.log("切换完毕")
        })
    })
});
```

折叠菜单

```
$(() => {
    $(".nav>li").click(function () {
        let $sub = $(this).children(".sub");
        $sub.slideDown(1000);
        let $other = $(this).siblings().children(".sub");
        $other.slideUp(1000);
    })

})
```

监听网页滚动事件（广告对联demo）

```
//监听网页滚动事件
$(() => {
    $(window).scroll(() => {
        let offset = $("html,body").scrollTop();
        if (offset >= 500) {
            $("div").show(1000);
        } else {
            $("div").hide(1000);
        }
    })
})
```

停止当前正在运行的动画，

建议在执行动画之前先调用stop方法

```
//停止当前正在运行动画stop
$(() => {
    $(".nav>li").mouseenter(function () {
        let $sub = $(this).children(".sub");
        $sub.stop();
        $sub.slideDown(1000);
    })
    $(".nav>li").mouseleave(function () {
        let $sub = $(this).children(".sub");
        $sub.stop();
        $sub.slideUp(1000);
    })
});
```

淡入淡出

```
$(() => {
    $("button").eq(0).click(function () {
        $("div").fadeIn(1000, function () {
            console.log("淡入完毕");
        })
    })
    $("button").eq(1).click(function () {
        $("div").fadeOut(1000, function () {
            console.log("淡出完毕");
        })
    })
    $("button").eq(2).click(function () {
        $("div").fadeToggle(1000, function () {
            console.log("切换完毕");
        })
    })
    $("button").eq(3).click(function () {
        $("div").fadeTo(1000, 0.5, function () {
            console.log("淡入到0.5完毕");
        })
    })
```

自定义动画：动画节奏

```
//参数1：接受一个对象，可以在对象中修改属性
//参数2：指定动画时长
//参数3：指定动画节奏，默认swing,匀速linear
//参数4：动画执行完毕之后的回调函数
$(() => {
    $("button").eq(0).click(function () {
        $(".one").animate({
            marginLeft: 500
        }, 3000, "linear", function () {
            console.log("自定义动画执行完毕");
        });
        $(".two").animate({
            marginLeft: 500
        }, 3000, function () {
            console.log("自定义动画执行完毕");
        });
    })
```

自定义动画：操作属性、累加属性、还有关键字

```
$("button").eq(1).click(function () {
    $(".one").animate({
        width: "+=100"
    }, 1000, function () {
        console.log("自定义动画执行完毕");
    })
})
$("button").eq(2).click(function () {
    $(".one").animate({
        // width: "hide"
        width: "toggle"
    }, 1000, function () {
        console.log("自定义动画执行完毕");
    })
})
```

delay和stop方法

```
$(() => {
    $("button").eq(0).click(function () {
        $(".one").animate({
            width: 500,
        }, 1000).delay(2000).animate({
            height: 400
        }, 1000)
    })
    $("button").eq(1).click(function () {
        //不传参、false、(f,f)表示立即停止当前动画，继续执行后续动画
        //传true、(t,f)表示立即停止当前和后续动画
        //(f,t)表示立即完成当前，继续执行后续动画
        //(t,t)表示立即完成当前并且停止后续所有动画
        $("div").stop();
    })
})
```