第二周

css

1. 接触overflow属性

```
overflow:visible  //默认值。内容不会被修剪，会呈现在元素框之外。
overflow:scroll  //内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容。
overflow:hidden //内容会被修剪，并且其余内容是不可见的。
overflow:auto   //如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容。
overflow:inherit  //规定应该从父元素继承 overflow    属性的值。
```

2. 接触新边框属性

```
border-shadow: h-shadow v-shadow blur spread color inset;
//水平阴影位置 垂直阴影位置 模糊距离 阴影大小 阴影颜色 从外层的阴影（开始时）改变阴影内侧阴影
border-image:
```

3. background-origin属性

```
background-origin:content-box|padding-box|border-box;
//利用这个属性可以设置盒子不同部分的背景图片
```

4. background-size属性

```
background-size:100px 60px;
background-size:100% 100%
```

5. background-clip属性

```
background-clip: border-box|padding-box|content-box;
```

6. 伪元素：before和：after使用

```
<style>
        .contain{
            width:500px;height:500px;
            margin:0;padding: 0;
        }
        .box1{
            position: relative;
            width:0px;height:0px;
            border-top:90px solid transparent;
            border-right:100px solid black;
```

```
            border-bottom: 100px solid transparent;
            }
            //做三角形
        .box1::after{
            content:'';
            position: absolute;
            top:-90px;
            left:2px;
            border-top:90px solid transparent;
            border-right:100px solid #ffffff;
            border-bottom:100px solid transparent;
        }
        //做箭头标志
</style>
<body>
    <div class="contain">
        <div class="box1"></div>
    </div>
</body>
```

7. 多列属性

```
column-count  //规定元素应该被分隔的列数。
column-gap    //规定如何填充列。
column-rule   //规定列之间的间隔。
```

8. 再练习一下动画

- css部分

```
body{
    background-color:grey;
}
.contain{
    background-color: white;
    color:black;
    border-color:black;
}
.contain div{
    background-color: aqua;
    color:black;
    border-color:black;
}
```

```
*{padding: 0;margin: 0;}
body{
    margin:1em 10%;
}
.contain{
    display:block;
    margin-top:100px;
    width:500px;height:500px;
    overflow: hidden;
```

```
}
.contain div{
    width:50px;height:50px;
    position: relative;
    top:0;left:0;
}
.box1{
    /* animation: name duration timing-function delay iteration-count direction fill-mode; */
    animation-name:show1;
    animation-duration: 3s;
    animation-fill-mode: forwards;
}
@keyframes show1{
    from{
        transform: rotate(0deg);
    }
    to{
        transform: rotate(-360deg);
        width:100%;height:100%;
    }
}
.box2{
    animation:show2 3s forwards 2 alternate;
}
@keyframes show2{
    0%{
        top:0;left:0;
    }
    50%{
        transform: rotate3d(1,1,1,360deg);
        width:100%;height:100%;
        top:0;left:0;
    }
    100%{
        transform: rotate3d(1,1,1,-360deg);
        width:50px;height: 50px;
        top:450px;left:450px;
    }
}
.box3{
    animation: show3 6s forwards 1 ;
}
@keyframes show3{
    0%{
        top:0;left:0;
        width: 50px;heigth:50px;
    }
    50%{
        top:0;left:0;
        width:500px;height:50px;
    }
    100%{
        top:0;left:450px;
        width:50px;height:50px;
    }
}
.box4{
    animation:show4 2s  1 forwards;
    animation-timing-function: cubic-bezier(0.5,1,0.5,1);
}
@keyframes show4{
```

```
    0%{
        top:0;left: 0;
        transform: scale(1,1)
    }
    25%{
        top:0;left:450px;
        transform: scale(1,1);
    }
    50%{
        top:0;left:475px;
        transform: scale(0.5,1.1)
    }
    75%{
        top:-25px;left:425px;
        transform: scale(0.5,1);
    }
    100%{
        top: 0px;left:400px;
        transform: scale(1,1);
    }
}

.box5{
    animation: show5 0.8s 1 forwards ;
    animation-timing-function: ease-in-out;
}
@keyframes show5{
    0%{
        transform: rotate(15deg);
        top:0px;left:200px;
    }
    25%{
        transform: rotate(15deg);
        top:450px;left:200px;
    }
    50%{
        transform: rotate(0deg);
        top:430px;left:190px;
    }
    75%{
        transform:rotate(-7.5deg);
        top:450px;left:180px;
    }
    100%{
        transform: rotate(0deg);
        top:450px;left:180px;
    }
}
```

- html部分

```
<body>
    <div class="contain">
        <div class="box1"></div>
    </div>
    <div class="contain">
        <div class="box2"></div>
    </div>
    <div class="contain">
```

```
        <div class="box3"></div>
    </div>
    <div class="contain">
        <div class="box4"></div>
    </div>
    <div class="contain">
        <div class="box5"></div>
    </div>
    <div class="contain">
        <div class="box6"></div>
    </div>
    <div class="contain">
        <div class="box7"></div>
    </div>
</body>
```

javascript

1. 几个dom方法

```
createElement("tagName")
createTextNode("textValue")
parent_el.appendChild("child_el")
insertbefore()    //函数
```

2. 几个属性

- style属性获取样式,可读可写（！！！只能返回内嵌样式，被坑得好惨T^T）
- previousSibling,nextSibling,parentNode,firstChild,lastChild只可读
- className属性

```
el.className="value" //设置类名
```

3. 几个函数

- 显示“缩略语表列”（原来百度百科的表列可以这样做）

```
function displayAbbreviations(){
    if(!document.getElementsByTagName||!document.createElement||!document.createTextNode) return
false;
    var abbreviations=document.getElementsByTagName("abbr");
    if(abbreviations.length<1) return false;
    var defs=new Array();
    for(var i=0;i<abbreviations.length;i++){
        var current_abbr=abbreviations[i];
        var definition=current_abbr.getAttribute("title");
        var key=current_abbr.lastChild.nodeValue;
        defs[key]=definition;
    }
    var dlist=document.createElement("dl");
```

```
    for(key in defs){
        var definition=defs[key];
        var dtitle=document.createElement("dt");
        var dtitle_text=document.createTextNode("key");
        dtitle.appendChild(dtitle_text);
        var ddesc=document.createElement("dd");
        var ddesc_text=document.createTextNode("definition");
        ddesc.appendChild(ddesc_text);
        dlist.appendChild(dtitle);
        dlist.appendChild(ddesc);
    }
    if(dlist.childNodes.length<1) return false;
    var header=document.createElement("h2");
    var header_text=document.createTextNode("Abbreviations");
    header.appendChild(header_text);
    document.body.appendChild(header);
    document.body.appendChild(dlist);

    addLoadEvent(displayAbbreviations);
}
```

- 生成“文献来源链接”

```
function displayAccesskeys(){
    if(!document.getElementsByTagName||!document.createElement||!document.createTextNode) return
false;
    var links=document.getElementsByTagName("a");
    var akeys=new Array();
    for(var i=0;i<links.length;i++){
        var current_link=links[i];
        if(!current_link.getAttribute("accesskey")) continue;
        var key=current_link.getAttribute("accesskey");
        var text=current_link.lastChild.nodeValue;
        akeys[key]=text;
    }
    var list=document.createElement("ul");
    for(key in akeys){
        var text=akeys[key];
        var str=key+":"+text;
        var item=document.createElement("li");
        var item_text=document.createTextNode("str");
        item.appendChild(item_text);
        list.appendChild(item);
        var header=document.createElement("h3");
        var header_text=document.createTextNode("Accesskeys");
        header.appendChild(header_text);
        document.body.appendChild(header);
        docement.body.appendChild(list);
    }
    addLoadEvent(displayAccesskeys);
}
```

- 快捷键清单

```
function displayCitations(){
    if(!document.getElementsByTagName||!document.createElement||!document.createTextNode) return
false;
```

```
    var quotes=document.getElementsByTagName("blockquote");
    for(var i=0;i<quote.length;i++){
        if(!quotes[i].getAttribute("cite")) continue;
        var url=quotes.getAttribute("cite");
        var quoteChildren=quote[i].getElementsByTagName("*");
        if(quoteChildren.length<1) continue;
        var elem=quoteChildren[quoteChildren.length-1];
        var link=document.createElement("a");
        var link_text=document.createTextNode("source");
        link.appendChild(link_text);
        link.setAttribute("href",url);
        var superscript=document.createElement("sup");
        superscript.appendChild("link");
        elem.appendChild(superscript);
        }
    function addLoadEvent(displayCitations);
}
```

4. setTimeout函数（计时器）

```
setTimeout("func",time)
//第一个参数是执行的语句或函数，第二个参数是等待的时间
clearTimeout(timeoutID)
//消除计时
```

5. addEventListener方法

```
element.addEventListener("action",func,useCapture);
//三个参数，分别是行为，执行函数，事件处理方式
//useCapture是布尔值.true:捕获阶段处理事件 false:冒泡阶段处理事件
```

jQurey

1. 了解jQ语法和引用

```
$(selecter).action()
//选择器 行为or函数
//选择器和css的选择器相同
("*")             //所有元素
("TagNeme")       //标签选择器
("#IDname")       //id
(".classname")    //class
("TagName.first") //第一个该标签元素
("ul li.first-Child")//每个ul标签的第一个li元素
("[href]")        //带有 href 属性的元素
("a[target='_blank']")//所有 target 属性值等于 "_blank" 的 <a> 元素
("a[target!='_blank']")
(":button")       //所有 type="button" 的 <input> 元素 和 <button> 元素
("tr:even")       //偶数位置的 <tr> 元素
("tr:odd")        //奇数位置的 <tr> 元素
```

2. 事件

```
click,dblclick,mouseenter,mouseleave//鼠标事件
keypress,keydown,keyup//键盘事件
focus,blur,submit,change//表单事件
load,unload,resize,scroll//文档or窗口事件
```

3. 效果

```
hide,show,toggle//隐藏，显示，切换
fadeIn,fadeOut,fadeToggle,fadeTo//淡入，淡出，切换，透明度
slideDown,slideUp,slideToggle//滑动效果
element.animate({}); //自定义动画效果
```

问题和收获

- css布局

1. ◦ 当img和文字放在同个div中，文字会默认接在图片的右下方，有没有什么对齐方式可以让两者垂直居中对齐?
2. 雪碧图

- 导航栏小图标的hover效果用由一串图片的位置动画做出的。

1. ◦ js中想得到元素的属性值（不是样式），然而style只能获得内嵌样式，currentStyle和getComputedStyle在网上看还是不会用（尴尬）。
2. ◦ js怎么做点击展开导航栏，再次点击按钮或其他区域然导航栏收起?

- 用jQurey的slideToggle可以直接实现导航栏的划入划出效果，不过用原生js要怎么写？怎么区分第一次点击和第二次点击?