day14

四.转换

4.2D转换

①位移

|  |
| --- |
| transform:translate(x,y); |

②缩放

|  |
| --- |
| transform:scale(n)  取值 scale(n) 同时设置x轴和y轴的比例  scale(x,y) 分别设置x轴和y轴的比例  scaleX(x) 单独设置x轴的比例  scaleY(y) 单独设置y轴的比例  n>1 放大  n=1 不变  0<n<1 缩小  -1<n<0 缩小并反转  n<-1 放大并反转 |

③旋转

|  |
| --- |
| transform：rotate(ndeg)  n + 顺时针 - 逆时针  注意：  1.旋转的转换，会被转换原点影响效果  2.旋转是连同坐标轴一起旋转的，会影响旋转之后的位移方向 |

④倾斜

|  |
| --- |
| transform:skew(ndeg)  1.skew(ndeg)等同于skewX(xdeg)  让元素向着x轴发生倾斜，实际上改变的是y轴的角度  值 ndeg 为正 y轴逆时针  为负 y轴顺时针    2.skewY(ydeg)  让元素向着y轴发生倾斜，实际上改变的是x轴的角度  值 ndeg 为正 x轴顺时针  为负 x轴逆时针    3.skew(x,y) |

练习

|  |
| --- |
| 02\_ex  创建div 200px\*200px 背景随意，鼠标悬停时  该元素向右偏移200px,向下偏移200px,旋转135deg,向x轴倾斜45deg  ,并且放大1.5倍  面试  使用css，让一个div在任何情况都处于屏幕的中心 |

5.3D转换，3D转换都是模拟出来

①透视距离

|  |
| --- |
| 模拟人的眼睛到3D转换元素之间的距离  透视距离不同，看到的效果就不同  perspective:px 此属性要写在3D转换元素的父元素上 |

②3D旋转

|  |
| --- |
| transform  取值：  1.rotateX(ndeg)  以x轴为中心轴，旋转元素(烤羊腿，老式爆米花机)  2.rotateY(ndeg)  以y轴为中心轴，旋转元素(旋转门，旋转木马)  3.rotateZ(ndeg)  以z轴为中心轴，旋转元素(电风扇，摩天轮)  4.rotate(x,y,z,ndeg)  x,y,z取值为0，代表不参与旋转  x,y,z>0,代表参与旋转 |

五.过渡（重点\*\*\*\*\*\*\*\*\*\*）

|  |
| --- |
| 让css的值，在一段时间内平缓的变化 |

1.语法

①指定参与过渡的属性

|  |
| --- |
| transition-property:  取值 all 所有支持过渡属性都参与  单独写某个css属性，多个属性间用空格分开  支持过渡的属性：  1.颜色属性  2.大多数取值为具体数字的属性  3.阴影  4.转换  5.visibility |

②指定过渡时长

|  |
| --- |
| transition-duration: s/ms  指定多长时间完成此次过渡操作 |

③过渡时间曲线函数

|  |
| --- |
| transition-timing-function:  取值：  1.ease 默认值，慢速开始，中间变快，慢速结束  2.linear 匀速  3.ease-in 慢慢开始，一直加速  4.ease-out 快速开始，一直减速  5.ease-in-out 慢速开始，慢速结束，中间先加速再减速 |

④延迟执行

|  |
| --- |
| transition-delay:s/ms |

⑤过渡代码编写的位置

|  |
| --- |
| 写在原来的选择器中，过渡效果有去有回  写在hover中，过渡效果有去无回 |

⑥过渡的简写方式

|  |
| --- |
| transition：property duration timing-function delay;  最简方式：transition:duration; |

六.动画，多个过渡效果放到一起

1.关键帧

|  |
| --- |
| 关键帧  1.动画执行的时间点  2.在这个时间点上的样式 |

2.动画的使用步骤

①使用关键帧定义动画

|  |
| --- |
| @keyframes 动画名称{  0%{}  ...  100%{}  } |

②调用动画

|  |
| --- |
| 1.选中要执行动画的名称  animation-name  2.动画持续时长  animation-duration  3.动画的时间曲线函数  animation-timing-function:ease/linear/ease-in.......  4.延迟执行  animation-delay |

3.动画的其它属性

①动画的播放次数

|  |
| --- |
| animation-iteration-count:  取值 具体的播放次数  infinite; 无限 |

②动画的播放顺序

|  |
| --- |
| animation-direction:  取值 normal 默认值 0%~100%  reverse 100%~0%  alternate; 轮流播放，奇数次正向，偶数次逆向 |

③动画的简写方式

|  |
| --- |
| animation:name duration timing-function delay count direction  最简方式  animation:name duration; |

④动画在播放前后的显示状态

|  |
| --- |
| animation-fill-mode:  取值：1.backwards 动画播放之前的延迟时间内，显示动画的第一帧  2.forwards 动画播放结束之后，停留在最后一帧  3.both; 1和2合体  4.none 默认值，不填充状态 |

⑤动画的播放与暂停

|  |
| --- |
| animation-play-state:  取值 running  paused; |

4.动画的兼容性

|  |
| --- |
| 如果低版本浏览器，想使用动画，需要在动画声明的时候添加前缀  @keyframes 动画名称{}  @-webkit-keyframes 动画名称{}  @-o-keyframes 动画名称{}  @-ms-keyframes 动画名称{}  @-moz-keyframes 动画名称{} |

5. animate.css

|  |
| --- |
| 项目中，一般使用第三方动画库animate.css  1.下载animate.css  2.在页面中导入  3.调用动画名称 |

七.CSS的优化

|  |
| --- |
| css优化的目的  1.减少服务器的压力  2.提升用户体验 |

1.css优化的原则

|  |
| --- |
| 尽量减少http的请求个数  在页面的顶部，引入css文件  将css和js写在外部单独的文件中 |

2.css代码优化

|  |
| --- |
| 合并样式(能写群组，就不单写，能用简写，就不单独定义属性)  缩小样式文件的大小(能重用的尽量重用，减少样式重写)  避免出现空的href和src  代码压缩 |

八.css reset和normalize.css

1.什么css reset

|  |
| --- |
| 由于不同浏览器，对html元素的默认样式解析不同  我们在开发css之前，把大多数默认样式统一重置(格式化)  然后再写代码。这个行为叫做css reset |

2.normalize.css

|  |
| --- |
| 是css reset的一种优质的替代方案 |

3.css reset和normalize.css的区别

|  |
| --- |
| css reset 太激进，太暴力  normalize.css比较温和 |

**BootStrap**

一.响应式布局 (css3)

1.什么是响应式网页

|  |
| --- |
| Responsive web page 响应式网页  可以根据浏览器设备不同(pc,pad,phone)  而自动改变布局，图片，文字效果，不会影响用户体验 |

2.响应式页面必须做到下面几件事

|  |
| --- |
| 1.布局，不能固定元素宽度，必须使用流式布局(默认+浮动)+弹性布局  2.文字和图片大小随着容器大小变化  3.媒体查询技术 |

作业：

1.使用animate.css

2.大体阅读一下normalize.css

3.保证webstrom可以正常使用

4.完成第一个响应式网页，使用@media 媒体查询