四.转换

1.什么是转换

|  |
| --- |
| 改变元素在页面中 位置，大小，角度，形状  2D转换，只在x轴和y轴上发生的转换  3D转换，添加了z轴 |

2.转换属性

|  |
| --- |
| transform:转换函数1 转换函数2…..  取值 1.none 没有转换效果  2.transform-function 转换函数，一个或者多个转换函数  多个转换函数用空格隔开  转换原点,影响旋转的圆心  transform-origin: x y (z)  x(left/center/right) y(top/center/bottom) |

3.2D转换

①位移

|  |
| --- |
| transform:translate(x,y)  x轴位移 + → - ←  y轴位移 + ↓ - ↑  translate(x)等同于translateX(x)  只在x轴上位移  translateY(y) 只在y轴上位移 |

②缩放

|  |
| --- |
| transform:scale(v1) 同时设置x,y的缩放  v1>1 放大  0<v1<1 缩小  -1<v1<0 缩小并旋转  v1<-1 放大并旋转  scale(x,y) 分别设置x轴和y轴的缩放比例  scaleX(x) 单独设置x轴缩放  scaleY(y) 单独设置y轴缩放 |

③旋转

|  |
| --- |
| transform:rotate(ndeg)  n + 顺时针，- 逆时针  注意：1.转换原点会影响旋转效果  2.旋转会带着坐标轴一起旋转，所以旋转之后的位移，会受到影响。 |

④倾斜

|  |
| --- |
| skewX(ndeg)  让y轴向着x轴倾斜ndeg  n + 逆时针 - 顺时针 |
| skewY(ndeg)  让x轴向着y轴倾斜60deg  n + 顺时针 - 逆时针 |
| skew(ndeg)等同于skewX(ndeg)  skewY(ndeg)  skew(x,y) |

4.3D转换，所有3D都是模拟的

①透视距离

|  |
| --- |
| 模拟人的眼睛到3D转换元素之间的距离  距离不同，同一个元素转换，被看到的效果不同  perspective：距离px。  定义在3d转换元素的父元素上 |

②3D旋转

|  |
| --- |
| tranform:取值：  1.rotateX(ndeg)  以x轴为中心轴旋转 烤羊腿,老式爆米花机  2.rotateY(ndeg)  以y轴为中心轴旋转 旋转木马，土耳其烤肉  3.rotateZ(ndeg)  以z轴为中心轴旋转 风车 摩天轮  4.rotate3D(x,y,z,ndeg)  x,y,z取值>0 代表参与旋转  =0 代表不参与 |

五.过渡

|  |
| --- |
| 什么是过渡  让css的属性值，在一段时间内，平缓变化的效果 |

1.语法transition

|  |
| --- |
| ①transition-property: 指定过渡的属性  取值：1.具体的属性名，多个属性名之间用空格分开  2.all，所有支持过渡效果的属性，都参与过渡  注意：transform,不管是否指定，都参与过渡  支持过渡效果的属性：  1.颜色属性  2.大多数值为具体数字的属性  3.阴影  4.转换  5.visibility |
| ②指定过渡的时长  transition-duration: s/ms为单位的数字 |
| ③时间曲线函数  transition-timing-function:  ease 默认值，先加速再减速  linear 匀速  ease-in 一直加速  ease-out 一直减速  ease-in-out 慢速开始，先加速，再减速，慢速结束 |
| ④延迟执行过渡  transition-delay: s/ms为单位的数字 |
| ⑤过渡代码的编写位置  写在原来的选择器中，过渡效果，有去有回  写hover中，过渡效果，有去无回 |
| ⑥简写  transition:property duration timing-function delay;  最简方式  transition:duration; |

六.动画

|  |
| --- |
| 使用过渡场合  2个点之间的转换，使用hover激活  使用动画场合  多于2个点的转换，一般不适用hover |

动画是什么

|  |
| --- |
| 使元素从一种样式逐渐变为另外一种样式  就是将多个过渡效果放到一起 |

1.关键帧

|  |
| --- |
| 关键帧，控制动画的每一个状态  1.这一帧执行的时间点  2.在这个时间点上的样式 |

2.动画的使用步骤

①声明动画

|  |
| --- |
| @keyframes 动画名称{  0%{样式} //from{样式}  …  100%{样式} //to{样式}  } |

②使用动画

|  |
| --- |
| 指定使用的动画名称  animation-name:动画名称  设置动画执行时长  animation-duration: s/ms  设置动画的时间曲线函数  animation-timing-function  ease/linear/ease-in/ease-out/ease-in-out （steps）  设置动画的延迟  animation-delay |

③动画特有的属性

|  |
| --- |
| 1.动画的播放次数  animation-iteration-count:  具体次数  infinite （无限）  2.动画的播放方向  animation-direction:  normal 0~100  reverse 100~0  alternate 奇数次正向，偶数次反向  3.动画的简写方式  animation:name duration timing-function delay count direction;  最简方式animation:name duration;  4.动画播放前后的填充状态  animation-fill-mode:  backwards 延迟显示第一帧  forward 播放完毕之后显示最后一帧  both 上面两项都要  5.设置动画的暂停和播放  animation-play-state:  paused 暂停  running 播放 |

④动画的兼容性

|  |
| --- |
| 向让动画兼容低版本浏览器，需要在声明动画的时候写前缀  @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项目中使用动画最多的方式

|  |
| --- |
| animate.css |

七.css优化

1.css优化的目的

|  |
| --- |
| 1.提升用户体验  2.减少服务器端的压力 |

2.css优化原则

|  |
| --- |
| 尽量减少http请求个数 (精灵图，雪碧图)  页面的顶部引入css文件  将js和css放入独立的文件中 |

3.css代码优化

|  |
| --- |
| 合并样式(能写群组，就不单写选择器，能用简写，就不单独定义)  缩小样式文件的大小(压缩)  减少样式的重写，能重用就重用  避免出现空的href和src |

八.CSS reset 和 normalize.css

1.什么是css reset

|  |
| --- |
| html标签在浏览器中有默认样式  不同浏览器，不同版本，对这些默认样式解析不同  所以，在开发之前，我们需要把css的默认进行重置/清空  让各个浏览器对样式统一 |

2.css reset很松散，没有统一格式

|  |
| --- |
| 很霸道，把原生语义的功能都去掉了，比如说p  有些项目经理热衷于使用css reset  有些项目经理很抵触使用css reset |

3.normalize.css

|  |
| --- |
| boot twitter  相对柔和  normalize.css是css reset一种替代方案 |