十.定位

总结

|  |
| --- |
| 脱离文档流  1.该元素不占用页面空间  2.后续元素上前补位  3.脱离文档流的元素都变成块级  4.脱离文档流的元素，如果不设置宽度，被内容撑开 |

7.高度坍塌

十一.显示方式

1.显示方式

|  |
| --- |
| 块级，行内，行内块，table  绝对元素在网页中的表现形式  display: inline; 行内  block 块级  inline-block 行内块  table 让元素的显示方式变为table方式  none 脱离文档流的隐藏 |

2.显示效果

|  |
| --- |
| visibility:visible 显示，默认值  hidden 不脱离文档流隐藏 |

3.透明度

|  |
| --- |
| opacity: 0~1;  rgba只会改变当前颜色的透明度  opacity,元素内部所有与元素相关的，透明度都会跟着改变 |

4.垂直对齐

|  |
| --- |
| 使用场景  tr/td/th vertical-align:top/middle/bottom  img vertical-align:top/middle/bottom/baseline（默认）  给img设置垂直对齐，改变的是img前后文本的位置  一般，我们会把img的vertical-align，设置成非baseline |

5.光标的设置

|  |
| --- |
| cursor：default 默认箭头  pointer 小手  text 文本  wait 等待  crosshair 十字  help 帮助 |

6.列表的样式

|  |
| --- |
| 1.设置列表标识项  list-style-type:none;  disc  circle  square  2.把列表项，设置成图片  list-style-image:url(true.png);  3.列表项的位置  list-style-position：inside/outside（默认）  4.简写方式 list-style:type img position;  项目中最常出现的方式 list-style:none; 取点 |

十二.定位---相对，绝对，固定

|  |
| --- |
| position：  取值 static 默认缺省值，静态(默认文档流)  relative 相对定位  absolute 绝对定位  fixed 固定定位  注意：  1.当一个元素被position修饰，并且取值为relative/absolute/fixed其中一种时，这个元素被称为已定位元素  2.已定位元素，解锁4个偏移属性  top 距离上方有多远 + ↓，- ↑  left 距离左边有多远 + →，- ←  right  bottom  附加属性发生冲突，应用top和left |

1.相对定位

|  |
| --- |
| position：relative； 4个偏移属性  1.元素只添加position:relative，不写偏移量，对周围其他元素没有任何影响。(跟没写relative一样)  2.相对定位，没有脱离文档流  3.相对自己原来位置做偏移  使用场合：  1.做元素的微调(类似于margin的效果)  2.通常作为绝对定位元素的已定位祖先元素使用 |

2.绝对定位

|  |
| --- |
| position:absolute；配合偏移量使用  1.绝对定位的元素，脱离文档流  2.当祖先级元素，没有已定位元素，相对于body左上角偏移  绝对定位的元素会相对，离自己“最近的”“已定位的”“祖先”元素的左上角偏移  注意：relative通常作为绝对定位元素的已定位祖先元素使用 |

3.堆叠顺序

|  |
| --- |
| 如果不设置堆叠顺序，最后一个元素堆叠顺序最高(html的最后)  z-index: 无单位的整数  数字越大，堆叠顺序越高，理论上最大是21亿多  默认的堆叠顺序，不大于1  定位的脱离文档流，和浮动的脱离文档流不是一个体系  只有对已定位的元素，才能使用z-index  堆叠顺序对于父子无效，儿子永远站在爹肩膀上 |

4.固定定位

|  |
| --- |
| position：fixed;配合偏移量使用  永远相对body定位 |

**CSS3 core**

一.复杂选择器

1.兄弟选择器

|  |
| --- |
| 同级元素，叫兄弟  兄弟选择器只能往后找 |

①相邻选择器

|  |
| --- |
| 选择器+选择器{}  获取紧紧挨在某元素后面的兄弟元素 |

②通用兄弟选择器

|  |
| --- |
| 选择器~选择器{}  获取某元素后面所有符合要求的兄弟 |

使用场景

|  |
| --- |
| 所有排除老大意外，其他兄弟都要写的样式 |

2.属性选择器

|  |
| --- |
| [attr]{} 匹配有attr这个属性的元素  [attr1][attr2]{} 匹配有attr1和attr2属性的元素  elem[attr]{} 匹配有attr属性的elem元素  [attr=val]{} 匹配有attr属性并且值为val的元素  衍生出来的选择器有：  elem[attr1][attr2]{} elem[attr1=val1][attr2=val2]{}  十分精确找到某个或某些元素  模糊属性值  [attr^=val] 匹配属性值，以val开头  [attr$=val] 匹配属性值，以val结尾  [attr\*=val] 匹配属性值中有val  [attr~=val] 匹配属性值中有val这个独立的单词 |