day13

CSS core

一.复杂选择器

1.兄弟选择器

|  |
| --- |
| 拥有同一个父元素的子元素之间，称为兄弟  兄弟选择器只能往后找，不能往前找  （只能通过哥哥找弟弟，不能通过弟弟找哥哥） |

①相邻兄弟选择器

|  |
| --- |
| 选择器1 + 选择器2{样式声明}  匹配紧紧挨在选择器1后面，并且符合选择器2的元素  p+.c1{color:red} |

②通用兄弟选择器

|  |
| --- |
| 选择器1 ~ 选择器2{样式声明}  匹配选择器1，后面所有符合选择器2的元素  p~p{color:Red} |

③项目中的使用场景

|  |
| --- |
| 同一组元素，除了第一个元素以外，其它所有兄弟都要执行的样式  可以使用 相邻兄弟选择器，也可以通用兄弟选择器  但是，通用兄弟选择器的执行效率低 |

2.属性选择器（css中使用较少，js用的多）

|  |
| --- |
| ①[attr]{样式声明}  匹配具备attr属性的元素，预定义属性和自定义属性都可以匹配  ②[attr1][attr2]{样式声明}  匹配，同时具备attr1和attr2属性的元素  ③[attr1][attr2][attr3].......{样式声明}  ④elem[attr]{样式声明}  匹配具备attr属性的elem的元素  ⑤elem[attr1][attr2][attr3].......{样式声明}  ⑥[attr=value]{样式声明}  匹配具备attr属性，并且取值为value的元素  elem[attr1=value1] [attr2=value2]....{样式声明}  关于属性值的模糊查询  [attr^=value] 匹配具有attr属性，并且值以value开头的元素  [attr$=value] 匹配具有attr属性，并且值以value结尾的元素  [attr\*=value] 匹配具有attr属性，并且值中包含value的元素  [attr~=value] 匹配具有attr属性，并且值中包含value这个独立单词的元素  独立单词：value前后有空格 |

3.伪类选择器

|  |
| --- |
| 我们学过的伪类 love hate :focus，都只能用单冒号  css对于伪类选择器最近的一次升级，要求，所有的伪类必须使用双冒号开头 ::  在这个时间点之前，推出的伪类，用:  之后，推出的伪类，用::  浏览器为了兼容低版本，多数能用::的伪类，:单冒号都可以使用 |

①目标伪类

|  |
| --- |
| 让被锚点激活的元素，匹配样式  :target{} |

②结构选择器

|  |
| --- |
| 选择器:first-child{}  两个条件  匹配这个选择器父元素的第一个儿子，这个儿子还得符合这个选择器  选择器:last-child{}  两个条件  匹配这个选择器父元素的最后一个儿子，这个儿子还得符合这个选择器  选择器:nth-child(n){}  匹配这个选择器父元素的第n个儿子，这个儿子还得符合这个选择器  练习  02\_ex.html  一个4\*4个表格，宽高200\*200px  要求第一行背景#0ff;  最后一行背景#ff0  第三行第二列背景色#f0f |

③空标签选择器

|  |
| --- |
| 匹配没有子元素的元素--------绝对丁克  内部不能有子元素，不能有文本，回车，空格  :empty{} |

④唯一子元素选择器

|  |
| --- |
| 匹配属于其父元素的唯一子元素------独生子女  :only-child{} |

⑤否定伪类

|  |
| --- |
| :not(selector){样式声明}  符合selector的元素，都不要 |

练习

|  |
| --- |
| 要求除了第三行，其它行背景色都是#afa  要求除了第二列，其它列背景色都是#f00;  要求除了第三行的第二列，其它单元格背景色都是#999 |

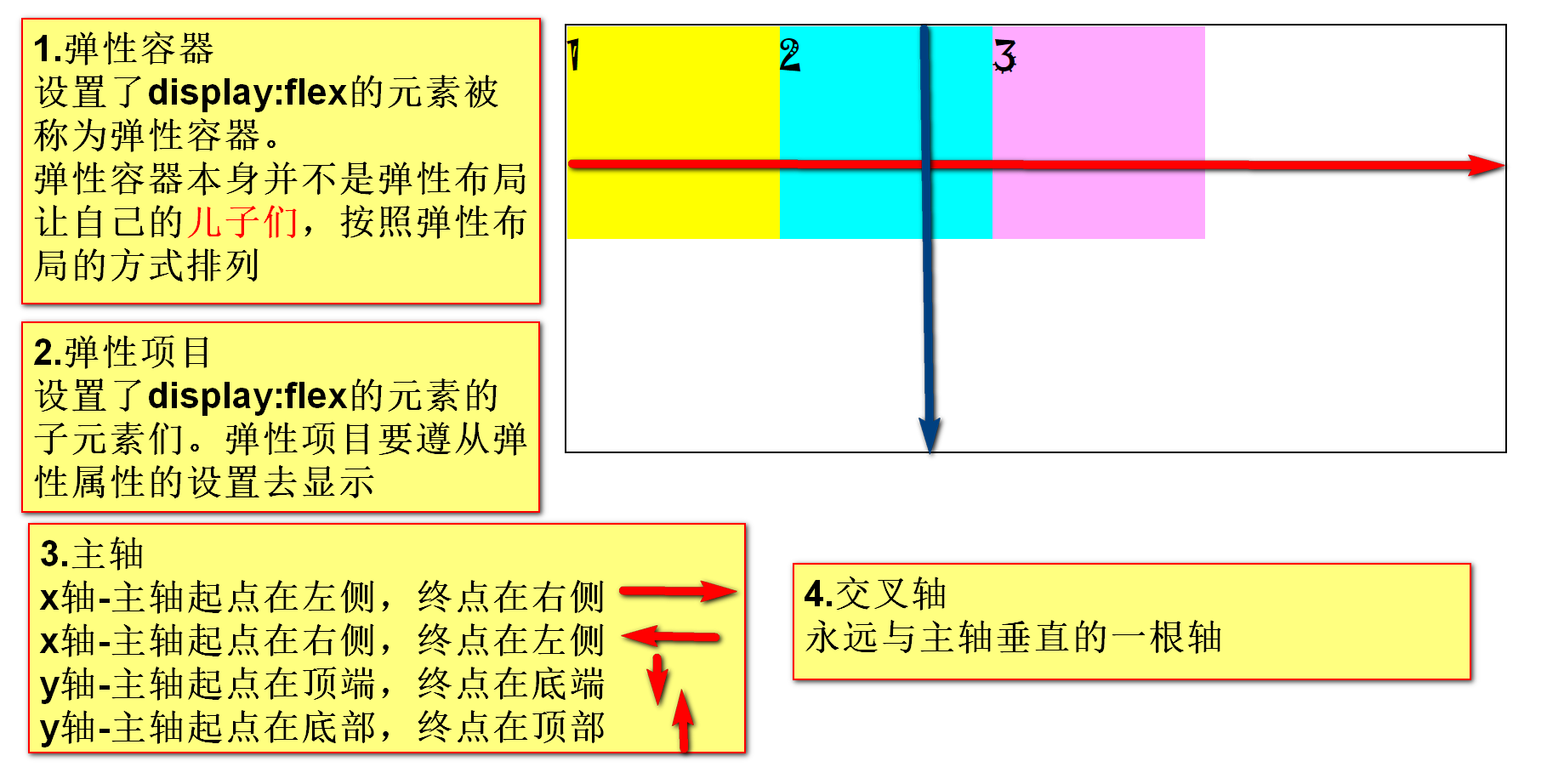
4.伪元素，内容生成

|  |
| --- |
| 使用css生成一个假的元素    #d1::before{content:"";}  #d1::after{content:"";} |
| ①使用伪元素内容生成，解决外边距溢出  #parent::before{  content: "";  display: table;  }  ②使用伪元素内容生成，解决高度坍塌  .clearfix::after {  display: block;  clear: both;  content: "";  } |

二.弹性布局

|  |
| --- |
| 是一种应用最广泛，最流行的布局方式 |

1.弹性布局的相关概念



3.语法

|  |
| --- |
| display:flex; 让块级和行内元素变为容器  inline-flex; 让行内元素变为容器  当外部容器为弹性容器，他的子元素们都是弹性项目  项目在容器中所有的对齐方式，都用主轴和交叉轴控制  容器的text-align失效  项目的float失效 |

4.容器的样式属性

|  |
| --- |
| 控制所有项目  ①主轴方向  flex-direction:  row 默认值，x轴，起点在左侧  row-reverse x轴，起点在右侧  column y轴，起点在顶部  column-reverse; y轴，起点在底部  ②项目在主轴方向上的换行  flex-wrap: nowrap; 默认值，主轴方向空间不足时，项目压缩，不换行  wrap; 主轴方向空间不足时,项目不压缩，自动换行  ③上面两个属性的简写  flex-flow:direction wrap;  ④设置项目在主轴上的对齐方式  justify-content:  flex-start 默认值，主轴起点对齐  center 主轴中间对齐  flex-end 主轴终点对齐  space-around 每个项目左右间距相同  space-between; 主轴两端对齐  ⑤设置项目在交叉轴上的对齐方式  align-items: flex-start 默认值，交叉轴起点对齐  center 交叉轴中间对齐  flex-end 交叉轴终点对齐  baseline 以项目中的文本基线对齐  stretch 如果项目在交叉轴方向上没有设置尺寸  让项目在交叉轴方向填满 |

5.项目的样式属性

|  |
| --- |
| 控制当前项目 |

①控制某一个项目交叉轴上的对齐方式

|  |
| --- |
| align-self: flex-start 默认值，交叉轴起点对齐  center 交叉轴中间对齐  flex-end 交叉轴终点对齐  baseline 以项目中的文本基线对齐  stretch 如果项目在交叉轴方向上没有设置尺寸  让项目在交叉轴方向填满  auto 使用父容器设置algin-items的值 |

作业：

1.使用弹性完成布局，使用定位完成图片



2.使用弹性布局完成学子底部



3.弹性所有属性的单词