day13

CSS core

一.复杂选择器

1.兄弟选择器

|  |
| --- |
| 一组元素，除了第一个以外，其它的设置统一样式 |

2.属性选择器(不是为写页面设计的，为js的dom准备)

|  |
| --- |
| [attr]{样式声明} 匹配具有attr这个属性的元素  elem[attr]{} 匹配具有attr属性的elem元素  [attr=value]{} 匹配具有attr属性并且值为value的元素  elem[attr=value]{} 匹配具有attr属性并且值为value的elem元素  elem[attr1=value1][attr2=value2].....{}  后期dom操作，使用自定义属性，查询目标对象  模糊属性值的匹配  [title^="gan"]{color:Red;} 匹配元素title属性值以gan开头  [title$="da"]{color:Red;} 匹配元素title属性值以da结尾  [title\*="meng"]{color:Red;} 匹配元素title属性值含有meng  [title~="meng"]{color:Red;} 匹配元素title属性值含有meng这个独立单词(meng前后有空格) |

3.伪类选择器

|  |
| --- |
| 已经学过的伪类 :link :visited :hover :active :focus |

①目标伪类（h5新出的）

|  |
| --- |
| 匹配元素被锚点激活的状态  selector:target{} |

②结构伪类

|  |
| --- |
| 通过html结构来匹配元素  selector：first-child{} 匹配selector父元素的第一个儿子，同时这个大儿子还必须符合selector  selector：last-child{} 匹配selector父元素的最后一个儿子，同时这个小儿子还必须符合selector  select:nth-child(n){} 匹配selector父元素的第n个儿子，同时这个儿子还必须符合selector |

练习

|  |
| --- |
| 创建4\*4的表格，200px\*200px  第一行背景色为#0ff  最后一行背景色为#ff0  第三行第二列背景色为#f00 |

③绝对丁克

|  |
| --- |
| :empty{}内部不能有其它元素，不能有文本，空格，回车 |

④独生子女

|  |
| --- |
| :only-child{} 匹配是其父元素唯一的子元素 |

⑤否定伪类

|  |
| --- |
| :not(selector){} 所有符合selector的元素都不要 |

4.伪元素选择器

|  |
| --- |
| 匹配某个元素的一部分  w3c最新的规定，推荐所有的伪类使用双::编写  但是之前已经出现的伪类，还使用单: |

①匹配首字符

|  |
| --- |
| :first-letter 或者 ::first-letter |

②匹配首行

|  |
| --- |
| :first-line 或者 ::first-line  注意，首字符和首行冲突，以首字符为准 |

③::selection

|  |
| --- |
| 只能写双::，匹配用户选中的文本  只能定义字体颜色和背景颜色  对首字符无效 |

5.内容生成

|  |
| --- |
| 使用css在元素的内容区域最前端或者最后端，添加一个假的元素  ::before 或者 :before  在元素的内容区域最前端添加一个假的行内元素  ::after 或者 :after  在元素的内容区域最后端添加一个假的行内元素  行内元素的内容使用content设置，但是只能设置文本和url(img)  我们可以自由的对这个假元素设置各种样式 |
| 内容生成可以解决的问题  1.外边距溢出，解决方案----在父级的内部添加一个空的table当做大儿子  #d2::before{  content:"";  display: table;  } |
| 2.解决高度坍塌，解决方案----在父级的内部追加一个空的div当做小儿子，这个div设置clear:both  #parent::after{  content:"";  display: block;  clear:both;  } |

二.弹性布局

1.什么是弹性布局

|  |
| --- |
| 主要解决某个元素中内部子元素的布局方式，为布局提供的非常大的灵活性和操控性  设置了display：flex的元素本身不是弹性布局  它的儿子们是弹性布局 |

2.弹性布局的专业术语

|  |
| --- |
| 1.弹性容器  当一个元素设置了display:flex  这个元素内部的子元素就按照弹性布局方式排列  这个元素叫做容器  2.弹性项目  当一个元素设置了display:flex  这个元素内部的子元素就按照弹性布局方式排列  这些子元素，叫做弹性项目  3.主轴  控制项目们的排列方向的一根轴  主轴的方向，  被称为主轴的起点和终点  4.交叉轴  永远与主轴垂直相交的一根轴 |

3.语法

|  |
| --- |
| display:  flex 经常用于让块级元素变为弹性容器  inline-flex; 经常用于让行内元素变为弹性容器  特点：  1.当元素变为弹性容器之后，这个元素的text-align失效，内部项目不会执行水平对齐  2.弹性项目，可以设置宽高  3.项目的浮动，clear：both都失效  总结：之前所有学习过的对齐方式，对于弹性项目都失效  弹性项目的对齐，取决于主轴、交叉轴 |

4.样式属性

①容器的样式

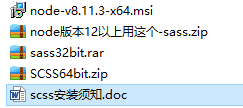
|  |
| --- |
| 所有项目都要用  1.主轴方向（4根主轴）  flex-direction:row; 默认值，主轴为x轴，主轴起点在左侧  row-reverse 主轴为x，主轴起点在右侧  column 主轴为y轴，主轴起点在顶部  column-reverse 主轴为y轴，主轴起点在底部 |
| 2.项目换行  flex-wrap:nowrap;默认值，主轴方向空间不够也不换行  wrap 主轴方向空间不够时，自动换行  wrap-reverse; 交叉轴终点对齐，主轴方向空间不够时，反方向换行 |
| 3.上面两个属性的缩写方式  flex-flow:direction wrap; |
| 4.定义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; 主轴的中心对齐  flex-end 主轴的终点对齐  space-around 每个项目的间隙相同  space-between 两端对齐 |
| 5.交叉轴上对齐方式 （2根）  align-items  flex-start 默认值。在交叉轴起点对齐  center 交叉轴中间对齐  flex-end 交叉轴终点对齐  baseline 每个项目中的文本，基线要对齐  stretch 项目在交叉轴方向上没有定义尺寸，项目将充满交叉轴 |

②项目的样式

|  |
| --- |
| 某一个项目使用  1.给某个项目单独设置交叉轴上的对齐方式  align-self  flex-start 默认值。在交叉轴起点对齐  center 交叉轴中间对齐  flex-end 交叉轴终点对齐  baseline 每个项目中的文本，基线要对齐  stretch 项目在交叉轴方向上没有定义尺寸，项目将充满交叉轴  auto 使用容器定义的align-items的值 |
| 2.order  取值为无单位整数  定义项目的排列顺序，值越小，越靠近主轴的排列方向，默认值为0 |
| 3.flex-grow  当主轴有多余的区域，项目是否放大，放大比例是多少  默认值为0，不放大  把多余的区域，按照比例分配，加上原始的尺寸，就是每个项目的实际尺寸 |
| 4.flex-shrink  当主轴空间不足时，项目是否缩小，怎么缩小  默认值1，不设置换行的情况下，会自动缩小  主轴不足的部分，按照比例，分配给各个项目去缩减 |
| 5.flex-basis  设置某个项目在主轴上占据的基本尺寸，通常取值为%  flex-basis优先级大于项目自己定义的尺寸 |
| 6.flex-grow flex-shrink flex-basis的缩写  flex:0 0 20%; |

作业1

完成scss的安装



作业2:



作业3:

