flex布局-阮一峰

网页布局

1. 传统布局基于盒状模型-依赖dispaly属性+position属性+float属性。
2. 特殊布局非常不便，比如垂直居中

盒状模型

3. 当对一个文档进行布局时，浏览器的渲染引擎会根据标准之一的 CSS 基础框盒模型，将所有元素表示为一个个矩形盒子。
4. content-box和border-box 的区别(叠加和递减width和height)

1.
```
div {
width: 160px;
height: 80px;
padding: 20px;
border: 8px solid red;
background: yellow;
}
.content-box {
box-sizing: content-box;
/* Total width: 160px + (2 * 20px) + (2 * 8px) = 216px
   Total height: 80px + (2 * 20px) + (2 * 8px) = 136px
   Content box width: 160px
   Content box height: 80px */
}

.border-box {
box-sizing: border-box;
/* Total width: 160px
   Total height: 80px
   Content box width: 160px - (2 * 20px) - (2 * 8px) = 104px
   Content box height: 80px - (2 * 20px) - (2 * 8px) = 24px */
}
```

5. 每个盒子的四个部分（区域）：
    1. content edge：如果 box-sizing 为 content-box（默认），则内容区域的大小可明确地通过 width、min-width、max-width、height、min-height 和 max-height 控制。
    2. padding area（内边距区域）：内边距的粗细可以由 padding-top、padding-right、padding-bottom、padding-left，和简写属性 padding 控制。
    3. border area（边框区域）：
        1. 边框的粗细由 border-width 和简写的 border 属性控制。
        2. 如果 box-sizing 属性被设为 border-box，那么边框区域的大小可明确地通过 width、min-width, max-width、height、min-height，和 max-height 属性控制。
        3. 假如框盒上设有背景（background-color 或 background-image），背景将会一直延伸至边框的外沿（默认为在边框下层延伸，边框会盖在背景上）。此默认表现可通过 CSS 属性 background-clip 来改变。

4. margin area（外边距区域）：
    1. 外边距区域的大小由 margin-top、margin-right、margin-bottom、margin-left，和简写属性 margin 控制。
    2. 在发生外边距合并的情况下，由于盒之间共享外边距，外边距不容易弄清楚。

display属性

position属性

1. position: static;
    1. HTML 元素的默认值，即没有定位，遵循正常的文档流对象。
    2. 静态定位的元素不会受到 top, bottom, left, right影响。
2. position: relative;
    1. 相对定位，相对于自身原有的位置进行改变
3. position: absolute;
    1. 绝对定位，对其父元素进行位置变动，如果该元素没有父元素，则已整个html为父元素
    2. 绝对定位的元素可以设置外边距（margins），且不会与其他边距合并。
4. position: sticky;
    1. 黏性定位，基于用户的滚动位置进行定位。
    2. 依赖用户滚动，在 position:relative 与 position:fixed 定位之间切换。
5. fixed 定位
    1. 元素的位置相对于浏览器窗口是固定位置。即使窗口是滚动的它也不会移动：

flex

容器属性

1. ```
   .box{
      display: -webkit-flex; /* Safari */
      display: flex;/* Webkit内核的浏览器必须加上-webkit前缀*/
      }
   ```

2. 设为 Flex 布局以后，子元素的
    1. float、
    2. clear（不允许浮动哪侧浮动元素）
    3. vertical-align(垂直对齐)属性将失效。
3. flex-direction属性决定主轴的方向（即项目的排列方向）。
    1. row（默认值）：主轴为水平方向，起点在左端。
    2. row-reverse：主轴为水平方向，起点在右端。
    3. column：主轴为垂直方向，起点在上沿。
    4. column-reverse：主轴为垂直方向，起点在下沿。
4. flex-wrap属性定义，如果一条轴线排不下，如何换行。默认情况下，项目都排在一条线（又称"轴线"）上。
5. flex-flow属性是flex-direction属性和flex-wrap属性的简写形式，默认值为row nowrap
6. justify-content属性定义了项目在主轴上（main axis）的对齐方式。
    1. flex-start（默认值）：左对齐
   ```

2. flex-end：右对齐
3. center： 居中
4. space-between：两端对齐，项目之间的间隔都相等。
5. space-around：每个项目两侧的间隔相等。所以，项目之间的间隔比项目与边框的间隔大一倍。

7. align-items属性定义项目在交叉轴上（cross axis）如何对齐。
    1. flex-start：交叉轴的起点对齐。
    2. flex-end：交叉轴的终点对齐。
    3. center：交叉轴的中点对齐。
    4. baseline: 项目的第一行文字的基线对齐。
    5. stretch（默认值）：如果项目未设置高度或设为auto，将占满整个容器的高度。
8. align-content属性定义了多根轴线的对齐方式。如果项目只有一根轴线，该属性不起作用。
    1. flex-start：与交叉轴的起点对齐。
    2. flex-end：与交叉轴的终点对齐。
    3. center：与交叉轴的中点对齐。
    4. space-between：与交叉轴两端对齐，轴线之间的间隔平均分布。
    5. space-around：每根轴线两侧的间隔都相等。所以，轴线之间的间隔比轴线与边框的间隔大一倍。
    6. stretch（默认值）：轴线占满整个交叉轴。

项目属性

1. 以下6个属性设置在项目上。
    1. order: 属性定义项目的排列顺序。数值越小，排列越靠前，默认为0。
    2. flex-grow: flex-grow属性定义项目的放大比例，默认为0，即如果存在剩余空间，也不放大。
        1. 如果所有项目的flex-grow属性都为1，则它们将等分剩余空间（如果有的话）。
        2. 如果一个项目的flex-grow属性为2，其他项目都为1，则前者占据的剩余空间将比其他项多一倍。
    3. flex-shrink: flex-shrink属性定义了项目的缩小比例，默认为1，即如果空间不足，该项目将缩小。
        1. 如果所有项目的flex-shrink属性都为1，当空间不足时，都将等比例缩小。
        2. 如果一个项目的flex-shrink属性为0，其他项目都为1，则空间不足时，前者不缩小。
    4. flex-basis: flex-basis属性定义了在分配多余空间之前，项目占据的主轴空间（main size）。
        1. 浏览器根据这个属性，计算主轴是否有多余空间。
        2. 它的默认值为auto，即项目的本来大小。
        3. 它可以设为跟width或height属性一样的值（比如350px），则项目将占据固定空间。
    5. flex: flex属性是flex-grow, flex-shrink 和 flex-basis的简写，默认值为0 1 auto。
        1. 后两个属性可选。
        2. 该属性有两个快捷值：auto (1 1 auto) 和 none (0 0 auto)。
        3. 建议优先使用这个属性，而不是单独写三个分离的属性，因为浏览器会推算相关值。
    6. align-self: align-self属性允许单个项目有与其他项目不一样的对齐方式，可覆盖align-items属性。
        1. 默认值为auto，表示继承父元素的align-items属性，如果没有父元素，则等同于stretch。
        2. 该属性可能取6个值，除了auto，其他都与align-items属性完全一致。