河南中医药大学信息技术学院（智能医疗行业学院）智能医学工程专业《互联网医疗服务开发》课程

第04章：布局

冯顺磊

河南中医药大学信息技术学院（智能医疗行业学院）
河南中医药大学信息技术学院智能医疗教研室
https://aitcm.hactcm.edu.cn
2025/2/27

1

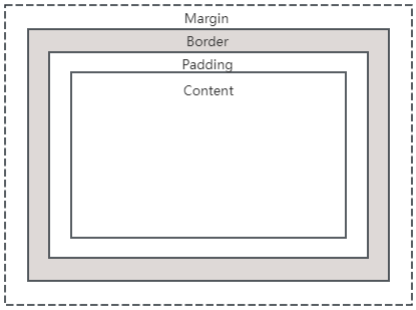
本章概要

- 盒模型
- 定位
- 布局
- 响应式

2

1. 盒模型

- CSS盒模型是CSS布局中的一个核心概念，描述了如何对文档树中的元素进行布局和渲染。每个元素都会生成一个矩形盒子，盒子的组成部分包括内容（content）、内边距（padding）、边框（border）和外边距（margin）。
  - 内容（Content）：元素的实际内容区域，包括文本或其他媒体内容，尺寸可以通过“width”和“height”属性来设置。
  - 内边距（Padding）：内边距位于内容和边框之间，是元素内部的空间。
  - 边框（Border）：边框环绕在内边距和内容周围，是可见的，边框可以指定样式（如虚线、实线等）、宽度和颜色。
  - 外边距（Margin）：外边距是元素边框之外的空间，用于分隔元素，是完全透明的，外边距可以设置为负值，会导致元素重叠。



1. 盒模型

- margin属性
  - margin 属性用于在元素的内容区域（包括内边距和边框）之外创建额外的空间，从而控制元素与其他元素之间的间距。它可以接受一个、两个、三个或四个值，不同数量的值有不同的含义。
  - 长度值：可以使用具体的长度单位（如 px、em、rem 等）来指定外边距的大小。
  - 百分比值：使用百分比指定外边距时，该百分比是相对于元素包含块的宽度计算的。
  - auto：当取值为 auto 时，浏览器会自动计算外边距。
  - 不同取值的含义：
    - 一个值：当 margin 只指定一个值时，该值会应用到元素的上、右、下、左四个边
    - 两个值：当指定两个值时，第一个值应用于元素的上和下外边距，第二个值应用于元素的左和右外边距。
    - 三个值：当指定三个值时，第一个值应用于元素的上外边距，第二个值应用于元素的左和右外边距，第三个值应用于元素的下外边距。
  - 除了使用 margin 简写属性，还可以使用 margin-top、margin-right、margin-bottom 和 margin-left 分别设置元素某一边的外边距。

1. 盒模型

- margin属性
  - 外边距合并：当两个垂直相邻的块级元素的外边距相遇时，它们的外边距会合并成一个外边距，其大小取两个外边距中的较大值。

```
p {
    margin-bottom: 20px;
}

h2 {
    margin-top: 30px;
}
```

  - 负外边距：margin 属性可以取负值，使用负外边距可以使元素与其他元素重叠或改变元素的位置。
  - 内联元素的外边距：内联元素（如 、<a> 等）的 margin 属性在水平方向上有效，但在垂直方向上不会影响元素的布局。如果需要在垂直方向上设置间距，通常需要将内联元素转换为块级元素或行内块元素。

示例4-1-1：margin属性

1. 盒模型

- padding属性
  - padding 定义了元素内容区域与边框之间的距离，它可以让元素的内容不会紧贴着边框，从而提升页面的美观度和可读性。与 margin 不同，margin 控制的是元素与其他元素之间的间距，而 padding 控制的是元素内部内容与边框的间距。
  - 长度值：可以使用具体的长度单位（如 px、em、rem 等）来指定外边距的大小。
  - 百分比值：使用百分比指定外边距时，该百分比是相对于元素包含块的宽度计算的。
  - 不同取值的含义：
    - 一个值：当 padding只指定一个值时，该值会应用到元素的上、右、下、左四个边
    - 两个值：当指定两个值时，第一个值应用于元素的上和下外边距，第二个值应用于元素的左和右外边距。
    - 三个值：当指定三个值时，第一个值应用于元素的上外边距，第二个值应用于元素的左和右外边距，第三个值应用于元素的下外边距。
  - 除了使用 margin 简写属性，还可以使用 padding-top、padding-right、padding-bottom 和 padding-left 分别设置元素某一边的外边距。

1. 盒模型

- padding属性
  - 盒模型影响：padding 会影响元素的实际尺寸。在使用固定宽度和高度的元素时，设置 padding 会使元素的整体尺寸增大。不过可以通过 box-sizing 属性来控制盒模型的计算方式，当 box-sizing: border-box 时，padding 和 border 会包含在元素设定的宽度和高度内。
  - 背景显示：元素的背景会延伸到 padding 区域。这意味着如果设置了元素的背景颜色或背景图像，它们会填充到内容和 padding 所占据的区域。
  - 内联元素的 padding：内联元素（如 、<a> 等）的 padding 在水平方向上会影响布局，但在垂直方向上不会影响行高。若要让垂直方向的 padding 影响布局，通常需要将内联元素转换为块级元素或行内块元素。

示例4-1-2：margin属性

1. 盒模型

- display属性
  - display 是 CSS 中一个非常重要的属性，它用于控制元素在页面中的显示方式，决定了元素如何布局以及与其他元素的交互方式。
  - block：将元素显示为块级元素。块级元素会独占一行，并且可以设置宽度和高度。如果不设置宽度，它会默认填满其父元素的宽度。
  - inline：将元素显示为内联元素。内联元素不会独占一行，会在一行内依次排列，并且宽度和高度由其内容决定，不能直接设置宽度和高度。
  - inline-block：元素会像内联元素一样在一行内排列，但又可以像块级元素一样设置宽度和高度。它结合了内联元素和块级元素的特点。
  - none：元素将不会在页面中显示，并且不会占据页面的空间，就好像该元素不存在一样。与 visibility: hidden 不同，visibility: hidden 只是让元素不可见，但仍然会占据页面空间。
  - flex：将元素显示为弹性容器，其直接子元素成为弹性项目。Flexbox 布局是一种一维布局模型，用于为盒状模型提供最大的灵活性，可以方便地控制子元素的排列、对齐和分布。
  - grid：将元素显示为网格容器，其直接子元素成为网格项目。Grid 布局是一种二维布局模型，用于创建二维网格结构，可以更方便地进行复杂的页面布局。

1. 盒模型

- display属性
  - 除了上述常见的值外，display 还有很多其他属性值，如 table、table-row、table-cell 等，用于模拟表格布局；list-item 用于将元素显示为列表项；run-in 等一些相对不那么常用的值。不同的值适用于不同的布局需求，可根据具体情况选择合适的 display 属性值来实现所需的页面布局。
  - 浏览器兼容性：虽然大多数现代浏览器都支持上述常见的 display 属性值，但对于一些较新的布局方式（如 flex 和 grid），在旧版本的浏览器中可能需要添加浏览器前缀或使用其他替代方案来确保兼容性。
  - 布局影响：display 属性的改变会影响元素的布局和与其他元素的交互方式，在修改 display 属性时，需要考虑对整个页面布局的影响。

示例4-1-3：display属性

1. 盒模型

- visibility属性
  - visibility 属性用于控制元素在页面中的可见性，它决定了元素是否显示在页面上，但与 display: none 不同，它不会改变页面的布局结构。
  - visible：元素正常显示，这是 visibility 属性的默认值。元素会按照其自身的样式和布局规则在页面上呈现。
  - hidden：元素会隐藏，但仍然会占据页面上的空间。也就是说，虽然元素不可见，但它在布局中的位置和大小仍然会被保留，不会影响其他元素的布局。
  - collapse：该值主要用于表格元素（如 <tr>、<td> 等）。当应用于表格行或列时，它会隐藏该行或列，并且不占用任何空间，就好像这些行或列不存在一样。对于非表格元素，collapse 的效果和 hidden 相同。
  - 动画效果：visibility 属性可以用于创建动画效果，例如通过 CSS 过渡（transition）或动画（animation）让元素在可见和隐藏状态之间平滑切换。
  - 浏览器兼容性：大多数现代浏览器都支持 visibility 属性及其常见值，但在使用 collapse 时，需要注意不同浏览器可能存在一些细微的兼容性差异，特别是在处理复杂表格布局时。

示例4-1-4：display属性

2. 定位

- 定位用于精确的控制元素在页面中的位置。
- position属性
  - position 用于控制元素在页面中的定位方式。通过设置 position 属性的值，可以改变元素的正常文档流行为，实现不同的布局效果。
  - static（静态定位）：static 是 position 属性的默认值。使用 static 定位的元素会按照正常的文档流进行布局，即元素会按照在 HTML 文档中出现的顺序依次排列。top、right、bottom、left 和 z-index 属性对静态定位的元素没有影响。
  - relative（相对定位）：元素首先会按照正常的文档流进行布局，然后相对于其正常位置进行偏移。使用 top、right、bottom 和 left 属性可以指定偏移量。相对定位的元素仍然会占据其在正常文档流中的空间。
  - absolute（绝对定位）：元素会脱离正常的文档流，不再占据原来的空间。它会相对于最近的已定位祖先元素（即 position 值不为 static 的祖先元素）进行定位。如果没有已定位的祖先元素，则相对于 <html> 元素进行定位。
  - fixed（固定定位）：元素会脱离正常的文档流，并且相对于浏览器窗口进行定位。无论页面如何滚动，固定定位的元素都会保持在浏览器窗口的固定位置。
  - sticky（粘性定位）：粘性定位是相对定位和固定定位的混合。元素在滚动到特定位置之前，会按照正常的文档流进行布局；当滚动到特定位置时，元素会固定在该位置，就像固定定位一样。sticky 定位的元素仍然会占据其在正常文档流中的空间。

2. 定位

- position属性
  - z-index 属性：当元素使用定位时，可能会出现元素重叠的情况。z-index 属性可以控制元素在垂直方向上的堆叠顺序，值越大的元素会显示在越上面。
  - 浏览器兼容性：大多数现代浏览器都支持 position 属性的各种取值，但在一些旧版本的浏览器中，sticky 定位可能需要添加浏览器前缀才能正常工作。

示例4-2-1：position属性

2. 定位

- float属性
  - float是 CSS 中用于控制元素浮动的属性，通过设置该属性可以改变元素的正常文档流行为，使元素向左或向右浮动，其他元素会围绕在其周围。
  - left：元素会向左浮动，直到它的左边缘碰到包含块的左边缘或者另一个浮动元素的右边缘。
  - right：元素会向右浮动，直到它的右边缘碰到包含块的右边缘或者另一个浮动元素的左边缘。
  - none：元素不会浮动，会按照正常的文档流进行布局，这是 float 属性的默认值。

- 高度塌陷问题：当一个父元素包含的子元素都设置了浮动时，父元素会失去高度，这就是高度塌陷问题。因为浮动元素脱离了正常的文档流，父元素不再包含这些浮动子元素，所以无法计算它们的高度。
  - 使用 clearfix 类：通过创建一个伪元素来清除浮动，使父元素包含浮动子元素。
  - 使用 overflow 属性：将父元素的 overflow 属性设置为 auto 或 hidden，可以触发 BFC（块级格式化上下文），从而包含浮动子元素。

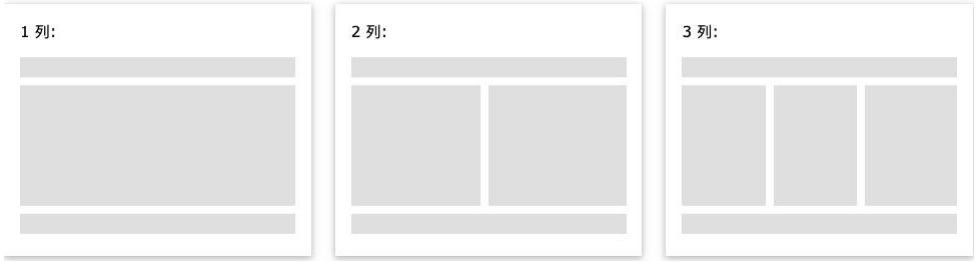
示例4-1-5：position属性

3. 布局

- 网页布局

3. 布局

- 内容布局



3. 布局

- 正常流布局
  - 元素默认的布局方式，元素会按照在 HTML 文档中出现的顺序依次排列。块级元素（如 、<p>等）会独占一行，从上到下垂直排列；行内元素（如、<a>等）会在一行内从左到右水平排列。
  - 适用场景：适用于简单的页面结构，元素按照默认顺序排列即可满足需求，如简单的文本页面。

示例4-3-1：正常流布局

3. 布局

- 浮动布局
  - 通过float属性让元素脱离正常流，向左或向右浮动，其他元素会围绕在其周围。常用的float属性值有left、right和none（默认值）。
  - 适用场景：常用于实现多列布局，如新闻网站的文章列表、图片画廊等。

示例4-3-2：浮动布局

3. 布局

- 定位布局
  - 通过position属性来控制元素的定位方式，常见的position属性值有static（默认值，元素按照正常流布局）、relative（相对定位，元素相对于其正常位置进行偏移）、absolute（绝对定位，元素相对于最近的已定位祖先元素进行定位）、fixed（固定定位，元素相对于浏览器窗口进行定位，滚动页面时位置不变）和sticky（粘性定位，元素在滚动到特定位置时固定）。
  - 适用场景：适用于创建弹出框、侧边栏菜单、固定导航栏等需要精确控制元素位置的场景。

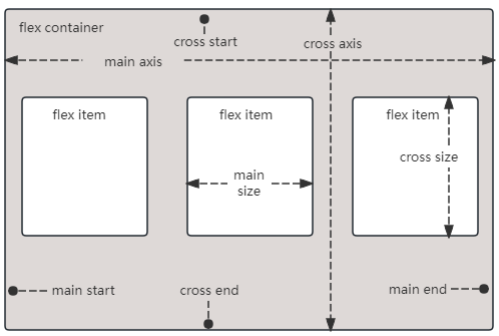
示例4-3-3：定位布局

3. 布局

- 弹性盒子布局
  - 一种一维布局模型，用于为盒状模型提供最大的灵活性。通过display: flex或display: inline-flex将元素设置为弹性容器，其直接子元素成为弹性项目。可以使用justify-content、align-items等属性控制弹性项目在主轴和交叉轴上的对齐方式。
  - 适用场景：适用于一维布局，如导航菜单、表单元素排列、卡片布局等。

3. 布局

- 弹性盒子布局
  - 设置了“display：flex;”的父元素被称之为弹性容器（flex container），在flex容器中表现为弹性的盒子的元素被称之为弹性项目（flex item）
  - 当元素表现为flex框时，元素沿着两个轴来布局。
    - 主轴（main axis）是沿着flex元素放置的方向延伸的轴（例如页面上横向的行、纵向的列），主轴的开始和结束被称为“main start”和“main end”。
    - 交叉轴（cross axis）是垂直于flex元素放置方向的轴，交叉轴的开始和结束被称为“cross start”和“cross end”。



3. 布局

- 弹性盒子布局
  - 弹性容器属性
    - display
      - 属性值：flex或inline-flex。
      - 作用：用于将一个元素定义为弹性容器。flex使元素作为块级弹性容器，inline-flex则使元素作为行内弹性容器。
    - flex-direction
      - 属性值：row（默认值）、row-reverse、column、column-reverse。
      - 作用：决定弹性项目在弹性容器中的排列方向。row表示水平从左到右排列；row-reverse是水平从右到左排列；column表示垂直从上到下排列；column-reverse是垂直从下到上排列。
    - flex-wrap
      - 属性值：nowrap（默认值）、wrap、wrap-reverse。
      - 作用：控制弹性项目是否换行。nowrap表示不换行，所有弹性项目都在一行或一列中显示；wrap表示在容器空间不足时换行，新行或新列在下方或右方；wrap-reverse也是换行，但新行或新列在上方或左方。

3. 布局

- 弹性盒子布局
  - 弹性容器属性
    - justify-content
      - 属性值：flex-start（默认值）、flex-end、center、space-between、space-around、space-evenly。
      - 作用：定义弹性项目在主轴上的对齐方式。flex-start使项目在主轴起始位置对齐；flex-end在主轴结束位置对齐；center在主轴中间对齐；space-between使项目两端对齐，项目之间间隔相等；space-around使项目周围的间隔相等；space-evenly使项目之间以及项目与容器边缘的间隔都相等。
    - align-items
      - 属性值：flex-start、flex-end、center、baseline、stretch（默认值）。
      - 作用：定义弹性项目在交叉轴上的对齐方式。flex-start使项目在交叉轴起始位置对齐；flex-end在交叉轴结束位置对齐；center在交叉轴中间对齐；baseline使项目基于文本基线对齐；stretch会拉伸项目以填充交叉轴空间。
    - align-content
      - 属性值：flex-start、flex-end、center、space-between、space-around、stretch（默认值）。
      - 作用：在弹性容器内有多行或多列弹性项目时，用于控制行与行或列与列之间在交叉轴上的对齐方式和分布。flex-start使行或列在交叉轴起始位置对齐；flex-end在交叉轴结束位置对齐；center在交叉轴中间对齐；space-between使行或列两端对齐，行或列之间间隔相等；space-around使行或列周围的间隔相等；stretch拉伸行或列以填充交叉轴剩余空间。

3. 布局

- 弹性盒子布局
  - 弹性项目属性
    - order
      - 属性值：整数，默认值为 0。
      - 作用：定义弹性项目的排列顺序，数值越小，排列越靠前。可以通过设置不同的order值来改变弹性项目在弹性容器中的显示顺序。
    - flex-grow
      - 属性值：数字，默认值为 0。
      - 作用：定义弹性项目的放大比例。当弹性容器有剩余空间时，具有flex-grow值的弹性项目会按照其指定的比例分配剩余空间，以放大自身尺寸。
    - flex-shrink
      - 属性值：数字，默认值为 1。
      - 作用：定义弹性项目的缩小比例。当弹性容器空间不足时，弹性项目会根据flex-shrink的值来决定是否缩小以及缩小的比例，以适应容器空间。

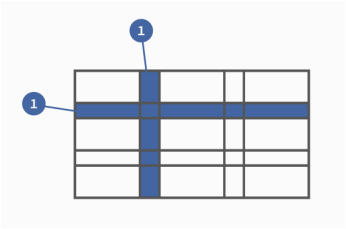
3. 布局

- 弹性盒子布局
  - 弹性项目属性
    - flex-basis
      - 属性值：长度值（如px、em等）或auto（默认值）。
      - 作用：指定弹性项目在主轴方向上的初始大小。浏览器会根据flex-basis的值来计算弹性项目在不考虑伸缩情况下的原始尺寸。
    - flex
      - 属性值：是flex-grow、flex-shrink和flex-basis三个属性的缩写，默认值为0 1 auto。
      - 作用：用于在一个属性中同时设置弹性项目的放大、缩小和初始尺寸属性，例如flex: 1 2 200px表示flex-grow: 1，flex-shrink: 2，flex-basis: 200px。
    - align-self
      - 属性值：auto（默认值）、flex-start、flex-end、center、baseline、stretch。
      - 作用：允许单个弹性项目在交叉轴上有不同于其他项目的对齐方式，可覆盖align-items属性的值。

示例4-3-4：弹性盒子布局属性详解
示例4-3-5：弹性盒子布局

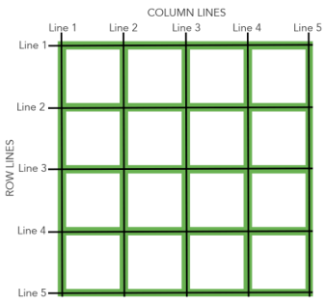
3. 布局

- 网格布局
  - 一种二维布局模型，用于创建二维网格容器和项目。通过display: grid或display: inline-grid将元素设置为网格容器，然后使用grid-template-columns和grid-template-rows定义网格的列和行，使用grid-gap设置网格项目之间的间距。
  - 适用场景：适用于二维布局，如网页的整体布局、复杂的表格布局等。
  - 容器里面的水平区域称为"行"（row），垂直区域称为"列"（column）。



3. 布局

- 网格布局
  - 行和列的交叉区域，称为"单元格"（cell）。正常情况下，n行和m列会产生n x m个单元格。比如，3行3列会产生9个单元格。
  - 划分网格的线，称为"网格线"（grid line）。水平网格线划分出行，垂直网格线划分出列。正常情况下，n行有n + 1根水平网格线，m列有m + 1根垂直网格线，比如三行就有四根水平网格线。



3. 布局

- 网格布局
  - 容器属性
    - display: grid | inline-grid
      - 作用：将元素定义为网格容器，grid使容器成为块级网格，inline-grid使其成为内联网格。
      - 示例：display: grid;
    - grid-template-columns / grid-template-rows
      - 作用：定义网格的列和行的大小。
      - 示例：grid-template-columns: 1fr 2fr 1fr;表示创建三列，第一列和第三列宽度相同，第二列是它们的两倍。
    - grid-template-areas
      - 作用：通过命名区域来定义网格的布局结构。
    - grid-gap / row-gap / column-gap
      - 作用：设置网格中单元格之间的间距，grid-gap是row-gap和column-gap的合并写法。
      - 示例：grid-gap: 10px;或row-gap: 10px; column-gap: 20px;
    - justify-items / align-items
      - 作用：justify-items控制单元格内容在水平方向的对齐方式，align-items控制垂直方向的对齐方式。
      - 示例：justify-items: center;使内容在单元格中水平居中，align-items: start;使内容在单元格顶部对齐。

3. 布局

- 网格布局
  - 容器属性
    - justify-content / align-content
      - 作用：justify-content用于控制整个网格在容器水平方向的对齐方式，align-content用于控制垂直方向的对齐方式。
      - 示例：justify-content: space-around;使网格在水平方向均匀分布，align-content: end;使网格在容器底部对齐。
    - grid-auto-columns / grid-auto-rows
      - 作用：定义自动生成的网格轨道的大小。
      - 示例：grid-auto-columns: 150px;表示自动生成的列宽度为 150px。
    - grid-auto-flow
      - 作用：控制自动布局算法如何放置网格项目。
      - 示例：grid-auto-flow: row dense;表示按行填充，并且尽可能紧凑地填充网格。

3. 布局

- 网格布局
  - 项目属性
    - grid-column-start / grid-column-end / grid-row-start / grid-row-end
      - 作用：指定网格项目在网格中的起始和结束位置，通过指定列线或行线的编号或名称来定位。
      - 示例：grid-column-start: 2; grid-column-end: 4;表示项目从第 2 列开始，到第 4 列结束。
    - grid-column / grid-row
      - 作用：是grid-column-start和grid-column-end以及grid-row-start和grid-row-end的合并写法。
      - 示例：grid-column: 1 / 3; grid-row: 2 / 4;表示项目跨越第 1 列到第 3 列，第 2 行到第 4 行。
    - grid-area
      - 作用：可以指定项目所在的命名区域，也可用于合并grid-row-start、grid-column-start、grid-row-end和grid-column-end属性。
      - 示例：grid-area: main;将项目放置在名为main的区域，或grid-area: 2 / 2 / 4 / 4;表示项目从第 2 行第 2 列开始，到第 4 行第 4 列结束。

3. 布局

- 网格布局
  - justify-self / align-self
    - 作用：用于单独控制某个网格项目在其单元格内的水平和垂直对齐方式，优先级高于容器的justify-items和align-items属性。
    - 示例：justify-self: end;使项目在单元格内右对齐，align-self: center;使其垂直居中。

示例4-3-6：网格布局属性详解
示例4-3-5：网格布局

3. 布局

- 多列布局
  - 允许将内容分割成多个列进行显示，类似于报纸或杂志的排版方式。可以使用column-count、column-width、column-gap和column-rule等属性控制列的数量、宽度、间距和分隔线。
  - 适用场景：适用于文本较多的页面，如文章页面、博客页面等，可提高页面的可读性。
  - column-count
    - 作用：指定元素应该被分割成的列数。
    - 取值：一个正整数，表示列的数量。
    - 示例：column-count: 3;将元素内容分为 3 列显示。
  - column-width
    - 作用：指定每列的宽度。浏览器会根据容器的大小和column-width的值来计算列数，以尽可能多地容纳列，同时保持每列的宽度不小于指定值。
    - 取值：可以是长度值（如 px、em 等）或auto。
    - 示例：column-width: 200px;表示每列的宽度为 200 像素。

3. 布局

- 多列布局
  - column-gap
    - 作用：指定列与列之间的间隙宽度。
    - 取值：长度值（如 px、em 等）。
    - 示例：column-gap: 20px;表示列与列之间的间距为 20 像素。
  - column-rule
    - 作用：用于设置列与列之间的边框样式，是column-rule-width、column-rule-style和column-rule-color的复合属性。
    - 取值：具体取值与常规的边框属性取值类似，如1px solid #ccc表示 1 像素宽、实线样式、颜色为 #ccc 的列边框。
    - 示例：column-rule: 2px dashed blue;将在列之间添加一条 2 像素宽、蓝色的虚线边框。
  - column-span
    - 作用：指定元素是否跨越所有列。
    - 取值：none表示元素不跨越列，在自己所在列显示；all表示元素跨越所有列。
    - 示例：在多列布局中，一个标题元素想要横跨所有列，可以设置column-span: all;

3. 布局

- 多列布局
  - column-fill
    - 作用：指定如何填充列。
    - 取值：auto表示内容自动填充列，直到内容结束；balance表示浏览器会尝试平衡各列的高度，使它们尽可能相等。
    - 示例：当有多段文本在多列布局中时，column-fill: balance;可让各列高度尽量一致，避免出现某一列过高或过低的情况。
  - columns
    - 作用：是column-width和column-count的复合属性，用于一次性设置列宽和列数。
    - 取值：先指定列宽，再指定列数，中间用空格隔开，如150px 3表示列宽为 150 像素，共 3 列。
    - 示例：columns: 200px 4;等价于分别设置column-width: 200px;和column-count: 4;

示例4-3-6：多列布局

4. 响应式

- 响应式设计
  - 响应式设计（Responsive Design）是一种网页设计方法，旨在使网页能够在各种不同的设备和屏幕尺寸上（如桌面电脑、笔记本电脑、平板电脑、手机等）都能提供良好的用户体验。它通过使用 HTML 和 CSS 技术，根据设备的屏幕大小、分辨率、方向等因素自动调整网页的布局、内容显示方式和元素大小，确保网页在任何设备上都能清晰、易读且功能正常。
  - 提升用户体验：如今，用户使用各种设备访问网页，响应式设计可以让用户无论使用何种设备，都能方便地浏览网页内容，减少用户在不同设备上的操作困扰，提高用户满意度。
  - 适应多平台需求：随着移动设备的普及，越来越多的用户通过手机和平板电脑访问互联网。响应式设计能够满足不同平台用户的需求，扩大网站的受众范围。
  - 提高搜索引擎排名：搜索引擎（如 Google）更倾向于优先展示响应式网站，因为它们提供了更好的用户体验。采用响应式设计可以提高网站在搜索引擎结果中的排名，增加网站的流量。
  - 降低开发和维护成本：与为不同设备开发多个独立的网站版本相比，响应式设计只需要开发和维护一个网站，大大降低了开发成本和维护工作量。

4. 响应式

- 关键技术
  - 弹性布局
    - 百分比布局：使用百分比来定义元素的宽度和高度，使元素能够根据父元素的大小自动调整。

```
.container {
    width: 80%;
    margin: 0 auto;
}

.box {
    width: 50%;
    float: left;
}
```

    - Flexbox（弹性盒子布局）：是一种一维的布局模型，提供了强大的弹性空间分配和对齐能力。

```
.container {
    display: flex;
    justify-content: space-between;
}

.box {
    flex-basis: 30%;
}
```


4. 响应式

- 关键技术
  - 弹性布局
    - Grid（网格布局）：是一种二维的布局模型，允许开发者将页面划分为行和列，然后将元素放置在网格的特定位置。

```
.container {
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    grid-gap: 20px;
}
```

4. 响应式

- 关键技术
  - 媒体查询
    - 媒体查询是响应式设计的核心技术之一，它允许开发者根据不同的设备特性（如屏幕宽度、高度、分辨率等）应用不同的 CSS 样式。

```css
/* 当屏幕宽度小于 768px 时应用以下样式 */
@media (max-width: 768px) {
    .box {
        width: 100%;
        float: none;
    }
}
```


4. 响应式

- 关键技术
  - 弹性图片和媒体元素
    - 使用 max-width: 100%; height: auto; 让图片和媒体元素能够根据父元素的大小自动调整尺寸，避免在小屏幕设备上出现溢出或变形的问题。

```css
img {
    max-width: 100%;
    height: auto;
}
```

4. 响应式

- 关键技术
  - viewport
    - 在移动设备普及之前，网页主要是为桌面浏览器设计的。当在移动设备上浏览这些网页时，由于屏幕尺寸较小，网页可能会出现显示不全、缩放异常等问题。而 viewport 元标签就是为了解决这些问题而引入的，它允许开发者控制网页在移动设备上的视口（viewport）大小和缩放比例，从而实现更好的响应式设计。

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

    - name="viewport"：指定该元标签的名称为 viewport，表示这是一个与视口相关的设置。
    - content 属性：包含了多个用于定义视口行为的参数，各个参数之间用逗号分隔。在 content="width=device-width, initial-scale=1.0" 中，涉及到两个重要参数：
    - width=device-width：
    - width 参数用于设置视口的宽度。device-width 表示将视口的宽度设置为设备的实际屏幕宽度。这样可以确保网页在不同屏幕尺寸的设备上都能以合适的宽度显示，避免出现横向滚动条或内容显示不全的问题。
    - initial-scale=1.0：
    - initial-scale 参数用于设置页面首次加载时的缩放比例。1.0 表示不进行缩放，即页面以原始大小显示。这有助于保证网页在不同设备上的显示效果一致，让用户能够以自然的方式浏览网页内容。

4. 响应式

- 关键技术
  - viewport
    - 除了 width 和 initial-scale，viewport 元标签的 content 属性还支持其他一些参数，以下是一些常见的参数及其作用：
      - minimum-scale：设置页面允许的最小缩放比例。例如，minimum-scale=0.5 表示用户不能将页面缩小到小于原始大小的 50%。
      - maximum-scale：设置页面允许的最大缩放比例。例如，maximum-scale=2.0 表示用户不能将页面放大到超过原始大小的 200%。
      - user-scalable：设置用户是否可以手动缩放页面。取值为 yes 或 no，例如，user-scalable=no 表示禁止用户手动缩放页面。

示例4-4-1：响应式网站

4. 响应式

- 栅格系统
  - 栅格系统（Grid System）也称为网格系统，是一种在平面设计和网页设计中广泛使用的布局结构，它将页面划分成一系列的行（rows）和列（columns），形成规则的网格框架，通过这些网格来组织和对齐页面上的元素，如文本、图片、按钮等。栅格系统能够帮助设计师创建出具有一致性、协调性和可读性的布局，使页面更加整洁、有序和易于浏览。
  - 网页设计中栅格系统的工作原理
    - 列（Columns）：是栅格系统的基本垂直结构单元。常见的栅格系统有 12 列、16 列或 24 列等，列的数量决定了页面布局的灵活性。例如，12 列栅格系统可以很方便地将页面划分为 1/2、1/3、1/4 等不同比例的区域。
    - 行（Rows）：水平地划分页面，用于容纳列。行的作用是将列组合在一起，形成不同的布局层次。
    - 槽（Gutters）：也称为间距，是列与列之间的空白区域，用于提供视觉上的分隔和呼吸空间，使页面元素不会过于拥挤。
    - 容器（Containers）：是包含行和列的父元素，用于限定栅格系统的宽度。容器可以是固定宽度的，也可以是响应式的，根据不同的屏幕尺寸自动调整宽度。

4. 响应式

- 栅格系统
  - 栅格系统的优势
    - 提高设计效率：栅格系统提供了一种标准化的布局框架，设计师可以快速地将元素放置在网格中，减少了反复调整布局的时间，提高了设计效率。
    - 增强一致性：通过使用统一的栅格结构，不同页面或不同设计师设计的页面能够保持一致的风格和布局，使整个网站或应用程序具有统一的视觉形象。
    - 提升可读性：合理的栅格布局可以使文本和图像等元素排列整齐，便于用户快速浏览和理解页面内容，提高了页面的可读性。
    - 响应式设计支持：栅格系统可以很容易地实现响应式设计，通过媒体查询和弹性布局，根据不同的屏幕尺寸调整列的宽度和布局方式，使页面在各种设备上都能完美显示。

4. 响应式

- 栅格系统
  - Bootstrap：是一个流行的前端框架，提供了强大的栅格系统。它基于 12 列布局，通过简单的类名就可以快速创建响应式布局。

```
<div class="container">
    <div class="row">
        <div class="col-md-4">
            <p>在中等屏幕及以上设备上占 4 列宽度。</p>
        </div>
        <div class="col-md-8">
            <p>在中等屏幕及以上设备上占 8 列宽度。</p>
        </div>
    </div>
</div>
```


4. 响应式

- 栅格系统
  - 响应式栅格系统是为了适应不同屏幕尺寸的设备而设计的，它可以根据设备的屏幕宽度自动调整列的宽度和布局方式。例如，在大屏幕设备上，可能采用 12 列布局；而在小屏幕设备上，列可能会堆叠在一起，以适应较小的屏幕空间。通过媒体查询和弹性布局技术，可以实现响应式栅格系统。以 Bootstrap 为例，它提供了不同的屏幕尺寸断点（如 xs、sm、md、lg、xl），可以根据这些断点来设置不同的列宽度：

```
<div class="container">
    <div class="row">
        <div class="col-xs-12 col-md-6 col-lg-4">
            <p>在超小屏幕设备上占 12 列，中等屏幕设备上占 6 列，大屏幕设备上占 4 列。
</p>
        </div>
    </div>
</div>
```

示例4-4-1：手动实现栅格系统
示例4-4-2：基于Bootstrap的栅格系统

案例演示

- 案例1：自适应布局
- 案例2：基本网页布局

扩展1：CSS中的数值与单位

- 在CSS中，数值和单位是密不可分的，共同定义了样式属性的具体值。CSS提供了多种单位帮助开发者精确地控制网页的布局和样式，大致可以分为以下几类。
  - 绝对单位
    - cm（厘米），基于物理尺寸的厘米。
    - mm（毫米），基于物理尺寸的毫米。
    - in（英寸），基于物理尺寸的英寸。
    - pt（点），传统印刷中使用的单位，大约等于1/72英寸。
    - pc（派卡），等于12点（pt），主要用于字体大小。
    - px（像素），在屏幕媒体上，像素是屏幕上显示的最小单位，在不同的设备上，一个CSS像素可能对应不同数量的物理像素。

扩展1：CSS中的数值与单位

- 在CSS中，数值和单位是密不可分的，共同定义了样式属性的具体值。CSS提供了多种单位帮助开发者精确地控制网页的布局和样式，大致可以分为以下几类。
  - 相对单位
    - em，相对与当前元素的字体尺寸，如果当前元素的字体大小未设置，则继承其父元素的字体大小。
    - rem，相对于根元素（“<html>”）的字体尺寸。
    - %，百分比单位，相对于其包含块（通常是父元素）的相应尺寸。
    - vm，视口宽度的百分比（Viewport Width），1vm等于视口宽度的1%。
    - vh，视口高度的百分比（Viewport Height），1vh等于视口高度的1%。
    - vmin，vm和vh的较小值。
    - vmax，vm和vh的较大值。
    - ch，数字0的宽度，基于当前字体的“0”字符的宽度。
    - ex，当前字体中“x”的高度，常用于垂直尺寸的计算。

扩展1：CSS中的数值与单位

- 在CSS中，数值和单位是密不可分的，共同定义了样式属性的具体值。CSS提供了多种单位帮助开发者精确地控制网页的布局和样式，大致可以分为以下几类。
  - 灵活单位：灵活单位通常用于响应式设计，以适应不同屏幕尺寸和分辨率。
    - fr，在网格布局，“fr”单位用于分配容器中剩余的空间，例如“1fr”，“2fr”将容器的剩余空间分为三等份，第一个占一份，第二个占两份。
  - 无单位值：有些CSS属性可以接受无单位的数值
    - line-height，当设置为数值时（如 1.5），表示对当前字体尺寸的倍数
    - z-index，虽然“z-index”属性接受整数和负数，但本质上并不依赖于任何单位

重点回顾

- 盒模型
  - 组成部分：由内容、内边距、边框和外边距构成。内容尺寸通过 “width” 和 “height” 设置；内边距在内容与边框间；边框环绕内边距和内容，可设置样式、宽度和颜色；外边距用于分隔元素，可设为负值致元素重叠。
  - 相关属性：“margin” 控制元素间距，取值多样且垂直相邻块级元素外边距会合并；“padding” 控制元素内部内容与边框间距，影响元素实际尺寸且背景会延伸至此；“display” 决定元素显示方式，不同值有不同布局效果；“visibility” 控制元素可见性，不改变布局结构 。
- 定位
  - position 属性：有静态、相对、绝对、固定、粘性定位几种方式，改变元素正常文档流行为。“z - index” 属性控制元素垂直堆叠顺序。
  - float 属性：使元素向左或向右浮动，其他元素围绕，会引发父元素高度塌陷问题，可通过创建 “clearfix” 类或设置 “overflow” 属性解决。
- 布局
  - 常见布局方式：正常流布局按 HTML 文档顺序排列元素；浮动布局用于多列布局；定位布局用于精确控制元素位置；弹性盒子布局是一维布局，可灵活控制子元素排列、对齐和分布；网格布局为二维布局，方便创建复杂页面布局；多列布局适用于文本较多页面。
  - 各布局属性：如弹性盒子布局有容器和项目属性，网格布局也有相应容器和项目属性，多列布局通过多个属性控制列的样式 。

重点回顾

- 响应式
  - 设计概念与优势：使网页适应不同设备，提升用户体验、满足多平台需求、提高搜索引擎排名、降低开发和维护成本。
  - 关键技术：包括弹性布局（百分比布局、Flexbox、Grid）、媒体查询、弹性图片和媒体元素、viewport 元标签设置、栅格系统等，其中栅格系统可借助如 Bootstrap 框架实现响应式布局。
- 扩展知识：CSS 中的数值与单位包括绝对单位、相对单位、灵活单位和无单位值，不同单位适用于不同场景，用于精确控制网页布局和样式。

信创智能医疗系统研发课程体系

河南中医药大学信息技术学院（智能医疗行业学院）

河南中医药大学信息技术学院（智能医疗行业学院）智能医疗教研室

河南中医药大学医疗健康信息工程技术研究所

51