CSS（层叠样式表）

基本语法

- selector1，selector2{}
    - 属性索引表
        - CSS 参考 - CSS：层叠样式表 | MDN (mozilla.org)
- selector是对标签对的选择
- 注释
    - /-----/

基本属性

- color:
    - color:green;
    - color:#66ccff;
- list-style-type:
    - list-style-type: none;
    - 去掉列表的默认风格
- font-size:
    - font-size:22px
- width:
    - width:22px
- background-color
- border:
    -
- font-family:字体族

    - 指定的是一个优先级从高到低的字体列表
    - 对字体的选择是逐字进行，不是选中第一个拥有的，而是选中合适的
    - <family-name>：
        - 带引号
    - <generic-name>：通用字体族
        - 不带引号，是关键字
- a
    - a:link
        - 未访问的链接
    - a:visited
        - 已访问的链接
    - a:hover
        - 鼠标移动到链接上
    - a:active
        - 选定的链接
        - (当你点击链接时，请尝试按住鼠标按钮
    - a:hover 必须被置于 a:link 和 a:visited 之后，才是有效的
    - a:active 必须被置于 a:hover 之后

函数

- calc():计算

选择器

- 选择器列表
    - 多个选择器具有相同的样式
    - 用,隔开

- 类型、类、id选择器
  - 类型（标签）
    - h1
  - 类（class）
    - .class
  - id
    - #unique
- 属性选择器
  - a[title]
  - li[class="a"]
    - 匹配只有一个a类的选择器，不过不会选中一部分值为a而另一部分是另一个用空格隔开的值的类
  - li[class~="a"]
    - 会匹配一个含有a类，不过也可以匹配一列用空格分开、包含a类的值
  - li[class^="a"]
    - 匹配了任何值开头为a的属性
  - li[class$="a"]
    - 匹配了任何值结尾为a的属性
  - `li[class*="a"]`
    - 匹配了任何值的字符串中出现了a的属性
  - li[class^="a" i]
    - 不区分大小写
- 伪类、伪元素选择器
  - 选择处于特定状态的元素
  - 伪类（：）

        - a:vistied
        - a:hover
    - 伪元素（：：）
        - `::first-line`
            - 选中第一行
        - `::before`与content属性结合
            - 将content：“sss”字符串插入文本开头
        - `::after`
        - 参考伪元素：伪类和伪元素 - 学习 Web 开发 | MDN (mozilla.org)
- 全局选择器
    -
- 关系选择器
    - 后代选择器：“空格”
        - .box p{}:选中的是box类 的p元素
    - 子代选择器：>
        - .parent > p{}
        - 只选择作为父元素直接子元素的元素，而不会选择嵌套在更深层次的子元素
    - 邻接兄弟：+
        - `body h1 + p .special`
        - 在 `<body>` 之内，紧接在 `<h1>` 后面的 `<p>` 元素的内部，类名为 special
    - 通用兄弟：~
        - `p ~ img`
        - 选中所有的`<p>`元素后任何地方的`<img>`元素

CSS应用方法

- 外部样式表
    - link css文件
- 内部样式表
    - 把 CSS 放置在包含在 HTML <head> 元素中的 <style> 元素内
- 内联样式
    - 标签的style属性

层叠与继承

- 层叠
    - 当应用两条同级别的规则到一个元素的时候，写在后面的就是实际使用的规则
    - 优先级高的覆盖优先级底的样式
- 优先级
    - 一个标签有多个规则（选择器）
    - 优先级计算
        - 一个选择器的优先级可以说是由三个不同的值（或分量）相加，可以认为是百（ID）十（类）个（元素）——三位数的三个位数
        - ID：选择器中包含 ID 选择器则百位得一分
            - 以#开头
        - 类：选择器中包含类选择器、属性选择器或者伪类则十位得一分
        - 元素：选择器中包含元素、伪元素选择器则个位得一分
    - 内联样式，即 style 属性内的样式声明，优先于所有普通的样式
- 继承
    - inherit(为属性如color设置的)
        - 子元素属性和父元素相同
    - initial
        - 元素的属性值设置为该属性的初始值

    - revert
        - 属性值重置为浏览器的默认样式
    - unset
        - 属性重置为自然值
        - 如果属性是自然继承那么就是 inherit，否则和 initial 一样

背景和边框

- background-color：背景色
- background-image
    - background-image: url(balloons.jpg);
    - 多个图像用，隔开
- background-repeat
- background-size
    - 长度
    - 百分比
- background-position
- 背景渐变色
    - 代码生成器：CSS Gradient — Generator, Maker, and Background
- background-attachment：背景滚动方式
    - scroll:使元素的背景在页面(网页)滚动时滚动
    - fixed:使元素的背景固定在视口上
    - local：将背景固定在它所设置的元素上，所以当你滚动该元素时，背景也随之滚动
- border-radius：盒子上的圆角

溢出

- CSS 中万物皆盒，设定盒子的大小后，里面东西太多会溢出

- overflow
    - overflow：visible
        - 溢出内容不隐藏
    - overflow：hidden
        - 溢出内容隐藏
    - overflow: scroll
        - overflow-y: scroll
            - 仅在 y 轴方向滚动
        - overflow: auto
- 可变尺寸
    - min-height:盒子就会一直保持大于这个最小高度但是如果有比这个盒子在最小高度状态下所能容纳的更多内容，那么盒子就会变大
    - max-width:在没有足够空间以原有宽度展示图像时，让图像缩小，同时确保它们不会比这一宽度大

表格

单位

- px
    - 像素
- em
    - 相对长度单位
    - 用来设置文本的字体尺寸的，相对于父级元素对象内文本的字体尺寸；如果没有人为设置当前对象内文本的字体尺寸，那么它相对的是浏览器默认的字体尺寸（16px）
- rem
    - rem只相对于根目录，即HTML元素
    - 解决em的缺点

- %
    - 它和em差不多一样，都是相对于父级元素
    - 但%可以在很多属性中使用，比如：width、height、font-size等。
- fr
    - 用于在网格布局中分配空间
    - fr单位被用于在一系列长度值中分配剩余空间，如果多个已指定了多个部分，则剩下的空间根据各自的数字按比例分配

文本

- color:颜色
    - 关键字：red
    - 十六进制：#66ccff
    - rgb(10,82,77)
- font-family：字体族
- font-size:字体大小
    - px
    - em
    - rem
- font-style
    - 用来打开和关闭文本 italic (斜体)
- font-weight
    - 设置文字的粗体大小
- text-transform
    - 字体转换（大小写）
- text-decoration
    - 设置/取消字体上的文本装饰 (你将主要使用此方法在设置链接时取消设置链接上的默认下划线。)

- text-shadow：文字阴影
- text-align
    - 控制文本如何和它所在的内容盒子对齐
- line-height
    - 设置文本每行之间的高
    - 可以设置一个无单位的值，作为乘数
- letter-spacing
    - 字母与字母之间的间距
- word-spacing
    - 单词与单词之间的间距

布局

- 正常布局流
    - 在不对页面进行任何布局控制时，浏览器默认的 HTML 布局方式
- display属性
    - 设置标签的布局方式，所有的标签都有默认的布局方式，如
        - <p>:display:block:英文段落显示在一个段落的下面
        - <a>:display:inline:链接将与文本的其余部分保持内联，并且不会打断到新行
- display: flex(弹性盒子)
    - 用于创建横向或是纵向的一维页面布局
    - 只需要在想要进行 flex 布局的父元素上应用`display: flex`，所有直接子元素都将会按照 flex 进行布局
    - 父元素被称为flex container，子元素被称为flex item
    - flex container属性
        - flex-direction：指定主轴的方向
            - row

        - column
    - 换行：`flex-wrap: wrap;`
        - 当子代超出了它们的容器
        - 同时在子代中添加`flex: 200px;`
- flex item属性
    - `flex: 1;`
        - 表示每个 flex 项沿主轴的可用空间大小
        - flex 值为 1，这表示每个元素占用空间都是相等的
    - `flex: 1 200px;`
        - 每个 flex 项将首先给出 200px 的可用空间，然后，剩余的可用空间将根据分配的比例共享
- 弹性盒子的真正价值可以体现在它的灵活性/响应性，如果你调整浏览器窗口的大小，或者增加一个 <article> 元素，这时的布局仍旧是好的
- align-items 控制 flex 项在交叉轴上的位置
    - `align-items: center;`
- justify-content 控制 flex 项在主轴上的位置
    - flex-start：使所有 flex 项都位于主轴的开始处
    - `flex-end` 来让 flex 项到结尾处
    - center
    - `space-around`：使所有 flex 项沿着主轴均匀地分布，在任意一端都会留有一点空间
    - `space-between`：和 `space-around` 非常相似，只是它不会在两端留下任何空间
- flex item顺序
    - 所有 flex 项默认的 order 值是 0
    - `order: 1;`
    - order 值大的 flex 项比 order 值小的在显示顺序中更靠后

        - `order: -1;`
- display: grid（网格）
    - Grid 布局则被设计用于同时在两个维度上把元素按行和列排列整齐
    - 父本划分网格
        - `grid-template-rows: 100px 100px;`（行宽度）（在父本中设置）
        - `grid-template-columns: 1fr 1fr 1fr;`（列宽度）（在父本中设置）
        - 重复构建列
            - `grid-template-columns: repeat(3, 1fr);`
        - 网格间隙：grid-gap: 20px;
            - 间隙距离可以用任何长度单位包括百分比来表示，但不能使用`fr`单位
    - 子本使用格子
        - grid-column 和 grid-row
            - grid-column CSS 属性是 grid-column-start (en-US) 和 grid-column-end (en-US) 的简写属性
            - 用“ /”分隔
            - 左闭右开
        - 指定每一个子元素应该从哪一行/列开始，并在哪一行/列结束
    - 隐式网格
        - 为了放显式网格放不下的元素，浏览器根据已经定义的显式网格自动生成的网格部分
        - 手动设置
            - `grid-auto-rows: 100px;`
    - minmax()
        - 为一个行/列的尺寸设置了取值范围

        - `minmax(100px, auto)`，那么尺寸就至少为 100 像素，并且如果内容尺寸大于 100 像素则会根据内容自动调整
- 浮动（float）
    - `left` — 将元素浮动到左侧
    - `right` — 将元素浮动到右侧
    - `none` — 默认值，不浮动
    - `inherit` — 继承父元素的浮动属性
    - 一个浮动元素会被移出正常文档流，且其他元素会显示在它的下方
- 定位（position）
    - 相对定位
        - 从它的默认位置按坐标进行相对移动
        - `position: relative; top: 30px; left: 30px;`
    - 绝对定位
        - 以坐标的形式相对于它的容器定位到 web 页面的任何位置
        - `position: absolute; top: 30px; left: 30px;`
    - 固定定位
        - 定位的坐标不会应用于"容器"边框（html）来计算元素的位置，而是会应用于视口（窗口）(viewport) 边框
        - 当滑动html时候，这个组件位置不变
        - `position: fixed; top: 30px; left: 30px`
    - 粘性定位
        - 它会在正常布局流中滚动，直到它出现在了我们给它设定的相对于容器的位置，这时候它就会停止随滚动移动，就像它被应用了`position: fixed`一样
        - `position: sticky; top: 30px; left: 30px;`

媒体查询

- 仅在浏览器和设备的环境与你指定的规则相匹配的时候 CSS 才会真的被应用，媒体查询是响应式 Web 设计的关键部分，因为它允许你按照视口的尺寸创建不同的布局
- 语法
    - `@media media-type and (media-feature-rule) { /* CSS rules go here */}`
    - 媒体类型：告诉浏览器这段代码是用在什么类型的媒体上
        - all
        - print
        - screen
        - speech
    - 媒体表达式：是一个被包含的 CSS 生效所需的条件
        - 媒体特征
            - `min-width`、`max-width`和`width`
                - 检测屏幕的宽度
            - `orientation`
                - 检测屏幕竖放（portrait）和横放（landscape）模式
        - 媒体查询的逻辑特征
            - and
            - 或：逗号
            - not