CSS基础

1.基本的语法规则

CSS是由CSS选择器与样式规则构成的, 而样式规则又是由属性与属性值构成的; 属性与属性之间使用分号进行分离, 最后一行属性值末尾可以不用添加分号.

> CSS的基本语法结构为:
> selector{
> prop : propValue
> prop : propValue
> prop : propValue
> }

2.在HTML中添加CSS语法

利用CSS语法, 可以为 html 文档增加页面样式. 有三种方式可以编写CSS代码.

> ***行内样式*** : 直接在 html 标签中添加 style 属性即可.
>
> > **优点:** 直接在标签中编写样式, 精准定位
> > **缺点:** 容易造成代码冗余, 结构混乱, 结构与样式没有分离
>
> ***内部样式*** : 统一在 html 文档的 <head> 标签中添加 <style> 标签, 并在 <style> 标签中利用 CSS 规则编写代码.
>
> > **优点:** 降低了行内的冗余, 让页面结构更加简洁
> > **缺点:** 并未完全实现结构与样式的分离, 当CSS代码量过大时, 程序员在 html 结构布局与 CSS 样式代码之间来回切换过于麻烦, 并且在低网速的情况下, 页面会先出现 html 结构, 再出现 CSS 样式, 影响用户体验.
>
> ***外部样式*** : 将CSS代码单独使用一个文件编写, 并通过 **<link>** 标签或者 **@import url();** 的方式导入, 注意, 这里的分号是必不可少的.
>
> > **优点:** 充分做到了结构与样式相分离

```
<!--使用 link 标签时-->
<link type="text/css" rel="stylesheet" href="path/cssFile.css"/>
<!--使用 @import url(); 时-->
<style>
    @import url("path/cssFile.css");
</style>
```

> ***<link> 标签与 @import url(); 方法的区别在哪?***
>
> 1. **语法不同**
> 2. **加载顺序不同**. 前者与 html 同时加载, 后者先加载 html 结构, 再加载 CSS 样式
> 3. **操作 dom 不同**. 前者可以通过 js 直接操作 dom, 后者不能操作 dom
> 4. **兼容性不同**. 前者兼容性较好, 后者兼容性不好

3.CSS样式的优先级

遵循就近原则, 一般性原则为: ***!important > 行内样式 > 内部样式 > 外部样式***

4.CSS选择器

在 CSS 中, 有三种简单的选择器, 分别是: ***基础选择器、层次选择器、伪类选择器***

> ***基础选择器.***
>
> > **标签选择器** : 通过标签的名字来选择对应的标签, 格式为 tagName{ }
> > **类型选择器** : 通过给标签自定义 class 属性, 按照该属性名筛选标签, 格式为 .className{ }
> > **ID选择器** : 通过为标签定义 id 属性值, 按照该属性值查找标签, id值唯一, 格式为 #idName{ }
> > **通配符选择器** : 选择 html 中的所有元素, 常用写法为 *{margin:0;padding:0}, 用来清除所有标签的默认外边距与内边距.
> > **群组选择器(并集选择器)** : 使用逗号 , 隔开各个选择器, 可以将列出的所有选择器执行一致的样式, 格式为 tagName1,tagName2,tagName3{ }
>
> ***层次选择器.***
>
> > **后代选择器** : 选择器选择的元素范围包括所有的目标后代元素, 格式为 fatherSelector subSelector{ }
> > **子选择器** : 选择器选择的元素范围仅包含亲子代, 格式为 fatherSelector > firstSubSelector{ }
>
> ***伪类选择器.***

> 通常, 在 <a> 超链接标签中应用动态伪类选择器. 有四个状态可供选择, 分别是:
>
> > a:link : 当超链接未被点击时的状态, 默认文本颜色为蓝色
> > a:visted : 当超链接至少被点击一次后的状态, 默认文本颜色为紫色
> > a:hover : 当用户鼠标悬停在超链接上时
> > a:active : 当用户使用鼠标点击超链接的状态
>
> 当需要在同一个标签中使用所有这些伪类时, 需要按照 link > visted > hover > active 的顺序使用.

4.1选择器权重

选择一个标签元素, 可以用上述不同的方法来实现, 当使用不同的方法选择到同一个元素时, 该元素应该应用哪一种样式, 需要依靠选择器的权重来进行判断.
一般地, 按照 !important > id选择器 > class类选择器 > tagName标签选择器 > * 通配符选择器

CSS样式的层叠性

> 当我们使用不同的选择器或者不同的样式表对同一个标签应用样式时, 会按照权重的大小决定标签应该使用哪种样式.
> 特别地, 当选择器的权重相同时, 浏览器会使用后出现的样式覆盖先出现的样式.

5.文本属性

与文本有关的属性有以下几种:

> **font-size : 设置字体大小.**
>
> > 在浏览器中, 默认的字体大小一般是 16px , 其中的 px 是单位, 意思是像素, 除此之外, 还有 pt(磅值), em(相对单位)等单位.
> > 一般浏览器中支持显示的最小字体是 10px/12px ,并且 1em=16px.
>
> **color : 设置文本颜色.**
>
> > 1. 使用英语单词设置颜色, 例如: red、green、blue 等
> > 2. 使用 # 与六位十六进制码来设置颜色, 当1\2位, 3\4位, 5\6位值相同时, 可以简写, 例如: #ff0123 #abc212 #bcbcdd #333 #1ac
> > 3. 使用 rgb(a) 的形式来设置颜色, 其中 a 表示透明度, 每一位取值范围为 0~255, 例如 rgba(255,40,50,0.4) rgb(112,210,21) rgb(100,100,250)
>
> **font-family : 设置字体格式.**

> 支持设置多个字体样式, 用逗号分隔, 浏览器会从左到右依次查找对应字体, 如果直到最后一个字体仍然没有得到支持, 会使用浏览器默认字体
> 特别地, 当字体名称是单个英文单词或者中文格式时, 不需要额外添加引号; 当字体名称是由多个英文单词组合而成, 需要使用引号引用该字体名称

font-weight : 设置字体粗细.

> 支持使用 100~900 的整数数字设置字体的粗细, 默认的字体粗细为 400 , 100 表示细体, 400 表示常规, 700 表示粗体, 900 表示更粗体
> 另外, 还可以使用英语单词来设置字体粗细, lighter、 normal、 bold、 bolder

font-style : 设置字体风格.

> 支持使用对应的英语单词来设置字体风格, italic、 oblique、 normal

text-align : 设置字体横向对齐方式.

> 支持使用对应的英语单词来设置字体风格, left、 center、 right、 justify

line-height : 设置文本行间距.

> 支持使用带有单位的数值设置文本行间距.
>
> > 特别地, 文本行间距可以对单行文本实现垂直居中效果, 通常是将文本的 line-height 属性值设置为容器的高度.

font : 复合属性.

> 使用 font 属性可以一次性为文本设置多个样式.
> 需要遵循以下格式:
>
> > ***font : font-style font-weight font-size/line-height font-family***
> > 需要注意, 使用时, 必须要带字体与字体大小, 并且字体要放在最后面. font-style 与 font-weight 属性可以交换位置.

text-decoration : 设置文本修饰线.

> 支持使用对应的英语单词来设置字体风格, underline、 line-through、 overline、 none

text-indent : 设置文本首行缩进.

> 一般地, 使用 em 相对单位来设置文本的首行缩进.

> **letter-spacing : 设置单词字母间间距(包括中文字符间间距).**
> **word-spacing : 设置单词词间距.**

6.背景属性

> **background-color : 设置标签的背景颜色.**
> 与文本属性中的 color 属性使用相同的属性值
>
> **background-image : 设置标签的背景图片.**
> 通用格式为 **background-image:url( )** 默认情况下, 设置的背景图片会被填充
>
> **background-repeat : 设置背景图片的重复方式.**
>
> > 有四个取值:
> > ***repeat*** : 在水平方向与垂直方向上重复
> > ***no-repeat*** : 不重复
> > ***repeat-x*** : 仅在水平方向上重复
> > ***repeat-y*** : 仅在垂直方向上重复
>
> **background-position : 设置背景图片的位置.**
> 可以使用带有单位的数值来描述位置, 通用格式为: **background-position : x y**, 其中 x 代表水平方向上的位置, y 代表垂直方向上的位置. **可以使用该属性完成精灵图(雪碧图, 妖怪图)的制作效果.**
> 也可以使用关键字来描述位置. 水平方向上的关键词有 **left center right**, 垂直方向上的关键词有 **top center bottom**
>
> **background-size : 设置背景图片的尺寸.**
> 可以使用带有单位的数值来描述位置, 通用格式为: **background-size : x y.** 与 background-position 类似, 其中 x 代表水平方向上的位置, y 代表垂直方向上的位置.
> 也可以使用关键字来描述位置. 包含两个值: **cover 与 contain**
>
> > **cover**: 按比例缩放图片, 图片可能会被裁剪
> > **contain**: 放大图片, 直到有一边充满容器, 可能会造成容器区域留白
>
> **background-attachment : 设置背景图片的附着方式.**
> 利用这个属性, 可以形成视觉差效果, 包含两个可选值: **scroll 与 fixed**
>
> > **scoll**: 背景图片会随着滚动条的滚动而滚动
> > **fixed**: 背景图片被固定在相对于浏览器的位置
>
> **background : 复合属性.**
> 使用该属性可以一次性设置多个背景属性

通用的格式为: **background-color background-repeat background-image background-position/background-size background-attachment**

> **可以忽略以上属性的排列顺序, 但是, 如果需要同时使用 background-position 与 background-size 属性, 需要使用 / 隔开, 并且后面的属性是 background-size.**

7.列表属性

list-style-type : 设置列表导航图标.
可设置 **disc、ciecle、square、none**

list-style-position : 设置列表导航图标的位置.
可设置 **outside、inside**

list-style-image : 自定义列表导航图标为图片.
通过 url() 链接图片路径进行设置

list-style : 复合属性.
使用该属性可以一次性设置上述所有内容, 并且不限制属性顺序, 也不限制属性个数. 一般地, 使用 **list-style : none** 来取消列表的默认显示效果

注意: 以上的 属性可以修饰 ul、ol, 甚至是 li 标签, li标签通过继承父属性 ul 与 ol, 能够取得同样的效果

8.边框属性

border-width : 设置边框的粗细.
可以使用带有单位的数值进行设置, 或者利用英语关键词进行设置: **thin、medium、thick**

border-color : 设置边框的颜色.
与文本属性中的 color 属性使用相同的属性值

border-style : 设置边框的样式.
使用关键词设置边框的样式: **solid、dotted、dashed、double、groove、inset、outset、ridge、hidden 等**

上述每个属性可以取四个值, 分别对应 上边框、右边框、下边框、左边框, 并且, **要知道, 背景颜色是可以蔓延到边框位置的**

border : 复合属性.
使用该属性可以一次性设置上述的所有样式. **font-style** 属性值是必须的, 不限制顺序.

> **border-top-width/style/color、border-right-width/style/color、border-bottom-width/style/color、border-left-width/style/color：分别设置单个方向的边框样式.**
>
> TIPS: 相邻边框边角上的正方形会被平均分为两部分填充, 可以使用这个特性, 利用各个方向的边框来制作三角形、矩形或者梯形效果

9.浮动属性

> **float：设置元素的浮动效果.**
> 使用关键词设置对应的效果: **left、right、none**
> 需要注意的是, 当元素设置了浮动的样式后, 原来的块级元素不再会在页面上占据空间, 而剩余的元素会依次填充这些空白. 并且, 所有被设置有浮动属性的元素会依次排列
> 当浮动元素周围存在文本时, 文本默认会产生环绕效果
>
> > ***特别地, 当设置浮动属性的元素宽度过长, 本行不够显示时, 会自动换行至下一行进行显示. 当设置浮动属性的元素高度不一致时, 后面的元素会优先在前一个元素的右侧或者下方排列, 当后面的元素高度与宽度能够填补前面的空白时, 不会向前填充.***
>
> **clear：清除浮动效果.**
> 使用关键词清除元素的浮动效果: **none、left、right、both**
> 元素在哪个方向浮动, 需要使用对应方向的关键词来清除浮动效果

10.盒子模型

在 HTML 中, 任何一个元素都是由四个区域组成的: **content、padding、border、margin**, 这些区域组成起来的矩形框被称为**元素盒子**, 也就**盒子模型**

> **盒子模型的内容区域**:
>
> > 对于所有的非行内元素, 都可以设置元素的 width 属性与 height 属性, 包含在这个宽度与高度之内的内容叫做内容区域
>
> **盒子模型的内边距：padding 属性.**
>
> > 与设置边框样式一样, 可以使用 **padding-top、padding-right、padding-bottom、padding-left 来分别设置对应方向的内边距**, 或者使用 **padding** 复合属性来一次性设置四个方向的内边距, 对应的方位分别是 **上、右、下、左**
>
> **盒子模型的边框：border 属性.**
>
> border属性与边框属性用法一致

盒子模型的外边距 : margin 属性.

> 与设置盒子模型的内边距一样, 可以使用 **margin-top、margin-right、margin-bottom、margin-left 来分别设置对应方向的外边距**, 或者使用 **margin** 复合属性来一次性设置四个方向的外边距, 对应的方位分别是 **上、右、下、左**
> **可以使用 margin : 0 auto 属性来使块级元素左右居中对齐**
>
> > ***关于盒子模型中 margin 与 padding 的取值说明.***
> >
> > **1.两者均可以取 0**
> >
> > **2.margin 可以取负数, padding 不能取负数**
> >
> > **3.使用 margin : 0 auto 可以使块级元素实现水平居中的效果(相对于页面居中, 相对于父元素居中), 仅限于文档流中的元素, 设置为浮动状态的元素不会触发**
> >
> > **4.在父子关系中的 margin-top, 当需要设置子元素与父元素之间的外边距时, 可以参照下列方法: 其中的 4.2、4.3、4.4 方法会触发 BFC 机制**
> >
> > **4.1为父元素设置边框**
> > **4.2为父元素或者子元素设置 float 属性**
> > **4.3为父元素添加 overflow : hidden 属性**
> > **4.4为父元素添加定位 position : absolute 或者 fixed**
> > **4.5为父元素设置 padding 属性取代 margin 属性, 同时需要修改父元素的容器大小**
> >
> > **5.在兄弟关系中的 margin, 在 HTML 中, 相邻元素之间的 margin 默认会以最大值为准, 较小的边距会被重叠, 要想取消这种效果, 可以通过为它的父元素添加 overflow : hidden 属性**

11.PC端布局

> 传统的 PC 端布局有 **版心布局** 与 **通栏布局**
>
> > **版心布局** : 当浏览器页面放大或者缩小的时候, 布局中心区域始终显示在页面居中位置. **版心布局一般有具体的宽度, 一般使用 margin : 0px auto 来实现左右居中效果**
> >
> > > **建议使用公共的版心, 可以节约代码**
> >
> > **通栏布局** : 无论浏览器放大或者缩小页面, 该区域总是横贯页面的左右两侧.
>
> PC 端布局的内容溢出属性:

> **overflow-x : 设置水平方向的溢出效果**
> **overflow-y : 设置垂直方向的溢出效果**
> **overflow : 复合属性**
>
> 利用复合属性, 同时设置水平方向与垂直方向的溢出效果. 有四个取值: **visible、hidden、scroll、auto**, 其中, **scroll 与 auto 的区别在于:前者会自动添加滚动条, 后者在内容超出时才会显示滚动条**
>
> TIPS: 在网页中要实现单行超出文本为省略号显示时, 需要使用以下步骤:
>
> > 1.为元素盒子设置宽度 **width : value**
> > 2.设置文本空白内容为不折行 **white-space : nowrap**
> > 3.设置元素盒子溢出属性 **overflow : hidden**
> > 4.设置文本溢出属性 **text-overflow : ellipsis**
>
> TIPS: 在网页中实现多行超出文本末行为省略号显示时, 不用设置盒子的 height 属性需要使用以下步骤:
>
> > 1.设置元素盒子宽度 **width : value**
> > 2.设置元素盒子为弹性盒 **display : -webkit-box**
> > 3.设置文本的排列方向 **-webkit-box-orient : vertical**
> > 4.设置文本在第几行进行省略 **-webkit-line-climp : number**
> > 5.设置多余文本内容的显示方式 **overflow : hidden**

12.元素类型及分类

> 在 HTML 中, 所有的元素可以被分为三大类: **行内元素、块级元素、行内块元素**
>
> > **行内元素**: 行内元素默认横向排列, 无法单独实现设置的 width 与 height 属性, 也**无法生效 padding-top、padding-bottom 与 margin-top、margin-bottom 属性**, 但是**可以生效 margin-left、margin-right 与 padding-left 与 padding-right 属性** 行内元素一般用来修饰文本. 常见的行内元素有: ***<b>、、<s>、<del>、、<i>、、<u>、<sub>、<sup>、<font>、<a>、<mark>等***
> > **块级元素**: 块级元素默认纵向排列, 每个块级元素独占一行, 可以单独实现设置的 width 与 height 属性, 通常用来做页面的布局, 常见的块级元素有: ***<p>、<ul>、<ol>、<li>、<dl>、<dt>、<dd>、 、<form>、<header>、<footer>、<article>、<nav>、<table>、<h1>~<h6>等***
> > **行内块元素**: 行内块元素既有行内元素的特性, 也有块级元素的特性. 即它既能够实现设置的 width 与 height, 同时也排列在行内中. 常见的行内块级元素有: ***<input>、<img>、<textarea>、<select>等***

> 对于每种元素的分类, 主要依靠 **display** 属性来区分
>
> **display : 设置元素的类型.**
>
> > **行内元素** 的 display : 一般地, 行内元素的 display 属性为 **inline**
> > **块级元素** 的 display : 一般地, 块级元素的 display 属性为 **block**
> >
> > > 特别地, 一些特殊的块元素的取值:
> > >
> > > 1. **<li>** 标签, display 属性为 **list-item**
> > > 2. **<table>** 标签, display 属性为 **table**
> > > 3. **<tr>** 标签, display 属性为 **table-row**
> > > 4. **<td>** 标签, display 属性为 **table-cell**
> >
> > **行内块元素** 的 display : 一般地, 行内块元素的 display 属性为 **inline-block**
> >
> > > 特别地, **<img>** 元素作为特殊的行内块元素, 它的 display 属性默认为 **inline**, 但是它具有行内块元素的特点
> >
> > **Q : 如何使 HTML 中的元素实现隐藏效果?**
> >
> > > 1. 设置元素的 display 属性为 none, 元素将会脱离文档流, 不占用页面空间
> > > 2. 设置元素的 width 与 height 属性值为 0, 元素将会脱离文档流, 不占用页面空间. 对于有文本的元素, 可以单独控制文本的 font-size 属性或者元素的 overflow 属性隐藏文本
> > > 3. 设置元素的透明度属性 opacity 为 0, 元素不会脱离文档流, 仍然占据页面空间
> > > 4. 设置元素的缩放属性 transform : scale(0), 元素不会脱离文档流, 仍然占据页面空间
> >
> > 通常,使用 **display** 的 **block** 与 **none** 属性可以用来制作**二级菜单的显示与隐藏效果**
>
> **vertical-align : 设置行内块元素的垂直对齐方式.**
>
> > 在一个文本块中, 有四条基准线, 分别是: ***topline***、***middleline***、***baseline***、***bottomline***, 默认地, 文本使用的是基线方式对齐, <img>元素下方有 3px 留白区域. 如果要取消这个留白区域, 需要更改 vertical-align 的值为**除 baseline 之外**的任何值. **并且, 该属性只针对行内块元素有效.**
> >
> > **置换元素与非置换元素**
> > **置换元素**: 置换元素往往没有实际的内容, 有自己的宽度与高度, 并且根据自身的属性与属性值决定元素显示的内容, 一般来说, 行内快元素都是置换元素
> > **非置换元素**: 除上述置换元素之外的所有其他元素都是非置换元素

13.元素的定位

> 在 CSS 中, 元素有**五种**定位方式: ***静态定位、相对定位、绝对定位、固定定位、粘性定位***, 无论使用哪种定位方式, 都需要 **position** 属性并配合 **top、left、right、bottom** 四个辅助属性来完成
>
> **postion : 设置元素的定位方式**
> 下面分别对应不同的定位方式对其值进行说明
>
> **top、left、right、bottom : 分别设置距离参考元素的上、左、右、下的偏移量**
>
> **z-index : 设置定位属性的层级**
> z-index 的取值可正可负, 取值越大, 元素越靠前显示
>
> **静态定位.**
>
> > 又叫做普通文档流定位, 默认的定位方式. 利用 **position : static** 来进行设置, 在这个定位中, **使用偏移量控制定位无法实现, 通常不使用该定位方式**
>
> **相对定位.**
>
> > 相对定位指的是相对于元素本身的位置进行定位, 利用 **position : relative** 来进行设置, 设定偏移量时, 可以取正负值. **相对定位常用来微调元素**
>
> **绝对定位.**
>
> > 一般将绝对定位用于子元素, 利用 **position : absolute** 来进行设置. 设定偏移量时, 参照元素视父元素是否有定位属性而定. **绝对定位通常用来大调元素位置**
> > **a.当父元素有定位属性(relative/abosolute/sticky/fixed)时, 偏移量是相对于父元素而言的**
> > **b.当父元素没有定位属性时, 继续向上查找, 如果某层父元素有定位, 则偏移量是相对于最近一层有定位的父元素而言的; 若所有父元素都没有定位属性, 则最终偏移量是相对于 <body> 元素而言的, 即相对于浏览器左上角**
>
> **固定定位.**
>
> > 固定定位通常用于返回**置顶、广告、楼梯层与遮罩层效果**的实现, 利用 **position : fixed** 来进行设置, 设定的偏移量是相对于浏览器窗口的左上角而言的
>
> **粘性定位.**
>
> > 介于绝对定位和固定定位之间, 经常用来实现**滚动吸顶**的效果, 利用 **position : sticky** 来进行设置. 配合 top 属性使用, 当页面滚动到一定区域的时候, 将元素吸附在顶部

关于定位的一些注意事项:

1. **当子元素是绝对定位时, 父元素可以是相对定位、绝对定位以及固定定位(一般来讲, 父元素使用相对定位, 子元素使用绝对定位)**
2. **添加绝对定位或者固定定位的元素, 会脱离文档流, 兄弟元素会进行占位**
3. **没有高度的元素盒子, 当它存在子元素的时候, 子元素的高度会增加父元素的高度, 当其中的子元素为绝对定位或者固定定位时, 该元素会脱离文档流, 其余元素会占用原元素位置, 导致父元素的高度缩小, 被称作高度塌陷**
4. **利用绝对定位或者固定定位可以将行内元素转变为块级元素(利用 float 属性也可以实现)**
5. **当给元素添加定位属性时, 默认地, 添加定位属性的元素会遮盖未添加定位的元素, 并且后添加定位属性的元素会遮盖先添加定位属性的元素, 这种层级顺序可以通过 z-index 属性来改变**
6. **为自适应元素(未设置元素宽度的元素)添加绝对定位或者固定定位属性后, 元素会显示失败, 如果需要显示自适应元素, 需要在元素中添加文本**
7. **当为设置有 margin : 0 auto 的元素设置绝对定位与固定定位属性时, 原元素不再会实现水平居中效果. 因为使用 margin : 0 auto 实现水平居中的效果只针对文档流中的元素有效, 而设置有绝对定位与固定定位的元素脱离了文档流**

包含块.

> 如果元素 position 属性值为 absolute, 那么包含块就是最近层次的设置有定位的父元素
> 如果元素 position 属性值为 fixed, 那么包含块就是 viewport(视口)浏览器当前的窗口

Q : 如何让子元素在父元素中实现水平垂直居中的效果?

1. **父元素 display : relative, 子元素绝对定位 : top 与 left 偏移量 50%, 再设置子元素的负外边距**
2. **父元素 display : relative, 子元素绝对定位 : top 与 left 偏移量 50%, 再设置子元素的 transform : translate(-50% , -50%)**
3. **父元素 display : relative, 子元素绝对定位 : top、left、right、bottom 设置 0, margin :auto**
4. **父元素 display : flex, 子元素设置 margin : auto**
5. **父元素 display : flex, justify-content : center, align-items : center**
6. **父元素 display : grid, justify-content : center, align-items : center**
7. **父元素 display : table-cell, vertical-align : middle, 子元素设置 margin : auto**

opacity : 设置元素透明度.

使用 opacity 设置元素透明度, 可以实现元素与其内容共同实现透明效果. 取值从 0(透明)~1(不透明).

> 在旧版本的 IE 浏览器中, 使用 **filter : alpha(opacity=number) 进行设置, 其中的 number 取值 [0,100]**

> 其余的浏览器使用兼容写法: **opacity:number, 其中的 number 取值 [0,1]**

Q : 浮动属性 float 与定位属性的区别?

- **不同点:**
  - **i. float 属性会让元素半脱离文档流, 文本会环绕在浮动元素周围; 定位属性属于全脱离文档流, 会遮盖住文本**
  - **ii. float 属性主要用于块级元素的横向排列, 定位属性主要用在定位**
- **相同点:**
  - **i. 都会触发 BFC 机制**
  - **ii. 都会改变元素的类型**
  - **iii. 都会脱离文档流**
  - **iv. 都会让 margin : 0 auto 失效**
  - **v. 都会让宽度自适应元素盒子自适应**
  - **vi. 都会让兄弟元素占位**

14.表格

14.1标签属性

> 在前端页面的开发中, 有两种不同的属性分类: **标签属性** 与 **CSS属性**
>
> > **标签属性**: 标签属性放置在 HTML 开头标签中, 使用 **名/值对 prop=value** 的方式, 使用 **空白** 连接多个属性与属性值
> > **CSS属性**: CSS属性使用 **prop : value;** 的方式展示属性与属性值, 使用 **分号 ;** 连接多个属性与属性值

> 在HTML 中新建表格, 需要使用 **<table>** 标签, 在 <table> 标签中, 有下列**标签属性**可用来调整表格内容与显示样式
>
> **border : 设置表格的边框.**
> border 属性值不需要接任何单位, 并且设置边框后, 表格的外层边框与内层单元格均会添加边框
>
> **width、height : 设置表格宽度与高度.**
> 设置表格的宽度与高度后, 默认地, 当表格中每个单元格内容数量一致的情况下， 表格内单元格的宽度与高度会被平均划分, 并且默认单位是 px. 在设置百分比高度时, 需要为 <table> 元素的父元素指定高度, 否则无法实现

bordercolor : 设置表格的边框颜色.

bgcolor : 设置表格的背景颜色.

align : 设置整个表格的对齐方式.
可以从 **left、center、right** 中选择取值(center已弃用)

cellspacing : 设置表格内单元格与单元格之间的边框间距.

cellpadding : 设置表格内单元格中的文本内边距.

rules : 设置表格内部边框的显示样式.
可以从 **group、rows、cols** 中选择取值

frame : 设置表格外边框的显示样式.
可以从 **above、below、lhs、rhs、box** 中选择取值

要在 <table> 标签中创建表格行, 需要使用 <tr> 标签, 在 <tr> 标签中, 有下列**标签属性**可用来调整表格行内容与显示样式

height : 设置本行单元格的高度.
在 <tr> 标签中, 无法设置 width 属性

bgcolor : 设置本行单元格背景颜色.

align : 设置本行文本水平对齐方式.
可以从 **left、center、right** 中选择取值

valign : 设置本行文本垂直对齐方式.
可以从 **top、middle、bottom** 中选择取值

要在 <table> 标签中创建单元格, 需要使用 <td> 属性, 有下列**标签属性**可用来调整单元格内容与显示样式

width、height : 设置单元格的宽度与高度.
为某个单元格设置好 width 与 height 之后, width **会影响这一列的宽度**, height **会影响这一行的高度**. 并且会覆盖父元素 <td> 的 width 与 height

bgcolor : 设置单元格背景颜色.

> **align : 设置单元格文本水平对齐方式.**
> 可以从 **left、center、right** 中选择取值
>
> **valign : 设置单元格文本垂直对齐方式.**
> 可以从 **top、middle、bottom** 中选择取值
>
> **colspan : 设置单元格横向合并的个数.**
> colspan 后面接一个数值, 数值是希望横向合并单元格的个数, 被合并的单元格需要手动注释或者删除
>
> **rowspan : 设置单元格纵向合并的个数.**
> rowspan 后面跟一个数值, 数值是希望纵向合并单元格的个数, 被合并的单元格需要手动注释或者删除

14.2CSS属性

> **一些需要引起注意的应用在 <table> 标签中的属性** 标**的属性只能用在 <table> 元素中才有效
>
> **border : 设置表格的外部边框.**
>
> **width、height : 设置表格的宽度与高度.**
>
> ****border-spacing : 设置表格中单元格之间的距离.**
>
> ****border-collapse : 设置表格中相邻单元格边框是否堆叠.**
> 取值为 **collapse** 表示相邻单元格的边框会相互折叠, 取值为 **separate** 表示不会相互折叠相邻单元格的边框
>
> **table-layout(表格的布局算法) : 设置表格单元格的宽度是否与单元格中的文本内长度正相关**
> 有两个取值: **auto、fixed**. 前者表示当单元格内的文本增加, 该单元格所在列的宽度会随之增加, 可以灵活调整表格宽度, 但是每次内容发生变化, 都会cch重新渲染; 后者表示固定单元格的宽度, 不随单元格内文本增加而增加单元格宽度
>
> **word-wrap : 设置文本在末尾是否执行自动换行.**
> 使用 **word-wrap : break-word** 强制文本在元素尾部换行, 默认是 normal 值

> **应用在 <td> 标签中的一些属性**
>
> **border : 设置单元格的边框**
>
> **padding : 设置单元格文本与单元格边框的距离**

> **empty-cells：设置隐藏或者显示表格中的空白单元格**
> 可以从 **hide、show** 中选择取值

14.3表格的其他标签及属性

> **标题相关**
>
> **<caption>(双标签)：设置表格的标题.**
> 该标签用来设置表格的标题, 默认标题效果为居中. 在使用该标签时, 需要放于 <table> 标签之后, 第一个 <tr>或者<th> 标签之前
>
> > **caption-side：设置表格标题的位置.**
> > 可以从 **top、bottom** 中选择取值
>
> **<th>(双标签)：设置表格中的标题字段**
> 默认地, 标题字段有加粗且在单元格中居中的效果. 允许在同一行中同时出现 <th> 与 <th>

> **行分组相关**
>
> **<thead>(双标签)：为表格设置头部分组.**
> 在表格中, 头部只能有一个
>
> **<tbody>(双标签)：为表格设置主体分组.**
> 在表格中, 允许有多个主体分组, 当没有设置表格的分组元素时, 浏览器会默认将所有的 <tr> 元素列在 <tbody> 标签中, 所以在使用 **子代选择器** 查找表格内元素时, 不要忘记 <tbody> 标签
>
> **<tfoot>(双标签)：为表格设置脚部分组.**
> 在表格中, 脚部只能有一个

> **列分组相关**
>
> **<colgroup>(双标签)：为表格的列分组**
>
> > 在 <colgroup> 标签中, 有一个标签属性 span, 属性值为一个数值. 该数值指定在表格中的列数. 例如: <colgroup span="2" bgcolor="red"></colgroup>, 表示要把第一列的所有单元格背景色改为红色, 并且已经有归属的列不会再计入下一个 <colgroup> 中, 即下一次分组时从未进行分组的列开始计算列数

15.表单

> 表单一般用作用户登陆界面、搜索框、信息搜集页面等页面
> 表单用来收集用户信息, 并将信息上传至服务器. 利用 <form>标签创建一个表单.

15.1<form>标签及标签属性

> **action : 设置提交表单数据的目标站点**
> 站点包括协议、域名、端口
>
> **method : 设置提交表单数据的方式**
> 使用 **get** 方法表示明文提交, 使用 **post** 方法表示密文提交
>
> **target : 设置提交数据后跳转页面的打开位置**
> 设置为 **_self** 表示在当前页面打开, 设置为 **_blank** 表示新建空白页面打开

15.2<input>标签及标签属性

> **type : 设置 <input> 标签的类型**
> 可以为 type 属性设置以下的值:
>
> > **text : 默认取值, 表示一个普通的输入框.**
> > 用来输入任意文本
> >
> > **passwored : 表示一个密码输入框**
> > 根据浏览器的不同, 输入的值会被替换成 星号* 或者 圆点·
> >
> > **reset : 设置为重置按钮**
> > 在表单中点击该按钮会清空所有已填写的数据
> >
> > **submit : 设置为提交按钮**
> > 在表单中点击该按钮会将数据提交至目标网站
> >
> > **button : 设置为普通按钮**
> > 点击该按钮不会有任何作用, 一般在 js 中为空白按钮添加点击事件
> >
> > **image : 设置为图片域按钮**
> > 图片域按钮需要使用 **src** 属性来设置图片路径, 当点击按钮后, 跳转页面网址中会携带一个用户点击图片坐标的参数
> >
> > **hidden : 设置隐藏域.**
> > 表示该信息为隐藏状态, 在浏览器中不可见, 但是会随着用户执行提交操作而提交数据至服务器

> **file：设置文件域**
> 利用该属性，可以向目标网站提交各种文件
>
> **radio：设置单选按钮**
> 为 <input> 标签设置单选按钮属性时，需要为不同的按钮添加一个 **相同的name** 属性值，有额外的 **checked** 属性表示默认选择该选项
>
> **checkbox：设置多选按钮**
> 为 <input> 标签设置多选按钮属性时，也可以使用额外的 **checked** 属性表示默认选择的选项

15.3<label>标签及标签属性

> 在 H5 中，新增了 <label> 双标签，主要是用来实现用户在点击文字(指双标签中的文本)时，也会选中对应的按钮
>
> **for：设置内容文本绑定的区域**
> 使用 for 属性时，后面接一个 **绑定目标标签的 id 值**

15.4<select>、<option>标签及标签属性

> 使用 <select> 标签创建一个下拉列表选项，一般与 <option> 标签配合使用. <select> 表示下拉列表，<option> 表示下拉列表选项.

> **<select>标签属性**
>
> **size：设置下拉列表选框显示列表项的个数.**
> 默认地，只会显示一个下拉列表选项，并且会显示第一个 <option> 标签中的列表项
>
> **multiple：设置允许下拉菜单同时选中列表项的个数.**
> 默认地，下拉列表只允许选中一个列表项. 并且，在添加该属性之后，size 的值将会变为 4，即在选框中会出现 4 个列表项

> **<option>标签属性**
>
> **selected：设置默认选中的列表项**
> 当父元素 <select> 设置了 multiple 属性后，允许同时为多个列表项添加 selected 属性

> **value：设置列表项的值.**
> 一般地, 设置该属性值为 <option> 标签中的文本内容

15.5<textarea>标签及标签属性、CSS属性

> 使用 <textarea> 可以创建一个文本输入区域, 使用该标签时, 一般不会分开开始标签与结束标签

> **标签属性**
>
> **cols、rows：设置文本输入区域允许输入的宽度与高度.**
> 一般地, 使用 CSS 属性 **width、height** 属性来控制文本输入区域的大小

> **CSS属性**
>
> **resize：设置是否允许用户调整文本输入区域的宽度与高度.**
> 默认情况下, 用户可以更改本文输入区域的大小. 取值为 **both**, 表示用户**可以更改文本输入区域的宽度与高度**; 取值为 **vertical**, 表示**只允许用户更改文本输入区域的高度**; 取值为 **horizontal**, 表示**只允许用户更改文本输入区域的宽度**; 取值为 **none**, 表示**不允许调整文本输入区域的宽度与高度**

15.6其他表单控件

> **<fieldset>与<legend>：设置字段集与字段集标题.**
> 一般用于区块划分
>
> **<iframe>：引入浮动框架集.**
> 使用 <iframe> 标签属性 **src** 来引入需要引入的页面区域.
> tips: <iframe> 标签是**行内块元素**

16.BFC机制

> BFC(Block Formatting Contexts), 被称作块级格式化上下文. 当 BFC 被触发之后, 元素会形成一个独立的区域, 不会干扰到外界, 也不会被外界干扰
>
> **Q：触发 BFC 的机制?**
>
> > **1. 有共同的父元素. 在 HTML 文件中, 顶层标签是 <html>, <html>标签默认是一个共同父元素**

> **2. 当 float 取值不为 none 时**
> **3. overflow 的取值不为 visiable 时**
> **4. display 取值为 inline-block / table-cell / table-caption / flex / inline-flex**
> **5. position 的取值为 absolute / fixed**

Q : 触发了 BFC 机制可以实现哪些效果?

> **1. 解决外边距重合的问题**
> **2. 让产生高度塌陷的父元素重新参与高度的计算**

17.自适应

17.1元素的宽度与高度自适应

> **窗口自适应**: 使用百分比为元素设置宽度与高度, 元素随着浏览器窗口高度与宽度的改变而改变. 需要添加 **heml,body{height:100%}**
> **宽度自适应**: 不为元素设置具体宽度, 默认值是 **width:auto**
>
> > 宽度自适应分为 **浏览器宽度自适应** 与 **父元素宽度自适应**
> > 当不设置块级元素的宽度时, 或者给块级元素设置 **width:100%** 后, 都可以实现"自适应", 但是两者有以下区别:
> >
> > > **未设置宽度的情况下, 添加盒子模型中的各项参数后, 块级元素宽度不会超出浏览器或者父元素的宽度**
> > > **设置元素宽度为 100% 的情况下, 该块级元素的初始内容会继承浏览器或者父元素宽度, 添加盒子模型中的各项参数后, 块级元素宽度会超出原浏览器或者父元素的宽度**
>
> **高度自适应**: 不为元素设置具体高度, 默认值是 **height:auto**
>
> > 高度自适应通常是让元素中的文本自行撑开高度
>
> **窗口自适应 : 为块级元素的宽度与高度设置百分比值.**
> 一般地, 我们需要为整个 html 文档添加一个高度, **html,body{height:100%}**

17.2最值属性

> 在 html 页面布局中, 通常有四个 CSS 属性允许设置元素的最大、最小高度与宽度:
>
> > **max-width : 设置元素的最大宽度.**
> > 元素的宽度将不允许超出这个值

> **min-width : 设置元素的最小宽度.**
> 元素的宽度将不允许低于这个值
>
> **max-height : 设置元素的最大高度.**
> 元素的高度将不允许超出这个值
>
> **min-height : 设置元素的最小高度.**
> 元素的高度将不允许低于这个值

17.3动态计算宽度与高度

> 又叫做**占剩余宽高度的所有**
> 使用 calc() 函数实现动态计算元素的宽高
> 通用格式为 **width / height : calc(parameter1 - parameter2), 参数之间可以进行四则运算, 且运算符前后有一个空格符, 参数可以接单位**
>
> 利用动态计算宽度与高度可以实现页面的 **两栏布局** 与 **三栏布局** 等结构
> 同样的布局也可以使用 **自适应** 来实现

18.伪元素/伪类选择器

> **::first-line 选择标签中文本的首行.**
>
> **::first-letter 选择标签中文本的首字符.**
>
> **::before {content : " "} 向标签文本中首字符前插入内容.**
> 被追加的内容是行内元素, 并且被添加的内容无法被选中
>
> **::after {content : " "} 向标签文本中末尾字符之后插入内容**
> 被追加的内容是行内元素, 并且被添加的内容无法被选中

19.高度塌陷问题

> 当父元素没有设置 height 属性时, 父元素中的 height 会自适应其内部的子元素高度. 当有子元素被设置为 **float** 属性或者 **固定定位与绝对定位** 属性时, 这些子元素不会再参与父元素的高度计算, 由此父元素的高度会降低, 这样的效果被称作高度塌陷. 它会影响页面的布局, 造成父元素中的子元素以及父元素之后的所有元素布局改变. 这里有以下几种方法可以解决高度塌陷:
>
> > **有元素未设置浮动属性或者固定定位与绝对定位属性时**

1. 为父元素设置未塌陷时的总高度
2. 为父元素中的其他子元素设置 clear : both 属性

所有子元素均设置了浮动属性或者固定定位与绝对定位属性时

1. 为父元素添加 overflow :hidden 属性
2. 在父元素中添加一个空的(块级)元素, 为该元素设置 float : both 属性

万能清除法

为父元素设置一系列属性

> ::after{content : " "; display : block; clear : both;} 如果上述 content 属性值有内容, 还需要增加 {width : 0px; height : 0px; font-size : 0px; overflow : hidden; (visivility : hidden; opacity : 0)}