day01

一.CSS概述

1.什么是 CSS

Cascading Style Sheets

层叠样式表，级联样式表，简称样式表

2.作用

设置HTML网页中的元素样式

3.CSS和HTML的关系

HTML：负责网页结构的搭建，内容的展示

CSS：负责网页的修饰

4.CSS与HTML的属性

W3C建议我们尽量使用css的方式来取代html

css样式

1.样式代码高度的重用

2.提高可维护性

二.CSS的语法规范

1.使用CSS的方法

常用的css属性

color：设置文本的颜色

取值：合法的颜色值(颜色的英文，#a-f\*3 #abc)

font-size：设置字体大小

background：设置背景颜色

1.内联样式

将CSS的样式定义在HTML标签的style属性中

<any style = "样式声明;样式声明"></any>

样式声明 样式属性:值 组成

多个样式声明用分号隔开

内联样式使用的非常少(禁止使用)

内联样式不能重用

内联样式优先级最高

项目中一般很少使用内联样式，只在学习和测试的时候使用

2.内部样式

在<head></head>标签中，添加<style></style>标签，在其内部定义样式规则

<style>

样式规则

</style>

样式规则：有选择器和样式声明组成

选择器{

样式声明;

样式声明;

}

内部样式在本页面可以重用，但是在其他页面不能重用。项目中使用不多，但是学习和测试中大量使用

3.外部样式

独立创建一个新文件\*.css

可以在任何网页中引用

<head>

<link rel="stylesheet" href="./my.css">

</head>

外部样式是项目中必须使用的样式写法

2.CSS样式的特性

1.继承性

大部分的CSS效果是可以直接被子元素继承的

必须是层级结构

2.层叠性

可以为一个元素定义多个样式规则

规则中属性不冲突(不重复)，可以同时作用到当前元素上

3.优先级

如果样式声明冲突时，按照样式规则优先级去应用

默认的优先级，从高到低

1.内联样式(行内样式)优先级最高

2.内部样式、外部样式，遵循就近原则

3.浏览器默认样式最低

4.调整优先级

p{

color: blue !important;

font-size: 24px

}

放在属性值之后，与值之间用空格隔开

作用是调整优先级，让没有important属性不能覆盖他

三.基础选择器

1.选择器的作用

规范了页面中那些元素能够使用定义好的样式，为了匹配元素

2.基础选择器详解

1.通用选择器

\*{样式声明}

\*效率极低，尽量少用

\*{margin:0;padding:0}外边距、内边距清零

2.元素选择器(标签选择器)

设置页面中一类元素的默认样式

标签名称(样式声明)

p{color:red}

3.类选择器

定义页面上某类元素的公共样式，谁想用我，就引用我的class名

元素通过class属性来引用该样式

.类名{样式的声明}

<any class="类名"></any>

注意："."不能省略

类名不能以数字开头

类名不能包含特殊字符(\_可以)

类选择器的特殊用法

1.多类选择器

让一个元素引用多个类选择器

<p class="font\_blue font\_size">烤生蚝</p>

2.分类选择器

将元素选择器和类选择器联合使用

元素选择器.类选择器{}

可以匹配到，指向了某个类的，具体某个元素

span.font\_blue{

background: #afa

}

<span class="font\_blue">水煮牛肉</span>

作用

1.指向更精确

2.优先级变高

4.id选择器

设置指定元素的样式，专属定制

为元素设置id值，使用#id值作为选择器，编写专属样式

5.群组选择器

将多个选择器放在一起，一块定义某公共样式

选择器1，选择器2，选择器3...{公共样式声明}

#content,p.mycolor,span{color: red}

6.后代选择器

通过元素的后代关系匹配元素

后代：一层或多层的嵌套关系

语法：选择器1 选择器2 选择器3{样式声明}

7.子代选择器

通过元素的子代关系匹配元素(一层嵌套)

选择器1>选择器2>选择器3

8.伪类选择器

匹配元素的不同状态下的样式

1.链接伪类

:link 元素尚未访问时的状态样式

:visited 元素访问过后的状态样式

#a1:link{color: red}

#a1:visited{color: yellow}

2.动态伪类

:active 匹配元素被激活是的状态

#btn:active{color:pink}

:focus 匹配元素获取焦点时的状态

#txt:focus{background: #aff}

:hover 鼠标悬停在元素时的状态

.img:hover{width: 500px}

9.选择器的权值

权值：标识当前选择器的重要程度，权值越大优先级越高

!important >1000

内联样式 1000

id选择器 100

类和伪类选择器 10

元素选择器 1

\*选择器 0

继承的样式无权值 无

总结：

当一个选择器中含有多个选择器时，需要将所有的选择器的权值进行相加，然后再进行比较，权值高的优先显示

注意：

1.选择器的权值计算不会超过其最大的数量级

2.群组选择器的权值，单独计算，不会相加

3.样式后面加!important该样式会获取最高优先级

内联样式不能添加!important

4.如果权值相同，使用就近原则

四.尺寸和边框

1.尺寸属性

改变元素的宽高

width

最大宽度max-width

最小宽度min-width

height

最大高度max-height

最小高度min-height

总结：页面中可以设置尺寸的元素

1.所有的块级元素都允许设置尺寸

2.所有的行内块元素都允许设置 表单元素(除了单选按钮和复选框)

3.本身具备width、height属性的元素

大部分的行内元素不允许修改尺寸

附加知识点：单位

1.尺寸单位

px像素

in英寸 1in=2.54cm

pt磅值，多数表示字体大小 1pt=1/72in

cm mm 厘米，毫米

em相对单位，相对父元素的倍数

rem相对单位，相对根元素的倍数

%相对单位，相对父元素的百分之多少

2.颜色单位

1.颜色的英文单词

2.#rrggbb RGB光学三原色 6个16进制

3.#aabbcc ---> #abc

4.rgb(r,g,b)

5.rgb(r%,g%,b%) 此变现形式使用极少

6.rgba(r,g,b,alpha) alpha透明度

day02

一.尺寸和边框

1尺寸属性

2.溢出处理

overflow overflow-x overflow-y

当内容多，元素区域小的时候，就会产生溢出效果

默认是纵向溢出

取值

1.visible 可见的，默认值，溢出可见

2.hidden 隐藏的，溢出的内容不可见

3.scroll 显示滚动条，不管是否溢出，都添加横向和纵向滚动条

4.auto 溢出的部分，显示滚动条，并可用

3.边框属性

border:width style color

width 边框的宽度

style 边框的样式

取值：solid 实线

dotted 虚线(点点)

dashed 虚线(短线)

double 双实线

color 边框的颜色，合法的颜色值(transparent 透明色)

这种写法，同时设置4个边框，是一种简写方式

取消边框 border:none/0

单边设置

border-top: 2px solid orange;

border-right: 4px dashed #f00;

border-bottom: 5px dotted #0ff;

border-left: 3px double #f0f;

单属性设置

border-color: #000;

单边单属性设置

border-left: 3px double #f0f;

border-top-style: double;

border-top-color: #44f

边框的倒角(圆角)

将直角倒成圆角

取值：以px为单位的数字

% 设置圆形(50%)

单角设置

使用两条边去确认一个叫，先写上下后写左右

border-top-left: 左上

border-top-right: 右上

border-bottom-left: 左下

border-bottom-right: 右下

边框阴影

box-shadow: h-shadow v-shadow blur spread color inset

h-shadow 水平方向阴影偏移，必须值

v-shadow 垂直方向偏移量，必须值

以下是可选参数

blur 阴影的模糊距离，可选值，越大模糊距离越明显

spread 阴影的尺寸，指定要在基础阴影上扩出的大小

color 阴影颜色

inset 把默认的外部阴影设置为内部阴影

轮廓

轮廓指的是边框的边框，绘制于边框外边的线条

outline: width style color;

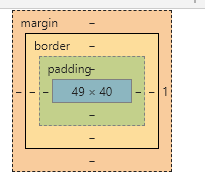
二.框模型(盒子模型)

页面元素皆为框(盒子)

定义了元素框处理元素内容，内边距，外边距以及边框的一种计算模式

定义了一个元素占地大小的计算方式

盒子模型默认的计算方式：



元素实际占地宽度=左外边距+左边框的宽度+左内边距+内容区域的宽度+右内边距+右边框的宽度+右外边距

元素实际占地高度=上外+上边框+上内+内容区域高度+下内+下边框+下外

内边距：元素边框与内容之间的空白间距

外边距：围绕在元素边框外的空白区域(元素与元素之间的距离)

1.外边距

margin: 定义四个方向的外边距

margin-top/right/bottom/left: 定义某一个方向的外边距

1.取值

1.以px为单位的数字

为正数 top，元素往下移动

left，元素往右移动

为负数 top，元素往上移动

left，元素往左移动

margin的效果，改变元素在页面上的位置，多用于微调

2.取值为%，相对于父元素尺寸的百分比

3.auto，自动计算块级元素的外边距，控制块级元素在水平方向居中对齐

2.简写方式

margin:value; 设置四个方向的外边距

margin:v1 v2; v1：上下外边距 v2：左右外边距

margin: 0 auto; 标准的块级元素在水平方向居中对齐

margin:v1 v2 v3; v1：上外边距 v2：左右外边距 v3：下外边距

margin:v1 v2 v3 v4; v1：上外边距 v2：右外边距 v3：下外边距 v4：左外边距

3.自带外边距的元素有哪些

h1~h6,p,body,ol,ul,dl,pre

通常要清除这些元素的自带外边距\*{margin:0;padding:0}

4.外边距的特殊效果

1.外边距的合并

当两个垂直外边距相遇时，他们将合并成一个外边距

最终的取值，取决于两个外边距中距离大的值

2.行内元素以及行内块元素的外边距

行内元素垂直外边距无效(img除外)，水平外边距相遇两个值为相加

行内块，设置垂直外边距，整行元素跟着发生变化

3.外边距溢出

在特殊条件下，给子元素设置的上外边距，会作用到父元素上

特殊条件：1.父元素没有上边框。2.只有第一个子元素设置上边框时

解决方案：

1.为父元素增加上边框，弊端：元素实际占地高度变大

2.为父元素添加上内边距，弊端：元素实际占地高度变大

3.在父元素的第一个子元素位置添加一个空的<table></table>

2.内边距

不会影响其他元素，但是会改变元素自己的占地尺寸，会变大

padding:value; 设置四个方向的内边距

padding-top/right/bottom/left 定义单方向的内边距

取值

以px为单位的数字

%

简写方式

padding:value; 设置四个方向的内边距

padding:v1 v2; v1;上下 v2左右 padding没有auto

padding:v1 v2 v3; v1上 v2左右 v3下

padding:v1 v2 v3 v4; 上 右 下 左

3.box-sizing属性

设置盒子模型的计算方式

默认计算方式

元素实际占地宽度=左外+左边框+左内+内容区域宽度+右内+右边框+右外

元素实际占地高度=上外+上边框+上内+内容区域高度+下内+下边框+下外

box-sizing取值

content-box默认值，会采用默认计算方式

border-box元素的尺寸(定义的宽高)，包含border,padding和内容区域宽高

元素实际占地宽度=左右外边距+设置的宽度width

元素实际占地高度=上下外边距+设置的高度height

块级元素，行内元素，行内块

块级元素：独立成行，可以设置宽高，margin上下有效

行内元素：与其他的行内元素和行内块共用一行，宽高无效，margin上下无效

行内块：与其他的行内元素和行内块共用一行，可以设置宽高，margin上下有效，一行行内元素，其中一个设置了上下margin，整行都跟着移动

day3

一.背景

1.背景颜色

background-color

取值

合法的颜色值

transparent 透明

注意：背景色的填充，默认从边框位置处开始

2.背景图片

background-image: url("资源路径")

3.背景图片的平铺

background-repeat:

取值

1.repeat 默认值，平铺

2.repeat-x 水平方向平铺

3.repeat-y 垂直方向平铺

4.no-repeat 不平铺，只显示一张

4.背景图的定位

background-position: x y;

取值

1.以px为单位的数字

指的是，图片在水平和垂直方向的偏移量

2.x% y%

0% 0% 左上

100% 100% 右下

3.关键字

x: left/center/right

y: top/center/bottom

5.背景图的尺寸

background-size: width height

取值

1.以px为单位的数字

2.%

3.cover 填满容器，图片宽高比是不变的，铺满整个容器的宽高，图片溢出的部分会被裁剪掉；将背景图同比缩放，知道背景图完整覆盖容器，不留一丝空白，图片溢出的部分会被裁剪掉；图片可能会显示不完整，但是内容完全被背景图充满

4.contain 图片自身宽高比不变，但是，要缩放到图片能够完全展示出来，所以会有空白区域

6.背景图的固定

background-attachment:

将背景图固定在网页某个位置，一直在可是的区域中显示，不会随着网页滚动条改变位置

取值：

1.scroll 默认值，会随着网页滚动条改变位置

2.fixed 固定，不会随着网页滚动条改变位置

7.背景的简写方式

在一个属性中，指定背景多个值

属性：background:

取值：color url repeat attachment position

二.渐变gradient

1.什么是渐变

多钟颜色平缓变化的一种显示效果

2.渐变的主要因素---色标

色标，一种颜色以及他出现的位置

一个渐变至少有两个色标

3.渐变的分类

1.线性渐变，以直线的方向来填充渐变色

2.径向渐变，以圆形的方式实现渐变色

3.重复渐变，将线性渐变，或径向渐变，重复几次

4.线性渐变

background-image:

取值

新版本：linear-gradient(方向,色标1,色标2...)

色标的写法：颜色的合法值 位置的百分比

方向取值：to top/right/bottom/left

角度 按顺时针计算

0deg to top

90deg to right

180deg to bottom

270deg to left

旧版本写法：-浏览器内核-linear-gradient(方向,色标1，色标2...)

方向 top/right/bottom/left

5.径向渐变

background-image: radial-gradient(半径at 圆心,色标1,色标2...)

半径：以px为单位的数字

圆心：x y，以px为单位的具体数值

x% y%，元素的宽高占比

关键字 x:left/center/right

y:top/center/bottom

6.重复线性渐变

repeating-linear-gradient(to right,#faa 0px,#ffa 50px)

注意在重复渐变中，色标的位置，要给绝对值数字，不要用%

7.重复径向渐变

repeating-radial-gradient(100px at center center,#060606 0px,#af8bd8 20px,#292727 35px)

8.浏览器兼容性

各个浏览器新版本都支持渐变

对于不支持的旧版本，可以添加浏览器内核前缀的方式，做兼容，让浏览器支持

Chrome&Safari：-webkit-

Firefox：-moz-

IE：-ms-

Opera：-o-

三.文本格式化

1.字体属性

1.指定字体的类型

font-family:"类型1","类型2"...

2.字体大小

font-size

以px、pt为单位的数字

em rem

3.字体加粗

font-weight:

bold/normal/lighter/bolder

无单位的数字 100-1000整百数 常用400-900

4.字体样式

font-style

normal

italic

5.小型大写字母

font-variant:

normal

small-caps

6.字体属性的简写方式

font:style variant weight size family

font: italic small-caps bold 30px "mv boli"

简写方式最少要有size和family

2.文本格式化

1.文本颜色

color:合法的颜色值

2.文本的水平对齐方式

text-align

left/center/right/justify(两端对齐)

3.行高

定义一行数据的高度

如果行高大于字体本身的大小，该行文本在行高内呈垂直居中的显示效果

line-height:

以px为单位的数字，一般与容器的高相同

无单位数字，line-height:2; 是字体大小的倍数

4.线条的修饰

text-decoration

1.none 没有线条，去掉a标签的下划线

2.overline 上划线

3.underline 下划线

4.line-through 删除线

5.首行缩进

text-indent:

以px为单位的数字

6.文本的阴影

text-shadow:

h-shadow v-shadow blur color

day4

一.表格的样式

1.表格的常用属性

之前学习的样式，在表格中，基本都可以使用

1.边框属性table和td

2.边距属性table都能使用，td没有margin

3.尺寸属性

4.文本格式属性font-\* text-\* line-height

5.背景属性 颜色 图片 渐变

6.vertical-align: top/middle/bottom 要写在td才生效

2.表格特有的属性

1.边框合并

border-collapse:

1.separate 默认，边框分离的模式

2.collapse 边框合并

2.边框的边距

设置单元格与单元格之间的距离，必须保证border-collapse: separate;

border-spacing:

1.取值一个value，垂直和水平距离相同

2.取值两个value1 value2 value1：水平的距离 value2：垂直距离

3.标题的位置

caption-side

top/bottom

4.表格的显示规则，告诉浏览器，我这张表格如何渲染，td尺寸的计算方式

table-layout

1.auto 默认值，自动表格布局，列的尺寸实际是有内容决定的

2.fixed 固定表格的布局，列的尺寸由设置的值决定

需要table有尺寸，td有尺寸

|  |  |
| --- | --- |
| 自动布局auto | 固定布局fixed |
| 单元格的大小会适应内容 | 单元格尺寸取决于设定的值 |
| 表格复杂时，加载的速度慢(缺点) | 任何情况下回加速加载表格(优点) |
| 布局会比较灵活(优点) | 布局会比较死板(缺点) |
| 适用于不确定每列大小，并且表格不太繁琐的时候 | 当确定每列尺寸的时候使用 |

二.定位

1.什么是定位

元素在页面中的位置

2.分类

1.普通流定位 默认文档流

2.浮动定位

3.相对定位

4.绝对定位

5.固定定位

3.普通流定位

又称文档流定位，页面中所有元素默认显示方式

1.每个元素在页面中都有自己的空间(盒子模型)

2.每个元素都是从父元素的左上角开始显示的

3.块级元素默认按照从上往下的方式逐个排列，每个元素独占一行

4.行内元素是多个元素在一行中显示，从左往右排列

4.浮动

浮动特点

1.元素一旦浮动起了，脱离文档流

不占页面控件，其他为浮动元素会上前补位

2.浮动元素会停靠在父元素的左边或者右边

或其他已经浮动元素的左/右边

3.浮动解决多个块级元素在同一行内显示的问题

语法

属性 float

取值

left：左浮动，元素浮动起来后，停靠在父元素的左侧，或者往左挨着已浮动元素

right：右浮动，元素浮动起来后，停靠在父元素的右侧，或者往右挨着已浮动元素

浮动元素引发的特殊情况

1.当父容器横向显示不下所有浮动元素的时候，最后一个元素将换行显示

1.1默认情况下，最后一个元素会优先显示在最高的位置，再往左排列

1.2会发生浮动元素占位的情况，浮动元素根据浮动方向，占据方向之上的位置，不允许最后一个元素占用

2.浮动对默认宽度的影响

块级元素不写宽度，宽度默认是父级元素的100%

元素一旦浮动起来，宽度以内容为准

前提：不设置width

3.元素一旦浮动起来，就变为块级元素，允许修改尺寸，可以设置垂直外边距

4.文本，行内元素，行内块，是不会被浮动元素压在下方的，而会巧妙避开，环绕着浮动元素显示

清除浮动

元素一旦浮动之后，会对后续的元素带来一定的影响，后续元素会上前补位

如果不希望后续元素上前补位，可以给后续元素添加清除浮动的属性

clear:

取值

left 清除左浮动元素对我带来的影响

right 清除右浮动元素对我带来的影响

both 清除所有浮动元素对我带来的影响

浮动元素对父元素高度带来的影响(高度坍塌)

块级元素不写高度，他的高度是根据内容判定的

如果内部元素都浮动了，内部元素都脱离了默认文档流

父级元素就失去了高度----高度坍塌

解决方案

1.给父级元素添加高度 弊端：不是每次都知道具体的高度是多少

2.设置父级元素也浮动 弊端：会影响父元素后面的其他元素

3.给父元素设置overflow:hidden/auto 弊端：如果内容想要溢出显示，显示不了

4.在父元素追加一个空的块级元素，只设置clear:both

三.显示

1.显示方式

决定了元素在网页中表现形式(块级，行内，行内块，table)

语法 display

取值：

1.none 不显示元素，隐藏

2.block 让元素表现为块级

块级元素特征，独占一行，可以设置尺寸以及上下外边距

3.inline 让元素表现为行内

行内元素特征，多个元素一行显示，不能设置宽高以及上下外边距

4.inline-block 让元素表现为行内块

行内块特征，多个元素一行显示，可以设置尺寸以及上下外边距

5.table 让元素表现为table

table的特征：尺寸以内容为准，独占一行，允许修改尺寸

2.显示的效果

属性：visibility

取值：visible/hidden

visibility: hidden和display: none的区别

visibility: hidden隐藏，不脱离默认文档流，依然占位

display: none隐藏，脱离了默认文档，不占页面位置

3.透明度

opacity:0~1 0是全透明，1是不透明

opacity和rgba的区别

opacity作用于元素，当一个元素设置了opacity之后

这个元素本身，已经所有子元素，与颜色相关的属性，都会受到影响

rgba(0~255,0~255,0~255,0~1)只会改变你设置的这个颜色透明度

4.垂直对齐

vertical-align

使用在table中，取值top/middle/bottom

使用在img中，取值 top/middle/bottom/baseline

控制图片与两边文字垂直对齐方式

5.光标

改变鼠标悬停在元素是的样式

cursor:

取值

1.default 箭头

2.pointer 小手

3.crosshair +

4.text I

5.wait 等待

6.help 帮助

四.列表

列表标识

list-style-type:

none

disc

circle

square

列表标识的位置

list-style-position: outside/inside

列表项引用图片

list-style-image: url("")

简写方式

list-style: type position url()

day5

一.相对/绝对/固定定位 position

position

取值

static 静态，默认值。文档流定位

relative 相对定位

absolute 绝对定位

fixed 固定定位

注意：将元素的position设置为relative/absolute/fixed中的任意一个，此元素被称为**已定位元素**。

偏移属性

top/right/bottom/left 距离那个方向多远

取值是以px为单位的数字

1.相对定位

元素相对于它原来的位置偏移

在元素位置作微调的时候使用/配合绝对定位使用

position: relative;配合偏移属性使用

相对定位不脱离文档流，不会让后面元素补位

2.绝对定位

position: absolute; 配合偏移属性使用

特点

脱离文档流，不占页面空间，后面元素要补位

绝对定位的元素，会相对于"最近的" "已定位的" "祖先元素"去做偏移量

绝对定位的元素，没有"已定位的""祖先元素"，相对于body偏移

绝对定位会让元素变为块级

出现堆叠效果，就使用绝对定位

3.固定定位

将元素固定在页面的某个位置，位置不会随着滚动条发生变化，一直固定在可是区域

固定定位始终相对于body去实现位置偏移

固定定位脱离文档流，不占页面空间，定位的元素会变成块级

position: fixed; 配合偏移属性一起使用

4.堆叠顺序

默认堆叠顺序，后发生定位的元素，堆叠层级高

默认堆叠顺序，不会大于设置属性z-index的顺序

z-index: 2^31~1

父子元素之间，子元素永远在父元素上面

只有已定位元素，才能使用z-index

二.css3 core

1.复杂选择器

1.兄弟选择器

作用于兄弟元素

兄弟元素：具备相同父级元素的平级元素之间，称为兄弟元素

1.1相邻兄弟选择器

选择器1+选择器2{} 在选择器1的后面，紧跟着一个元素，这个元素还得匹配到选择器2

1.2通用兄弟选择器

获取某元素后面所有满足条件的兄弟元素

选择器1~选择器2{}

2.属性选择器

允许通过元素所带的属性以及值来匹配元素

2.1基本语法

[attr]{} ex:[id]{} 匹配有id属性的标签

2.2elem[attr]

div[title]{} 匹配有title属性的div标签

2.3[attr1][attr2]

[class][title]{} 匹配同时具有class属性和title属性的元素

2.4[attr=value]{}

[id=p2]{} 匹配id值为p2的元素

[attr\*=value] 属性值中，包含指定值的元素

[attr~=value] 属性值中，包含指定**单词**的元素

[attr^=value] 属性值中，以value开头的元素

[attr$=value] 属性值中，以value结尾的元素

3.伪类选择器

3.1目标伪类

突出显示获得的HTML锚点元素，匹配被激活的锚点

选择器:target{} div:target{}

3.2结构伪类

1.:first-child{} 匹配的元素是其父元素的第一个子元素

2.:last-child{} 匹配的元素是其父元素的最后一个元素

3.:nth-child(n){} 匹配的元素是其父元素的第n个子元素，n从1开始

4.:empty

div:empty{} 匹配没有子元素的元素

5.:only-child

p:only-child{} 匹配属于其父元素的唯一子元素

6.否定伪类 :not(selector)

div:not(:empty){} 匹配非空的div

3.3伪元素选择器

h2::first-letter{color:#f00;}

匹配某个元素的首字字符

h2::first-line{color:#aaf;}

匹配某个元素的首行字符，首行与首字符冲突，优先首字符

h2::selection{color:#ffa;}

匹配用户选取的内容，只能修改文本颜色和背景颜色

4.内容生成

使用css动态的向元素中插入一段内容

1.::before或者:before

匹配到元素的内容区域之前，插入的伪元素

2.::after或者:after

匹配元素的内容区域之后，插入的伪元素

5.内容生成可以解决的问题

1.外边距溢出

父元素没有上边框，给第一个子元素设置上外边距

day6

一.弹性布局

1.什么是弹性布局

弹性布局就是一种布局方式，主要解决**某元素中的子元素**的布局方式，为布局提供最大的灵活性

2.弹性布局的相关概念和专业术语

1.容器

要布局的子元素的父元素称之为容器，容器中写display: flex

2.项目

要布局的子元素称之为项目

3.主轴

项目们排列的方向，诚挚为主轴(水平和垂直)

如果项目们是按照横向排列，x轴就是主轴

如果项目们是按照纵向排列，y轴就是主轴

4.交叉轴

与主轴垂直相交的方向轴叫做交叉轴

语法

将元素变为弹性容器，他所有的子元素将变成弹性项目，按照弹性布局的方式去排列显示

display

取值：flex，将块级元素变为容器

inline-flex，将行内元素变为容器

元素设置为flex容器之后，子元素一些样式属性会失效

float/clear/vertical-align失效

子元素允许修改尺寸(项目是行内元素也可以修改尺寸)

容器的对齐方式text-align失效

3.弹性容器的样式属性

1.flex-direction主轴方向

取值：

1.row默认值，主轴是x轴，主轴起点在左端

2.row-reverse，主轴是x轴，主轴起点在右端

3.column 主轴是y轴，主轴起点在顶端

4.column-reverse 主轴是y轴，主轴起点在底部

2.flex-wrap

当一个主轴排列不下所有项目时，项目的显示方式

取值：

1.nowrap默认值，空间不够，不换行，项目会自动压缩

2.wrap空间不够，就换行，项目不压缩

3.wrap-reverse

3.flex-wrap

是flex-direction和flex-wrap的缩写

取值 direction wrap;

ex:flex-flow:row wrap;

4.justify-content

定义项目在主轴上的对齐方式

取值

1.space-between 两端对齐

2.space-around 每个间距相同

3.flex-start 默认值，主轴的起点对齐

4.flex-end 在主轴的终点对齐

5.center 在主轴上居中对齐

5.align-items

项目们的交叉轴上的对齐方式

取值

1.flex-start 交叉轴起点对齐

2.flex-end 交叉轴终点对齐

3.center 交叉轴居中对齐

4.baseline 交叉轴基线对齐

5.stretch 如果项目未设置高度，在交叉轴上充满容器

4.项目的属性

是单独设置给一个项目的，不影响容器和其他项目

1.order

取值为无单位的整数，定义项目的排列顺序，值越小，离起点越近，默认值为0

2.flex-grow

定义项目的放大比例

如果容器有足够大的剩余空间，项目将按照比例放大

取值为无单位整数

默认值为0不放大，取值越大，占据的剩余空间越多

3.flex-shrink

定义项目的缩小比例，空间不足时，项目该如何缩小

取值为无单位整数

默认值为1，空间不足，等比缩小

0，不缩小

取值越大，占据的空间越小

4.align-self

控制当前项目在交叉轴上对齐方式，与其他项目无关

取值

1.flex-start 交叉轴起点对齐

2.flex-end 交叉轴终点对齐

3.center 交叉轴居中对齐

4.baseline 交叉轴基线对齐

5.stretch 如果项目未设置高度，在交叉轴上充满容器

6.auto 继承容器的align-items的效果

CSS hack

由于不同的浏览器，比如IE6 IE7 Firefox

对css的解析认识不同，会导致生成的页面效果不同，无法得到我们预期的效果

这个时候我们需要针对不同浏览器去写不同的css

让代码同时兼容不同的浏览器

这个针对不同浏览器写不同css代码的过程，叫做CSS hack

二.转换

1.什么是转换

改变元素在页面中的 位置 大小 角度 形状等的一种方式

2D转换，在x轴和y轴上，发生转换效果

3D转换，添加了z轴

2.转换属性

transform:

取值 transform-function

none

如果有多个转换函数，用空格隔开

转换原点

transform-origin

取值：px为单位的数字/%/关键字

2个值，表示原点在x轴和y轴上的位置

3个值，x，y，z轴

3.2D转换函数

1.位移，改变元素在页面中的位置

transform: translate(x)指定元素在x轴上的位移距离

+ 元素往右

- 元素往左

transform: translate(x, y)指定元素在x和y轴上的位移距离

x取值同上

y取值 + 元素往下 - 元素往上

transform: translateX(x)指定元素在x轴上的位移距离

transform: translateY(y)指定元素在y轴上的位移距离

2.缩放

改变元素在页面中的尺寸

transform: scale(n)

取值：无单位的数字

>1 根据转换原点放大

0<n<1 根据转换原点缩小

-1<n<0 翻转缩小(水平和垂直方向都翻转了180度)

n<-1 翻转放大

transform: scale(x, y) x和y缩放尺寸

transform: scaleX(x)单独设置x的缩放

transform: scaleY(y)单独设置y的缩放

3.旋转

改变元素在页面中的角度

transform: rotate(ndeg)

n为正 顺时针

n为负 逆时针

注意：

转换原点会影响最后的旋转效果

选择是连同坐标轴一起旋转的，会影响旋转后的位移效果

4.倾斜

改变元素在页面中的形状

transform: skew(x)等同于skewX(x)

让元素向着x轴发生倾斜，实际上改变y轴的角度

+ 逆时针

- 顺时针

transform: skewY(y)

让元素向着y轴发生倾斜，实际上改变x轴的角度

+ 顺时针

- 逆时针

transform: skew(x,y)

4.3D转换

浏览器不支持3D的位移，z轴看不到，只能模拟

1.透视距离

模拟人的眼睛到3D转换元素之间的距离

perspective

该属性要加载转换元素的父元素上

2.3D旋转

transform: rotate3d(0,0,1,20deg)

取值

1.rotateX(xdeg)

以x轴为中心轴，旋转元素的角度

2.rotateY(ydeg)

以y轴为中心轴，旋转元素的角度

3.rotateZ(zdeg)

以z轴为中心轴，旋转元素的角度

4. rotate3d(z,y,z,20deg)

x,y,z取值大于0的数字是，表示该轴参与旋转

取值为0，表示不参与旋转

三.过渡

CSS属性值，在一段时间内平缓的变化

transition-property:background/all

能够使用过渡效果的属性

1.颜色

2.取值为数字的属性

3.转换

4.阴影

5.渐变

6.visibility

指定过渡持续时间

transition-duration: 1s/1000ms

简写方式

transition: all 1s;

transition: 1s;

不要写在hover中，写在元素本身的样式里

day7

一.过渡

某元素的css属性值在一段时间内，平滑改变到另外一个值

过渡主要是看的过程和结果

设置能过渡的属性

支持过渡效果的样式属性

颜色的属性

取值为数值

transform

渐变属性

visibility

阴影

指定本次过渡生效的属性

transition-property:上面的css属性/all

指定过渡的时长

transition-duration:过渡的时长 单位/s/ms

指定过渡时间曲线函数

transition-timing-function:

1.ease 默认值，慢--->快--->慢 慢速开始，快速变快，慢速结束

2.linear 匀速

3.ease-in 慢--->快 慢速开始，快速结束

4.ease-out 快速开始，慢速结束

5.ease-in-out 慢速开始，先加速在减速，慢速结束

指定延迟执行过渡的时间

transition-delay: s/ms

过渡属性的编写位置

1.将过渡放在元素声明的样式中(元素自己的样式里)，过渡效果有去有回

2.将过渡放在元素的触发操作中(hover)，过渡效果有去无回

过渡的简写

transition: property duration timing-function delay;

二.动画

使元素从一种样式，改变到另外一种，在改变到其他样式......

相当于将很多个过渡效果放在一起使用

关键帧

1.动画的执行时间点

2.该时间点上的样式

动画的实现步骤

1.声明动画及动画关键帧

@keyframes 动画名称{

//定义关键帧

0%{动画开始时的样式}

......

100%{动画结束时的样式}

}

2.调用动画

animation-name: 动画名称;

animation-duration: 动画播放一个周期的时间;

3.动画的其他属性

animation-delay:

4.动画的速度时间曲线函数

animation-timing-function:

ease/linear/ease-in/ease-out/ease-in-out

5.animation-iteration-count:

指定动画的播放次数

取值，具体的字数/infinite 无限次

6.animation-direction:

动画的播放方向

取值 normal 正常 0%~100%

reverse 逆向 100%~0%

alternate 轮流播放

奇数次正向播放

偶数次逆向播放

7.简写方式

animation: name duration timing-function delay iteration-count direction;

8.animation-fill-mode:

指定动画播放前后的显示状态

1.none 默认值

2.forwards 动画完成后，保持在最后一个关键帧上

3.backwards (需要有delay)动画开始之前，保持在第一个关键帧上

4.both 同时设置forwards和backwards

动画的兼容性

如果要兼容低版本浏览器，需要在声明动画的时候加前缀

@keyframes 动画名称{}

@-webkit-keyframes

@-moz-keyframes

@-ms-keyframes

@-o-keyframes

三.css优化

目的：减少服务器压力，提升用户体验

1.优化原则

尽量减少HTTP请求的个数

页面顶部引入css文件

将css和js放到外部独立的文件夹中

2.css代码优化

缩小样式文件

减少样式的重写

避免出现空的src和href

选择更优的样式属性值(能使用符合，简写的写法，就不要单独定义)

代码压缩

二.BootStrap

www.bootcss.com

1.响应式布局

responsive web page 响应式/自适应的网页

可以根据浏览器的设备不同(pc/pad/phone)自动调用对应的布局，图片，文字效果，从而不会降低用户体验

2.响应式网页必须做到的前提

**1.布局：不能固定宽度，必须是流式布局(尽量少用定位，可以浮动)**

**2.文字和图片，大小随着容器大小而改变 em rem**

**3.媒体查询技术**

响应式网页存在的问题：

页面的复杂度极大的增加

只适用于内容不太多的页面(企业的官网，门户网站)

媒体查询技术属于h5/c3的技术

boot把媒体查询这件事，封装了，不需要我们自己写了

3.如何测试响应式网页

1.使用真实设备测试

优点：真实，可靠

缺点：测试任务量巨大

2.使用第三方的模拟测试软件

优点：无需添置太多真实设备，测试方便

缺点：测试效果有限，有待进一步验证

3.使用浏览器自带的模拟器测试

优点：简单，方便

缺点：测试效果有限，需要进一步验证

4.如何编写响应式布局

1.在元数据标签中定义viewport---视口

name="viewport"

content 设置能够允许网页进行操作

width=device-width表示视口宽度就是设备宽度

initial-scale=1.0表示视口宽度师傅可以缩放 1.0不能缩放

maximum-scale=1.0允许缩放的最大倍率

user-scalable=0是否允许用户手动缩放 1/0 yes/no

一般的设置：

<meta name="viewport" content="width=device-width,initial-scale=1">

2.所有内容/文字/图片，使用相对尺寸，不能使用10px这种绝对的值

3.流式布局+弹性布局，再搭配媒体查询技术来完成响应式布局

float flex

4.使用CSS3 Media Query技术做响应式网页

Media：媒体，指浏览网页的设备 如：screen(pc/pad/phone) tv print

Media Query媒体查询，可以自动根据当前浏览器设备的不同(尺寸、解析度、方向不同)，有选择的执行一部分CSS而忽略其他部分的CSS

<link rel="stylesheet" href="pc.css" media="screen and (min-width:992px)">

注意：此方案使用较少

所有的css文件都会被加载，每个css文件中，存在很多重复代码

根据媒体查询的结果，执行同一个css文件下的不同代码块

@media screen and (min-width:768px) and (max-width:991px){

.content{

color: #00f;

background: #faa;

font-size: 24px;

}

}

day8

一.Bootstrap

1.所有使用boot的代码都必须放在container类的包裹中

二.全局css样式

1.按钮相关的class

btn基本按钮

颜色按钮

btn-danger

btn-success

btn-warning

btn-info

btn-primary

btn-secondary

btn-dark

btn-light

btn-link

不同的边框按钮

btn-outline-颜色

按钮大小

btn-lg

btn-sm

btn-block

2.图片相关的class

rounded 倒角

rounded-circle 圆

img-thumbnail 缩略图

img-fluid 响应式布局，图片会缩放，但不会超过原始大小

3.文字相关的class

h1~h6 字体大小

字体颜色

text-danger/success/info/primary/warning...

文字对齐

text-\*-left/right/center/justify

\*代表：lg/sm 在大屏幕下对齐，在小屏幕下对齐

大小写

text-uppercase/lowercase/capitalize 大写、小写、首字母大写

粗体

font-weight-bold

斜体

font-italic

4.列表相关的class

ul

list-unstyled 去除标识符

list-group 创建列表组

li

list-group-item 创建列表项

active 激活状态

disabled 禁用状态

li的颜色

list-group-item-danger/success/......

5.table相关class

table的class

table 基本类

table-bordered 带边框的表格

table-striped 隔行变色

table-hover 鼠标悬停效果

table-danger/wraning/......

响应式表格

需要在table的父元素上，添加类table-responsive

6.辅助类

边框

border 基础类

border-top/right/bottom/left 设置4个方向的边框

border-0 border-top/right/bottom/left-0 取消边框

border-danger/warning...... 边框颜色

rounded/rounded-0 设置/取消圆角

rounded-top/right/bottom/left

浮动

float-\*-left/right \*:xl/lg/md/sm

clearfix

显示

visible/invisible 显示/隐藏

背景

bg-danger/warning......

居中对齐

m-auto

margin

m-\*(0/1/2/3/4/5)(0/0.25/0.5/1/1.5/3rem)

mt/mr/mb/ml-\*

padding

p-\*(0/1/2/3/4/5)(0/0.25/0.5/1/1.5/3rem)

pt/pr/pb/pl-\*

宽度/高度

w-\*(25/50/75/100)

h-\*同上

三.栅格布局

1.web页面布局的方法

|  |  |  |  |
| --- | --- | --- | --- |
|  | table布局 | div+css布局 | Bootstrap中的栅格布局 |
| 优点 | 简单，容易控制 | 语义正确，渲染效率高 | 简单，容易控制，语义正确，渲染率高，支持响应式 |
| 缺点 | 语义错误，渲染效率低(最后一个数据加载完成后，才能统一开始渲染) | 控制比较麻烦，媒体查询写起来更繁琐 | 页面如果太复杂，不适合使用 |

总结：

1.最外层需要使用容器

container定宽容器，最大宽度是写死的

container-fluid变宽容器，宽度是父元素的100%

2.容器中声明row，弹性布局(一行内等分为12个单元格)

3.行中声明列col-n(每个列都需要制定宽度占比n/12)