前端基础

CSS

CSS叫做层叠样式表，用来设置页面中元素的样式。背景颜色、字体颜色、字体大小。。。

CSS负责结构、表现、行为中的表现

编写的位置

1.内联样式

将样式编写到标签的style属性中

<p style="color:red;"></p>

这种样式只会对当前标签起作用，不能对样式进行复用，不方便后期维护，不推荐使用

2.内部样式表

将样式表编写到head中的style标签中

<style type="text/css"></style>

使用内部样式表，进一步将表现和结构分离，可以同时为多个元素设置样式，方便后期的维护

3.外部样式表

将样式表编写到外部的CSS文件中，然后通过link标签将外部文件引入

<link rel="stylesheet" type="text/css" href="文件的路径"/>

将样式编写到外部样式表中，可以在不同的页面中使用同一个样式表，完全将表现和结构分离，方便后期的维护，推荐使用的方式

基本语法

选择器

通过选择器可以选中页面中的一组元素，然后为其设置样式

元素选择器

根据标签名，选中页面中的指定元素

语法：标签名{ }

例子：

div{}

p{}

h1{}

id选择器

根据元素的id属性值选中一个唯一的元素

语法：#id {}

例子：

#box1{}

#hello{}

类选择器

根据元素的class属性值，选中一组元素

语法：.class{}

例子：

.hello{}

.box{}

通配选择器

选中页面中的所有元素

语法：\*{}

通配选择器的性能比较差，尽量避免使用

并集选择器

可以同时选中符合多个选择器的元素

语法：选择器1,选择器2,选择器N{}

例子：

div,p,#box,.hello{}

交集选择器

可以选中满足多个条件的元素

语法：选择器1选择器2选择器N{}

例子：p.hello{}

后代元素选择器

选中指定元素的指定后代元素

语法：祖先元素 后代元素{}

例子：

div span {}

div p{}

子元素选择器

选中指定元素的指定子元素

语法：父元素 > 子元素 {}

例子：

div > span {}

div > p{}

声明块

声明块中实际上就是一个一个CSS声明

声明

每一个CSS声明都是一个样式，实际上就是一个名值对的结构

名和值之间使用:链接

:左边是样式的名字

:右边是样式的值

每一个声明以;结尾

例子

color:red;

font-size:20px;

元素之间的关系

父元素

直接包含子元素的的元素叫做父元素

子元素

直接被父元素包含的元素叫做子元素

祖先元素

直接或间接包含后代元素的元素叫做祖先元素，父元素也是祖先元素

后代元素

直接或间接被祖先元素包含的元素叫后代元素，子元素也是后代元素

兄弟元素

拥有相同父元素的元素叫做兄弟元素

块元素和内联元素

块元素

块元素会独占页面中的一行，无论他的内容的多少

一般使用块元素对页面进行布局

常见的块元素

div

p

h1~h6

内联元素

内联元素只占用自身的大小，不会独占一行

内联元素也叫行内元素（inline）

一般内联元素都是用来为文本来设置效果

常见的内联

span

a

img

包裹规则

一般都是使用块元素去包裹内联元素，而不会使用内联去包裹块元素

a元素可以包含任意元素，除了a本身

p元素不能包含任何块元素

伪类和伪元素

伪类和伪元素用来表示元素所处的一个特殊的状态，或者是一个特殊的位置

:link

表示一个普通的链接（未访问过的链接）

:visited

表示访问过的链接

:hover

鼠标移入的链接，也可以为其他元素设置hover

:active

正在被点击的链接，也可以为其他元素设置active

:focus

表示元素获取焦点的状态，一般用于文本框

::selection

表示内容被选中的状态

在火狐中使用::-moz-selection来代替

:first-letter

表示第一个字符

:first-line

表示文字的第一行

:before

选中元素的最前边

一般该伪类都会结合content一起使用，通过content可以向指定位置添加内容

:after

选中元素的最后边

一般该伪类都会结合content一起使用，通过content可以向指定位置添加内容

属性选择器

根据元素的属性选择指定元素

[属性名]

选取含有指定属性的元素

[属性名="属性值"]

选取属性值等于指定值的元素

[属性名^="属性值"]

选取属性值以指定内容开头的元素

[属性名$="属性值"]

选取属性值以指定内容结尾的元素

[属性名\*="属性值"]

选取属性值中包含指定内容的元素

兄弟元素选择器

选取后一个兄弟元素

前一个 + 后一个

选取后边所有的兄弟元素

前一个 ~ 后边所有

子元素的伪类

:first-child

寻找父元素的第一个子元素，在所有的子元素中排序

:last-child

寻找父元素的最后一个子元素，在所有的子元素中排序

:nth-child

寻找父元素中的指定位置子元素，在所有的子元素中排序

例子

p:nth-child(3)

可以使用even，来找到偶数的子元素

可以使用odd，来找到奇数的子元素

:first-of-type

寻找指定类型中的第一个子元素

:last-of-type

寻找指定类型中的最后一个子元素

:nth-of-type

寻找指定类型中的指定子元素

否定伪类

从一组元素中将符合要求的元素剔除出去

语法：

:not(选择器)

例子：

.abc:not(div)

样式的继承

为祖先元素设置样式，会同时应用到它的后代元素上，这一特性称为样式的继承。

通过样式的继承可以将一些样式统一设置个祖先元素，这样所有的后代都会应用到相同的样式。

但是并不是所有的样式都会继承，比如：背景相关的，边框相关的，定位相关的。具体参考文档

选择器的优先级

当使用选择器为元素设置样式时，如果样式发生了冲突，采用哪个样式由选择器的优先级来决定。

优先级

内联样式

1000

id选择器

100

类和伪类选择器

10

元素选择器

1

通配选择器

0

继承的样式

没有优先级

当样式发生冲突时，需要将相关的选择器优先级进行求和计算，优先级高的优先显示，如果优先级一样，则显示靠后的样式

优先级计算时，总大小不能超过他的最大的数量级

可以在样式后边添加一个!important，如果样式中添加了该内容，则该样式将会获取最大的优先级，将会优先于所有的样式显示，包括内联样式，但是这个属性要慎用。

选择器的性能

浏览器在解析一组选择器时，是从后边往前一个一个的解析的

如果选择器编写过于长的话，浏览器解析起来性能会比较差，所以在编写选择器时，越短越好。

\*通配选择器，性能也比较差，要避免使用通配选择器

单位

长度单位

px

像素，像素就是构成一个图片的最小的单位，我们的屏幕就是由一个一个像素点构成

一个像素指的就是一个像素点

在不同的显示器中，一个像素的大小是不同的，越清晰的屏幕像素越小

%

可以将一个元素的样式值设置为一个百分比的值，这样浏览器将会根据父元素的值去计算出相应的值

当父元素的值改变时，子元素的值会按照一定比例一起改变，经常用于自适应的页面

em

em会相对于当前元素的字体大小来计算

1em = 1font-size

em经常用于设置文字相关的一些样式，因为当文字大小发生改变时，em会随之改变

颜色单位

颜色单词

直接使用英文单词来表示颜色

red green blue orange

RGB值

所谓RGB值就是通过红 绿 蓝三元色的不同组合来搭配出各种不同的颜色

语法：

rgb(红色,绿色,蓝色)

这三个值需要一个0-255之间的值

0表示没有

255表示最大

rgb(50,200,30)

也可以使用百分数来设置RGB值，需要0%-100%之间的值

百分数最终也是转换为0-255的

0%相当于0

100%相当于255

rgb(100%,0%,0%)

十六进制RGB值

也是一种RGB值的表示方式，不同的是它使用的是16进制数字来表示而不是10进制

语法：

#红色绿色蓝色

这里的颜色需要一个00-ff之间的值

例子：

#ff0000

如果颜色的是两位两位重复的，可以进行简写

比如 #aabbcc 可以写成 #abc

比如 #bbffaa 可以写成 #bfa

文本样式

字体

color

字体的颜色

font-size

字体的大小

浏览器中默认的字体大小一般都是16px，而我们开发时一般会统一为12px

font-family

设置文字的字体

font-style

设置斜体

font-weight

设置文字的加粗

font-variant

小型大写字母

font

文字的简写属性

可以同时设置所有的字体相关的样式

语法：

font: [加粗 斜体 小大字母] 大小[/行高] 字体

加粗，斜体，小大字母，顺序无所谓，写不写都行，如果不写在使用默认值

文本大小，和字体必须写，且大小必须是倒数第二个，字体必须是最后一个

大小后可以设置行高，可写可不写，如果不写则使用默认值

文本样式

line-height

行高

文本默认都是在行高中垂直居中的

通过line-height可以修改行高

行间距 = 行高 - 字体大小

text-transform

设置文本的大小写

text-decoration

设置文本修饰

text-align

设置文本对齐

text-indent

设置首行缩进

它需要一个长度单位，如果是正值则首行向右移动，如果是负值则向左移动

letter-spacing

字符间距

word-spacing

单词间距

背景

background-color

背景颜色

background-image

背景图片

需要一个url地址作为参数

例子：background-image:url(图片的路径)

background-repeat

设置背景图片重复方式

可选值：

repeat

默认值，背景图片会平铺显示

沿x轴和y轴双方向重复

no-repeat

背景图片不重复

repeat-x

背景图片沿水平方向重复

repeat-y

背景图片沿垂直方向重复

background-position

设置背景图片的位置

设置方式一

可以直接通过几个位置的关键字来设置图片的位置

top

left

right

bottom

center

可以通过以上关键字两两组合的形式，将背景图片设置到元素的任意位置

如果仅仅指定一个值，则第二个值默认是center

设置方式二

可以直接指定两个值，来设置背景图片的偏移量

例子：

background-position : x轴偏移量 y轴偏移量;

x轴偏移量，用来指定图片的水平位置

如果指定一个正值，则图片向右移动

如果指定一个负值，则图片向左移动

y轴偏移量，用来指定图片的垂直位置

如果指定一个正值，则图片向下移动

如果指定一个负值，则图片向上移动

background-attachment

用来设置背景是否随页面滚动

可选值

scroll

默认值，背景图片会随页面一起滚动

fixed

背景图片不随页面滚动，会固定在页面的指定位置

设置该属性的背景，则背景会永远相对于浏览器窗口进行定位

一般这种背景都会设置给body

background

背景的简写属性

可以通过它来设置所有的背景相关的样式

该简写属性没有顺序的要求，也没有数量的要求，不写的属性使用默认值

opacity

用来设置背景的不透明度

可选值

0-1

0表示完全透明

1表示完全不透明

0.5半透明

IE8及以下的浏览器不支持该样式，可以使用滤镜来代替

filter:alpha(opacity=值)

这里值需要一个0-100之间的值

0相当于完全透明

100完全不透明

HTML

网页的结构

一个网页有三个部分组成

结构

结构是页面的整体结构，哪里是标题，哪里是段落，哪里是图片

结构使用HTML来编写

表现

表现是页面的外在的样式，比如字体，字体大小，字体颜色，背景。。。

使用CSS来设置页面中元素的样式

行为

页面和用户之间的交互行为

使用JavaScript来设置页面的行为

一个设计优良的网页要求结构、表现、行为三者分离

在开发中总是要面临一个问题，就是程序之间的耦合，三者分离就是为了解耦合

HTML，超文本标记语言

负责页面中的结构，定义出页面中的各个组成部分

HTML是采用纯文本的形式的编写，采用HTML标签来标识出页面中的不同部分

标签

成对出现

<标签名></标签名>

自结束标签

<标签名 />

属性

通过属性可以设置标签的效果

属性需要定义在开始标签中或这自结束标签中

属性实际上是一组一组名值对结构

例子：

<标签名 属性名="属性值" 属性名="属性值"></标签名>

<标签名 属性名="属性值" 属性名="属性值" />

HTML页面的基本结构

文档声明

<!doctype html>

用来标识当前页面的html的版本

该声明用来告诉浏览器，当前的页面是使用HTML5的标准编写的

常用标签

<html>

网页的根标签

一个页面中有且只有一个根标签

网页中的所有内容都需要写在html标签的内部

<head>

网页的头部

该标签中的内容不会在网页中直接显示

该标签用于帮助浏览器解析页面

子标签

<title>

用来设置网页的标题

默认会在浏览器的标题栏中显示

搜索引擎检索网页时，会主要检索title中的内容，它会影响到页面在搜索引擎中的排名

<meta>

用来设置网页的元数据，比如网页使用的字符集

<meta charset="utf-8" />

设置网页的关键字

<meta name="keywords" content="关键字,关键字,关键字,关键字"/>

设置网页的描述

<meta name="description" content="网页的描述"/>

请求的重定向

<meta http-equiv="refresh" content="秒数;url=地址" />

<body>

网页的主体

网页中所有的可见部分都需要在body中编写

<h1> ~ <h6>

标题标签

在html中一共有六级标题

六级标题中，h1最重要，h6最不重要，一般页面中只会使用h1~h3

h1的重要性仅次于title，浏览器也会主要检索h1中的内容，以判断页面的主要内容

一般一个页面中只能写一个h1

<p>

段落标签

<br />

换行标签

<hr />

水平线标签

内联框架

可以向一个页面中引入其他的外部页面

<iframe></iframe>

属性

src

外部页面的地址，可以使用相对路径

width和height

可以设置框架的宽度和高度

name

可以为内联框架指定一个名字

可以将该属性值设置为超链接的target属性的值

这样当点击超链接时，页面将会在相应的内联框架中打开

内联框架中的内容不会被搜索引擎所检索，所以开发中尽量不要使用内联框架

超链接

可以使当前页面跳转到其他的页面

<a>链接的文字</a>

属性

href

指向链接跳转的目标地址，可以是一个相对路径

还可以是#id属性值，这样当点击超链接以后，将会跳转到当前页面的指定位置

可以使用mailto:来创建一个发送电子邮件的超链接

target

指定在哪个窗口中打开链接

可选值

\_self

默认值，默认在当前窗口打开链接

\_blank

在新窗口中打开链接

内联框架的name属性值

在指定的内联框架中打开链接

注释

语法

<!-- 注释内容 -->

注释中的内容不会在页面中显示，但是会在源码中显示，我们可以通过注释来说明网页的代码

也可以通过注释隐藏一些页面中不想显示的内容

实体

在HTML页面中一些特殊符号是不能直接使用，需要使用实体来代替这些特殊符号

实体也可以称为转义字符

实体的语法

&实体名;

常用的实体

空格

&nbsp;

<

&lt;

>

&gt;

版权符号

&copy;

图片标签

<img />

使用图片标签可以向页面中引入一个外部图片

属性

src

指向一个外部图片的路径，可以使用相对路径

alt

指定一个在图片无法加载时对图片的描述

搜索引擎主要通过该属性来识别图片的内容

如果不写该属性则搜索引擎会对图片进行收录

width

设置图片的宽度

height

设置图片的高度

图片的格式

JPEG

颜色丰富的图片，如，照片

GIF

颜色单一，简单透明的图片，动态图

PNG

颜色丰富，复杂透明的图片

图片选择的原则

效果一致，用小的

效果不一致，用效果好的

相对路径

相对于当前资源所在的目录的路径

可以使用../返回一级目录，返回几级使用几个../

xHtml语法规范

1.HTML中不区分大小写，但是尽量使用小写

2.HTML的注释不能嵌套

3.标签必须结构完整

要么成对出现

要么自结束标签

4.标签可以嵌套但是不能交叉嵌套

5.属性必须有值，且值必须加引号，单引号双引号都可以

文本标签

<em>

表示语气上的强调

<strong>

表示内容的重要性

<i>

表示单纯的斜体

<b>

表示单纯的加粗

<small>

表示细则一类的内容

<cite>

表示参考的内容，凡是加书名号的都可以使用cite

<q>

短引用，行内引用

<blockquote>

长引用，块级引用

<sup>

上标

<sub>

下标

<del>

删除的内容

<ins>

插入的内容

<pre>

预格式标签，可以保留代码中空格换行这些格式

<code>

表示程序代码

列表

无序列表

使用ul来创建一个无序列表，在列表中使用li来表示一个列表项

无序列表使用符号作为项目符号

有序列表

使用ol来创建一个无序列表，在列表中使用li来表示一个列表项

使用有序的序号作为项目符号

定义列表

列表相关的元素都是块元素，他们之间可以互相嵌套

去除项目符号

list-style:none

布局

浮动

使用float来设置元素浮动

可选值

none

默认值，不浮动，元素在文档流中

left

元素向左浮动

right

元素向右浮动

特点

1.元素浮动以后会完全脱离文档流

2.浮动以后元素会一直向父元素的最上方移动

3.直到遇到父元素的边框或者其他的浮动元素，会停止移动

4.如果浮动元素的上边是一个块元素，则浮动元素不会覆盖块元素

5.浮动元素不会超过他上边的浮动的兄弟元素，最多一边齐

6.浮动元素不会覆盖文字，文字会自动环绕在浮动元素的周围，可以通过浮动来实现文字环绕的效果

浮动以后元素的特点

元素浮动以后，会使其完全脱离文档流。

块元素

块元素脱离文档流以后

1.不会独占一行

2.宽度和高度都被内容撑开

内联元素

内联元素脱离文档流以后会变成块元素

高度塌陷

父元素在文档流中高度默认是被子元素撑开的，当子元素脱离文档流以后，将无法撑起父元素的高度，也就会导致父元素的高度塌陷

父元素的高度一旦塌陷所有元素的位置将会上移，导致整个页面的布局混乱

方法一

开启父元素的BFC或hasLayout

BFC

Block Formatting Context

块级格式化环境

BFC是元素的隐含属性，默认是在关闭状态的

可以通过一些特殊的样式，来开启BFC

开启BFC以后元素将会具有如下特性

1、父元素的垂直外边距不会与子元素重叠

2、开启BFC的元素不会被浮动元素所覆盖

3、开启BFC的元素可以包含浮动子元素

开启BFC的方式

1.设置元素浮动

2.设置元素绝对定位

3.设置元素的类型为inline-block

4.设置overflow为一个非默认值

一般都是使用overflow:hidden来开启BFC

hasLayout

在IE6中没有BFC，但是有一个和BFC类似的hasLayout

在IE6中可以通过开启hasLayout来解决高度塌陷的问题

副作用最小的开启方式

zoom:1

当为元素设置宽度非默认值时，会自动开启hasLayout

方法二

在塌陷的父元素的最后添加一个空白的div，然后对该div进行清除浮动

使用这种方式会在页面中添加多余的结构

方法三

使用after伪类，向父元素后添加一个块元素，并对其清除浮动

该种方式的原理和方法二原理一样，但是不用向页面中添加对于的结构

定位

通过定位可以将页面中的元素，摆放到页面的任意位置

使用position来设置元素的定位

可选值

static

默认值，元素没有开启定位

relative

开启元素的相对定位

absolute

开启元素的绝对定位

fixed

开启元素的固定定位

相对定位

1.开启元素的相对定位后，如果不设置偏移量元素不会发生任何变化

2.相对定位元素相对于其自身在文档流中的位置来定位

3.相对定位的元素不会脱离文档流

4.相对定位不会改变元素的性质，块元素还是块元素，内联元素还是内联元素

5.相对定位的元素会提升一个层级

绝对定位

1.元素设置绝对定位以后，如果不设置偏移量，元素的位置不会发生变化

2.绝对定位的元素是相对于距离他最近的开启了定位的祖先元素进行定位，如果所有的祖先元素都没开启定位，则相对于浏览器窗口进行定位。

3.绝对定位的元素会完全脱离文档流

4.绝对定位会改变元素的性质。内联变块，块的高度和宽度都被内容撑开，并且不独占一行

5.绝对定位会使元素提升一个层级

固定定位

固定定位是一种特殊的绝对定位，它的特点大部分都和绝对定位一样

不同的是，固定定位的元素永远都是相对于浏览器窗口进行定位的。并且他不会随滚动条滚动

IE6不支持固定定位

层级

定位元素 > 浮动元素 > 文档流中的元素

当元素开启了定位以后，可以通过z-index来设置元素的层级

z-index值越高元素越优先显示

如果z-index值一样，或者都没有z-index则优先显示下边的元素

父元素永远不会盖住子元素

偏移量

当元素开启了定位以后，可以通过偏移量来设置元素的位置

left

元素距离定位位置的左侧距离

top

元素距离定位位置的上边距离

right

元素距离定位位置的右侧距离

bottom

元素距离定位位置的底部距离

一般情况下，只使用两个值即可定义一个元素的位置。

课程简介

软件的架构

C/S，客户端/服务器

1.一般我们使用的软件都是C/S架构

2.比如系统的中的软件QQ、360、office、XMind

3.C表示客户端，用户通过客户端来使用软件

4.S表示服务器，服务器负责处理软件的业务逻辑

特点

1.软件使用前必须得安装

2.软件更新时，服务器和客户端得同时更新

3.C/S架构的软件不能跨平台使用

4.C/S架构的软件客户端和服务器通信采用的是自有协议，相对来说比较安全

B/S，浏览器/服务器

1.B/S本质上也是C/S，只不过B/S架构的软件，使用浏览器作为软件的客户端

2.B/S架构软件通过使用浏览器访问网页的形式，来使用软件

3.比如：京东 淘宝 12306 知乎 新浪微博

特点

1.软件不需要安装，直接使用浏览器访问指定的网址即可

2.软件更新时，客户端不需要更新

3.软件可以跨平台，只要系统中有浏览器，就可以使用

4.B/S架构的软件，客户端和服务器之间通信采用的是通用的HTTP协议，相对来说不安全

进制

几进制就是满几进一

二进制

0 1

10 11 100 101 110 111

十进制

0 1 2 3 4 5 6 7 8 9

10 11 12 13 14 。。。

十六进制

满16进1

0 1 2 3 4 5 6 。。。 9 a b c d e f

10 11 12 ... 19 1a 1b 1c 1d 1e 1f

16进制由于是满16进1，所以必须设置几个特殊的字符来表示10 11 12 13 14 15

使用a b c d e f分别表示10 11 12 13 14 15

八进制

满8进1

0 1 2 3 4 5 6 7

10 11 12 13 14 15 16 17 20 21 22

乱码的问题

乱码出现的原因

计算机是一个非常笨的机器，它只认识两个东西 0 1

在计算机中保存的任何内容，最终都需要转换为0 1这种二进制编码来保存，包括网页中的内容

比如：中国，在计算机底层，可以能需要转换为 1010001001010101011010

在读取内容时，需要将二进制编码，在转换为正确的内容

编码

依据一定的规则，将字符转换为二进制编码的过程

解码

依据一定的规则，将二进制编码转换为字符的过程

字符集

编码和解码所采用的规则，我们称为字符集

常见的字符集

ASCII

ISO-8859-1

GBK

GB2312

中文系统的默认编码

UTF-8

万国码，支持地球上所有的文字

ANSI

自动以系统的默认编码来保存文件

产生乱码的根本原因是，编码和解码采用的字符集不同

在中文系统的浏览器中，默认都是使用GB2312进行解码的

盒子模型

CSS中将每一个元素都设置为了一个矩形的盒子

将所有的元素都设置为盒子，是为了方便页面的布局

当这些元素都是盒子以后，我们的布局就变成了在页面中摆放盒子

盒子模型

每个盒子都由如下几部分构成

内容区

内容区相当于盒子存放东西的空间

内容区在盒子的最里边

元素的所有的子元素都是放在父元素的内容区

内容区设置

width

内容区的宽度

height

内容区的高度

内边距

内边距指的是内容区和边框之间的距离，内边距会影响盒子的大小

盒子中一共有四个方向的内边距

padding-top

padding-right

padding-bottom

padding-left

padding

可以同时设置四个方向的内边距，规则和border-width一致

padding:上 右 下 左

padding:上 左右 下

padding:上下 左右

padding:上下左右

盒子的可见框大小由内容区、内边距和边框共同决定

边框

边框盒子可见框最外侧，边框是盒子的边缘

设置边框

设置边框需要同时设置三个样式，缺一不可

border-width

边框的宽度

可以同时指定四个边框的宽度，也可以分别指定

规则

四个值

border-width : 10px 20px 30px 40px;

border-width : 上 右 下 左;

三个值

border-width : 10px 20px 30px;

border-width : 上 左右 下 ;

两个值

border-width : 10px 20px;

border-width : 上下 左右 ;

一个值

border-width : 10px;

border-width : 上下左右 ;

border-color

边框的颜色

border-style

边框的样式

除了这三个样式，CSS中还提供了

border-xxx-width

border-xxx-color

border-xxx-style

xxx可以是

top

right

bottom

left

通过这些样式可以单独指定四个边的颜色，宽度和样式

边框的简写属性

border

border-left

border-top

border-right

border-bottom

这个几个属性可以同时设置边框相关的的样式

border可以同时设置四个边的颜色，宽度，样式

border-xxx可以单独设置某一个边

规则

使用这些样式可以同时设置border-width border-style border-color，不同的属性使用空格隔开，并且没有顺序要求

外边距

外边距盒子和其他盒子之间的距离，外边距不会影响可见框的大小，但是会影响盒子的位置

也是具有四个方向的外边距

margin-top

margin-right

margin-bottom

margin-left

简写属性

margin

规则和padding一致

margin值

可以设置为auto

如果单独将左右外边距设置为auto，则会将左或右外边距设置为最大值

如果同时将左右外边距设置为auto，则会将左右外边距设置为一个相等的值，通过这种方式使一个子元素在它的父元素中水平居中

margin:0 auto

可以设置为负值

如果将margin设置为负值，则元素会向相反的方向移动

外边距的重叠

相邻的垂直外边距会发生重叠现象

相邻元素的外边距会取最大值

子元素的外边距会传递给父元素

水平外边距不会重叠，而是取和

内联元素的盒子模型

width

height

padding

支持水平方向的padding

垂直方向的padding也支持，但是不会影响布局

border

支持四个方向边框，但是垂直的边框不会影响布局

margin

支持水平方向外边距

不支持垂直方向的

不支持

盒模型相关的样式

display

设置元素的显示类型

可选值

none

元素不会在页面中显示，并且不会占据页面的位置

block

元素会作为块元素显示

inline

元素会作为内联元素显示

inline-block

元素会作为行内块元素显示

既具有内联元素的特点也具有块元素的特点

不独占一行

可以设置宽高

visibility

设置元素是否在页面中显示

可选值

visible

默认值，元素在页面中正常显示

hidden

元素不在页面中显示，但是依然在页面中占据位置

overflow

设置元素如何处理溢出内容

可选值

visible

默认值，不会处理溢出的内容，在父元素以外的地方显示

hidden

溢出的内容会被隐藏不会显示

scroll

在父元素中同时添加水平和垂直方向的滚动条

不内容是否溢出都会添加滚动条

auto

根据需要自动生成滚动条

文档流

文档流指的是网页中的一个位置

文档流是网页的基础，是网页的最底层，所有的元素默认都是在文档流中排列

元素在文档流中默认自左向右，自上向下排列（和我们的书写习惯一致）

块元素

1.块元素在文档流中自上向下排列

2.块元素在文档流中宽度默认是父元素的100%

3.块元素在文档流中高度默认被内容撑开

内联元素

1.内联元素在文档流中自左向右排列，如果一行中不足以容下所有的内联元素，则换到下一行，继续自左至右排列

2.内联元素在文档流中宽度和高度默认都被内容撑开

CSS

CSS叫做层叠样式表，用来设置页面中元素的样式。背景颜色、字体颜色、字体大小。。。

CSS负责结构、表现、行为中的表现

编写的位置

1.内联样式

将样式编写到标签的style属性中

<p style="color:red;"></p>

这种样式只会对当前标签起作用，不能对样式进行复用，不方便后期维护，不推荐使用

2.内部样式表

将样式表编写到head中的style标签中

<style type="text/css"></style>

使用内部样式表，进一步将表现和结构分离，可以同时为多个元素设置样式，方便后期的维护

3.外部样式表

将样式表编写到外部的CSS文件中，然后通过link标签将外部文件引入

<link rel="stylesheet" type="text/css" href="文件的路径"/>

将样式编写到外部样式表中，可以在不同的页面中使用同一个样式表，完全将表现和结构分离，方便后期的维护，推荐使用的方式

基本语法

选择器

通过选择器可以选中页面中的一组元素，然后为其设置样式

元素选择器

根据标签名，选中页面中的指定元素

语法：标签名{ }

例子：

div{}

p{}

h1{}

id选择器

根据元素的id属性值选中一个唯一的元素

语法：#id {}

例子：

#box1{}

#hello{}

类选择器

根据元素的class属性值，选中一组元素

语法：.class{}

例子：

.hello{}

.box{}

通配选择器

选中页面中的所有元素

语法：\*{}

通配选择器的性能比较差，尽量避免使用

并集选择器

可以同时选中符合多个选择器的元素

语法：选择器1,选择器2,选择器N{}

例子：

div,p,#box,.hello{}

交集选择器

可以选中满足多个条件的元素

语法：选择器1选择器2选择器N{}

例子：p.hello{}

后代元素选择器

选中指定元素的指定后代元素

语法：祖先元素 后代元素{}

例子：

div span {}

div p{}

子元素选择器

选中指定元素的指定子元素

语法：父元素 > 子元素 {}

例子：

div > span {}

div > p{}

声明块

声明块中实际上就是一个一个CSS声明

声明

每一个CSS声明都是一个样式，实际上就是一个名值对的结构

名和值之间使用:链接

:左边是样式的名字

:右边是样式的值

每一个声明以;结尾

例子

color:red;

font-size:20px;

元素之间的关系

父元素

直接包含子元素的的元素叫做父元素

子元素

直接被父元素包含的元素叫做子元素

祖先元素

直接或间接包含后代元素的元素叫做祖先元素，父元素也是祖先元素

后代元素

直接或间接被祖先元素包含的元素叫后代元素，子元素也是后代元素

兄弟元素

拥有相同父元素的元素叫做兄弟元素

块元素和内联元素

块元素

块元素会独占页面中的一行，无论他的内容的多少

一般使用块元素对页面进行布局

常见的块元素

div

p

h1~h6

内联元素

内联元素只占用自身的大小，不会独占一行

内联元素也叫行内元素（inline）

一般内联元素都是用来为文本来设置效果

常见的内联

span

a

img

包裹规则

一般都是使用块元素去包裹内联元素，而不会使用内联去包裹块元素

a元素可以包含任意元素，除了a本身

p元素不能包含任何块元素

伪类和伪元素

伪类和伪元素用来表示元素所处的一个特殊的状态，或者是一个特殊的位置

:link

表示一个普通的链接（未访问过的链接）

:visited

表示访问过的链接

:hover

鼠标移入的链接，也可以为其他元素设置hover

:active

正在被点击的链接，也可以为其他元素设置active

:focus

表示元素获取焦点的状态，一般用于文本框

::selection

表示内容被选中的状态

在火狐中使用::-moz-selection来代替

:first-letter

表示第一个字符

:first-line

表示文字的第一行

:before

选中元素的最前边

一般该伪类都会结合content一起使用，通过content可以向指定位置添加内容

:after

选中元素的最后边

一般该伪类都会结合content一起使用，通过content可以向指定位置添加内容

属性选择器

根据元素的属性选择指定元素

[属性名]

选取含有指定属性的元素

[属性名="属性值"]

选取属性值等于指定值的元素

[属性名^="属性值"]

选取属性值以指定内容开头的元素

[属性名$="属性值"]

选取属性值以指定内容结尾的元素

[属性名\*="属性值"]

选取属性值中包含指定内容的元素

兄弟元素选择器

选取后一个兄弟元素

前一个 + 后一个

选取后边所有的兄弟元素

前一个 ~ 后边所有

子元素的伪类

:first-child

寻找父元素的第一个子元素，在所有的子元素中排序

:last-child

寻找父元素的最后一个子元素，在所有的子元素中排序

:nth-child

寻找父元素中的指定位置子元素，在所有的子元素中排序

例子

p:nth-child(3)

可以使用even，来找到偶数的子元素

可以使用odd，来找到奇数的子元素

:first-of-type

寻找指定类型中的第一个子元素

:last-of-type

寻找指定类型中的最后一个子元素

:nth-of-type

寻找指定类型中的指定子元素

否定伪类

从一组元素中将符合要求的元素剔除出去

语法：

:not(选择器)

例子：

.abc:not(div)

样式的继承

为祖先元素设置样式，会同时应用到它的后代元素上，这一特性称为样式的继承。

通过样式的继承可以将一些样式统一设置个祖先元素，这样所有的后代都会应用到相同的样式。

但是并不是所有的样式都会继承，比如：背景相关的，边框相关的，定位相关的。具体参考文档

选择器的优先级

当使用选择器为元素设置样式时，如果样式发生了冲突，采用哪个样式由选择器的优先级来决定。

优先级

内联样式

1000

id选择器

100

类和伪类选择器

10

元素选择器

1

通配选择器

0

继承的样式

没有优先级

当样式发生冲突时，需要将相关的选择器优先级进行求和计算，优先级高的优先显示，如果优先级一样，则显示靠后的样式

优先级计算时，总大小不能超过他的最大的数量级

可以在样式后边添加一个!important，如果样式中添加了该内容，则该样式将会获取最大的优先级，将会优先于所有的样式显示，包括内联样式，但是这个属性要慎用。

选择器的性能

浏览器在解析一组选择器时，是从后边往前一个一个的解析的

如果选择器编写过于长的话，浏览器解析起来性能会比较差，所以在编写选择器时，越短越好。

\*通配选择器，性能也比较差，要避免使用通配选择器

单位

长度单位

px

像素，像素就是构成一个图片的最小的单位，我们的屏幕就是由一个一个像素点构成

一个像素指的就是一个像素点

在不同的显示器中，一个像素的大小是不同的，越清晰的屏幕像素越小

%

可以将一个元素的样式值设置为一个百分比的值，这样浏览器将会根据父元素的值去计算出相应的值

当父元素的值改变时，子元素的值会按照一定比例一起改变，经常用于自适应的页面

em

em会相对于当前元素的字体大小来计算

1em = 1font-size

em经常用于设置文字相关的一些样式，因为当文字大小发生改变时，em会随之改变

颜色单位

颜色单词

直接使用英文单词来表示颜色

red green blue orange

RGB值

所谓RGB值就是通过红 绿 蓝三元色的不同组合来搭配出各种不同的颜色

语法：

rgb(红色,绿色,蓝色)

这三个值需要一个0-255之间的值

0表示没有

255表示最大

rgb(50,200,30)

也可以使用百分数来设置RGB值，需要0%-100%之间的值

百分数最终也是转换为0-255的

0%相当于0

100%相当于255

rgb(100%,0%,0%)

十六进制RGB值

也是一种RGB值的表示方式，不同的是它使用的是16进制数字来表示而不是10进制

语法：

#红色绿色蓝色

这里的颜色需要一个00-ff之间的值

例子：

#ff0000

如果颜色的是两位两位重复的，可以进行简写

比如 #aabbcc 可以写成 #abc

比如 #bbffaa 可以写成 #bfa

文本样式

字体

color

字体的颜色

font-size

字体的大小

浏览器中默认的字体大小一般都是16px，而我们开发时一般会统一为12px

font-family

设置文字的字体

font-style

设置斜体

font-weight

设置文字的加粗

font-variant

小型大写字母

font

文字的简写属性

可以同时设置所有的字体相关的样式

语法：

font: [加粗 斜体 小大字母] 大小[/行高] 字体

加粗，斜体，小大字母，顺序无所谓，写不写都行，如果不写在使用默认值

文本大小，和字体必须写，且大小必须是倒数第二个，字体必须是最后一个

大小后可以设置行高，可写可不写，如果不写则使用默认值

文本样式

line-height

行高

文本默认都是在行高中垂直居中的

通过line-height可以修改行高

行间距 = 行高 - 字体大小

text-transform

设置文本的大小写

text-decoration

设置文本修饰

text-align

设置文本对齐

text-indent

设置首行缩进

它需要一个长度单位，如果是正值则首行向右移动，如果是负值则向左移动

letter-spacing

字符间距

word-spacing

单词间距

背景

background-color

背景颜色

background-image

背景图片

需要一个url地址作为参数

例子：background-image:url(图片的路径)

background-repeat

设置背景图片重复方式

可选值：

repeat

默认值，背景图片会平铺显示

沿x轴和y轴双方向重复

no-repeat

背景图片不重复

repeat-x

背景图片沿水平方向重复

repeat-y

背景图片沿垂直方向重复

background-position

设置背景图片的位置

设置方式一

可以直接通过几个位置的关键字来设置图片的位置

top

left

right

bottom

center

可以通过以上关键字两两组合的形式，将背景图片设置到元素的任意位置

如果仅仅指定一个值，则第二个值默认是center

设置方式二

可以直接指定两个值，来设置背景图片的偏移量

例子：

background-position : x轴偏移量 y轴偏移量;

x轴偏移量，用来指定图片的水平位置

如果指定一个正值，则图片向右移动

如果指定一个负值，则图片向左移动

y轴偏移量，用来指定图片的垂直位置

如果指定一个正值，则图片向下移动

如果指定一个负值，则图片向上移动

background-attachment

用来设置背景是否随页面滚动

可选值

scroll

默认值，背景图片会随页面一起滚动

fixed

背景图片不随页面滚动，会固定在页面的指定位置

设置该属性的背景，则背景会永远相对于浏览器窗口进行定位

一般这种背景都会设置给body

background

背景的简写属性

可以通过它来设置所有的背景相关的样式

该简写属性没有顺序的要求，也没有数量的要求，不写的属性使用默认值

opacity

用来设置背景的不透明度

可选值

0-1

0表示完全透明

1表示完全不透明

0.5半透明

IE8及以下的浏览器不支持该样式，可以使用滤镜来代替

filter:alpha(opacity=值)

这里值需要一个0-100之间的值

0相当于完全透明

100完全不透明

HTML

网页的结构

一个网页有三个部分组成

结构

结构是页面的整体结构，哪里是标题，哪里是段落，哪里是图片

结构使用HTML来编写

表现

表现是页面的外在的样式，比如字体，字体大小，字体颜色，背景。。。

使用CSS来设置页面中元素的样式

行为

页面和用户之间的交互行为

使用JavaScript来设置页面的行为

一个设计优良的网页要求结构、表现、行为三者分离

在开发中总是要面临一个问题，就是程序之间的耦合，三者分离就是为了解耦合

HTML，超文本标记语言

负责页面中的结构，定义出页面中的各个组成部分

HTML是采用纯文本的形式的编写，采用HTML标签来标识出页面中的不同部分

标签

成对出现

<标签名></标签名>

自结束标签

<标签名 />

属性

通过属性可以设置标签的效果

属性需要定义在开始标签中或这自结束标签中

属性实际上是一组一组名值对结构

例子：

<标签名 属性名="属性值" 属性名="属性值"></标签名>

<标签名 属性名="属性值" 属性名="属性值" />

HTML页面的基本结构

文档声明

<!doctype html>

用来标识当前页面的html的版本

该声明用来告诉浏览器，当前的页面是使用HTML5的标准编写的

常用标签

<html>

网页的根标签

一个页面中有且只有一个根标签

网页中的所有内容都需要写在html标签的内部

<head>

网页的头部

该标签中的内容不会在网页中直接显示

该标签用于帮助浏览器解析页面

子标签

<title>

用来设置网页的标题

默认会在浏览器的标题栏中显示

搜索引擎检索网页时，会主要检索title中的内容，它会影响到页面在搜索引擎中的排名

<meta>

用来设置网页的元数据，比如网页使用的字符集

<meta charset="utf-8" />

设置网页的关键字

<meta name="keywords" content="关键字,关键字,关键字,关键字"/>

设置网页的描述

<meta name="description" content="网页的描述"/>

请求的重定向

<meta http-equiv="refresh" content="秒数;url=地址" />

<body>

网页的主体

网页中所有的可见部分都需要在body中编写

<h1> ~ <h6>

标题标签

在html中一共有六级标题

六级标题中，h1最重要，h6最不重要，一般页面中只会使用h1~h3

h1的重要性仅次于title，浏览器也会主要检索h1中的内容，以判断页面的主要内容

一般一个页面中只能写一个h1

<p>

段落标签

<br />

换行标签

<hr />

水平线标签

内联框架

可以向一个页面中引入其他的外部页面

<iframe></iframe>

属性

src

外部页面的地址，可以使用相对路径

width和height

可以设置框架的宽度和高度

name

可以为内联框架指定一个名字

可以将该属性值设置为超链接的target属性的值

这样当点击超链接时，页面将会在相应的内联框架中打开

内联框架中的内容不会被搜索引擎所检索，所以开发中尽量不要使用内联框架

超链接

可以使当前页面跳转到其他的页面

<a>链接的文字</a>

属性

href

指向链接跳转的目标地址，可以是一个相对路径

还可以是#id属性值，这样当点击超链接以后，将会跳转到当前页面的指定位置

可以使用mailto:来创建一个发送电子邮件的超链接

target

指定在哪个窗口中打开链接

可选值

\_self

默认值，默认在当前窗口打开链接

\_blank

在新窗口中打开链接

内联框架的name属性值

在指定的内联框架中打开链接

注释

语法

<!-- 注释内容 -->

注释中的内容不会在页面中显示，但是会在源码中显示，我们可以通过注释来说明网页的代码

也可以通过注释隐藏一些页面中不想显示的内容

实体

在HTML页面中一些特殊符号是不能直接使用，需要使用实体来代替这些特殊符号

实体也可以称为转义字符

实体的语法

&实体名;

常用的实体

空格

&nbsp;

<

&lt;

>

&gt;

版权符号

&copy;

图片标签

<img />

使用图片标签可以向页面中引入一个外部图片

属性

src

指向一个外部图片的路径，可以使用相对路径

alt

指定一个在图片无法加载时对图片的描述

搜索引擎主要通过该属性来识别图片的内容

如果不写该属性则搜索引擎会对图片进行收录

width

设置图片的宽度

height

设置图片的高度

图片的格式

JPEG

颜色丰富的图片，如，照片

GIF

颜色单一，简单透明的图片，动态图

PNG

颜色丰富，复杂透明的图片

图片选择的原则

效果一致，用小的

效果不一致，用效果好的

相对路径

相对于当前资源所在的目录的路径

可以使用../返回一级目录，返回几级使用几个../

xHtml语法规范

1.HTML中不区分大小写，但是尽量使用小写

2.HTML的注释不能嵌套

3.标签必须结构完整

要么成对出现

要么自结束标签

4.标签可以嵌套但是不能交叉嵌套

5.属性必须有值，且值必须加引号，单引号双引号都可以

文本标签

<em>

表示语气上的强调

<strong>

表示内容的重要性

<i>

表示单纯的斜体

<b>

表示单纯的加粗

<small>

表示细则一类的内容

<cite>

表示参考的内容，凡是加书名号的都可以使用cite

<q>

短引用，行内引用

<blockquote>

长引用，块级引用

<sup>

上标

<sub>

下标

<del>

删除的内容

<ins>

插入的内容

<pre>

预格式标签，可以保留代码中空格换行这些格式

<code>

表示程序代码

列表

无序列表

使用ul来创建一个无序列表，在列表中使用li来表示一个列表项

无序列表使用符号作为项目符号

有序列表

使用ol来创建一个无序列表，在列表中使用li来表示一个列表项

使用有序的序号作为项目符号

定义列表

列表相关的元素都是块元素，他们之间可以互相嵌套

去除项目符号

list-style:none

布局

浮动

使用float来设置元素浮动

可选值

none

默认值，不浮动，元素在文档流中

left

元素向左浮动

right

元素向右浮动

特点

1.元素浮动以后会完全脱离文档流

2.浮动以后元素会一直向父元素的最上方移动

3.直到遇到父元素的边框或者其他的浮动元素，会停止移动

4.如果浮动元素的上边是一个块元素，则浮动元素不会覆盖块元素

5.浮动元素不会超过他上边的浮动的兄弟元素，最多一边齐

6.浮动元素不会覆盖文字，文字会自动环绕在浮动元素的周围，可以通过浮动来实现文字环绕的效果

浮动以后元素的特点

元素浮动以后，会使其完全脱离文档流。

块元素

块元素脱离文档流以后

1.不会独占一行

2.宽度和高度都被内容撑开

内联元素

内联元素脱离文档流以后会变成块元素

高度塌陷

父元素在文档流中高度默认是被子元素撑开的，当子元素脱离文档流以后，将无法撑起父元素的高度，也就会导致父元素的高度塌陷

父元素的高度一旦塌陷所有元素的位置将会上移，导致整个页面的布局混乱

方法一

开启父元素的BFC或hasLayout

BFC

Block Formatting Context

块级格式化环境

BFC是元素的隐含属性，默认是在关闭状态的

可以通过一些特殊的样式，来开启BFC

开启BFC以后元素将会具有如下特性

1、父元素的垂直外边距不会与子元素重叠

2、开启BFC的元素不会被浮动元素所覆盖

3、开启BFC的元素可以包含浮动子元素

开启BFC的方式

1.设置元素浮动

2.设置元素绝对定位

3.设置元素的类型为inline-block

4.设置overflow为一个非默认值

一般都是使用overflow:hidden来开启BFC

hasLayout

在IE6中没有BFC，但是有一个和BFC类似的hasLayout

在IE6中可以通过开启hasLayout来解决高度塌陷的问题

副作用最小的开启方式

zoom:1

当为元素设置宽度非默认值时，会自动开启hasLayout

方法二

在塌陷的父元素的最后添加一个空白的div，然后对该div进行清除浮动

使用这种方式会在页面中添加多余的结构

方法三

使用after伪类，向父元素后添加一个块元素，并对其清除浮动

该种方式的原理和方法二原理一样，但是不用向页面中添加对于的结构

定位

通过定位可以将页面中的元素，摆放到页面的任意位置

使用position来设置元素的定位

可选值

static

默认值，元素没有开启定位

relative

开启元素的相对定位

absolute

开启元素的绝对定位

fixed

开启元素的固定定位

相对定位

1.开启元素的相对定位后，如果不设置偏移量元素不会发生任何变化

2.相对定位元素相对于其自身在文档流中的位置来定位

3.相对定位的元素不会脱离文档流

4.相对定位不会改变元素的性质，块元素还是块元素，内联元素还是内联元素

5.相对定位的元素会提升一个层级

绝对定位

1.元素设置绝对定位以后，如果不设置偏移量，元素的位置不会发生变化

2.绝对定位的元素是相对于距离他最近的开启了定位的祖先元素进行定位，如果所有的祖先元素都没开启定位，则相对于浏览器窗口进行定位。

3.绝对定位的元素会完全脱离文档流

4.绝对定位会改变元素的性质。内联变块，块的高度和宽度都被内容撑开，并且不独占一行

5.绝对定位会使元素提升一个层级

固定定位

固定定位是一种特殊的绝对定位，它的特点大部分都和绝对定位一样

不同的是，固定定位的元素永远都是相对于浏览器窗口进行定位的。并且他不会随滚动条滚动

IE6不支持固定定位

层级

定位元素 > 浮动元素 > 文档流中的元素

当元素开启了定位以后，可以通过z-index来设置元素的层级

z-index值越高元素越优先显示

如果z-index值一样，或者都没有z-index则优先显示下边的元素

父元素永远不会盖住子元素

偏移量

当元素开启了定位以后，可以通过偏移量来设置元素的位置

left

元素距离定位位置的左侧距离

top

元素距离定位位置的上边距离

right

元素距离定位位置的右侧距离

bottom

元素距离定位位置的底部距离

一般情况下，只使用两个值即可定义一个元素的位置。

课程简介

软件的架构

C/S，客户端/服务器

1.一般我们使用的软件都是C/S架构

2.比如系统的中的软件QQ、360、office、XMind

3.C表示客户端，用户通过客户端来使用软件

4.S表示服务器，服务器负责处理软件的业务逻辑

特点

1.软件使用前必须得安装

2.软件更新时，服务器和客户端得同时更新

3.C/S架构的软件不能跨平台使用

4.C/S架构的软件客户端和服务器通信采用的是自有协议，相对来说比较安全

B/S，浏览器/服务器

1.B/S本质上也是C/S，只不过B/S架构的软件，使用浏览器作为软件的客户端

2.B/S架构软件通过使用浏览器访问网页的形式，来使用软件

3.比如：京东 淘宝 12306 知乎 新浪微博

特点

1.软件不需要安装，直接使用浏览器访问指定的网址即可

2.软件更新时，客户端不需要更新

3.软件可以跨平台，只要系统中有浏览器，就可以使用

4.B/S架构的软件，客户端和服务器之间通信采用的是通用的HTTP协议，相对来说不安全

进制

几进制就是满几进一

二进制

0 1

10 11 100 101 110 111

十进制

0 1 2 3 4 5 6 7 8 9

10 11 12 13 14 。。。

十六进制

满16进1

0 1 2 3 4 5 6 。。。 9 a b c d e f

10 11 12 ... 19 1a 1b 1c 1d 1e 1f

16进制由于是满16进1，所以必须设置几个特殊的字符来表示10 11 12 13 14 15

使用a b c d e f分别表示10 11 12 13 14 15

八进制

满8进1

0 1 2 3 4 5 6 7

10 11 12 13 14 15 16 17 20 21 22

乱码的问题

乱码出现的原因

计算机是一个非常笨的机器，它只认识两个东西 0 1

在计算机中保存的任何内容，最终都需要转换为0 1这种二进制编码来保存，包括网页中的内容

比如：中国，在计算机底层，可以能需要转换为 1010001001010101011010

在读取内容时，需要将二进制编码，在转换为正确的内容

编码

依据一定的规则，将字符转换为二进制编码的过程

解码

依据一定的规则，将二进制编码转换为字符的过程

字符集

编码和解码所采用的规则，我们称为字符集

常见的字符集

ASCII

ISO-8859-1

GBK

GB2312

中文系统的默认编码

UTF-8

万国码，支持地球上所有的文字

ANSI

自动以系统的默认编码来保存文件

产生乱码的根本原因是，编码和解码采用的字符集不同

在中文系统的浏览器中，默认都是使用GB2312进行解码的

盒子模型

CSS中将每一个元素都设置为了一个矩形的盒子

将所有的元素都设置为盒子，是为了方便页面的布局

当这些元素都是盒子以后，我们的布局就变成了在页面中摆放盒子

盒子模型

每个盒子都由如下几部分构成

内容区

内容区相当于盒子存放东西的空间

内容区在盒子的最里边

元素的所有的子元素都是放在父元素的内容区

内容区设置

width

内容区的宽度

height

内容区的高度

内边距

内边距指的是内容区和边框之间的距离，内边距会影响盒子的大小

盒子中一共有四个方向的内边距

padding-top

padding-right

padding-bottom

padding-left

padding

可以同时设置四个方向的内边距，规则和border-width一致

padding:上 右 下 左

padding:上 左右 下

padding:上下 左右

padding:上下左右

盒子的可见框大小由内容区、内边距和边框共同决定

边框

边框盒子可见框最外侧，边框是盒子的边缘

设置边框

设置边框需要同时设置三个样式，缺一不可

border-width

边框的宽度

可以同时指定四个边框的宽度，也可以分别指定

规则

四个值

border-width : 10px 20px 30px 40px;

border-width : 上 右 下 左;

三个值

border-width : 10px 20px 30px;

border-width : 上 左右 下 ;

两个值

border-width : 10px 20px;

border-width : 上下 左右 ;

一个值

border-width : 10px;

border-width : 上下左右 ;

border-color

边框的颜色

border-style

边框的样式

除了这三个样式，CSS中还提供了

border-xxx-width

border-xxx-color

border-xxx-style

xxx可以是

top

right

bottom

left

通过这些样式可以单独指定四个边的颜色，宽度和样式

边框的简写属性

border

border-left

border-top

border-right

border-bottom

这个几个属性可以同时设置边框相关的的样式

border可以同时设置四个边的颜色，宽度，样式

border-xxx可以单独设置某一个边

规则

使用这些样式可以同时设置border-width border-style border-color，不同的属性使用空格隔开，并且没有顺序要求

外边距

外边距盒子和其他盒子之间的距离，外边距不会影响可见框的大小，但是会影响盒子的位置

也是具有四个方向的外边距

margin-top

margin-right

margin-bottom

margin-left

简写属性

margin

规则和padding一致

margin值

可以设置为auto

如果单独将左右外边距设置为auto，则会将左或右外边距设置为最大值

如果同时将左右外边距设置为auto，则会将左右外边距设置为一个相等的值，通过这种方式使一个子元素在它的父元素中水平居中

margin:0 auto

可以设置为负值

如果将margin设置为负值，则元素会向相反的方向移动

外边距的重叠

相邻的垂直外边距会发生重叠现象

相邻元素的外边距会取最大值

子元素的外边距会传递给父元素

水平外边距不会重叠，而是取和

内联元素的盒子模型

width

height

padding

支持水平方向的padding

垂直方向的padding也支持，但是不会影响布局

border

支持四个方向边框，但是垂直的边框不会影响布局

margin

支持水平方向外边距

不支持垂直方向的

不支持

盒模型相关的样式

display

设置元素的显示类型

可选值

none

元素不会在页面中显示，并且不会占据页面的位置

block

元素会作为块元素显示

inline

元素会作为内联元素显示

inline-block

元素会作为行内块元素显示

既具有内联元素的特点也具有块元素的特点

不独占一行

可以设置宽高

visibility

设置元素是否在页面中显示

可选值

visible

默认值，元素在页面中正常显示

hidden

元素不在页面中显示，但是依然在页面中占据位置

overflow

设置元素如何处理溢出内容

可选值

visible

默认值，不会处理溢出的内容，在父元素以外的地方显示

hidden

溢出的内容会被隐藏不会显示

scroll

在父元素中同时添加水平和垂直方向的滚动条

不内容是否溢出都会添加滚动条

auto

根据需要自动生成滚动条

文档流

文档流指的是网页中的一个位置

文档流是网页的基础，是网页的最底层，所有的元素默认都是在文档流中排列

元素在文档流中默认自左向右，自上向下排列（和我们的书写习惯一致）

块元素

1.块元素在文档流中自上向下排列

2.块元素在文档流中宽度默认是父元素的100%

3.块元素在文档流中高度默认被内容撑开

内联元素

1.内联元素在文档流中自左向右排列，如果一行中不足以容下所有的内联元素，则换到下一行，继续自左至右排列

2.内联元素在文档流中宽度和高度默认都被内容撑开