# 认识网页

## 浏览器内核，渲染引擎

Ie内核：triteent

谷歌/欧鹏：blink

火狐：gecko

苹果：webkit

渲染引擎是出现兼容性的根本问题

## URL地址

就是我们所说的网址：[www.jd.com](http://www.jd.com)

# 认识HTML

## 超文本标记语言（Hyper Text Markup Language）

超文本：超链接（实现页面跳转）

结构规则：HTML（Hyper Text Markup Language）

样式规则：CSS（Cascading Style Sheets）

行为规则：JS（JavaScript）

## Html结构标准

1. **<!DOCTYPE html> //声明文档类型**
2. **<html> //根标签**
3. **<head> //头部标签**
4. **<title></title> //标题标签**
5. **</head>**
6. **<body> //主题标签**
7. **</body>**
8. **</html>**

后缀名不能决定文件的格式，只能决定文档打开的方式

## Html标签分类

单标签：

注释标签

<!—注释内容-->

换行标签

<br/> :上下不会产生空白行

水平线标签

<hr/>

双标签

|  |  |  |  |
| --- | --- | --- | --- |
| **标签名字** | 标签作用 | 标签属性 | *Display* |
| **<!DOCTYPE >** | 定义文档类型 |  |  |
| **<p></p>** | 段落标签，文本内容（上下自动产生一个空白行，<br/>不会） |  |  |
| **<h1-h6>** | 标题标签：一个页面只能出现一次h1  因为H1搜索权重高  影响SEO搜索优化 | 块级元素 |  |
| **<font>** | 文本标签 |  |  |
| **<a>** | 定义一个超链接 | Href=“链接的网址”  title=“提示的文字”  target=“链接在何处打开” | Line |
| **<abbr>** | 定义缩写 | Title=“缩写前的全称” | Line |
| **<area>** | 定义热点，总是嵌套在<map>中实用   1. <img src="images/2s.jpg" usemap="#name1" /> 2. <map name="name1"> 3. <area shape="circle" coords="0,0,50" href="#"> 4. </map> | Alt：定义替换的文字  Cords：热点坐标值  圆形（circle）：（x，y，r）  X,y定义了远点坐标，R半径  方形（rect）：（x1,y1,x2,y2,）  X1,y1:定义了左上角顶点  2x,y2:定义了右下角顶点  (网页图片左上角坐标是：0，0)  Href：目标的链接  Target：何地打开链接  图片的usemap属性要和map的那么属性一样，并且图片的usemap属性值要加井号# |  |
| **Base** | 为页面内所有链接规定默认地址或者默认目标  位于head内部 | Href=“链接地址URL”  Target=“在何地打开链接” |  |
| **Bdo** | 定义文字方向 | Dir=“ltr（左）/rtl（右）” |  |
| **Body** | 定义文档的主题 |  |  |
| **Button** | 定义按钮 | Disabled=“disabled”禁用此按钮  Name：按钮名称  Type=“sibmit/reset/button”  提交/重置/普通按钮  Value=“按钮的初始值” | Line |
| **Caption** | 定义表格标题，必须紧跟table标签，只能对每个标签定义一个标题，通常居中与表格之上 |  |  |
| **Code** | 定义计算机代码文本 |  |  |
| **Col** | 为一个或多个列设置属性 | Span：规定行横跨的列数  Align水平对其方式  Valign：垂直对其方式 |  |
| **Div** | 定义文档中的分区 |  |  |
| **Dl，dt，dd** | 自定义列表 |  |  |
| **Form** | 表单标签 |  |  |
| **Frame** | 定义一个框架 |  |  |
| **Frameset** | 定义一个框架集 | Col：定义框架的列数  Rew：定义框架水平行 |  |
| **Img** | 定义图像 | Src=“图像路径”  Alt=“替换文本”  Title=“提示文字”  Width、height |  |
| **Input** | 定义输入控件 | Type=“text/button/password/checkbox/radio/submit/reset”  文本/按钮/密码/复选框/单选框/提交按钮/重置按钮  Value=“元素的默认值” |  |
| **Lable** | 为input元素定义标注，  Lable属性的for应与被绑定的元素的id元素相同 |  |  |
| **Link** | 定义文档与外部资源的联系，最常见的用途是链接样式表 | Charset=“链接文档的字符编码方式”  Href=“被链接文档的路径”  Rel=“当前文本与被链接的文本的关系”  Rev=“定义被链接的文本与当前文本的关系” |  |
| **Map** | 定义热点链接 | Id/name |  |
| **Meta** | 定义文档的元数据 | Content=“内容”  http-equiv=“把content属性关联到http头部”  Name=“description/keywords/” |  |
| **Object** | 定义内嵌对象 |  |  |
| **Ol、ul、** | 定义有序/无序列表 |  |  |
| **Select>Option** | 定义下拉列表 | Selected=“selected”定义默认被选中项  Value=“定义被发往服务器的值”  Disabled=”disabled”被禁用的项 |  |
| **Span** | 定义文档中的行内元素 |  | Line |
| **Style** | 定义内嵌式样式表 | Type=“text/css” |  |
| **Sub/sup** | 定义下标/上标文本 |  |  |
| **Table/tr/td** | 定义表格 | Border=“边框” |  |
| **Th** | 定义表头单元格 | <tr><th></th></tr> |  |
| **Textarea** | 定义多行文本输入框 | Cols=“文本可见宽度”  Rows=“文本可见高度”  Readonly=“readonly”规定只读 |  |
| **Tfoot** | 定义表格页脚 |  |  |

## HTML标准属性

|  |  |  |
| --- | --- | --- |
| **属性** | 值 | 说明 |
| **Class** | Classname | 规定元素的类名 |
| **Id** | Idname | 规定元素的唯一ID |
| **Style** |  | 规定元素的行内样式 |
| **Title** | Text | 规定元素的额外信息 |
| **语言属性** |  |  |
| **Dir** | Ltr/lt | 规定文本的方向 |
| **键盘属性** |  |  |
| **Accseekey** |  | 设置访问元素的键盘快捷键 |
| **Tabindex** |  | 设置元素的tab键控制次序 |

## http方法

在客户机和服务器之间进行请求-响应时，两种最常被用到的方法是：GET 和 POST。

### get：从指定的资源请求数据

有关 GET 请求的其他一些注释：

* GET 请求可被缓存
* GET 请求保留在浏览器历史记录中
* GET 请求可被收藏为书签
* GET 请求不应在处理敏感数据时使用
* GET 请求有长度限制
* GET 请求只应当用于取回数据

### Post：向指定的资源提交要被处理的数据

有关 POST 请求的其他一些注释：

* POST 请求不会被缓存
* POST 请求不会保留在浏览器历史记录中
* POST 不能被收藏为书签
* POST 请求对数据长度没有要求

### 比较 GET 与 POST

|  |  |  |
| --- | --- | --- |
|  | GET | POST |
| 后退按钮/刷新 | 无害 | 数据会被重新提交 |
| 书签 | 可收藏为书签 | 不可收藏为书签 |
| 缓存 | 能被缓存 | 不能缓存 |
| 编码类型 | application/x-www-form-urlencoded | application/x-www-form-urlencoded 或 multipart/form-data。为二进制数据使用多重编码。 |
| 历史 | 参数保留在浏览器历史中 | 参数不会保存在浏览器历史中。 |
| 对数据长度的要求 | 是的。当发送数据时，GET 方法向 URL 添加数据；URL 的长度是受限制的（URL 的最大长度是 2048 个字符）。 | 无限制 |
| 对数据类型的限制 | 只允许 ASCII 字符。 | 没有限制。也允许二进制数据。 |
| 安全性 | 与 POST 相比，GET 的安全性较差，因为所发送的数据是 URL 的一部分。 | POST 比 GET 更安全，因为参数不会被保存在浏览器历史或 web 服务器日志中 |
| 可见性 | 数据在 URL 中对所有人都是可见的。 | 数据不会显示在 URL 中。 |

## Html标签关系分类

父子关系

并列关系

## **文本格式化**

<strong>语气加重、加粗

<b>加粗（尽量不用）

<em>文本倾斜

<i>文本倾斜（尽量不用）

**<del>删除线**

**<s>删除线（尽量不用）**

**<ins>下划线**

**<u>下划线（尽量不用）**

**◆注意：之所以使用<string><em><del><ins>，是因为这些标签更具有语义化**

## **img图片标签**

<img src=”路径” alt=“替换图片文本，对SEO优化有好处” title=“提示文本（鼠标放到图片上显示的文字）”

单独更改img的宽或者高 图片等比缩放 不会变形

同时设置img的宽高会导致图片变形

没有定义宽高的时候按照图片尺寸的100%的尺寸显示

## **路径**

●相对路径：相对于自己出发找其他文件

跳出文件夹就要使用 （ . . / ）

★往下一级目录用（ / ） 往上一级目录用（ . . / ）

●绝对路径：文件完整的路径

## 超链接

href：链接到的地址（必写属性）

target：新链接在哪打开 \_blank 在新窗口打开

title：提示文本

<base target=”\_blank”>写在head之中，作用是让所以的超链接都在新窗口打开

Base 作用是使页面内所有超链接指向同一目标或者同一个链接

## **列表**

ul 无序列表

★type属性

■ square 实心方块

● disc 实心圆圈

○ circle 空心圆圈

ol 有序列表

★type属性

1、A、a、I、i

★start 从哪个数开始

dl 自定义列表

dl定义自定义列表

dt 定义标题

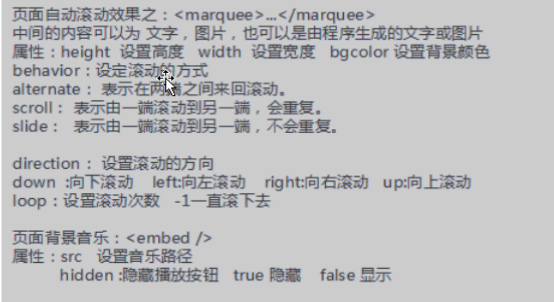
dd 定义内容

## 音乐标签

Embed

Hidden=“true” 隐藏播放器

## 滚动

marquee

## HTML头部标签

|  |  |
| --- | --- |
| **<head>** | 定义关于文档的信息 |
| **<title>** | 定义文档标题 |
| **<base>** | 定义页面上所有链接的默认地址或默认目标 |
| **<link>** | 定义文档与外部资源的关系 |
| **<meta>** | 定义HTML文档的元数据 |
| **<script>** | 定义客户端脚本 |
| **<style>** | 定义文档的样式信息 |
|  |  |

## **Base元素**

1. <head>
2. <base href="http://www.w3school.com.cn/images/" />
3. <base target="\_blank" />
4. </head>

Base标签为页面所有链接都指向默认地址或者默认目标

## Link元素

1. <head>
2. <link rel="stylesheet" type="text/css" href="mystyle.css" /> //外链样式
3. <link rel=”shortcut icon（只用icon也可以）” type=“image/x-icon” href=”图标路径”> //网页快捷图标
4. </head>

Link：标签定义文档与外部资源之间的关系

* 1. Rel：属性定义该标签关联的样式表标签
  2. Href：定义该便签所链接的外部资源的URL地址，可相对地址，可以是绝对地址
  3. Type：定义文档类型
  4. Media：设置属性调用的资源主要针对什么设备而使用
     1. Screen：针对计算器机显示器的样式
     2. All：所以的设备
        1. <style type=”text/css”>
        2. @import URL(“css/base.css”);
        3. </style>//不建议使用

## Meta元数据

元数据metadata是关于数据的信息，元数据不会显示在页面上，对与机器是可读的。

Meta元素被用来规定页面的描述、关键词、文档的作者、最后的修改时间以及其他元数据。

Meta标签始终位于head元素中。

元数据可用于浏览器（如何显示内容或重新加载页面），搜索引擎（关键词），或者其他web服务。

◆ 一些搜索引擎会利用meta元素的name和conteng属性来索引你的页面

下面的 meta 元素定义页面的描述：

1. <meta name="description" content="Free Web tutorials on HTML, CSS, XML" />

下面的 meta 元素定义页面的关键词：

1. <meta name="keywords" content="HTML, CSS, XML" />

下面的http-equiv=“refresh”，定义了网页的重新定向，自动跳转到新网址

1. <meta http-equiv=”refresh” content=”5;www.baidu.com”>

## Script元素

<script>用来定义客户端脚本，典型的JavaScript

## HTML脚本

JavaScript是HTML文档具有更强的动态性和交互性

JavaScript最常用于图片操作、表单验证、动态内容更新

<noscript> 标签提供无法使用脚本时的替代内容，比方在浏览器禁用脚本时，或浏览器不支持客户端脚本时。

noscript 元素可包含普通 HTML 页面的 body 元素中能够找到的所有元素。

只有在浏览器不支持脚本或者禁用脚本时，才会显示 noscript 元素中的内容：

1. <script type="text/javascript">
2. document.write("Hello World!")
3. </script>
4. <noscript>Your browser does not support JavaScript!</noscript>

## HTML字符实体（特殊符号）

在 HTML 中，某些字符是预留的

在 HTML 中不能使用小于号（<）和大于号（>），这是因为浏览器会误认为它们是标签。

在HTML文件中的多个空格会被解释成只有一个空格



## HTML统一资源定位器URL

URL（Unifrom Resource Locator）也被称为网址

URL可由单词组成，比如[www.taobao.com](http://www.taobao.com)，或者因特网协议（IP）地址：192.168.12.122。大多数人上网的时候会输入网站的域名。

当您点击 HTML 页面中的某个链接时，对应的 <a> 标签指向万维网上的一个地址。

统一资源定位器（URL）用于定位万维网上的文档（或其他数据）。

网址，比如 [http://www.w3school.com.cn/html/index.asp](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/html/index.asp)，遵守以下的语法规则：

scheme://host.domain:port/path/filename

◆解释：

scheme - 定义因特网服务的类型。最常见的类型是 http

host - 定义域主机（http 的默认主机是 www）

domain - 定义因特网域名，比如 w3school.com.cn

:port - 定义主机上的端口号（http 的默认端口号是 80）

path - 定义服务器上的路径（如果省略，则文档必须位于网站的根目录中）。

filename - 定义文档/资源的名称

|  |  |  |
| --- | --- | --- |
| Scheme（服务类型） | 访问 | 用于… |
| http | 超文本传输协议 | 以http://开头的普通网页，不加密 |
| https | 安全超文本传输协议 | 安全网页，加密所有信息交换 |
| ftp | 文件传输协议 | 用于将文件上传或者下载 |
| File |  | 您计算机上的文件 |

## URL编码

URL只能使用ASCII字符集通过因特网进行发送

由于 URL 常常会包含 ASCII 集合之外的字符，URL 必须转换为有效的 ASCII 格式。

URL 编码使用 "%" 其后跟随两位的十六进制数来替换非 ASCII 字符。

URL 不能包含空格。URL 编码通常使用 + 来替换空格。

## HTML WEB Server

**ISP**：因特网服务提供商

大多数小公司会把网站存放到由 ISP 提供的服务器上

## HTML多媒体

Web 上的多媒体指的是音效、音乐、视频和动画。

HTML5多媒体标签

<embed src=“路径” width=“” height=“”></embed>

# XHTML

## 什么是XHTML？

* XHTML 指的是可扩展超文本标记语言
* XHTML 与 HTML 4.01 几乎是相同的
* XHTML 是更严格更纯净的 HTML 版本
* XHTML 是以 XML 应用的方式定义的 HTML
* XHTML 是 [2001 年 1 月](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/w3c/w3c_xhtml.asp)发布的 W3C 推荐标准
* XHTML 得到所有主流浏览器的支持

## 为什么使用 XHTML？

因特网上的很多页面包含了“糟糕”的 HTML。

XHTML 是一种必须正确标记且格式良好的标记语言

## 与 HTML 相比最重要的区别

### 文档结构

* XHTML DOCTYPE 是强制性的
* <html> 中的 XML namespace 属性是强制性的
* <html>、<head>、<title> 以及 <body> 也是强制性的

### 元素语法

* XHTML 元素必须正确嵌套
* XHTML 元素必须始终关闭
* XHTML 元素必须小写
* XHTML 文档必须有一个根元素

### 属性语法

* XHTML 属性必须使用小写
* XHTML 属性值必须用引号包围
* XHTML 属性最小化也是禁止的

XHTML 文档必须进行 XHTML 文档类型声明

（XHTML DOCTYPE declaration）。

## 如何从 HTML 转换到 XHTML

1. 向每张页面的第一行添加 XHTML <!DOCTYPE>
2. 向每张页面的 html 元素添加 xmlns 属性
3. 把所有元素名改为小写
4. 关闭所有空元素
5. 把所有属性名改为小写
6. 为所有属性值加引号

# Sublimt编辑器

## 常用快捷键

|  |  |
| --- | --- |
| **Html：xt** | **调出HTML基本结构代码** |
| **Ctrl+shift+D** | **快速复制一行** |
| **Ctrl+shift+K** | **快速删除一行** |
| **Ctrl+鼠标左键** | **选择集体输入** |
| **Ctrl+H** | **查找替换** |
| **Ctrl+F** | **查找** |
| **Ctrl+?** | **注释** |
| **Ctrl+L** | **快速选择一行** |
| **Ctrl+shift+↑↓** | **快速上下移动一行** |
| **Alt+shift+12345789** | **界面布局** |
| **F11** | **全屏显示界面** |
| **Ctrl+K+B** | **显示/隐藏侧边栏** |

# ****CSS（****Cascading Style Sheets****）基础****

## CSS概述

* CSS指层叠样式表（*C*ascading *S*tyle *S*heets）
* 样式定义***如何显示* HTML** 元素
* 样式通常存储在*样式表*中
* 把样式添加到 HTML 4.0 中，是为了*解决内容与表现分离的问题*
* *外部样式表*可以极大提高工作效率
* 外部样式表通常存储在 *CSS 文件*中
* *多个样式定义可层叠为一*
* 所有的主流浏览器均支持层叠样式表。

*◆格式优先级*

1. 浏览器缺省设置
2. 外部样式表
3. 内部样式表（位于 <head> 标签内部）
4. *内联样式（在HTML 元素内部）14*
5. *！important 级别最高*

## CSS语法

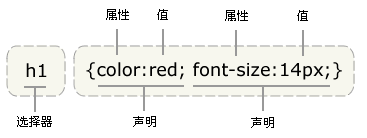
CSS 规则由两个主要的部分构成：选择器，以及一条或多条声明。

1. selector {declaration1; declaration2; ... declarationN }

◆选择器通常是您需要改变样式的 HTML 元素。

◆每条声明由一个属性和一个值组成。

◆属性（property）是您希望设置的样式属性（style attribute）。每个属性有一个值。属性和值被冒号分开。



◆如果值为若干单词，则要给值加引号

◆是否包含空格不会影响 CSS 在浏览器的工作效果，同样，与 XHTML 不同，CSS 对大小写不敏感。不过存在一个例外：如果涉及到与 HTML 文档一起工作的话，class 和 id 名称对大小写是敏感的

## CSS高级语法

* 选择器的分组

被分组的选择器就可以分享相同的声明。用逗号将需要分组的选择器分开

* 继承inherit

子元素从父元素继承属性，但也有意外

子元素将继承最高级元素

## 创建CSS

* 外部样式表

当样式需要应用于很多页面时，外部样式表将是理想的选择。在使用外部样式表的情况下，你可以通过改变一个文件来改变整个站点的外观。

1. <head>
2. <link rel=”stylesheet” type=”text/css” href=”路径” />
3. </head>

不要在属性值与单位之间留有空格

* 内部样式表

当单个文档需要特殊的样式时，就应该使用内部样式表。你可以使用 <style> 标签在文档头部定义内部样式表

1. <head>
2. <style type=”text/css”>
3. p{
4. color:red;
5. }
6. </style>
7. </head>

* 内嵌样式

由于要将表现和内容混杂在一起，内联样式会损失掉样式表的许多优势。请慎用这种方法，例如当样式仅需要在一个元素上应用一次时。

要使用内联样式，你需要在相关的标签内使用样式（style）属性。Style 属性可以包含任何 CSS 属性。

1. <p style="color: sienna; margin-left: 20px">
2. This is a paragraph
3. </p>

* 多重样式

如果某些属性在不同的样式表中被同样的选择器定义，那么属性值将从更具体的样式表中被继承过来。

内部有的继承内部的，内部没有外部有，继承外部的

1. h3 {
2. color: red;
3. text-align: left;
4. font-size: 8pt;
5. } //外链样式表
6. h3 {
7. text-align: right;
8. font-size: 20pt;
9. } //内联样式表

最终H3的样式为

1. color: red; //继承外部样式表的
2. text-align: right; //继承内部样式表的
3. font-size: 20pt; //继承内部样式表的

## CSS样式

### 基本属性

Background：颜色 背景图片url（路径） 平铺（no-repeat） 固定位置（fixed） 背景图片位置x y；

同时设置背景图片和背景颜色时：背景图片覆盖背景颜色

Padding 内边距 撑大盒子

Margin 外边距 盒子周围空隙变大

Font：font-width（加粗0-700） 字体大小（size）/字体行距（line-height（div中行距等于div的高，文字居中） 字体font-falimy）

List-style（列表样式）：

List-style-image：url（../images.jpg）覆盖预设标记用图代替

List-style-type：square（项目符号实心方快）

List-style-position：inside（项目标记放在文本以内）

### CSS背景

* Background-color 属性不能被继承

可以为所有元素设置背景色

如果一个元素没有指定背景色，那么背景就是透明的，这样其祖先元素的背景才能可见。

* Background-imge 背景图片

要设置一个图片做背景 属性里必须有url：（）；

1. Body{
2. Backgrpnd-image：url（路径）；
3. }

背景图像平铺background-repeat

Repeat：纵向横向都平铺（默认属性）

Repeat-x：水平平铺

Repeat-y：垂直平铺

No-repeat：不平铺

* 背景定位

Background-position：center/left/right/bottom/top

改变图像在背景中的位置

如果属性是单个关键字:另一个默认是center

* 属性值可以为数字

100px 100px 表示从左上角0.0 向下移动100px，向右移动100px

* 属性值也可以是百分比

10% 10% 表示从左上角0.0 向下移动10%，向右移动10%

**百分数值同时应用于元素和图像**

* 固定背景Background-attachment：fixed

放置因背景图片过小而引起的滚动条滚动后没有背景图片

### CSS文本

* CSS 文本属性可定义文本的外观。

通过文本属性，您可以改变文本的颜色、字符间距，对齐文本，装饰文本，对文本进行缩进，等等。

* Text-indent缩进文本：所有元素的第一行都可以缩进一个给定的长度，甚至该长度可以是负值。可以为百分比数值，**首行缩进效果**

◆可以为所有块级元素应用 text-indent，但无法将该属性应用于行内元素，图像之类的替换元素上也无法应用 text-indent 属性 ，不过，如果一个块级元素（比如段落）的首行中有一个图像，它**会随该行的其余文本移动。**

**◆** 如果想把一个行内元素的第一行“缩进”，可以用左内边距或外边距创造这种效果。

Text-indent属性可继承：

* Text-align水平对齐：left/center/right

Justify：两端对齐

* + 将块级元素或表元素居中，要通过在这些元素上适当地设置左、右外边距来实现。

◆Text-center与<center>

<CENTER> 不仅影响文本，还会把整个元素居中。text-align 不会控制元素的对齐，而只影响内部内容。元素本身不会从一段移到另一端，只是其中的文本受影响。

* 间隔

Word-spacing：字体（单词）之间的间隔

Letter-spacing：字母之间的间隔

* 字符转换

Text-transform：none/uppercase(转换大写)/lowercase(转换小写)/capitalize(首字母大写)

* 文本装饰text-decoration：none/underline/overline/line-through/blink

Underline：下划线

Overline：上划线

Line-through：删除线

Blink：文本闪烁

★同时设置两个相同的属性：后面的那个生效

可以为超链接去除下划线

1. a{
2. text-decortion:none;
3. }

* 文本方向direction:lrt/rtl
* Line-height:行高，可以垂直居中 行高=元素高度

### CSS字体

* Font-famil：字体系列
* 字体风格font-style：normal/italic/pblique/inherit（继承）

斜体（italic）是一种简单的字体风格，对每个字母的结构有一些小改动，来反映变化的外观。与此不同，倾斜（oblique）文本则是正常竖直文本的一个倾斜版本。

通常情况下，italic 和 oblique 文本在 web 浏览器中看上去完全一样。

* 字体变形

Font-variant：small-capa；小型大写字母

* 字体加粗

Font-weight：100-900（400等价于正常，700等价于blod）

* 字体大小size（默认大小是16px=1em）

绝对大小px

相对大小em（相对于上级元素按倍数调整，***w3c推荐使用em设置字体大小***）

### Css链接

链接的四种状态

* a:link - 普通的、未被访问的链接
* a:visited - 用户已访问的链接
* a:hover - 鼠标指针位于链接的上方
* a:active - 链接被点击的时刻

### CSS列表

* CSS 列表属性允许你放置、改变列表项标志，或者将图像作为列表项标志。

List-style-type：项目符号。Square（方块）disc（实心圆）cricle（空心圆）

List-style-image：项目符合改图片

List-style-position：项目符号位置,inside(在里面)/outside(默认值，在外面，inherit继承)

List-style：项目图片 项目符号 项目符号位置

### CSS表格

* **CSS 表格属性可以帮助您极大地改善表格的外观**
* **表格边框：border：宽度 类型 颜色；**
* **Border-collapse：**将表格边框折叠为单一边框：
* **Width、height：表格宽和高**
* **Text-align、vertical-align：表格文本水平和垂直对齐**
* **Padding、margin：内外边距**
* **Background-color、color：背景颜色、字体颜色**

### **CSS轮廓**

* **Outline轮廓**：是绘制于元素周围的一条线，位于边框边缘的外围，可起到突出元素的作用。
* CSS outline 属性规定元素轮廓的样式、颜色和宽度。
* Outline：颜色 样式 宽度

## CSS样式表属性（CSS样式2）

### 字体Font

* **Font**
* Font：style | small-caps(小型大写字母) | weight | size/ling-height | family
* 使用复合属性必须按照如上的顺序来写属性值
* 每个参数仅允许有一个值
* 忽略的属性将使用该属性的默认值
* **Color**
* Color：#(FFFFFF)/六位十六进制数

#(FFF)/三位十六进制书

Rgb（255，255，255）RGB模式

Rgba（255，255，255，0.5）RGBA模式a（0-1）

1：完全不透明

0：完全透明

* **Text-decoration：字体装饰**
* Underline：下划线
* Overline：上划线
* Line-through：删除线
* None：无装饰
* 有href属性的a标签默认属性是undeline
* 没有文本对象的内容此属性不起作用
* Blink：闪烁（不知道其作用是什么）
* **Text-shadow：字体阴影**
* Text-shadow：color | length（水平延伸距离）| length（垂直延伸距离）| opacity（模糊效果的程度，数值越大越模糊）
* Opacity不可为负值
* Length可为负值
* 可以被用于伪类[:first-letter](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/p_firstletter.html)和 [:first-line](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/p_firstline.html)。
* 可以设定多组效果，方式是用逗号隔开。
* **Font-variant：small-caps小型大写字母**
* Normal 正常的字体
* **Text-transform：字母大小写转换**
* None：不转换
* Uppercase：转换为大写
* Lowercase：转换为小写
* Capitalize：首字母大写
* **Letter(word)-spacing：文字（单词）之间的间隔**
* Normal：默认间隔
* Length：数值，允许为负值，由浮点数或者单位标识组成

### 文本Text

* **Text-lntent：首行缩进**
* Length：百分比数字|由浮点数字和单位标识符组成的长度值，允许为负值。
* 在被另一个对象（如）断开的对象内不能应用本属性。
* **Text-overflow：设置或检索是否使用一个省略标记（...）标示对象内文本的溢出。**
* Clip：不显示省略标记（…），而是简单的裁切
* Ellipsis：当对象文本溢出时显示 省略标记（…）
* **Text-align：文本对齐**
* Left/right/center/justify
* Justify：两端对其
* **Layout-flow:文本流动的方向**
* Horizontal：正常的从左往右
* vertical-ideographic **:** 对象中的内容自上而下流入，下一行在前一行左面。
* **direction：文本流动方向**
* ltr | rtl | inherit ：从左往右 | 从右往左 | 继承
* **unicode-bidi:** 用于同一个页面里存在从不同方向读进的文本显示。与[direction](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_direction.html)属性一起使用。
* unicode-bidi : **normal** | bidi-override | embed
* **normal :** 对象不打开附加的嵌入层，对于内联要素，隐式重排序跨对象边界进行工作
* **embed :** 对象打开附加的嵌入层，[direction](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_direction.html)属性的值指定嵌入层，在对象内部进行隐式重排序
* **bidi-override :** 严格按照[direction](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_direction.html)属性的值重排序。忽略隐式双向运算规则

1. direction: rtl;
2. unicode-bidi:bidi-override;

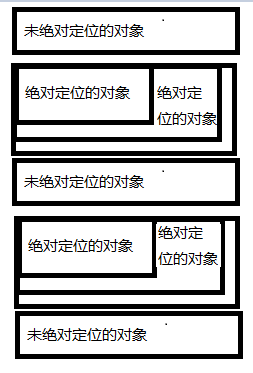
* 以上代码可实现，汉字从右往左流动
* **Work-break：文本换行**
* word-break : normal | break-all | keep-all
* normal：正常换行
* break-all：允许非亚洲语言文本行的**任意字内断开**
* keep-all：对于中文，韩文，日文，不允许字断开
* **write-space：对象对空格的处理**
* white-space : normal | pre | nowrap
* normal：默认处理方式，多个合成一个
* pre：按照代码内输入的输出，用等宽字体显示预先格式化的文本。不合并字间的空白距离和进行两端对齐。
* nowrap：强制在同一行内显示所有文本，直到文本结束或者遭遇br对象
* **所有的text标签**

1. [text-indent](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_textindent.html) [text-overflow](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_textoverflow.html) [vertical-align](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_verticalalign.html) [text-align](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_textalign.html) [layout-flow](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutflow.html) [writing-mode](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_writingmode.html) [direction](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_direction.html) [unicode-bidi](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_unicodebidi.html) [word-break](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_wordbreak.html) [line-break](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_linebreak.html) [white-space](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_whitespace.html) [word-wrap](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_wordwrap.html) [text-autospace](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_textautospace.html)
2. [text-kashida-space](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_textkashidaspace.html) [text-justify](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_textjustify.html) [ruby-align](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_rubyalign.html) [ruby-position](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_rubyposition.html) [ruby-overhang](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_rubyoverhang.html)
3. [ime-mode](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_imemode.html)
4. [layout-grid](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutgrid.html) [layout-grid-char](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutgridchar.html) [layout-grid-char-spacing](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutgridcharspacing.html) [layout-grid-line](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutgridline.html)
5. [layout-grid-mode](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutgridmode.html) [layout-grid-type](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_layoutgridtype.html)

### 背景background

* **Background**
* **background :** [**background-color**](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_backgroundcolor.html) || [**background-image**](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_backgroundimage.html) || [**background-repeat**](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_backgroundrepeat.html) || [**background-attachment**](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_backgroundattachment.html) || [**background-position**](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_backgroundposition.html)
* 复合属性：颜色 | 图片 | 重复 | 滚动 | 定位
* 默认值为：transparent（透明） none repeat scroll 0% 0%。
* **Background-attachment：scroll/fixed 背景是否滚动**
* Scroll：随滚动条滚动
* Fixed：不随滚动条滚动，固定背景
* **Background-color：color/transparent 背景颜色**
* Color：自选颜色
* Transparent：透明色
* **Background-image：none/url（“路径”）背景图片**
* 实用绝对地址或者相对地址指定背景图片
* **Background-position：length length图片的位置**
* 百分百/浮点数 百分百/浮点数：
* Top/left/right/bottom/center：用位置表现图片的位置
* Background-positionX：在X轴上的位置
* Background-positionY：在Y轴上的位置
* **Background-repeat：repeat | no-repeat | repeat-x | repeat-y背景平铺**
* **repeat :** 背景图像在纵向和横向上平铺（默认选项）
* **no-repeat :** 背景图像不平铺
* **repeat-x :** 背景图像在横向上平铺
* **repeat-y :** 背景图像在纵向平铺

### 定位:position

* **Position：relative | absolute | fixed | static**
* Relative：相对定位，对象不可层叠，但将依据[left](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_left.html)，[right](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_right.html)，[top](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_top.html)，[bottom](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_bottom.html)等属性在正常文档流中偏移位置。
*  Absolute：绝对定位，将对象从文档流中拖出，使用[left](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_left.html)，[right](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_right.html)，[top](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_top.html)，[bottom](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_bottom.html)等属性进行绝对定位。而其层叠通过[z-index](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_zindex.html)属性定义。此时对象不具有[边距](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_margin.html)，但仍有[补白](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_padding.html)和[边框](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_border.html)，绝对定位也占据空间，如果多个绝对定位的对象在一起，可重叠，但是如果文档流有一个不是绝对定位的，那么下面的绝对定位的要被挤下去，去占下一个空间，这个不是绝对定位的要单独占一个空间
* Fixed：固定定位：
* Static：默认流式定位
* **Z-index：auto | number层叠顺序**
* Auto：遵从父对象的定位顺序
* Number：无单位的整数值，可为负数
* 如两个绝对定位对象的此属性具有同样的值，那么将依据它们在HTML文档中声明的顺序层叠

### 尺寸：dimensions

* **Height/wight：auto | number**
* 按照样式表的规则，对象的实际高度为其下列属性值之和：
* [margin-top](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_margintop.html) + [border-top](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_bordertop.html) + [padding-top](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_paddingtop.html) + height + [padding-bottom](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_paddingbottom.html) + [border-bottom](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_borderbottom.html) + [margin-bottom](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_marginbottom.html)
* Min/max-height：最小/最大高度
* 如果min设置的比max大，将会自动转为max

### 布局：layout

1. **Clear：none | both | left | right**
   1. None：允许两边都可以有浮动
   2. Both：不允许有浮动
   3. Left/right：不允许左边/右边有浮动
2. **Float：none | left | right**
   1. None：无浮动
   2. Left：左浮动
   3. Right：右浮动
   4. 当该属性不等于none引起对象浮动时，对象将被视作块对象(block-level)，即[display](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_display.html)属性等于block。也就是说，浮动对象的[display](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_display.html)特性将被忽略。
3. **Clip：auto | rect（上 右 下 左）；**
   1. 设置对象的可视区域。区域外的部分是透明的。
   2. Auto：对象无剪切
   3. Rect（**上 右 下 左**）；设置对象左上角为(0,0)坐标计算的四个偏移数值，其中任一数值都可用auto替换，即此边不剪切，（设置前两个就行，后俩设置成auto）
   4. 必须将[position](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_position.html)的值设为absolute，此属性方可使用。
4. **Overflow：设置当对象的内容超过其指定高度及宽度时如何管理内容。**
   1. Visible：不剪切内容也不添加滚动条，[clip](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_clip.html%3e)属性设置将失效
   2. Auto：此为body对象和textarea的默认值。在需要时剪切内容并添加滚动条
   3. Hidden：不显示超过对象尺寸的内容
   4. Scroll：总是显示滚动条
   5. Overflow-X/Y：水平/垂直方向上的；
   6. 对于table来说，假如[table-layout](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_tablelayout.html)属性设置为fixed，则td对象支持带有默认值为hidden的overflow属性。如果设为hidden，scroll或者auto，那么超出td尺寸的内容将被剪切。如果设为visible，将导致额外的文本溢出到右边或左边（视[direction](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_direction.html)属性设置而定）的单元格。
5. **Disaplay： 固定属性应该生成的框类型**
   1. Block：区块元素，div，p，ul，li，可设置宽高，独占一行
   2. Inline：行内元素，a，span，strong，不可设置宽高，在行内显示
   3. Inline-block：行内块元素，具备区块元素和行内元素所有的特征，在行 内按照文档流排布
   4. Inherit：继承父级元素的display属性
6. **Visibility：元素是否可见**
   1. Visible：默认值，可见
   2. Visibility：设置元素不可见
   3. Collapse：在表格中使用，隐藏表格的一行或一列，留下空白位置
   4. Inherit：继承父元素的值

### 外边距：margin

1. Margin：auto|length
   1. 由浮点数字和单位标识符组成的长度值 | 百分数。百分数是基于父对象的高度。对于内联对象来说，左右外延边距可以是负数值。
   2. 如果提供全部四个参数值，将按上－右－下－左的顺序作用于四边。
   3. 如果只提供一个，将用于全部的四边。
   4. 如果提供两个，第一个用于上－下，第二个用于左－右。
   5. 如果提供三个，第一个用于上，第二个用于左－右，第三个用于下。
   6. 外延边距始终透明。

### 轮廓：outline

1. Outline：color style width；
2. Outline：设置对象外的轮廓
   1. outline画在[border](mk:@MSITStore:C:\Users\YZZ\Desktop\css参考手册.chm::/css2/c_border.html)外面，并且不一定是矩形。
   2. color：invert（使用背景色的反色）
   3. style：dashed（虚线），dotted（点状线），solid（实线），double（双实线）
   4. width：medium/默认宽度、thin/小于默认宽度、thick/大于默认宽度、length/浮点数值，不可为负值

### 边框

1. Borde
   1. Border：width style color；
   2. color：invert（使用背景色的反色）
   3. style：dashed（虚线），dotted（点状线），solid（实线），double（双实线）
   4. width：medium/默认宽度、thin/小于默认宽度、thick/大于默认宽度、length/浮点数值，不可为负值

### 内边距：padding

1. 设置元素的内边距属性
2. 行内非替换元素上设置的内边距不会影响行高计算；
   1. **替换元素：替换元素是浏览器根据其标签的元素与属性来判断显示具体的内容。**

**● HTML中的<img>、<input>、<textarea>、<select>、<object>都是替换元素，这些元素都没有实际的内容**

* 1. **非替换元素：(X)HTML 的大多数元素是不可替换元素，他们将内容直接告诉浏览器，将其显示出来。**

1. 如果一个元素既有内边距又有背景，从视觉上看可能会延伸到其他行，有可能还会与其他内容重叠。元素的背景会延伸穿过内边距。不允许指定负边距值。
2. **注释：不允许使用负值。**

### 列表属性：list

1. List-style：在一个声明中设置所有的列表属性。
   1. List-style-type：设置列表项目符号
   2. List-style-position：设置列表符号的位置
      1. Inside：列表项目标记放置在文本以内，且环绕文本根据标记对齐。
      2. Outside：默认值。保持标记位于文本的左侧。列表项目标记放置在文本以外，且环绕文本不根据标记对齐。
      3. Inherit：继承
   3. List-style-image：用图片做项目符号
      1. List-style-type：url（“图片路径”）；
   4. Inherit：继承

### 表格属性

1. Border-collapse：规定是否合并表格边框
   1. Separate：默认值。，边框分开
   2. Collapse：合并边框线
   3. Inherit：继承
2. Border-spacing：规定表格的外边距
   1. Length length：规定相邻单元的边框之间的距离。使用 px、cm 等单位。不允许使用负值。
      1. 如果定义一个 length 参数，那么定义的是水平和垂直间距。
      2. 如果定义两个 length 参数，那么第一个设置水平间距，而第二个设置垂直间距。
   2. Inherit：继承
3. Caption：定义表格标题
4. Caption-side：top/bottom/inherit；规定表格标题的位置
5. Empty-cells：隐藏表格中空单元格的边框和背景
   1. Show：默认值；显示空单元格的边框
   2. Hide：隐藏空单元格的边框
   3. Inherit：继承
6. **Table-layout：规定表格的布局算法(没搞明白)**
   1. Automatic：默认值，列宽度有单元格内容设定
   2. Fixed：列表由列宽度和宽度决定

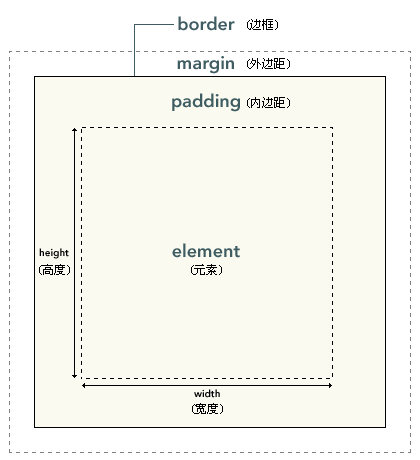
### 多列属性：columus

1. **Columus-count:**
   1. **Internet Explorer 10 和 Opera 支持 column-count 属性。**
   2. **Firefox 支持替代的 -moz-column-count 属性。**
   3. **Safari 和 Chrome 支持替代的 -webkit-column-count 属性。**
   4. Columus：number
      1. 元素内容被划分的列数
2. Columus-gap:两列之间的距离

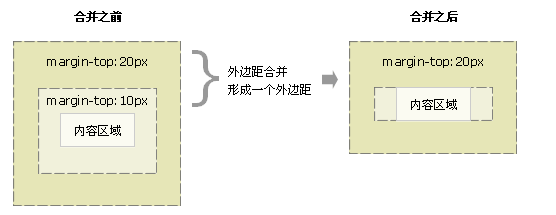
## CSS框模型

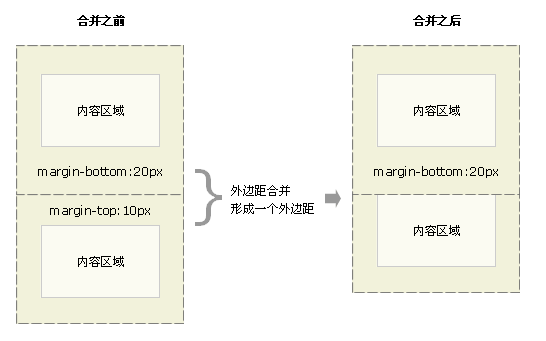
* **CSS 框模型 (Box Model) 规定了元素框处理元素内容、**[**内边距**](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/css/css_padding.asp)**、**[**边框**](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/css/css_border.asp) **和** [**外边距**](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/css/css_margin.asp) **的方式。**
* 元素框的最内部分是实际的内容，直接包围内容的是内边距。内边距呈现了元素的背景。内边距的边缘是边框。边框以外是外边距，外边距默认是透明的，因此不会遮挡其后的任何元素。
* 背景应用于由内容和内边距、边框组成的区域。
* width 和 height 指的是内容区域的宽度和高度。增加内边距、边框和外边距不会影响内容区域的尺寸，但是**会增加元素框的总尺寸。**
* 内边距、边框和外边距可以应用于一个元素的所有边，也可以应用于单独的边。
* 外边距可以是负值，而且在很多情况下都要使用负值的外边距。

★ **不要给元素添加具有指定宽度的内边距，而是尝试将内边距或外边距添加到元素的父元素和子元素。**



* **Padding内边距**：元素的内边距在边框和内容区之间。控制该区域最简单的属性是 padding 属性。
* CSS padding 属性定义元素边框与元素内容之间的空白区域。
* padding 属性接受长度值或百分比值，但**不允许使用负值**。
* 可以单独设置top/bottom/left/right的内边距
* **百分数值是相对于其父元素的 width 计算的**，这一点与外边距一样。所以，如果父元素的 width 改变，它们也会改变。
* **CSS边框border**
* 元素的边框 (border) 是围绕元素内容和内边距的一条或多条线。
* CSS border 属性允许你规定元素边框的样式、宽度和颜色。
* Border：宽度 样式 颜色；
* **边框绘制在“元素的背景之上”。**
* **透明边框：**边框颜色值 transparent
* 利用 transparent，使用边框就像是额外的内边距一样；此外还有一个好处，就是能在你需要的时候使其可见。这种透明边框相当于内边距，因为元素的背景会延伸到边框区域（如果有可见背景的话）。
* 边框类型:solid（单线）、double（双线）、none（无边框）、dotted（点状线）、dashed（虚线）
* **CSS外边距**
* 围绕在元素边框的空白区域是外边距。设置外边距会在元素外创建额外的“空白”。
* 设置外边距的最简单的方法就是使用 margin 属性，这个属性接受任何长度单位、百分数值甚至负值。
* 百分数是相对于父元素的 width 计算的。
* 外边距合并
* 外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。
* **合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者**。

****

**当一个元素包含在另一个元素中时（假设没有内边距或边框把外边距分隔开），它们的上和/或下外边距也会发生合并**

* **只有普通文档流中块框的垂直外边距才会发生外边距合并。行内框、浮动框或绝对定位之间的外边距不会合并。**

## CSS定位

### CSS 定位 (Position) 属性允许你对元素进行定位。

* CSS定位和浮动：CSS 为定位和浮动提供了一些属性，利用这些属性，可以建立列式布局，将布局的一部分与另一部分重叠，还可以完成多年来通常需要使用多个表格才能完成的任务。
* 定位的基本思想很简单，它允许你定义元素框相对于其正常位置应该出现的位置，或者相对于父元素、另一个元素甚至浏览器窗口本身的位置。
* 浮动不完全是定位，不过，它当然也不是正常流布局。
* div、h1 或 p 元素常常被称为块级元素。这意味着这些元素显示为*一块内容*，即“块框”。与之相反，span 和 strong 等元素称为“行内元素”，这是因为它们的内容显示在行中，即“行内框”。
* 您可以使用 [display 属性](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/cssref/pr_class_display.asp)改变生成的框的类型。这意味着，通过将 display 属性设置为 block，可以让行内元素（比如 <a> 元素）表现得像块级元素一样。还可以通过把 display 设置为 none，让生成的元素根本没有框。这样的话，该框及其所有内容就不再显示，不占用文档中的空间。
* 但是在一种情况下，即使没有进行显式定义，也会创建块级元素。这种情况发生在把一些文本添加到一个块级元素（比如 div）的开头。即使没有把这些文本定义为段落，它也会被当作段落对待：

1. <div>
2. some text
3. <p>Some more text.</p>
4. </div>

* 在这种情况下，这个框称为无名块框，因为它不与专门定义的元素相关联。

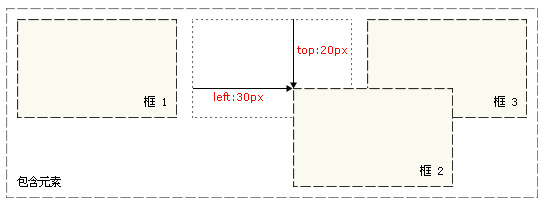
### CSS定位机制

* CSS 有三种基本的定位机制：**普通流、浮动和绝对定位**
* 除非专门指定，否则所有框都在普通流中定位。
* 块级框从上到下一个接一个地排列，框之间的垂直距离是由框的垂直外边距计算出来。
* 行内框在一行中水平布置。可以使用水平内边距、边框和外边距调整它们的间距。但是，**垂直内边距、边框和外边距不影响行内框的高度。**由一行形成的水平框称为*行框（Line Box）*，行框的高度总是足以容纳它包含的所有行内框。不过，***设置行高可以增加这个框的高度***。

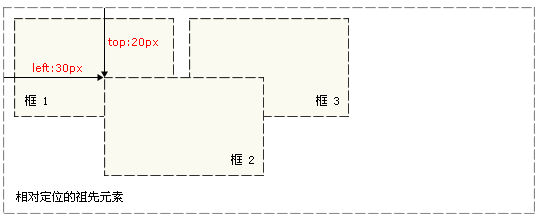
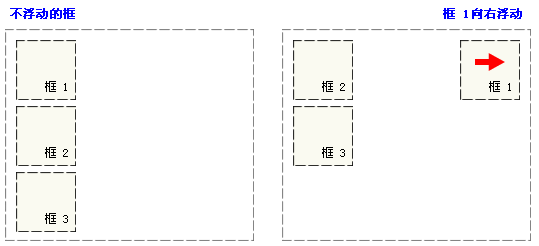
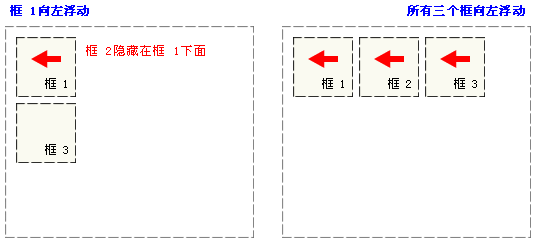
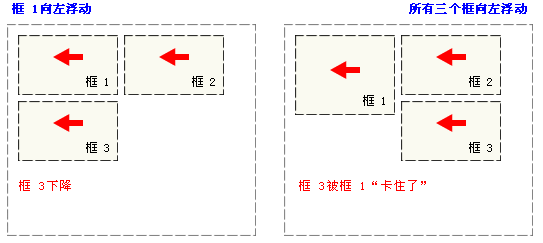
### CSS position属性

* Static：元素框正常生成。块级元素生成一个矩形框，作为文档流的一部分，行内元素则会创建一个或多个行框，置于其父元素中。
* **Relative：（相对定位）**元素框偏移某个距离。元素仍保持其未定位前的形状，它原本所占的空间仍保留。
* **Absolute：（绝对定位）**元素框从文档流完全删除，并相对于其包含块定位。包含块可能是文档中的另一个元素或者是初始包含块。元素原先在正常文档流中所占的空间会关闭，就好像元素原来不存在一样。元素定位后生成一个块级框，而不论原来它在正常流中生成何种类型的框。
* **Fixed：（固定定位）**元素框的表现类似于将 position 设置为 absolute，不过其包含块是视窗本身。
* 相对定位实际上被看作普通流定位模型的一部分，因为元素的位置相对于它在普通流中的位置。

### CSS相对定位

* 对一个元素进行相对定位，它将出现在它所在的位置上。然后，可以通过设置垂直或水平位置，让这个元素“相对于”它的起点进行移动。
* 在使用相对定位时，无论是否进行移动，**元素仍然占据原来的空间。**因此，**移动元素会导致它覆盖其它框**。

### CSS绝对定位

* 设置为绝对定位的元素框从文档流完全删除，并相对于其父级块定位，父级块可能是文档中的另一个元素或者是初始包含块。元素原先在正常文档流中所占的空间会关闭，就好像该元素原来不存在一样。元素定位后生成一个块级框，而不论原来它在正常流中生成何种类型的框。
* 绝对定位使元素的位置与文档流无关，因此不占据空间。这一点与相对定位不同，相对定位实际上被看作普通流定位模型的一部分，因为元素的位置相对于它在普通流中的位置。
* **绝对定位的元素的位置相对于*最近的已定位祖先元素*，如果元素没有已定位的祖先元素，那么它的位置相对于*最初的包含块*。**
* 因为绝对定位的框与文档流无关，所以它们可以覆盖页面上的其它元素。可以通过设置 [z-index 属性](mk:@MSITStore:C:\Users\YZZ\Desktop\HTML手册.chm::/www.w3school.com.cn/cssref/pr_pos_z-index.asp)来控制这些框的堆放次序。
* CSS浮动：float：left/right/none/inherit
* 浮动的框可以向左或向右移动，直到它的外边缘碰到包含框或另一个浮动框的边框为止。
* 由于浮动框不在文档的普通流中，所以文档的普通流中的块框表现得就像浮动框不存在一样。
* 当把框 1 向右浮动时，它脱离文档流并且向右移动，直到它的右边缘碰到包含框的右边缘：
* 当框 1 向左浮动时，它脱离文档流并且向左移动，直到它的左边缘碰到包含框的左边缘。因为它不再处于文档流中，所以它不占据空间，实际上覆盖住了框 2，使框 2 从视图中消失。
* 如果把所有三个框都向左移动，那么框 1 向左浮动直到碰到包含框，另外两个框向左浮动直到碰到前一个浮动框。
* 如果包含框太窄，无法容纳水平排列的三个浮动元素，那么其它浮动块向下移动，直到有足够的空间。如果浮动元素的高度不同，那么当它们向下移动时可能被其它浮动元素“卡住”：
* Clear：left/right/both/none/inherit
* Left：不允许左浮动
* Right：不允许右浮动
* Both：左右都不允许浮动
* None：默认值，允许浮动元素出现在左右两侧
* Inherit：继承

## CSS选择器

\* 通配符，选择页面所有元素

权重计算：

行内样式 1000 ID选择器 100 类选择器 10 标签选择器 1

**！important >行内样式** > id选择器 > 类选择器 > 标签选择器

### **基本选择器**

* Id选择器

i**d 选择器可以为标有特定 id 的 HTML 元素指定特定的样式**

id选择器以“#”来定义

每个元素只能有一个ID

在现代布局中，id选择器多用来简历派生选择器

* 类选择器

类名的第一个字符不能使用数字

和 id 一样，class 也可被用作派生选择器**：**

**一个元素可以有多个类名，类名之间用空格隔开**

### 特殊选择器

* 包含选择器（派生选择器、后代选择器）：div p{}

选择div中所有的p标签（div和p用一个空格隔开）

* 子选择器（对象选择器）：div > p{}

只选择div中直属与div的p，不会选择后代的P

IE7一下不支持

* 相邻选择器：div + p

选择和div相邻的拥有同一个父级标签的p标签

* 属性选择器
* E [ class ]：匹配具有class属性的元素
* E [ class = “value” ]：匹配具有class=“value”的元素
* E [ class ~= “value” ]：匹配具有class并且属性值是空格符隔开的字段，其中一个字段等于value。
* E [ class |= “value” ] 匹配具有class并且属性值是连字符“-”隔开的字段，其中一个字段等于value。

## 伪对象

：after ：berofr

# HTML第二天 2016.07.29

### !+tab html5文档类型

### Charset 编码格式:和计算机交流，翻译人的语言给计算机听

* <meta charset=”utf-8”>
* ASCII码、Ansi码、Unicode、Gbk 汉字、Gb2315 汉字、Big5繁体字符
* Utf-8 :最大的字符集，包含一百多个国家的文字

### Keywords，关键字,给搜索引擎看的

* <meta name=”keywords” content=”关键字”/>

### 网页描述 description

* <meta name=”description” content=”描述内容”>

### 网页重定向

* http-equiv=”refresh” content=”5;(加分号，跳转时间,单位秒) [www.baidu.com](http://www.baidu.com)(重新定向的网页)”

### link：链接外部资源

* 链接外部CSS文件(stylesheet)、JS文件(script)、icon图标(icon)

### Table表格

* 展示数据
* 对网页重构（css+div）的一个有益补充
* Cellspacing：单元格之间的距离，会撑大表格大小
* Width、height、
* Cellpadding：内容到单元格边框的距离，会撑大单元格大小
* border-collapse:collapse；：合并单元格边框
* Align：left、center、right；表格对齐方式
* 给table加align是确定表格的对齐方式
* 给tr、td加align是确定行、单元格的对齐
* Bgcolor：背景颜色
* Caption：定义表格标题，紧跟table标签，居中显示
* Colspan：合并单元格\_行，不可以和表头表尾合并
* Rowspan：合并单元格\_列，不可以和表头表尾合并
* Valign：top、middle、center 垂直对齐方式
* HTML 表单中有两种类型的单元格：
* 表头单元格 - 包含表头信息（由 th 元素创建）
* 标准单元格 - 包含数据（由 td 元素创建）
* th 元素内部的文本通常会呈现为**居中的粗体文本**，
* td 元素内的文本通常是左对齐的普通文本。
* bordercolor=“red”：改变表格边框颜色

1. <table> //表格标准结构
2. <thead> //表格头部
3. <tr>
4. <td><td>
5. </tr>
6. </thead>
7. <tbody> //表格主题
8. <tr>
9. <td><td>
10. </tr>
11. </tbody>
12. <tfoot> //表格尾部
13. <tr>
14. <td><td>
15. </tr>
16. <tfoot>
17. </table>

### 表单form

* 作用:收集用户输入信息
* **表单的组成**
* 1.提示信息
* 2.表单控件
* 3.表单域：输入内容的区域
* **<form action=”提交到地址” method=”get/post””>**
* Action:处理信息
* Method：提交方法
* Get：通过地址栏提交信息，安全性差
* Post：通过文件提交信息，安全性高
* **Input标签：**
* password：密码输入框
* Text：文本输入框

maxlength：最大输入字符

readonly：只读（不能编辑）

disabled：输入框未激活，不可用

name：命名输入框，提交值的时候区分每个输入框

value：初始值，默认显示值，将默认值传给处理文件

* Radio：单选框

Checked=checked：默认选中项

必须使用相同的name，否则会使两项都可以被选中

**Submit/reset：提交/重置 按钮**

* Select>option：下拉列表

Selectked：selectked：默认选项

Multiple：multiple 可多选

Size：定义列表显示的项

Optgroup：对option进行分组

**Hidden=”hidden”：隐藏控件**

1. <select multiple=“multiple” size=“2”>
2. <optgroup liable=”组名”>
3. <option></option>
4. <option></option>
5. <option></option>
6. </optgroup>
7. </select>

* Checkbox：多选框

Checked：checked；默认选项

* **Textarea：多行文本框 单独标签，不需要input**

Cols：每行可输入的长度

Rows：可输入的行数

1. <textarea colw=”30” rows=”10”>

* file文件上传控件
* 按钮

Submit：提交按钮

Button：普通按钮，value=“按钮上的文字”

Image：图片按钮可实现提交功能

1. <input type=”image” src=”路径”>

Reset：重置按钮，将信息重置到默认状态

* Fieldset：表单信息分组

Legend：表单信息分组名称

* **HTML5表单标签**

<Input type=“url”>：网址提交；

<Input type=“date”>：显示日期控件

<Input type=“time”>：时间控件

<input type=”emial”>：邮件控件

<input type=”number” step=“5”(默认1，值是几每次跳几)>：数字控件

<input type=”range” step=“5”>：滑块控件

Step：默认1，值是几每次跳几

* **标签语义化**

根据内容的结构化（内容语义化），选择合适的标签（代码语义化）

好的语义化的网站**标准是去掉样式表后，结构依然很清晰**

* 标签语义化意义
  1. 网页结构合理
  2. 有利于seo和搜索引擎建立良好沟通，有了良好的结构和语义你的网页内容自然容易被搜索引擎抓取。
  3. 方便其他设备解析（如屏幕阅读器、盲人阅读器、移动设备）
  4. 便于团队开发和维护
* 标签语义化（注意事项）
  1. 尽可能少的使用无语义的标签div和span；
  2. 在语义不明显时，既可以使用div和p时，尽量用p，因为p在默认情况下有上下间距，对兼容特殊终端有利。
  3. 不要实用纯样式标签，如：b / font / u等，改用CSS设置
  4. 需要强调的文本，可以包含在strong或者em标签中，strong默认样式是加粗（不要b），em是斜体（不用i）。

# CSS第一天 2016.07.30

### CSS概念

* CSS（Cascading Style Sheets）：层叠样式表（级联样式表）
* CSS使得结构与样式分离
* 用来美化HTML标签，相当于给页面化妆

|  |  |
| --- | --- |
| **属性** | 解释 |
| **Width：20px；** | 宽 |
| **Height：20px；** | 高 |
| **Background-color：red、#f00、#ff0000** | 背景颜色 |
| **Font-size：20px；** | 文字大小 |
| **Text-align：center、left、right** | 内容水平对齐方式 |
| **Text-indent：2em** | 首行缩进 |
| **Color：red** | 文本颜色 |

### 选择器

* 写法

选择器是选择谁（标签）的过程

1. 选择器{属性：值；}

* 外链样式表

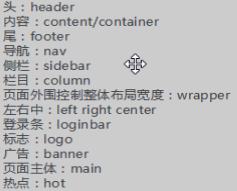
1. <link rel=”stylesheet” href=”路径” type=“text/css”>

* 内部样式表

1. <style type=”text/css” >
2. 选择器{属性：值；}
3. </style>

* 内联样式

1. <p style=“属性一：值一”>行内样式</p>

* 基础选择器
  1. 标签选择器
     1. 标签｛属性：值；｝
     2. 选择所有相同标签名字的标签
     3. 所有被选中标签都会执行相同的样式
  2. ***类选择器***
     1. .自定义类名{属性：值；…… 属性：值；}
     2. 一个标签可以调用多个类选择器
     3. 多个标签可以共用一个选择器
     4. 类命名规则
        1. 不能用纯数字，不能以数字开头
        2. 不能实用特殊符号，和特殊符号开头，**除 “\_” 外**
        3. 不建议使用汉字来定义类名
        4. 不推荐使用属性和属性值来做类名
  3. ID选择器
     1. #自定义ID名｛属性：值；…… 属性：值；｝
     2. 一个ID选择器在一个页面只能使用一次，如果使用大于一次，不符合W3C规范，JS调用会出问题
     3. 一个标签只能调用一个ID选择器，只能有一个ID
     4. 类选择器和ID选择器不冲突
* 良好的代码：重用、有序、精简
  1. 通配符选择器
     1. \*｛属性：值；······ 属性：值；｝
     2. 选择页面所有标签元素。给所有标签相同的样式
     3. 不推荐使用
* 复合选择器：两个或者两个以上的基础选择器通过不同方式连接在一起
  1. 交集选择器
     1. 标签+类（id）选择器｛属性：值；｝
     2. 既要满足标签选择器，又要满足类（id）选择器
  2. 后代选择器（重点）
     1. 选择器+空格+选择器+···+选择器+空格+选择器｛属性：值｝
     2. 无限制隔代，选择器可自用组合
  3. 子带选择器
     1. 选择器>选择器｛属性：值；｝
     2. 选择直接子代，不选择子代以后的
  4. 并集选择器
     1. 选择器，选择器，选择器…｛属性：值；｝

P标签不能包含p标签

### 颜色的显示方式

* 1. 直接写颜色名称
  2. 十六进制显示颜色
     + 0-9 a-f
     + #FFFFFF：红（red）绿（green）蓝（blue）
     + #FFFFFF：白色 #000000：黑色
  3. RGB显示模式color：rgb（255，255，255）；
  4. RGBA显示模式 color：rgba（255，255，255，0.5）；
     + A是alpha不透明度，可选值为0-1；
     + 0是完全透明，1是完全不透明

### 文本元素

* Font
* Font-size:文字大小
* Font-weight：值100-900，700约等于blod（不推荐用），文字粗细
* Font-family：字体，最常用字体（微软雅黑、宋体）
* Font-style：字体风格italic/斜体、normal/正常
* Line-height：行高

★ font：weight style size/ling-height family；

* 复合属性*字体大小*和*字体*必不可少
* 文字的表达方式

1. 直接写文字
2. 写字体的英文名称
3. 写字体的Unicode
4. 微软雅黑：Microsoft yahei
5. 宋体：SimSun
6. *在浏览器console（控制台输入）Escape（“字体名”）然后回车，得到字体Unicode编码*

定义宽切不浮动，Margin：0 aoto； div居中 上下边距0，左右边距自动平均分配

# CSS第二天 2016.08.01

### CSS的书写位置

* **内联样式：写在head内部**
* 只能作用当前文件
* 没有真正实现结构表现分离

1. <style type=”text/css”>
2. 选择器｛属性：属性值；｝
3. </style>

* **外链样式：独立的CSS文件，link标签导入**
* 实现表现结构分离
* 作用范围是当前站点，范围广

1. <link rel=”stylesheet” **href**=”CSS文件路径”>

* **italic：斜体**
* **行内样式表：嵌入在行内的**
* 作用范围仅限于当前标签
* 结构表现混在一起
* 不推荐使用

1. <div style=”属性：属性值”></div>

### HTML标签分类（显示方式）：line/block

* **Block：块元素**
* Div/p/h1-h6/li/ul：典型块元素
* 特点：独占一行
* 可定义宽高，不定义宽度，默认宽度是父级元素的宽度
* 不定义高度且无内容，会不显示
* **文字默认大小16px**
* **Line：行内元素**
* **a/span/font/strong/：在一行内显示**
* 不能直接定义宽高
* 元素的宽高就是内容的宽高
* **Line-block:行内块元素**
* Imput/image：在一行显示，可设置宽高

### 块/行内元素之间的转换

* **块元素转换为行内元素**
* Display：inlne
* 具备所有行内元素的特点
* **行内元素转换为块元素**
* Display：block；
* 具备所有快元素的特点
* ***块/行内元素转行内块***
* Display：inline-block；
* 空格、回车都会被解释成一个空格

***每个元素都要转换，不能包含转换，行内元素不能包含块元素***

### CSS特性

* **层叠性**
* 当多个样式作用到同一个标签上，样式发生冲突。
* 按照浏览器从上往下的解释顺序，***后来居上***
* 后面的代码覆盖（层叠）前面的代码
* **继承性**
* Inherit：
* 需要是包含关系
* 文字颜色color、文字大小font-size、文字字体font-family、字体粗细font-weight、文字风格font-style 。
* ***文字的所以属性都可以继承***
* H1-h6显示的不是继承的数值，因为好h1-h6有浏览器定义的倍率（em）大小
* a标签不能继承颜色
* **CSS选择器优先级**
* 默认样式<标签选择器<类样式<ID选择器<行内样式<!important
* 0 1 10 100 1000 1000<
* **继承的属性优先级（权重为0）小于本身的属性**
* 自己为定义属性的时候继承父级属性，本身定义了属性就显示自己的属性
* **权重可以叠加（相加）**

### 链接伪类

1. **A:link｛属性：属性值｝；超链接默认状态样式**
   1. 和a｛属性：属性值｝一样
   2. 清除浏览器缓存
2. **a:visited｛属性：属性值｝；链接访问之后的状态**
3. **a:hover｛属性：属性值｝；鼠标放到链接之上的状态**
4. **a:active｛属性：属性值｝；鼠标激活时的链接状态**
5. **：facus｛属性：属性值｝；获取焦点**

* 文版修饰:text-decoration:none/underline/overline/line-through

### 背景属性：background

1. **Background-color：背景颜色**
2. **Background-image：背景图片，同时设置背景图片和背景颜色，图片覆盖颜色**
   1. Background-image：url（“图片路径”）；
3. **Background-repeat：背景平铺（默认平铺：repeat）**
   1. Repeat：平铺
   2. No-repeat：不平铺
   3. Repeat-X/Y：在X/Y轴上平铺
4. **Background-position：背景定位**
   1. 属性值：Left/right/center/top/bottom
   2. 特点：
      1. 当你只写一个值的时候另一个值默认是centr
      2. 写两个值的时候，值的顺序无要求
      3. 写两个具体值的时候，第一个值代表水平方向，第二个值代表垂直方向
5. **Background-attcchment：背景滚动**
   1. Scroll：滚动，随滚动条而滚动
   2. Fixed：固定，固定位置，不随滚动条滚动
      1. **此属性会使背景定位以浏览器当前页面为参考**
6. **复合属性**
   1. Background：color image repeat position attachment；
   2. ***（url（“路径”））图片必须写***，别的都可以不写
   3. **顺序无要求**

# CSS第三天 2016.08.02

### 行高：line-height

1. **行高是文字基线与基线之间的距离，下图两条红色横线就是基线**
2. **行高=上边距到下边距的距离**
3. **文字大小等于底线到顶线的距**
4. **一行文字行高等于父元素高度的时候，文字垂直居中显示**
5. 浏览器默认字体大小16px
6. **行高的单位**
   1. Px：
   2. Em：
   3. %：
   4. 数值：

|  |  |  |  |
| --- | --- | --- | --- |
| 单位 | 文字大小 | 行高 | 总大小 |
| Px | 20px | 20px | 20px |
| Em | 20px | 2em | 40px |
| % | 20px | 150% | 30px |
| 数值 | 20px | 2 | 40px |

* 1. 除了以像素为单位，行高都是与文字的乘积

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 单位 | 行高 | 父元素 | | 子元素 | | 最终行高 |
|  |  | 文字大小 | 行高大小 | 文字大小 |  |  |
| Px | 40px | 20px | 20px | 30px |  | 40px |
| Em | 2em | 20px | 2em | 30px |  | 40px |
| % | 150% | 20px | 150% | 30px |  | 30px |
| 数值 | 2 | 20px | 2 | 30px |  | 60px |
|  |  |  |  |  |  |  |

* 1. **子元素没有设置行高的时候**
     1. **当行高数值由em、%为单位是，子元素的行高是继承父元素的行高乘字体大小的结果**
     2. **当行高数值是px为单位时，子元素直接继承父元素的行高**
     3. **当行高无单位直接是数值的时候，子元素行高是子元素的文字大小和子元素的行高相乘**
  2. **子元素设置行高的时候使用子元素的行高，不继成**

### DIV盒子模型

1. **盒子的宽度=外边距+边框+内边距+定义宽度**
2. 盒子居中：margin ： 0 auto；
3. **盒子模型：DIV*没有语义***
   1. **内容：content**，主题内容
   2. **内边距：padding**，内容到边框的距离
      1. **内边距会撑大盒子**
      2. **内边距会影响盒子的宽度**
      3. **边框会影响盒子的宽度**
      4. **继承的盒子不会被撑大**
         1. **包含（嵌套） 的盒子，如果子盒子没有定义宽度（宽度从父盒子继承来），给子盒子设置水平内边距（小于父盒子宽度的情况下），一般不会被撑大**
         2. **垂直方向没有此规则**
         3. **垂直方向如果没有设置高度，可用内边距撑开盒子**
   3. **边框：border**，边框的宽度
      1. Border - top/bottom/left/right – width/style/olor
      2. 复合属性：宽度 样式 颜色**（没有顺序限制，样式是必写属性）**
      3. Dotted点线 / dashed虚线 solid实线
      4. 如果边框是虚线，虚线部分也显示背景颜色
      5. **盒子大小要加上边框的宽度**
      6. 边框合并：border-collapse：collapse
      7. **Td的边框覆盖table的边框**
   4. **外边距：margin**，与另外一个盒子的距离
      1. margin-left/bottom/right/top；
      2. **垂直方向上的外边距合并**
         1. **两个盒子一个设置上外边距，一个设置下外边距，取设置相对较大的值**
      3. **嵌套的盒子，垂直方向外边距塌陷（给子盒子设置上外边距后，父盒子会随着子盒子同时向下移动）**
         1. **解决方法**
            1. **给父盒子设置边框border（不推荐使用）**
            2. **给父盒子设置：overflow：hidden**
            3. **（**了解BFC：格式化上下文**）**
      4. margin：上 右 下 左
      5. 写两个值，第一个代表上下，第二个代表左右
      6. 写三个值，第一个代表上，第二三个代表左右，第三个代表下
4. **表单控制案例**
   1. Outline:none 去掉轮廓线
   2. Border：0 none；去掉边框
   3. Label:绑定标签：点击“**显示文字**”可获取光标焦点，也可以执行

：focus伪类，获取焦点发生的样式变换

1. <label for=”绑定标签ID名”>显示文字<label>
2. <input type=”text” id=”ID名”>

#### FireWorks

1. Fireworks的基本实用
   1. Ctrl+R：导入
   2. Ctrl+Alt+R：调出/隐藏标尺
   3. 测量两条辅助线之间的距离：选择指针工具在两条辅助线之间按住Shift键

# CSS第四天 2016/08/04

1. **字体大小要放在行高前面**
2. **文档流布局**
   1. 元素自上而下，从左往右，
   2. 块元素独占一行，每个控件后有一个换行符
   3. 行内元素在行内显示，达到碰到父元素边框换行

### 浮动：float

* 1. 让盒子（div）在一行显示
  2. 标签浮动之后脱离了文档流，**不占据原来的位置**
  3. **行内元素设置浮动之后（不推荐使用，因为浮动之后会脱离标准流，最好使用display：inline-block）**
     1. 能定义宽高
  4. 左浮动：left
  5. 右浮动：right
  6. **浮动的作用**
     1. **文本环绕图片**
        1. 文字和浮动元素没有层叠关系
        2. 文字不参与浮动
     2. **制作 导航栏**
        1. 让li在一行显示，设置li浮动，设置ul浮动不可以
        2. 父元素和子元素有相同的宽度时，无法垂直居中（**好像不是这样的**）
        3. Li的高度不能继承，默认高度是内容+2px（上下边框宽度）
     3. 网页布局
        1. **盒子高度不会继承**

### 清除浮动：清楚浮动产生的不利影响，

* 1. 当父元素没有定义高度，嵌套的盒子浮动之后，下边的元素位置发生错误。
  2. **清除浮动的方法**
     + 1. **额外标签法；Clear：left/right/both**
          1. 工作中用的最多的是clear：both
          2. 在最后一个标签添加一个标签（用的不多，影响服务器响应速度）

<div class="main">

<div class="content"></div>

<div class="sidebar"></div>

<div style="clear: both"></div>

</div>

* + - 1. **给父元素使用overflow：hidden**
         1. 如果有内容出了盒子，不能使用该方法
         2. BFC快元素格式化上下文
      2. **.clearfix：after 为元素清楚浮动**
      3. .clearfix：after｛
      4. Content：“.”；
      5. Display：block；
      6. Height：0；
      7. Line-height：0；
      8. Visibility：hidden；
      9. Clear：both；
      10. ｝
      11. .clearfix｛
      12. Zoom：1；/\*兼容IE浏览器\*/
      13. ｝

### CSS初始化：为了更好的用户体验

* 1. 为了考虑浏览器的兼容性，不同浏览器标签默认值不同，如果没有CSS对页面进行初始化，会出现在不同浏览器页面显示效果不同
  2. 火狐默认字体：微软雅黑
  3. Chrome默认字体：宋体

### Overflow：处理溢出的内容

* 1. Hidden：溢出的内容将被修剪，不显示
  2. Auto：溢出时出现滚动条，不溢出时正常显示，不显示滚动条
  3. Visible：默认值，不处理溢出内容，正常显示
  4. Scroll：滚动条，右边和下边不管溢出与否都会出现滚动条

### 定位：position

1. **定位的方向：**left、right、top、bottom
2. **静态定位：**static，默认值，按文档流默认位置定位
3. **绝对定位：**absolute，
   1. 使用之后**不占据原来的位置**（脱标）
   2. 元素使用绝对定位，位置是**从浏览器出发**的
   3. 嵌套的盒子，父元素没有使用定位，子元素使用定位，子元素**定位从浏览器出发**
   4. 嵌套的盒子，父元素使用绝对定位，子元素使用绝对定位，子元素定位**从父元素开始**
      1. **给行内元素使用绝对定位（不推荐使用，推荐使用display：inline：block）后，转换为行内块元素，脱标，可以设置宽高**
      2. **Z-index：调节标签层叠顺序（0-999）值越大，元素越在上面**
4. **相对定位：relative**
   1. 相当于自己的初始坐标移动
   2. 仍然占据原来的位置，不脱标（标准流）
   3. **子绝父相（子元素绝对定位，父元素相对定位）**
   4. 对行内元素使用相对定位不能转换为行内块
5. **固定定位：fixed**
   1. 定位后不占据原来位置，脱标
   2. 定位从浏览器出发
   3. 行内元素使用固定定位后，会转换为行内块元素（**不推荐**，**推荐使用display：inline-block**）

### 标签包含规范

◆div可以包含任意标签

◆p标签不能div h1-h6, ul ,可以包含行内元素。

◆hn可以包含div,p,ul等元素。

### 规避脱标流

◆尽量使用标准流。

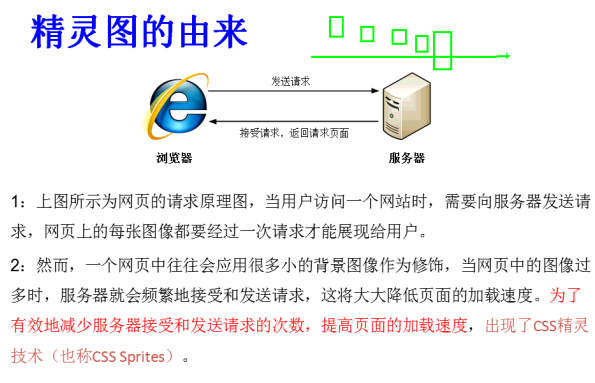
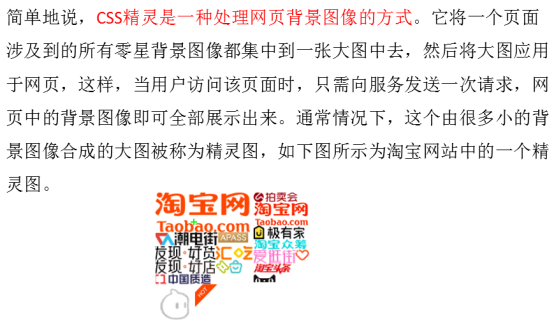
◆标准流解决不了的使用浮动。

◆浮动解决不了的使用，使用定位。

◎使用margin-left（right）:auto;可以将盒子冲到父盒子的右边(左边)。

# CSS第五天 2016/08/05

14期班委群qq群106754216

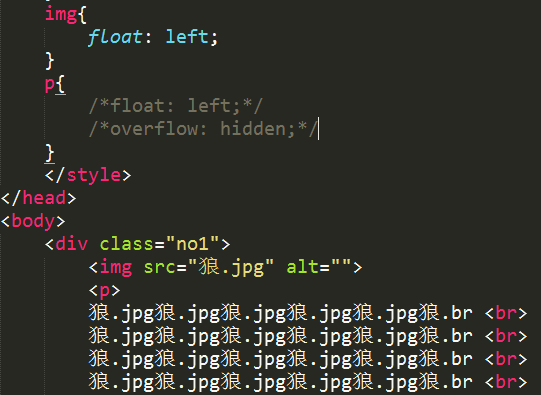
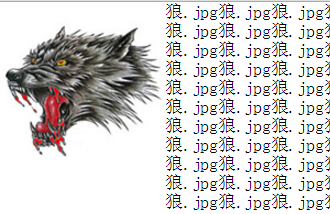
1. **浏览器初始化**
   1. a的下划线：text-decoration：none；
   2. img边框（某些低版本IE会给img自动加边框）：border：0；
2. **已定位的盒子（div）居中显示**
   1. Text-align是内容居中
      1. **Margin：0 auto；是盒子居于父元素中间**
         1. **不能对已定位和盒子使用（因为已定位的盒子脱标了）**
         2. **只能让标准流的盒子居中显示**
   2. **已定位的盒子定位居中方式**
3. Position：absolute；（子绝父相）
4. Left：50%；
5. Margin-left：子元素盒子的一半（负值）
6. **使列表横向显示需注意的问题**
   1. 用display：block-inline；方法使列表li横向显示，每个li后面的回车会产生一个空格，
   2. 使用浮动float：left；的办法就不会产生此问题
7. **定位以最近的已定位的父元素为移动基准**
8. **Border-radius：边框圆角**
9. **标签包含规范**
   1. DIV可包含所有元素
   2. P不能包含DIV、H1，P元素一般只能包含行内元素
   3. H系列系列可以包含DIV、P，但是一般不这么用
   4. 行内元素不推荐包含行内元素（**废话**），一般都这么用
   5. 行内元素不能包含块元素
   6. Li里面可以放置任何元素
10. **规避脱标流**
    1. 尽量使用标准流（文档流）
    2. 当标准流解决不了的时候，使用浮动
    3. 浮动解决不了的时候，使用定位
    4. **Margin-left：aoto；内容靠左显示**
    5. **Margin-right：quto；内容靠右显示**
11. **图片和文字垂直居中**
    1. 图片的下边默认和文字的基线对齐
    2. 对行内块使用
    3. Vertical-align：垂直对齐
       1. Vertical：baseline，基线对齐
       2. Vertical：middle，图片文字垂直居中对齐
       3. Vertical：sub/sup，对齐文字的下标/上标
12. **CSS可见性**
    1. Overflow：hidden：溢出隐藏
       1. 设置一个父盒子溢出隐藏
       2. 设置三个和父元素相同大小、颜色不同的子元素
       3. 锚点链接到每个子元素
       4. 点击链接就链接到不同颜色的子元素
    2. **Visibility：hidden**；隐藏标签
       1. 隐藏后仍旧**占据**原来的位置
    3. **Display：none；**隐藏标签
       1. 隐藏后**不占据原**来的位置
       2. 常配合JS使用
    4. Display：block；元素可见
13. **CSS内容移除（网页优化）**
    1. LOGO一般放在H1标题里面，这样做权重高，有益于SEO
    2. 用text-indent:-5000em；移除文字
    3. 用DIV的padding移除文字
       1. 设置DIV高度为0
       2. 此时DIV中的内容在DIV外面
       3. 给div一个上内边距
       4. 设置DIV的overflow：hidden；
14. **精灵图**
15. 
16. **CSS精灵图技术（也成CSS Sprites）**把许多小图标放在同一张png图片上，根据坐标寻找小图标
    1. 背景默认居左居顶（左上角）显示
    2. 背景图片往左移动，left：负值
    3. 背景图片往上移动，top：负值
    4. 精灵图的制作
       1. 尽量靠一边，方便取坐标值
       2. 尽量减少图片尺寸（内存）大小
       3. 一定要PNG（透明）格式的
17. **属性选择器**
    1. 标签[属性=“属性值”] [属性=“属性值”]…｛属性：属性值；｝
18. **Emment快捷键**
    1. ！+tab 生成H5结构
    2. **^** 往上一级 ^^ 往上升两级
    3. Ul > li.abc$\*4

# CSS第六天 2016/08/07

### 滑动门

1. 背景默认从左上角开始显示
   1. 给背景添加一个right可使背景从右显示
   2. a span：hover｛属性：属性值｝，给A里面的span添加伪类
2. **内容自动撑大盒子，盒子大小自适应宽高度**
   1. 是对象成为bfc（块级元素格式化上下文）对象，
      1. 创建bfc对象方法
         1. Overflow：hidden/scroll/auto（除visible之外的属性值）；
         2. Float：left/right（除none之外的值）；
         3. Display：inline-block、table-cells、table-cpation。
         4. Position：绝对定位absolute或者固定定位fixed。
3. Vertical：top/middle/bottmo垂直对齐方式（对行内块最敏感，规避了脱标流）
4. Div里有图片，尽量定义宽高
5. 完整的一个滑动门的案例：按钮可根据内容自动调节大小
6. <html>
7. <head>
8. <style type="text/css">
9. \*{margin: 0;padding: 0;}
10. .nav{
11. background: url(微信案例/weixin\_bg1d20af.jpg);
12. height: 74px;
13. }
14. .center{
15. width: 980px;
16. margin: 0 auto;
17. }
18. li{
19. list-style: none;
20. float: left;
21. }
22. li a{
23. color: #fff;
24. display: inline-block;
25. text-decoration: none;
26. height: 33px;
27. line-height: 33px;
28. padding-left: 14px;
29. margin: 20px 20px 0 20px;
30. background: url("微信案例/bg.png") 0 -192px;
31. }
32. li a span{
33. background: url("微信案例/bg.png") right -192px;
34. overflow: hidden;
35. display: inline-block;
36. height: 33px;
37. padding-right: 10px;
38. }
39. li a:hover{
40. background: url("微信案例/bg.png") 0 -144px;
41. }
42. li a span:hover{
43. background: url("微信案例/bg.png") right -144px;
44. }
45. </style>
46. </head>
47. <body>
48. <div class="nav">
49. <div class="center">
50. <ul>
51. <li><a href="#"><span>首首首首页</span></a></li>
52. <li><a href="#"><span>首页页页</span></a></li>
53. <li><a href="#"><span>首页页</span></a></li>
54. <li><a href="#"><span>首页</span></a></li>
55. <li><a href="#"><span>页</span></a></li>
56. </ul>
57. </div>
58. </div>
59. </body>
60. </html>
61. **块元素的高不能继承**
62. 标签调用的样式越少越好

# CSS第七天 2016/08/08

1. **梅兰商贸案例**
2. 不给盒子宽高会造成文字环绕
3. 图文排版
   1. 给img标签浮动，设置P的属性overflow：hidden
   2. 此时文字和图片顶端对齐
   3. 可用外边距调整位置
   4. 不给P标签overflow：hidden：是文字环绕图片
   5. 给P标签overflow：hidden；（设置浮动也是如此，把P标签创建为一个bfc对象）属性后是
4. **一个元素层叠在另一个元素上的时候用定位**
5. **大的布局定位的时候用浮动，小的元素用定位**
6. 内在盒子中容水平居中，text-align：center；
7. 行高之可以一行文字在水平居中
8. 阿斯达ad

# HTML复习

## 2016/07/29 HTML第一天复习

* 单标签：<br/>:换行 <hr/>：水平线 <!---->:注释标签
* 双标签：<h1-h6>:h1只能出现一次
* <p>段落标签
* 图片标签<img src=”图片的来源” alt=“替换的文本” title=”提示文本” width=“宽” height=“高”>
* 只设置宽或者高，图片会等比缩放。同时设置宽和高可能会失真
* 相对路径：相对于自己的路径，从自身出发寻找其他文件
* 绝对路径：文件的完整路径
* 超链接<a href=“去往的页面” target=“\_blank \_self（默认）” title=“提示文本” />
* 链接锚点 <a href=”#ID名” />
* 链接优化写法：<base target=”\_blank”>，页面所有链接都在新页面打开窗口
* 特殊字符：&nbsp 空格 &lt < &gt > &copy ©
* 列表：ul li 无序列表 ol li 有序列表 dl dt dd 自定义列表
* 文本格式化：
* <strong> < b> 加粗
* <em> <i> 斜体
* <ins> <u> 下划线
* <del> <s> 删除线
* <sub> 下标
* <sup> 上标

## 2016/07/30 HTML第二天复习

### Meta 元数据

* Meta name=”ketwords” conter=”内容”
* Meta name=”description“ conter=”描述内容”
* http-equiv=”refresh” conter=”5;**http://(不能少http：//)**[www.baodu.com](http://www.baodu.com)”
* link rel=”stylesheet” href=”路径” type=”text/css”
* link rel=”style” href=”路径” type=”text/js”
* link rel=”icon” href=”路径” type=”x-icon”

## 2016/08/01 CSS第一天复习

### CSS样式表

* **外联样式表**

1. <link rel = “stylesheet” href=”样式表路径” type=”text/css”>

* **内嵌样式表**

1. <Style>
2. 选择器：{属性：属性值}；
3. </style>

* **行内样式表**

1. <p style=”属性:属性值”> <p>

* **Text-align：内容对齐方式**
* Center/left/right
* **Text-indent：首行缩进**
* 数值
* **选择器：选择谁的过成**
* 美化HTML标签
* 基本选择器：标签选择器
* **类选择器**：.自定义类名｛属性：属性值｝；一个标签可调用多个类名，多个类目也可作用同一个标签（类名用空格隔开
* **ID选择器命名也遵循一下规则**
* 不能用纯数字命名
* 不能用特殊符号（“\_”除外）、数字开头
* 不推荐使用汉字来定义类名
* 不推荐类名（id名）与属性名（值）相同
* ID选择器：#自定义ID名｛属性：属性值｝；
* 一个标签只能调用一个ID选择器
* 一个ID选择器在一个页面**只能使用一次**
* 因为不符合W3C操作规范，JS调用会报错
* 通配符选择器：\*｛属性：属性值｝；
* **选择页面所有的元素，不推荐使用**
* **复合选择器：两个或者两个以上选择器组成**
* **交集选择器：标签选择器.类选择器/#id选择器｛属性：属性值｝；**
* 既满足标签选择器：又满足**类/id**选择器
* **后代选择器：选择器(空格)选择器…｛属性：属性值｝；**
* 基本前提选择器必须是包含关系：父在前，子在后
* 无限制隔代
* 可以是任意选择器，自由组合
* **子代选择器：选择器(>)选择器｛属性：属性值｝；**
* 只能选择下一代
* **并集选择器：选择器1，选择器2，…选择器n｛属性：属性值｝；**
* **属性选择器：标签名[属性=属性值]｛属性：属性值｝；**
* 标签名[属性|=属性值]｛属性：属性值｝；
* 选择（属性=属性值）用“-”连接的并且包含属性值的的标签
* 标签名[属性~=属性值]｛属性：属性值｝；
* 选择器

## 2016/08/02 CSS第二天复习

1. **CSS三种书写方式**
   1. 内嵌
      1. <style>选择器｛属性：属性值；｝</style>
   2. 外链
      1. <link sel=”stylesheet” href=”css文件路径”>
   3. 行内
      1. <div style=”属性：属性值；”><div>
2. **HTML标签分类（显示方式）**
   1. 块元素
      1. 典型块元素Div p h-h6 li…
      2. 独占一行
      3. 可设置宽高
      4. 没有定义宽高时，宽高为父级元素的宽高
   2. 行内元素
      1. 不能直接设置宽高
      2. 在一行显示
      3. 宽高为内容撑开的宽高
   3. 行内块元素
      1. 典型：img input
      2. 在一行显示
      3. 可以设置宽高
   4. 元素转换
      1. Display：inine/block/inline-block
3. **CSS三大特性**
   1. 层叠性：样式发生冲突时，总是执行后面的样式，和调用顺序无关
   2. 继承性：
      1. 文字的所有属性都可以继承
      2. 特殊h1-h6不能继承文字大学奥
      3. A标签不能继承文字颜色
4. **优先级**：
   1. 默认样式<标签样式<类元素<ID选择器<行内样式<！important

0 1 10 100 1000 1000+

* 1. 继承的权重为0，权重会叠加

1. **链接伪类**
   1. A:link：默认状态链接的样式
   2. A:visited：点击过的链接样式
      1. 如果点击连接后，再次刷新还是显示点击过的链接样式
      2. 请清空浏览器缓存
   3. A:hover：鼠标悬停与链接上方的样式
   4. A:active：激活（按住链接不放鼠标）状态下链接样式
   5. :focus：获取焦点

**★如果四个伪类都要使用，要按顺序书写**

1. **背景**
   1. Background-color：颜色
   2. Background-image：url（）背景图片
   3. Background-repeat：no-repeat|repeat|repeat-X/Y平铺方式
   4. Background-position：center/top/bottom/left/right|normal背景定位
      1. 只写一个值的时候 另一个值默认是center
   5. Background-attachment：fixed|scroll 背景滚动方式

## 2016/08/04 CSS第三天复习

1. **盒子的宽度=定义的宽度+左右内边距+左右边框**
2. 两个盒子垂直方向上的外边距会发生重叠，
3. 两个盒子垂直方向上会发生塌陷
   1. 解决方法：
      1. 给父盒子设置边框（不推荐使用）
      2. 给父盒子设置overflow：hidden
4. 盒子模型
   1. Border：width solid color （style必须写）
   2. Padding：影响盒子大小，不可以为负值
   3. Margin：并列盒子垂直方向上发生边距合并，取大值
      1. 嵌套盒子发生塌陷
      2. 左右不会发生边距合并
   4. Content：内容
   5. Border-collapse：collapse；边框合并
   6. **行内元素可以定义左右内外边距，上下边距不起作用**

## 2016/08/05 CSS第四天复习

1. 文档流：从左往右，从上到下
2. 浮动：float：left/right/
3. 清除浮动：清除浮动产生的不利影响
   1. 额外标签法
   2. Overflow：hidden
      1. .clearfix：after｛
      2. Content：“.”；
      3. Display：block；
      4. Height：0；
      5. Line-height：0
      6. Visibility：hidden；
      7. Clear:both;
      8. ｝
      9. .clearfix{
      10. Zoom:1;//兼容IE浏览器
      11. }
4. **Position 定位**
   1. Static：默认值，文档流
   2. Relative：相对定位，相当于自己的位置移动
   3. Absolute：绝对定位，相对于最近的以定位的父元素位置移动
   4. Fixed：固定定位
5. 度



# JS（JavaScript）介绍

## 概念

1. js是一门在客户端进行交互的网页编程语言
2. 组成
   1. Ecmascript：JS标准
   2. DOM：可以利用JS操作网页元素
   3. BOM：通过api来操作浏览器
3. 特点
   1. 简单易用、基于对象、基于过程、
   2. 解释执行：JS引擎来执行源代码
   3. 编译执行：java/c#源代码-->可执行文件.dll-->电脑执行.dll文件
4. 使用范围
   1. 表单验证
   2. 轮播图特效
   3. 游戏开发
5. JS书写位置
   1. 内嵌式写法
   2. 推荐Js代码最好放在body结束标签的后边。

<script type=”text/javascript”>

JS信息

</script>

* 1. 外联式写法
  2. 推荐将多个js文件合并为一个js文件在浏览器中执行。

1. <script src=“路径.js”>…</script>

## 在页面中输出消息的几种写法

1. Alert(“内容”)；弹窗口
2. confirm(“信息”) 在页面出现一个弹出框，将信息显示，常与if判断语句结合使用。
   1. **如果用户点击确认；返回值为true**
   2. **如果用户点击取消；返回值为false**
3. prompt(“信息”) 弹出一个文本框，可以在文本框接收内容。常用于接收信息。
   1. **如果用户点击确认，那么返回值为输入的值。**
   2. **如果用户点击取消，那么返回值为 null**
4. console.log(“信息”)将信息输出到控制台，常用于代码调试。
5. Document. Write(“信息”) 直接在页面输出消息。

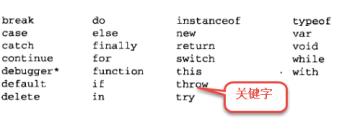
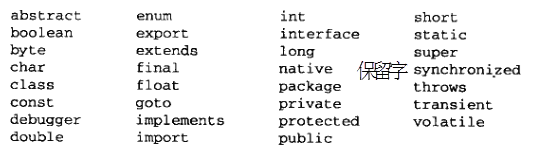
* **document.write(“信息”)，不仅可以输出内容，还可以输出html标签。输入HTML标签要加双引号**

1. //单行注释，/\* \*/多行代码注释
2. 每行代码结束后必须使用分号 **加 ；**

## 变量

1. 变量就是变化的量
2. 变量的作用就是用来装数据的容器。
3. 定义变量
   1. 定义变量用**var 变量名；**
   2. 可以实用赋值运算直接给变量赋值
      1. Var a = 10; //正确
      2. Var a = 10,b = 12;//正确
   3. 赋值运算符
      1. 一个等号“=”：赋值运算
      2. 两个等号”==”：判断是否相同
      3. 三个等号“===”：判断是否全等（包括数据类型）
4. 变量命名规范
   1. 不能使用纯数字或者数字开头来定义变量名。
   2. 不推荐使用汉字来定义变量名。
   3. 不能使用特殊符号或者特殊符号开头来定义变量名

（\_除外）

* 1. 不能以关键字来定义变量名
  2. 
  3. 不能使用保留字来定义变量名
  4. 
     + **注意：在JS中，是严格区分大小写的**

## 数据类型

1. **简单类型**
   1. **Number：数字类型**，包括小数，正负数
      1. 取值范围：
         1. 最大值±1.7976931348623157乘以10的308次方
         2. 最小值是±5 乘以10的-324次方
      2. 表示方式
      3. 十进制
      4. 十六进制：以0x开头，0-9、a-f
      5. 八进制：以0开头，0-7组成
   2. **String：字符型**
      1. **凡是用””或者’’引起来的全是字符串类型**
      2. 转义字符：
         1. \”转双引号、\’转单引号、\r回车、\b换行
   3. **Boolean：布尔类型**
      1. True:实际运算中代表“1”，正确
      2. False:实际运算中代表“0”，错误
   4. **undefined未初始化**
      1. 定义了变量，未赋值
   5. **Null**
      1. object(值为空) 引用为空， 内存里找不到这个变量
2. **复杂类型**
   1. **Object：对象**
   2. **Array：数组，一个变量储存一组值**
      1. 声明数组
         1. Var arr=Array（4）；//声明的同时给予数组长度
         2. Var arr=Array（）； //不给数组长度
         3. Var arr=Array（”no1”,”no2”,”no3”…..）；//声明的同时赋值
         4. Var arr=[”no1”,”no2”,”no3”…..];//也可以声明数组，**不推荐**
      2. 数组下标从0开始
      3. 数组可以存字符串、数字、变量、布尔值
         1. 还可以是另一个数组的元素
            1. Arr1[1] = arr2[5];//把arr2的第六个元素赋值给arr1的第二个元素
         2. 还可以是其他数组（二维数组）
            1. Arr[0][0]:第一个下标表示行，第二个下标表示列

|  |  |  |
| --- | --- | --- |
| [0][0] | [0][1] | [0][2] |
| [1][0] | [1][1] | [1][2] |
| [2][0] | [2][1] | [2][2] |

* + 1. **关联数组**//没明白
       1. Var arr[自定义下标]=“数值”
       2. For in：专门遍历关联数组

1. **比较运算符**

**>= <= != < >**

1. **判断数据类型**
   1. **Typeof（变量名）；**
2. **算数运算符**
   1. **（+）：**
      1. 两个数据类型都为数字类型变量相加，得到的是数字类型
      2. 一个为数字类型的变量和一个为字符串类型的变量相加，得到的是一个字符串类型，加号起一个连接的作用
      3. **数+数=数 数+字=字**
   2. **（-）：**
      1. 如果两个变量都为数字类型相减，得到的是数字类型
      2. 如果一个为数字类型的变量，一个为数字字符串，相减得到的是数字类型。
      3. 如果一个为数字类型的变量，一个为非数字字符串，相减得到的NaN,数字类型
      4. **数-数=数 数-数字字符串=数 数-字=NaN 字-字=**NaN
      * NaN not a number
      1. 两个都为字符串的变量相减，得到的是NaN.
   3. （/）乘除运算
      1. 两个都为数字类型的变量，相除得到的是数字类型
      2. 如果一个为数字类型的变量，一个为数字字符串变量，相除得到的是数字类型
      3. 如果一个为数字类型的变量，一个为非数字字符串，相除得到的NaN,数字类型
      4. 如果0作为除数，得到的是infinity(无限大)，是一个数字类型。
      5. **数/数=数 数/数字字符串=数 数/字=NaN 数/0=infinity（无限大）**
   4. %模运算 取余
   * **() 优先级 先计算()括号里边的值**
   1. 赋值运算（=）

## 语法

1. 语法：
   1. 广义的包含语句、单词、标点符号
   2. 狭义的含义是特指语句结构的各方面规则
   3. Js的语法和java和C++语言有很多相似之处
2. **语句**
   1. JS或者任何一种编程语言编写出来的脚本都是由一系列指令构成，这些指令被称为语句，只有按照正确的语法编写出来的语句才能得到正确的解释。
   2. 每句结束后必须加分号结束
   3. 注释标签 // /\*\*/
3. **If判断语句**：if（）｛｝else if（）｛｝else｛｝
4. **For循环语句：**
   1. for（var i=0；i<n；i++）｛循环体｝
   2. while（循环条件）｛循环体；｝
   3. do｛循环体｝while（循环条件）：是否符合循环条件都会循环一次
   4. break:跳出循环
   5. continue:跳出本次循环
5. **&& 与 || 或 ！非**

# JS第一天2016/08/10

### JS介绍

1. 94年网景公司，研发第一款浏览器
2. 95年sun公司，java语言诞生
3. 网景公司和sun公司合作 Java + Script=JavaScript
4. 布兰登艾奇10天设计了JavaScript
   1. 借鉴C的基本语法
   2. Java的数据类型和内存管理
   3. 借鉴Schrome语言
5. **W3c规范**
   1. 结构标准：html
   2. 表现标准：css
   3. 行为标准：js
6. EcmaScript：javascript标准
7. ES6：2015年
8. Ajax：异步交互技术
   1. 表单验证
9. Node.js：是js可以开发服务器程序

### JS概念

1. Js是一款运行在客户端的网页编程语言
2. 组成
   1. Ecmascript JS标准
   2. Dom：通过js操作网页上的元素
   3. Bom：通过api（程序接口）操作浏览器
3. 特点
   1. 简单易用
   2. 解释执行
4. 基于对象：更简单，更直接
   1. 没有继承和多态
5. 面向对象
   1. 面向过程
6. JS使用范围
   1. 表单验证
   2. 轮播图特效
   3. 游戏开发
7. 书写位置
   1. 内嵌
   2. 外链：

<script src="js.js">..</script>

* + **推荐写在HTML结束标签之后**
  + **CSS是<link rel=”stylesheet” type=”text/css” href=””>**
  + **每行后都要加分号，不能省**
  1. 页面在下载必须等脚本执行完毕才能继续
  2. 把js代码写在一个文件内

1. 在页面中输出信息的几种方法
   1. **Alert**:弹出对话框
   2. **Confirm**:弹出对话框
   3. **Console.log**：输出在控制台
   4. **Prompt**:弹出对话框 用户输入信息
   5. Document.write();在页面输出消息
2. 定义变量
   1. Var bl;
3. 给变量赋值
   1. Var bl = “值”；
4. Alert(“”):引号之间是直接量，输出变量的时候不要加引号
5. **变量的命名规范**
   1. **不能用使用数字和特殊符号开头（\_）除外**
   2. **不推荐使用关键字和保留字**
   3. **严格区分大小写**

* **数据类型转换**
  1. **转换成字符串：toString（）；**
  2. **转换成数字：**
     1. **parseInt（）；**
     2. **parseFloat（）；**

### 数据类型

* 1. **Typeof(变量名)：输出变量的类型**
  2. **简单类型**
     1. **Number:数字类型**
        1. 十进制：0-9
        2. 八进制：0-7 0开头
        3. 十六进制：0-9 a-f 0x开头
     2. **String:字符串类型**
        1. 用双/单引号，括起来的
     3. **Boolean布尔类型** 
        1. true(正确) false(错误)
     4. **undefined:变量未初始化**
        1. 定义了变量未引用（未赋值）
     5. **Null:在内存中找不到这个变量**
        1. 空字符串在内存中存在
  3. **复杂类型**
     1. Array:数组
     2. Object:对象

1. **比较运算符**
   1. > < <= >= != ==
2. **算术运算符**
   1. +：加号
      1. 两个number相加得到numbrt
      2. 一个数字一个字符串得到字符串
      3. 连接两个字符串
         1. 字符串+字符串=两者直接拼接
         2. 字符串+数值=数值转换为字符串再拼接
      4. **将一个值转换成字符串：数值+空字符串（“”）**
   2. -：减号
      1. 两个数字相减得到数字类型
      2. 一个数字和一个数字字符串，得到数字类型
      3. 一个数字类型和一个非数字字符串，得到NaN,是一个数字类型，不是数字
      4. **将字符串转换成数值：数值型字符串 – 0**
      5. **对操作的数进行取反操作**
      * **NaN: not a number**
   3. **/ 除法**
      1. 两个都为数字类型的变量，相除得到的是数字类型
      2. 如果一个为数字类型的变量，一个为数字字符串变量，相除得到的是数字类型
      3. 如果一个为数字类型的变量，一个为非数字字符串，相除得到的NaN,数字类型
      4. 如果0作为除数，得到的是infinity(无限大)，是一个数字类型。
   4. **%取余**
3. **带操作的赋值运算**
   1. **+= -= \*= /+**

# JS第二天2016/08/11

### Date对象

* 1. 处理日期和时间
  2. New：创建一个新的对象
  3. Date（）；返回当日的日期和时间
  4. getDay（）：返回当前周几（0-7）
  5. getMonth（）+1：返回当前月
  6. getFullYear（）：当前年份
  7. getHours（）：返回当前小时
  8. getMinutes（）：返回当前分钟
  9. setSeconds（）：返回当前秒

### math对象

* 1. 处理数学问题
  2. 
  3. Math.Ceil（）:相上取整，天花板函数
  4. Math.floor（）：相下取整，地板函数
  5. Math.pow（n1,n2）：返回n1的n2次方
  6. Math.rount（）：四舍五入
  7. Math.random（）：返回（0-1）随机数，不包括0、1
  8. **生成min到max之间的随机数**
     1. **var s = Math.floor(Math.random()\*(max-min+1)+min**

### 字符类型转换

#### 强制转换

1. **字符串转换数字**
   1. **Numbnr();**
      1. 数字类型的字符串，转换之后得到数字
      2. **非数字字符串，转换之后是NaN**
      3. 小数类型的字符串，转换后是原数字
   2. **parseInt（）**
      1. 数字类型的字符串，转换之后得到数字
      2. 以数字开头的字符串，转换后得到开头的数字
      3. 非数字开头的字符串，转换后得到NaN
      4. 小数转换后，是前面的整数数字（正转正，负转负）
   3. **parseFloat（）**
      1. 数字类型的字符串，转换之后得到数字
      2. 以数字开头的字符串，转换后得到开头的数字
      3. 非数字开头的字符串，转换后得到NaN
      4. 小数转换后，转换为原数字
2. **转布尔类型 boolean（）**
   1. Boolean（）
      1. 数字（0除外）和字符串转换后是true
         1. 数字转换后得到true

**0转换后得到false**

**Infinity转换后得到true**

* + - 1. 字符串类型转化和得到true
    1. Undefined、null、0转换后得到false

#### 隐式数据类型转换

1. 运算过程中，程序自动进行的类型的数据转换（程序员没有操作）

### 逻辑运算符

逻辑运算只有两个结果，一个为true，一个为false.

1. **且 &&：**
   1. 两个都为true真才为真true
   2. 有一个为false假就是假true
2. **或 ||：**
   1. 有一个为真true就为真false
   2. 两个都为false假才为假false
3. **非 ！:true和false对换**
   1. 输出相反的结果
   2. 连续两个！！，可以将任意数值转换为boolean类型

### 比较运算符

#### 大小关系比较

1. > 、<、>=、<=，大小关系比较
   1. 仅一个是数值，将另一个数值转换数值后比较
   2. 字符串比较，逐字符串比较unicode值
   3. 字符串与非数值比较，将非数值的转换为字符串比较
   4. 非数值也非字符串，转换为字符串比较
   5. 比较数无法转换为字符串，返回fslse
   6. 与NaN比较，返回false
2. == 、!= 相等比较
   1. 字符串与数值比较，字符串转换为数值
   2. null与undefined相等，比较不进行转换
   3. NaN 与谁都不相等，包括自身
   4. 对象比较
      1. ==：属于同一对象
      2. !=：不属于同一对象
3. ===、!== 比较
   1. 值比较需要类型相同切数值相等
   2. 值类型与引用类型肯定不想等
   3. 引用类型的比较。比较他们的引用地址

#### 等值关系比较

1. ==：等于（内容相等）
2. ===：全等于（内容和数据类型都相等）
3. !=：不等于
4. !==：不全等于
5. **变量集体赋值，逗号隔开**

### IF条件判断

#### 第一种

If（条件表达式）｛

执行语句；

｝

Else{

执行语句；

}

#### 第二种，嵌套

If（条件表达式）｛

执行语句；

｝

Else if（条件表达式）｛

执行语句；

If（表达式）｛

执行语句

｝

Else｛

执行语句

｝

｝

…

Else{

执行语句；

}

### 三元表达式

**表达式？执行语句一：执行语句二；**

表达式结果为true执行语句一

如果结果为false执行语句二

### 代码调试Sources

1. **运行代码**
2. **F12网页控制台，代码调试工具**
3. **打开Sources（原因）窗口**
4. **设置断点**
5. **点击选中要监视的属性（语句）右键add to watch**
6. **刷新分步运行，看监视窗口watch**

### 课堂反馈

Math.pow（次方）

Math.random（随机数）

Math.floor（向下取整）

Math.ceil（向上取整）

Math.round（四舍五入）

Number：数=数；字符串=NaN；小数字符串=原小数

parseInt：数字字符串=数字；数字开头字符串=开头数字；

小数字符串=小数整数部分；字符串=NaN

ParseFloat：数字字符串=数字；数字开头字符串=开头数字；

数字符串=小数整数部分；小数字符串=原小数

# JS第三天2016/08/13

### Switch

Switch（变量）｛

Case 变量1：

执行语句1；

break；//阻止代码向下执行

Case 变量2：

执行语句2；

break；

default：//变量1和变量2都不满足的时候执行default

执行语句3：

｝

* **break：阻断代码执行，不能省略**
* **switch变量的数据类型必须和case值的数据类型相同**

1. **case值的集体声明**

switch（变量）｛

case “变量1”： case “变量2”： case “变量3”：

执行语句；

break；

default：

执行语句；

｝

### 自增自减

1. **I++ i—**
2. **在不参与运算的情况下**
   1. **i**++，++I 相等，都是在变量的基础上+1
3. **在参与运算的情况下**
   1. n2 = n1 ++;//先将值赋给n2,然后n1再自增+1
   2. n2 = ++n1;//n1先自增+1，然后再赋值给n2

### 循环

#### While循环

While（表达式）｛

只要条件表达式为true，循环一直执行下去；

当条件表达式结果为false，循环结束

｝

* **循环结束之前，后面的代码不会执行**
* **While循环的变量需要在循环体外定义**

##### do …while

do｛

首先执行该代码依稀，然后只要条件表达式为true，循环 一直执行下去；

当条件表达式结果为false，循环结束

｝While（表达式）

* **表达式是否成立，都会循环一次**

#### for循环（重要）

for (定义变量; 条件表达式 ; 自增自减 ){

循环体；

}

1. **执行顺序**
   1. 定义变量
   2. 如果条件表达式结果为true。执行循环体
      1. 执行完之后执行，自增自减，之后在进行表达式判断
   3. 如果为false，结束执行循环
2. + 起连接作用，document.write（“”），引号内的内容原样输出，如果引号内是html标签，则按照标签内容输出
3. **Break：结束当前循环**
   1. 当前循环指break所在的循环
4. **Continue：跳过本次循环，继续下一次循环**

### 数组

1. 可以存放很多数据
2. **创建方法：**

Var ary = new Array(); //用new方法来创建一个数组

Var ary = []; //直接创建一个数组

1. **赋值**

ary[0] = 12; //通过下标给数组赋值

var ary = [12,23,45,”中国人”,”Mr.L”]；//数组的初始化

1. **数组的遍历**
   1. 用for循环对数组进行遍历
   2. 数组下标从0开始
   3. **.length:计算数组长度，数组里面有多少个数据**

for(var I = 0 ; i < ary.length ; i++){

alert(ary[i]);

}

1. **数组的合并concat**
   1. 新数组 = 第一个数组 . concat（第二个数组）；
2. **.join 追加一个字符，把数字类型转换为字符串类型**
   1. 在每个字符之间添加一个字符
   2. 返回一个字符串类型数组

### 函数function

1. 函数是完成特定功能的一段代码，
2. 定义函数

function test（）｛//用function创建函数

函数主体；

}

### 课堂反馈

1. 自增自减 i++ ++i
2. Switch（变量）｛case 变量1：break； case 变量2：break；… default：｝
3. While（判断语句）｛循环体；｝
4. Do｛循环体｝while（判断语句）：先执行一次再判断
5. For（变量声明；判断语句；自增自减）｛循环体｝
6. Break结束当前循环
7. Continue：结束本次循环，执行下一次循环
8. 数组的定义：var arr = [ ];
9. 函数的定义：function test（变量）｛函数主体；｝

# JS复习

## JS第一天复习2016/08/11

1. 客户端的编程语言
   1. Ecma：标准
   2. Dom：通过JS操作网页元素
   3. Bom：通过API操作浏览器
2. 推荐把JS放在body后面
3. Alert（）、
4. 变量：存储数据的容器
5. 赋值，比较。运算
6. 严格区分大小写
7. 简单数据类型
   1. Number
   2. String
   3. Boolean：true false
   4. Undefined：定义没赋值
   5. Null：销毁一个对象
8. 复杂数据类型
   1. Array数组
   2. Object：对象
9. 比较运算符：> < >= <= == !=
10. 算术运算符 + - \* / %
    1. 数+数=数
    2. 数+字=字
    3. 数-数=数
    4. 数-数字符串=数
    5. 字-字=NaN
    6. 0做除数结果infinity（无限大）
    7. 括号优先级最高

## JS第二天复习

1. Math
   1. Math.ciel
   2. Math.floor
   3. Math.pow
   4. Math.random
   5. Math.count
2. 特殊类型：null，undefined
3. 强制数据类型转换：
   1. Number：不识别字符串开头的字符串，小数转小数
   2. parseInt/parseFloat:数字开头的字母转换为开头的数字，parseInt小数转换为小数整数部分，parseFloat小数转换为原小数
4. 逻辑运算符

&& || ！

1. 等号运算符

= == === != !==

1. 逗号运算符
2. If..else..分支结构
3. 三元运算符
   1. 判断？语句一：语句二；
   2. 判断为真执行语句一，否则执行二