[什么是web标准 2](#_Toc266395701)

[内容/结构/表现 2](#_Toc266395702)

[**id的命名规范** 2](#_Toc266395703)

[**推荐命名习惯** 2](#_Toc266395704)

[**使用CSS的几种方式** 3](#_Toc266395705)

[优先级: 4](#_Toc266395706)

[**CSS样式表推荐书写顺序** 4](#_Toc266395707)

[**块元素 和 内联元素** 4](#_Toc266395708)

[一些CSS的实用技巧 5](#_Toc266395709)

[cursor: 光标形状 5](#_Toc266395710)

[Filter(CSS滤镜, 仅IE) 6](#_Toc266395711)

[注释: /\* ... \*/ 6](#_Toc266395712)

[CSS选择器类型 7](#_Toc266395713)

[DOCTYPE 8](#_Toc266395714)

[名字空间 9](#_Toc266395715)

[编码问题 9](#_Toc266395716)

[元信息标记----meta标签:在服务器和客户之间传达隐含信息。 9](#_Toc266395717)

[**<title>** 11](#_Toc266395718)

[**<base>** 11](#_Toc266395719)

[布局 12](#_Toc266395720)

[盒子模型 12](#_Toc266395721)

[**定位** 15](#_Toc266395722)

[长度单位 15](#_Toc266395723)

[绝对长度单位: 15](#_Toc266395724)

[相对长度单位: 15](#_Toc266395725)

[百分比值 16](#_Toc266395726)

[常用代码 16](#_Toc266395727)

[**DW中CSS设计** 17](#_Toc266395728)

[**类型(文字)** 17](#_Toc266395729)

[**背景** 20](#_Toc266395730)

[**区块** 22](#_Toc266395731)

[**方框** 23](#_Toc266395732)

[**边框** 24](#_Toc266395733)

[**列表** 25](#_Toc266395734)

[**定位** 26](#_Toc266395735)

[**扩展** 28](#_Toc266395736)

[笔记 29](#_Toc266395737)

[格式化模型 29](#_Toc266395738)

[尺寸 30](#_Toc266395739)

[溢出 和 剪切 30](#_Toc266395740)

[实例 32](#_Toc266395741)

[所有内容居中 32](#_Toc266395742)

[块居中 32](#_Toc266395743)

[项目列表 32](#_Toc266395744)

[单行文本输入框 33](#_Toc266395745)

[表单提交按钮 33](#_Toc266395746)

[淘宝色调 34](#_Toc266395747)

## 什么是web标准

|  |
| --- |
| 1. 结构标准语言: XML和XHTML  2. 表现标准语言CSS  3. 行为标准:DOM和ECMScript |

## 内容/结构/表现

|  |
| --- |
| 1. 内容:  2. 表现  3. 行为 |

**id的命名规范**

|  |
| --- |
| 1. 不能有空格  2. 除了下划线之外不应该包含标点符号.  3. 分配的名称必须放在引号内.  4. 不能以数字开头.  5. 在相同文档中不可以分配多个名字. |

**推荐命名习惯**

|  |  |
| --- | --- |
| **结构** | **名称** |
| 顶导航 | topnav |
| 主导航 | mainnav |
| 子导航 | subnav |
| 子菜单 | submenu |
|  |  |
| 头部 | header |
| 内容 | content |
| 底部 | footer |
| 页脚 | footet |
| 容器 | container |
|  |  |
| 商标 | label |
| 标题 | title |
|  |  |
| 侧栏 | sidebar |
| 左侧栏 | leftsidebar |
| 右侧栏 | rightsidebar |
|  |  |
| 标志 | logo |
| 标语 | banner |
| 注释 | note |
|  |  |
| 搜索 | search |
| 登陆 | login |

**使用CSS的几种方式**

|  |  |
| --- | --- |
| **外部样式**表 (引用外部单独的CSS文件)  **链接式** | 1.编写\*.css文件:  p{属性1:值1;  属性2:值2;}  2. 在head标签对中添加:  **<link rel="stylesheet" href="images/style2.css" type="text/css"/>** |
| **输入样式**表 (CSS中包含另外一个css文件)  **导入样式** | <style type="text/css" media="screen, projection">  <!--  **@import url(http://www.it315.org/style.css);** /\* 必须放在最前面 \*/  **@import url(http://www.swu.edu.cn/style.css);** /\*后面的覆盖前面的定义 \*/  p{属性1:值1;  属性2:值2;}  //-->  </style>  说明: **@import**的调用方法也可以写在CSS文件中, 用来调用其他的CSS. |
| **嵌入样式**表 (集中放在head标签对中)  **内嵌式** | 在head标签对中添加:  **<style type="text/css" media="screen, projection">**  **<!--**  **p{属性1:值1;**  **属性2:值2;}**  **//-->**  **</style>** |
| **内联样式**表 (跟每个html标签联系起来)  **行内样式** | 1.在head标签对中添加:  **<meta http-equiv="Content-Style-Type" content="text/css"/>**  2.在html标签中添加:  <p **style="color:#90F"** >内联样式表</p> |
| **链接式** 和 **导入样式** 的区别: **@import**的调用方法只能使用在样式文件中, 即只能在调用的样式文件(或者style元素)中才能正常使用. | |

## 优先级:

id选择器>class选择器>html标签选择器

|  |  |
| --- | --- |
| 3种调用样式表的方法, 优先级的排序: | 行内样式>嵌入样式>链接样式 |
| 当页面多重定义, 或者嵌套定义的CSS时, 的优先级: | 1. 当类型选择符和类选择符同时定义时, 类选择符的优先级高于类型选择符.  2. ID选择符的优先级比类选择符更高.  3. 最近优先原则 |

**CSS样式表推荐书写顺序**

|  |
| --- |
| 1. 显示属性(display, list-style, position, float, claer)  2. 自身盒子模型属性(width, height, margi, padding, border, background)  3. 文本属性(color, font, text-decoration, text-align, vertical-align, white-space, other text, content) |

**块元素 和 内联元素**

|  |  |
| --- | --- |
| 块元素 | 指div, body等, 特点:每一个块元素都是从一个新的行开始, 一般可以包含其他块元素和内联元素. 在css中可以给块元素加上浮动, 控制块元素的显示位置, 而不总是从一个新的行开始. |
| 内联元素 | 指a, span等, 特点:不必在新的一行开始, 同时, 也不强迫其他的元素在新的一行显示. 内联元素可以做其它元素的子元素. 在css中, 给内联元素加上display:block属性, 就具有块元素的特性. |

## 一些CSS的实用技巧

|  |  |
| --- | --- |
| 1. 合理使用选择符分组. | 可以统一定义几个选择符的属性, 节约大量代码. 把相同的部分定义在一起, 然后把特殊的部分单独定义. |
| 2. 合理地使用子选择符 | 可以节约代码, 同时也减少了自定义选择符的数量. |
| 3. 同一个元素的多重定义 | 这样可以减少选择符的数量: <div class="one two"></div>, 其中one和two两个类之间, 用空格分开. 最终的表现效果是两个类中属性的叠加 |

# cursor: 光标形状

|  |  |
| --- | --- |
| hand | 手型 |
| crosshair | “十”型 |
| text | “|” 型 |
| wait | 等待 |
| default | 默认 |
| help | 帮助 |
| e-resize | 东箭头 |
| ne-resize | 东北箭头 |
| n-resize | 北箭头 |
| nw-resize | 西北箭头 |
| w-resize | 西箭头 |
| sw-resize | 西南箭头 |
| s-resize | 南箭头 |
| se-resize | 东南箭头 |
| auto | 自动 |

# Filter(CSS滤镜, 仅IE)

|  |  |
| --- | --- |
| Alpha  透明度 | opacity=100代表完全不透明 |
| blue  模糊 |  |
| chroma  透明色(直接将指定颜色去掉) |  |
| flip  翻转 | {filter:flipv fliph} |
| mask  遮罩(哪些可见) |  |
| wave  波浪 |  |

# 注释: /\* ... \*/

css 超链接样式设置

|  |
| --- |
| a.A\_See:link{text-decoration:underline; color:Green;}  a.A\_See:visited{text-decoration:underline; color:Green;}  a.A\_See:hover{text-decoration:none; color:Green;}  a.A\_See:active{text-decoration:underline; color:Green;}  <a href="See.aspx?id=1>" target="\_blank" class="A\_See">查看详情</a>  1：解释：  link:设置a对象在未被访问前的样式表属性。  visited:设置a对象在其链接地址已被访问过时的样式表属性。  hover:设置对象在其鼠标悬停时的样式表属性。  active:设置对象在被用户激活（在鼠标点击与释放之间发生的事件）时的样式表属性。  2：hover和a:visited书写顺序的重要性，  css中关于超链接的四个属性正确顺序为：  a:link {}  a:visited {}  a:hover {}  a:active {} |

# CSS选择器类型

|  |  |
| --- | --- |
| 选择符类型 | 代码示例 |
|  |  |
| ID选择符 | #id1{...} |
| 类选择符 | .class1{...} |
|  |  |
| 选择符 分组 | h1, h2, h3{...} |
| 通配 选择符 | \*{ ... } |
| 类型 选择符 | h4{ ... } |
| 派生(包含) 选择符 | p strong{...} |
|  |  |
| 伪元素 |  |
| 伪类:  按照特征归类的元素, 而非用元素名称, 属性, 内容 | |  |  | | --- | --- | | 首行伪类 | p:first-line{} | | 首字母伪类 | p:first-letter{} | | 链接通过键盘属性或其他表单获取焦点时 | a:focus{} | | 未访问  已访问  鼠标悬停  链接正在被点击 | a:link{}  a:visited{}  a:hover{}  a:active{} |   选择符:伪类{属性:属性值}  伪类和伪元素一般是以":"开头, 伪类和伪元素在CSS中是指定的, 不能随意的命名和定义.  CSS超链接的状态如下：  a:link——默认常规链接的样式，非访问超链接  a:visited——访问超链接  a:hover——访问者操作鼠标滑过链接  a:active——点击链接  a:link类型出现的时间为最早，因为它可应用于所有的链接。  a:visited类型排第二，它将取代任何链接的a:link格式。若a:link类型紧跟a:visited的话，a:link可能会取代a:visited。  其次是a:hover类型，此类型只应用于访问鼠标下的链接。  最后是a:active |
|  |  |
| 子选择符 (IE支持不好) | 由 ">" 分隔  ul > li > a{...} |
| 紧邻选择符 (IE支持不好) | 由 "+" 分隔 |
| 属性选择符 (IE支持不好) |  |

## DOCTYPE

|  |  |  |
| --- | --- | --- |
| 1. 过渡的(**Transitional**) | <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 **Transitional**//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-**transitional**.dtd"> | 允许在页面中使用HTML4.01的标识(符合xhtml语法标准) |
| 2. 严格的(**Strict**) | <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 **Strict**//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1- **Strict**.dtd"> | 允许使用任何表现层的标识和属性, 例如<br />等. |
| 3. 框架的(**Frameset**) | <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 **Frameset**//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1- **Frameset**.dtd"> | 当页面中含有框架元素时, 就要采用这种DTD. |
| 备注: DOCTYPE一定要放在XHTML文档的顶部   |  |  | | --- | --- | | <!DOCTYPE | 声明的开始 | | html | 类型是html, 注意大小写, 因为使用的XHTML对大小写 | | PUBLIC | 说明文档是个公共文件 | | "-//W3C//DTD XHTML 1.0 Transitional//EN" | 说明文档是由W3C指定. DTD文件用来验证文档是否符合DTD语法规定; XHTML 1.0 Transitional中的Transitional是"过渡性"的含义 | | "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd" | DTD文件的网址 | | | |

## 名字空间

|  |
| --- |
| <html xmlns="http://www.w3.org/1999/xhtml">  名字空间是namespace, 其含义是通过一个网址指向来识别页面上的标签. 在XHTML中使用的是xmlns(XHTML namespace的缩写), 用来识别XHTML页面上的标签的网址指向是http://www.w3.org/1999/xhtml" |

## 编码问题

|  |
| --- |
| <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  为了兼容较旧版本的浏览器, 还要在原声明下追加一段编码语言的声明. 如下:  <meta http-equiv="Content-Language" content="utf-8" /> |

## 元信息标记----meta标签:在服务器和客户之间传达隐含信息。

|  |  |
| --- | --- |
| name | 指定文档中附加信息的名称. |
| http-equiv | 和name属性类似, 用来指定附加信息的名称. 在浏览器加载页面之前, 服务器会把http-equiv属性定义的相关信息发送给浏览器, 便于在浏览器中正确显示页面. |
| content | 指定文档中附加信息的值 |

|  |
| --- |
| <meta name="keyword" content="设置搜索引擎搜索的关键字，关键字之间用逗号分割">总长度最好不要超过1000个Character (约44个字)。 |
| <meta name="description" content="描述网页主题，简要描述，帮助搜索引擎搜索">总长度最好不要超过200个Character (约15个字)，若文章真的太长，可以切割成两个部分，较不重要的部分置入下一个Description。 |
| <meta name="generator" content="Macromedia Drenmweaver MX 2004"> |
| <meta name="author" content="作者的姓名"> |
| <meta http-equiv="refresh" content="跳转时间（秒）; url=链接地址"> //设置自动跳转，当链接地址省略时，就变成了自动刷新功能。 |
| <meta http-equiv="content-type" content="text/html; charset=gb\_2312(简体中文)（英文是ISO\_8859-1"> 或 |
| <meta http-equiv="content-language" content="zh\_CN（简体中文）"> //说明网站的格式及编码方式，另外这个功能也可以拿来说明网站的名称。 |
| <meta http-equiv="expires" content="GMT时间格式（wed,27 february 2005 11:00:00 GMT"> //网页到期时间 |
| <meta http-equiv="cache-control" content="no-cache"> 或 |
| <meta http-equiv="pragma" content="no-cache"> //禁止从缓存中调用 |
| <meta http-equiv="set-cookie" content="到期的时间GMT时间格式"> //删除过期cookie(如果网页过期，则删除存盘的cookie) |
| <meta http-equiv="windows-target" content="\_top"> // windows-target表示新网页的打开方式,\_top表示打开一个独立的页面. 强制打开新窗口 |
| <meta http-equiv="page-enter" content=“revealtrans(duration=过度效果持续时间默认秒为单位,transition=过渡方式)"> // |
| <meta http-equiv="page-exist" content=“revealtrans(duration=过度效果持续时间默认秒为单位,transition=过渡方式)"> // |
| <meta http-equiv="refresh content="2"> //刷新当前文档, 间隔时间为2秒  <meta http-equiv="refresh content="2; url=跳转地址"> //页面在一定时间后, 跳转到其他页面 |
| <meta name="robots" content="限制搜索方式">   |  |  | | --- | --- | | robots的值 | 含义 | | All | 页面将被检索，且页面上的链接可以被查询。 | | None | 页面不能被检索，且页面上的链接不可以被查询。 | | Index | 页面将被检索，但页面上的链接不可以被查询。 | | Follow | 页面上的链接可以被查询。 | | Noindex | 页面不能被检索，但页面上的链接可以被查询。 | | Nofollow | 页面能被检索，但页面上的链接却不可以被查询。 | |
| <meta http-equiv="page-enter" >进入页面  <meta http-equiv="page-exist" >离开页面   |  |  | | --- | --- | | 编号 | 过渡方式 | | 0 | 盒状收缩 | | 1 | 盒状放射 | | 2 | 圆形收缩 | | 3 | 圆形放射 | | 4 | 由下往上 | | 5 | 由上往下 | | 6 | 从左至右 | | 7 | 从右至左 | | 8 | 垂直百叶窗 | | 9 | 水平百叶窗 | | 10 | 水平格状百叶窗 | | 11 | 垂直格状百叶窗 | | 12 | 随意溶解 | | 13 | 从左右两端向中间展开 | | 14 | 从中间向左右两端展开 | | 15 | 从上下两端向中间展开 | | 16 | 从中间向上下两端展开 | | 17 | 从右上角向左下角展开 | | 18 | 从右下角向左上角展开 | | 19 | 从左上角向右下角展开 | | 20 | 从左下角向右上角展开 | | 21 | 水平线状展开 | | 22 | 垂直线状展开 | | 23 | 随即产生一种过渡方式 | |

**<title>**

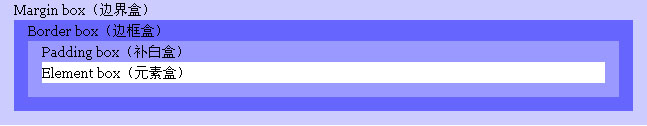
<title>标题</title>总长度不要超过85个Character (约10个字)。

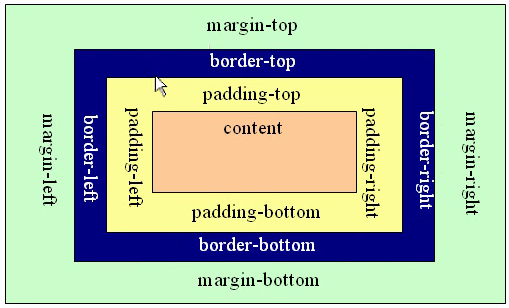
**<base>**

|  |  |
| --- | --- |
| <base href="链接基底网址" target="新窗口的打开方式"> | |
| 属性值 | 新窗口的打开方式 |
| target="框架名称" | 指定框窗 |
| \_parent | 在上一级窗口中打开 |
| \_self（默认方式） | 在同一窗口中打开 |
| \_top | 在浏览器的本窗口的整个窗口中打开，忽略任何框架。 |
| \_blank | 跟\_new差不多，只是会一直出现新的窗口，而不会在同一个新窗口开启 |
| \_new | 在一个新窗口开启，不过只会一直在同一个新窗口开启，不会另外在开一个新窗口 |

# 布局

# 盒子模型





margin: 间隙宽度、外边距

padding: 填充宽度、内边距

border: 边框宽度

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| body | body基本设置   |  | | --- | | body {  font: 100% 宋体, 新宋体;  background: #666666;  margin: 0; /\* 最好将 body 元素的边距和填充设置为 0 以覆盖不同的浏览器默认值 \*/  padding: 0;  text-align: center;/\* 在 IE 5\* 浏览器中，这会将容器居中。文本随后将在 #container 选择器中设置为默认左对齐 \*/  color: #000000;  } |   第1种情况: body下的主体<div id="container"> 主体居中   |  | | --- | | #container {  width: 780px; /\* 使用比最大宽度 (800px) 小 20px 的宽度可显示浏览器界面元素，并避免出现水平滚动条 \*/  background: #FFFFFF;  margin: 0 auto; /\* 自动边距（与宽度一起）会将页面居中 \*/  border: 1px solid #000000;  text-align: left; /\* 这将覆盖 body 元素上的“text-align: center”。 \*/  } |   第2种情况: body下的主体<div id="container"> 主体平铺   |  | | --- | | #container {  width: 100%;/\*浏览器有好宽, 主体就有好宽\*/  background: #FFFFFF;/\* 边距：0 自动；自动边距（与宽度一起）会根据需要将页面居中 \*/  text-align: left; /\* 这将覆盖 body 元素上的“text-align: center”。 \*/  } | |
| header | header   |  | | --- | | #header {  background: #DDDDDD;  padding: 0 10px 0 20px; /\* 此填充会将出现在它后面的 div 中的元素左对齐。如果 #header 中使用的是图像（而不是文本），您最好删除填充。 \*/  } |   header的最后一个元素(避免边距重叠 即div之间出现的空白)   |  | | --- | | #header h1 {  margin: 0; /\* 将 #header div 中最后一个元素的外边距设置为零将避免边距重叠（即 div 之间出现的无法解释的空白）。如果 div 周围有边框，则不必将边距设置为零，因为边框也会避免边距重叠 \*/  padding: 10px 0; /\* 使用填充而不使用边距将可以使元素远离 div 的边缘 \*/  } | |
| footer | footer   |  | | --- | | #footer {  padding: 0 10px; /\* 此填充会将它div里面的所有元素左对齐。 \*/  background: #DDDDDD;  } |   footer的第一个元素(避免边距重叠 即div之间出现的空白)   |  | | --- | | #footer p {  margin: 0; /\* 将脚注中第一个元素的外边距设置为零将避免出现可能的边距重叠（即 div 之间出现的空白）\*/  padding: 10px 0; /\* 就像边距会产生空白一样，此元素上的填充也将产生空白，但不会出现边距重叠问题 \*/  } | |
| 浮动 | 左浮动的侧边栏   |  | | --- | | #sidebar1 {  float: left; /\* 由于此元素是浮动的，因此必须指定宽度 \*/  width: 200px; /\* 宽度或者=12em, 在符合标准的浏览器中或者在 Internet Explorer 中的标准模式下，此 div 的实际宽度除了包括内容宽度外，还包括填充和边框 \*/  background: #EBEBEB; /\* 将显示背景色，其宽度等于栏中内容的长度，\*/  padding: 15px 10px 15px 20px;  }  #sidebar1 h3,  #sidebar1 p {  margin-left: 10px; /\* 对于将要放在侧栏中的每个元素，都应当设置左边距和右边距 \*/  margin-right: 10px;  } |   右浮动的侧边栏   |  | | --- | | #sidebar1 {  float: right; /\* 由于此元素是浮动的，因此必须指定宽度 \*/  width: 200px; /\* 在符合标准的浏览器中或者在 Internet Explorer 中的标准模式下，此 div 的实际宽度除了包括宽度外，还包括填充和边框 \*/  background: #EBEBEB; /\* 将显示背景色，其宽度等于栏中内容的长度，\*/  padding: 15px 10px;  } |   不浮动的主体栏   |  | | --- | | #mainContent {  margin: 0 250px 0 0; /\* 此 div 元素的右边距会在页面的右下方创建栏 无论 sidebar1 div 中包含多少内容，都将保留栏空白。如果您希望在 #sidebar1 中的内容结束时，用 #mainContent div 的文本填充 #sidebar1 空白，则可以删除此边距。 \*/  padding: 0 20px; /\* 请记住，填充是 div 方块内部的空间，边距则是 div 方块外部的空间 \*/  } |   清除浮动   |  | | --- | | .clearfloat { /\* 此类应当放在 div 或 br(break) 元素上，而且该元素应当是 完全包含浮动的容器 关闭之前的最后一个元素 \*/  clear: both;  height: 0;  font-size: 1px;  line-height: 0px;  } | |

**定位**

|  |
| --- |
| 1 float(浮动)  2 position 定位  2.1 position:absolute(绝对定位)  2.2 position:relative(相对定位)  3 Z-index(空间位置) |

# 长度单位

## 绝对长度单位:

|  |  |
| --- | --- |
| in | 英寸Inches(1英寸=2.54厘米) |
| cm | 厘米Centimeters |
| mm | 毫米Millimeters |
| pt | 磅Points(1 磅 = 1/72 英寸)(打印时常用) |
| pc | 皮卡Picas(1 皮卡 = 12 磅) |

## 相对长度单位:

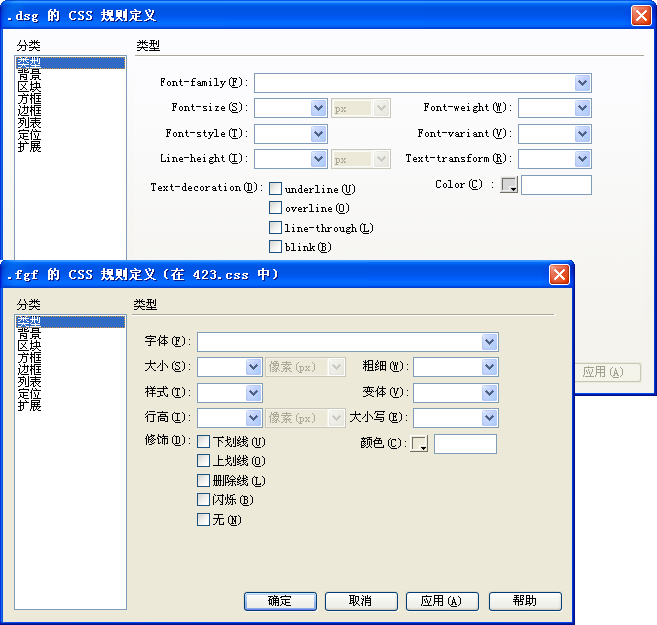
|  |  |
| --- | --- |
| em | EM中字体的高度, 也就是文本中font-size的值. 例如定义某个元素的文字大小为12pt, 那么对于这个元素来说1em就是12pt. 单位em 的实际大小是受到字体尺寸影响的. |
| ex | ex中字母x的高度, 因为不同的字体x的高度是不同 , 所以ex的实际大小, 受到字体和字体尺寸两个因素的影响. |
| px | 像素Pixels, 实际上px的大小是受到屏幕分辨率影响的, 也就是和划分屏幕个子的方式有关. |
| % | 百分比Percentage |

## 百分比值

# 常用代码

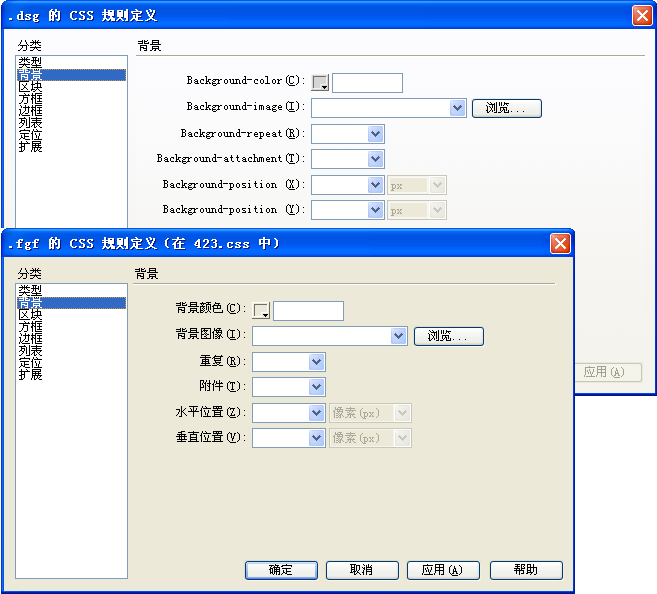
**DW中CSS设计**

**类型(文字)**



|  |  |
| --- | --- |
| font-family | 字体 |
| font-size | 绝对大小: xx-small, x-small, small, medium, large, x-lare, xx-large  相对大小: larger, smaller  绝对单位: px像素, mm, cm, in英寸, pt磅=1/72in, pc帕=12pt  相对单位: em当前属性值大小的的几倍, ex:当前字体是x字母的几倍, %:方法几倍 |
| font-style | 斜体还是正常 |
| color | #rrggbb  rgb(r, g, b)  red |

**背景**



|  |
| --- |
|  |

背景图片显示位置

|  |  |  |
| --- | --- | --- |
| 关键字 | 等量百分比值 | 图示 |
| top left  left top | 0% 0% | |  |  |  | | --- | --- | --- | | 1 | 0 | 0 | |  |  |  | |  |  |  | |
| top  top center  center top | 50% 0% | |  |  |  | | --- | --- | --- | | 0 | 2 | 0 | |  |  |  | |  |  |  | |
| right top  top right | 100% 0% | |  |  |  | | --- | --- | --- | | 0 | 0 | 3 | |  |  |  | |  |  |  | |
| left  left center  center left | 0% 50% | |  |  |  | | --- | --- | --- | |  | 0 | 0 | | 4 |  |  | |  |  |  | |
| center  center center | 50% 50% | |  |  |  | | --- | --- | --- | | 0 |  | 0 | |  | 5 |  | |  |  |  | |
| right  right center  center right | 100% 50% | |  |  |  | | --- | --- | --- | | 0 | 0 |  | |  |  | 6 | |  |  |  | |
| bottom left  left bottom | 0% 100% | |  |  |  | | --- | --- | --- | |  | 0 | 0 | |  |  |  | | 7 |  |  | |
| bottom  bottom center  center bottom | 50% 100% | |  |  |  | | --- | --- | --- | | 0 |  | 0 | |  |  |  | |  | 8 |  | |
| bottom right  right bottom | 100% 100% | |  |  |  | | --- | --- | --- | | 0 | 0 |  | |  |  |  | |  |  | 9 | |

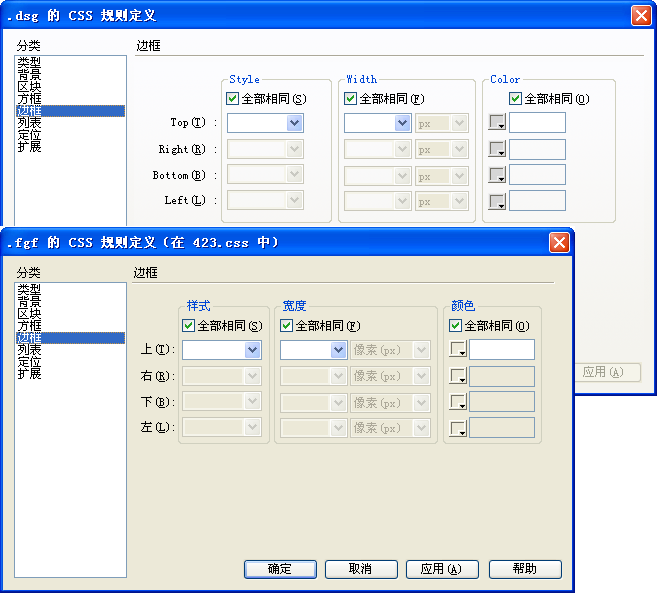
**区块**



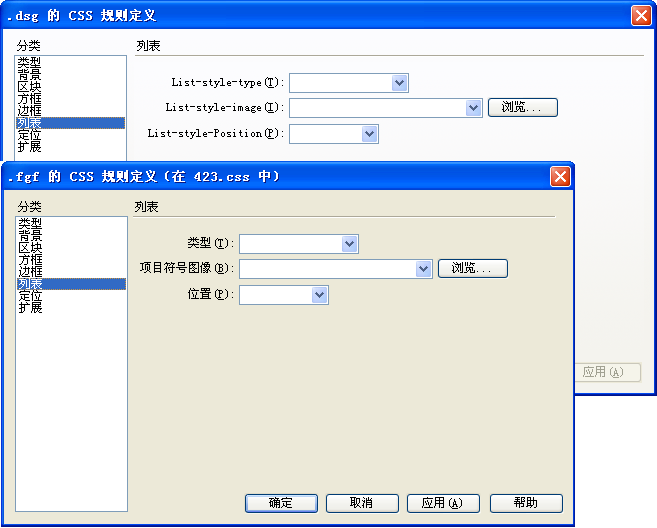
**方框**



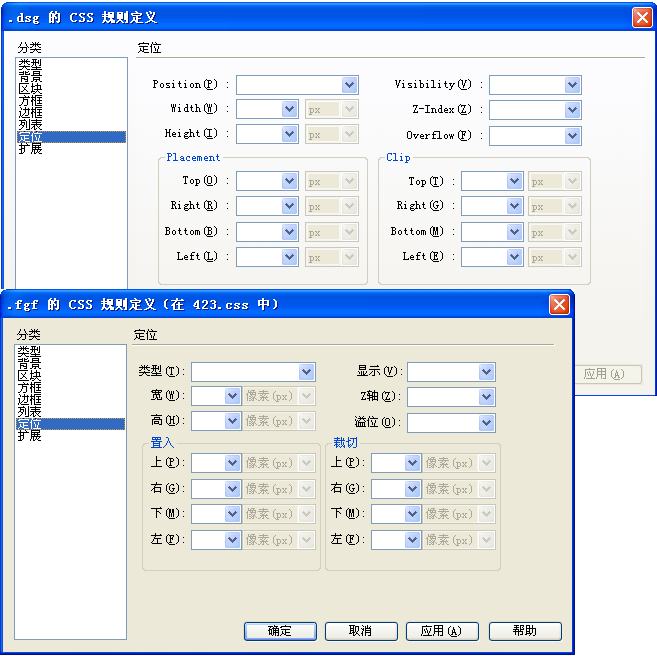
**边框**



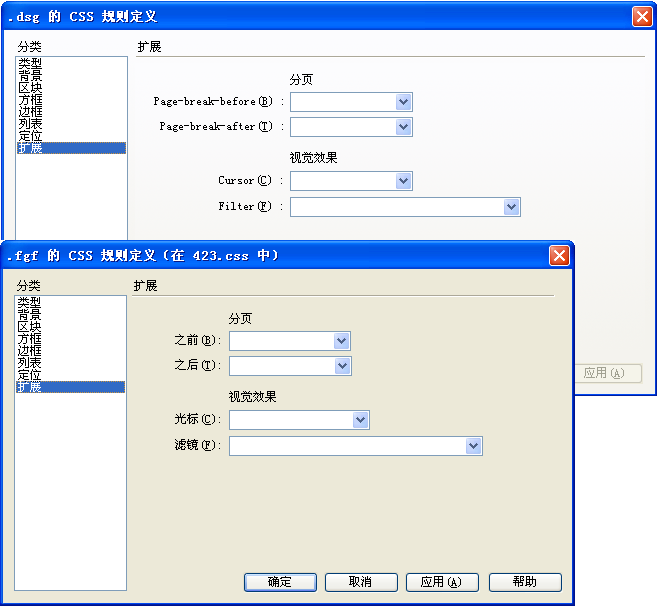
**列表**



**定位**



**扩展**



# 笔记

## 格式化模型

|  |  |
| --- | --- |
| 显示 或 隐藏 对象display | inline: 行内容器  block: 块级容器  list-item: 列表项目  none: 不显示 |
| 绝对定位 和 相对定位position | static: 不会对容器定位产生影响.  relative: 相对. 容器仍然是普通布局, 但按照它应该所在的位置进行定量的left, top偏移后显示.  absolute: 绝对. 和relative相对, 是绝对定位, 容器已经离开了普通的页面布局, 完全抛弃容器的原来位置, 放在页面上的任意地方.绝对定位的容器不会和任何其他元素融合边距.  fixed: 定位方式和absolute相同, 但它的作用一般仅限于在打印或其他显示媒体上进行绝对定位, 而很少用在普通网页制作上.  具体的位置坐标由top/right/bottom/left属性设定 |
| 对象位置位移 | 长度值 | 百分比值 | auto | 继承值   |  |  | | --- | --- | | top | 元素相对于其父元素上边线的距离 | | right | 元素相对于其父元素右边线的距离 | | bottom | 元素相对于其父元素下边线的距离 | | left | 元素相对于其父元素左边线的距离 | |
| 对象在页面上浮动 float | left | right | none | 继承值 |
| 对象周边内容清除clear | |  |  | | --- | --- | | left | 不允许左边有浮动元素 | | right | 不允许右边有浮动元素 | | both | 两边都不允许由浮动元素 | | none | 允许两边都有浮动元素 | | 继承值 |  |   clear属性通常和float属性结合使用, |
| 上下层叠 和 覆盖z-index | auto | 整数 | 继承值 |
| 控制文本方向 direction 和 unicode-bidi | ltr | rtl | 继承值 |

## 尺寸

|  |  |
| --- | --- |
| width | 长度值 | 百分比值 | auto | 继承值 |
| height | 长度值 | 百分比值 | auto | 继承值 |
| min-width  max-width  min-height  max-height | 长度值 | 百分比值 | 继承值 |
| line-height 文本行高 | normal | 数字 | 长度值 | 百分比值 | 继承值 |
| vertical-align  行内元素 或 单元格元素: 垂直对齐方式 | |  |  | | --- | --- | | baseline默认 | 底端对齐, 即和下侧基准线对齐 | | middle | 中线对齐 | | sub | 下标位置对齐 | | super | 上标位置对齐 | | text-top | 和文本上沿对齐 | | text-bottom | 和文本下沿对齐 | | top | 和容器顶端对齐 | | bottom | 和容器底端对齐 | | 百分比值 | 用百分比值控制垂直对齐方式 | | 长度值 | 用绝对长度值控制垂直对齐方式 | | 继承值 |  | |

## 溢出 和 剪切

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 内容溢出处理overflow | |  |  | | --- | --- | | visible | 自动扩大容器边框(默认) | | hidden | 益处部分被剪切而不可见 | | scroll | 即使内容不益处也出现滚动条 | | auto | 只有内容益处才出现滚动条 | | 继承值 |  | |
| 内容剪切处理clip | clip: rect(顶端坐标, 右侧坐标, 下方坐标, 左侧坐标) | 默认auto | 继承值  clip属性仅作用于绝对定位的元素的矩形剪切, 它定义一个容器的哪个部分可见, 哪个部分被剪切. 默认auto为不剪切 |
| 内容可见性 visibility | |  |  | | --- | --- | | visible | 可见 | | hidden | 隐藏 | | collapse | 隐藏表格中的行和列 | | 继承值 |  |   visibility隐藏: 任占有空间  display隐藏: 不占有空间 |

间隔 和 空白

|  |  |
| --- | --- |
| letter-spacing | normal | 继承值  (汉字之间/字母之间)的距离 |
| word-spacing | normal | 继承值  单词之间的距离, 汉字中国没有单词这个文字单位, 所以对汉字不起作用. |
| white-space | |  |  | | --- | --- | | normal | 默认值, 忽略多余的空白 | | pre | 不忽略多余的空白 | | nowrap | 文本保持同一行显示, 直到文本结束或者遇到br元素 | |

# 实例

!important(优先权)

|  |
| --- |
| 使用!important的属性具有较高的优先权, 也就是说, 会被优先使用.  主要用来兼容IE6.0和Firefox的. |
| !important声明一般写在定义属性值之后, 结束符";"之前. 语法格式: 属性:属性值 !important; |
| |  |  |  | | --- | --- | --- | | Firefox中 | 同时给一个元素定义相同的属性, 最终使用的是优先级比较高的值. |  | | IE6.0中 | 也识别这个属性, 只不过是!important声明没有起作用 |  | |

# 在所有浏览器中都有效的方法

|  |
| --- |
| <div style=*"border-bottom: 0px; position: relative; border-left: 0px; background-color: #aff; width: 760px; margin-left: -380px; border-top: 0px; border-right: 0px; left: 50%"*>  　　在所有浏览器中都有效的方法：  <div> |

## 所有内容居中

|  |
| --- |
| #content{  margin-left: auto;  margin-right: auto;  margin-top:-10px;  width: 950px;    background-color:#CCC;  } |

## 块居中

|  |
| --- |
| #logo {  margin-top: 20px;    text-align:center;  margin-right:auto;  margin-left:auto;  } |

## 项目列表

|  |
| --- |
| ul{  font:74%/1.4em Arial,Sans-Serif;  color:#A0A0A0;  font-size:95%;  list-style:none outside none;  margin:5px 0 20px 15px;  padding:0;  }  li{  font:74%/1.4em Arial,Sans-Serif;  color:#A0A0A0;  font-size:95%;  list-style-type:square;  margin:0;  padding:0;  } |

## 单行文本输入框

|  |
| --- |
| input.text {  margin: 0;  width: 110px;  border: 1px solid #F29900;  background: #FFF;  color: #808080;  font-size:18px;  width: 122px;  height: 25px;  } |

## 表单提交按钮

|  |
| --- |
| input.searchbutton {  margin: 0;  font-size: 100%;  font-family: Arial, Sans-serif;  border: none;  background: #FFFFFF;  color: #808080;  padding: 1px;  font-weight: bold;  } |

## 淘宝色调

|  |  |
| --- | --- |
| 蓝色 | #498CD0 |
| 橙色 | #F68911 |
| 文本输入框样式 | border: 1px solid #C8C8C8;  height: 18px;  line-height: 18px; |
|  | border: 1px solid #7F9DB9; |

## table属性

|  |
| --- |
| **<style type="text/css" media="screen, projection">**  **<!--**  .myTable{  border-collapse:*collapse*;  border:*dotted 1px #666*;  }  .myTable td{  padding:*8px 10px 8px 10px*;  border: *solid 1px #666*;  }  **//-->**  **</style>** |

## 背景图片(大图中的一部分)

|  |
| --- |
| style=*"background:url(../image/content5\_bg.png) no-repeat scroll 1px -76px transparent;"*  注释:  transparent:透明 |

# 浏览器兼容问题

## IE6/IE7/IE8/FF的区别

|  |
| --- |
| /\*For IE6\*/  \* html .photo-image-border {  width: 128px;  height: 72px;  text-align: center;  }  width:expression(document.body.clientWidth > 800? "800px": "auto" );  this.width  this.height |

在firefox，某些时候content高度无法随内容自增长，解决办法：.content{overflow: hidden;}

图片作提交form按钮

|  |
| --- |
| <form action=*"app/session/findPassword"* name=*"user"* method=*"post"*>  <input type=*"submit"* value=*""* style="width:*0*; height:*0*; padding:*0*; border:*none*;" />  <a href=*"javascript:void(0);"* onclick="document.user.submit(); return false;"><img src=*"./images/findpass.png"*></a>  </form> |

## checkbox 和radio 与文字居中对齐

|  |
| --- |
| input[type="checkbox"],input[type="radio"] {  vertical-align: middle;  margin: 0 3px;  padding: 0;  }  label {  vertical-align: middle  } |

设置透明度, 兼容各浏览器

filter:alpha(opacity=0); -moz-opacity:0; -khtml-opacity: 0; opacity: 0;

# 字体

|  |  |
| --- | --- |
| font-family: Constantia,Georgia; |  |
|  |  |

# html 强制换行

|  |
| --- |
| 通常我们在编写html页面的时候，我们一般都会在<td>或者<div>里面通过设定样式来控制 table 或者 div 的宽度  例如:  <table width="603" border="1" cellspacing="0" cellpadding="0">   <tr>     <td style="width:200px;">哈哈哈哈哈哈哈哈哈哈哈哈哈哈哈哈哈哈哈</td>     <td style="width:200px;">1231231231</td>     <td style="width:200px;">123123123</td>   </tr> </table>  效果:  http://hi.csdn.net/attachment/201109/7/0_1315361936yxUg.gif  但是如果有恶意输入呢？     <td style="width:200px;">111111111111111111111111111111111111111111111111111111111111</td> 效果：  http://hi.csdn.net/attachment/201109/7/0_1315362099rEdD.gif  这样我们的设定的width就无效了，原因是因为：浏览器在解析我们页面的时候，给这一串1当成一个单词了，这样就不会自动切断字符串而进行换行。  解决方案：  <td style="width:200px;word-wrap:break-word;word-break:break-all">111111111111111111111111111111111111111111111111111111111111</td>  效果：  http://hi.csdn.net/attachment/201109/7/0_131536294094Xf.gif  word-wrap:break-word; 和 word-break:break-all 的区别  1，word-break:break-all 例如div宽200px，它的内容就会到200px自动换行，如果该行末端有个英文单词很长（congratulation等），它会把单词截断，变成该 行末端为conra(congratulation的前端部分)，下一行为tulation（conguatulation）的后端部分了。 2，word-wrap:break-word 例子与上面一样，但区别就是它会把congratulation整个单词看成一个整体，如果该行末端宽度不够显示整个单词，它会自动把整个单词放到下一行，而不会把单词截断掉的。  word-break;break-all 支持版本：IE5以上 该行为与亚洲语言的 normal 相同。也允许非亚洲语言文本行的任意字内断开。该值适合包含一些非亚洲文本的亚洲文本。 WORD-WRAP:break-word 支持版本：IE5.5以上 内容将在边界内换行。如果需要，词内换行( word-break )也将发生。 |

# 打印

普通纸张按克重可分为60gsm、80gsm、100gsm、105gsm、120gsm、157gsm、200gsm、250gsm、300gsm、350gsm、400gsm。

GSM是PP编制产品的一种计量单位，名字为“克重”，gsm = g per square meter 每平方米的重量(克）

打印纸尺寸：

|  |  |
| --- | --- |
| A3  A4  A5  打印纸尺寸（单位是毫米）  A0=1189\*841  A1=841\*594  A2=594\*420  A3=420\*297  A4=210\*297  A5=148\*210 |  |
|  |  |

firefox打印宽度

|  |  |
| --- | --- |
| A4 |  |
| A5 | width:756px; height:500px; margin:0 auto; overflow:hidden; |