精通CSS 高级Web标准解决方案

1. 代码结构
2. 语义元素
3. ID和类名 使用现有的元素，并且通过添加ID或类名给它们赋予额外的意义。

ID用于标识界面上的特定元素，必须唯一。

同一个类名可以应用于页面上任意多个元素，因此类的功能强大的多。类非常适合标示内容的类型或其他相似的条目。

类名和 ID区分大小写。

1. Div span 有助于在文档中添加结构的一个元素是div元素。Div代表division。 Div用于块级元素，span用于行内元素。
2. 为样式找到应用目标

2.1． 常用的选择器：类型选择器和后代选择器。类型选择器用来寻找特定类型的元素，如段落或标题元素。

类选择器和后代选择器。 后代选择器由其他两个选择器之间的空格表示。

ID选择器由#字符表示，类选择器由一个点号 . 表示。

2.12．伪类 文档结构之外的其他条件对元素应用样式，例如表单元素或链接的状态，需要伪类选择器完成。

2.2. 通用选择器 通配符 由 \* 组成。

2.3 高级选择器 子选择器 和相邻同胞选择器 第一个高级选择器是子选择器。后代选择器选择一个元素的所有后代，而子选择器只选择元素的直接后代，即子元素。

相邻同胞选择器可以用于定位同一个父元素某个元素之后的元素。

属性选择器 属性选择器可以根据某个属性是否存在或属性的值来寻找元素。

1. 层叠和特殊性

同一元素可能有两个或更多规则。CSS通过一个叫层叠的过程处理这种冲突。层叠给每个规则分配一个重要度。

层叠次序排序：

标有 ！important 的用户样式

标有！important 的作者样式

作者样式

用户样式

浏览器/用户代理应用的样式。

1. 特殊性 为了计算规则的特殊性，为每种选择器都分配一个数字值。

选择器的特殊性分成四个成分等级：a、b、c和d。

如果样式是行内样式，那么a=1;

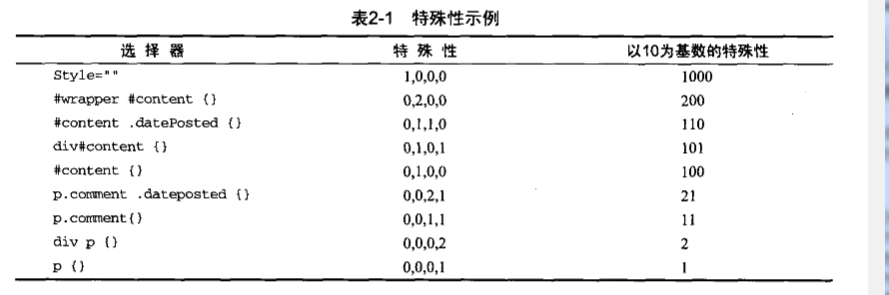
B等于ID选择器的总数。

C等于类、伪类和属性选择器的数量。

D等于类型选择器和伪元素选择器的数量。

A B C D分别是 10的三次方 10的二次方 10的一次方 10的0次方。

总结一下： 用style属性编写的规则总是比其他任何规则特殊。具有ID选择器的规则比没有的要特殊。如果两个骨子额特殊性相同，那么后定义的规则优先。



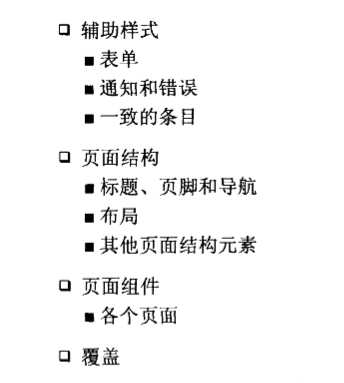
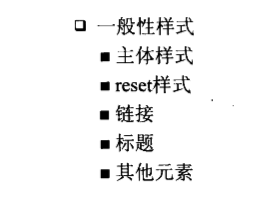
1. 继承 应用样式的元素的后代会继承样式的某些属性。比如颜色和色号。
2. 规划、组织和维护样式表

对文档应用样式 CSS允许将所有的样式放在一个或多个外部样式表中。可以链接它们，导入它们。

设计代码的结构

为了方便维护，先写一般规则，然后特殊样式和辅助样式。

整个文档结构



第三章 可视化盒模型

要掌握的3个最重要的CSS概念是浮动、定位和盒模型。

1. 盒模型

盒模型是CSS的基石之一，它指定元素如何显示以及（在某种程度上）如何相互交互。页面上每个元素被看做一个矩形框。这个框由元素的内容、内边距、边框和外边距组成。

内边距出现在内容区域的周围。 如果在元素上加背景，那么背景会应用于由内容和内边距组成的区域。因此，我们常常使用内边距在内容周围创建一个隔离带，使内容不会与背景混在一起。

外边距是透明的，一般使用它控制元素之间的间隔。

外边距叠加：当两个或更多垂直外边距相遇时，它们将形成一个外边距、这个外边距的高度等于两个发生叠加的外边距的高度中的较大者。

1. 定位概述

可视化格式模型和定位模型，理解这两个模型的细微差异。

1. 可视化格式模型

P,h,div元素常称为块级元素。即块框。Strong,span常称为行内元素，因为它们的内容显示在行中，即行框。

Display属性设置为block，可以让行内元素表现得像块级。将display设置为none，让生成的元素没有框。框及内容不再显示。

块级框从上到下一个接一个地垂直排列。框之间的垂直距离由框的垂直外边距计算出来。

行内框在一行中水平排列。可以使用水平内间距、边框和外边距计算它们的水平间距。

1. 相对定位

如果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”它的起点移动。如果将top设置为20像素，那么框将出现在原位置顶部下面20像素的位置。

在使用相对定位时，无论是否移动，元素仍占据原来的空间。因此移动元素会导致它覆盖其他框。

1. 绝对定位

相对定位实际上被看作普通流定义模型的一部分，以为元素的位置是相对于它在普通流中的位置。与之相反，绝对定位使得元素的位置与文档流无关，因此不占据空间，普通文档流中其他元素的布局就像绝对定位的元素不存在一样。

相对问题： 相对定位相对于元素在文档流中初始位置。

绝对定位： 相对于距离它最近的已定位祖先元素，如果不存在已定位的祖先元素，那么相对于初始包含块。

绝对定位的元素与文档流无关，所以它们不影响普通流中的框。因此，尺寸的任何改变都会导致绝对定位的框产生重叠，从而破坏精心调整的布局。

* 1. 固定定位

固定定位是绝对定位的一种，差异在于固定元素的包含块是视口（viewport），这使我们能够创建总是出现在窗口相同位置的浮动元素。

4.浮动 浮动的框可以左右移动，直到它的外边缘碰到包含框或另一个浮动框的边缘。

浮动会让元素脱离文档流，不再影响不浮动的元素。

行框和清理

浮动元素旁边的行框被缩短，从而给浮动元素留出空间，因此行框围绕浮动框。创建浮动框可以使文本围绕图像。

阻止行框围绕在浮动框的外边，需要对包含行框的元素应用clear属性。Claser属性可以是left,right,both或者none.它表示哪些边不该挨着浮动框。

Overflow属性定义在包含的内容对于指定的尺寸太大的情况下元素应该如何。

4.浮动

第八章 布局

1. 计划布局 结构性布局 内容布局
2. 设置基本结构
3. 假设三列博客模板

<div class=”wrapper”>

<div class=”header”></div>

<div class=”content”></div>

<div class=”footer”></div>

</div>

使用外边距让设计居中(定义div宽度，水平外边距设为auto) auto左右自适应，两边距离一样。

.wrapper{ width:920px; margin: 0 auto;}

Text-align 居中，不止是文本。

1. 基于浮动的布局

基于浮动的布局容易使用，可靠。

1. 两列的浮动布局

<div class=”content”>

<div class=”primary”>”</div>

<div class=”secondary”>”</div>

</div>

主要内容中增加一点内边距。Display:inline 用来防止双外边距浮动产生的bug。

.content .primary{ Padding-right: 20px; float: right; display:inline}

.content .secondary{ width: 230px; float: left; display:inline}

因为元素是浮动的吗，它们不在文档流中占据任何空间，。这会导致页脚向上升。避免这种情况，需要对它们的父元素（这里是div）应用溢出方法，从而清除浮动元素。

.content{ overflow: hidden;}

1. 固定宽度、流式和弹性布局

到目前为止，所有实例中的宽度都以像素为单位，这种布局类型称为固定宽度的布局。

固定宽度布局优点： 简便、常用 固定宽度布局缺点：不适应灵活的web.

为了解决这些问题，可以使用流式布局或弹性布局替代固定宽度布局。

流式布局问题：窗口宽度较窄，行很耐阅读。有必要添加min-width.从而防止布局太窄。 不要横跨浏览器而是让容器只跨越宽度的一部分，比如80%,

以容器宽度的百分数形式设置主内容区域和次要内容区域。

.wrapper{ width: 76.8%; margin: 0 auto; text-align:left;}

.content .primary{ width:72.82% float:right; display:inline}

.content .secondary { width:25% float:left; display:inline}

对于比较小的窗口尺寸，布局会挤。添加以em为单位的min-width.

.wrapper{width:76.8%;margin: 0 auto; text-align:left;max-width:125em; min-width:62em}

1. 弹性布局

在大分辨率显示器上，行仍然会过长，让用户不舒服。