1. 基础知识

想要建立好的CSS，首先要有好的html。

html尽量只考虑内容，布局问题交给CSS，尤其不要使用表格布局，太落后

有意义的标记尽量优先使用，如果没有，用div和span描述

独一无二的用id，同一类型的用class

id名和class名用小写，连字符分隔，名字最好能描述内容

不同版本css和html

现行的css主要版本是css2.1，css3已经被浏览器厂商支持，但支持程度还不够

现行html版本：

HTML4.01

XHTML1.0 与HTML4.01主要差别在于遵守XML规定，XHTML属性必须包含引号，元素必须封闭，相比之下更加严格

XHTML1.1 作为XML发给浏览器的，只要有一个错误浏览器不显示页面，已被放弃

HTML5 火的飞起

DTD

文档类型，一组机器可读的规则，告诉浏览器当前页面是那个版本HTML5? XHTML?

doctype

声明使用那个DTD，一般包含指定DTD文件的URL但是不读取这些文件

有效性验证

浏览器必须知道使用什么DTD才能正确处理页

浏览器模式

为了向后兼容

标准模式，混杂模式

混杂模式模仿老浏览器，体现向后兼容

doctype切换用来区分老式文档和符合标准的文档

doctype存在且完整，表现为标准模式；选择错误的doctype或者不完整的doctype会使用混杂模式，出现意向不到的问题。根据doctype是否存在选择呈现模式称为doctype侦测

每个页面上最好包含完成的doctype声明，选择严格的DTD

除非要将页面作为XML，否则不要使用XML声明

如果doctype不是页面的第一个元素IE6会自动切换混杂模式

1. 给样式找到应用目标

常用选择器

不要指定太多了类，尽量将一个类应用到共同的祖先元素，然后使用后代选择器定位到要应用样式的元素

伪类选择器

：link , :visited 链接伪类，只应用于锚元素

:hover , :active , :focus 动态伪类，应用于各种元素

IE6只注意应用于链接的:hover , :active

IE7支持任何元素上的:hover , 但是忽略:active , :focus。

高级选择器

对于重要的元素尽量少用高级选择器（考虑向后兼容的问题）IE6和更低版本不支持

子选择器

#nav>li

选择直接后代

在IE6上模仿子选择器

#nav li{ background: … ; left: 10px;}

#nav li \* {background: none; left : 0px;}

IE7上，如果父元素和子元素之间有注释，出问题

相邻同胞选择器

h2+p {…}

定位于同一个父元素下某个元素之后的元素

IE7上，如果父元素和子元素之间有注释，出问题

属性选择器

p[title] 选择具有title属性的p元素

p[title = “ zq ”] 选择具有title属性且值为zq的p元素，一些属性可能具有多个值，只要其中一个符合选择条件就被选中

IE7一下不支持属性选择器，最好优雅降级

层叠和特殊性

层叠的重要次序：

标有！important的用户样式

标有！important的作者样式

作者样式

用户样式

浏览器默认样式

选择器的特殊性

a,b,c,d

a表示用style属性规定样式

b表示id选择器的总数

c表示类选择器的总数

d表示元素选择器的总数

div#content 0101 101

divlass 0011 11

101 > 11 第一个选择器的特殊性大于第二个，优先级更高

如果两个选择器具有相同的优先级，后一个覆盖前一个

css不起作用一般是特殊性冲突，给要选择的元素的父元素增加一个ID

如果要多次覆盖一个样式，在较一般的样式表中将其删除，在特殊表中定义

给body添加不同的类或者ID完成换主题功能

继承

应用样式的元素的后代元素也会拥有同样的样式

如，body文本大小为1.4em，color：black；后面的元素字体大小/颜色和body相同。

表格中的字号的继承必须指定或者单独设定

继承而来的样式特殊性为空，即浏览器默认样式也会覆盖继承来的样式

body的字体样式被h1默认样式覆盖

规划组织和维护样式表

样式表添加方式

<link …>

@import url( …)这个方法导入样式较慢

建议把样式表放到同一个文件中，提高加载速度，便于维护

注释是个好习惯

添加注释标头 /\*@group\*/ 方便寻找特定样式，样式表可能会非常长

编写CSS从一般到特殊

上传到服务器的时候尽可能的删除掉注释减少代码量

最好的方式是使用服务器压缩，大大减少加载负荷

GZIP压缩的文件浏览器一般都能呈现

留一个带注释的版本用于本地调试

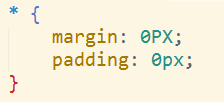
1. 可视化格式模型

盒模型

给元素添加背景，背景区域包括内容和内边距

外边距一般是透明的，控制元素之间的距离

内边距外边距有浏览器默认值，去除浏览器影响，全局reset将他们设置为0



width, height指的是内容区域的宽和高

outline属性是在整个元素框外绘制轮廓，且轮廓线不占空间，不影响页面布局

IE盒模型

IE6及更早版本在混杂模式中使用自己的特殊盒模型

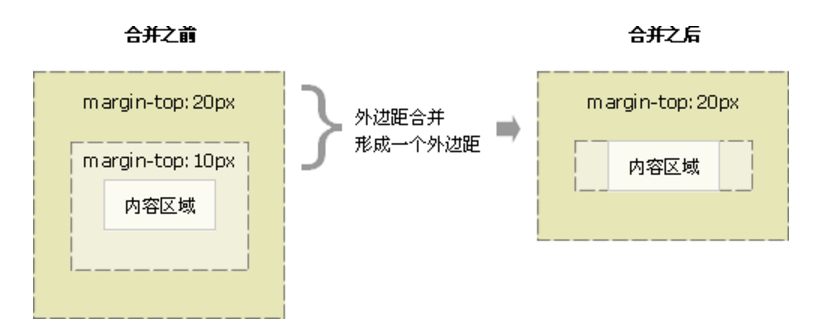
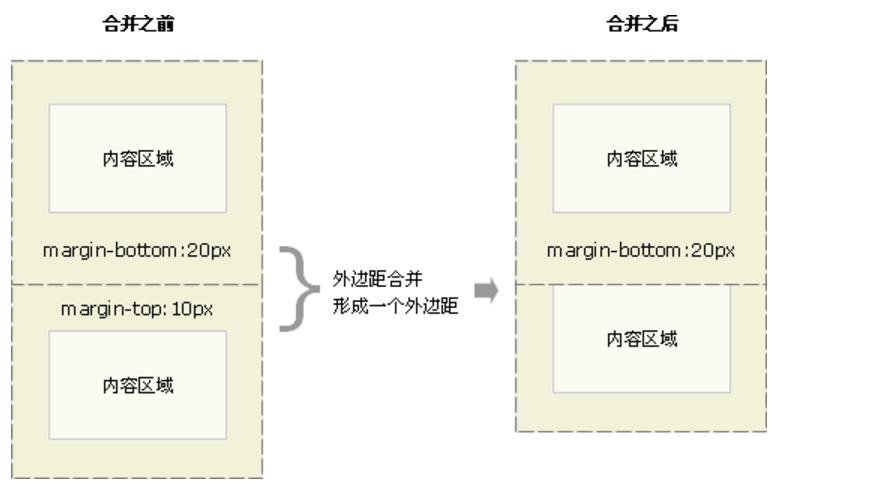
width表示整个元素框的宽度

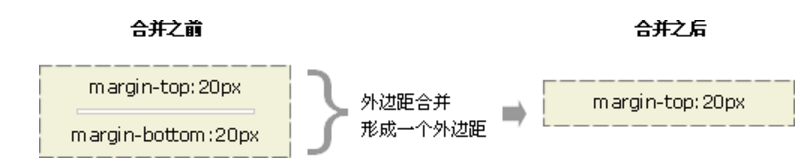
解决办法：

不要给元素添加指定宽度的内边距，尝试将内边距和外边距加到元素的父元素或子元素

外边距叠加

当两个或者多个垂直外边距相遇时，他们将形成一个外边距，这个外边距的高度是叠加的外边距中最大的外边距的高度





只有文档流中的块框会产生外边距叠加，行内框，浮动框，绝对定位框，相对定位框不会产生叠加

可视化格式模型

块级元素，在文档流中形成块框，按照html中顺序垂直排列

行内元素，在一行中水平排列。

display属性：块级block 以块元素形式呈现

行内：inline 以行内元素形式呈现

inline-block 以块元素的形式在一行上水平排列

none 不显示这个框，并且不占据文档中的空间

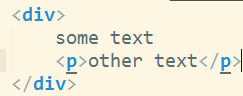
可以修改display属性控制元素的呈现方式

行内框：

可以通过内边距外边距边框调整水平间距，但是不能通过他们调整行内框的高度，在行内框显示设置宽度和高度也是没有卵用，修改行高line-height可以改变高度。

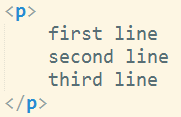
同一行的行内框会形成一个总的水平框称为行框，行框包含了这些行内框，行框的高度是所包含行内框的最大高度

匿名块框



some text会以块级元素方式显示

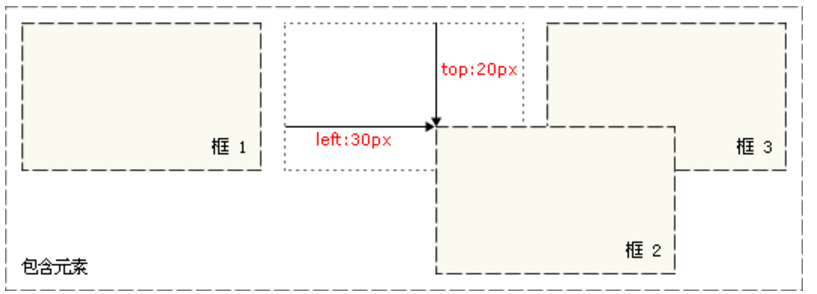
匿名行框



三行文本会以三个行内元素形式呈现

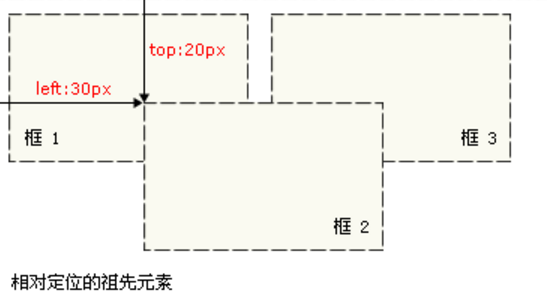
无法直接对匿名块框和匿名行框应用样式

相对定位



相对与元素在文档流中的位置进行定位，元素仍然占据原来的空间，可能会覆盖其他框

绝对定位



相对于已经定位的离它最近的祖先元素（一般祖先进行相对定位）进行定位，元素脱离文档流不占据原来的空间，文档流中其他元素就当该元素不存在继续按照文档流排列

覆盖其他元素时，可以通过设置z-index来控制叠放顺序，z-index大的在上面

绝对定位必须注意元素尺寸，改变尺寸会对布局产生很大影响

IE6的bug：

相对与相对定位的框的右边或底部进行绝对定位，必须确保相对定位的框已经设置了尺寸，不然IE会相对于画布定位这个框

固定定位

相对于视口进行绝对定位

浮动

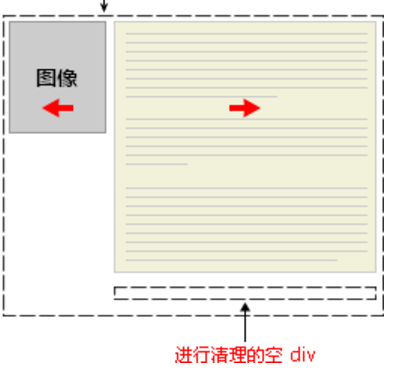
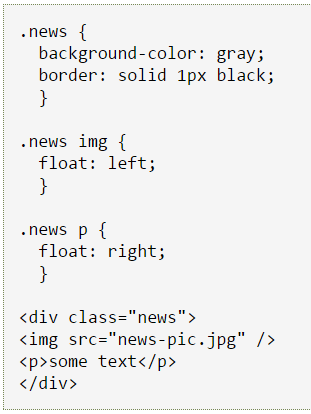
脱离文档流，文档流中其他的元素当该元素不存在进行排列

浮动框可以左右移动，直到碰到包含框或者另一个浮动框的边缘

文档流中不浮动的框不管浮动的框继续排列，但不浮动的框的文字内容会受到影响，会压缩从而给浮动框留出空间而不是覆盖

clear属性阻止行框围绕在浮动框

做法是给clear的元素增加上外边距，使元素垂直下降到浮动框下面，为前面的浮动元素留出足够的垂直距离

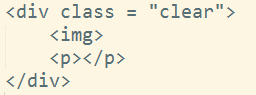
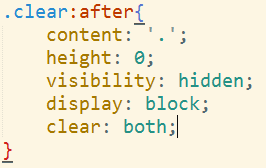


对于这种情况，<img> <p>会脱离文档流，使得<div>不占空间，想要让<div>视觉上包围浮动元素，就需要在<div>里包含一个空元素并应用clear: both; 样式。

可以浮动布局中的所有东西，然后用页脚元素进行clear

overflow属性应用值为hidden或auto的时候，会自动清理包含的任何浮动元素

CSS动态的添加清理元素到页面中

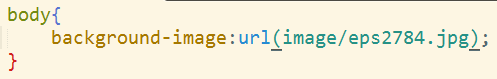


CSS的内容表示添加一个点到.clear的元素后面，并且不显示这个点

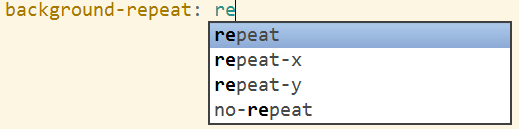
display: block；是因为被清除的元素在他们顶外边距添加了空间

1. 背景图像效果

应用背景图像



尽量不要直接在html中插入图片，通过css的背景图把内容与表现分离



repeat表示X,Y方向上平铺背景图片

repeat-X表示x方向平铺

repeat表示不平铺

background-position：left center

水平左，竖直方向居中

background-position：20% 20%

百分比表示使用图片上距离图片左上角20%，20%的点对应到距离父元素左上角20%，20%的点