目录

[客户端与服务器 2](#_Toc47169458)

[补充1（行间距）： 2](#_Toc47169459)

[HTML5语义化概念： 2](#_Toc47169460)

[标签讲解 4](#_Toc47169461)

[CSS层叠样式表： 5](#_Toc47169462)

[文档流： 7](#_Toc47169463)

[盒模型 8](#_Toc47169464)

[浮动难点 8](#_Toc47169465)

[BFC(Block Formatting Context) 块级格式化环境 10](#_Toc47169466)

[clear 11](#_Toc47169467)

[查看04\_float---07高度塌陷的最终解决方案 12](#_Toc47169468)

[定位（position） 12](#_Toc47169469)

[- 相对定位（relative）： 13](#_Toc47169470)

[- 绝对定位（absolute） 13](#_Toc47169471)

[包含块( containing block ) 14](#_Toc47169472)

[- 固定定位（fixed）： 14](#_Toc47169473)

[修改元素层级（开启了potision的元素） 15](#_Toc47169474)

[vertical-align（垂直对齐） 设置元素垂直对齐的方式 15](#_Toc47169475)

[为内容多的段落设置省略号可以看 16](#_Toc47169476)

[背景相关知识 16](#_Toc47169477)

[图片背景的闪烁问题 17](#_Toc47169478)

[解决图片闪烁的问题(利用雪碧图CSS-Sprite) 18](#_Toc47169479)

[雪碧图CSS-Sprite的使用步骤： 19](#_Toc47169480)

[渐变 19](#_Toc47169481)

[Table的相关知识 20](#_Toc47169482)

[补充知识2 20](#_Toc47169483)

[补充知识3 21](#_Toc47169484)

[过渡、动画、变形（在x、y、z轴上移动） 22](#_Toc47169485)

[css3新知识：弹性容器（display:flex） 22](#_Toc47169486)

[响应式布局（关键点：媒体查询== @media 查询规则{}） 22](#_Toc47169487)

# 客户端与服务器

通常情况下，现在的软件一般分为两个部分：客户端和服务器

客户端：用户通过客户端来使用软件。（本地）

服务器：服务器负责在远程处理业务逻辑。（服务器）

程序员要么写客户端、要么写服务器。

服务器开发语言：Java、PHP、C#、Python、

Node.js（利用js代码编写）【必须学、且要学号】

前端工程师更多的是写客户端。

**客户端的形式**：

文字客户端、图形化客户端、

网页

H5文档声明：

<!Doctype HTML>

# 补充1（行间距）：

特别记住一点：行高line-height = font-size + 行间距；所以可以利用这个式子来计算一些段落的样式

# HTML5语义化概念：

在网页中HTML专门用来负责网页的结构，所以在使用html标签时，应该关注的是标签的语义，而不是它的样式。

在页面中独占一行的元素称为块元素,如h标签和p标签

在页面中不会独占一行的元素叫做行内元素，如em标签

<q>短引用（行内元素）、<blockquote>长引用（块元素）

块元素（block element）

在页面中一般通过块元素对页面进行布局

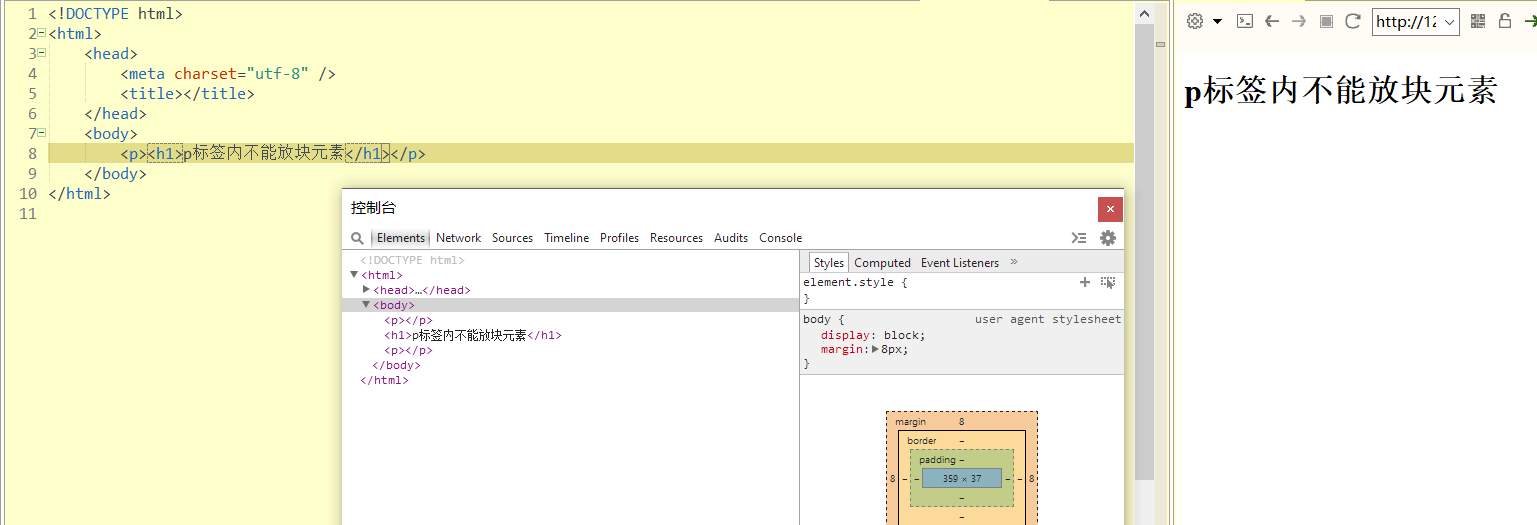
行内元素（inline elemnet）

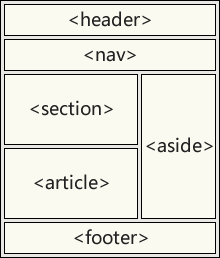
行内元素主要用于包裹文字

一般情况下会在块元素内放行内元素，而不会在行内元素放块元素

（这里就有一个例外就是，a标签虽然是行内元素，但是a标签内部既可以放行内元素也可以放块元素，唯独a标签内不能放a标签）

块元素中基本上什么都能放，而规定p元素不能放任何块元素

如果在出现一些错误排版，浏览器在解析浏览器的时候回检测错误

如上图所示在p标签内加入了h1标签，浏览器在识别的时候自动修改成了，p标签+h1标签+p标签。(浏览器的修改体现在内存中，不会去改你自己的代码，这就是为什么要在console上看elements的原因)

<nav>表示页面中的导航，aside表示侧边栏，article表示独立的文章。都是语义化的标签。，section表示一个独立的区块，当前面几个都不能表示的时候就用section。

Div是块元素，span是行内元素，不像上面的标签，他们没有任何的语义。

# 标签讲解

<a>标签的href可以跳到一个页面的指定锚点如href=#mya会跳到一个属性id名为mya处

<img>这种元素属于替换元素（块和行元素之间，具有两种元素的特点）。而img元素类的alt属性的目的是为了在搜索引擎会根据alt中的内容来识别图片，如果不写alt属性则图片不会被搜素引擎所识别。

Base64编码，对图片进行编码。将图片装换为base64字符串。一般在图片和页面一起加载时才会用，网页加载并不是所有内容都一起加载的，有的图片可能是在网页加载完成后才会再加载这些图片。

音视频文件在引入时，即audio和video在引入时，默认情况下是不允许用户在直接的控件上播放停止的，无法操作。要操作的话要加上一个

controls属性，一个没有值的属性，要播放的话就加上这个属性就好了、、还有一个

autoplay属性，在打开页面时是否自动播放。但是大多数不支持。

loop属性，循环播放。

除了使用src属性指定播放文件的路径，还可以使用下面这个方式引入路径（用法就是在浏览器在不支持某种音乐的格式的时候会用其他的格式。）

<auto controls>

<source src=”-----”>

<source src=”-----”>

</auto>

video标签也是一样的，也可以通过两种方式引入，但是也推荐使用source这种，兼容性更好。

# CSS层叠样式表：

就像一个蛋糕一样，我们只能看到最外层，我们可以为页面的每一层设置样式，当我们只能看到最顶成的样式。

网页其实是一个多层结构。css可以用来设置网页中元素的样式。

将样式编写到外部的CSS文件中，可以使用到浏览器的缓存机制，从而加快网页的加载速度，一搞用户的浏览体验。

声明块由一个一个的声明组成，声明是一个键值对结构，

一个样式名对应一个样式值，名和值之间以：连接，以； 结尾

交集选择器：

作用：选择同时符合对个条件的元素

语法：选择器1选择器2选择器3选择器n…

例子：div.myclass 意思就是选择div下的class名为myclass的元素

注意点：交集选择器中如果有元素选择器，必须使用元素选择器开头，像上面这个例子中只能是div.myclass不能是.classdiv。

并集选择器：

作用：同时选择多个选择器对应的元素

语法：选择器1,选择器2,选择器3…

例子：#b1, .a, .b, h1, div{}

伪类选择器：

看Hbuilder里面的笔记

样式的继承，可以通过文档查看摸个属性是否会继承，比如background-color属性就不会继承，而font-size属性就会继承

选择器的权重：

内联样式： 1000

id选择器 0100

类选择器和伪类选择器 0010

元素选择器 0001

通配选择器 \* 0000

继承的样式 没有优先级

比较元素优先级时，需要将所有的选择器的优先级进行相加计算，最后优先级越高，则越优显示（分组选择器是单独计算的）

--选择器的累加不会超过其最大的数量级，类选择器再高也不会超过id选择器：

--意思就是你写十个以上class类写在一个元素标签上，然后通过交集选择器去选，也不会超过id选择器的优先级。上面权重的写法就验证了这个说法，不可能跨优先级。

--如果优先级计算后相同，则此时优先使用靠下的样式。

--可以在某一个样式的后边添加 !important ，则此时该样式会获取到最高的优先级，甚至超过内联样式，

注意：在开发中这个玩意一定要慎用！

# 文档流：

文档流（normal flow）

- 网页是一个多层的结构，一层摞着一层

- 通过CSS可以分别为每一层来设置样式

- 作为用户来讲只能看到最顶上一层

- 这些层中，最底下的一层称为文档流，文档流是网页的基础

我们所创建的元素默认都是在文档流中进行排列

--要记住一点，文档流是一个位置

- 对于我们来元素主要有两个状态

在文档流中

不在文档流中（脱离文档流）

- 元素在文档流中有什么特点：

- 块元素

- 块元素会在页面中独占一行(自上向下垂直排列)

- 默认宽度是父元素的全部（会把父元素撑满）

- 默认高度是被内容撑开（子元素）

- 行内元素

- 行内元素不会独占页面的一行，只占自身的大小

- 行内元素在页面中左向右水平排列，如果一行之中不能容纳下所有的行内元素

则元素会换到第二行继续自左向右排列（书写习惯一致）

- 行内元素的默认宽度和高度都是被内容撑开

# 盒模型

盒模型、盒子模型、框模型（box model）

- CSS将页面中的所有元素都设置为了一个矩形的盒子

- 将元素设置为矩形的盒子后，对页面的布局就变成将不同的盒子摆放到不同的位置

- 每一个盒子都由一下几个部分组成：

内容区（content）

内边距（padding）：决定盒子内容与边框的距离

边框（border）

外边距（margin）:决定盒子的位子，与其他盒子的距离

display显示类型、visibility显示状态

盒子类型:border-sizing分为两种：

（内容盒子）content-box 默认值，宽度和高度用来设置内容区的大小

（边框盒子）border-box 宽度和高度用来设置整个盒子可见框的大小

width 和 height 指的是内容区 和 内边距 和 边框的总大小

# 浮动难点

浮动元素不会超过它上边的浮动的兄弟元素，最多最多就是和它一样高

下面拿这个例子对这句话做一个解释：

<div class="box1"></div>

<div class="box2"></div>

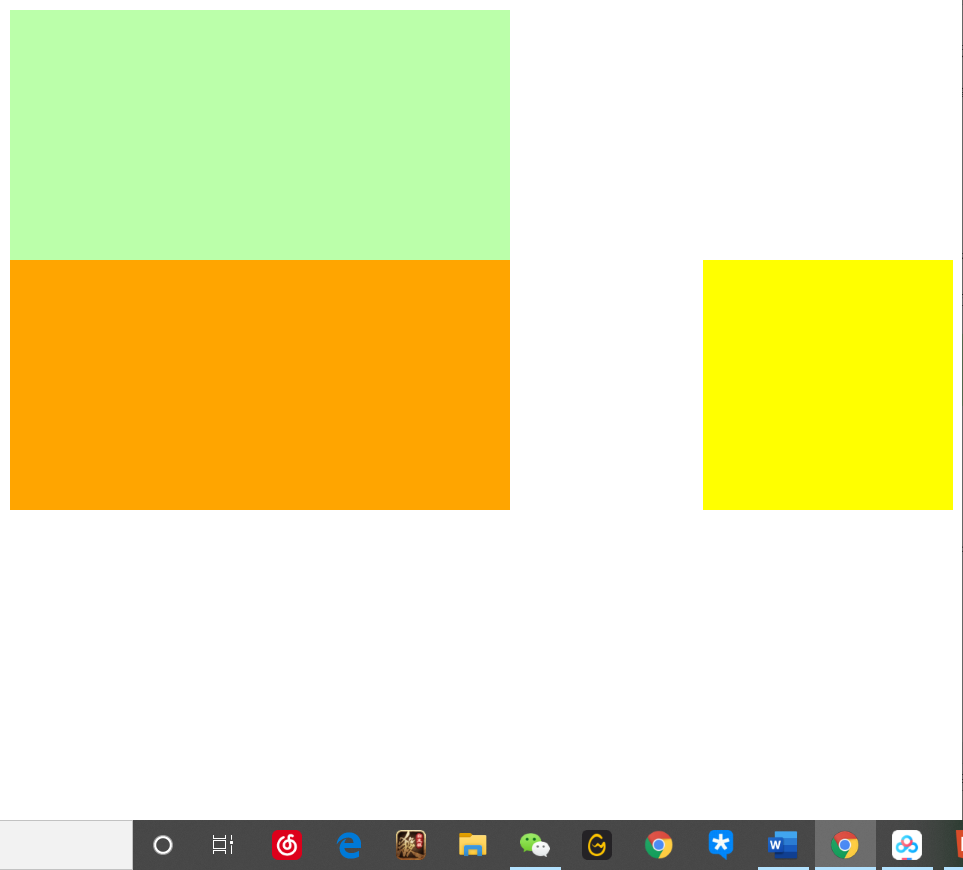
<div class="box3"></div>

这里有三个div，其中box1和box2向左浮动，box3向右浮动

页面正常大小如下图所示：



如果将页面缩小



会发现box3的位置，虽然设置的是右浮动，她也确实右浮动了但是它的位置是和box2同高，而不是与box1对齐，上面留出了一块空白，这就是为什么说一个浮动元素的高度不会超过紧接着它上面的那个浮动元素，最多是和他一样高

浮动元素不会盖住文字，文字会自动环绕在浮动元素的周围，

所以我们可以利用浮动来设置文字环绕图片的效果

# BFC(Block Formatting Context) 块级格式化环境

- BFC是一个CSS中的一个隐含的属性，可以为一个元素开启BFC

开启BFC该元素会变成一个独立的布局区域

- 元素开启BFC后的特点：

1.开启BFC的元素不会被浮动元素所覆盖

2.开启BFC的元素子元素和父元素外边距不会重叠

3.开启BFC的元素可以包含浮动的子元素

- 可以通过一些特殊方式来开启元素的BFC：

1、设置元素的浮动（不推荐）

2、将元素设置为行内块元素（不推荐）

3、将元素的overflow（溢出属性）设置为一个非visible的值

- 常用的方式 为元素设置 overflow:hidden 开启其BFC 以使其可以包含浮动元素

如果我们不希望某个元素因为其他元素浮动的影响而改变位置，

可以通过clear属性来清除浮动元素对当前元素所产生的影响

# clear

- 作用：清除浮动元素对当前元素所产生的影响

- 可选值：

left 清除左侧浮动元素对当前元素的影响

right 清除右侧浮动元素对当前元素的影响

both 清除两侧中最大影响的那侧

-原理：

设置清除浮动以后，浏览器会自动为元素添加一个上外边距，以使其位置不受其他元素的影响

# 查看04\_float---07高度塌陷的最终解决方案

在08clearfix中展示了一种解决高度塌陷和外边距重叠的集中方法，利用到了伪元素::after、::before在一个元素的头部和尾部添加空内容，并将其显示属性display由默认的inline行内元素改成table元素

代码：.clear::before,

.clear::after{

content:’’;

display:table;

clear:both;

}

# 定位（position）

- 定位是一种更加高级的布局手段

- 通过定位可以将元素摆放到页面的任意位置

- 使用position属性来设置定位

可选值：

static 默认值，元素是静止的没有开启定位

relative 开启元素的相对定位

absolute 开启元素的绝对定位

fixed 开启元素的固定定位

sticky 开启元素的粘滞定位

## - 相对定位（relative）：

- 当元素的position属性值设置为relative时则开启了元素的相对定位

- 相对定位的特点：

1.元素开启相对定位以后，如果不设置偏移量元素不会发生任何

的变化

2.相对定位是参照于元素在文档流中的位置（定位位置）进行定

位的

3.相对定位会提升元素的层级（可以覆盖其他的元素）

4.相对定位不会使元素脱离文档流

5.相对定位不会改变元素的性质块还是块，行内还是行内

## - 绝对定位（absolute）

- 当元素的position属性值设置为absolute时，则开启了元素的绝对定位

- 绝对定位的特点：

1.开启绝对定位后，如果不设置偏移量元素的位置不会发生变化

2.开启绝对定位后，元素会从文档流中脱离

3.绝对定位会改变元素的性质，行内变成块，块的宽高被内容撑开

4.绝对定位会使元素提升一个层级

5.绝对定位元素是相对于其包含块进行定位的

绝对定位的水平布局会有一点细微的变化：如下：

水平布局

left + margin-left + border-left + padding-left + width + padding-right + border-

right + margin-right + right = 包含块的内容区的宽度

- 当我们开启了绝对定位后:

水平方向的布局等式就需要添加left 和 right 两个值

此时规则和之前一样只是多添加了两个值：

当发生过度约束：

如果9个值中没有 auto 则自动调整right值以使等式

满足

如果有auto，则自动调整auto的值以使等式满足

- 可将值设置为auto的属性

margin width left right

- 因为left 和 right的值默认是auto，所以如果不指定left和

right，则等式不满足时，会自动调整这两个值

垂直方向布局的等式的也必须要满足

top + margin-top/bottom + padding-top/bottom + border-

top/bottom + height = 包含块的高度

# 包含块( containing block )

- 正常情况下：

包含块就是离当前元素最近的祖先块元素

<div> <div></div> </div>

<div><span><em>hello</em></span></div>

- 绝对定位的包含块:

包含块就是离它最近的开启了定位的祖先元素，如果所有的祖

先元素都没有开启定位（这里只要不是position:static就都算

是开启了定位）则根元素就是它的包含块

- html（根元素、初始包含块）

## - 固定定位（fixed）：

- 将元素的position属性设置为fixed则开启了元素的固

定定位

- 固定定位也是一种绝对定位，所以固定定位的大部分特

点都和绝对定位一样

唯一不同的是固定定位永远参照于浏览器的视口进行定

位固定定位的元素不会随网页的滚动条滚动

而视口和根元素又是不一样的，视口的特点是固定不动的，即使你滚动滚动条也不会消失，而根元素就会随网页的滚动而滚动（例子可以在代码中看）

# 修改元素层级（开启了potision的元素）

对于开启了定位元素，可以通过z-index属性来指定元素的层级

z-index需要一个整数作为参数，值越大元素的层级越高

元素的层级越高越优先显示

如果元素的层级一样，则优先显示靠下的元素

祖先的元素的层级再高也不会盖住后代元素

# vertical-align（垂直对齐） 设置元素垂直对齐的方式

可选值：

baseline 默认值 基线对齐

top 顶部对齐

bottom 底部对齐

middle 居中对齐

需要注意的是当一张图片包含在一个块元素中时，会发现

图片与p（设置了边框）之间存在一个缝隙，这个缝隙既不是内边距也不是外边距，只要是图片都会出现这样一个缝隙，这个距离就是图片的基线，图片属于替换元素，它的特点就和字是一样的，它的对齐方式默认也是靠着基线去对齐的，所以可以通过图片的将vertical-align属性设置为非baseline值即可。

# 为内容多的段落设置省略号可以看

C:\Users\EddiePeng\Desktop\JavaScript高级程序设计（第三版）阅读计划\HTML+CSS\尚硅谷李立超前端基础教程（四合一）\code\06\_font&background\08.文本的样式.html

# 背景相关知识

设置背景的范围

background-clip

可选值：

border-box 默认值，背景会出现在边框的下边

padding-box 背景不会出现在边框，只出现在内

容区和内边距

content-box 背景只会出现在内容区

background-origin 背景图片的偏移量计算的原点

padding-box 默认值，background-position从内边

距处开始计算

content-box 背景图片的偏移量从内容区处计算

border-box 背景图片的偏移量从边框处开始计

算

background-size 设置背景图片的大小

第一个值表示宽度

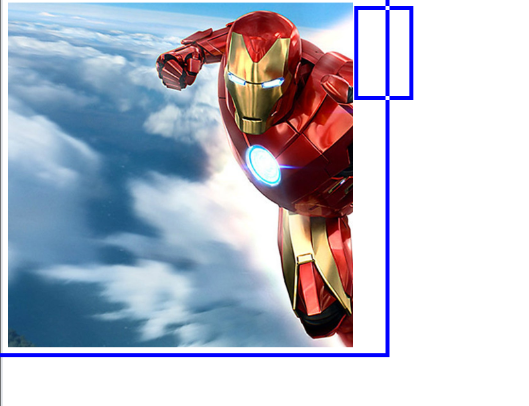
第二个值表示高度

- 如果只写一个，则第二个值默认是 auto

cover 图片的比例不变，将元素铺满

contain 图片比例不变，将图片在元素中

完整显示

cover，将盒子铺满contain，将元素完全显示，两个都不会改变图片的比例。

background-attachment

- 背景图片是否跟随元素移动

- 可选值：

scroll 默认值 背景图片会跟随元素移

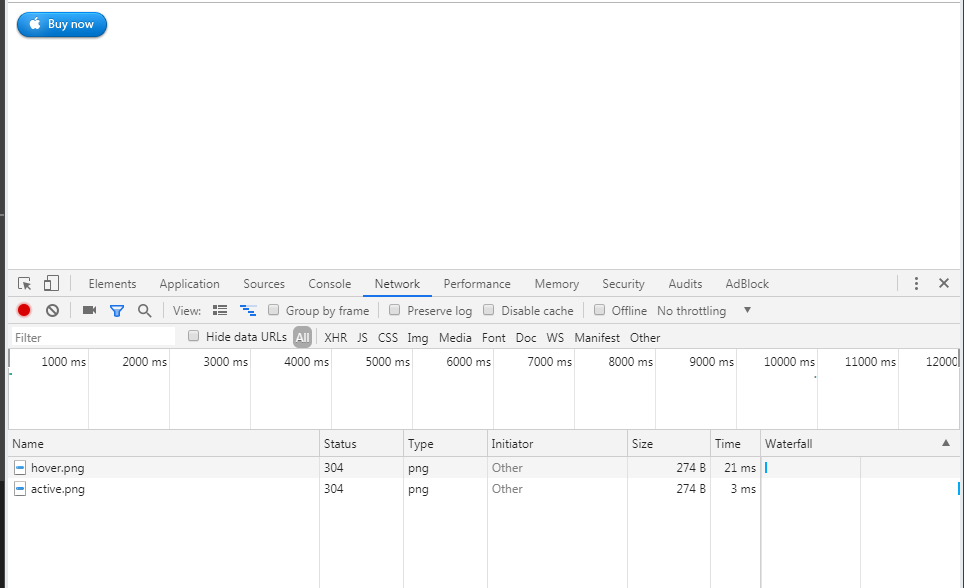
动

fixed 背景会固定在页面中，不会随元素

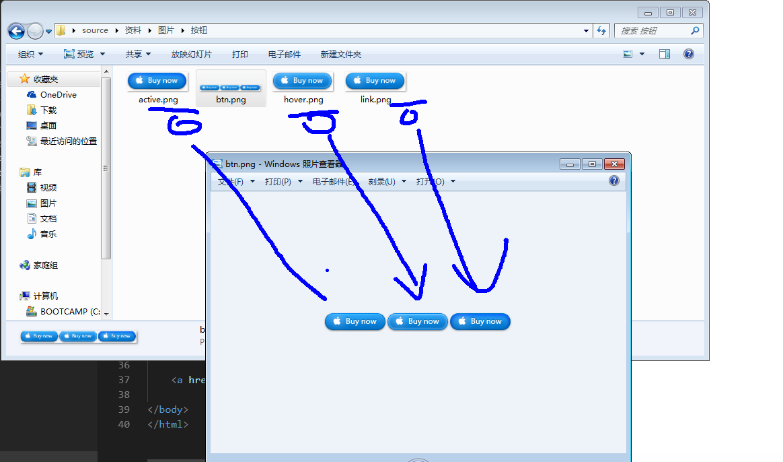
移动

# 图片背景的闪烁问题

在利用图片制作背景按钮的时候可能会出现闪烁的问题，比如图片按钮的三个状态：a:link、a:hover、a:active链接、移入、点击三个状态使用三张不同状态的图片进行展示，在浏览器中首次进行状态切换的时候会出现空白闪烁的情况，是因为：

图片属于网页中的外部资源，外部资源都需要浏览器单独发送请求加载，浏览器加载外部资源时是按需加载的，用则加载，不用则不加载像我们上边的练习link会首先加载，而hover和active会在指定状态触发时才会加载

这一点可以在首次打开网页的时候进入network点击ALL所有请求时看到，可以查看请求的资源是什么。

下面是解决方案：

# 解决图片闪烁的问题(利用雪碧图CSS-Sprite)

可以将多个小图片统一保存到一个大图片中，然后通过调整

background-position来显示的图片

这样图片会同时加载到网页中就可以有效的避免出现闪烁的

问题

这个技术在网页中应用十分广泛，被称为CSS-Sprite，这种图

我们称为雪碧图

# 雪碧图CSS-Sprite的使用步骤：

1.先确定要使用的图标

2.测量图标的大小

3.根据测量结果创建一个元素

4.将雪碧图设置为元素的背景图片

5.设置一个偏移量以显示正确的图片

雪碧图的特点：

一次性将多个图片加载进页面，降低请求的次数，加快访问速度，提升用户的体验

# 渐变

通过渐变可以设置一些复杂的背景颜色，可以实现从一个颜色向其他颜色过渡的效果

！！渐变是图片，需要通过background-image来设置

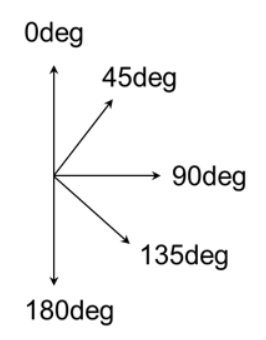
线性渐变，颜色沿着一条直线发生变化

linear-gradient() (译：线性梯度)

linear-gradient(red,yellow) 红色在开头，黄色在结尾，中

间是过渡区域

- 线性渐变的开头，我们可以指定一个渐变的方向

 to left

to right

to bottom

to top

xxdeg deg表示度数（上面这个图可以很好的解释）

（度数表示从哪个角度开始渐变比如to bottom的度数表示为0deg，to top的度数表示为180deg，top bottom right【下右方向】的度数表示为135deg）

turn 表示圈(表示转多少圈)

1渐变可以同时指定多个颜色，多个颜色默认情况下平分布，

2也可以手动指定渐变的分布情况

指明线性渐变的颜色、起点、终点。翻译成中文就是：颜色+空格+百分比或者长度值。

比如下面这种方式

background-image: linear-gradient(red 50px,yellow);

表示的是红色从50px处的位置开始一直到最后向黄色渐变，所以渐变起点为50px，终点为最底部，如果指明了的话，终点就是指明的那个值。当然渐变的区域也可以用百分比的形式来表示。0%~起点就是第一个颜色的实色，起点~终点就是渐变的区域，终点~100%就是第二个颜色的实色【说明：这里是两个颜色举例，更多颜色可以自行总结】

repeating-linear-gradient() 可以平铺的线性渐变

还有一种渐变就是径向渐变：radial-gradient()，具体使用参考资料。

# Table的相关知识

如果表格中没有使用tbody而是直接使用tr，

那么浏览器会自动创建一个tbody，并且将tr全都放到tbody中

tr不是table的子元素

# 补充知识2

如何同时改变两个元素的边框颜色，要利用到下一个兄弟选择器 +

例如一个input和一个button，在对input进行聚焦的时候，会同时改变输入款和按钮的颜色：以下是示例代码：

.search-wrapper .search-inp:focus,

.search-wrapper .search-inp:focus + button{

border-color: #FF6700;

}

# 补充知识3

设置页面导航栏，那种在页面固定不动的类似可以回到页面顶部的那种导航栏的解决方案（可以参看尚硅谷视频112节）

可以先将该元素的定位元素positon设置为fixed，将right值设置为视口宽度的50%（使用50%的原因是为了方便计算，使用其他的值也可以。），视口宽度会随着浏览器的缩放而改变，

而布局的等式如下：（这里胜省略了padding和border）

left + margin-left + width + margin-right + right = 视口的宽度

auto + 0 + 26 + 0 + 60 = 视口宽度

auto + 0 + 26 + -639px + 50% = 视口宽度

为什么不对margin-left做修改：

原因：右侧始终都是right在占用50%宽度的视口，而左侧则是由浏

览器来进行自动调节，所以无论是使left值固定还是margin-left值

固定都改变不了浏览器会让左侧所有边距的宽度加起来占视口50%的事实，而修改margin-right的话，让它为负值right始终是50%所以浏览器不会再对它进行调整，他也不会对右边的边框、内边距进行调整，他们默认为0，因为此时有比它们优先级更高的left值来进行修改用于平衡，所以就会调高left的值使等式保持一个视口的宽度。

这里的margin-right的值 = -（开启固定定位的元素的宽度 + 要向右移动的距离）

如果是要固定在左边，反过来即可。

# 过渡(transition)、动画(animate)、转换(transform)（在x、y、z轴上移动）

看视频教程或则以后看书了解。

# css3新知识：弹性容器（display:flex）

# 响应式布局（关键点：媒体查询== @media 查询规则{}）

视频中的第十节code>10\_flex>媒体查询

# 补充知识4

想让一个元素内部的元素在垂直方向居中显示，可以将该元素的display属性设置为

table-cell让他变成单元格 td，就可以使用vertial-align（垂直对齐属性）：middle居中显示