# Git 命令

Pwd 用于显示当前目录

$mkdir learngit （创建库）

$cd learngt （cd）

$ pwd

/user/Michael/leargit

Git init 可将目录变成GIT可以管理的仓库(初始化一个GIT仓库)

$ git init

Initalizzed empty Git repository in /Users/Michael/leargit/.git/

Git add 把文件添加到仓库

$ git add readme.txt

Git commit 告诉Git把文件提交到仓库

$ git commit –m”wrote a readme file”

简单解释一下git commit命令，-m后面输入的是本次提交的说明，可以输入任意内容，当然最好是有意义的，这样你就能从历史记录里方便地找到改动记录。

嫌麻烦不想输入-m "xxx"行不行？确实有办法可以这么干，但是强烈不建议你这么干，因为输入说明对自己对别人阅读都很重要。实在不想输入说明的童鞋请自行Google，我不告诉你这个参数。

git commit命令执行成功后会告诉你，1个文件被改动（我们新添加的readme.txt文件），插入了两行内容（readme.txt有两行内容）。

为什么Git添加文件需要add，commit一共两步呢？因为commit可以一次提交很多文件，所以你可以多次add不同的文件。

Git status 查询仓库当前的状态

Git diff 查看文件修改内容

Git log 显示从最近到最远的提交日志（若输出信息过于繁杂可以加上 –pretty=oneline）（用于回到过去的版本）

Git reset --hard HEAD^ /HEAD~100/234234(id)返回版本上一个/上100个版本/特定ID的版本

Cat ….(readme.txt) 查看当前版本的工作区内容

git reflog 记录每一次命令以及命令的id 便于查找返回的版本ID（用于回到未来版本）

git checkout -- file可以丢弃工作区的修改

git reset HEAD …可以回退,..文件版本，同时把暂存区的修改会退到工作区

**1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout -- file。**

**场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD file，就回到了场景1，第二步按场景1操作。**

**场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考版本回退一节，不过前提是没有推送到远程库。**

Git remote add 库名 [git@github.com:Github账户名/](mailto:git@github.com:Github账户名/)learngit.git

Git push 把当前分支master推送到远程库中

Git pull 将最新的提交从远程库中抓取下来（对应的分支应提前建立链接如，本地dev和远程库origin/dev的链接：git branch --set-upstream dev origin/dev）

Git clone [git@github.com:Github](mailto:git@github.com:Github)账户名/gitskills.git

Git checkout –b 创建并切换分支 –d删除分支

Git branch 分支名 ：查看当前分支

Git branch 列出所有分支，当前分支前面有\*号

Cd ../ 转换cd

Git merge 分支名字 合并当前分支到指定分支上去

Git merge --no-ff –m”jifeji” dev 强制仅用fast forward 并创建一个新的commit。（该做法使得分支历史能够通过git log查看。Git log --graph --pretty=oneline --abbrev-commit）

Git stash 储存进度到某个地方

Git stash pop 恢复数据的同时将stash 内容删除相当于（1：git stash apply回复；2：git stash drop删除stash内容）

如果要丢弃一个没有被合并过的分支，可以通过git branch -D <name>强行删除。

**解决冲突，conflict**

1首先，可以试图用git push origin branch-name推送自己的修改；

2如果推送失败，则因为远程分支比你的本地更新，需要先用git pull试图合并；

3如果合并有冲突，则解决冲突，并在本地提交；

4没有冲突或者解决掉冲突后，再用git push origin branch-name推送就能成功！

如果git pull提示“no tracking information”，则说明本地分支和远程分支的链接关系没有创建，用命令git branch --set-upstream branch-name origin/branch-name。

在rebase的过程中，也会出现conflict，此时git会停止rebase并让你去解决冲突；在解决完冲突后，用"git-add"命令去更新这些内容的索引(index), 然后，你无需执行 git-commit,只要执行:$ git rebase –continue。

这样git会继续应用(apply)余下的补丁。

在任何时候，你可以用--abort参数来终止rebase的行动，并且"mywork" 分支会回到rebase开始前的状态

$ git rebase –abort

git rebase 把一个分支的修改合并到当前分支

$git checkout mywork//切换到本地mywork分支

$git rebase origin//把mywork分支里的每个commit取消掉，并且将它们临时保存为本地补丁（patch）这些不定放到“.git/rebase”目录中，然后把mywork分支更新为最新的origin分支，最后吧保存的这些补丁应用到mywork分支上。

git stash 存储当前工作内容进度到某个区域，便于后期读取进度

# HTML/CSS

**编码规范：**

* 以 UTF-8 无 BOM 编码作为文件格式；
* 在HTML中文档中用 <meta charset="utf-8" /> 来指定编码；
* 为每个CSS文档显示的定义编码，在文档首行定义 @charset "utf-8";

**待办事项：**

用 TODO 标示待办事项和正在开发的条目

<!-- TODO: 图文混排 -->

<div class="g-imgtext">

<img src="1.png" alt="" />

/\* TODO: 图文混排 comm: g-imgtext \*/

.g-imgtext { sRules; }

#### 省略嵌入式资源协议头

* 省略图像、媒体文件、样式表和脚本等URL协议头部声明 ( http: , https: )。如果不是这两个声明的URL则不省略。
* 省略协议声明，使URL成相对地址，防止内容混淆问题和导致小文件重复下载（这个主要是指http和https交杂的场景中）。

注：省略协议头在 IE7-8 下会有一点小问题，外部CSS文件（ link 和 @import ）会被下载两遍，所以该条目的约定看具体项目。

#### 代码有效性

* 使用 [W3C HTML Validator](http://validator.w3.org/) 来验证你的HTML代码有效性；
* 使用 [W3C CSS Validator](http://jigsaw.w3.org/css-validator/) 来验证你的CSS代码有效性。

#### 文档类型

* 统一使用HTML5的标准文档类型：<!DOCTYPE html>；

HTML5文档类型具备前后兼容的特质，并且易记易书写

* 在文档doctype申明之前，不允许加上任何非空字符；

任何出现在doctype申明之前的字符都将使得你的HTML文档进入非标准模式

* 不允许添加 <meta> 标签强制改变文档模式。

避免出现不可控的问题

#### 省略type属性

在调用CSS和JavaScript时，可以将type属性省略不写

不允许：

<link type="text/css" rel="stylesheet" href="base.css" />

<script type="text/javascript" src="base.js"></script>

应该：

<link rel="stylesheet" href="base.css" />

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

#### 省略属性值

非必须属性值可以省略

不允许：

<input type="text" readonly="readonly" />

<input type="text" disabled="disabled" />

应该：

<input type="text" readonly />

<input type="text" disabled />

这里的 readonly 和 disabled 属性的值是非必须的，可以省略不写，我们知道HTML5表单元素新增了很多类似的属性，如: required

#### 嵌套规则

不允许非法的子元素嵌套

不允许：

<ul>

<h3>xx列表</h3>

<li>asdasdsdasd</li>

<li>asdasdsdasd</li>

</ul>

应该：

<div>

<h3>xx列表</h3>

<ul>

<li>asdasdsdasd</li>

<li>asdasdsdasd</li>

</ul>

</div>

* 不推荐inline元素包含block元素

不推荐：

<span>

<h1>这是一个块级h1元素</h1>

<p>这是一个块级p元素</p>

</span>

推荐：

<div>

<h1>这是一个块级h1元素</h1>

<p>这是一个块级p元素</p>

</div>

规则可参考：

HTML4/XHTML1.0 Strict: [嵌套规则](http://www.cs.tut.fi/~jkorpela/html/strict.html)。

HTML5: [嵌套规则](http://www.w3.org/TR/html5/)

举个例子，在HTML5中，a元素同时属于 Flow content, Phrasing content, Interactive content, Palpable content 4个分类，那些子元素是 phrasing 元素的元素可以是 a 的父元素，a 允许的子元素是以它的父元素允许的子元素为准，但不能包含 interactive 元素。

#### 多媒体替代方案

* 为img元素加上alt属性；
* 为视频内容提供音轨替代；
* 为音频内容提供字母替代等等。

不推荐：

<img src="banner.jpg" />

推荐：

<img src="banner.jpg" alt="520即将到来，爱就大声说出来" />

alt属性的内容为对该图片的简要描述，这对于盲人用户和图像损毁都非常有意义，即无障碍。对于纯粹的装饰性图片，alt属性值可以留空，如 alt=""

#### 有效操作

为表单元素label加上for属性

不允许：

<input type="radio" name="color" value="0" /><label>蓝色</label>

<input type="radio" name="color" value="1" /><label>粉色</label>

应该：

<input type="radio" id="blue" name="color" value="0" /><label for="blue">蓝色</label>

<input type="radio" id="pink" name="color" value="1" /><label for="pink">粉色</label>

for属性能让点击label标签的时候，同时focus到对应的 input 和 textarea上，增加响应区域

注释内容左右两边保留和注释符号有1个空格位，在注释内容内不允许再出现中划线“-”，某些浏览器会报错。

注释风格保持与原生HTML的语法相似：成对出现 <!-- comment --><!-- /comment -->

#### 格式

* 将每个块元素、列表元素或表格元素都放在新行；
* inline元素视情况换行，以长度不超过编辑器一屏为宜；
* 每个子元素都需要相对其父级缩进（参见[缩进约定](#indentation)）。

虽然使用p标签，也可以通过CSS去定义它的外观和标题相同，但p标签本身的并不是表示标题，而是表示文本段落

#### 模块化

* 每个模块必须有一个模块名；
* 每个模块的基本组成部分应该一致；
* 模块的子节点类名需带上模块名（防止模块间嵌套时产生不必要的覆盖）；
* 孙辈节点无需再带模块名。

代码如：

<section class="m-detail">

<header class="m-detail-hd">

<h1 class="title">模块标题</h1>

</header>

<div class="m-detail-bd">

<p class="info">一些实际内容</p>

</div>

<footer class="m-detail-ft">

<a href="#" class="more">更多</a>

</footer>

</section>

其中 .m-detail-hd, .m-detail-bd, .m-detail-ft 为可选，视具体模块情况决定是否需要抽象为这种 **头，中，尾** 的结构

#### 命名-组成元素

* 命名必须由单词、中划线①或数字组成；
* 不允许使用拼音（约定俗成的除外，如：youku, baidu），尤其是缩写的拼音、拼音与英文的混合。

#### 命名-词汇规范

* 不依据表现形式来命名；
* 可根据内容来命名；
* 可根据功能来命名。

不推荐：  
left, right, center, red, black  
推荐：  
nav, aside, news, type, search

#### 命名-缩写规范

* 保证缩写后还能较为清晰保持原单词所能表述的意思；
* 使用业界熟知的或者约定俗成的。

不推荐：  
navigation   => navi  
header   => head  
description  => des

推荐：  
navigation   => nav  
header       => hd  
description  => desc

#### 命名-前缀规范

|  |  |  |
| --- | --- | --- |
| 前缀 | 说明 | 示例 |
| g- | 全局通用样式命名，前缀g全称为global，一旦修改将影响全站样式 | g-mod |
| m- | 模块命名方式 | m-detail |
| ui- | 组件命名方式 | ui-selector |
| js- | 所有用于纯交互的命名，不涉及任何样式规则。JSer拥有全部定义权限 | js-switch |

* 选择器必须是以某个前缀开头

不推荐：

.info { sRules; }

.current { sRules; }

.news { sRules; }

*因为这样将给我们带来不可预知的管理麻烦以及沉重的历史包袱。你永远也不会知道哪些样式名已经被用掉了，如果你是一个新人，你可能会遭遇，你每定义个样式名，都有同名的样式已存在，然后你只能是换样式名或者覆盖规则。*

推荐：

.m-detail .info { sRules; }

.m-detail .current { sRules; }

.m-detail .news { sRules; }

*所有的选择器必须是以 g-, m-, ui- 等有前缀的选择符开头的，意思就是说所有的规则都必须在某个相对的作用域下才生效，尽可能减少全局污染。*

#### Css格式规范书写格式

* 选择器与大括号之间保留一个空格；
* 分号之后保留一个空格；
* 逗号之后保留一个空格；
* 所有规则需换行；
* 多组选择器之间需换行。

#### 规则与分号

每条规则结束后都必须加上分号

#### 10.0与小数

如果是0开始的小数，前面的0可以省略不写

#### 去掉uri中引用资源的引号

不要在url()里对引用资源加引号

不推荐：

body {

background-image: url("sprites.png");

}

@import url("global.css");

推荐：

body {

background-image: url(sprites.png);

}

@import url(global.css);

#### HEX颜色值写法

* 将所有的颜色值小写；
* 可以缩写的缩写至3位。

#### 属性书写顺序

* 遵循先布局后内容的顺序。

.g-box {

　　　display: block;

　　　float: left;

　　　width: 500px;

　　　height: 200px;

　　　margin: 10px;

　　　padding: 10px;

　　　border: 10px solid;

　　　background: #aaa;

　　　color: #000;

　　　font: 14px/1.5 sans-serif;

}

*这个应该好理解，比如优先布局，我们知道布局属性有 display, float, overflow 等等；内容次之，比如 color, font, text-align 之类。*

**不同的浏览器对某些CSS代码解析会存在一定的差异，因此就会导致不同浏览器下给用户展示的页面效果不一样，碰到这样的情况就需要开发人员通过针对不同的浏览器对应写出不同的CSS代码，从而达到兼容不同浏览器的目的，不会让页面因为浏览器的不同而产生有差异的显示，这种技术有个专门的名称就是CSS Hack**

#### hack规范

* 尽可能的减少对Hack的使用和依赖，如果在项目中对Hack的使用太多太复杂，对项目的维护将是一个巨大的挑战；
* 使用其它的解决方案代替Hack思路；
* 如果非Hack不可，选择稳定且常用并易于理解的。

#### .避免低效率选择器

* 避免类型选择器

不允许：

div#doc { sRules; }

li.first { sRules; }

应该：

#doc { sRules; }

.first { sRules; }

*CSS选择器是由右到左进行解析的，所以 div#doc 本身并不会比 #doc 更快*

* 避免多id选择器

不允许：

#xxx #yyy { sRules; }

应该：

#yyy { sRules; }

#### 属性缩写与分拆

* 无继承关系时，使用缩写

不推荐：

body {

　　　margin-top: 10px;

　　　margin-right: 10px;

　　　margin-bottom: 10px;

　　　margin-left: 10px;

}

推荐：

body {

　　　margin: 10px;

}

* 存在继承关系时，使用分拆方式

不推荐：

.m-detail {

　　　font: bold 12px/1.5 arial, sans-serif;

}

.m-detail .info {

　　　font: normal 14px/1.5 arial, sans-serif;

}

要避免错误的覆盖：

.m-detail .info {

　　　font: 14px sans;

}

如果你只是想改字号和字体，然后写成了上面这样，这是错误的写法，因为 font 复合属性里的其他属性将会被重置为 user agent 的默认值，比如 font-weight 就会被重置为 normal。

推荐：

.m-detail {

　　　font: bold 12px/1.5 arial, sans-serif;

}

.m-detail .info {

　　　font-weight: normal;

　　　font-size: 14px;

}

在存在继承关系的情况下，只将需要变更的属性重定义，不进行缩写，避免不需要的重写的属性被覆盖定义

* 根据规则条数选择缩写和拆分

不推荐：

.m-detail {

　　　border-width: 1px;

　　　border-style: solid;

　　　border-color: #000 #000 #f00;

}

推荐：

.m-detail {

　　　border: 1px solid #000;

　　　border-bottom-color: #f00;

}

#### 背景图

* 使用PNG格式而不是GIF格式，因为PNG格式色彩更丰富，还能提供更好的压缩比；
* 在需要兼容IE6的项目中，尽可能选择PNG8，而不是使用PNG24+滤镜。

#### .前景图

* 内容图片建议使用JPG，可以拥有更好地显示效果；
* 装饰性图片使用PNG。

Html

无序列表ul 有序列表ol 单位元li

Dl自定义列表 dt名词 dd解释内容

coords 属性与 shape 属性配合，可以规定 object 或 img 元素中链接的尺寸、形状和位置。  
  
<area> 标签的 coords 属性定义了客户端图像映射中对鼠标敏感的区域的坐标。坐标的数字及其含义取决于 shape 属性中决定的区域形状。可以将客户端图像映射中的超链接区域定义为矩形、圆形或多边形等。  
  
下面列出了每种形状的适当值：  
  
圆形：shape="circle"，coords="x,y,z"  
这里的 x 和 y 定义了圆心的位置（"0,0" 是图像左上角的坐标），r 是以像素为单位的圆形半径。  
  
多边形：shape="polygon"，coords="x1,y1,x2,y2,[x3](https://www.baidu.com/s?wd=x3&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1dBnyP9Pju9uWb3mvn1mvf40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnWfvPW0vrH0v),y3,..."  
每一对 "x,y" 坐标都定义了多边形的一个顶点（"0,0" 是图像左上角的坐标）。定义三角形至少需要三组坐标；高纬多边形则需要更多数量的顶点。  
  
多边形会自动封闭，因此在列表的结尾不需要重复第一个坐标来闭合整个区域。  
  
矩形：shape="rectangle"，coords="x1,y1,x2,y2"  
第一个坐标是矩形的一个角的顶点坐标，另一对坐标是对角的顶点坐标，"0,0" 是图像左上角的坐标。请注意，定义举行实际上是定义带有四个顶点的多边形的一种简化方法。  
  
例如，下面的 XHTML 片段在一个 100x100 像素图像的右下方四分之一处，定义了一个对鼠标敏感的区域，并在图像的正中间定义了一个圆形区域。

# Css

**特殊性**

选择器的特殊性分成4个成分等级

样式是行内样式，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等于类型选择器和为玄素选择器的数量

特殊性=a\*1000+b\*100+c\*10+d

如果遇到似乎没有起作用的CSS规则，很可能出现了特殊性冲突。可以在需要的选择器中添加它的一个父元素的ID，从而提高它的特殊性。如果能够解决问题，说明样式表中其他地方很可能有更特殊的规则，它覆盖了需要选择的规则。如果是这样，需要检查代码解决特殊性冲突，让代码尽可能简洁。

为了避免过分混乱，尽量保持一般性样式非常一般，特殊性样式尽可能特殊，这样就不需要覆盖特殊样式了。如果发现不得不多次覆盖一般样式，那么从更一般的规则中删除需要覆盖的生命，并且将它显式地应用于需要它的每个元素。

**设计代码结构**

为了便于维护，最好把样式表划分为几大块。显然，常常把最一般的规则放在最前面。这包括应用于body标记的、应该由站点上所有元素继承的样式。接下来是可能需要的所有全局reset样式，然后是链接、标题和其他元素。

完成一般样式之后，开始处理更特殊的样式和辅助样式。这些是在整个站点中使用的一般类，包括表单和错误消息等方面。然后，处理布局和导航等结构性元素。

随着在样式表中的移动，在一层样式上构建另一层样式，处理的样式越来越特殊。处理完页面结构元素之后，把注意力转到与特定页面相关的组件上。最后再文档的地不处理覆盖和例外情况。整个文档结构向下面一样

* 一般样式
  + 主题样式
  + Reset样式
  + 链接
  + 标题
  + 其他元素
* 辅助样式
  + 表单
  + 通知和错误
  + 一致的条目
* 页面结构
  + 标题、页脚和导航
  + 布局
  + 其他页面结构元素
* 页面组件
  + 各个页面
* 覆盖

使用一种风格统一的大注释块分隔每个部分

/\* @group general styles----------------\*/

/\* @group helper styles-----------------\*/

/\* @group page structure-------------\*/

/\*@group page components------\*/

/\*@group overrides-----------------\*/

许多元素将由用户代理样式表设置外边距和内边距。可以通过将元素的margin或padding设置为零来覆盖这些浏览器样式。但这种技术不区分元素，对option等元素又不利影响。因此使用reset把内边距和外边距显式地设置为零可能更安全。。

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或子元素。

相对定位，无论是否移动，元素仍占据原来的控件，移动元素会导致它覆盖其他框。是“相对于”元素在文档流中的初始位置。

绝对定位，与文档流无关，不占据控件。普通文档流中其他元素的布局就想绝对定位的元素不存在时一样。“相对于”距离它最近的已定位最暗元素，如果不存在已定位的祖先元素，那么相对于初始包含块。

Background-image属性定义要使用的所有图像。接下来指定它们是否应该重复显示最后用background-position属性设置他们的额位置。

Border-radius属性用来设置边框角的半径，实现圆角效果

Clear属性值可以是left、right、none或both，它表示框的哪边不挨着浮动框。在清理元素时，浏览器在元素顶上添加足够的外边距，使元素的顶边缘垂直下降到浮动框下面。

Box shadow属性，需要4个值：垂直、水平偏移、投影宽度（也就是模糊程度）和颜色。可以与border-radius属性相互配合，在圆角框上创建投影。

Opcity不透明度 直接设置0-1如0.8 不透明的程度

RGBa是一种同时设置颜色和不透明度的机制，RGB代表红色、绿色和蓝色，a代表透明度使用方法如下：

.alter{

Background-color:rgba(0,0,0,0.8)

Border-radius:2em;

}前三个数字表示颜色的红蓝绿值，黑色是三个都为0，最后一个数字是十进制的不透明度值，0表示完全透明。

PNG透明度

图像替换：FIR与Phark都会在关闭图片打开CSS时出现访问性问题，对于重要信息或导航信息，做好避免这两种方法的使用。 IFR 非常方便的FLASH图像替换方法，细节以及源代码可以去[www.shauninman.com/plete/2004/04/ifr-revisited-and-revised](http://www.shauninman.com/plete/2004/04/ifr-revisited-and-revised)

链接伪类选择器①link伪类选择器：用来寻找没有被访问过的链接

②visited伪类选择器：用来寻找被访问过的链接。

针对链接应用样式的选择器①:hover动态伪类选择器：用来讯在鼠标悬停处的元素，最好与:focus伪类一起用

②:active动态伪类选择器：用来寻找被激活的元素

激活发生在连接被单击时。

IE7和更低版本不支持在除连接之外的其他元素上使用伪类选择器。

选择器的次序非常重要，鼠标悬停激活样式中 顺序反过来就不起作用了

如正常a:link, a:visited a:hover, a:focus, a:active

颠倒次序a:hover, a:focus, a:active a:link, a:visited原理是，两个规则具有相同的特殊性，后定义的规则优先。因此发生覆盖。因此应用链接样式顺序为a:link, a:visited， a:hover, a:focus, a:active

可以使用包含片段标识符的链接链接到页面的特定部分。实现方法是href末尾加一个#字符，然后加上要链接的元素ID。

为了解决内容多难以找到目标元素的问题，可使用：target伪类目标元素设置样式，使目标元素能够突出显示。

为外部链接添加图标→在所有外部链接上加一个类，然后将图标作为背景图像应用，给链接设置少量的（右）内边距从而给图标留出空间，然后将图标作为背景图像应用于链接的某个角。

可以应用属性选择器让CSS判断链接是否是外部链接，比如用属性选择器寻找以文本http:开头的所有连接

a[href^=”http:”]{

background: url(exernalLink.gif) no-repeat right top;

padding-right: 10px

}为了避免这种选择器选择了指向自己站点的所有连接，需要重新设置这些链接，删除它们的外部链接图标。方法是匹配只想自己域名的链接，删除外部链接图标，重新设置右内边距。

a[href^=”http://www.yoursite.com”], a[href^=”http://yoursite.com”]{

background-image:none;

padding-right:0;

}

IE6和更低版本不支持属性选择器，可以通过在每个元素中添加类，使用JavaScript和DOM实现相似的效果。

锚<a>是行内元素，意味着只有在单击链接的内容时它们才会激活。若要实现像按钮的效果，有更大的可单击区域，可以将锚的display属性设置为block，然后修改width、height和其他属性来创建需要的样式和单击区域。

使用line-height控制按钮高度，能够使按钮中的文本垂直居中，如果设置height，就必须使用内边距将文本压低，模拟出垂直居中效果。

Pixy样式翻转，用一个图像在不同状态下的对齐方式来改变block化锚按钮的外观，但会产生轻微闪烁。解决方法①需要将翻转状态应用于链接的父元素；②在IE专用的CSS文件中包含启用背景缓存的代码。

Html{

Filter:expression(document.execCommand(“BackgroundImageCache”, false, true));

}

CSS精灵——把所有按钮、图标和各种图形集中在一个地方，提高可维护性，并且加快了下载速度。

Css3的渐变样式参数？倒影？

将标签与表单控件关联起来非常容易，可以采用两种方式之一来实现①隐式方式，即把表单元素嵌套在label元素中

<label>email<input name=”email” type=”text”/></label>

②显式方式，即把label的for属性设置为相关联的表单元素的id名称：

<label for=”email”>email</label>

<input name=”email” id=”email” type=”text”/>

在表单输入控件和标签之间创建关联需要id属性，而将表单数据发送回服务器需要name属性，id和name不必相同。

For属性 label中的

为了防止CSS布局中，两个元素向同一个方向浮动在不同浏览器中产生浮动偏移BUG，导致浮动元素下降，可以不适用水平外边距或内边距来建立隔离带，而是让一个元素向左另一个元素向右浮动。