**window.onbeforeunload绑定窗口监听事件，当窗口关闭时自动提示**

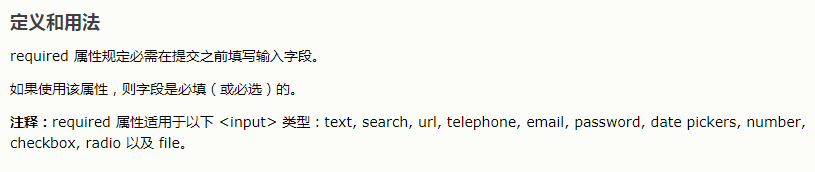
**window.parent，指的是页面的父窗口。类别window.top**

**cursor属性用于修改鼠标移上去的样式：**

**not-allowed显示禁用状态。**

**pointer显示为一只手**

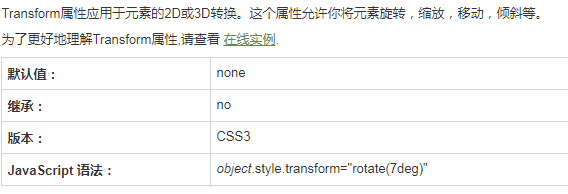
**[required](https://www.runoob.com/tags/att-input-required.html)属性**



[hidden属性](https://www.runoob.com/tags/att-global-hidden.html)

**<div id=”id” hidden=”hidden”>、<div class=”class” hidden=”hidden”>**

**[CSS3属性transform](https://www.runoob.com/cssref/css3-pr-transform.html)**



例如，单行代码实现垂直居中。

<div>

<div style=”position:relative;top:50%;transform:translateY(-50%);”></div>

</div>

隐藏元素中的display:hidden;与visibility:hidden;都达到了隐藏的效果，但

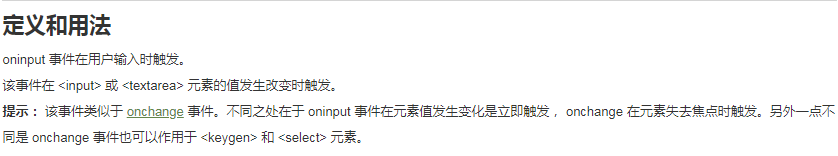
前者不占用DOM空间，后者会占用。

[html单引号、双引号](https://blog.csdn.net/fn0723/article/details/82083887)

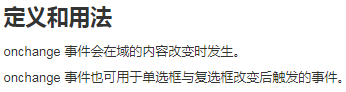


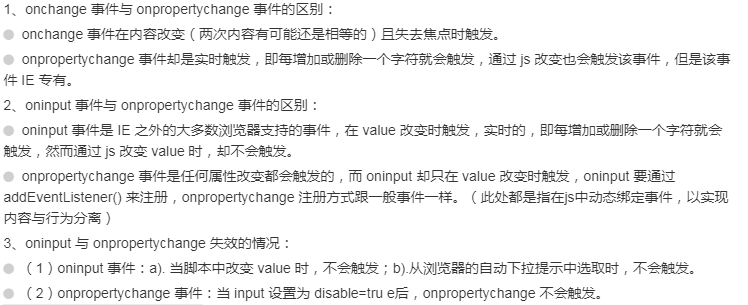


[oninput](https://www.runoob.com/jsref/event-oninput.html)



[onchange](https://www.runoob.com/jsref/event-onchange.html)





[normal关键字](http://www.w3school.com.cn/cssref/pr_font_weight.asp)

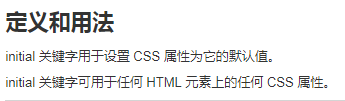
用于设置font-weight

font-weight: normal;



[initial关键字](http://www.runoob.com/cssref/css-initial.html)

用于设置它的默认值



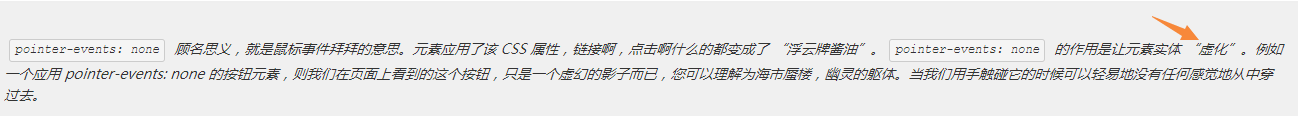
transform-orgin

允许用户改变元素的x和y轴。3D转换元素，还可以更改元素的z轴

[www.runoob.com/cssref/css3-pr-transform-origin.html](http://www.runoob.com/cssref/css3-pr-transform-origin.html)

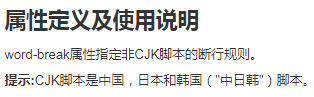
css的**[pointer-events:none](http://www.zhangxinxu.com/wordpress/2011/12/css3-pointer-events-none-javascript/)属性，可以让一个元素进行虚化。**

**使用场景之一，让超链接失效。**



<https://www.cnblogs.com/zichi/p/9068481.html>

断行规则(换行)word-break属性



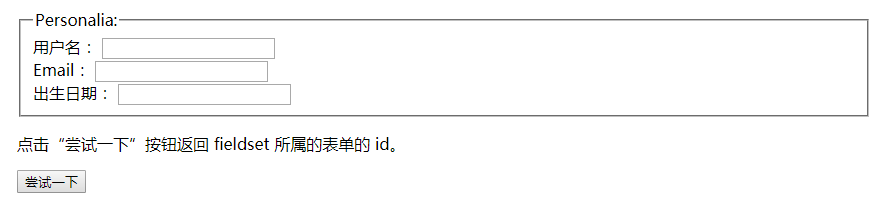


<http://www.runoob.com/cssref/css3-pr-word-break.html>

[vertical-align属性](http://www.runoob.com/cssref/pr-pos-vertical-align.html)

该属性定义【行内元素】的基线相对于该元素所在行的基线的垂直对齐

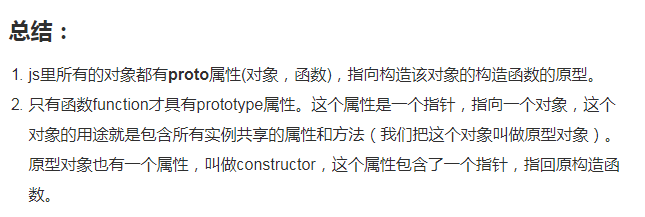
[fieldset form属性](http://www.runoob.com/jsref/dom-obj-fieldset.html)



[onclick、click、on执行优先级](https://blog.csdn.net/xsf1840/article/details/77268350)

onclick > click > on

[\_\_proto\_\_和prototype](https://www.jianshu.com/p/80bcf8b2004e)



[splice](http://www.runoob.com/jsref/jsref-splice.html)

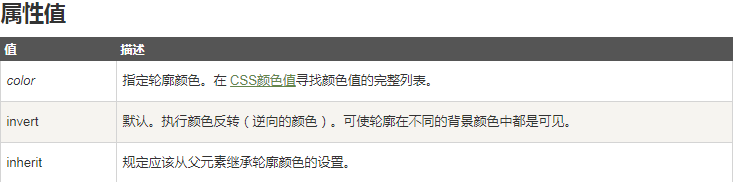




[outline](http://www.runoob.com/cssref/pr-outline.html)

在元素周围绘制一条线，位于边框边缘的外围。

**不属于元素尺寸的一部分，因此元素的宽高不包括轮廓的宽度。**



[CSS选择器](http://www.runoob.com/cssref/css-selectors.html)

|  |  |  |  |
| --- | --- | --- | --- |
| [.](https://www.runoob.com/cssref/sel-class.html)*[class](https://www.runoob.com/cssref/sel-class.html)* | .intro | 选择所有class="intro"的元素 | 1 |
| [#](https://www.runoob.com/cssref/sel-id.html)*[id](https://www.runoob.com/cssref/sel-id.html)* | #firstname | 选择所有id="firstname"的元素 | 1 |
| [\*](https://www.runoob.com/cssref/sel-all.html) | \* | 选择所有元素 | 2 |
| *[element](https://www.runoob.com/cssref/sel-element.html)* | p | 选择所有<p>元素 | 1 |
| *[element,element](https://www.runoob.com/cssref/sel-element-comma.html)* | div,p | 选择所有<div>元素和<p>元素 | 1 |
| *[element](https://www.runoob.com/cssref/sel-element-element.html)**[element](https://www.runoob.com/cssref/sel-element-element.html)* | div p | 选择<div>元素内的所有<p>元素 | 1 |
| *[element](https://www.runoob.com/cssref/sel-element-gt.html)*[>](https://www.runoob.com/cssref/sel-element-gt.html)*[element](https://www.runoob.com/cssref/sel-element-gt.html)* | div>p | 选择所有父级是 <div> 元素的 <p> 元素 | 2 |
| *[element](https://www.runoob.com/cssref/sel-element-pluss.html)*[+](https://www.runoob.com/cssref/sel-element-pluss.html)*[element](https://www.runoob.com/cssref/sel-element-pluss.html)* | div+p | 选择所有紧接着<div>元素之后的<p>元素 | 2 |
| [[](https://www.runoob.com/cssref/sel-attribute.html)*[attribute](https://www.runoob.com/cssref/sel-attribute.html)*[]](https://www.runoob.com/cssref/sel-attribute.html) | [target] | 选择所有带有target属性元素 | 2 |
| [[](https://www.runoob.com/cssref/sel-attribute-value.html)*[attribute](https://www.runoob.com/cssref/sel-attribute-value.html)*[=](https://www.runoob.com/cssref/sel-attribute-value.html)*[value](https://www.runoob.com/cssref/sel-attribute-value.html)*[]](https://www.runoob.com/cssref/sel-attribute-value.html) | [target=-blank] | 选择所有使用target="-blank"的元素 | 2 |
| [[](https://www.runoob.com/cssref/sel-attribute-value-contains.html)*[attribute](https://www.runoob.com/cssref/sel-attribute-value-contains.html)*[~=](https://www.runoob.com/cssref/sel-attribute-value-contains.html)*[value](https://www.runoob.com/cssref/sel-attribute-value-contains.html)*[]](https://www.runoob.com/cssref/sel-attribute-value-contains.html) | [title~=flower] | 选择标题属性包含单词"flower"的所有元素 | 2 |
| [[](https://www.runoob.com/cssref/sel-attribute-value-lang.html)*[attribute](https://www.runoob.com/cssref/sel-attribute-value-lang.html)*[|=](https://www.runoob.com/cssref/sel-attribute-value-lang.html)*[language](https://www.runoob.com/cssref/sel-attribute-value-lang.html)*[]](https://www.runoob.com/cssref/sel-attribute-value-lang.html) | [lang|=en] | 选择 lang 属性以 en 为开头的所有元素 | 2 |
| [:link](https://www.runoob.com/cssref/sel-link.html) | a:link | 选择所有未访问链接 | 1 |
| [:visited](https://www.runoob.com/cssref/sel-visited.html) | a:visited | 选择所有访问过的链接 | 1 |
| [:active](https://www.runoob.com/cssref/sel-active.html) | a:active | 选择活动链接 | 1 |
| [:hover](https://www.runoob.com/cssref/sel-hover.html) | a:hover | 选择鼠标在链接上面时 | 1 |
| [:focus](https://www.runoob.com/cssref/sel-focus.html) | input:focus | 选择具有焦点的输入元素 | 2 |
| [:first-letter](https://www.runoob.com/cssref/sel-firstletter.html) | p:first-letter | 选择每一个<P>元素的第一个字母 | 1 |
| [:first-line](https://www.runoob.com/cssref/sel-firstline.html) | p:first-line | 选择每一个<P>元素的第一行 | 1 |
| [:first-child](https://www.runoob.com/cssref/sel-firstchild.html) | p:first-child | 指定只有当<p>元素是其父级的第一个子级的样式。 | 2 |
| [:before](https://www.runoob.com/cssref/sel-before.html) | p:before | 在每个<p>元素之前插入内容 | 2 |
| [:after](https://www.runoob.com/cssref/sel-after.html) | p:after | 在每个<p>元素之后插入内容 | 2 |
| [:lang(](https://www.runoob.com/cssref/sel-lang.html)*[language](https://www.runoob.com/cssref/sel-lang.html)*[)](https://www.runoob.com/cssref/sel-lang.html) | p:lang(it) | 选择一个lang属性的起始值="it"的所有<p>元素 | 2 |
| *[element1](https://www.runoob.com/cssref/sel-gen-sibling.html)*[~](https://www.runoob.com/cssref/sel-gen-sibling.html)*[element2](https://www.runoob.com/cssref/sel-gen-sibling.html)* | p~ul | 选择p元素之后的每一个ul元素 | 3 |
| [[](https://www.runoob.com/cssref/sel-attr-begin.html)*[attribute](https://www.runoob.com/cssref/sel-attr-begin.html)*[^=](https://www.runoob.com/cssref/sel-attr-begin.html)*[value](https://www.runoob.com/cssref/sel-attr-begin.html)*[]](https://www.runoob.com/cssref/sel-attr-begin.html) | a[src^="https"] | 选择每一个src属性的值以"https"开头的元素 | 3 |
| [[](https://www.runoob.com/cssref/sel-attr-end.html)*[attribute](https://www.runoob.com/cssref/sel-attr-end.html)*[$=](https://www.runoob.com/cssref/sel-attr-end.html)*[value](https://www.runoob.com/cssref/sel-attr-end.html)*[]](https://www.runoob.com/cssref/sel-attr-end.html) | a[src$=".pdf"] | 选择每一个src属性的值以".pdf"结尾的元素 | 3 |
| [[](https://www.runoob.com/cssref/sel-attr-contain.html)*[attribute](https://www.runoob.com/cssref/sel-attr-contain.html)*[\*=](https://www.runoob.com/cssref/sel-attr-contain.html)*[value](https://www.runoob.com/cssref/sel-attr-contain.html)*[]](https://www.runoob.com/cssref/sel-attr-contain.html) | a[src\*="runoob"] | 选择每一个src属性的值包含子字符串"runoob"的元素 | 3 |
| [:first-of-type](https://www.runoob.com/cssref/sel-first-of-type.html) | p:first-of-type | 选择每个p元素是其父级的第一个p元素 | 3 |
| [:last-of-type](https://www.runoob.com/cssref/sel-last-of-type.html) | p:last-of-type | 选择每个p元素是其父级的最后一个p元素 | 3 |
| [:only-of-type](https://www.runoob.com/cssref/sel-only-of-type.html) | p:only-of-type | 选择每个p元素是其父级的唯一p元素 | 3 |
| [:only-child](https://www.runoob.com/cssref/sel-only-child.html) | p:only-child | 选择每个p元素是其父级的唯一子元素 | 3 |
| [:nth-child(](https://www.runoob.com/cssref/sel-nth-child.html)*[n](https://www.runoob.com/cssref/sel-nth-child.html)*[)](https://www.runoob.com/cssref/sel-nth-child.html) | p:nth-child(2) | 选择每个p元素是其父级的第二个子元素 | 3 |
| [:nth-last-child(](https://www.runoob.com/cssref/sel-nth-last-child.html)*[n](https://www.runoob.com/cssref/sel-nth-last-child.html)*[)](https://www.runoob.com/cssref/sel-nth-last-child.html) | p:nth-last-child(2) | 选择每个p元素的是其父级的第二个子元素，从最后一个子项计数 | 3 |
| [:nth-of-type(](https://www.runoob.com/cssref/sel-nth-of-type.html)*[n](https://www.runoob.com/cssref/sel-nth-of-type.html)*[)](https://www.runoob.com/cssref/sel-nth-of-type.html) | p:nth-of-type(2) | 选择每个p元素是其父级的第二个p元素 | 3 |
| [:nth-last-of-type(](https://www.runoob.com/cssref/sel-nth-last-of-type.html)*[n](https://www.runoob.com/cssref/sel-nth-last-of-type.html)*[)](https://www.runoob.com/cssref/sel-nth-last-of-type.html) | p:nth-last-of-type(2) | 选择每个p元素的是其父级的第二个p元素，从最后一个子项计数 | 3 |
| [:last-child](https://www.runoob.com/cssref/sel-last-child.html) | p:last-child | 选择每个p元素是其父级的最后一个子级。 | 3 |
| [:root](https://www.runoob.com/cssref/sel-root.html) | :root | 选择文档的根元素 | 3 |
| [:empty](https://www.runoob.com/cssref/sel-empty.html) | p:empty | 选择每个没有任何子级的p元素（包括文本节点） | 3 |
| [:target](https://www.runoob.com/cssref/sel-target.html) | #news:target | 选择当前活动的#news元素（包含该锚名称的点击的URL） | 3 |
| [:enabled](https://www.runoob.com/cssref/sel-enabled.html) | input:enabled | 选择每一个已启用的输入元素 | 3 |
| [:disabled](https://www.runoob.com/cssref/sel-disabled.html) | input:disabled | 选择每一个禁用的输入元素 | 3 |
| [:checked](https://www.runoob.com/cssref/sel-checked.html) | input:checked | 选择每个选中的输入元素 | 3 |
| [:not(](https://www.runoob.com/cssref/sel-not.html)*[selector](https://www.runoob.com/cssref/sel-not.html)*[)](https://www.runoob.com/cssref/sel-not.html) | :not(p) | 选择每个并非p元素的元素 | 3 |
| [::selection](https://www.runoob.com/cssref/sel-selection.html) | ::selection | 匹配元素中被用户选中或处于高亮状态的部分 | 3 |
| [:out-of-range](https://www.runoob.com/cssref/sel-out-of-range.html) | :out-of-range | 匹配值在指定区间之外的input元素 | 3 |
| [:in-range](https://www.runoob.com/cssref/sel-in-range.html) | :in-range | 匹配值在指定区间之内的input元素 | 3 |
| [:read-write](https://www.runoob.com/cssref/sel-read-write.html) | :read-write | 用于匹配可读及可写的元素 | 3 |
| [:read-only](https://www.runoob.com/cssref/sel-read-only.html) | :read-only | 用于匹配设置 "readonly"（只读） 属性的元素 | 3 |
| [:optional](https://www.runoob.com/cssref/sel-optional.html) | :optional | 用于匹配可选的输入元素 | 3 |
| [:required](https://www.runoob.com/cssref/sel-required.html) | :required | 用于匹配设置了 "required" 属性的元素 | 3 |
| [:valid](https://www.runoob.com/cssref/sel-valid.html) | :valid | 用于匹配输入值为合法的元素 | 3 |
| [:invalid](https://www.runoob.com/cssref/sel-invalid.html) | :invalid | 用于匹配输入值为非法的元素 | 3 |

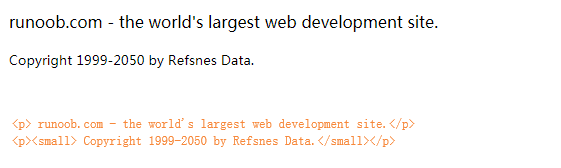
[!DOCTYPE html](http://www.runoob.com/tags/tag-doctype.html)



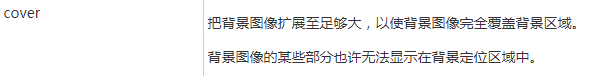
[后缀htm&html区别](https://www.cnblogs.com/fzuhyj/p/7154704.html)

htm 是来源于老的 8.3 文件格式，DOS 操作系统**只能支持长度为三位的后缀名**，所以是 htm,但在 windows 下无所谓 HTM 与 HTML，html 是为长文件名的格式命名的。所以 **htm 是为了兼容过去的DOS命名格式存在的**，在效果上没有区别的。以前 htm 和 html 作为不同的服务器上的超文本文件，但现在通用。

[small标签(小型文本和旁注)](http://www.runoob.com/tags/tag-small.html)



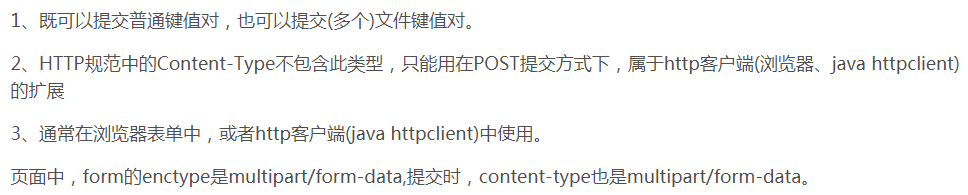
background-size: cover;



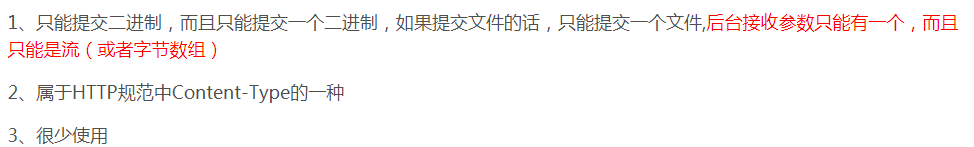
<https://www.w3cschool.cn/cssref/pr-background-size.html>

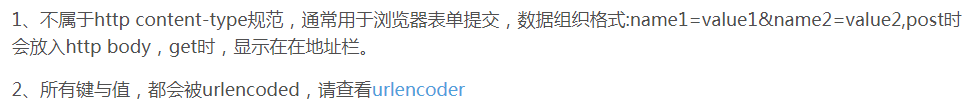
multipart/form-data、application/octet-stream、application/x-www-form-urlencoded的区别

multipart/form-data ：

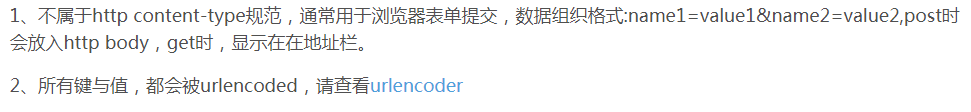


application/octet-stream ：





application/x-www-form-urlencoded：



<https://blog.csdn.net/wangjun5159/article/details/49644507>

html中绑定多个事件，如onblur="evnet1(),event2(),..."

function event1() {}

function event2() {}

[CSS中的@规则](https://blog.csdn.net/zcy_wxy/article/details/80652247)

at-rule是一个声明，为CSS提供执行或怎么表现的指令。每个声明以@开头,后紧跟一个可用的关键字，这个关键字充当一个标识符，用于表示CSS该做什么。这是一个通用的语法，尽管每个at-rule有其它语法变体。

urce      源  
dst = destination 目的

File file = FiltUtil.copyFile(src, dst);

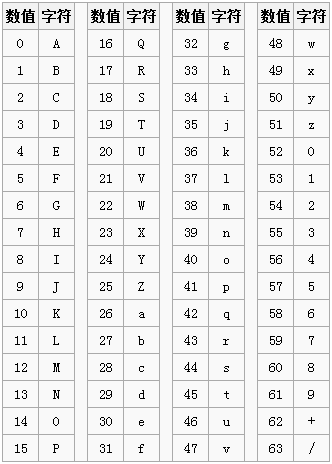
base64

一种用64个字符来表示任意二进制数据的方法。

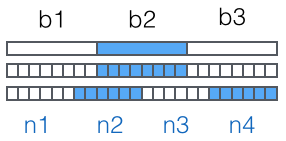
为什么会有Base64编码呢？

因为有些网络传送渠道**并不支持所有**的字节，例如传统的邮件只支持可见字符的传送，像ASCII码的控制字符就不能通过邮件传送。这样用途就受到了很大的限制，比如图片二进制流的每个字节不可能全部是可见字符，所以就传送不了。最好的方法就是在不改变传统协议的情 况下，做一种扩展方案来支持二进制文件的传送。**把不可打印的字符也能用可打印字符来表示，问题就解决了**。Base64编码应运而生，Base64就是一种基于64个可打印字符来表示二进制数据的表示方法。

**Base64索引表**



Base64的编码只有6个bit表示，正常的字符是8个bit表示，6和8的最小公倍数24，那么就是4个base64字符可以表示3个标准ascii字符。



例子：

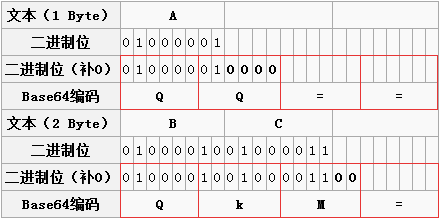
1、把abc三个字符转换为base64

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字符串 | a | b | c |  |
| ASCII | 97 | 98 | 99 |  |
| 8bit | 01100001 | 01100010 | 01100011 |  |
| 6bit | 011000 | 010110 | 001001 | 100011 |
| 十进制 | 24 | 22 | 9 | 35 |
| 对应base64编码 | Y | W | J | j |

2、把man三个字符转换为base64

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字符串 | m | a | n |  |
| ASCII | 109 | 97 | 110 |  |
| 8bit | 01101101 | 01100001 | 01101110 |  |
| 6bit | 011011 | 010110 | 000101 | 101110 |
| 十进制 | 27 | 22 | 5 | 46 |
| 对应base64编码 | b | W | F | u |

3、当转换到最后的字符不足3个字符咋办，如果不足三个字符的话，我们直接在最后添加“＝”号即可（左边开始）



但是由于 “=” 字符会在URL和Cookie中引起歧义，所以很多Base64编码会把后边的“=”字符去掉。那么如果去掉了岂不是破坏了完整性，到时怎么解码？因为Base64编码的长度永远都是4的倍数，所以保证Base64编码的长度是4的倍数就可以正常解码了（加“=”字符）。

套路总结：

字符转base64，找ascii对照表 - 换二进制 - 位数切换 - 找base64对照表

base64转字符，找base64对照表 - 换二进制 - 位数切换 - 找ascii对照表

其中

转base64时，二进制位数不够用‘0’来凑

转字符时，base64码长度不够用‘=’来凑（4的倍数）

[Base64原理解析](https://www.cnblogs.com/diligenceday/p/6002382.html)

[base64](https://www.liaoxuefeng.com/wiki/001374738125095c955c1e6d8bb493182103fac9270762a000/001399413803339f4bbda5c01fc479cbea98b1387390748000)

data类型URL，其实它也是一个**URL**

使用Base64作为背景图片

page {

overflow:hidden;

position:fixed;

/\*

background-image:url('http://p0d5ombx1.bkt.clouddn.com/login.png');

\*/

background: url(**data:image/jpeg;base64**,/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAkGBwgHBgkICAgKCgkLDhcPDg0NDhwUFREXIh4jIyEeICAlKjUtJScyKCAgLj8vMjc5PDw8.........................0P/aAAgBAwEBPwAWA//Z);

margin: 0px;

/\*下面的是关键的设置100%显示\*/

background-size: auto 100%;

}

**data: ---- 获取数据类型名称**

**image/gif; ----- 指数据类型名称(还有image/png，image/jpeg等等)  
base64 ----- 指编码模式**

**其它数据类型名称**

data:,<文本数据>

data:text/plain,<文本数据>

data:text/html,<HTML代码>

data:text/html;base64,<base64编码的HTML代码>

data:text/css,<CSS代码>

data:text/css;base64,<base64编码的CSS代码>

data:text/javascript,<Javascript代码>

data:text/javascript;base64,<base64编码的Javascript代码>

data:image/gif;base64,base64编码的gif图片数据

data:image/png;base64,base64编码的png图片数据

data:image/jpeg;base64,base64编码的jpeg图片数据

data:image/x-icon;base64,base64编码的icon图片数据

**使用base64编码做图片好处是可以节省http请求,加快小图像的加载时间，坏处是经过编码后文件体积比源文件大30%左右。**

**[图片Base64编码的利与弊分析](https://www.imooc.com/article/27804)**

# [data类型的Url的格式](https://www.cnblogs.com/augustine/p/3669644.html)

页面中引入CSS

// 行内样式

1、直接在标签中写入

// 内部样式

2、<style type=”text/css”></style>

// 外部样式，链接式

3、<style type=”text/css” rel=”stylesheet” href=”text.css” />

// 外部样式，导入式

4、<style type=”text/css”>

@import url(“text.css”);

@import url(“text2.css”);

</style>

优先级：行内样式 > 内部样式 > 外部样式

# [CSS三：CSS的三种引入方式](https://www.cnblogs.com/dotnet261010/p/7198892.html)