CSS类面试题目总结

## 用 CSS 实现元素垂直居中，有哪些好的方案？

链接：https://www.zhihu.com/question/20543196/answer/57757836

（1）不知道自己高度和父容器高度的情况下, 利用绝对定位只需要以下三行：

parentElement{

position:relative;

}

childElement{

position: absolute;

top: 50%;

transform: translateY(-50%);

}

（2）若父容器下只有一个元素，且父元素设置了高度，则只需要使用相对定位即可

parentElement{

height:xxx;

}

.childElement {

position: relative;

top: 50%;

transform: translateY(-50%);

}

（3）Flex 布局：

不考虑兼容老式浏览器的话，用Flex布局简单直观一劳永逸：

parentElement{

display:flex;/\*Flex布局\*/

display: -webkit-flex; /\* Safari \*/

align-items:center;/\*指定垂直居中\*/

}

## 水平垂直居中

## 块级元素、行内元素、内联块级元素

块级元素：结构性标记。**display:block;**

<div>、 <p>、<h1>、<form>、<ul> 、<li>。

特点：

1. 总是从新的一行开始
2. 高度、宽度、行高、顶和底边距都可设置
3. 宽度没有设置时，默认为100%
4. 块级元素中可以包含块级元素和行内元素

行内元素（内联元素）：描述性标记。**display:inline;**

<span>、<a>、<label>、 <strong> 和<em>

特点：

1. 和其他元素都在一行上；
2. 元素的高度、宽度及顶部和底部边距不可设置；
3. 元素的宽度就是它包含的文字或图片的宽度，不可改变。
4. 行内元素只能行内元素，包含块级元素语法不会报错，但是不提倡

内联块级元素：**display:inline-block;**

<img>、<input>

1. 和其他元素都在一行上
2. 元素的宽、高、行高、顶部和底部边距可设置

## CSS盒子模型

在一个文档中，每个元素都被表示为一个矩形的盒子。

盒子模型分为：W3C标准盒子模型、IE怪异盒子模型。模型描述了元素所占空间的内容。

所有HTML元素都可以看做盒子，包括：外边距，边框，填充，实际内容



注意！！！如果未设置box-sizing:border-box；设置的width和height是设置的内容区域的宽度和高度。

**W3C标准盒模型**：

元素的宽=content

盒子的宽：margin-left + border-left + padding-left + width + padding-right + border-right + margin-right

**IE怪异盒子模型：**

元素的宽=content + padding + border

盒子的宽= margin-left + width + margin-right

**解决方案：box-sizing**

// W3C 标准盒模型（浏览器默认）

box-sizing: content-box;

// IE 怪异盒模型

box-sizing: border-box;

当设置box-sizing:border-box;时，border和padding就被包含在了宽高之内，和IE盒模型是一样的。

为了避免CSS在不同浏览器上表现不同，写法最好为

\*, \*:before, \*:after {

-moz-box-sizing: border-box;

-webkit-box-sizing: border-box;

box-sizing: border-box;

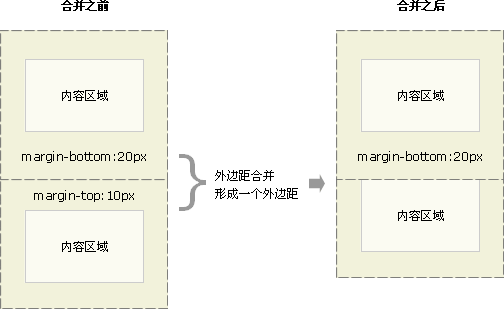
}

## 外边距合并（垂直）

外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。

合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。

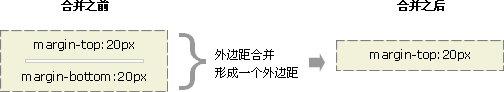
1. 一个元素出现在另一个元素上面，第一个元素的下边距与第二个元素的上边距合并。



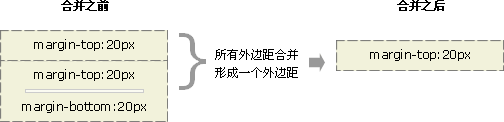
1. 一个元素包含在另一个元素中（外层元素没有设置内边距或边框把外边距分隔开）



1. 外边距与自身合并。一个空元素，有外边距，没有边框或填充，其上下外边距会合并



这个外边距遇到另一个元素的外边距，还会继续合并：



注意：只有普通文档流中块框的垂直外边距才会发生外边距合并。行内框、浮动框或绝对定位之间的外边距不会合并。

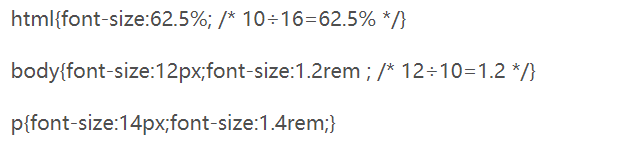
## CSS常用单位：px、em、rem

* 1. px：固定大小。一个像素等于电脑屏幕上的一个点（屏幕分辨率的最小分割）
  2. em：相对长度单位。相对于当前对象内文本的字体尺寸，如果当前对象内文本的字体尺寸未设置，则相对于浏览器的默认字体尺寸。
     1. em的值不是固定的
     2. em会继承父级元素的字体大小

浏览器的默认字体高是16px。所有未经调整的浏览器都符合：1em=16px。0.625em=10px。为了简化font-size的换算，在css中的body选择器中声明Font-size=62.5%，即将默认字体大小设置为16px\*62.5=10px。1em=10px。这样只需要将之前设置的px数值除以10，换上em作为单位即可。

缺点：由于em继承父元素的字体大小，当父元素字体改变时，需要重新计算。

* 1. rem：css3新增的相对长度单位，只相对根目录即HTML元素。只要在html标签上设置字体大小为标准，文档中的字体大小都会以此为参照



兼容性：IE9+，Firefox、Chrome、Safari、Opera 的主流版本都支持，为了兼容不支持 rem 的浏览器，我们需要在 rem 前面写上对应的 px 值，这样不支持的浏览器可以优雅降级。

* 1. %：类似于“em”。当使用百分比单位,你的文字在移动设备上仍然保持完全的可伸缩性和可访问性。1 em = 16 px = 100%

在一般的PC端网页制作过程中，px和em用的比较多，在移动端一般采用rem。

## 隐藏标签的方式

1. display：none；
2. visibility：hidden
3. opacity：0
4. 设置height，width等盒模型属性为0
5. 设置position，top，left等属性，将元素移出可视区域
6. clip-path

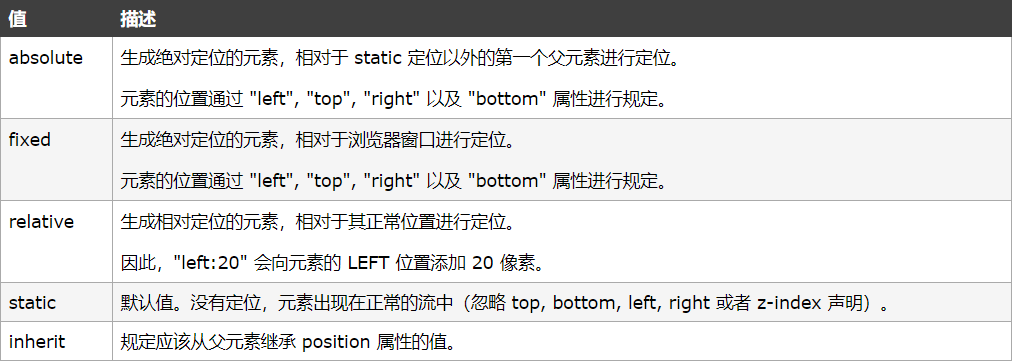
元素隐藏后的事件响应：

1. display:none：元素彻底消失，很显然不会触发其点击事件
2. visibility:hidden：无法触发其点击事件，有一种说法是display:none是元素看不见摸不着，而visibility:hidden是看不见摸得着，这种说法是不准确的，设置元素的visibility后无法触发点击事件，说明这种方法元素也是消失了，只是依然占据着页面空间。
3. opacity:0：可以触发点击事件，原因也很简单，设置元素透明度为0后，元素只是相对于人眼不存在而已，对浏览器来说，它还是存在的，所以可以触发点击事件
4. height:0：将元素的高度设置为0，并且设置overflow:hidden。使用这种方法来隐藏元素，是否可以触发事件要根据具体的情况来分析。如果元素设置了border，padding等属性不为0，很显然，页面上还是能看到这个元素的，触发元素的点击事件完全没有问题。如果全部属性都设置为0，很显然，这个元素相当于消失了，即无法触发点击事件。

通过JS可以触发被设置为display:none的元素的事件。无法触发点击事件的真正原因是鼠标无法真正接触到被设置成隐藏的元素！！！

## position属性

任何元素都可以定位，绝对或固定定位元素会生成一个块级框，无论元素是什么类型。相对定位元素会相对于他在正常流中的默认位置偏移。



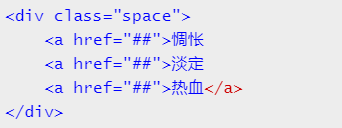
sticky：粘性定位。结合了relative和fixed定位。设置阈值（top | bottom | left | right），当未超过阈值时relative，到达阈值后fixed。

Margin-top和top的区别：

margin-top指的是自己的顶部距离容器其他元素的距离，top指的是自己的顶部距离容器顶部的距离，有点绝对定位的意思。所以margin-top通常与relative定位结合使用，top和absolute定位结合使用。

## 去除inline-block元素间间距的N种方法

1. 元素留白间距出现的原因：标签段之间的空格。
2. 去掉HTML中的空格，即连续书写标签，可读性差
3. margin负值
4. 不闭合标签。为了向下兼容IE6/IE7等喝蒙牛长大的浏览器，最后一个列表的标签的结束（闭合）标签不能丢。



1. 使用font-size:0。基本上可以解决大部分浏览器下inline-block元素之间的间距(IE7等浏览器有时候会有1像素的间距)。
2. 使用letter-spacing或word-spacing

## 轮播图实现原理

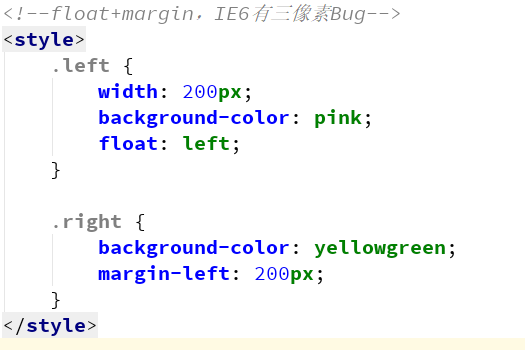
设置一个固定的父容器作为展示部分，overflow：hidden，一个子容器存放所有的轮播图片，通过设置子容器的position属性实现不同图片在父容器中的展示。例如左右切换的轮播，向左就设置子容器.left-=图片宽度，向右就设置子容器.left+=图片宽度。为了避免轮播的空白页面的出现，在第一张图片前增添最后一张图片，在最后一张图片后添加第一张图片。当到达边界值时，直接将left值设置为0或最大值；

## Bootstrap的使用，特性等

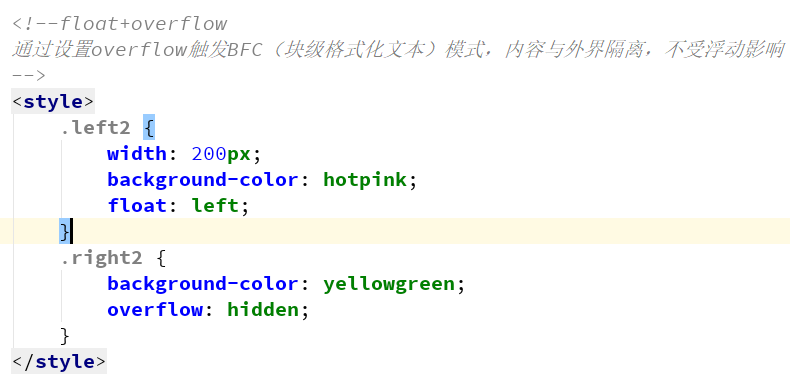
1. **less或sass，对CSS预编译器有没有了解**

## 两列布局，左定宽，右边宽度自适应

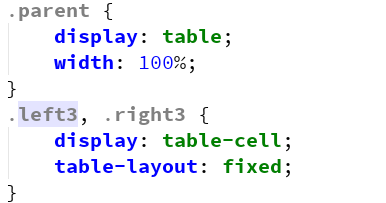
1. Float+margin：

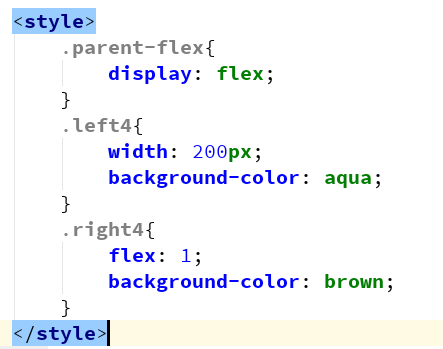


1. Float+overflow：hidden



1. Table



1. Flex
   * 1. 

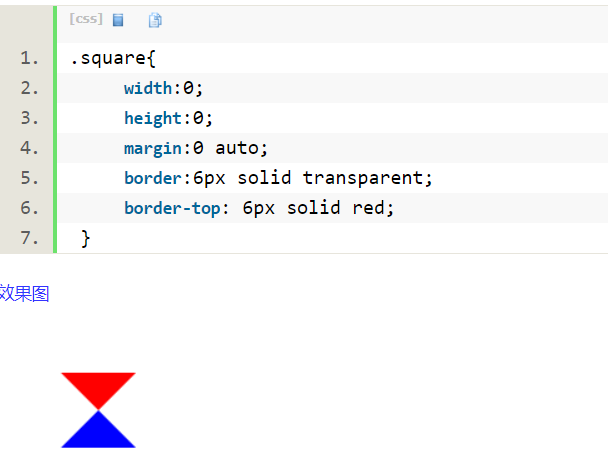
## 三角形实现方式

1. 使用border

第一步 保证元素是块级元素

第二步 设置元素的边框

第三步 不需要显示的边框使用透明色



1. 利用”◆“字符实现三角形：设置font-size和top/bottom/left/right值
2. 利用 CSS3 transfrom 旋转 45 度实现三角形：先进行top/bottom/left/right位移，再旋转

## 瀑布流布局

## CSS hack

针对不同的浏览器/不同版本写相应的CSS code的过程，叫做CSS hack!

### CSS hack的原理

由于不同的浏览器和浏览器各版本对CSS的支持及解析结果不一样，以及CSS优先级对浏览器展现效果的影响，我们可以据此针对不同的浏览器情景来应用不同的CSS。

### CSS hack分类

CSS Hack大致有3种表现形式，CSS属性前缀法、选择器前缀法以及IE条件注释法（即HTML头部引用if IE）Hack，实际项目中CSS Hack大部分是针对IE浏览器不同版本之间的表现差异而引入的。

* 属性前缀法(即类内部Hack)：例如 IE6能识别下划线"\_"和星号" \* "，IE7能识别星号" \* "，但不能识别下划线"\_"，IE6~IE10都认识"\9"，但firefox前述三个都不能认识。
* 选择器前缀法(即选择器Hack)：例如 IE6能识别\*html .class{}，IE7能识别\*+html .class{}或者\*:first-child+html .class{}。
* IE条件注释法(即HTML条件注释Hack)： 针对所有IE(注：IE10+已经不再支持条件注释)： <!--[if IE]>IE浏览器显示的内容 <![endif]-->，针对IE6及以下版本： <!--[if lt IE 6]>只在IE6-显示的内容 <![endif]-->。这类Hack不仅对CSS生效，对写在判断语句里面的所有代码都会生效。

### CSS hack利弊

一般情况下，我们尽量**避免使用**CSS hack，但是有些情况为了顾及用户体验实现向下兼容，不得已才使用hack。比如由于IE8及以下版本不支持CSS3,而我们的项目页面使用了大量 CSS3新属性在IE9/Firefox/Chrome下正常渲染，这种情况下如果不使用css3pie或htc或条件注释等方法时,可能就得让IE8- 的专属hack出马了。使用hack虽然对页面表现的一致性有好处，但过多的滥用会造成html文档混乱不堪，增加管理和维护的负担。

## @import和link之间的区别

**两者都是外部引用CSS的方式，但是存在一定的区别**：

* 老祖宗的差别。link是XHTML标签，除了加载CSS外，还可以定义RSS等其他事务；@import属于CSS范畴，只能加载CSS。
* 加载顺序的差别。link引用CSS时，在页面载入时同时加载；@import需要页面完全载入以后加载。
* 兼容性的差别。link是XHTML标签，无兼容问题；@import是在CSS2.1提出的，低版本的浏览器不支持。
* 使用dom控制样式时的差别。link支持使用Javascript控制DOM去改变样式；而@import不支持。

<**style**>  
 **@import url**(**"style.css"**);  
</**style**>

<**link rel="stylesheet" href="style.css"**>

## web页面重构怎么操作？

页面重构就是根据原有页面内容和结构的基础上，通过div+css写出符合web标准的页面结构。

具体实现要达到以下三点：

* 结构完整，可通过标准验证
* 标签语义化，结构合理
* 充分考虑到页面在站点中的“作用和重要性”，并对其进行有针对性的优化

## 清除浮动

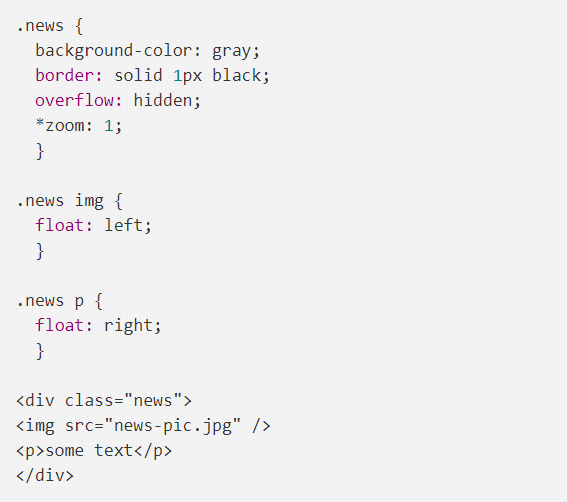
### ****1、使用空标签清除浮动****

空标签可以是div标签，也可以是P标签。也有很多人用<hr>，只是需要另外 为其清除边框，但理论上可以是任何标签。对于使用额外标签清除浮动（闭合浮动元素），是W3C推荐的 做法。

这种方式是在需要清除浮动的父级元素内部的所有浮动元素后添加这样一个标签清除浮动，并为其定义CSS代 码：clear:both。此方法的弊端在于增加了无意义的结构元素。

### 2、使用overflow属性。

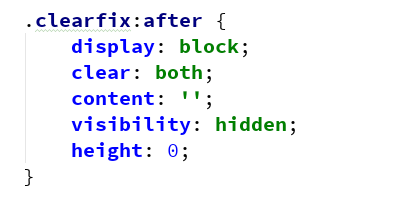
此方法有效地解决了通过空标签元素清除浮动而不得不增加无意代码的弊端：给浮动元素的容器添加overflow:auto或overflow：hidden。在IE6中需要触发hasLayout，为容器添加\*zoom:1； 本质上是使父容器形成BFC，BFC需要对浮动元素计算高度



### ****3、使用after伪对象清除浮动。****

 该方法只适用于非IE浏览器 。使用中需注意以下几点。一、该方法中必须为需要清除浮动元素的伪对象中设置height:0，否则该元素会比实际高出若干像 素；二、content属性是必须的，但其值可以为空。

在包裹浮动元素的父容器上设置：



清除浮动的方法分为两类：

* 1. clear属性，在浮动元素的末尾添加一个带有clear：both属性的空div
  2. 触发浮动元素父元素的BFC，使该父容器可以包含浮动元素

## 在网页中的应该使用奇数还是偶数的字体？

* 偶数字号相对更容易和 web 设计的其他部分构成比例关系。比如：当我用了 14 px 的正文字号，我可能会在一些地方用 14 × 0.5 = 7 px 的 margin，在另一些地方用 14 × 1.5 = 21 px 的标题字号。
* Windows 自带的点阵宋体（中易宋体）从 Vista 开始只提供 12、14、16 px 这三个大小的点阵，而 13、15、17 px 时用的是小一号的点阵（即每个字占的空间大了 1 px，但点阵没变），于是略显稀疏。

## box-sizing常用的属性有哪些？分别有什么作用？

(1) box-sizing: content-box|border-box|inherit;

　　(2) content-box:**宽高不包括边框和内边距，只包含内容区域。**宽度和高度分别应用到元素的内容框。在宽度和高度之外绘制元素的内边距和边框(元素默认效果)。

border-box:**宽高包括边框和内边距**。元素指定的任何内边距和边框都将在已设定的宽度和高度内进行绘制。通过从已设定的宽度和高度分别减去边框和内边距才能得到内容的宽度和高度。

## CSS优先级

4个等级的定义如下：

第一等级：代表内联样式，如style=""，权值为 1000

第二等级：代表id选择器，如#content，权值为100

第三等级：代表类，伪类和属性选择器，如.content，权值为10

第四等级：代表标签选择器和伪元素选择器，如div p，权值为1

注意：通用选择器（\*），子选择器（>），和相邻同胞选择器（+）并不在这个等级中，所以他们的权值为0

**内联》ID选择器》伪类=属性选择器=类选择器》元素选择器【p】》通用选择器(\*)》继承的样式**

## 块级格式化上下文BFC

1、BFC概念：决定了元素如何对其内容进行定位，以及与其他元素的关系和相互作用。表现原则是无论内部怎么样，都不会影响外部，反之亦然。

1. 如何触发BFC
   1. float不为none
   2. Position不是relative和static
   3. display：inline-block、table-cell、table-caption、flex、inline-flex
   4. overflow：hidden、auto、scroll。不是visible
2. BFC布局规则：
   1. BFC内部的box在垂直方向一个接一个的放置
   2. box垂直方向的距离由margin决定，可能会margin重合
   3. 盒子的左外边缘会触碰到容器的左外边缘
   4. BFC的区域不会与float box重叠
   5. BFC是一个隔离的独立容器，容器中的不影响外面，反之亦然
   6. 计算BFC高度时，浮动元素也参与计算

## CSS雪碧图

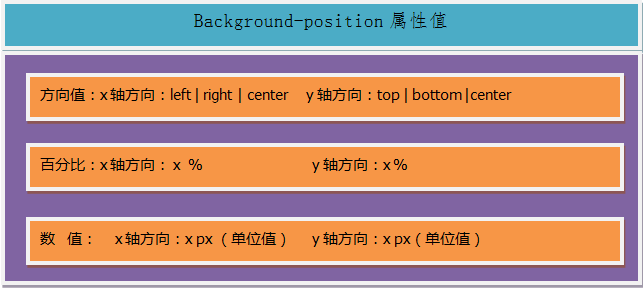
CSS雪碧图即CSS Spirit，又叫做CSS精灵，是一种CSS图像合并技术，将小图标和背景图合并到一张图上，然后利用CSS的背景定位来显示需要显示的图片部分。

使用background-position属性实现

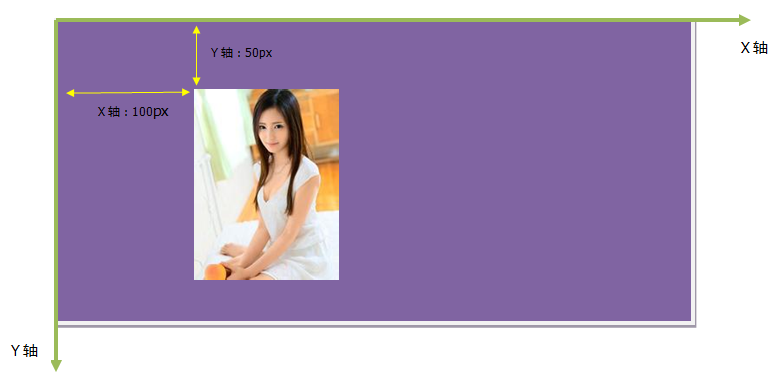
#### background-position

设置背景图像的起始位置，如果背景图要重复，从此位置开始。

有两个属性值： background-position：x | y，用法上可以对其一个属性单独使用 background-position-x 和 background-position-y。



* 1. 属性值为像素



* 1. 属性值为%

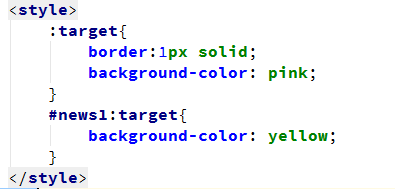
（容器自身的宽度/高度 - 图片自身的宽度/高度） x 百分比

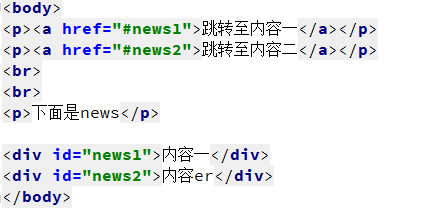


* 1. 不设置background-position：默认是0% 0%
  2. 方向值和百分比的计算方式是一样的，它们可以相互转换，left：0%，right：100%，center：50%。
  3. 如果background-position属性值只设置一个，那么另一个默认为center。

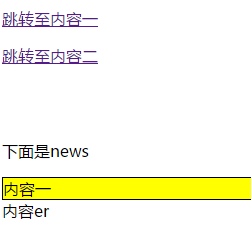
## CSS实现标签页动态切换效果

<a>标签的href属性锚点+：target伪类





点击“跳转至内容一”后：



## 文字水平垂直居中text-align、line-height、vertical-align的用法

参考：<http://www.jb51.net/css/29328.html>

text-align

line-height

base-line

## transition、transform、animation

1. transition：平滑的改变CSS的值。无论是点击事件，焦点事件，还是鼠标hover，只要值改变了，就是平滑的，就是动画。转换过程

transiton属性是下面几个属性的缩写：

* + 1. transition-property：指定过渡的属性值，比如transition-property:opacity就是只指定opacity属性参与这个过渡。
    2. transition-duration：指定这个过渡的持续时间
    3. transition-delay：延迟过渡时间
    4. transition-timing-function：指定过渡动画缓动类型，有ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier()

其中，linear线性过度，ease-in由慢到快，ease-out由快到慢，ease-in-out由慢到快在到慢。

1. transform：拉伸（skew），压缩（scale），旋转（rotate），偏移（translate）。

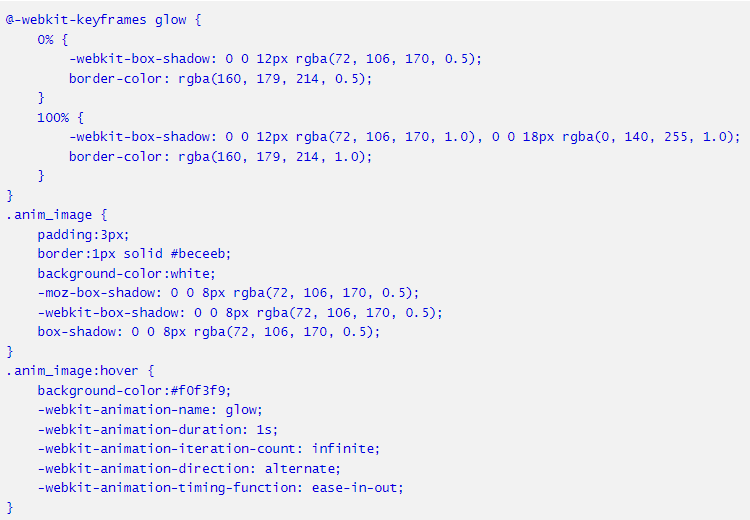
transform通常用来表示达到的效果

1. animation

与keyframes结合使用

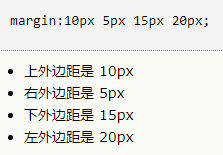
animation: name duration timing-function delay iteration-count direction;

|  |  |
| --- | --- |
| **值** | **描述** |
| [*animation-name*](http://www.w3school.com.cn/cssref/pr_animation-name.asp) | 规定需要绑定到选择器的 keyframe 名称。。 |
| [*animation-duration*](http://www.w3school.com.cn/cssref/pr_animation-duration.asp) | 规定完成动画所花费的时间，以秒或毫秒计。 |
| [*animation-timing-function*](http://www.w3school.com.cn/cssref/pr_animation-timing-function.asp) | 规定动画的速度曲线。 |
| [*animation-delay*](http://www.w3school.com.cn/cssref/pr_animation-delay.asp) | 规定在动画开始之前的延迟。 |
| [*animation-iteration-count*](http://www.w3school.com.cn/cssref/pr_animation-iteration-count.asp) | 规定动画应该播放的次数。 |
| [*animation-direction*](http://www.w3school.com.cn/cssref/pr_animation-direction.asp) | 规定是否应该轮流反向播放动画。 |



## margin、padding

margin：元素外边距。简写属性。顺时针方向设置。只有父容器有border或者padding时，margin-top属性才会正常，否则是只能是相对于body的margin-top的效果。



|  |  |
| --- | --- |
| **值** | **描述** |
| auto | 浏览器计算外边距。 |
| *length* | 规定以具体单位计的外边距值，比如像素、厘米等。默认值是 0px。 |
| *%* | 规定基于父元素的宽度的百分比的外边距。 |
| inherit | 规定应该从父元素继承外边距。 |

padding：内填充。取值与margin相同。但是，margin允许负值，padding不允许