[盘点8种CSS实现垂直居中水平居中的绝对定位居中技术](http://blog.csdn.net/freshlover/article/details/11579669)



**Ⅰ.绝对定位居中(Absolute Centering)技术**

我们经常用margin:0 auto来实现水平居中，而一直认为margin:auto不能实现垂直居中……实际上，实现垂直居中仅需要声明元素高度和下面的CSS:

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center {
2. **margin**: auto;
3. **position**: absolute;
4. top: 0; left: 0; bottom: 0; right: 0;
5. }

我不是这种实现方法的第一人，可能这只是非常常见的一种小技术，我斗胆将其命名为**绝对居中(Absolute Centering)**，虽然如此，但是大多数讨论垂直居中的文章却从来不提这种方法，直到我最近浏览《[How to Center Anything WithCSS](http://designshack.net/articles/css/how-to-center-anything-with-css" \t "_blank)》这篇文章的评论时候才发现这种用法。在评论列表中Simon和Priit都提及了此方法。

如果你有任何扩展的功能或建议，可以在此跟帖：

[CodePen](http://codepen.io/shshaw/details/gEiDt)

[SmashingMagazine](http://coding.smashingmagazine.com/2013/08/09/absolute-horizontal-vertical-centering-css/)

[Twitter @shshaw](https://twitter.com/shshaw)

**优点：**

1.支持跨浏览器，包括IE8-IE10.

2.无需其他特殊标记，CSS代码量少

3.支持百分比%属性值和min-/max-属性

4.只用这一个类可实现任何内容块居中

5.不论是否设置padding都可居中（在不使用box-sizing属性的前提下）

6.内容块可以被重绘。

7.完美支持图片居中。

**缺点：**

1.必须声明高度（查看可变高度Variable Height）。

2.建议设置overflow:auto来防止内容越界溢出。（查看溢出Overflow）。

3.在Windows Phone设备上不起作用。

**浏览器兼容性：**

**Chrome,Firefox, Safari, Mobile Safari, IE8-10.**

绝对定位方法在最新版的Chrome,Firefox, Safari, Mobile Safari, IE8-10.上均**[测试](http://lib.csdn.net/base/softwaretest" \o "软件测试知识库" \t "_blank)**通过。

**对比表格：**

绝对居中法并不是唯一的实现方法，实现垂直居中还有些其他的方法，并各有各的优势。采用哪种技术取决于你的浏览器是否支持和你使用的语言标记。这个对照表有助于你根据自己的需求做出正确的选择。

| **Technique** | **Browser Support** | **Responsive** | **Overflow** | **resize:both** | **Variable Height** | **Major Caveats** |
| --- | --- | --- | --- | --- | --- | --- |
| [**Absolute Centering**](http://s.codepen.io/shshaw/fulldetails/gEiDt#Overview) | Modern & IE8+ | Yes | Scroll, can overflow [**Container**](http://lib.csdn.net/base/docker) | Yes | Yes\* | [Variable Height](http://s.codepen.io/shshaw/fulldetails/gEiDt#Height) not perfect cross-browser |
| [**Negative Margins**](http://s.codepen.io/shshaw/fulldetails/gEiDt#Negative-Margins) | All | No | Scroll | Resizes but doesn't stay centered | No | Not responsive, margins must be calculated manually |
| [**Transforms**](http://s.codepen.io/shshaw/fulldetails/gEiDt#Transforms) | Modern & IE9+ | Yes | Scroll, can overflow container | Yes | Yes | Blurry rendering |
| [**Table-Cell**](http://s.codepen.io/shshaw/fulldetails/gEiDt#Table-Cell) | Modern & IE8+ | Yes | Expands container | No | Yes | Extra markup |
| [**Inline-Block**](http://s.codepen.io/shshaw/fulldetails/gEiDt#Inline-Block) | Modern, IE8+ & IE7\* | Yes | Expands container | No | Yes | Requires container, hacky styles |
| [**Flexbox**](http://s.codepen.io/shshaw/fulldetails/gEiDt#Flexbox) | Modern & IE10+ | Yes | Scroll, can overflow container | Yes | Yes | Requires container, vendor prefixes |

**解释**：

通过以上描述，绝对居中（AbsoluteCentering）的工作机理可以阐述如下：

1、在普通内容流（[normal content flow](http://taligarsiel.com/Projects/howbrowserswork1.htm" \l "Layout" \t "_blank)）中，margin:auto的效果等同于margin-top:0;margin-bottom:0。

[W3C](http://www.w3.org/TR/CSS2/visudet.html#normal-block)中写道If 'margin-top', or'margin-bottom' are 'auto', their used value is 0.

2、position:absolute使绝对定位块跳出了内容流，内容流中的其余部分渲染时绝对定位部分不进行渲染。

[Developer.mozilla.org](https://developer.mozilla.org/en-US/docs/Web/CSS/position#Absolute_positioning):...an element that is positioned absolutely is taken out of the flow and thustakes up no space

3、为块区域设置top: 0; left: 0; bottom: 0; right: 0;将给浏览器重新分配一个边界框，此时该块block将填充其父元素的所有可用空间，父元素一般为body或者声明为position:relative;的容器。

[Developer.mozilla.org](https://developer.mozilla.org/en-US/docs/Web/CSS/position#Notes):For absolutely positioned elements, the top, right, bottom, and left propertiesspecify offsets from the edge of the element's containing block (what theelement is positioned relative to).

4、  给内容块设置一个高度height或宽度width，能够防止内容块占据所有的可用空间，促使浏览器根据新的边界框重新计算margin:auto

[Developer.mozilla.org](https://developer.mozilla.org/en-US/docs/Web/CSS/position#Notes): The margin of the[absolutely positioned] element is then positioned inside these offsets.

5、由于内容块被绝对定位，脱离了正常的内容流，浏览器会给margin-top,margin-bottom相同的值，使元素块在先前定义的边界内居中。

[W3.org](http://www.w3.org/TR/CSS2/visudet.html#abs-non-replaced-height): If none of the three [top, bottom,height] are 'auto': If both 'margin-top' and 'margin-bottom' are 'auto', solvethe equation under the extra constraint that the two margins get equal values.AKA: center the block vertically

**这么看来， margin:auto似乎生来就是为绝对居中(Absolute Centering)设计的，所以绝对居中(Absolute Centering)应该都兼容符合标准的现代浏览器。**

**简而言之（TL;DR）：**绝对定位元素不在普通内容流中渲染，因此margin:auto可以使内容在通过top: 0; left: 0; bottom: 0;right: 0;设置的边界内垂直居中。

**居中方式：**

**一、容器内（Within Container）**

内容块的父容器设置为position:relative，使用上述绝对居中方式，可以使内容居中显示于父容器。

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Center-Container {
2. **position**: relative;
3. }
5. .Absolute-Center {
6. **width**: 50%;
7. **height**: 50%;
8. **overflow**: auto;
9. **margin**: auto;
10. **position**: absolute;
11. top: 0; left: 0; bottom: 0; right: 0;
12. }



以下其余的demo默认上面的CSS样式已引用包括进去，在此基础上只提供额外的类供用户追加以实现不同的功能。

**二、视区内（Within Viewport）**

想让内容块一直停留在可视区域内？将内容块设置为position:fixed;并设置一个较大的z-index层叠属性值。

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center.is-Fixed {
2. **position**: fixed;
3. **z-index**: 999;
4. }



注意：对MobileSafari，若内容块不是放在设置为position:relative;的父容器中，内容块将垂直居中于整个文档，而不是可视区域内垂直居中。

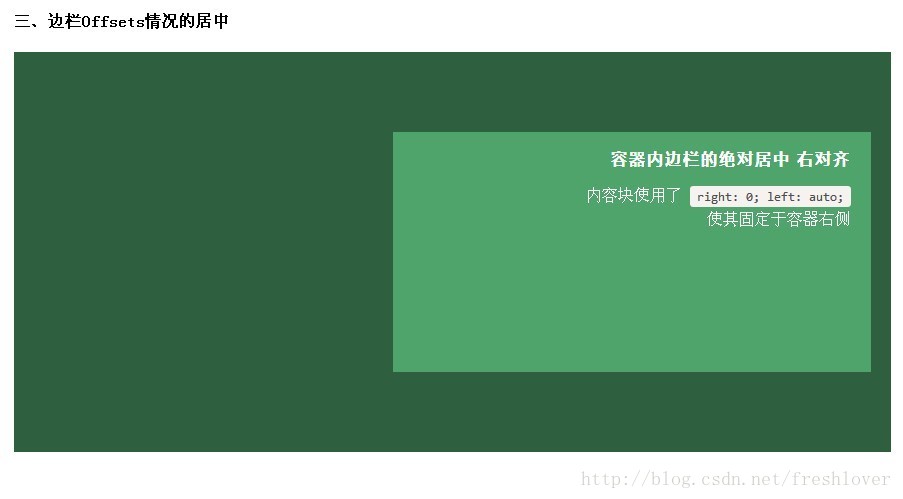
**三、边栏 (Offsets)**

如果你要设置一个固顶的头或增加其他的边栏，只需要在内容块的样式中加入像这样的CSS样式代码：top:70px;bottom:auto;由于已经声明了margin:auto;，该内容块将会垂直居中于你通过top,left,bottom和right属性定义的边界框内。

你可以将内容块固定与屏幕的左侧或右侧，并且保持内容块垂直居中。使用right:0;left:auto;固定于屏幕右侧，使用left:0;right:auto;固定与屏幕左侧。

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center.is-Right {
2. left: auto; right: 20px;
3. **text-align**: right;
4. }
6. .Absolute-Center.is-Left {
7. right: auto; left: 20px;
8. **text-align**: left;
9. }



**四、响应式/自适应(Responsive)**

绝对居中最大的优势应该就是对百分比形式的宽高支持的非常完美。甚至min-width/max-width 和min-height/max-height这些属性在自适应盒子内的表现也和预期很一致。



**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center.is-Responsive {
2. **width**: 60%;
3. **height**: 60%;
4. **min-width**: 200px;
5. **max-width**: 400px;
6. **padding**: 40px;
7. }

给内容块元素加上padding也不影响内容块元素的绝对居中实现。

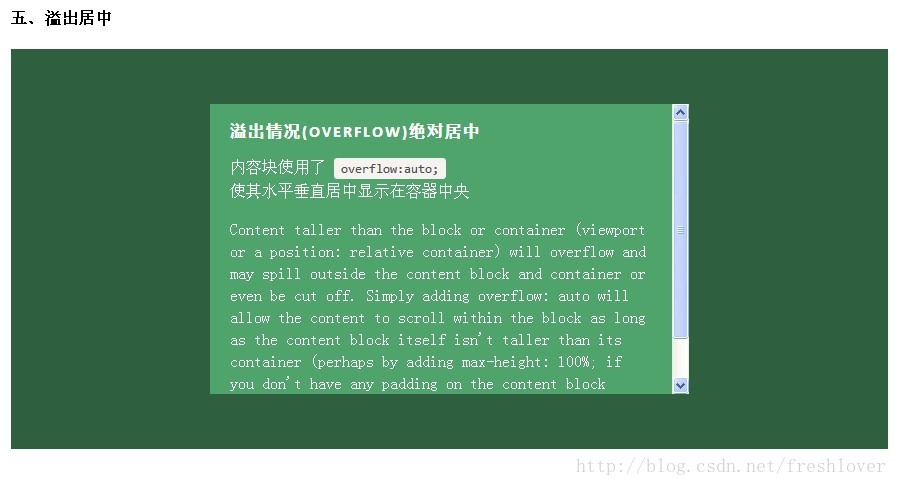
**五、 溢出情况(Overflow)**

内容高度大于块元素或容器（视区viewport或设为position:relative的父容器）会溢出，这时内容可能会显示到块与容器的外面，或者被截断出现显示不全（分别对应内容块overflow属性设置为visible和hidden的表现）。

加上overflow: auto会在内容高度超过容器高度的情况下给内容块显示滚动条而不越界。

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center.is-Overflow {
2. **overflow**: auto;
3. }



如果内容块自身不设置任何padding的话，可以设置max-height: 100%;来保证内容高度不超越容器高度。

**六、重绘(Resizing)**

你可以使用其他class类或**[JavaScript](http://lib.csdn.net/base/javascript" \o "JavaScript知识库" \t "_blank)**代码来重绘内容块同时保证居中，无须手动重新计算中心尺寸。当然，你也可以添加resize属性来让用户拖拽实现内容块的重绘。

绝对居中（Absolute Centering）可以保证内容块始终居中，无论内容块是否重绘。可以通过设置min-/max-来根据自己需要限制内容块的大小，并防止内容溢出窗口/容器。

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center.is-Resizable {
2. **min-width**: 20%;
3. **max-width**: 80%;
4. **min-height**: 20%;
5. **max-height**: 80%;
6. resize: both;
7. **overflow**: auto;
8. }



如果不使用resize:both属性，可以使用**[CSS3](http://lib.csdn.net/base/css3" \o "CSS3知识库" \t "_blank)**动画属性transition来实现重绘的窗口之间平滑的过渡。一定要设置overflow:auto;以防重绘的内容块尺寸小于内容的实际尺寸这种情况出现。

**绝对居中（AbsoluteCentering）是唯一支持resize:both属性实现垂直居中的技术。**

**注意：**

1. 要设置max-width/max-height属性来弥补内容块padding，否则可能溢出。
2. 手机浏览器和IE8-IE10浏览器不支持resize属性，所以如果对你来说，这部分用户体验很必要，务必保证对resizing你的用户有可行的退路。
3. 联合使用resize 和 transition属性会在用户重绘时，产生一个transition动画延迟时间。

**七、图片(Images)**

绝对居中**（AbsoluteCentering）**也适用于图片。对图片自身应用class类或CSS样式，并给图片添加height:auto样式，图片会自适应居中显示，如果外层容器可以resize则随着容器的重绘，图片也相应重绘，始终保持居中。

需要注意的是height:auto虽然对图片居中有用，但如果是在图片外层的内容块上应用了height:auto则会产生一些问题：规则的内容块会被拉伸填充整个容器。这时，我们可以使用可变高度(Variable Height)方式解决这个问题。问题的原因可能是渲染图片时要计算图片高度，这就如同你自己定义了图片高度一样，浏览器得到了图片高度就不会像其他情况一样去解析margin:auto垂直居中了。所以我们最好对图片自身应用这些样式而不是父元素。



HTML:

**[html]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. **<img** src="http://placekitten.com/g/500/200" class="Absolute-Center is-Image" alt="" **/>**

CSS:

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Absolute-Center.is-Image {
2. **height**: auto;
3. }
5. .Absolute-Center.is-Image img {
6. **width**: 100%;
7. **height**: auto;
8. }

最好是对图片自身应用此方法，效果如下图：



**八、可变高度（Variable Height）**

这种情况下实现绝对居中（AbsoluteCentering）必须要声明一个高度，不管你是基于百分比的高度还是通过max-height控制的高度，还有，别忘了设置合适的overflow属性。对自适应/响应式情景，这种方法很不错。

与声明高度效果相同的另一种方法是设置display:table;这样无论实际内容有多高，内容块都会保持居中。这种方法在一些浏览器（如IE/FireFox）上会有问题，我的搭档[Kalley](http://stackoverflow.com/users/2539605/kalley" \t "_blank)

在[ELL Creative](http://blog.csdn.net/freshlover/article/details/ellcreative.com" \t "_blank)（访问ellcreative.com ）上写了一个基于Modernizr插件的检测函数，用来检测浏览器是否支持这种居中方法，进一步增强用户体验。

Javascript：

**[javascript]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. /\* Modernizr Test for Variable Height Content \*/
2. Modernizr.testStyles('#modernizr { display: table; height: 50px; width: 50px; margin: auto; position: absolute; top: 0; left: 0; bottom: 0; right: 0; }', **function**(elem, rule) {
3. Modernizr.addTest('absolutecentercontent', Math.round(window.innerHeight / 2 - 25) === elem.offsetTop);
4. });

CSS：

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .absolutecentercontent .Absolute-Center.is-Variable {
2. **display**: table;
3. **height**: auto;
4. }



**缺点：**

浏览器兼容性不太好，若Modernizr不能满足你的需求，你需要寻找其他方法解决。

1.      与上述重绘(Resizing)情况的方法不兼容

2.      Firefox/IE8:使用display:table会使内容块垂直居上，不过水平还是居中的。

3.      IE9/10: 使用display:table会使内容块显示在容器左上角。

4.      Mobile Safari:内容块垂直居中；若使用百分比宽度，水平方向居中会稍微偏离中心位置。

**Ⅱ.其他居中实现技术**

**绝对居中（Absolute Centering）**是一种非常不错的技术，除此之外还有一些方法可以满足更多的具体需求，最常见的推荐：NegativeMargins, Transforms,Table-Cell, Inline-Block方式和新出现的Flexbox.方式。这些方法许多文章都有深入讲解，这里只做简单阐述。

**九、负外边距(Negative Margins)**

这或许是当前最流行的使用方法。如果块元素尺寸已知，可以通过以下方式让内容块居中于容器显示：

外边距margin取负数，大小为width/height（不使用box-sizing: border-box时包括padding，）的一半，再加上top: 50%; left: 50%;。即：

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .is-Negative {
2. **width**: 300px;
3. **height**: 200px;
4. **padding**: 20px;
5. **position**: absolute;
6. top: 50%; left: 50%;
7. **margin-left**: -170px; /\* (width + padding)/2 \*/
8. **margin-top**: -120px; /\* (height + padding)/2 \*/
9. }



测试表明，这是唯一在IE6-IE7上也表现良好的方法。

**优点：**

1.      良好的跨浏览器特性，兼容IE6-IE7。

2.      代码量少。

**缺点：**

1.      不能自适应。不支持百分比尺寸和min-/max-属性设置。

2.      内容可能溢出容器。

3.      边距大小与padding,和是否定义box-sizing: border-box有关，计算需要根据不同情况。

**十、变形（Transforms）**

这是最简单的方法，不近能实现绝对居中同样的效果，也支持联合可变高度方式使用。内容块定义transform: translate(-50%,-50%)必须带上浏览器厂商的前缀，还要加上

top: 50%; left: 50%;

代码类：

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .is-Transformed {
2. **width**: 50%;
3. **margin**: auto;
4. **position**: absolute;
5. top: 50%; left: 50%;
6. -webkit-transform: translate(-50%,-50%);
7. -ms-transform: translate(-50%,-50%);
8. transform: translate(-50%,-50%);
9. }



**优点：**

1.      内容可变高度

2.      代码量少

**缺点：**

1.      IE8不支持

2.      属性需要写浏览器厂商前缀

3.      可能干扰其他transform效果

4.      某些情形下会出现文本或元素边界渲染模糊的现象

进一步了解transform实现居中的知识可以参考CSS-Tricks的文章《[Centering PercentageWidth/Height Elements](http://css-tricks.com/centering-percentage-widthheight-elements/" \t "_blank)》

**十一、表格单元格（Table-Cell）**

总的说来这可能是最好的居中实现方法，因为内容块高度会随着实际内容的高度变化，浏览器对此的兼容性也好。最大的缺点是需要大量额外的标记，需要三层元素让最内层的元素居中。

HTML：

**[html]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. **<div** class="Center-Container is-Table"**>**
2. **<div** class="Table-Cell"**>**
3. **<div** class="Center-Block"**>**
4. <!-- CONTENT -->
5. **</div>**
6. **</div>**
7. **</div>**

CSS：

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Center-Container.is-Table { **display**: table; }
2. .is-Table .Table-Cell {
3. **display**: table-cell;
4. **vertical-align**: middle;
5. }
6. .is-Table .Center-Block {
7. **width**: 50%;
8. **margin**: 0 auto;
9. }



**优点：**

1.      高度可变

2.      内容溢出会将父元素撑开。

3.      跨浏览器兼容性好。

**缺点：**

需要额外html标记

了解更多表格单元格实现居中的知识，请参考Roger Johansson发表在456bereastreet的文章《[Flexibleheight vertical centering with CSS, beyond IE7](http://blog.csdn.net/freshlover/article/details/www.456bereastreet.com/archive/201103/flexible_height_vertical_centering_with_css_beyond_ie7/" \t "_blank)》

**十二、行内块元素（Inline-Block）**

很受欢迎的一种居中实现方式，基本思想是使用display: inline-block, vertical-align: middle和一个伪元素让内容块处于容器中央。这个概念的解释可以参考CSS-Tricks上的文章《[Centering in the Unknown](http://css-tricks.com/centering-in-the-unknown/" \t "_blank)》

我这个例子也有一些其他地方见不到的小技巧，有助于解决一些小问题。

如果内容块宽度大于容器宽度，比如放了一个很长的文本，但内容块宽度设置最大不能超过容器的100%减去0.25em，否则使用伪元素:after内容块会被挤到容器顶部，使用:before内容块会向下偏移100%。

如果你的内容块需要占据尽可能多的水平空间，可以使用max-width: 99%;（针对较大的容器）或max-width: calc(100% -0.25em)（取决于支持的浏览器和容器宽度）。

HTML：

**[html]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. **<div** class="Center-Container is-Inline"**>**
2. **<div** class="Center-Block"**>**
3. <!-- CONTENT -->
4. **</div>**
5. **</div>**

CSS：

**[css]** [view plain](http://blog.csdn.net/freshlover/article/details/11579669) [copy](http://blog.csdn.net/freshlover/article/details/11579669)

1. .Center-Container.is-Inline {
2. **text-align**: center;
3. **overflow**: auto;
4. }
6. .Center-Container.is-Inline:after,
7. .is-Inline .Center-Block {
8. **display**: inline-block;
9. **vertical-align**: middle;
10. }
12. .Center-Container.is-Inline:after {
13. **content**: '';
14. **height**: 100%;
15. **margin-left**: -0.25em; /\* To offset spacing. May vary by font \*/
16. }
18. .is-Inline .Center-Block {
19. **max-width**: 99%; /\* Prevents issues with long content causes the content block to be pushed to the top \*/
20. /\* max-width: calc(100% - 0.25em) /\* Only for IE9+ \*/
21. }

这种方法的优劣和单元格Table-Cell方式差不多，起初我把这种方式忽略掉了，因为这确实是一种hack方法。不过，无论如何，这是很流行的一种用法，浏览器支持的也很好。



**优点：**

1.      高度可变

2.      内容溢出会将父元素撑开。

3.      支持跨浏览器，也适应于IE7。

**缺点：**

1.需要一个容器

2.水平居中依赖于margin-left: -0.25em;该尺寸对于不同的字体/字号需要调整。

3.内容块宽度不能超过容器的100% - 0.25em。

更多相关知识参考ChrisCoyier的文章《[Centeringin the Unknown](http://css-tricks.com/centering-in-the-unknown/" \t "_blank)》

**十三、Flexbox**

这是CSS布局未来的趋势。Flexbox是**[css3](http://lib.csdn.net/base/css3" \o "CSS3知识库" \t "_blank)**新增属性，设计初衷是为了解决像垂直居中这样的常见布局问题。相关的文章如《[Centering Elements with Flexbox](http://coding.smashingmagazine.com/2013/05/22/centering-elements-with-flexbox/" \t "_blank)》

记住Flexbox不只是用于居中，也可以分栏或者解决一些令人抓狂的布局问题。



**优点：**

1.内容块的宽高任意，优雅的溢出。

2.可用于更复杂高级的布局技术中。

**缺点：**

1.      IE8/IE9不支持。

2.      Body需要特定的容器和CSS样式。

3.      运行于现代浏览器上的代码需要浏览器厂商前缀。

4.      表现上可能会有一些问题

有关Flexbox Centering的文章可以参考David Storey的文章《[Designing CSS Layouts WithFlexbox Is As Easy As Pie](http://coding.smashingmagazine.com/2013/05/22/centering-elements-with-flexbox/" \t "_blank)》

**建议：**

每种技术都有其优劣之处。你选择哪一种技术取决于支持的浏览器和你的编码。使用上面的对照表有助于你做出决定。

作为一种简单的替代方案，绝对居中(Absolute Centering)技术表现良好。曾经你使用负边距（Negative Margins）的地方，现在可以用绝对居中(Absolute Centering)替代了。你不再需要处理讨厌的边距计算和额外的标记，而且还能让内容块自适应大小居中。

如果你的站点需要可变高度的内容，可以试试单元格(Table-Cell)和行内块元素(Inline-Block)这两种方法。如果你处在流血的边缘，试试Flexbox，体验一下这一高级布局技术的好处吧。

