**六种实现元素水平居中**

作者：大漠 日期：2013-03-08 点击：170329

[水平居中](https://www.w3cplus.com/blog/tags/71.html)

[垂直居中](https://www.w3cplus.com/blog/tags/131.html)

[](https://study.miaov.com/)

编辑推荐：诚征广告商金主入驻此广告位置，如有感兴趣的金主，欢迎邮件至：airenliao@gmail.com咨询相关合作事宜!!!(^\_^)

居中效果在CSS中很是普通的效果，平时大家所看到的居中效果主要分为三大类：水平居中、垂直居中和水平垂直居中。而其中水平居中相对于后两者来说要简单得多。早期总结了一下互联网上有关于水平垂直居中的几种实现方案，比如说《[CSS制作水平垂直居中对齐](http://www.w3cplus.com/css/vertically-center-content-with-css)》中介绍了八中实现水平垂直的方案，而在《[CSS制作图片水平垂直居中](http://www.w3cplus.com/node/131)》一文介绍了四种实现图片垂直居中的方案，并且在《[CSS3实现水平垂直居中](http://www.w3cplus.com/codes/vertically-center-content-with-css3.html)》使用了css3的flexbox的属性轻松实现多行文本水平垂直居中的方法。当然大家有可能认为这些方法对于浏览嘎嘎的兼容性处理太烦了，也有人使用jQuery的方法实现水平垂直居中效果，比如在《[jQuery制作元素在屏幕中水平垂直居中效果](http://www.w3cplus.com/codes/center-element-screen-jquery.html)》中介绍的。

回到我们今天的话题，水平居中的实现方案，大家最熟悉的莫过开给元素定一个显示式的宽度，然后加上margin的左右值为auto。如：

.center {

width: 960px;

margin-left: auto;

margin-right: auto;

}

这种方法给知道了宽度的元素设置居中是最方便不过的了，但有很多情况之下，我们是无法确定元素容器的宽度。换句话说，未有明确宽度的时候，上面的方法无法让我们实现元素水平居中。那要怎么办呢？这也就是我们今天需要讨论的问题。

为了更好的说明问题，我们来看一个制作分页效果的代码：

**HTML**

<div class="pagination">

<ul>

<li><a href="#">Prev</a></li>

<li><a href="#">1</a></li>

<li><a href="#">2</a></li>

<li><a href="#">3</a></li>

<li><a href="#">4</a></li>

<li><a href="#">5</a></li>

<li><a href="#">Next</a></li>

</ul>

</di

给分页加上样式：

.pagination li {

line-height: 25px;

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

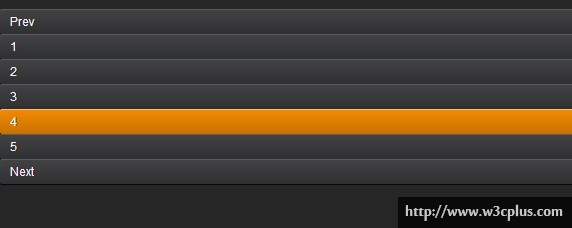
text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

}

这是一个极普通的样式代码，初步的效果：

[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/demo.html)

这很显然不是我们需要的效果，接下来我们分几种方案来制作：

**一、margin和width实现水平居中**

第一种方法是最古老的实现方案，也是大家最常见的方案，在分页容器上定义一个宽度，然后配合margin的左右值为“auto”实现效果：

**.pagination {**

**width: 293px;**

**margin-left: auto;**

**margin-right: auto;**

**}**

.pagination li {

line-height: 25px;

**display: inline;**

**float: left;**

**margin: 0 5px;**

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

}

代码中绿色部分是为了实现分页居中效果而添加的代码。（下文中没有特殊声明，绿色部分代码表示新增加的代码。），先来看看效果：

[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/horizontal-center/demo1.html)

效果是让我们实现了，但其扩展性那就不一定强了。示例中只显示了五页和向前向后的七个显项，但往往我们很多情况下是不知道会有多少个分页项显示出来，而且也无法确定每个分页选项的宽度是多少，也就无法确认容器的宽度。

**优点：**实现方法简单易懂，浏览器兼容性强；

**缺点：**扩展性差，无法自适应未知项情况。

**二、inline-block实现水平居中方法**

这个方法早期在《[如何解决inline-block元素的空白间距](http://www.w3cplus.com/css/fighting-the-space-between-inline-block-elements)》和《[CSS3制作的分页导航](http://www.w3cplus.com/css3/creat-pager-list-with-css3)》中都有涉及到，但未单独提取出来。此次，将这种方法拿出来说。

仅inline-block属性是无法让元素水平居中，他的关键之处要在元素的父容器中设置text-align的属性为“center”，这样才能达到效果：

**.pagination {**

**text-align: center;**

**font-size: 0;**

**letter-spacing: -4px;**

**word-spacing: -4px;**

**}**

.pagination li {

line-height: 25px;

margin: 0 5px;

**display: inline-block;**

**\*display: inline;**

**zoom:1;**

**letter-spacing: normal;**

**word-spacing: normal;**

**font-size: 12px;**

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

}

效果如下：

[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/horizontal-center/demo2.html)

这个方法相对来说也是简单易懂，但使用了inline-block解决了水平居中的问题，却又产生了一个新的问题，就是分页项与分页项由回车符带来的空白间距，那么不知情的同学就会不知道如何解决？（而且这个间距并不是所有浏览器都有），所以需要解决下inline-block带来的间距，详细的解决方法可以阅读《[如何解决inline-block元素的空白间距](http://www.w3cplus.com/css/fighting-the-space-between-inline-block-elements)》一文。

**做点：**简单易懂，扩展性强；

**缺点：**需要额外处理inline-block的浏览器兼容性。

**三、浮动实现水平居中的方法**

刚看到标题，大家可能会感到很意外，元素都浮动了，他还能水平居中？大家都知道，浮动要么靠左、要么靠右，还真少见有居中的。其实略加处理就有了。

**.pagination {**

**float: left;**

**width: 100%;**

**overflow: hidden;**

**position: relative;**

**}**

**.pagination ul {**

**clear: left;**

**float: left;**

**position: relative;**

**left: 50%;*/\*整个分页向右边移动宽度的50%\*/***

**text-align: center;**

**}**

.pagination li {

line-height: 25px;

margin: 0 5px;

**display: block;**

**float: left;**

**position: relative;**

**right: 50%;*/\*将每个分页项向左边移动宽度的50%\*/***

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

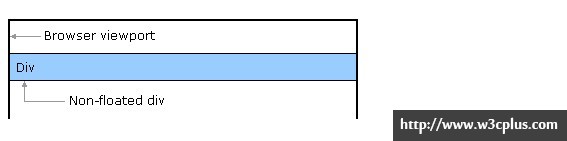
}

效果如下所示：

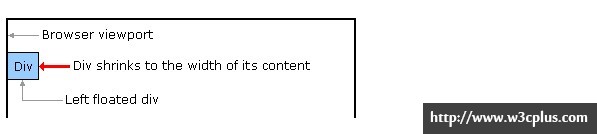
[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/horizontal-center/demo3.html)

这种方法实现和前面的与众不同，使用了浮动配合position定位实现。下面简单的介绍了一下这种方法实现原理，详细的可以阅读[Matthew James Taylor](http://matthewjamestaylor.com/about)写的《[Horizontally Centered Menus with no CSS hacks](http://matthewjamestaylor.com/blog/beautiful-css-centered-menus-no-hacks-full-cross-browser-support)》一文。

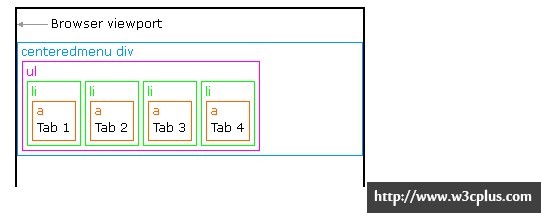
没有浮动的div：大家都知道div是一个块元素，其默认的宽度就是100%，如图所示：



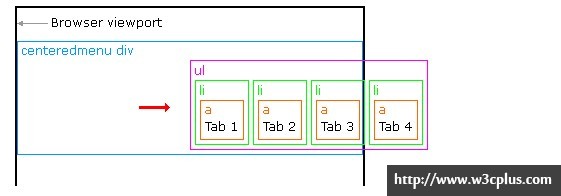
如果div设置了浮动之后，他的内容有多宽度就会撑开有多大的容器（除显式设置元素宽度值之外），这也是我们实现让分页导航居中的关键所在：



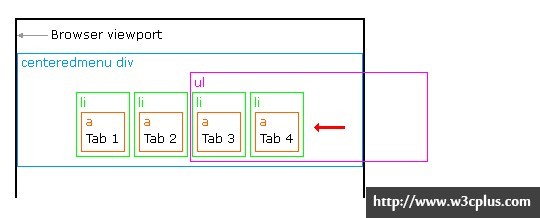
接下来使用传统的制作方法，我们会让导航浮动到左边，而且每个分页项也进行浮动，就如下图所示一样：



现在要想的办法是让分页导航居中的效果了，在这里是通过“position:relative”属性实现，首先在列表项“ul”上向右移动50%（left:50%;），看到如下图所示：



如上图所示一样，整个分页向右移动了50%的距离，紧接着我们在“li”上也定义“position:relative”属性，但其移动的方向和列表“ul”移动的方向刚好是反方向，而其移动的值保持一致：



这样一来就实现了float浮动居中的效果。

**特别声明：**方法三思想来源于[Matthew James Taylor](http://matthewjamestaylor.com/about)写的《[Horizontally Centered Menus with no CSS hacks](http://matthewjamestaylor.com/blog/beautiful-css-centered-menus-no-hacks-full-cross-browser-support)》一文，并且引用其文中演示的示意图。

**优点：**兼容性强，扩展性强；

**缺点：**实现原理较复杂。

**四、绝对定位实现水平居中**

绝对定位实现水平居中，我想大家也非常的熟悉了，并且用得一定不少，早期是这样使用的：

.ele {

position: absolute;

width: 宽度值;

left: 50%;

margin-left: -(宽度值/2);

}

但这种实现我们有一个难题，我并不知道元素的宽度是多少，这样也就存在如方法一所说的难题，但我们可以借助方法三做一点变通：

**.pagination {**

**position: relative;**

**}**

**.pagination ul {**

**position: absolute;**

**left: 50%;**

**}**

.pagination li {

line-height: 25px;

margin: 0 5px;

**float: left;**

**position: relative;*/\*注意，这里不能是absolute，大家懂的\*/***

**right: 50%;**

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

}

效果如下所示：

[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/horizontal-center/demo4.html)

**优点：**扩展性强，兼容性强;

**缺点：**理解性难。

**五、CSS3的flex实现水平居中方法**

CSS3的[flex](http://www.w3cplus.com/blog/666.html)是一个很强大的功能，她能让我们的布局变得更加灵活与方便，唯一的就是目前浏览器的兼容性较差。那么第五种方法，我们就使用flex来实现，其实这种方法早在《[CSS3实现水平垂直居中](http://www.w3cplus.com/codes/vertically-center-content-with-css3.html)》一文有介绍，我们把水平居中的部分代码取出来：

**.pagination {**

**display: -webkit-box;**

**-webkit-box-orient: horizontal;**

**-webkit-box-pack: center;**

**display: -moz-box;**

**-moz-box-orient: horizontal;**

**-moz-box-pack: center;**

**display: -o-box;**

**-o-box-orient: horizontal;**

**-o-box-pack: center;**

**display: -ms-box;**

**-ms-box-orient: horizontal;**

**-ms-box-pack: center;**

**display: box;**

**box-orient: horizontal;**

**box-pack: center;**

**}**

.pagination li {

line-height: 25px;

margin: 0 5px;

**float: left;**

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

}

效果如下：

[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/horizontal-center/demo5.html)

**优点：**实现便捷，扩展性强

**缺点：**兼容性差。

**六、CSS3的fit-content实现水平居中方法**

今天看《[Horizontal centering using CSS fit-content value](http://www.red-team-design.com/horizontal-centering-using-css-fit-content-value)》一文，让我体验了一下"fit-content"制作水平居中的方法。我也将这种方法收进来。

“fit-content”是CSS中给“width”属性新加的一个[属性值](http://dev.w3.org/csswg/css3-box/#the-width-and-height-properties)，他配合margin可以让我轻松的实现水平居中的效果：

**.pagination ul {**

**width: -moz-fit-content;**

**width:-webkit-fit-content;**

**width: fit-content;**

**margin-left: auto;**

**margin-right: auto;**

**}**

.pagination li {

line-height: 25px;

margin: 0 5px;

**float: left;**

}

.pagination a {

display: block;

color: #f2f2f2;

text-shadow: 1px 0 0 #101011;

padding: 0 10px;

border-radius: 2px;

box-shadow: 0 1px 0 #5a5b5c inset,0 1px 0 #080808;

background: linear-gradient(top,#434345,#2f3032);

}

.pagination a:hover {

text-decoration: none;

box-shadow: 0 1px 0 #f9bd71 inset,0 1px 0 #0a0a0a;

background: linear-gradient(top,#f48b03,#c87100);

}

效果如下：

[](http://www.w3cplus.com/sites/default/files/blogs/demos/horizontal-center/demo6.html)

**优点：**简单易懂，扩展性强;

**缺点：**浏览器兼容性差

上面总共为大家整理了六种实现水平居中的方法，希望对大家有所帮助。如果您有更好的建议，希望能与我们一起分享。

如需转载，烦请注明出处：<http://www.w3cplus.com/css/elements-horizontally-center-with-css.html>

上一篇: [深入了解 Flexbox 伸缩盒模型](https://www.w3cplus.com/blog/666.html) | 下一篇: [通俗的英文指南——Javascript的原型](https://www.w3cplus.com/js/a-plain-english-guide-to-javascript-prototypes.html)

著作权归作者所有。  
商业转载请联系作者获得授权,非商业转载请注明出处。  
原文: <https://www.w3cplus.com/css/elements-horizontally-center-with-css.html> © [w3cplus.com](https://www.w3cplus.com/)