**一、CSS元素隐藏**

在CSS中，让元素隐藏（指屏幕范围内肉眼不可见）的方法很多，有的占据空间，有的不占据空间；有的可以响应点击，有的不能响应点击。后宫选秀——一个一个看。

{ display: none; /\* 不占据空间，无法点击 \*/ }

{ visibility: hidden; /\* 占据空间，无法点击 \*/ }

{ position: absolute; clip:rect(1px 1px 1px 1px); /\* 不占据空间，无法点击 \*/ }

{ position: absolute; top: -999em; /\* 不占据空间，无法点击 \*/ }

{ position: relative; top: -999em; /\* 占据空间，无法点击 \*/ }

{ position: absolute; visibility: hidden; /\* 不占据空间，无法点击 \*/ }

{ height: 0; overflow: hidden; /\* 不占据空间，无法点击 \*/ }

{ opacity: 0; filter:Alpha(opacity=0); /\* 占据空间，可以点击 \*/ }

{ position: absolute; opacity: 0; filter:Alpha(opacity=0); /\* 不占据空间，可以点击 \*/ }

{

zoom: 0.001;

-moz-transform: scale(0);

-webkit-transform: scale(0);

-o-transform: scale(0);

transform: scale(0);

/\* IE6/IE7/IE9不占据空间，IE8/FireFox/Chrome/Opera占据空间。都无法点击 \*/

}

{

position: absolute;

zoom: 0.001;

-moz-transform: scale(0);

-webkit-transform: scale(0);

-o-transform: scale(0);

transform: scale(0);

/\* 不占据空间，无法点击 \*/

}

**欢迎补充！**

以上隐藏方法如果针对最小辈的子元素（膝下无子女），例如<img>图片元素，则上面的注释说明都是OK的(若有误，请指正)。

我以前就提过，为了便于理解，我们会使用情感化思维，类比现实，在内心形成只属于自己的具象化的想法。这是很OK的，但是，要知道，事物是具有两面的，具象的东西容易让我们陷入固定思维模式，尤其在现实世界中非特殊情境的验证下，从而让我们不经意间产生一些错误的认识。

上面的话有点像老油条——难嚼。要让老少皆懂，很简单，举几个简单的示例就可以了。而这里的示例就是上头几个隐藏的例子：由于具象化的固定思维，您可能并不知道下面要提到的几种CSS元素隐藏“失效”的情况！

**二、display:none和visibility:hidden**

我记得校园招聘那会儿，技术面的时候有问我display:none和visibility:hidden的区别在哪儿？

“这个问题简单？”我心里头暗自得意，按耐住自己得意又紧张的小心脏，自信满满地说，“这两个声明都可以让元素隐藏，不同之处在于display:none隐藏后的元素不占据任何空间，而visibility:hidden隐藏的元素空间依旧存在。”

想在想想，自己实在是浮躁了点。这种程度的答案说明你在HTML/CSS方面的造诣只是了解这个层面，并不能让面试官眼前一亮。放眼整个中国，写页面的人何其多，像这种程度回答的人一定就像河里的沙子一样，一抓一大把。随便用脚趾头想想也知道，像度娘，企鹅公司花费那么多人力和财力来校园是肯定来挑金子的，不起眼的沙子就让它随波逐流，流到大海暖化石去吧。

目前，我所知道的不同有三点（欢迎补充）：

1. 空间占据
2. 回流与渲染
3. 株连性

第一点，想必都知道；第二点我之前有[多篇译文](http://www.zhangxinxu.com/wordpress/?s=%E5%9B%9E%E6%B5%81)提过，display:none隐藏产生reflow和repaint(回流与重绘)，而visibility:hidden没有这个影响前端性能的问题；第三点估计是不少同行不知道的，就是“株连性”方面的差异。

所谓“株连性”，就是如果祖先元素遭遇某祸害，则其子子孙孙无一例外也要遭殃。我顿时想起了《地球反击战》或是《木乃伊之蝎子王》，一旦把母体搞跛了，小辈啊、下属啊什么的都瞬间烟消云散。display:none就是“株连性”明显的声明：一旦父节点元素应用了display:none，父节点及其子孙节点元素全部不可见，而且无论其子孙元素如何不屈地挣扎都无济于事。

在实际的web应用中，我们要经常实现一些显示隐藏的功能，由于display:none本身特性以及jQuery潜在的驱动，使得我们对display:none这种隐藏特性相当熟知。因此，久而久之会形成比较牢固的情感化认识，并无法避免地将这种认识迁移到其他类似表现属性(eg. visibility)的认识上，再加上一些常规经验……

举例来说吧，通常情况下，我们给一个父元素应用visibility:hidden，则其子孙后代也都会全部不可见。于是，我们就会有类似的认识迁移：应用了visibility:hidden声明下的子孙元素如何不屈地挣扎都摆脱不了不可见被抹杀的命运。而实际上却存在隐藏“失效”的情况。

何时隐藏“失效”？很简单，如果子孙元素应用了visibility:visible，那么这个子孙元素又会刘谦般地显现出来。

您可以狠狠地点击这里：[CSS visibility属性的株连性测试demo](http://www.zhangxinxu.com/study/201202/css-visibility-break-parent.html)



visibility就是这样一个funny的属性。

**对比总结：**  
display:none是个相当惨无人道的声明，子孙后代全部搞死（株连性），而且连块安葬的地方都不留（不留空间），导致全体民众哗然（渲染与回流）。  
visibility:hidden则具有人道主义关怀，虽然不得已搞死子孙，但是子孙可以通过一定手段避免（伪株连性），而且死后全尸，墓地俱全（占据空间），国内民众比较淡然（无渲染与回流）。

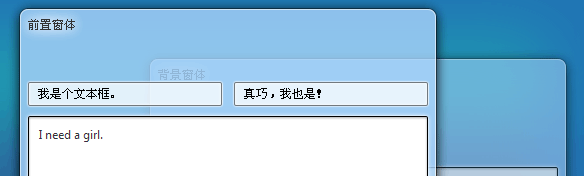
**附上**  
本节末了，您可以狠狠地点击这里：[display:none株连性测试demo](http://www.zhangxinxu.com/study/201202/css-display-none-break-parent.html)

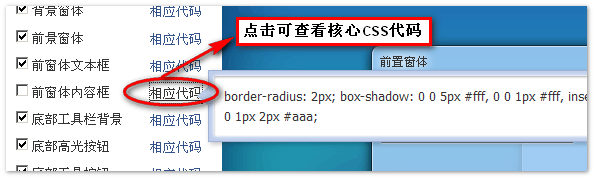
可以看到demo页面中张美女都跟靓仔过情人节去了。

**三、visibility隐藏“失效”妙用**

visibility隐藏“失效” 的应用其实在老早的时候就有了，等下，让我找找~~

啊哈，找到了。  
去年的去年5月份[CSS3模拟window7炫酷界面效果demo](http://www.xuwanting.com/study/201005/css3-window7-ui.html)，原文点击[这里](http://www.zhangxinxu.com/wordpress/?p=810)。





复选框勾选与否对应的部分显示与隐藏。例如，偶们要让父标签的圆角啊、投影啊、背景啊什么的都隐掉，只留里面的文本框。该怎么实现？如果是要display:none，则里面的文本框框也会被一并抹杀掉的，如果是要脚本控制style，哇哦哦，这么多样式，想想脑袋就大了。毋庸置疑，这里使用visibility控制是最高效也是最巧妙的方法。

**四、height:0和overflow:hidden的组合**

overflow:hidden用中文理解就是“溢出隐藏”，也就是盒子以外的内容都咔嚓掉不可见的。加上height:0，只要是一般的非inline水平元素，则元素内部所有子孙都应该是不可见的。这是我们具象化的认识，而且N多的经验验证都是如此，但是实际上是这样子吗？

先看一个反例demo，您可以狠狠地点击这里：[0高度溢出隐藏“失效”demo](http://www.zhangxinxu.com/study/201202/css-overflow-hidden-break-parent.html)



height:0和overflow:hidden组合隐藏“失效”的条件如下：祖先元素没有position:relative/absolute/fixed声明，同时内部子元素应用了position:absolute/fixed声明。

**为什么会这样？**  
先看下CSS2.1规范中对overflow的一点解释：

This property specifies whether content of a block container element is clipped when it overflows the element’s box. It affects the clipping of all of the element’s content except any descendant elements (and their respective content and descendants) whose containing block is the viewport or an ancestor of the element.

什么意思呢？我琢磨了又琢磨，唉，只叹理解不够深入，也有些理不清。我想大致意思应该这样（若有不对应该指正）：overflow可以剪裁超出块状元素之外的元素。除非超出元素的包含块是整个视区或是该overflow元素的祖先元素。然后绝对定位元素的包含块应该就是含有position:relative/absolute/fixed的祖先元素。

因此：

body

height: 0; overflow: hidden;

position: absolute; /\* 不会被隐藏 \*/

position: relative;

height: 0; overflow: hidden;

position: absolute; /\* 不会被隐藏 \*/

height: 0; overflow: hidden; position: relative;

position: absolute; /\* 会被隐藏 \*/

height: 0; overflow: hidden;

position: relative;

position: absolute; /\* 会被隐藏 \*/

记住这个：position: absolute元素溢出overflow: hidden元素的时候，如果其第一个含有position属性(static除外)的祖先元素（一直到body）是overflow: hidden元素祖先元素的时候，则不隐藏；否则，隐藏。

或者记住这个：火星人在地球触发了法律，如果火星人的老爸在这法律之外，则这个火星人啥事没有；否则，坐牢！

**五、height:0和overflow:hidden“失效”妙用**

前不久在[“absolute元素在text-align属性下的对齐显示”](http://www.zhangxinxu.com/wordpress/?p=2124)一文中介绍了一种简单使用text-align属性实现“返回顶部”按钮定位的方法。

您可以狠狠地点击这里：[text-align:right声明下的返回顶部定位demo](http://www.zhangxinxu.com/study/201112/text-align-right-position-fixed-top.html)

后来有人反映，此方法好是好，但是有一个问题，因为标签内打了个&nbsp;空格，产生了了空白高度啊！

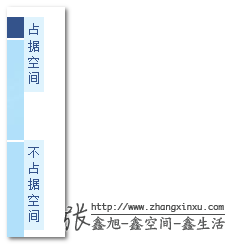
如何fix这里的空格产生的高度呢？使用line-height: 0; ? 或者使用font-size: 0;? 如果不是因为line-height和font-size具有继承性，这确实是个好方法（font-size还有Chrome默认文字无法小于12像素问题，考古版本IE下0像素文字呈现小点的问题）。

经过我多方测试，以在OOCSS开发模式下，这里使用height:0和overflow:hidden是最好的方法。

h0 ovh tr

&nbsp; abs

您可以狠狠地点击这里：[text-align下返回顶部不占据空间demo](http://www.zhangxinxu.com/study/201202/text-align-right-position-absolute-height-0.html)



demo中，上面一个条条&nbsp;产生高度，因此头部背景色，而下面这个条条就没有。因为定位元素是绝对定位的，因此，不要担心会被overflow不小心干掉！