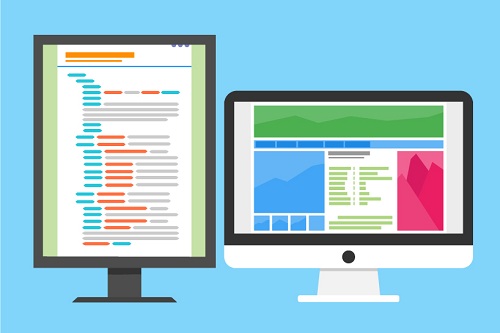
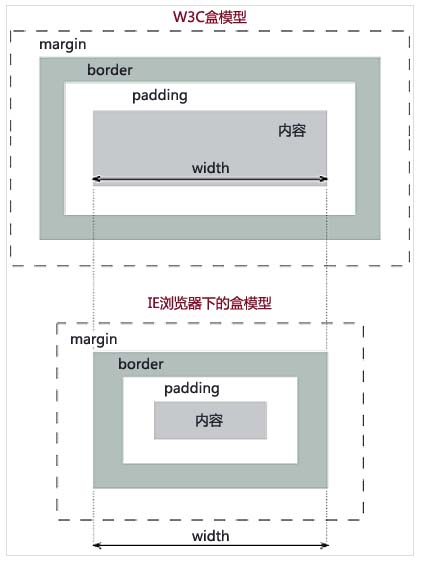
我们先了解下前端经典的“盒子模型”，聪明的程序猿们巧妙的用盒子模型这一形象将枯燥的css代码表示出来：想象一个盒子，它有：外边距（margin）、边框（border）、内边距（padding）、内容（content）四个属性；这些加起来就是一个完整的元素。

[](https://www.yaxi.net/wp-content/uploads/2017/01/20170106181644.jpg)

但盒子模型又分为标准w3c盒模型和IE盒模型。那这有什么区别呢？见下图所示：

[](https://www.yaxi.net/wp-content/uploads/2017/01/20170106181652.jpg)

从上图可以看到标准 w3c 盒子模型的范围包括 margin、border、padding、content，其中内容content的宽度不包括padding、margin、border等部分，也就是说这个盒模型实际占用页面的宽高是内容宽高加上其他部分的宽高，即：总宽度 = margin-left + border-left + padding-left + width + padding-right + border-right + margin-right。

而在IE盒子模型中，内容content的宽度包含了 border 和 pading，这个盒模型实际占用页面的宽高是内容宽高加上外边距的宽高，即总宽度 = margin-left + width + margin-right。

可能说到这里，你还一头雾水，不知我说这个有啥用，不要着急。

在目前流行前端布局中，前端工程师默认的页面布局是采用标准的w3c盒子模型（需要在页面html声明处加上doctype 声明），告知浏览器按照标准盒模型的方式进行渲染页面。

这个时候会出现一个问题，就是元素的宽高包含了实际内容的宽高加上内外边距以及边框的宽高，如果我们想调一下内边距padding或者边框border的尺寸，那么整个元素的实际尺寸就会发生变化，随着而来的就是布局乱掉，可谓是牵一发而动全身。

但css3改变了这一现状，在CSS3中引入了box-sizing属性, 它可以允许改变默认的CSS盒模型对元素宽高的计算方式。

### box-sizing简介

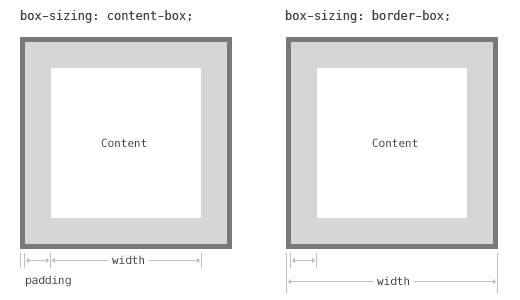
box-sizing属性包括content-box（default），border-box。

1、content-box，border和padding不计算入width之内

2、border-box，border和padding计算入width之内，（竟然是IE盒子模型标准）

注意：ie8级以上浏览器支持content-box和border-box;

所以各位看官明白了吧，如果我们使用了box-sizing：border；这一属性之后，你页面所有元素的宽高就会随着你设置的width、height而固定，无论你再怎么折腾padding、border，整个元素的尺寸是不会有任何变化的，只是实际内容content的尺寸会随着被压缩拉伸。这样的整体布局就稳定了许多，不再会因为某一个元素的边框微调而发生崩塌。

[](https://www.yaxi.net/wp-content/uploads/2017/01/20170106181741.jpg)

虽然IE在大家的眼里是一个奇葩的存在，而ie盒模型标准也跟w3c盒模型标准有出入，但笔者个人认为在盒模型上，ie盒模型更加合理化。因为常规的布局思维是我定好整体的宽高之后，里面元素的整体尺寸会确定下来，后面才是考虑这个元素的内边距以及边框，如果我已经确定好了大体的布局，再去微调边框而导致了整体布局的崩塌，这显然是不科学的，所以这也是我极力推崇前端布局使用box-sizing：border;的原因。

然而一个残酷的显示就是IE8 以下的浏览器对css3是不支持的，所以box-sizing：border;属性在ie8以下的浏览器中是无效的，这可能是影响此css元素流行的重大原因，不知道什么时候业内才能同仇敌忾，一起抵制IE8以下浏览器，还我们前端人员一片净土。

如果你的项目不需要考虑ie6/ie7的用户，那么请大胆的使用box-sizing：border;相信它会让你的前端布局灵活很多，记得在css reset（样式重置）的时候就加上它，一劳永逸....