Appendix I IE Hacks

One of the sad realities of web programming is that Microsoft's Internet Explorer browser simply does not follow the rules. There are a large number of major bugs and incompatibilities in IE that can make us web coders want to tear our hair out. Microsoft has chosen simply not to follow web standards. In particular, IE does not implement the CSS box model the way it is supposed to be implemented. A page that looks one way in IE may look a completely different way in Firefox or Safari. A page that shows up in one browser may be blank or badly misformatted in another. This leaves web programmers with a tough decision: Which users do I want to be able to see my page correctly?

The worst part is, despite the fact that Microsoft's IE team continues to release new versions of their browser (version 8 is being readied as this book goes to press), they choose not to correct these problems. It's not clear whether this is because they want to retain compatibility with existing pages that use the old incorrect behaviors, or as a way of intentionally balkanizing the web and making it harder for the web as a platform to reduce the importance of Microsoft's Windows operating system. Either way, it can be a royal pain to deal with IE, as you'll see if you try to do so as a web dev.

Fortunately, if you are aware of some of IE's most major deficiencies and workarounds for them, you can often create a page that works in both IE and standards-compliant browsers. In this section, we list some of the more common quirks of IE and suggest ways to deal with them. The section is intentionally kept short, because we generally object to Microsoft's decisions here and want to focus on and follow web standards as much as possible.

The Broken Box Model

Various versions of Internet Explorer misinterpret the `width` and `height` properties. It considers them to include the padding and border around the element, which it shouldn't. This is disastrous for web developers who want a consistent appearance to their pages, because the width you set for a standards-compliant browser simply won't match that seen in IE, unless your elements have no padding. The element will appear too thin in IE, since the same number of pixels now have to accommodate the padding and border. Thanks a lot, IE development team.

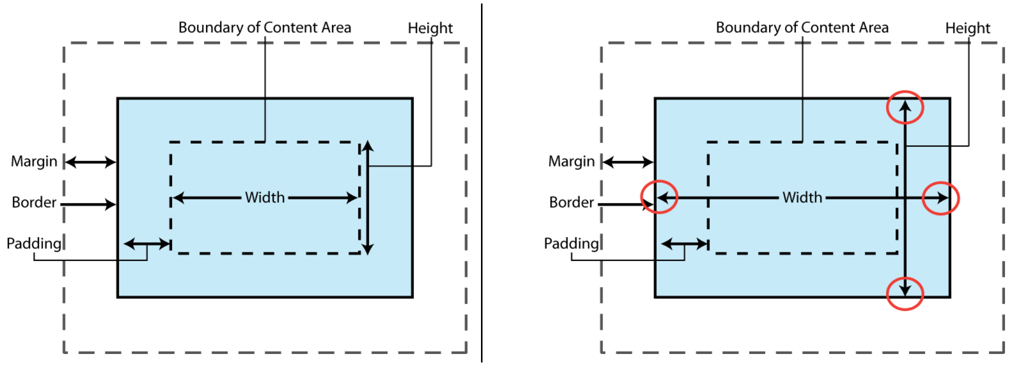


Figure I.1 Box model, standard-compliant (left) and IE (right)

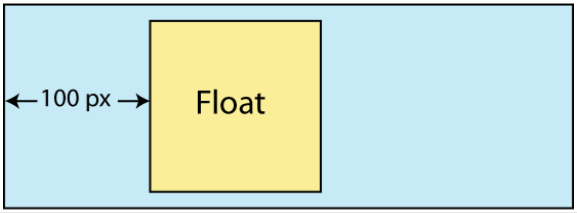
To make matters worse, Internet Explorer 6 also doesn't support the `min-width`, `min-height`, `max-width`, or `max-height` properties. What a pain! Many web developers have come up with hacks to try to emulate the effect of properties like `min-width` in IE, such as the following sites:

- http://www.cssplay.co.uk/boxes/minwidth.html
- http://www.webreference.com/programming/min-width/

The Broken Float Model

If you float an element inside another block element and want to space it from the edge a bit, you can set a margin on the appropriate side. For example, the code in Example I.1 is supposed to space a `div` 100px from the page's left edge. It works properly in standards-compliant browsers.

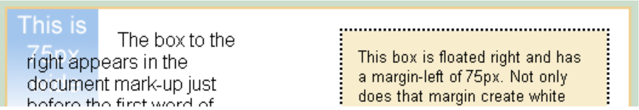
```
div#floating {
  float: left;
  margin-left: 100px;
}
```



Example I.1 Float with left margin

In Internet Explorer 6 when an element is floated left, its left margin is mistakenly doubled by the browser. In this case our floating element actually appears 200px from the left edge.

This is not the only bug related to floating elements; there is a lengthy list of such bugs. For example, when you float elements, often IE6 will mistakenly indent the first line of nearby text with the same amount of indentation as the floating element's `margin-left` setting.



This is 75px

The box to the right appears in the document mark-up just before the first word of

This box is floated right and has a margin-left of 75px. Not only does that margin create white

Figure I.2 IE6 float text indentation bug

A suggested workaround for IE is to set the floated element's `display` to `inline`, which does mysteriously fix the bug. But this is simply a hack and shouldn't make such a drastic change to the page's layout.

```
div#floating {
  display: inline;
  float: left;
  margin-left: 100px;
}
```

Example I.2 Display inline hack for float bug

Here's another float bug. If you float an element to the left and then position other block content underneath it, the block element is supposed to lay itself out normally. But in IE, if the underlying block element has a `width` declaration, it will also be mysteriously turned into a floating element and placed to the side of the floating one. This bug still exists in IE7 (it is fixed in IE8), so the majority of users of the web potentially face this issue. IE also often adds 3px or so of margin between floating elements and other elements, even when they are supposed to touch flush against each other. On and on. The following site has a floating elements test that IE7 and below fail miserably:

- http://css-class.com/articles/explorer/floats/floatandcleartest1.htm

As you can see, getting floating layouts to render correctly in IE can be a grueling experience.

I.1.1 Workarounds for IE Flaws

A lot of times you can make a page look right in Internet Explorer if you can apply different CSS rules to it in IE than you do to other browsers. The idea is to supply one set of CSS rules to IE that account for its misbehavior, and another set to everybody else who follows the rules.

The "Underscore Hack"

Over the years a bunch of CSS "hacks" have been found that can be used to target IE exclusively. For example, recall how IE's box model deals with widths and heights improperly. Some crafty CSS developer discovered that IE will pay attention to CSS rules even if they have certain bad characters in them, such as underscores. For example, if you try to set a property named `_width`, standards-compliant browsers will completely ignore it, but IE will happily set the element's width. People use this to set IE to use a larger width to give space for the element's horizontal padding and borders, as in the code shown in Example I.3.

```
div {
  width: 100px;
  padding: 10px;
  border: 10px solid black;
  _width: 140px;
}
```

Example I.3 IE "underscore hack" (not recommended!)

There are other equally objectionable hacks, such as calling it `w\idth` or using a bogus context selector that only IE will recognize, such as `* html div { ... }`. We strongly discourage against using hacks like these. Many of them will break your CSS file so that it does not pass the W3C validator. IE support isn't worth that.

Fortunately there's a better way to get this same effect of providing a special set of style rules to IE. This can be achieved by a special IE-only proprietary HTML feature called conditional comments. A *conditional comment* is a comment that can contain an `if` statement in it, causing a piece of HTML to be included only if some condition is met. The syntax for conditional comments is:

```
<!--[if condition]>
  HTML code
<![endif]-->
```

The most common thing to write under condition is simply the letters `IE`, meaning, "if this browser is any version of IE." All other browsers will ignore this text and treat it as a large HTML comment. Another variation of a condition checks for a specific IE version, such as `lte IE 6`, since different versions of IE have had different bugs and require different fixes.

Example I.4 demonstrates Internet Explorer's conditional comments. The code uses them to link to a file ie_hacks.css on all versions of IE, and to an additional file ie6_more_hacks.css if the browser is specifically IE6. The ie_hacks.css file contains the 140px width to correct for IE's broken box model, so that the actual content width of the element will be the desired 100px to match the standard-compliant browsers.

```
<!DOCTYPE html>
<html>
  <head>
    <!--[if IE]>
      <link href="ie_hacks.css" type="text/css" rel="stylesheet" />
    <![endif]-->

    <!--[if lte IE 6]>
      <link href="ie6_more_hacks.css" type="text/css" rel="stylesheet" />
    <![endif]-->
    ...
```

```
/* ie_hacks.css file contents */
div {
  width: 140px;
}
```

Example I.4 Internet Explorer conditional comments

References

- Internet Explorer Layout Hacks and Fixes:
  a) http://www.tdrake.net/ie7-hacks/
  b) http://www.positioniseverything.net/ie-primer.html
  c) http://www.satzansatz.de/cssd/onhavinglayout.html