

/CSS/create layouts with floating boxes



Based on the 2000AD Books website, Craig Grannell reveals how to generate a floating-boxes layout that's ideal for an online portfolio. Even better, you can rework the result again and again

Knowledge needed Intermediate CSS and HTML

Requires A text editor

Project time 20 minutes

This issue, I'm going to nobble three flying creatures with one hefty rock. Firstly, the main theme is using floating boxes to create a layout for accessing further content, based on the one at 2000AD Books (www.2000adonline.com/books). On that site, covers are displayed in reverse-chronological order of release date and the thumbnails provide access to further information on each title. When you place the cursor over one of the covers, the colour of its border changes to a vibrant hue, so it's obvious that it's clickable. Secondly, the system is adaptable, so it can be used for a range of sites!

The third 'bird' is reworking this system into a portfolio, so that you can have a specific example to play with. Note that because the gallery code is modular, you can totally rework the layout if you wish.

The web page

Copy the 'floated boxes tutorial' folder to your hard drive. This contains the final web page, blank CSS documents and a load of images in the 'assets' folder. Open 'index.html'. This is an XHTML Strict document, which imports style sheet 'floated-boxes.css', links to the JavaScript document 'image-swapper.js', and uses a conditional comment to link to 'ie-hacks.css' for pre-7 versions of Internet Explorer.

In terms of structure, the page is simple: its content is placed within a wrapper div, which has a contentArea div nested within. This is followed by a level-one heading and divs that make up the top section of the design.

```
<div id="introductoryText">
<p>Lorem ipsum dolor...</p>
<p>Nulla facilisi...</p>
<div class="separator"><!-- x --></div>
<p id="caption">Photo 1 caption</p>
</div>
<div id="portfolioImage">

</div>
<div class="separator"><!-- x --></div>
```

There are essentially two blocks of content within suitably named divs, introductoryText and portfolioImage. The introductory text is followed by a separator div (which will later be styled with a bottom border) and the image caption. The two divs are followed by a further separator div, because they'll both be floated – the separator will be used to clear floated content. (Note that you could use horizontal rules instead of divs for content area underlines; however, I wanted the design to be tight, and the display of margins around <hr /> elements across browsers is variable at best.)

Next on the page is the thumbnails div, which first contains a level-two heading (with a nested 'span' element, which will be floated right), and then a dozen divs with thumbnails within, an example of which is below.

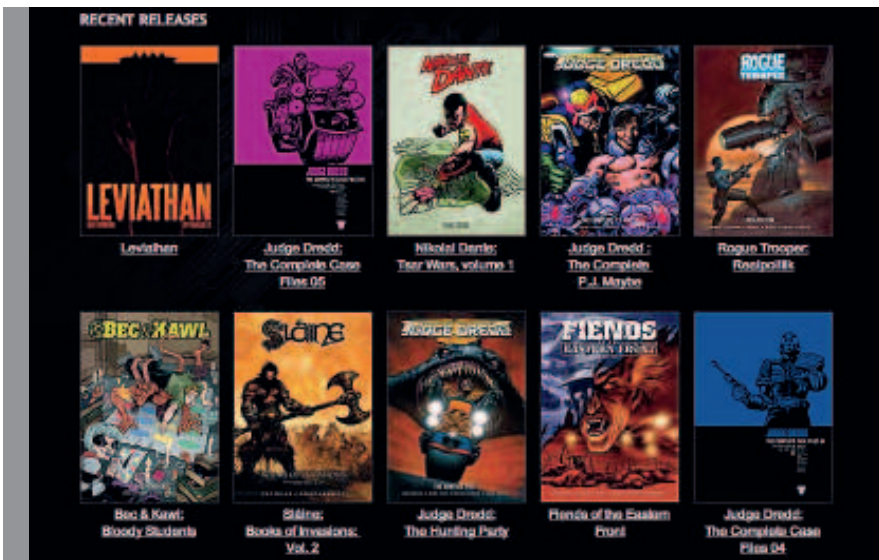
```
<div>
<a href="assets/gallery/photo-1.jpg" onclick="javascript:swapPhoto('photo-
1.jpg;Photo 1 caption'); return false;"></a>
<p><a href="assets/gallery/photo-1.jpg" onclick="javascript:swapPhoto('photo-
1.jpg;Photo 1 caption'); return false;">Lorem ipsum dolor sit amet</a></p>
</div>
```

As you can see, this is a plain vanilla div, with an image and some text. The text and the image are both surrounded by identical anchor elements, which can be broken down as follows: the 'href' attribute links to the related large image/photo, which is accessed if the user has disabled JavaScript; the 'onclick' value invokes the 'swapPhoto' function (open 'image-swapper.js' to see the function), and provides two values – the filename of the target photo (the path to the photos is defined in the function) and a brief caption for the image; and 'return false', which ensures that the 'onclick' details are used instead of the 'href' value for JavaScript-enabled browsers.

Styling the page structure

Open the file 'floated-boxes.css'. In the defaults section, add rules to remove padding and margins from all elements, and to set default colours for text and the page background.

```
* {
padding: 0;
margin: 0;
}
```



The inspiration Take a look at the 2000AD Books website to get an idea of the type of layout you're about to create. It was designed by our very own Craig Grannell. The thumbnails are used to access information on recent releases, which is a great use for the system



Flexible code The code used for this tutorial is modular when stripped to its bare bones, so it can be altered and reused. Here, it's being used at www.snubcommunications.com/iceland/

```
body {
color: #ffffff;
background: #000000;
}
```

Add the following five rules to the structure section, thereby styling the containing wrapper div (setting its width, centring it via the 'margin' values, and adding padding to the top and bottom), the introductoryText and portfolioImage divs (floating both right, setting right-hand margin values, and assigning fixed widths), the separator divs (clearing floated content, setting a height of 1px, and adding a solid, grey bottom border), and the separatorInvisible div (which is as per the separator divs, but without any background colour or border).

```
#wrapper {
width: 780px;
margin: 0 auto;
padding: 5px 0;
}
```

```
#introductoryText {
float: right;
width: 272px;
margin-right: 12px;
}
```

```
#portfolioImage {
float: right;
margin-right: 20px;
width: 465px;
}
```

```
.separator {
clear: both;
height: 1px;
background: #000000;
border-bottom: 1px solid #7f7f7f;
}
```

```
.separatorInvisible {
clear: both;
height: 1px;
}
```

As stated earlier, the thumbnails are all placed in divs within the thumbnails div. This containing div is styled using the following rule, which defines a background

Expanding the JavaScript

Go further with the script used in this tutorial

In the main example this issue, the JavaScript function is used to swap the portfolio image and its caption. However, this can easily be expanded. For example, the Githead site (www.githead.com) uses a similar script to also swap a photo credit and download link, which could be used to grab high-resolution artwork. If you load www.githead.com/githead.js, you'll see changes in the script's first line (adding 'theCredit,theLink'), and three additional lines, two of which deal with swapping out the credit content ('var displayedCredit...' and the following line), and one of which deals with amending the

'href' value of the download link (document.getElementById...). In this site's case, the link is to an image, and so the path to the relevant folder is built into the script.

On the web page, the credit is within a paragraph with an 'id' value of 'credit', while the download link's start tag has an 'id' value of 'downloadlink'. Amending the 'onclick' attribute in this issue's example page to include the two new content types would just be a case of adding another two values: onclick="javascript: swapPhoto('photo-1.jpg','Photo 1 caption','Photo credit','a-document-location.jpg'); return false;".



A great example The Githead website uses a similar script to the gallery in the tutorial, but adds a credit for each photograph and a link to download hi-res artwork

(a dark, greyscale version of one of the images, which fades towards the edges) and sets padding at the left of the div:

```
#thumbnails {
background: url(assets/background.jpg) no-repeat;
padding-left: 10px;
}
```

The containing divs for the thumbnails are styled using the rule below. Note that because the divs are all floated, they stack horizontally, instead of being displayed in a linear fashion. Because of this, these divs require a set height – if the heights were varied, subsequent divs could get stuck on any overly long ones, leading to a ragged and messy layout.

```
#thumbnails div {
width: 182px;
}
```

Subsequent divs could get stuck on any overly long ones, leading to a ragged and messy layout

```
>> height: 190px;
float: left;
margin-right: 10px;
}
```

To clear the floated content (to enable subsequent content to be positioned correctly, even though there's not actually any content under the thumbnails in the example), a separatorInvisible div is placed before the end tag of the thumbnails div. However, with the previous rule setting 'float' and 'height' values for all divs nested within the thumbnails div, overrides are required for the relevant separatorInvisible div, as shown in the following rule:

```
#thumbnails .separatorInvisible {
float: none;
height: 1px;
}
```

Styling headings

In the 'fonts' section, add the following two rules, which enable you to define font sizes in ems, using values that are a tenth of the target pixel size.

```
html {
font-size: 100%;
}
```

```
body {
font-size: 62.5%;
}
```

Add the next two rules, which style the level-one and level-two headings. Note the bottom border in the 'h1' rule, which makes it easy to differentiate the title from the page content.

```
h1 {
font: normal 1.5em/2em "Lucida Grande", "Lucida Sans Unicode", Verdana, Arial, sans-serif;
padding-left: 10px;
color: #e1e1e1;
border-bottom: 1px solid #7f7f7f;
text-transform: uppercase;
margin-bottom: 10px;
}
```

```
h2 {
font: normal 1.3em/1.4em "Lucida Grande", "Lucida Sans Unicode", Verdana, Arial, sans-serif;
padding-top: 10px;
}
```

Resources Find out more online



Floatutorial at Max Design offers a variety of useful starting points for producing layouts that use floated divs, including column-based layouts and list-based navigation bars. css.maxdesign.com.au/floatutorial/index.htm



There are a number of websites out there that explain how the box model works, but none do so quite as elegantly as Jon Hicks' 3D effort, available at the Hicksdesign website. www.hicksdesign.co.uk/boxmodel

The line-height setting of 1.6em is quite high, but this provides more space between each line

```
margin-bottom: 10px;
text-transform: uppercase;
color: #e1e1e1;
}
```

Earlier, a 'span' was shown, added to the level-two heading. This houses some basic instructions for users not familiar with online galleries. Although this should be clearly visible, it shouldn't be as prominent as the heading. Therefore, add the following rule, which darkens this block of text, renders it in lower case, and floats it right (with some right-hand padding, so it lines up with the thumbnails).

```
.instructions {
color: #aaaaaa;
text-transform: lowercase;
float: right;
padding-right: 12px;
}
```

Paragraphs and captions

Add the following rule to define the default style for paragraphs. The line-height setting of 1.6em is quite high, but this provides more space between each line of text, thereby making everything easier to read.

```
p {
font: 1.2em/1.6em "Lucida Grande", "Lucida Sans Unicode", Verdana, Arial, sans-serif;
margin-bottom: 0.8em;
}
```

By default, all paragraphs are left-aligned at this point, but the thumbnails look better with centrally aligned text. Also, the large 'line-height' value looks somewhat suspect on multi-line thumbnail text. Finally, there's no need for a 'margin-bottom' setting when the box heights have already been defined. Therefore, add the following rule to deal with all of these issues. (The 1.3em value for 'line-height' more or less returns the leading to its default value, which is a little low for reading large blocks of text onscreen, but it's fine for reading a few words under a thumbnail.)

```
#thumbnails div p {
margin-bottom: 0;
text-align: center;
line-height: 1.3em;
}
```

The final two rules to be added to the 'fonts' section deal with the caption. The first adds a 'margin-top' value so the caption paragraph doesn't hug the preceding separator div. The second, which has a selector '#caption:before', adds content prior to the caption – the word 'left' in grey. This makes it more obvious what the caption is referring to (although the separator line already makes the user visually aware of relationships between image and text); but sadly, IE7 doesn't render this content – boo, hiss! (Obviously, this is only nice to have, rather than something essential to the site. But with Microsoft's browser still commanding a huge market share, its idiosyncrasies must be taken into account when working on designs.)

```
#caption {
margin-top: 10px;
}
```

```
#caption:before {
content: 'left: ';
color: #aaaaaa;
}
```

Some styles are needed to deal with the links (to be housed in the 'links' section). The 'a' rule sets the default link colour to white, like the body copy (hence keeping the default underline, so links can be differentiated from static text). The 'a:hover, #thumbnails div a:hover' rule turns the link text green when a cursor hovers over it.

```
a {
color: #ffffff;
}

a:hover, #thumbnails div a:hover {
color: #14d873;
}
```

Styling the images

Images are the main content, so let's add some styles to the 'images' section of the CSS. The key element is image borders. For '#portfolioImage img' (which styles the main image, housed within the portfolioImage div) and '#thumbnails div img' (which styles the images within the divs nested in the thumbnails div), the borders are given solid outlines, a one-pixel thickness, and a dark grey colour. For the hover state, the 'border-color' value is changed to the same green used for the link hover state earlier ('#14d873'), via the '#thumbnails div a:hover img' rule.

```
#portfolioImage img {
border: 1px solid #353535;
margin-bottom: 8px;
}
```

```
#thumbnails div img {
border: 1px solid #353535;
margin-bottom: 3px;
}
```

```
#thumbnails div a:hover img {
border-color: #14d873;
}
```

Open 'ie-hacks.css' – a style sheet that will deal with IE6's weaknesses. In IE6, the top part of the design is wrecked, because it applies double margins to the right-hand edge of right-floated elements, and the relevant values are halved in the IE-specific style sheet. IE7 no longer has this bug, hence the conditional comment targeting IE6 and below. The two rules to add are shown here:

```
#introductoryText {
margin-right: 6px;
}
```

```
#portfolioImage {
margin-right: 10px;
}
```

And that's it! With the hacks file saved, the design will work in Opera, Firefox, Safari and the most recent two releases of Internet Explorer. If you want better compatibility with IE5.5, add another style sheet via a conditional comment, using 'lt IE6' in the opening comment, and add the following rules to centre the site.

```
body {text-align: center;}
#wrapper {text-align: left;}
```

On the CD, you'll find the folder 'floated boxes tutorial – complete', which contains finished versions of the files that you can use and abuse as you see fit! ●

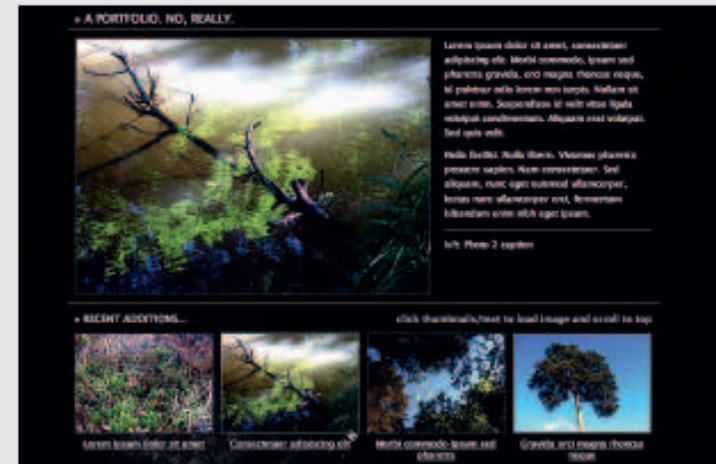


About the author

Name Craig Grannell
Site www.snubcommunications.com
Areas of expertise Information architecture, site concepts, graphics, interface and front-end design
Clients Swim~, Rebellion, IDG
Favourite 70s TV show *The Muppet Show*

The good, bad and ugly

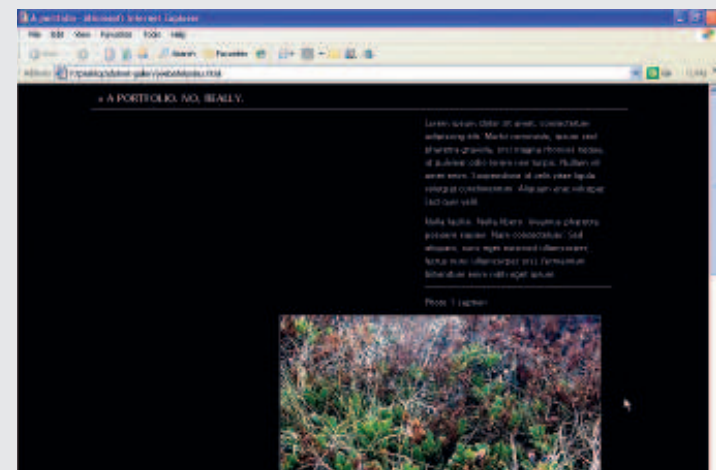
The final result – and two less impressive ones



One we made earlier The completed web page, with floated boxes housing the thumbnails and captions. Note the vibrant thumbnail border colour on the hover state, making it all the more obvious that the thumbnail is a clickable element



A text disaster Take a terrifying trip to the past, back to a time when fonts weren't styled and text colours were never changed. So, yes, here's what the page looks like before text, images and links have been styled. Ugh



Without the hacks file Internet Explorer 6 – how we're going to (not at all) miss you when you're dead, buried, exhumed, kicked, burnt, spat on, and buried again. Because of one of its many bugs, here's how the site looks without the hacks file. Nice, eh?