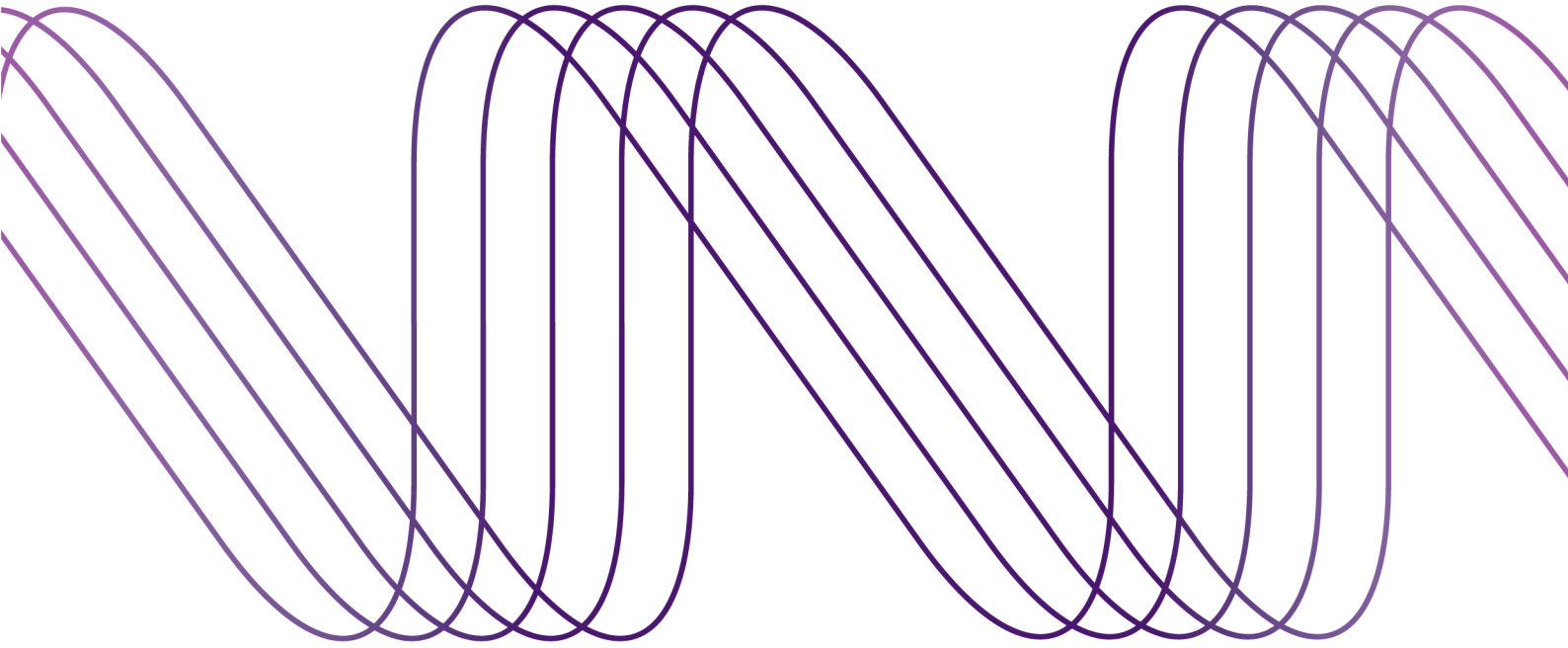
T: 0191 222 5000
E: enquiries@netskills.ac.uk
www.netskills.ac.uk

JISC netskills

The CSS Box Model

Practical Exercises



Overview

The CSS Box Model is fundamental to understanding how to use the range of page layout tools that CSS has to offer. By treating all page elements as boxes with dimensions, borders, padding, margins and position, CSS allows the page to be broken down into layout components which may be independent of the content they contain.

Information

At the end of these tasks you should be able to:

- Apply borders, margins and padding to elements in an HTML page
- Centre content in a web page using CSS
- Use box properties to format other HTML element

What you will need to use these materials

You only need a text-file editor and a current web browser to start adding CSS to XHTML documents.

The exact type(s) and version(s) available to you will depend on your local set up. Common operating system default editors are Notepad (Windows) and TextEdit (MacOSX) and Gedit (Linux).

Suitable web browsers include Mozilla Firefox, Google Chrome, Apple Safari and Microsoft Internet Explorer.

Contents

1. Getting Started

The documents you are about download already have some CSS applied. You can find details of this in *Introducing CSS*. Even if you have a complete set of files from those exercises, we recommend downloading this fresh set for these tasks.

Task 1-1 Create a new folder in the local file area, called `boxes`

Task 1-2 In your web browser (Internet Explorer, Firefox etc.) go to the following location:

http://www.netskills.ac.uk/resources/css/boxes/

You should see the following list of files:

```
format.css
about.html
on-site.html
portfolio.html
netskills.png
```

Right-click on the link to `about.html` and select **Save Target As...** from the pop-up menu (or **Save Link As...** if you are using Firefox) and save the file to the folder you just created.

Repeat this for all files in the list.

Task 1-3 Open your local copy of `about.html` in your web browser.

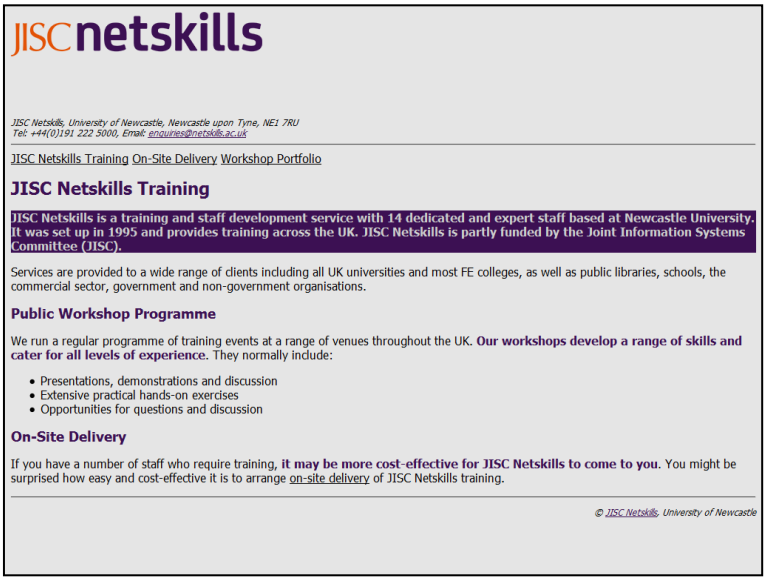


Figure 1-1. `about.html`

Take a moment to familiarise yourself with the layout and content in the page before using the links at the top of the page to browse to the other parts in the example site.

Figure 1-2. Example site links

External links Some of the links on these pages may take you away from the example site you will be working with. If you follow such a link you may get a page warning you that you are about to leave the example web site. Just use your browser's back button to return.

Task 1-4 Open `about.html` in your text editor and take a look at the source code it contains.

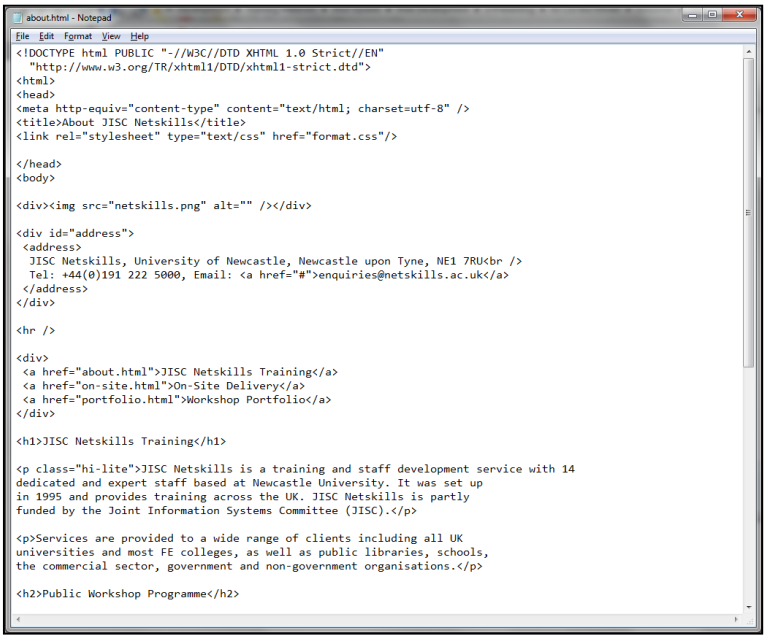


Figure 1-3. `about.html` in Windows Notepad

You should find that there are no specific formatting or layout instructions – just plain XHTML tags and attributes.

There is however, a link to an external style sheet called `format.css`

```
<link rel="stylesheet" type="text/css" href="format.css"/>
```

This style sheet supplies the basic rules that currently dictate the appearance of the pages in the site. This style sheet is examined in more detail as part of the Netskills materials *Introducing CSS*.

If you do wish to look at these rules more closely then you can of course open `format.css` in your text editor – but do not change any of the rules in this file as they are needed for the subsequent tasks in this section!

Important Note If you take look at the source code for the other HTML files you will notice that they appear to have a number of different tags and attributes that are not in `about.html`.

Don't panic! – You will be editing `about.html` to add some of this code yourself – we have just saved you the effort of having to add the same code to 3 files each time!

2. The CSS Box Model

In CSS every element in the page can be treated as a box with its own borders, margins, padding and height and width dimensions. This box model is not restricted to block-level elements and is crucial to the application of all CSS layout and positioning rules.

Task 2-1

Open `about.html` in your text editor and add a new `<div>` (with an `id` of `content`) to enclose *all* of the `<body>` content.

```
<body>
<div id="content">
<div id="logo"><img src="netskills.png" alt="" /></div>
...
etc
...
</div>
</body>
</html>
```

Save `about.html`

Task 2-2

Open a new file in your text editor and add the following CSS rule:

```
#content {border-style: solid;}
```

Save this new file as `boxes.css`

Now return to `about.html` and add a link to your new CSS style sheet as shown below:

```
<head>
<meta http-equiv="content-type"
  content="text/html; charset=iso-8859-1" />
<title>About Netskills</title>

<link rel="stylesheet" type="text/css" href="format.css"/>
<link rel="stylesheet" type="text/css" href="boxes.css"/>

</head>
```

Save `about.html` and reload it in your browser.

You should now see the outline of the new content `<div>` that you created in the task above.

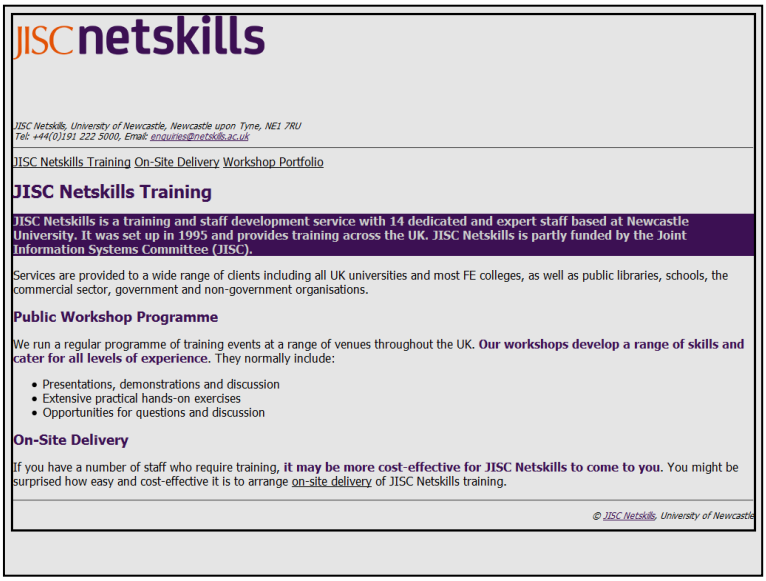


Figure 2-1. CSS box border applied

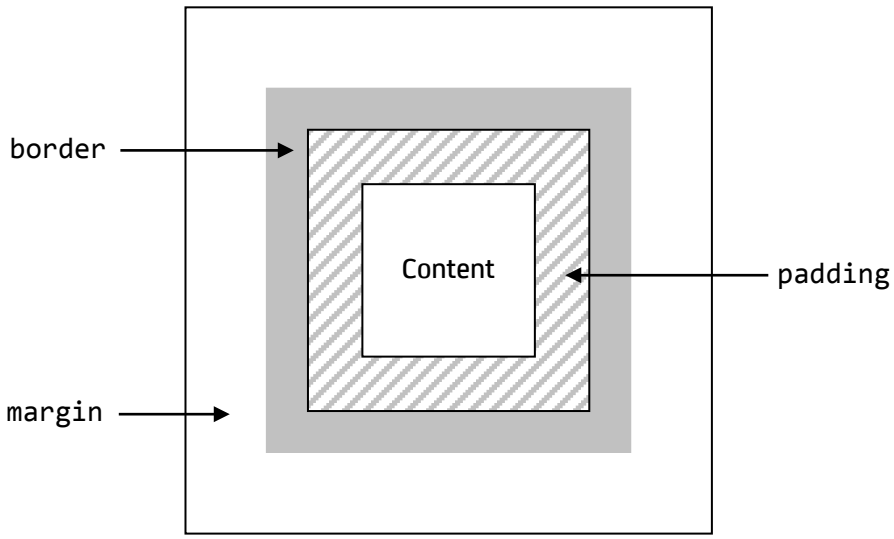
If you browse through the site you should find the same effect on all the pages.

What happened?

The CSS rule you created gave the browser instructions to change one of the box properties of `#content` (the `id` of the `<div>` you added to the page). In this case it has applied a solid border around the whole box using `border-style: solid;`

The CSS Box Model

Using CSS, every element in the page has an associated box that can be manipulated with CSS style rules. The key properties are shown below.



Property	Defines...
`margin`	Distance between the current box and those around it.
`padding`	Distance between the content in the box and the inner edge of its border.
`border`	The size and style of the border of an individual box.

The defaults for each property are for them to hold no value – you have to set the ones you wish to use.

Property Syntax

Each box property can be applied equally on all four sides of the box or individually to each side. Properties set on one box are not passed on to any boxes nested inside it.

There are several ways to specify box properties. Examples of some of the most common for border are shown below.

Declaration	Applies...
`border-style: solid;` `border-width: thin;`	A single value for each property applied to all four sides simultaneously
`border-left-width: 20px;`	Explicit value for one property – for one side
`border-width: 15px 50px;`	Set top/bottom borders to 15px and left/right borders to 50px
`border-width: 15px 5px 30px 50px;`	Set four different values in one rule Order is top – right – bottom – left
`border: thin solid yellow;`	Set single value for all three properties – for all four sides as width – style – colour.

This is not an exhaustive list of syntax and shorthand! You will see some of these (and others) in use throughout these tasks.

Margins and padding are slightly easier to control as they are just single properties applied to top, right, bottom and left. e.g.

```
margin-left: 5%;
padding-left: 10px;
```

A useful reference for the box property syntax can be found at:

http://www.devguru.com/Technologies/css/quickref/ css_border.html

http://www.devguru.com/Technologies/css/quickref/ css_margin.html

http://www.devguru.com/Technologies/css/quickref/ css_padding.html

3. Margins

The CSS margin properties dictate the spacing between elements on a web page. They are typically used to offset content to create an indented or centred effect.

Task 3-1 Add a rule to `boxes.css` to apply a left margin to the `#content` box.

```
#content {border-style: solid;
          margin-left: 15%;}
```

Save `boxes.css` and return to your browser to see the effect this new rule has.

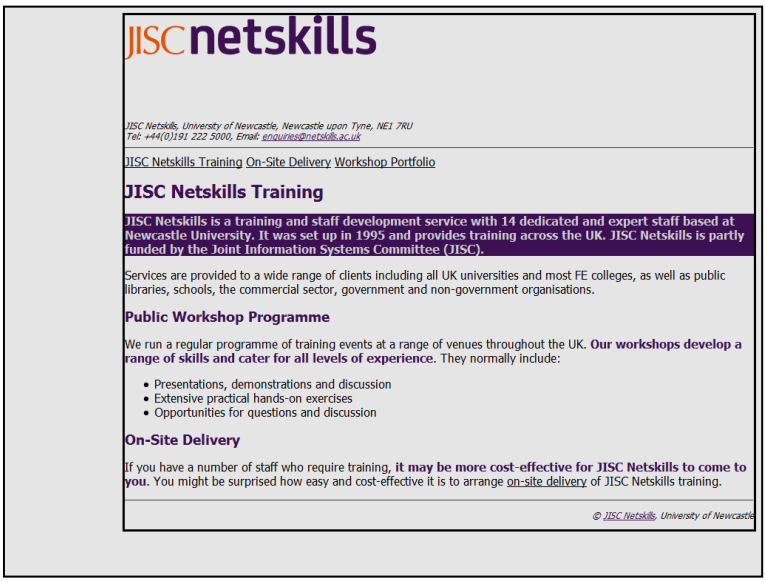


Figure 3-1. Left margin rule applied

Task 3-2 Return to your text editor and increase the margin to **45%** and add right-hand margin:

```
#content {border-style: solid;
          margin-left: 45%;
          margin-right: 15%;}
```

Notice how the browser has wrapped content to allow it to remain visible.

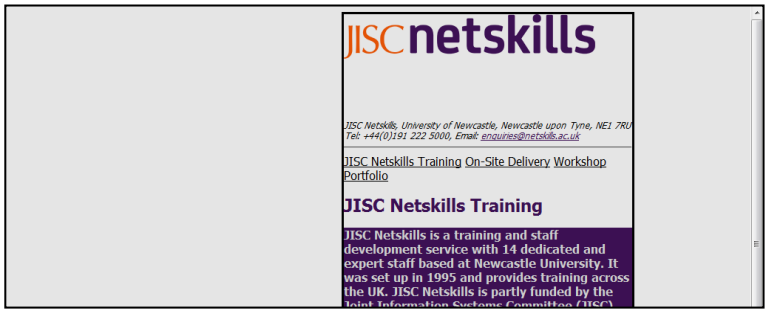


Figure 3-2. Setting left and right margins

Task 3-3 Try setting top and bottom margins as shown below:

```
#content {border-style: solid;
          margin-left: 45%;
          margin-right: 15%;
          margin-top: 15%;
          margin-bottom: 15%;}
```

Save and reload.

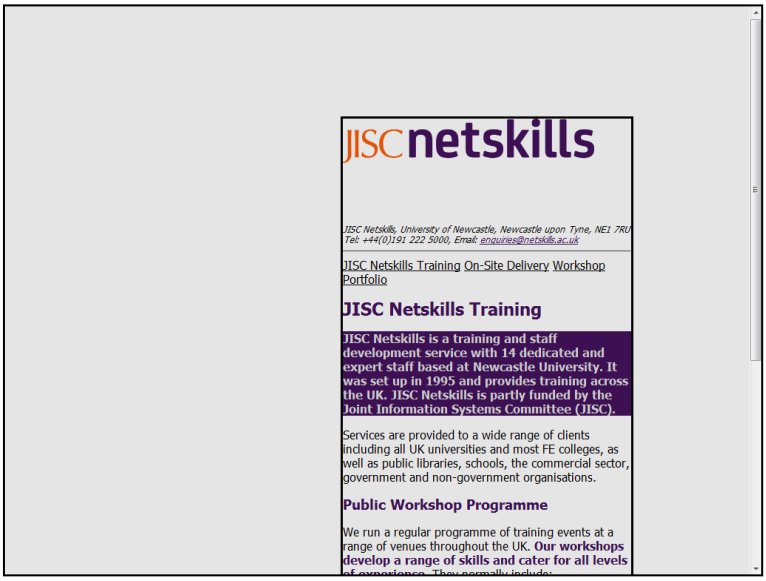


Figure 3-3. Adding top and bottom margins

Note Microsoft Internet Explorer doesn't consistently support the `margin-bottom` property. If it is set it ignores it! But it is there so users with other, browsers will be able to see the bottom margin.

If you have access to a copy of an alternative browser such as Firefox you will be able to see the bottom margin being applied.

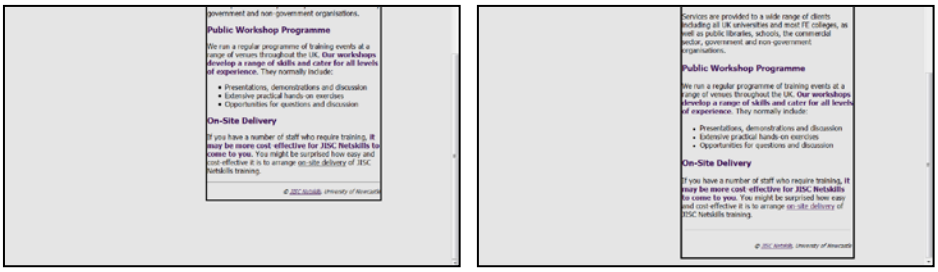


Figure 3-4. Bottom margin in Firefox (right) and IE (left)

4. Width and Centring Content

As you will see in these tasks there is a very elegant way of centring page content using CSS margins.

Task 4-1

Open `boxes.css` in your text editor and convert the individual margin properties set for `#content` in to a shorthand declaration as shown below:

```
#content {border-style: solid;
          margin: 0% 15%;}
```

Save this change and reload your web page in the browser. You should now find that the top and bottom margins now have no margin set and left and right margins are set to 15%.

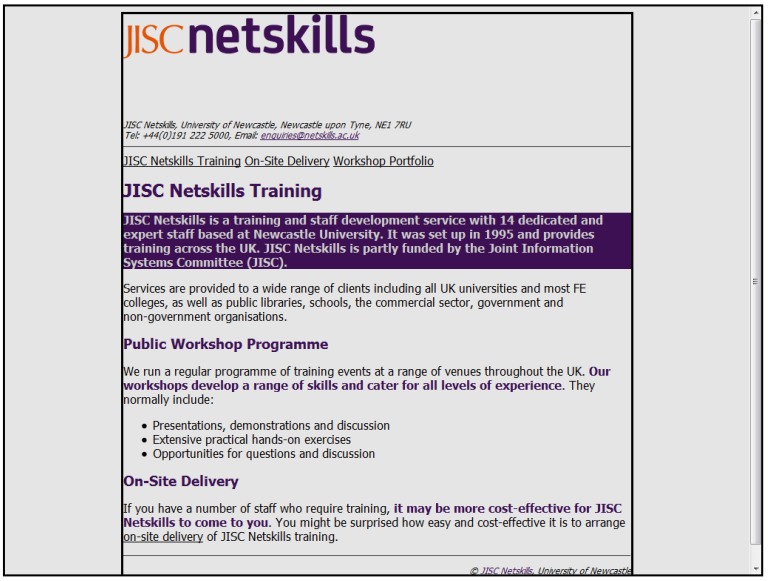


Figure 4-1. Centred content?

This appears to have the effect of centring the content in the page, but for most web authors this is not the most satisfactory way to achieve this effect.

If you try resizing your browser window you should find that the size of the content block changes to allow the browser to maintain the margins at 15%. In reality most authors would like the *margins to change* to keep the content block in the middle of the window.

Task 4-2

Return to `boxes.css` and add the following:

```
#content {border-style: solid;
          margin: 0% 15%;
          text-align: center;}
```

Save the style sheet.

Now take a look at your web pages – does this solve the problem?

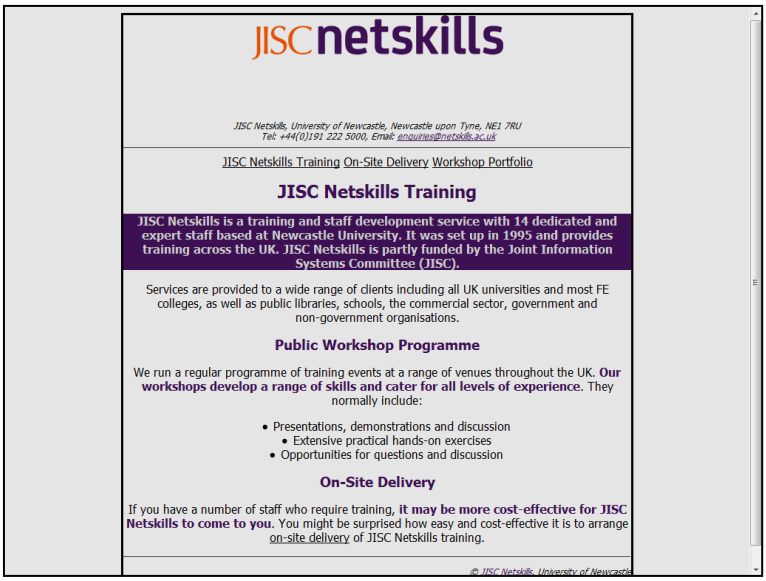


Figure 4-2. Centred content?

This is an easy trap to fall into – `text-align: center;` looks like it should do the job – sadly the only thing that it applies to is the *contents* of a box – not the box itself.

Task 4-3 Remove the `text: align` declaration from your style sheet and replace it with a dimension for the width of the box.

```
#content {border-style: solid;
          margin: 0% 15%;
          width: 70%;}
```

This should appear to have the same effect as Task 4-1 so what difference has the `width` made?

Consider what happens if you want the width to be fixed at a smaller size. At the minute, all three dimensions you have added make up 100% of the available width.

`margin-left:15%`	`width:70%`	`margin-right:15%`

Task 4-4 Change the width dimension for `#content` to 45%.

```
#content {border-style: solid;
          margin: 0% 15%;
          width: 45%;}
```

Save and reload.

You should notice that the only margin the browser can set accurately is the left-hand one.

Figure 4-3. Off-centre content

As the page is processed the left margin is set, then the `#content` is given 45% of whatever is left.

That leaves *more than* 15% on the right-hand side but the browser cannot pull `#content` over any further (as it has a specific width set). The end result is some off-centre content!

The standards, compliant solution involves specifying left and right margins – but *allowing the browser to calculate them*.

Task 4-5 Now set the dimensions for the margins of `#content` as follows:

```
#content {border-style: solid;
          margin: 0% auto;
          width: 45%;}
```

Save and reload.

Hopefully you should now be seeing some bona fide CSS centred content!

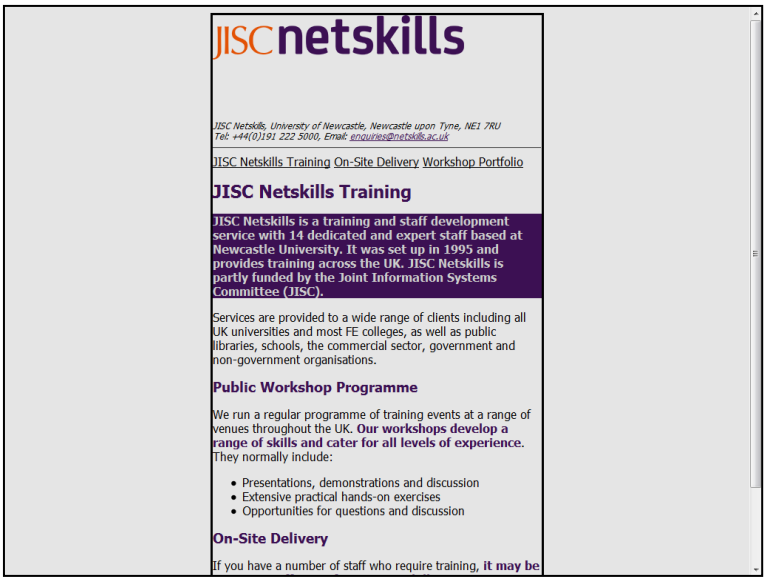


Figure 4-4. Centred content!

Now only `#content` has a specified `width` and the browser allocates that first.

The left and right margins for `#content` are set to `auto` – this allows each to occupy the maximum remaining space available on their respective sides of the box.

The end result is that `#content` always appears in the centre of the page!

Relative or absolute units?

At this point the width of the content block is specified using a *relative* dimension – a percentage (%). This means that the browser is still allowed to resize the content to fit the browser window (try it and see).

As your CSS skills improve you will find that it is possible to create truly fluid CSS layouts only using relative units but it often seems easier to get started with a few fixed dimensions using *absolute* units such as pixels (px).

Task 4-6

Finally, set the width of the `#content` block to `70%`

```
#content {border-style: solid;
          margin: 0% auto;
          width: 70%;}
```

Save `boxes.css` and move on to the next tasks.

5. Borders

The border properties can seem a little complex at first. There are three that work together to control the overall effect `border-width`, `border-style` and `border-color`.

Task 5-1

Open `boxes.css` in your text editor and look at the declaration for the `border` of `#content`.

```
#content {border-style: solid;
          margin: 0% auto;
          width: 70%;}
```

This currently uses shorthand to set all four borders to the same style.

See the notes at the end Task 2 of for more information on CSS shorthand.

Other possible values for `border-style` include:

`none` | `solid` | `dashed` | `dotted` | `double` | `groove` | `ridge` | `inset` | `outset`

Try some of these for yourself to see the effect if you wish.

Figure 5-1. border-style examples

When you have finished set the `border-style` back to `solid` before continuing.

Task 5-2

Now add a declaration for the `border-width`.

```
#content {border-style: solid;
          border-width: thin;
          margin: 0% auto;
          width: 70%;}
```

Save and reload. You should find that the border has been reduced in size.

Units for border-width

There are a small set of relative, pre-set values you can use for `border-width`. The browser determines the final thickness but they are useful for quick and easy formatting. The values you can use are `thin`, `medium` (the default) and `thick`.

If you want precise control you can set a specific pixel width e.g.

```
border-width: 20px;
```

Task 5-3 Finally set the `border-color` to fit in with the rest of the colour scheme.

```
#content {border-style: solid;
          border-width: thin;
          border-color: #3C1053;
          margin: 0% auto;
          width: 70%;}
```

Save `boxes.css` and check that all the rules are being applied.

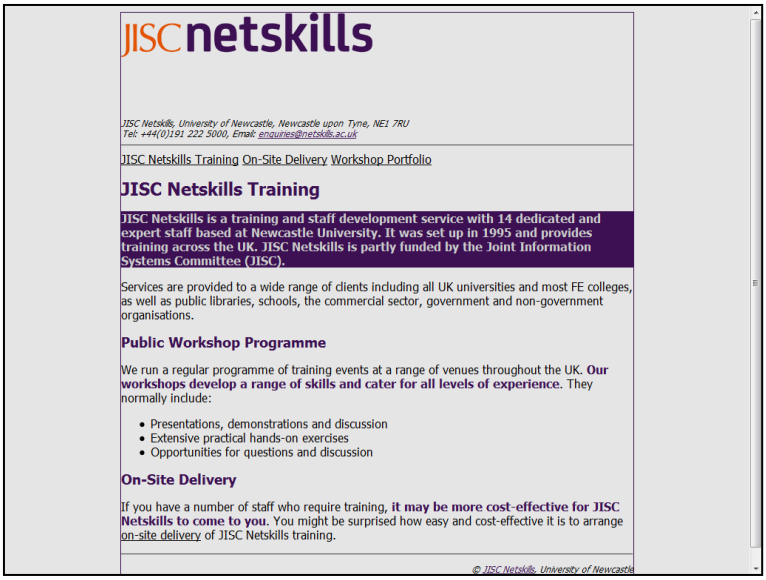


Figure 5-2. `about.html` with border properties applied

6. Padding

The CSS property padding controls the distance between the content of an XHTML element and the inside edge of its border.

You do not have to have a border applied to use padding.

Task 6-1

Add a declaration to `boxes.css` to apply some padding to `#content` as shown below:

```
#content {border-style: solid;
          border-width: thin;
          border-color: #3C1053;
          margin: 0% auto;
          width: 70%;
          padding: 0.5em;}
```

Save your style sheet and reload your web pages. You should see that the spacing between the content and the border increases.

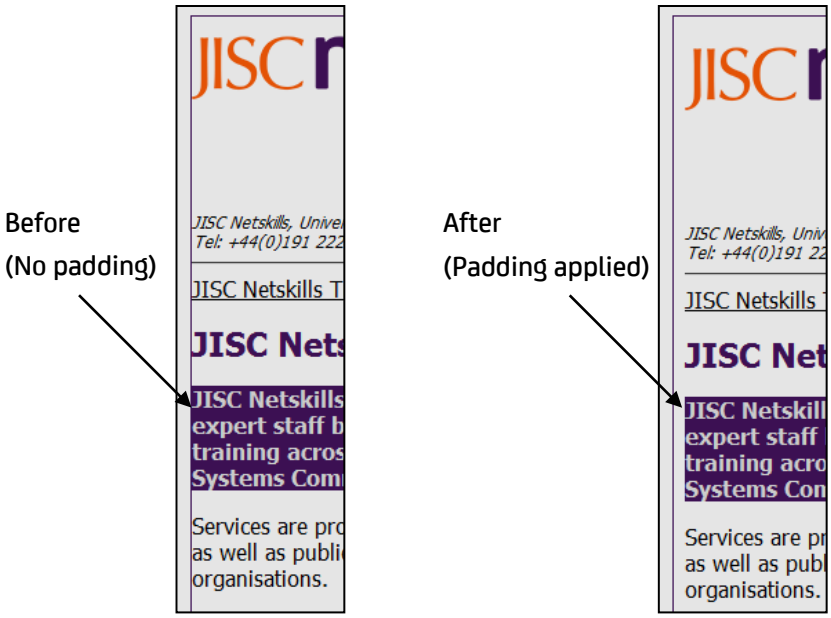


Figure 6-1. Adding internal padding

The `padding` property sets the distance between the content of a CSS box and its border.

Like the other box properties it can be applied to all four sides of the box and you can use the shorthand notation to specify padding for more than one side at a time. Every box is entitled to its own set of padding dimensions

Task 6-2 Add the new rule shown below to your style sheet, `boxes.css`

```
#content {border-style: solid;
          border-width: thin;
          border-color: #3C1053;
          margin: 0% auto;
          width: 70%;
          padding: 0.5em;}


p.hi-lite {padding: 0.5em;}
```

This will apply padding to all four sides of the opening paragraph of each page. (They are the paragraphs in the `hi-lite` class)

Save `boxes.css` and reload your web page(s) in your browser see the effect.

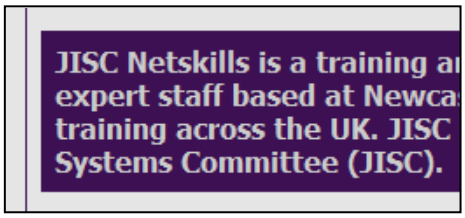


Figure 6-2. Applying padding to `p.hi-lite`

Task 6-3 Finally add a shorthand rule to `boxes.css` to apply only left and right padding to all the remaining paragraphs.

```
p {padding: 0 0.5em;}
```

This should have the effect of lining all the paragraph text back up with the `hi-lite` paragraphs.

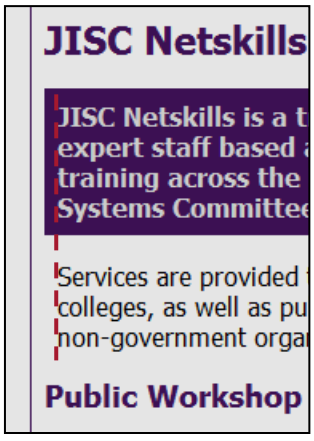


Figure 6-3. Re-aligning paragraphs

7. More with Boxes

The CSS box model is very versatile so think creatively! Once you have got used to the idea that you can apply a huge range of box properties to anything in a web page, you can start using CSS to create more interesting effects.

Task 7-1

Open `about.html` in a web browser and look at the navigation bar at the top of the page.

Figure 7-1. Navigation bar

Now open `about.html` in your text editor and take a look at the XHTML that makes up the menu. The code you are looking for is near the top of the document and is shown in bold below.

```
<hr />

<div>
 <a href="about.html">Netskills Training</a>
 <a href="on-site.html">On-Site Delivery</a>
 <a href="portfolio.html">Workshop Portfolio</a>
 <a href="tonic.html">TONIC</a>
</div>

<h1>Netskills Training</h1>
```

Add the `id` shown below to the `<div>`:

```
<hr />

<div id="nav-bar">
  <a href="about.html">Netskills Training</a>...
```

Save `about.html`

Task 7-2

Open `boxes.css` in your text editor.

Add the following, new, CSS rule to `boxes.css`

```
#nav-bar a {border-bottom: solid thin;
            border-right: solid thin;
            border-color: #3C1053;
            padding: 0.1em 0.1em;
            font-weight: bold;
            text-decoration: none;}
```

Save `boxes.css`

Notes

The rule you have just added sets CSS box properties for the selector:

```
#nav-bar a
```

This will match `<a>` elements (links) but *only* if they occur inside the `#nav-bar` element (the `<div>` containing the navigation links). That means that these declarations will not be applied to the other links in the page – just the ones in the navigation bar.

Task 7-3

Reload `about.html` in your browser to see the effect of the rule you have just added.

Figure 7-2. CSS formatted navigation bar

Notice how the appearance of the navigation menu has been altered by using CSS box properties.

text-decoration

This rule also introduces the CSS `text-decoration` property.

Other possible values for text-decoration include `underline`, `overline` and `line-through`. The default value is `none`.

In this example `none` is used to "turn off" the default underlined appearance of a hyperlink. This is a common technique you will see used on many web pages; although care should be taken as many users expect to see hyperlinks underlined!

Appendix 1. boxes.css

At the end of these tasks you should have a series of web pages that all get their formatting instructions from two external CSS style sheets.

One of them, `format.css` was pre-prepared. The second style sheet, called `boxes.css` has been built up in stages over the preceding tasks. If you have completed all the tasks `boxes.css` should look like this:

```
#content {border-style: solid;
          border-width: thin;
          border-color: #3C1053;
          margin: 0% auto;
          width: 70%;
          padding: 0.5em;}

p.hi-lite {padding: 0.5em;}

p {padding: 0 0.5em;}

#nav-bar a {border-bottom: solid thin;
            border-right: solid thin;
            border-color: #3C1053;
            padding: 0.1em 0.1em;
            font-weight: bold;
            text-decoration: none;}
```

Appendix 2. about.html

At the end of these tasks `about.html` should look similar to the screenshot below: