Appendix 13: Cascading Style Sheets (CSS) — controlling appearance and positioning (WDD)

The following CSS declarations control the appearance and position of HTML page elements:

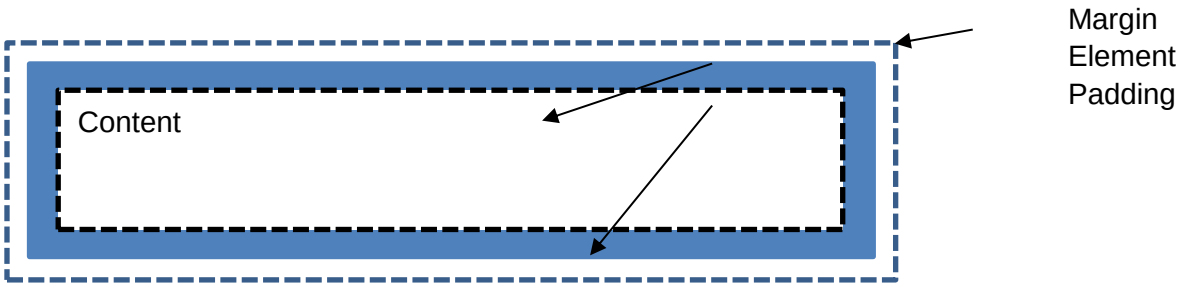
- display — block, inline, none
- float — left, right
- clear — both
- margins/padding
- sizes — height, width

The following examples have been taken from the *Higher Computing Science example website*. The example website can be downloaded as a zip file from the Higher Computing Science page on SQA's website. To view the full code in context, open and view the source code from the pages noted.

The box model

Margins and padding are used to push content away from the outer and inner edge of elements.

The box model allows us to define the space between elements.



Margin: declares a transparent area around the outside of an element. This pushes the element away from other adjacent elements.

Padding: declares a transparent area inside the edge of the element. This pushes content in from the edge of the element.

Universal selector

Browsers use default settings for margins and padding when displaying HTML elements. To override these defaults, a universal selector (*) can be used at the top of a stylesheet to set the margin and padding of every element to 0.

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

Using the universal selector this way allows candidates to witness only the margins and paddings they actually code.

Note: universal selectors are not included in Higher content but can be useful when teaching.

Margins (all pages): main page areas
Margins can be declared as:

- margin
- margin-top
- margin-bottom
- margin-left
- margin-right

These properties may have the values:

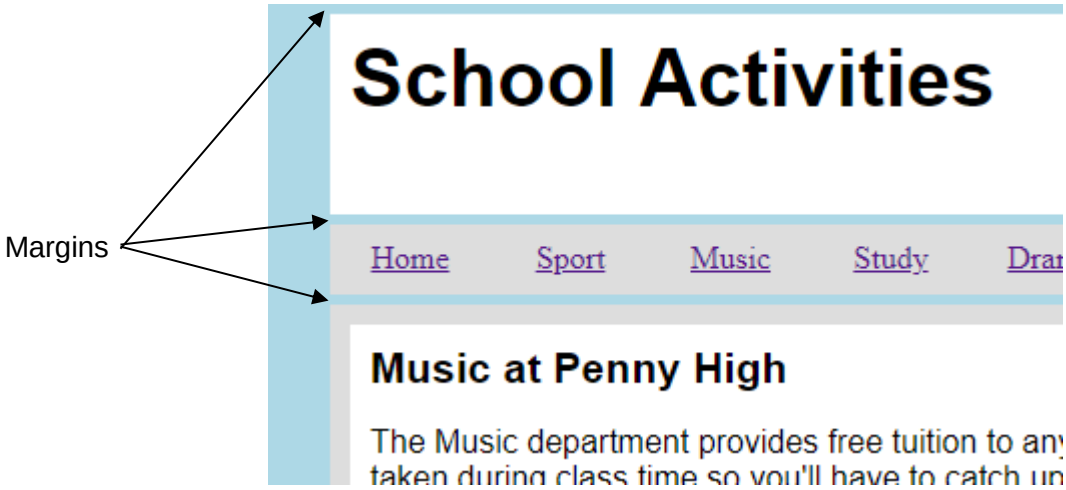
- auto (calculated by the browser)
- length (usually in pixels)

Margins can be declared on all four sides of an element simultaneously using the abbreviation: `margin:10px` .

In the example website, CSS declarations for the main page elements (<header>, <nav>, and <footer>) are used to separate out content when displayed.

The small gap at the top of each area is implemented by styling a margin of 5 pixels at the top of the four elements. For example:

```
header, nav, main, footer {margin-top:5px}
```



Grouping selectors using commas reduces the amount of code required and is more efficient than writing each CSS declaration separately.

Separate declarations

```
header {margin-top:5px}
nav {margin-top:5px}
main {margin-top:5px}
footer {margin-top:5px}
```

Grouped declaration

```
header, nav, main, footer {margin-top:5px}
```

Auto margins are used to position an element in the middle of the browser's window or within another element. In the example website, the 800 pixel wide page <body> element is always positioned in the middle of the window:

```
body{margin:auto}
```

Padding (drama page): sub-division of content
Padding can be declared as:

- padding
- padding-top
- padding-bottom
- padding-left
- padding-right

These properties may have the value:

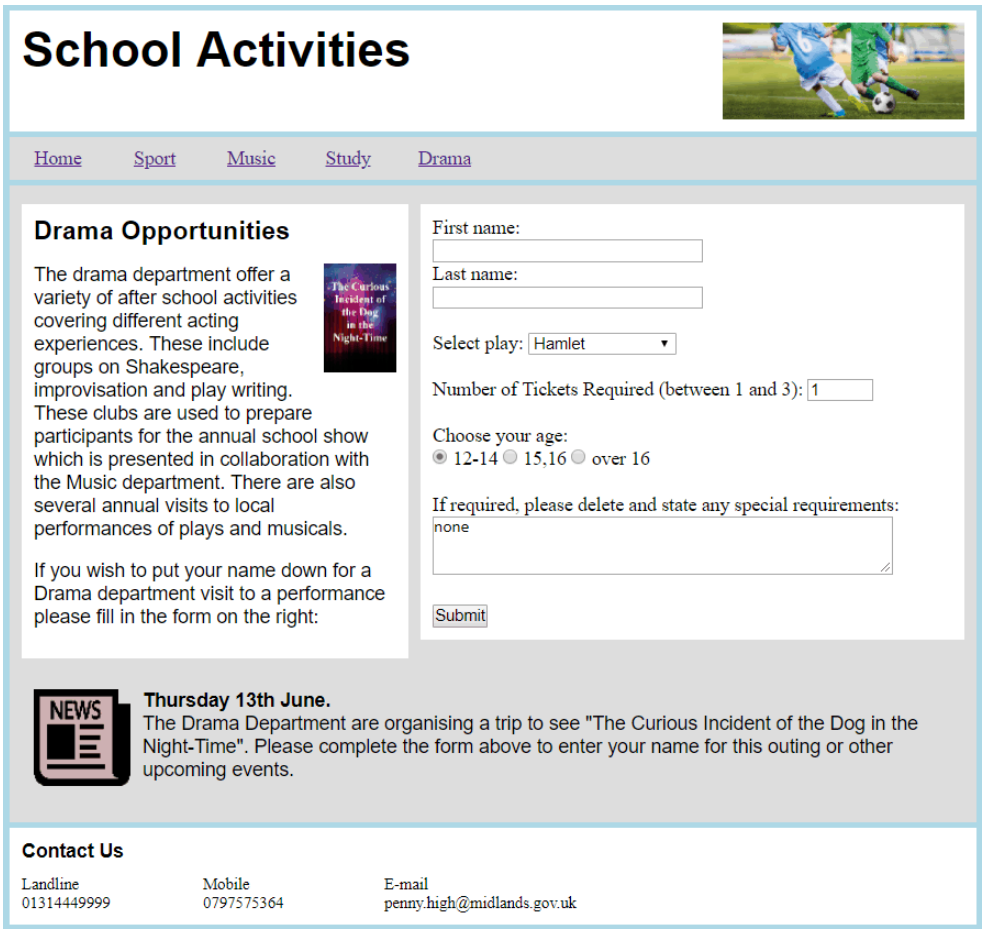
- length (usually in pixels)

Padding can be declared on all four sides of an element simultaneously using the abbreviation: `padding:10px`

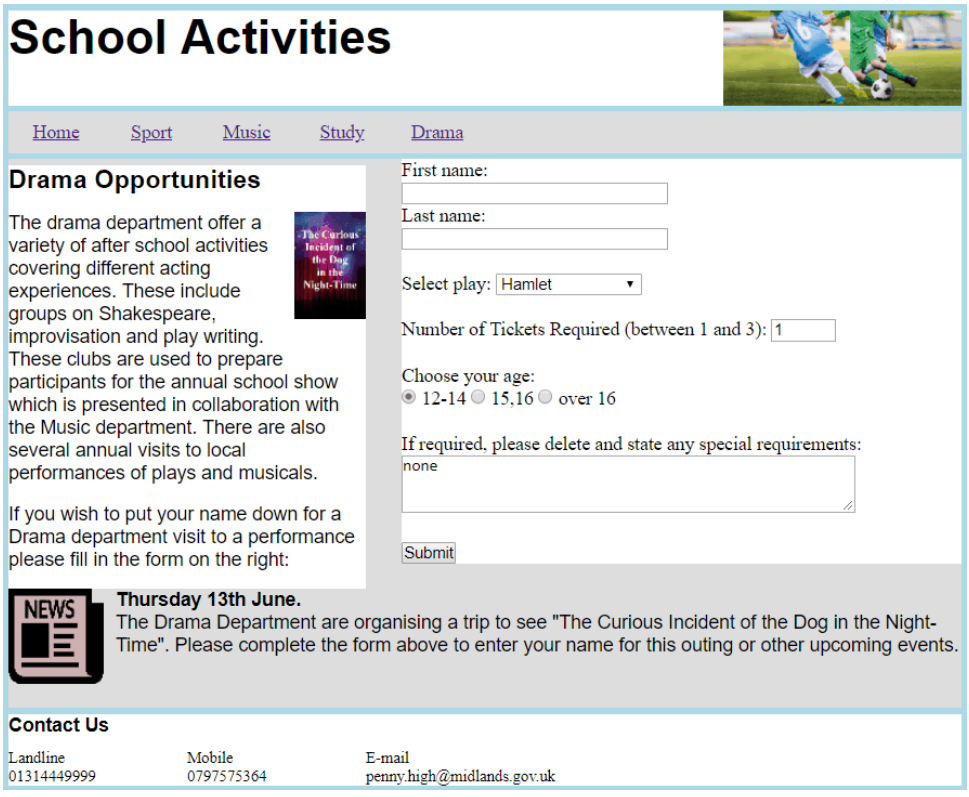
White space around page content, aids the usability of a web page. Appropriate use of white space can also aid readability of text.

Using padding to move content in from the edge of an element is one method to generate white space.

With padding: `header, div, main, footer, section {padding:10px}`



Without padding: `header, div, main, footer, section {padding:0px}`



Sizes — height/width (all pages)
Web page design and implementation involves creating areas that either have a fixed size or change size with content or display (changing window size or resolution).

In the example website, the <body> of each web page is set to a fixed width of 800px:

```
body{width:800px}
```

The height of the <header>, <nav> and <footer> elements are all set to a fixed size, remaining constant throughout the website:

```
header {height:80px}
footer {height:60px}
nav {height:35px}
```

The <main> element, which holds content of the pages, is not declared as a fixed size, as it changes according to the differing amount of content on each page.

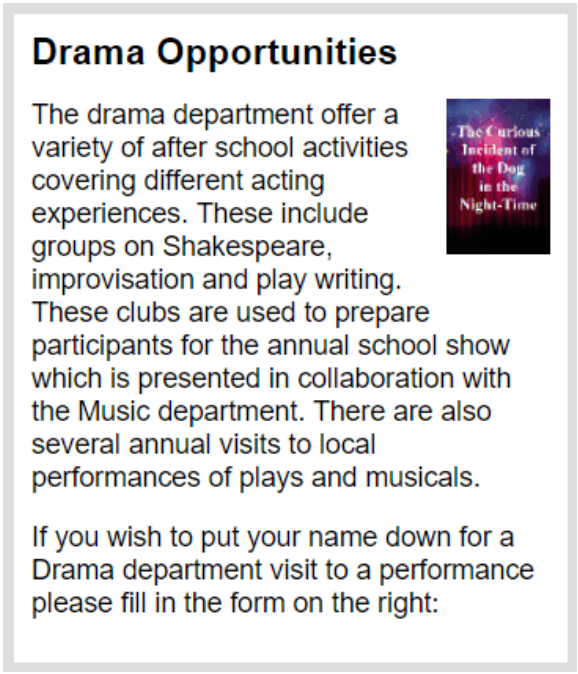
Float — left/right (all pages)
An element can be positioned on the left or right of its container, using the float property.

In the example website, the <img> element has been positioned on the right of its container, the <header> element:

HTML
<header>
<h1>School Activities</h1>
<img class="imageBanner" src="../images/activitiesBanner.jpg">
</header>

CSS
.imageBanner {width:200px;height:80px;float:right}

Float can also be used to word-wrap text around an image, as demonstrated in the drama page below. The addition of margins creates white space between the graphic and the text.



HTML

```
<p> <img class="imageIconRight"
src="../images/curiousIncident.jpg">
The drama department offer a variety of after school
activities covering different acting experiences;
Shakespeare, improvisation, play writing.  These clubs are
used to prepare participants for the annual school show
which is presented in collaboration with the Music
department.  There are also several annual visits to local
performances.</p><p>If you wish to put your name down for a
Drama department visit to a performance please fill in the
form:</p>
```

CSS

```
.imageIconRight {width:60px;height:90px;float:right;margin-
left:10px;margin-bottom:10px}
```

Clear (all pages):
The effect of floating elements continues until cancelled, using the clear property on a subsequent element.

To ensure that the four main page elements <header>, <nav>, <main> and <footer> start a new line and remain unaffected by any float properties applied elsewhere in the page, `clear:both` was implemented for these elements.

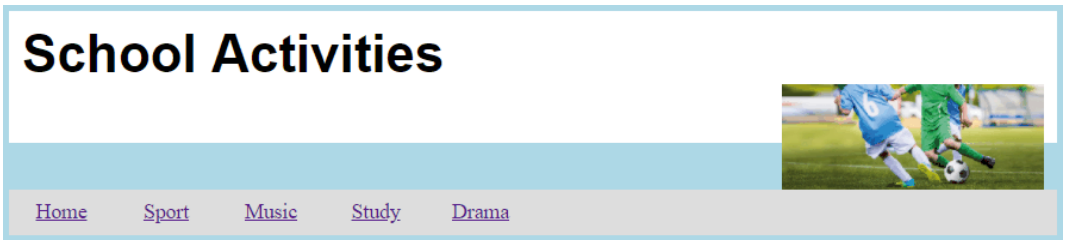
```
header, nav, main, footer {display:block;clear:both}
```

Display — block/inline/none (all pages):
HTML elements have default display values depending on the type of element. This value specifies how or if an element is to be displayed. The default display value for most elements is block or inline:

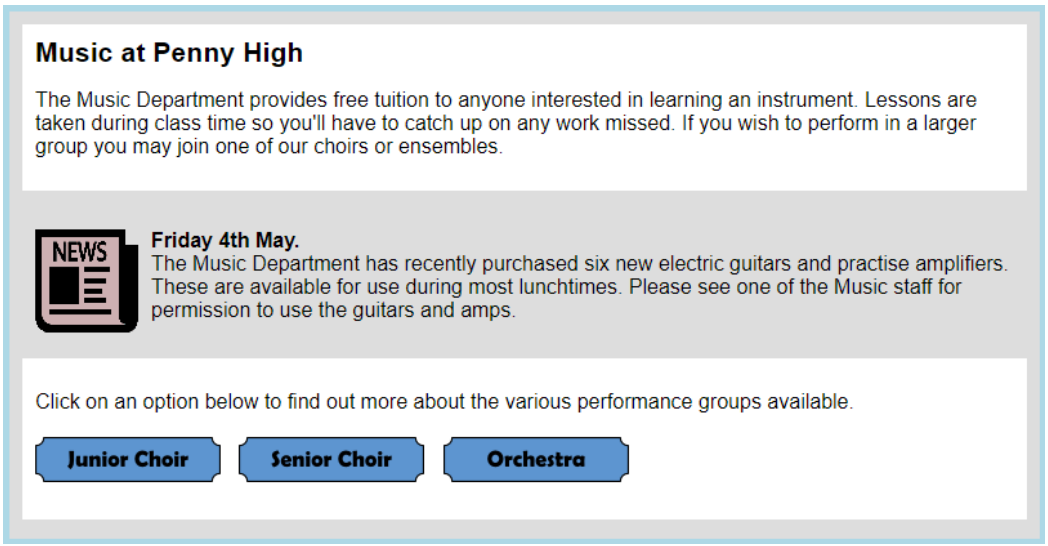
`display:block` — an element takes up the entire width of its container
`display:inline` — an element takes up only as much room as necessary
`display:none` — the element is not visible. The space where the element should be collapses as if there was no content in that place.

In the <header> element of the example website, the <h1> heading uses `display:inline`. This allows the image to be floated level with the heading.

If the <h1> element used `display:block` it would force the image to float on a new line.



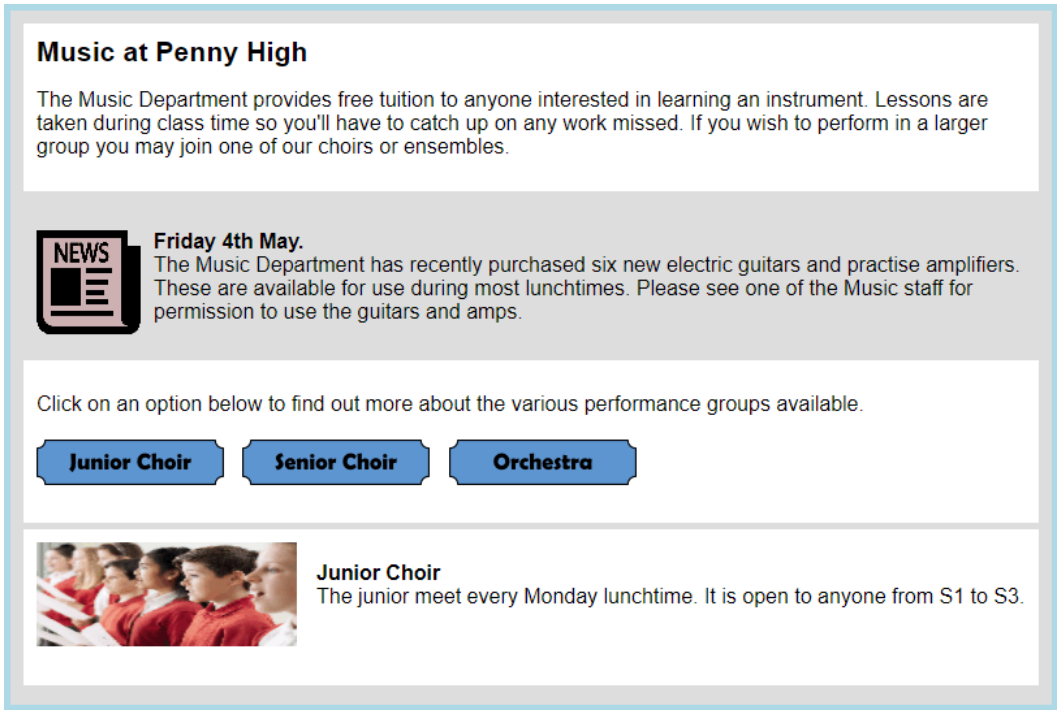
In the music page, `display:none` is used to hide three <section> elements when the page loads.



HTML

```
<section id="junior" style="display:none;height:100px">
<img class="largeImages" src="../images/juniorChoir.jpg">
<p><b>Junior Choir</b><br>The junior meet every Monday
lunchtime.  It is open to anyone from S1 to S3.</p>
</section>
```

The above section is revealed when the 'Junior Choir' image on the page is clicked. This is achieved using a JavaScript onclick event to change the display property of the <section> to block.



For more details, see appendix 17: JavaScript.