
success
in life
is just a

tag
away
©WORDS & UNWORDS

Web Technology

Styling Your Website

1495 UNIVERSITY OF ABERDEEN

computing@aberdeen

Alright, I know the basics.
Show me something complicated already!

Alright, you asked for it…

Question

- Which of the following is NOT valid CSS:
  - A) .mycls {background-color: blue; font-size:12;}
  - B) a {color: #123456;}
  - C) h4 [color:white; font-weight: bold;]
  - D) div {border-width: 10px; padding-left: 10px; }

What we'll cover

- Applying CSS to entire HTML pages
- Divs and CSS
  - The Box Model
  - Padding, Margins and Borders
  - Positioning

So what we've learned is…

- CSS provides us with a modular framework for styling our websites.
- CSS allows us to separate style from content.
- We can create assign styles to particular HTML tags, classes and IDs.
- We can apply CSS in a multitude of ways (Externally, embedding and inline).


CSS Workshop
RIVETS
SHARPENING
SOCKETS

Styling the “Body”

- Like all HTML tags, CSS can be applied to it. The <body> tag is no exception.
- Here, we can specify the background colour or image of a webpage, the fonts that will be used and other general features.
- Styling the Body tag is the first stage in creating a template for your website.

An example…

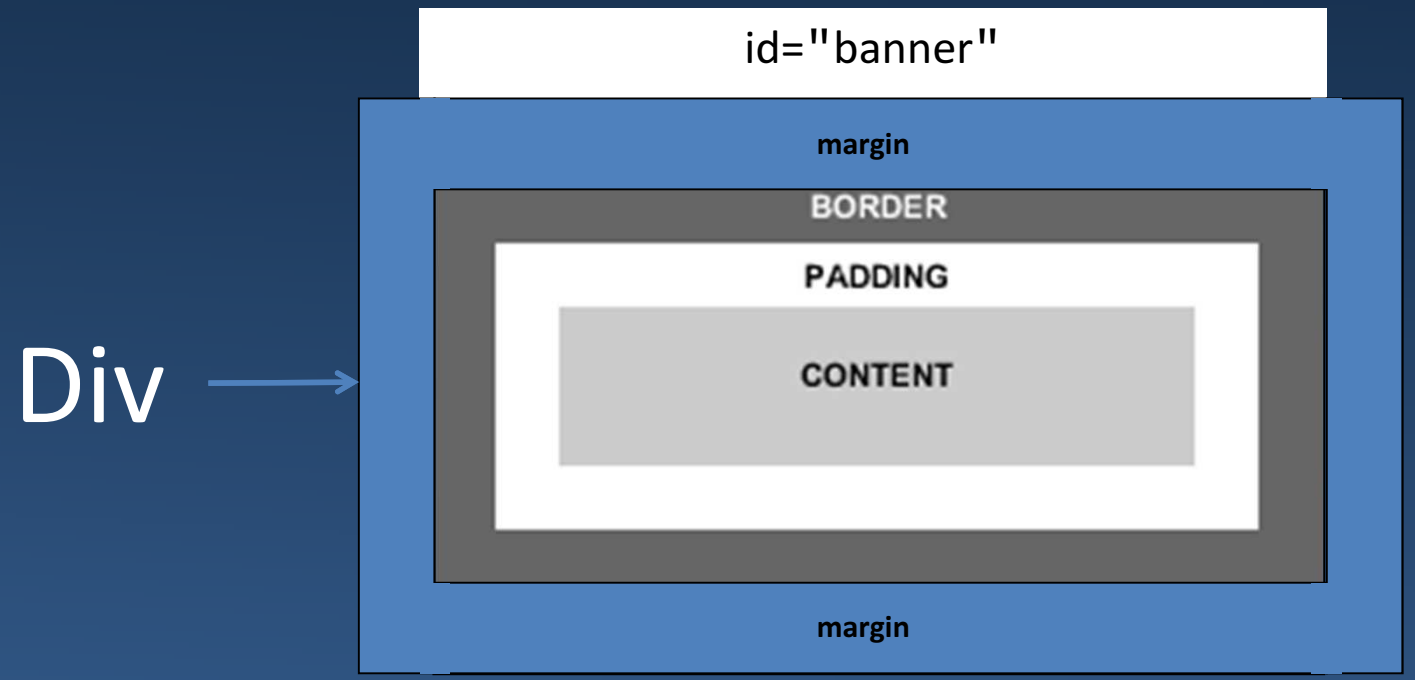
```
body {
   background-color:#ece5d7;
   font-family:"Lucida Sans", "Lucida Grande";
   font-size:12px;
   color:#000000;
}
```

http://www.computerhope.com/tips/tip143.htm

Styling a Div

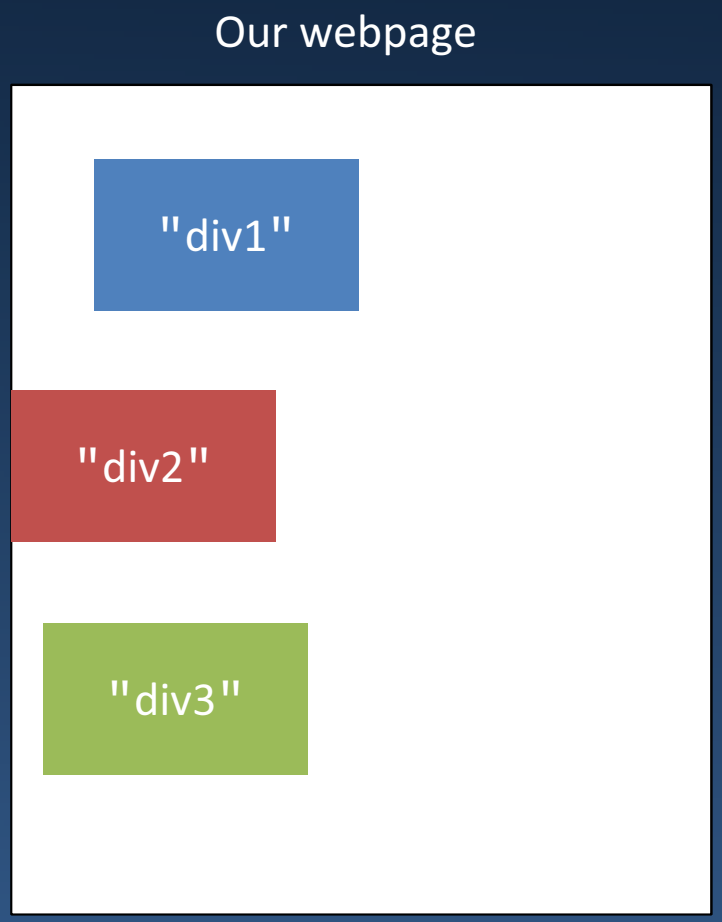
The bulk of our website will be constructed from multiple divs (within divs, within divs).

The Box Model



Margins

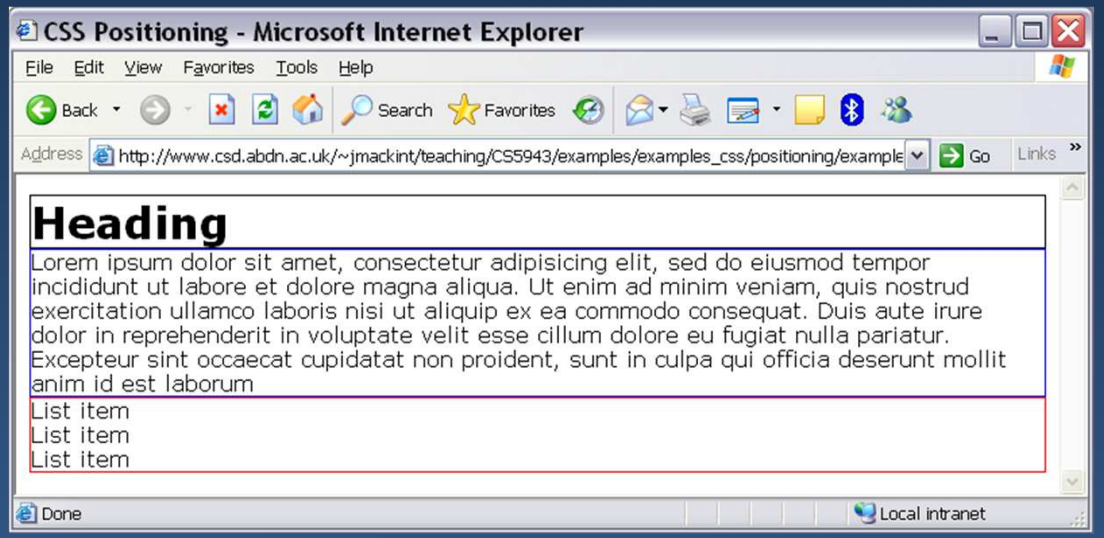
```
#div1 {
   background-color: blue;
   margin: 20px;
}

#div2 {
   background-color: red;
}

#div3 {
   background-color: green;
   margin-left: 10px;
   margin-top: 20px;
   margin-right: 0px;
   margin-bottom: 0px;
}
```



Design Tip

- Most HTML properties have default margins,
  - in particular, Paragraphs, Headings and Lists.
- To remove these, we can simply do:

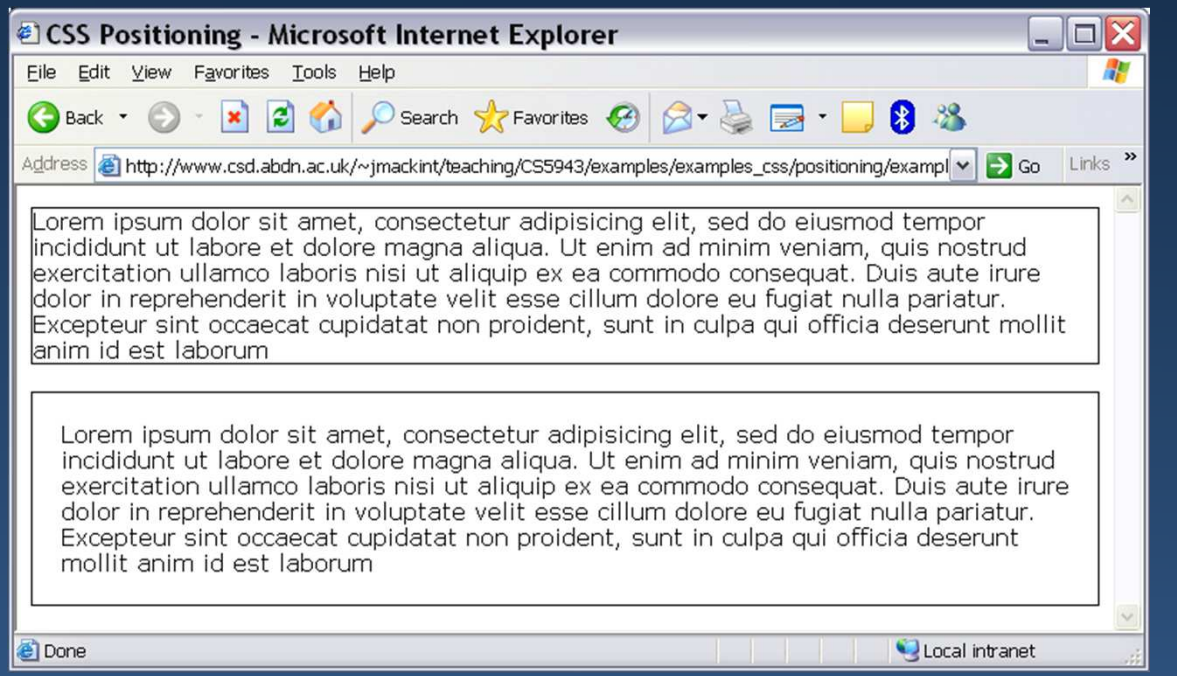
```
h1 {
    margin:0px;
}
p {
    margin:0px;
}
ul {
    margin:0px;
}
```



Padding

Works the same as Margins do, except the effect is applied within the box.

```
<style type="text/css">
.noPadding {
    padding: 0px;
}
.padding {
    padding: 20px;
    margin-top: 20px;
}
</style>
```



Design Tip

- You may find it useful to remove all of the default margins and padding.
- This leaves you fully responsible for all of the margins and padding, and is the preferred approach for most developers.
- This line is usually the first line in a cascading style sheet.

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

Borders

```
.borderClass {
   border-width: 1px;
   border-style: solid;
   border-color: #000000;
}
```

Same as

```
.borderClass {
   border: 1px solid #000000;
}
```

Same as

```
.borderClass {
   border-top: 1px solid #000000;
   border-right: 1px solid #000000;
   border-bottom: 1px solid #000000;
   border-left: 1px solid #000000;
}
```

Question

```
.ex {
   width:250px;
   padding:10px;
   border:5px solid gray;
   margin:10px;
}
```

```
<div class="ex">
some content.
</div>
```

What is the full width of the above div element?

1. 250px
2. 275px
3. 280px
4. 300px


Here is when it gets a bit difficult…

Positioning

- There are two kinds of positioning in CSS:
- Relative
  - The item in question is displayed after the previous item.
  - It fits in the natural flow of a document.
- Absolute
  - An item is placed at an exact location on the page, regardless of any other items.
  - If it overlays other items, so be it.

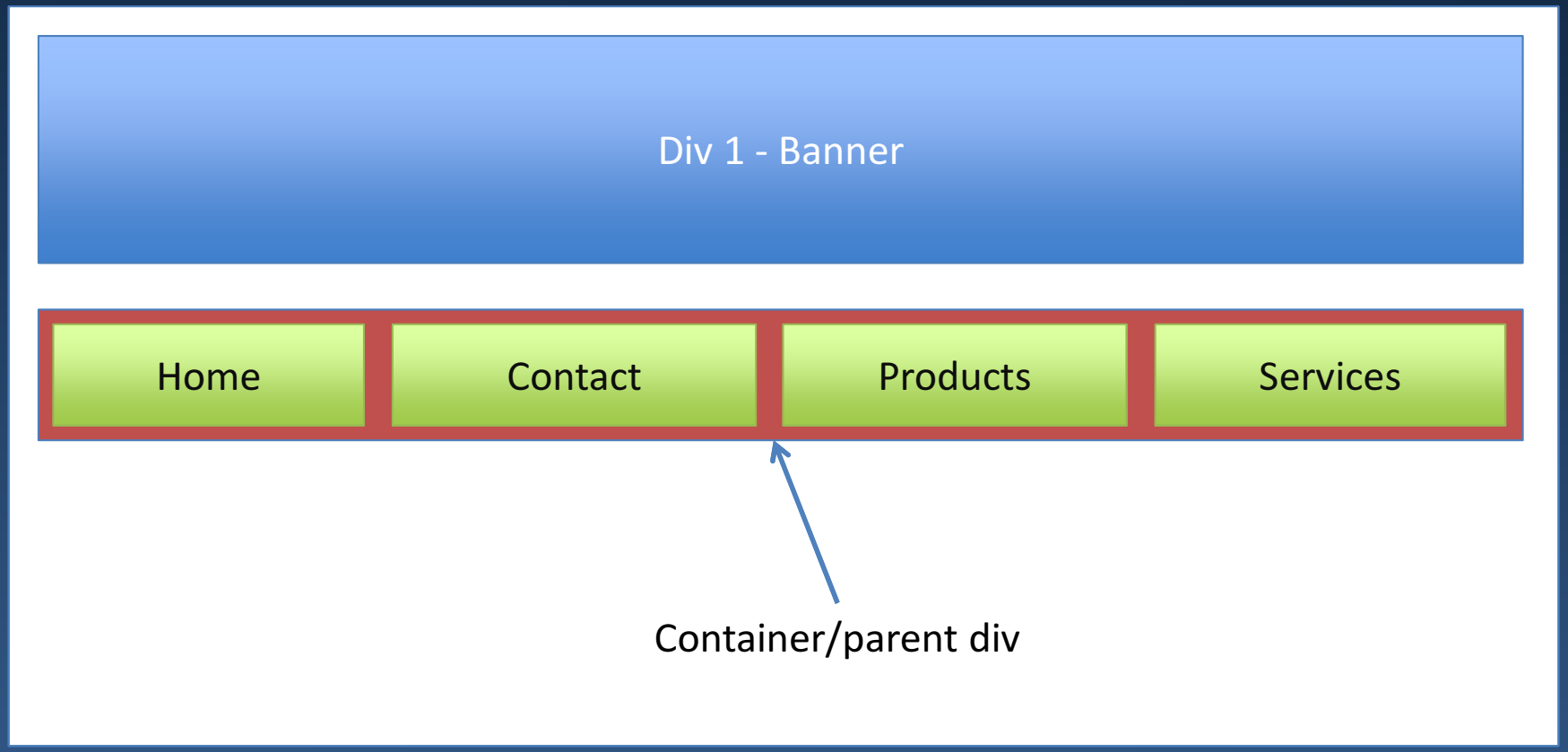
Relative Positioning

- This is the most widely used approach, because different boxes adapt to the size of the monitor displaying the webpage.

Relative Positioning
Parents / Containers

- The concept that's crucial to divs (or any elements) which are relatively positioned is the idea of parent boxes aka container boxes.
  - Hence the idea, divs within divs within divs
- **If we want to get multiple divs on the same line, then we must create a container div to hold them in.**

Relative Positioning
Parents / Containers



Relative Positioning
Parents / Containers


```
.container {
    width: 900px;
    position: relative;
    padding: 5px;
    …
}

.linkBox {
    width: 200px;
    position: relative;
    margin-right: 10px;
    …
}
```

```
<div class="container">
    <div class=”linkBox">
        <a href="home.html">Home</a>
    </div>
    <div class="linkBox">
        <a href="contacts.html">Contacts</a>
    </div>
    <div class="linkBox">
        <a href="products.html">Products</a>
    </div>
    <div class="linkBox">
        <a href=”services.html">Services</a>
    </div>
</div>
```

… There is however, one more thing we need…

- We need to tell the browser which direction to display our boxes. Either left-to-right or right-to-left.

  float: left;
  float: right;

- By specifying or box to **EITHER** float to the left or the right, we decide at which side of the parent div will our new div start at.

Therefore…


```
.container {
    width: 900px;
    position: relative;
    padding: 5px;
    …
}

.linkBox {
    width: 200px;
    position: relative;
    margin-right: 10px;
    float:left;
    …
}
```

```
<div class="container">
    <div class="linkBox">
        <a href="home.html">Home</a>
    </div>
    <div class="linkBox">
        <a href="contacts.html">Contacts</a>
    </div>
</div>
```

Or…


```
.container {
    width: 900px;
    position: relative;
    padding: 5px;
    …
}

.linkBox {
    width: 200px;
    position: relative;
    margin-right: 10px;
    float:right;
    …
}
```

```
<div class="container">
    <div class="linkBox">
        <a href="home.html">Home</a>
    </div>
    <div class="linkBox">
        <a href="contacts.html">Contacts</a>
    </div>
</div>
```

Putting it all together..

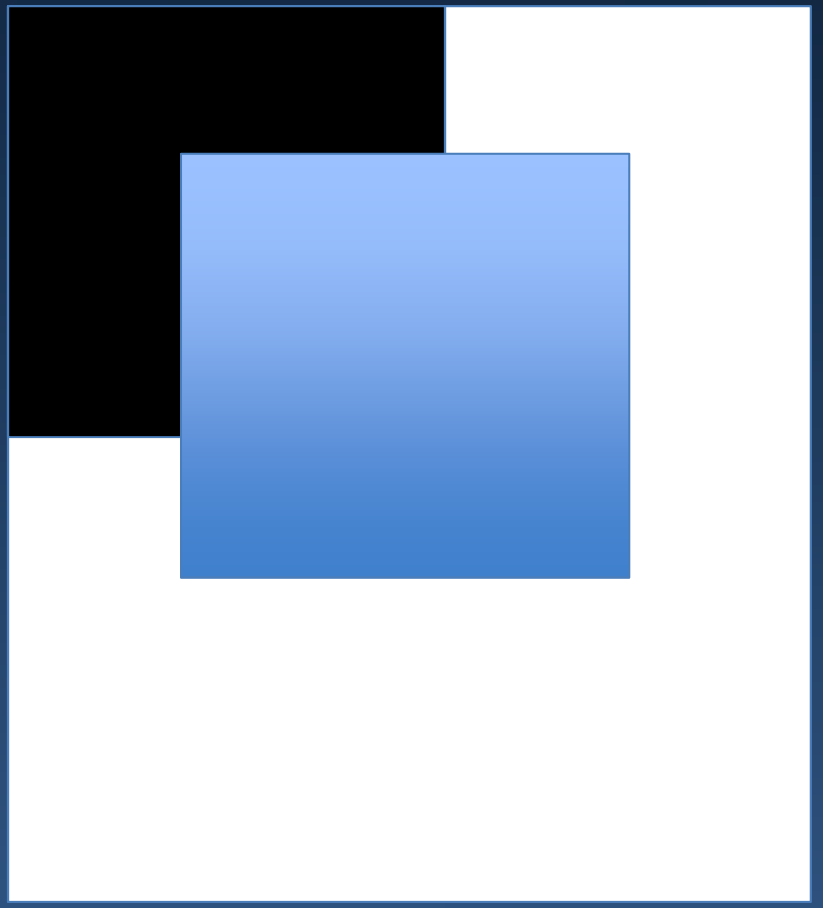
- Relative positioning requires:
  - A container/parent div with a fixed length.
  - A child div with a fixed length.
  - A float left or right.

Absolute Positioning

- Specify the exact position of an element on the page.
- Where it should be displayed in terms of either:
  - The page itself
  - A container element
- These elements are rendered outside normal flow.

Absolute Positioning

```
div {
     position: absolute;
     width: 100px;
     height: 100px;
     background-color: black;
     text-align: center;
}
div.new {
     top:40px;
     left:40px;
     background-color: blue;
}

<body>
     <div>…</div>
     <div class="new">…</div>
</body>
```



Question

- Which of the following is NOT true

1. Absolute positioning renders elements according to left, right, top, bottom
2. Float can be set to top or bottom
3. Absolute positioning renders elements on top of each other
4. Divs can be contained within other divs

References

- CSS Properties - http://htmldog.com/reference/cssproperties/
- Color codes - http://www.computerhope.com/tips/tip143.htm