



RESPONSIVE DESIGN

WITH **GUY ROUTLEDGE**

@GUYROUTLEDGE | #FEWD



BOXES - HTML & CSS REVIEW



MOBILE BOXES

FIXED VS RESPONSIVE

Checkout these **Fixed** sites

- Yeovalley.co.uk
- CoventGardenLondonUK.com
- HMRC.gov.uk
- Google.com

FIXED VS RESPONSIVE

Checkout these **Responsive** Sites

- Generalassemb.ly
- Kallo.com
- Ward-thomas.co.uk
- blog.webershandwick.co.uk

FIXED VS RESPONSIVE

The important takeaway is that a responsive site uses the same codebase to achieve optimised layout across multiple screens.

FIXED LAYOUT

- Simple and a good starting point
- Relies on a container of fixed width

FLUID LAYOUT

FLUID LAYOUT

- Different styles for different screen widths
- Uses an elastic/fluid layout
- Fluid: sized in percentages
- Elastic: sized in ems

EMS VS REMS

EM

Sized based on the width of the letter "M"

<http://alistapart.com/articles/howtosizetextincss>

Size is relative to the parent

```
body { font-size:16px; }  
h1 { font-size:2em; } /* font size is 32px */
```

EMS VS REMS

REM

- Stands for "Root" em.
- Based on the font-size of html (root) element
- Browser support (IE9+) is not as good as em so a px fallback is needed.

MEDIA QUERIES

MEDIA QUERIES

```
@media only screen and (max-width: 800px) {  
    /* styles only apply when viewport width < 800px */  
}
```


MEDIA QUERIES

A bit like a conditional statement

```
if ( condition ) {  
    // do stuff if condition is true  
}
```

MEDIA QUERIES

```
width | min-width | max-width
height | min-height | max-height
device-width | min-device-width | max-device-width
device-height | min-device-height | max-device-height
aspect-ratio | min-aspect-ratio | max-aspect-ratio
device-aspect-ratio | min-device-aspect-ratio | max-device-a
spect-ratio
color | min-color | max-color
color-index | min-color-index | max-color-index
monochrome | min-monochrome | max-monochrome
resolution | min-resolution | max-resolution
scan | grid
```

Separate multiple clauses with "and"

MOBILE DISPLAY

```
<meta  
  name="viewport"  
  content="width=device-width, initial-scale=1" >
```

Why is this necessary?

USING MEDIA QUERIES

```
/* float boxes into columns */  
.box {  
    float:left;  
}  
@media only screen and (max-width:800px) {  
    .box {  
        float:none;  
    }  
}
```

MEDIA QUERIES - MOBILE FIRST

- Start with a single column
- Add media queries using `min-width`
- Performance benefits

MEDIA QUERIES - DESKTOP FIRST

- Start with the full-width version
- Add media queries using `max-width`
- Can be more intuitive
- Can cause trouble scaling complex layout to narrow screen



RESPONSIVE BOXES

FLEXIBLE IMAGES

FLEXIBLE IMAGES

Images have fixed dimensions that can break our fluid layouts.
We can make them flexible by setting `max-width`

```
img {  
  max-width: 100%;  
}
```


