Class 9: Responsive Web Design

A short history of web design for media
One Web and Mobile First
The three components of responsive layouts:
- Flexible containers
- Flexible images
- Media queries

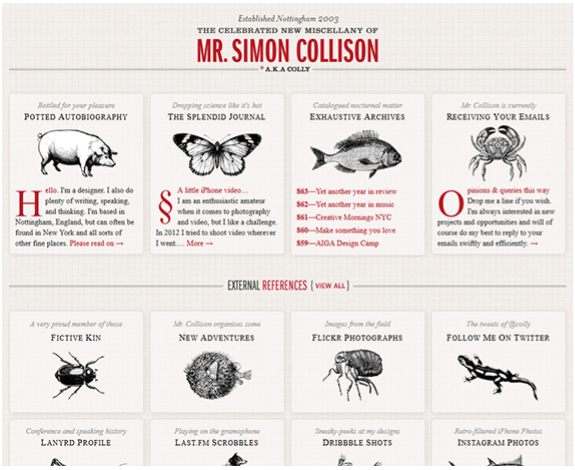
Fixed/Fluid/Adaptive/Responsive
Content-out
How to design websites now

References

Learning Web Design (5th Ed.) by Jennifer Robbins
Responsive Web Design (2nd Ed.) by Ethan Marcotte
Implementing Responsive Design by Tim Kadlec
The Mobile Book by Smashing Magazine chapters:
- Responsive Design Strategy by Trent Walton
- Responsive Design Patterns by Brad Frost

Responsible Responsive Design by Scott Jehl
Responsive Design: Patterns and Principles by Ethan Marcotte
Smashing Book #5: Real-Life Responsive Web Design
- A Modern Responsive Designer's Workflow by Dan Mall
- Responsive Design Patterns and Components by Vitaly Friedman

https://web.dev/articles/responsive-web-design-basics
http://alistapart.com/article/dao
http://alistapart.com/article/responsive-web-design/
http://mediaqueri.es/
http://learn.shayhowe.com/advanced-html-css/responsive-web-design/
http://coding.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/
http://blog.teamtreehouse.com/beginners-guide-to-responsive-web-design



Course materials: Design for Web Content

Responsive Web Design

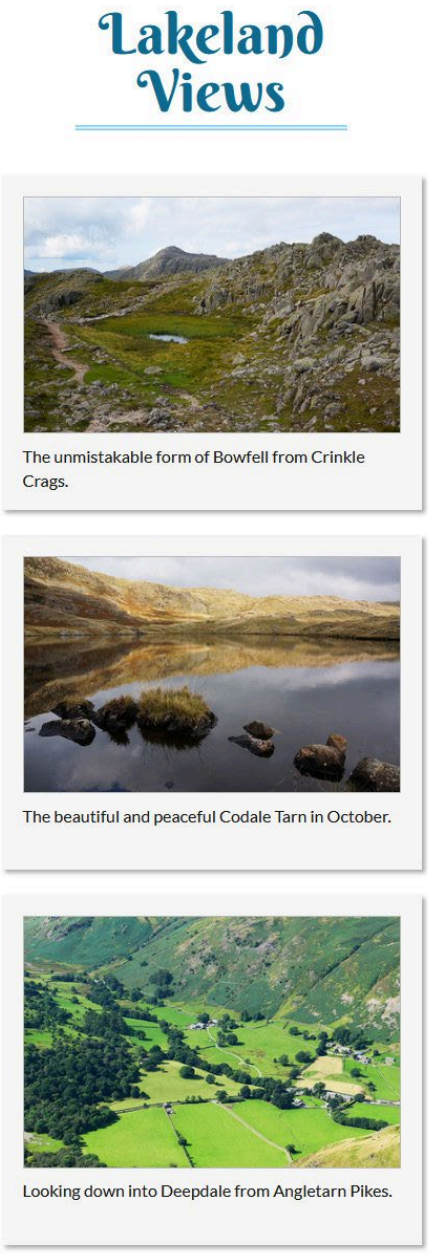
Responsive Web Design begins with a "mobile first" design as a default. This mobile design is then progressively enhanced as the viewport widens. Breakpoints, where the design begins to fail are identified and a media query is used to modify the design from that point onwards as the viewport continues to widen. It may be necessary to introduce several breakpoints to create a design that works well at all viewport widths.

The example CSS below is from a gallery of images. The default, mobile design (CSS not shown) allows the images to stack, one above the other, to form a single column. There is no need for any grid declarations because the content in the single column just flows in content order (`<figure>` is a block-level element and will stack naturally). This layout is shown on the right.

We've determined that when the viewport reaches 800px wide, the images are too big, and it would be better to display them in two columns. So, we create a media query `@media (min-width: 800px)` and add the declarations needed to transform our single column into two columns.

We're using *min-width*, which means that any declaration we make will be true when the viewport is 800px wide or wider – there is no upper limit. So, *display: grid* will be true for this media query and any subsequent media queries. This means we don't need to repeat it in media queries for wider viewports.

The next breakpoint comes at 1300px, where we have determined that the two-column images are too large and a change to three columns would be appropriate (layout shown below). In the 1300px media query, we only need to change the *grid-template-columns* declaration because everything else remains the same (*display* is still set to "grid" and *grid-column-gap* is still set to "1.5em" because 1300 is greater than 800.)

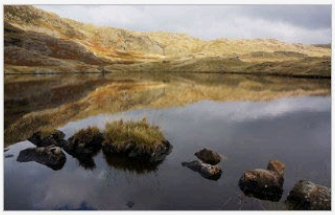



```css
@media (min-width: 800px) {
    main {
        display: grid;
        grid-template-columns: repeat(2, 1fr);
        grid-column-gap: 1.5em; /* same as margin-top on figure */
    }
}
@media (min-width: 1300px) {
    main {
        grid-template-columns: repeat(3, 1fr);
    }
}
@media (min-width: 1800px) {
    main {
        grid-template-columns: repeat(4, 1fr);
    }
}
```

Course materials: Design for Web Content

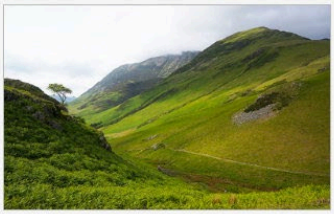
Lakeland Views

The unmistakable form of Bowfell from Crinkle Crags.

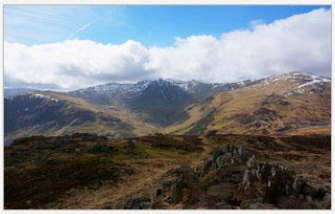


The beautiful and peaceful Codale Tarn in October.

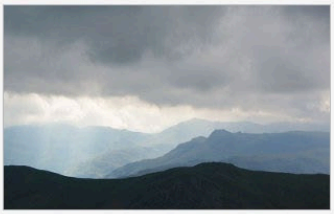
Looking down into Deepdale from Angletarn Pikes.



Whiteless Pike and Grasmoor from Low Bank.



A snowy Helvellyn as seen from Sheffield Pike.



The flat top of Harrison Stickle with Bowfell beyond from Dove Crag.

The final breakpoint, at 1800px is where we decide the layout should change to a four-column grid. Again, we only need to change the value of *grid-template-columns*.

Note that we are using 12 elements in our grid. Each element is a figure with two child elements, an image, and a caption. The number 12 is important because it modulates perfectly with 1, 2, 3, and 4 columns, leaving no “orphan” elements.

The files used in this example are available for you to study, and you can view the final version.

Although grid layouts are now the norm for modern websites, it is still important that you understand how floated layouts work too. You can look at this responsive gallery layout using floats for more information.

Top tips for Responsive Web Design

1. Start with Mobile First CSS (no breakpoints required for this).

2. Use the *min-width* property as a conditional for your breakpoints.

3. Start with the narrowest breakpoint and increase, step-by-step.

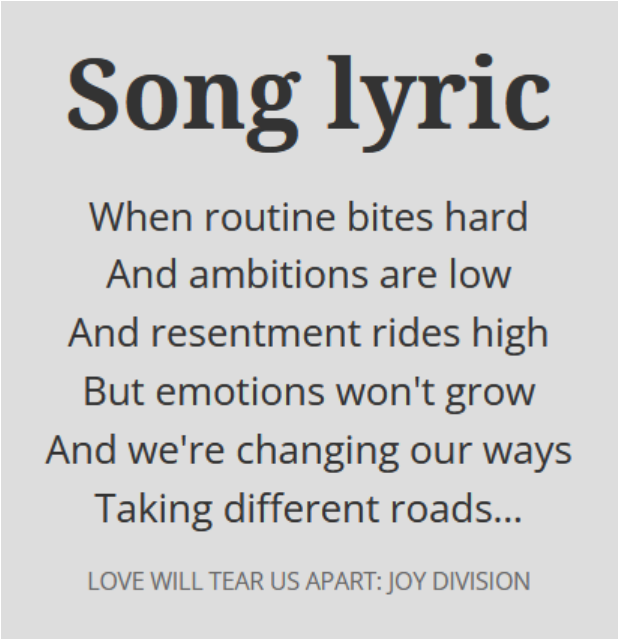
4. Always arrange your media queries with narrowest first and widest last (assuming you are using *min-width*).

5. Use the cascade to minimise code. For example, if a condition is true for all breakpoints, you only need to declare it for the first (narrowest) media query because the rest will inherit it (again, assuming you are using *min-width*).

Other methods

For the most part, you will be using media queries to control how your content looks at different viewport widths, but there are other methods. For example, if your page layout is very simple, you may find that working with some of the newer viewport-derived units provides a good, simple solution.

This example of responsive typography is achieved using the `vmin` CSS unit. Each `vmin` unit is 1% of the minimum viewport dimension, either height or width (whichever is smaller), and is very useful for ensuring that a content element remains visible, above the fold as the viewport shifts between landscape and portrait format. Find out more about the various CSS values and units.



In the example above, the font size is set to `font-size: 5.7vmin;` on `<body>`, and no media queries are required for the text to be completely responsive. Of course, page layouts are rarely this simple, but it is an illustration of what can be achieved with a good understanding of CSS units.

Class 9 Homework

Read: Responsive Web Design (if you haven't already done so!)
Chapter 18 of Learning Web Design
Complete the LinkedIn Learning tutorial: CSS: Advanced Layouts with Grid

Continue with the Small Business Website project. Start thinking about how you will implement a responsive design, starting from Mobile First. The best way to proceed is to design the mobile page layout first (no media queries) and then find your breakpoints and add media queries (using *min-width*) as the viewport widens. Remember, the breakpoints are related to the content, they are not related to devices. You are not designing for phone/tablet/desktop per se, you are designing for the content. If you design for content, your designs will work with any device.

During the winter break, I recommend that you spend some time reinforcing the learning you have done this term. One way to do this is to work through Level Up: CSS Layout.

Course materials: Design for Web Content