Exercise 11 – Introducing Grid Systems

Objective

In this exercise, we will use the Foundation CSS framework to create a responsive page.

Overview

In part 1 of the exercise, you will create a grid-based layout, in part 2 you will create a sub grid, which divides the page and use the responsive tools the framework provides. In part 3, you will create a nested grid and finally in part 4, you will have Foundation help you with the navigation.

Exercise Instructions

Part 1 – Using Foundation

Foundation is a mobile first responsive grid system and very useful for our needs. In the first part of the page, we will learn to add it to our web pages.

1. Open **foundation.html** from the **Exercise 11** starter. Before we do it, we will examine the CSS file to get a greater understanding of how foundation works. Within the CSS folder, you will locate foundation.css (not the min version) and open it.
2. Examine the file, scroll within it and locate the .cell class - the building block of the grid system within foundation.
3. Switch back to **foundation.html** and locate the comments. Insert a div with class "grid-container" as follow. This will keep the grid centred.

```
<div class="grid-container">
  <!-- header -->
  <!-- main body -->
  <!-- page footer -->
</div>
```

4. We will use the cell combined with earlier learned CSS techniques to create a header for our document. Locate the header comment and we'll start our page construction here. Implement the following mark-up.

```
<div class="gradient-one">
  <div class="grid-x">
       <div class="cell large-12">
         <header>
           <h1>
             East Kent Amateur Brewers
           </h1>
         </header>
       </div>
     </div>
</div>
```

5. This creates a stack of information; the header is contained within a cell and the “gradient–one” class wraps everything, so we can create a coloured header that spreads across the entire width of the document.
6. The **gradient–one** class is located in the **brew–styles.css** file, it provides a simple **rgba** colour that stretches the width of the page.
7. We want to make this a little better looking by using a font-face for the <h1> element. Give it a class of **babas-header**. Just before the closing outermost tag, create a element and give it a class of **stripe**. Then test the page in Chrome and use the **Responsive mobile view** to test the page at different device sizes. The grid should adapt to a series of different displays.
8. As we discussed, the default grid for foundation is broken down into 12 separate columns. You can tell a block element within a cell and how many of the columns it should utilise. Add a class of **large–12** to the class **.cell**, save the page and test it in Chrome.
9. Adding a class to the **<h1>** element of **hide-for-small-only**. Save the page and test it in Chrome, then use the Responsive Mobile Web viewer to test the page on a smaller display. Based upon the media queries built into foundation the heading should disappear.

Part 2 – Dividing the Grid

In this part of the exercise, we will divide the main body of the page into two columns, then subdivide these columns further to give us the main section of the document.

10. Locate the comment **<!-- main body -->**. We will surround this region of the page with a background graphic. We will then create 2 cells. Add the following HTML to achieve this:

```
<div class="kettle">
     <div class="grid-x">
       <div class="cell large-9">
         <article>
           <p>Article Test Data</p>
         </article>
        </div>
       <div class="cell large-3">
         <aside>
           <p>Aside Test Data</p>
         </aside>
       </div>
     </div>
</div>
```

11. We should examine the CSS styles currently being used, **kettle** which is located in the brew–styles CSS file, quickly check them and be sure you're happy with what they're doing – if you are not, then ask your instructor for help. Note that the default font for the body text, also comes from the brew-styles class.
12. Now save the file and check it in Chrome.

13. Next we will sub divide the <article> column – remove the <p> element you added in step 9 and then add the following HTML:

```
<header>


</header>
<section>


</section>
<section>


</section>
<footer>


</footer>
```

14. Open **article-header.txt** from the **content** folder and paste the contents into the **<header>** element you added in the previous step. Save your page and check it in Chrome.
15. Next turn your attention to the first section element, give it a class of **callout** then check the page in the browser. You should note the block has a greyish background colour. The **callout** class allows you to easily highlight blocks of content. The silver does not work very well with our style and we can easily override the existing classes where we need to. Open brew-styles.css from the CSS folder and add a new class called **callout** and set the background colour to be **rgba (146, 131, 48, 0.69)** then check the page in Chrome again.
16. Within this section now add the markup from **article-section.txt** and test the page in the browser. You should find the page is starting to take shape!
17. In the second <section> element, we are going to use the grid to help us position images. Simulating what often happens in design, we do not yet have the graphic files. We need to use placeholders in our page development. To help us with this, we will use a service from http://placehold.it, this will allow us to specify the width of the graphic and the service will provide the relevant data.
18. We will position the data using the default grid system Foundation. Give the second <section> classes of **grid-x grid-padding-x large-up-4**. Then add the following markup four times within this section:

```
<div class="cell"><img src="http://placehold.it/128x128">
</div>
```

19. Save the file and test it in Chrome and the place holder should now show as 4 equally spaced graphics. Utilise the Responsive viewer and test what happens on mobile devices. It is all working well except that 4 graphics across a 4" device may be a little small. One of the features both Foundation grid systems have, is to define a different grid layout for mobile and larger devices. Add the class **small-up-2** to the <section> element. Now check it on a desktop and mobile device, using the Responsive Grid Viewer – you should note the grid switches to two items per row.
20. To the article's footer element, add the text from the **article-footer.txt** file. Save the page and test it in Chrome, we should now have a pretty functional grid based page.

Part 3 – Nested Grids

Grids within Foundation are infinitely divisible – we are going to explore this by dividing the content within the <aside> into its own internal grid.

21. Within the <aside>, remove the existing <p> element and create a **div** using the **grid-x** class in its place. And then add a further **div** with class **cell**. Add a <h1> using the **babas-header** class and the text *Contact Us*.
22. Save the page and check it in Chrome – you'll note there is a problem: the light colour we are using for the font is being washed out on the white background. Before we use the grid, we are going to use an experimental CSS3 feature called text-stroke. It works like a block outline and allows us to set an outline on text without adding to the element's box size.
23. Open the brew-styles.css file and locate the **babas-header** and add the following css:

```
-webkit-text-stroke-width: 1px;
-webkit-text-stroke-color: rgba(146, 131, 48, 0.4);
```

24. Save then check the page in Chrome – you should note it has the brown background colour we have used in our page. Unfortunately, this only works in webkit browsers at time of writing (though please check http://caniuse.com/text-stroke to see if anything has changed). WebKit has us covered here, we can set the text fill colour with another proprietary property:

```
-webkit-text-fill-color: rgba(255,255,255,0.7)
```

25. The next thing for us to do is create a **card**. First under h1, create a div with a class "card" and then nested within it another div with a class "card-section". Then adding the following HTML:

```
 <ul>
   <li class="fn">EKAB</li>
   <li class="street-address">123 Wort Ave.</li>
   <li class="locality">Canterbury</li>
   <li><span class="state">Kent</span>, <span class="zip">CT1
3PB</span></li>
  <li class="email"><a href="#">ekab@cmail.com</a></li>
 </ul>
```

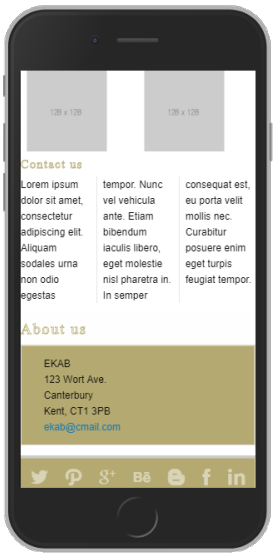
26. Add the following styling for the class card in the css:

```
background-color: rgba(146, 131, 48, 0.69);
```

27. Next set the class card list-style to none.

```
.card li{
  list-style: none;
}
```

28. Save and test the page in the browser.
29. Lastly in this section, we will add the footer with our social network icons – open the page-footer.txt file from the content folder and add it beneath the comment **<!-- page footer -->.**
30. Save and test the page in responsive mode. It should look similar to the following diagram.



Part 4 – Adding Navigation

Finally, in this exercise, we will use some of Foundations helper classes to create some navigation on our page with the assistance of link buttons and a side navigation bar.

31. Within the page level <header>, above the <h1> East Kent Amateur Brewers</h1>, add the following markup:

```
<nav class="float-right">
   <ul class="button-group">
     <li><a class="button">Link 1</a></li>
     <li><a class="button">Link 2</a></li>
     <li><a class="button">Link 3</a></li>
   </ul>
 </nav>
```

32. Save the page and investigate the buttons on mobile proportioned devices.
33. The colour scheme is not really working for us at the moment, create the following CSS in the **brew-styles.css** file –
    a) Add a selector for buttons within the button group and the following CSS:
        i) background-color: rgba(146, 131, 48, 0.75);
        ii) border-color: #ccc;
    b) A hover selector that sets the opacity of the button to 1
    c) A selector that sets the list-style to be none

34. Finally, we will add a side vertical menu at the top of the <aside>.

```
<div class="grid-x">
 <div class="cell">
  <ul class="vertical menu button-group">
   <li><a class="button" href="#">News</a></li>
   <li><a class="button" href="#">Events</a></li>
   <li><a class="button" href="#">Recipies</a></li>
   <li><a class="button" href="#">Blogs</a></li>
   <li><a class="button" href="#">Competitions</a></li>
  </ul>
 </div>
</div>
```

35. Save the page and check it in Chrome.