GRID LAYOUTS

INTRODUCTION

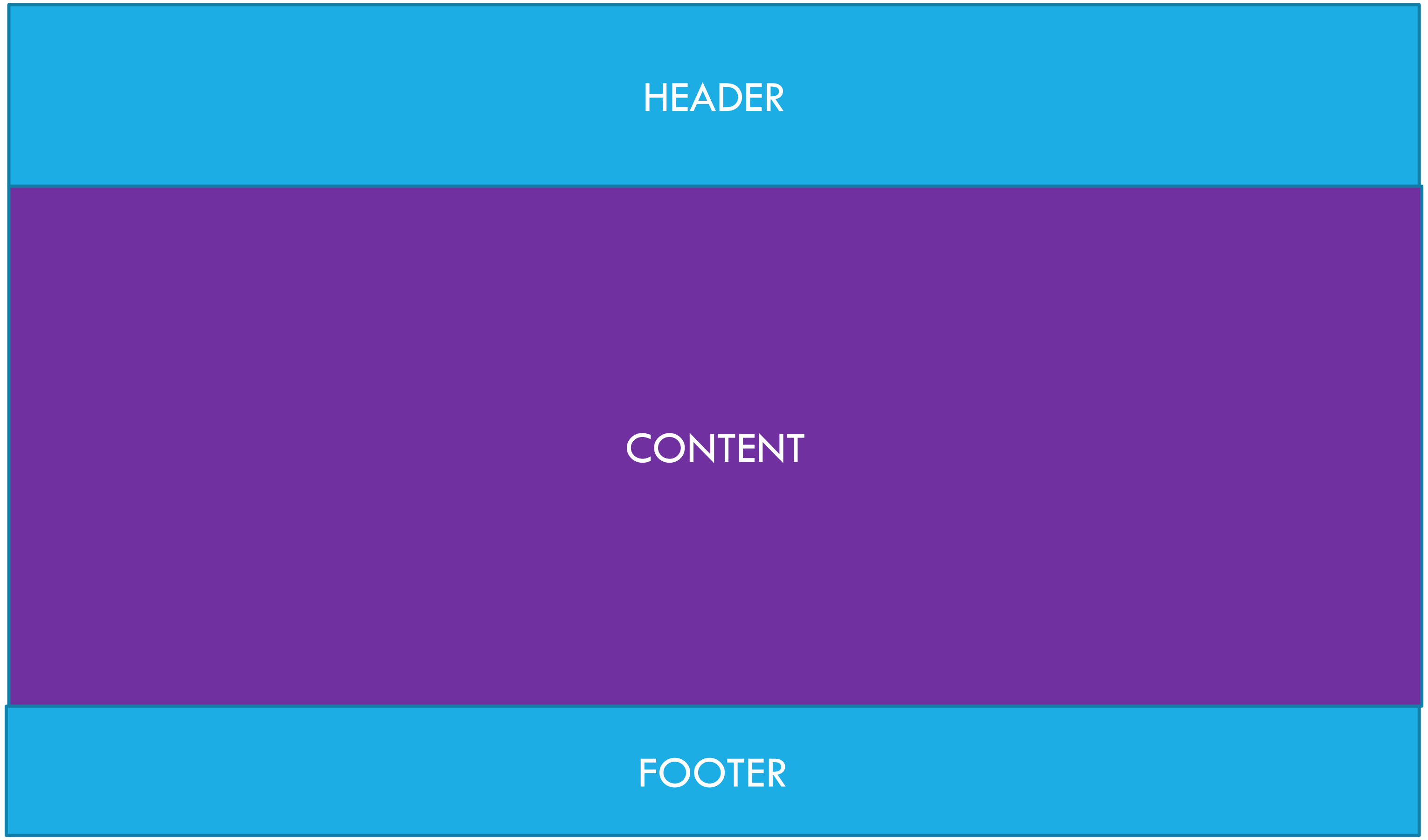
We have introduced some of the syntax for grid and seen in basic form that it is fairly good at designing the layout of your page.

To get everyone really on board with grid, let’s show off some of the layouts grid can create that will give your site a more professional layout and feel.

Note that just because your site is well laid out, does not mean it will look good! That is only part of the equation.

STARTING POINT

Let’s have a look at a basic layout that will feel familiar to almost everyone. We start with a header, have some content and then end off with a footer:



HEADER

CONTENT

FOOTER

Looks fairly grid like! Let’s see if we can make this with just a few lines of HTML & CSS.

STARTING HTML

To get the HTML starting point, let’s add the most basic semantic and content tags we can:

```
<body>
    <header id="page_header">
        <h1>HEADER</h1>
    </header>
    <main id="page_content">
        <h1>CONTENT</h1>
    </main>
    <footer id="page_footer">
        <h1>FOOTER</h1>
    </footer>
</body>
```

This gives us 3 children of the body. A good starting point.

STARTING CSS

Now let’s see how easy it is to create the picture from before with CSS, starting with the grid container:

```
body {
    display: grid;

    width: 100%;
    margin: 0px;
    padding: 0px;
}
```

We set the body to display grid, the rest of the CSS is just to make the site take up the whole viewport.

STARTING CSS CONT.

Next, let’s tackle the header:

```
#page_header {

    min-height: 50px;

    background-color: dimgray;

}
```

This one is easy in our toy example. We just give the header a **min-height** that works on all screen sizes and can expand if it needs to. Also, set the background color so we can see it.

STARTING CSS CONT.

Next, the content:

```
#page_content {
    min-height: calc(100vh – 100px);
    background-color: darkslategray;
}
```

Here we see something new, the **calc** function. This handy CSS function allows you to calculate dimensions when you must mix and match.

For our toy example, I know that the header and the footer are both going to be at minimum 50px in height. Our calc function ensures the body will be at least at tall as the page minus the header and footer.

STARTING CSS CONT.

Finally, the footer:

```
#page_footer {
    min-height: 50px
    background-color: lightslategray;
}
```

This is very similar to the header; we simply give a min-height of 50px and a background color to show the boundaries.

STARTING POINT SUMMARY

If you followed that code, you should have something that looks like the image shown in the slides before (minus the text being in the center).

This is a good starting point for almost any website. Obviously, you need to add in your own content instead of just having background colors and one text tag per section.

There is also a very good chance that the header, main and footer tags themselves would be grids! Remember grid is a layout tool that can be used for big and small layout changes. You can use it to influence the flow of the entire page, or simply just to move an image to the center of an article.

KNOWLEDGE CHECK

The repeat function can make your life much easier when it comes to making responsive and adaptive layouts. Even more basic, it can create good looking structures layouts.

1. Create a folder in Scratch called BasicGridStarting
2. Connect this folder to Git/GitHub
3. Create an index.html and style.css
4. Add a header, main and footer tag
5. Use a mix of grid and min-height to give your site a starting point as outlined in the previous slides
6. Add commit and push your code
7. Add in an image, and nav tag to the header
    1. Place 4 a tags inside the nav tag
8. Make it so all content within the header will stack vertically as the screen gets smaller.
9. Add commit and push your code.

GRID LAYOUTS

Once we are comfortable getting our layout down, we can really start to focus on how to actually style the page in terms of looking nice.

You can really think about creating a page in this order:

1. Get the content on the page
2. Lay the content out in the correct spot
3. Make it look pretty with colors, fonts, margin, padding, etc.