Department of
Computer Science and Engineering

Title:Implementation of Page Layout in CSS

Web Programming Lab

CSE 302

Green University of Bangladesh

1 Objective(s)

- To gather knowledge of different CSS layout such as grid, flexbox, and table layout.
- To implement an responsive layout using CSS grid layout.

2 Problem analysis

The CSS **Grid** Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning. CSS Grid Layout (aka "Grid" or "CSS Grid"), is a two-dimensional grid-based layout system that, compared to any web layout system of the past, completely changes the way we design user interfaces. CSS has always been used to layout our web pages, but it's never done a very good job of it. First, we used tables, then floats, positioning and inline-block, but all of these methods were essentially hacks and left out a lot of important functionality (vertical centering, for instance). **Flexbox** is also a very great layout tool, but its one-directional flow has different use cases — and they actually work together quite well! Grid is the very first CSS module created specifically to solve the layout problems we've all been hacking our way around for as long as we've been making websites. **Floating** an element alters the behavior of both the element and the block level elements that follow it in normal flow. The floating element is shifted to the left or right and out of regular flow, while the surrounding content floats around it. The float property has four potential values: left: Floats the element to the left. Right: Floats the element to the right. None: Indicates that no floating occurs. This is the default value. inherit: Defines whether the value of the float property should be inherited from the element's parent element. When looking at the source code for older websites, you may see that **tables** were utilized to lay out forms. HTML tables should be used to show tabular data. Using tables for anything other than tabular data has several drawbacks: table layouts are rigid, require a lot of markup, are difficult to debug, and are semantically incorrect. When you utilize table markup on a webpage, a collection of CSS attributes dictate how the table will appear. These same characteristics can also be used to layout items that aren't tables, a practice known as "using CSS tables".

3 Implementation in HTML and CSS

3.1 Grid

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=<device-width>, initial-scale=1.0">
    <title>Document</title>

    <style>
        .item1{grid-area: header;}
        .item2{grid-area: menu;}
        .item3{grid-area: main;}
        .item4{grid-area: aside;}
        .item5{grid-area: footer;}


        .grid-container{
            display: grid;
            background-color: cadetblue;
           /* grid-template-columns: auto auto auto auto auto auto;*/

           grid-template-areas:'header header header header header header'
                               'menu main main main main aside'

                               'footer footer footer footer footer footer' ;
```

```
        }

        .grid-container > div{
            border: 3px solid blueviolet;
            padding:10px;
            text-align: center;

        }

      /*  .item1{
            grid-column: 1/7;

        }
        .item2{
            grid-column: 1/2;
            grid-row:2/4;

        }
        .item3{
            grid-column: 2/6;
        }

        .item4{
            grid-column: 6/7;
            grid-row: 2/4;
        }

        .item5{
            grid-column: 2/6;

        }*/


    </style>
</head>
<body>
    <div class="grid-container">

        <div class="item1">
            <h1>Header</h1>
        </div>

        <div class="item2">
            <h1>Menu</h1>
        </div>

        <div class="item3">
            <h1>Lorem ipsu. Lorem ipsum dolor sit amet consectetur adipisicing
               elit. Animi sequi aspernatur vel quis est eveniet quasi quas
               repudiandae illum sunt quo qui perspiciatis deleniti praesentium,
                fugit, excepturi laborum veritatis molestias? . Lorem ipsum
               dolor sit amet consectetur, adipisicing elit. Ducimus qui
               provident voluptate temporibus atque? Explicabo, adipisci?
               Veritatis quae hic dolorum, vitae aperiam voluptates, doloremque
               necessitatibus impedit quia corporis non ex. Lorem ipsum dolor
               sit amet consectetur adipisicing elit. Voluptas dolor excepturi,
               libero a aperiam sequi nesciunt exercitationem consequuntur nulla
                facilis hic reiciendis deserunt doloribus autem nihil nemo
```

```
                quaerat rem inventore! Lorem ipsum dolor sit amet consectetur
                adipisicing elit. Sint praesentium nostrum aut ipsum animi
                deleniti dicta amet quod. Eligendi numquam, amet illo velit alias
                 laborum delectus non repellat quaerat voluptate. </h1>
        </div>

        <div class="item4">
            <h1>Aside</h1>
        </div>

        <div class="item5">
            <h1>Footer</h1>
        </div>

    </div>
</body>
</html>
```

3.2 Flexbox

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>Create a Flex Container</h1>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

3.3 Float

```
<!DOCTYPE html>
<html>
<head>
<style>
```

```
img {
  float: right;
}
</style>
</head>
<body>

<h2>Float Right</h2>

<p>In this example, the image will float to the right in the paragraph, and the
   text in the paragraph will wrap around the image.</p>

<p><img src="pineapple.jpg" alt="Pineapple" style="width:170px;height:170px;
   margin-left:15px;">
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet,
   nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim
   ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor
   vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula,
   facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus
   interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed
   ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed
    ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer
   fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta.
   Cras ac leo purus. Mauris quis diam velit.</p>

</body>
</html>
```

3.4 Table Layout

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border: 1px solid;
}

table {
  width: 100%;
}
</style>
</head>
<body>

<h2>Full-width Table</h2>

<table>
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
  </tr>
  <tr>
    <td>Peter</td>
    <td>Griffin</td>
  </tr>
  <tr>
```

```
28      <td>Lois</td>
29      <td>Griffin</td>
30    </tr>
31  </table>
32
33  </body>
34  </html>
```

4 Input/Output

The output of the program is given below.

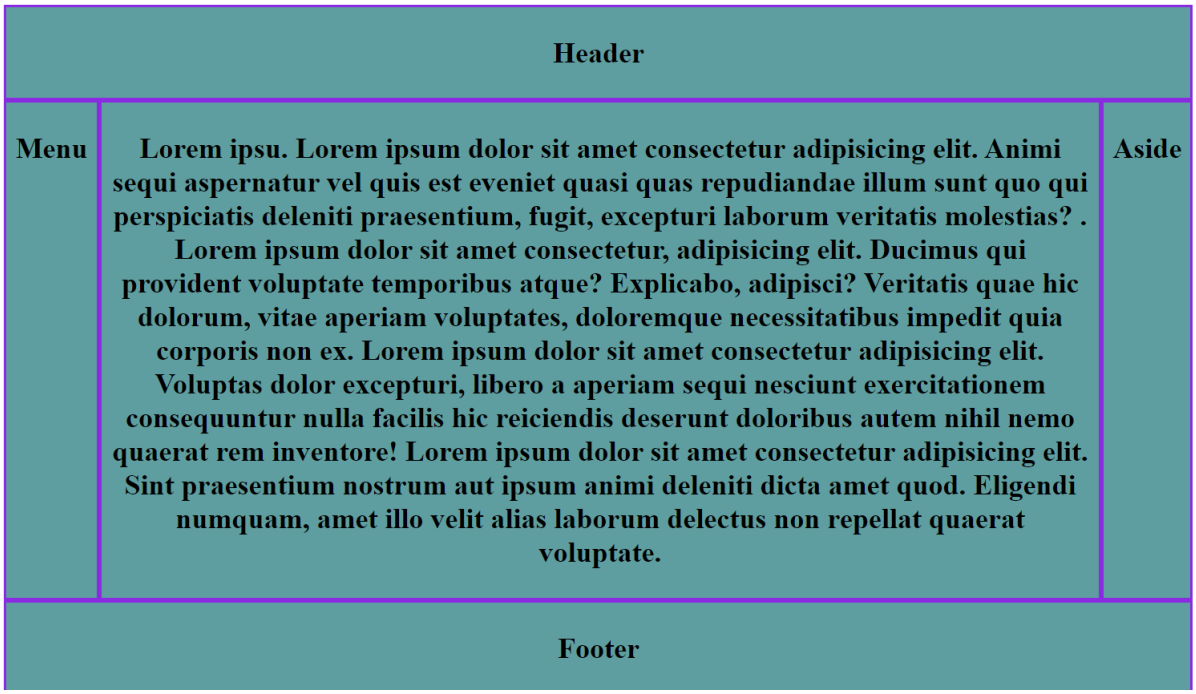

Figure 1: CSS Grid layout

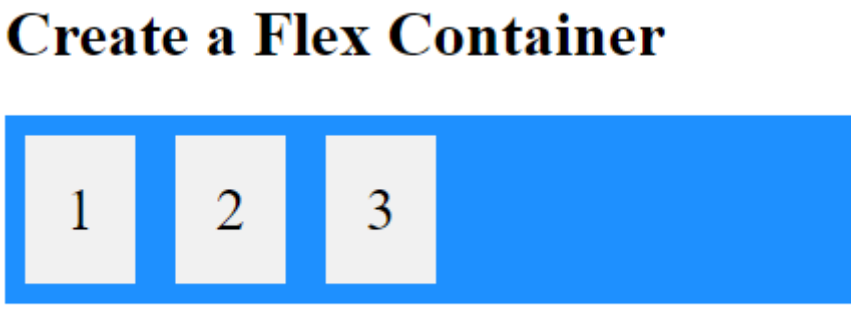

Figure 2: Flexbox layout

Float Right

In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.

Figure 3: Float Right layout

Full-width Table

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Figure 4: Table layout

5 Discussion & Conclusion

Based on the objective(s) to understand about different CSS layout, the additional lab exercise made me more confident about learning different layout techniques.

6 Lab Task (Please implement yourself and show the output to the instructor)

1. Implement the CSS layout.

6.1 Problem analysis

Design a responsive page layout using CSS grid, flex, float, table. By using CSS grid/flex/float/table area property you should design the layout. The responsiveness of the can be added by using media screen property.

7 Lab Exercise (Submit as a report)

- Implement the page layout using CSS.
- Implement the responsive property of CSS in the page layout.

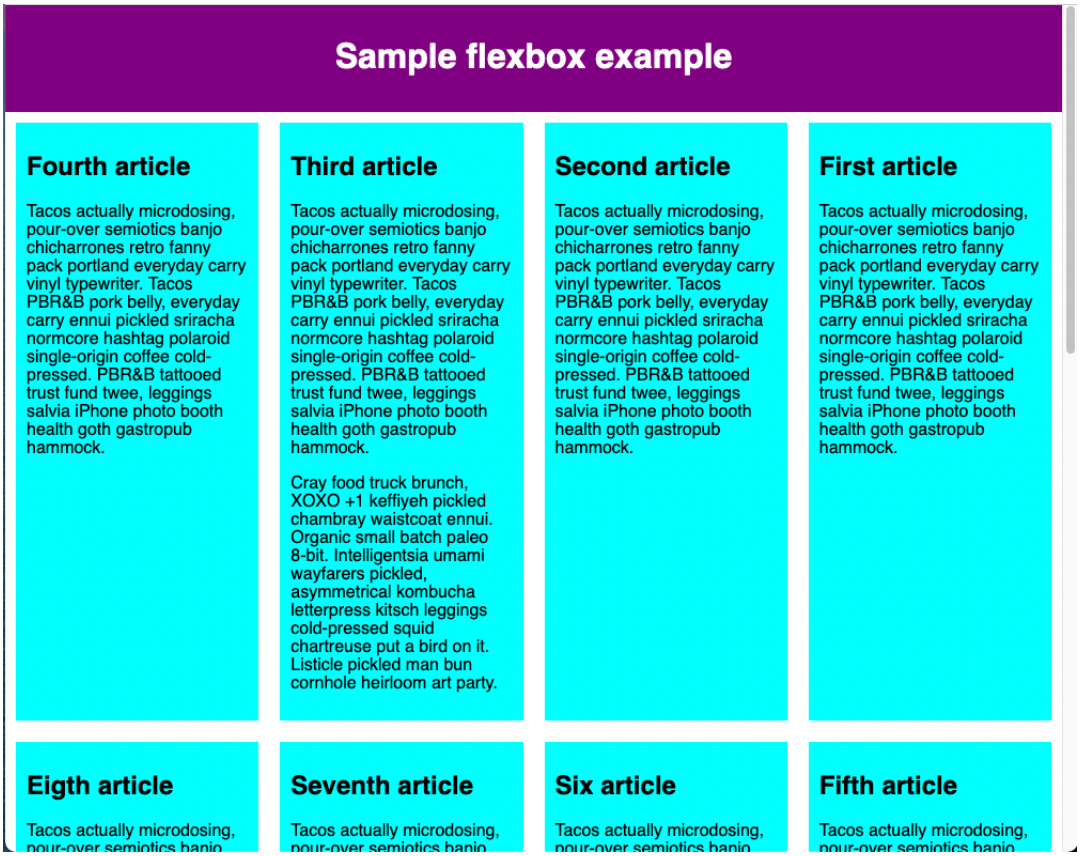

Figure 5: CSS page layout using Flexbox Layout

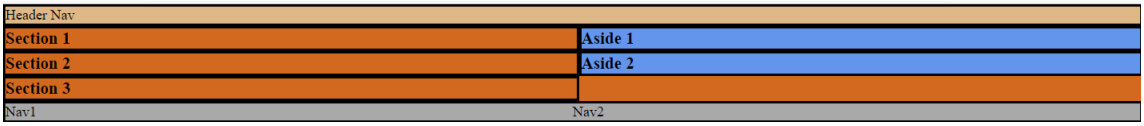

Figure 6: Sample page layout using Table Layout

Figure 7: CSS page layout using Grid Layout

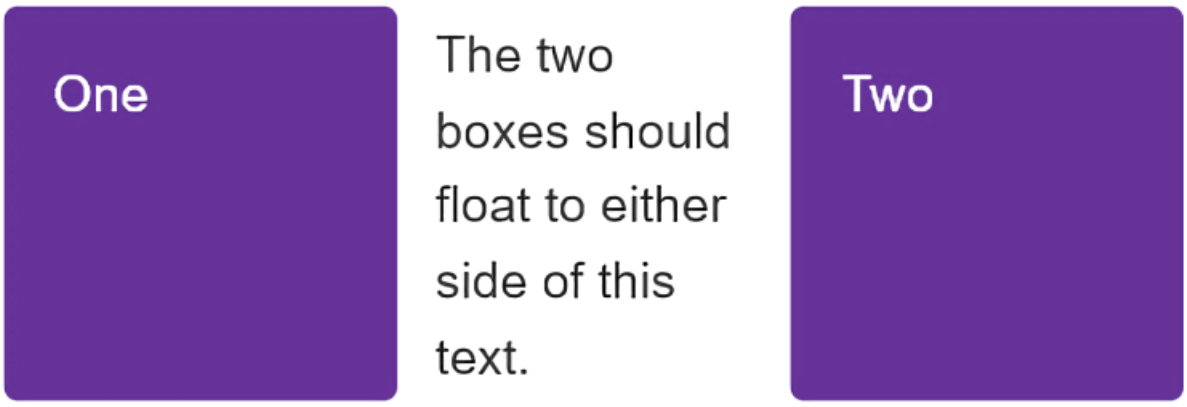

Figure 8: CSS page layout using Float Layout

8 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.