

COMP335 Web Application Development

CSS

1

- Group Project
 - group name and short (one paragraph) description
 - <http://goo.gl/forms/eo5wnbMtXf>

- Today
 - CSS Layout
 - Responsive Design
 - CSS Frameworks

- Next class
 - JavaScript

- **Assignment#2-CSS**
 - by Friday 2/26
- **In-class programming exam (tentative)**
 - 3/14 (M)


2

CSS Layout

- One of the main problems faced by web designers is that **the size of the screen** used to view the page can vary quite a bit.
 - 21-inch wide screen monitor (1920x1080 pixels)
 - older iPhone with a 3.5-inch screen (320x480 pixels)
- Satisfy both users can be difficult
- Most designers **used to take one of three basic approaches** to dealing with the problems of screen size.
 1. Fixed Layout
 2. Liquid (Fluid) Layout
 3. Hybrid Layout

Fixed Layout

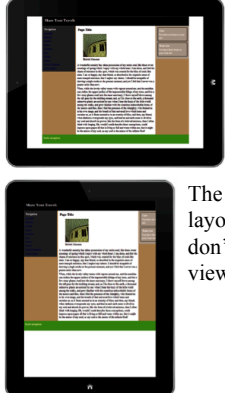
- In a **fixed layout**, the basic width of the design is set by the designer, typically corresponding to an "ideal" width based on a **"typical" monitor resolution**.
- The **advantage** of a fixed layout is that it is **easier to produce** and generally has a predictable visual result.
- Fixed layouts have **drawbacks**:
 - For **larger screens**, there may be an **excessive amount of blank space** to the left and/or right of the content.
 - It is also optimized for typical desktop monitors; however, as more and more user visits are happening via **smaller mobile devices**.



```
div#wrapper {  
  width: 960px;  
  background-color: tan;  
}  
  
<body>  
  <div id="wrapper">  
    <header>  
    ...  
    </header>  
    <div id="main">  
    ...  
    </div>  
    <footer>  
    ...  
    </footer>  
  </div>  
</body>
```

960px Extra space to right

960px Equal space to the left and to right



The problem with fixed layouts is that they don't adapt to smaller viewports

Liquid Layout

- In a **liquid layout** (also called a **fluid layout**) widths are not specified using pixels, but **percentage** values.
- Advantage:
 - adapts to different browser sizes,
- Disadvantages:
 - Liquid layouts can be more difficult to create because **some elements**, such as images, **have fixed pixel sizes**.
 - The **line length** may become **too long** or **too short**.



Fluid layouts are based on the browser window.

However, elements can get too spread out as browser expands



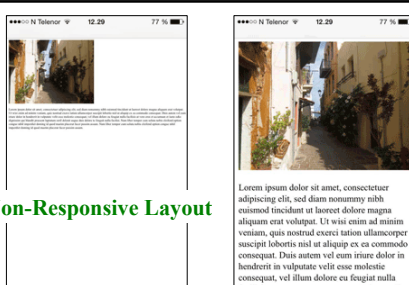
Hybrid Layout

- A **hybrid layout** combines pixel and percentage measurements.
 - Fixed pixel** measurements might make sense for a **sidebar column** containing mainly graphic advertising images that must always be displayed and which **always are the same width**.
 - Percentages** would make more sense for the main content or navigation areas, with perhaps **min** and **max size limits** in pixels set for the navigation areas

Responsive Design

- In a **responsive design**, the page “responds” to changes in the browser size.
- In a **responsive layout**, **images may be scaled down** (not fixed size any more) and **navigation elements may be replaced** as the browser shrinks.
- There are several components that make responsive design work.
 - Scaling images to the **viewport size**
 - Setting viewports via the **<meta>** tag
 - Customizing the CSS for different viewports using **media queries**

Non-Responsive Layout



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

Responsive Layout using viewport

example: seikyung.com/index.html



Ideal Responsive Layout Using media queries

example: seikyung.com/index.html



Notice how some elements are scaled to shrink as browser window reduces in size



When browser shrinks below a certain threshold, then layout and navigation elements change as well.

In this case, the **** list of hyperlinks changes to a **<select>** and the two-column design changes to one column

Media Queries

- The other key component of responsive designs is **CSS media queries**.
- A media query is a way to **apply style rules** based on the **medium** that is displaying the file.
- You can use these queries to **look at the capabilities of the device**, and then define CSS rules to target that device.

```
<link href="mobile.css" media="screen and (max-width:480px)">
<link href="tablet.css" media="screen and (min-width:481px) and (max-width:768px)">
<link href="desktop.css" media="screen and (min-width:769px)">
```

mediaQueries.html

Defines this as a media query Device has to be a screen CSS rules to use if device matches these conditions

@media only screen and (max-width:480px) { ... }

Only use this style if both conditions are true

Use this style if width of viewport is no wider than 480 pixels

```
@media only screen and (max-width:480px)
{
    small {display: none;}
    iframe {max-width: 100%;}
}
```

seikyung.com/index.html

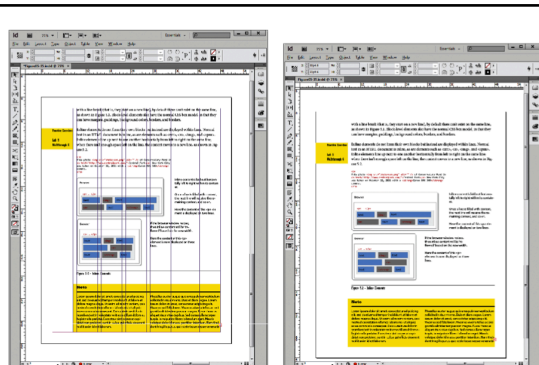
CSS Frameworks

- A **CSS framework** is a **pre-created set** of CSS classes or other software tools that make it easier to use and work with CSS.
- Grid systems** make it easier to create multicolumn layouts.
 - Bootstrap (<http://getbootstrap.com/getting-started/>)
- Another useful framework
 - <https://fortawesome.github.io/Font-Awesome/examples/>

Grid Systems

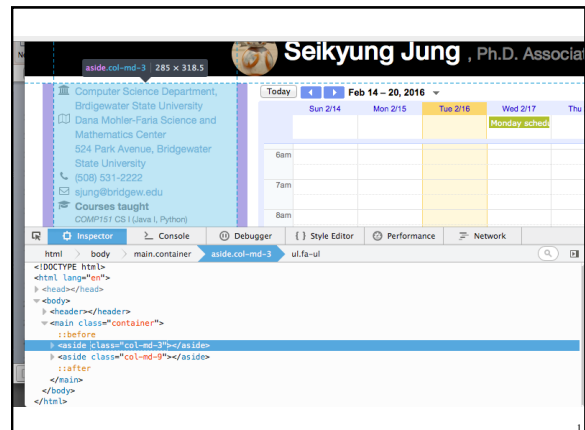
- Print designers typically use grids as a way to achieve visual uniformity in a design.
- In print design, the very first thing a designer may do is to construct a **5- or 7- or 12-column grid**.
- The rest of the document, whether it be text or graphics, will be aligned and sized according to the grid

examples: noBootstrapIndex.html vs. index.html



Most page design begins with a grid. In this case, a seven-column grid is being used to layout page elements in Adobe InDesign.

Without the gridlines visible, the elements on the page do not look random, but planned and harmonious.



Grid Systems - Bootstrap Code

```
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
<main class="container"> <!-- centered -->
  <aside class="col-md-3">
    <ul class="fa-ul">
      <li><icon class="fa-li fa fa-institution"></icon></li>
      <li><icon class="fa-li fa fa-map-o"></icon></li>
      <li><icon class="fa-li fa fa-phone"></icon></li>
      <li><icon class="fa-li fa fa-envelope-o"></icon></li>
      <li><icon class="fa-li fa fa-graduation-cap"></icon></li>
    </ul>
  </aside>
  <aside class="col-md-9">
    <iframe ...></iframe>
  </aside>
</main>
</body>
```

<http://getbootstrap.com/css/>

In-class Exercise

- Modify "bootstrapEx1.html" to integrate "bootstrap.min.css"
 - @media: Remove images in smaller browser size
 - Grid system: <http://getbootstrap.com/css>

Assignment#2

- Modify your homepage with media query to make responsive design (due by Friday 2/26)
 - Must use CSS
 - Must use @media
 - Viewport is not required, but recommended
 - Grid system is not required, but recommended. Use either bootstrap or HTML5 and CSS3