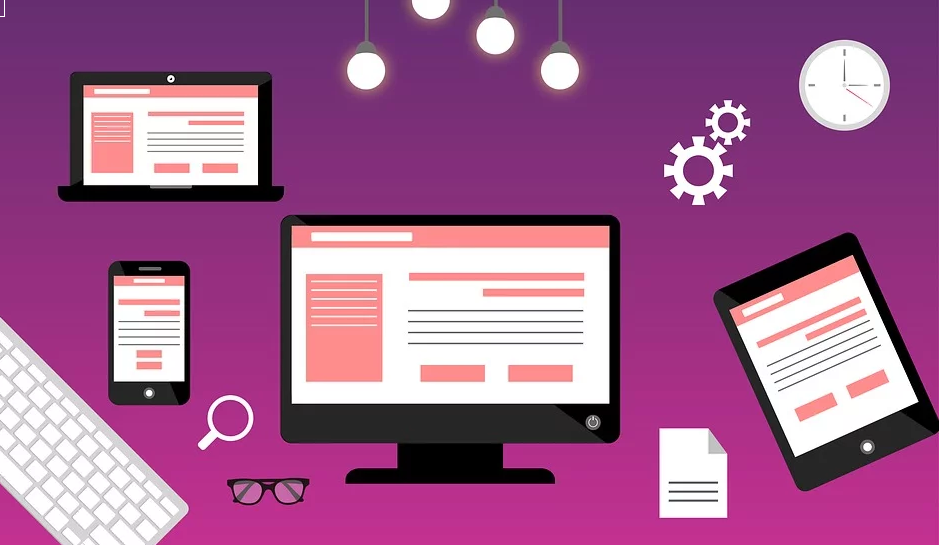
Consider the water or liquid stored in a cup, jug, and bottle.



The water adjusts according to the shape of the container in which it is stored. This can be related to responsive web design. The water can be assumed as the content of the web page and the water storage containers can be assumed as devices (mobile, tablet, and desktop) in which the content is rendered.

A Responsive page is a web page that renders well in all the devices like desktop, laptop, tablet, and mobiles, which have different screen sizes.



We can make a web page's layout change according to the screen size using the below techniques

* Fluid Layouts
* Flexible Images
* Media Queries

**Flexible Images**

The image grows or shrinks according to the container dimensions. We need to specify the width of the image using the percentage of its container width.

**Setting the width of foreground images**

We will apply relative dimensions to the <img> element

Problem: Consider we have an image that needs to be displayed at 150X150 size inside a 720 px wide element and we want it to scale it as the parent element scales.

Solution: For the above problem, we can have a container exclusively for the image, set its relative size on the container, and let the image fill container width.

i.e., 150/720 = 20.8333%

and to scale the image when the parent scales but limit the extent to which it scales up/down we can set max-width/min-width on the parent.

**Setting the width of the background images**

Problem: Consider we want the background image to scale when the element for which background is set to scales.

Solution: For the above problem, we can set background-size: 100%

**Fluid Layouts**

Layout grows and shrinks according to the viewport dimensions. We need to use relative units such as % instead of absolute units (px).

In order to fit the layout to viewport width, we will replace absolute units for dimensions of elements(px) with relative units (%)

The width of the topmost element (<html>) is browser width (due to viewport tag)

Thus specifying the width of all elements using % will scale the layout to the viewport width. Such a layout is called fluid layout

We need to represent all px widths as % of the corresponding parent width.

**Media queries**

Media queries apply CSS styling based on the device form factors. Hence the layout can differ on a desktop, tablet, and mobile

1. @media screen and (max-width: 520px) {
2. .body {
3. background: none;
4. }
5. }

Line 1: This media query hides a background image for the element with the class body when the page is viewed on a screen and browser viewport width is lesser than 520px,

@media starts a media query

screen: CSS to be applied only to screen media type

max-width: Apply CSS if the screen size is 520px wide, or less

**Media Types**

The important media types are

|  |  |
| --- | --- |
| **Media Type** | **Description** |
| all | Applies to all devices (default value) |
| handheld | Applies to handheld devices |
| print | Applies to printer |
| screen | Applies to any type of screen |
| tv | Applies to TV screens |

**Media Features**

The important media types are

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Values** | **min/max** | **Description** |
| device-height, device-width | number | Yes | Height/Width of the output device (specified in px, em,  etc.) |
| height, width | number | Yes | Height/Width of the viewport |
| orientation | landscape | portrait | No | The orientation of the device |
| resolution | number | Yes | Resolution(pixel density) of the device (in dpi, dpcm, etc.) |
| device | number | Yes | The number of device pixels per CSS pixel. Used to detect high-resolution screens like iPad retina |

It is observed that sometimes mobile devices display a full desktop site. In such cases, the computed page width is much higher than the expected and hence the appropriate media queries will not be applied. It also results in pages being zoomed out on mobile devices.

To overcome the above problem we can use the viewport meta tag

1. <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scale=0">

When we set width=device-width the page will be as wide as the device which prevents default zooming out actions on some mobiles

When we set initial-scale=1.0 the page is unscaled initially and cannot be further scaled by the user

The viewport is the area on a browser where the page is rendered.

Viewport width > Browser width > page is wider than browser

**Highlights:**

* Understand the usage of media queries

**Demo steps:**

1. Create Mediaqueries.html and write the code as shown below

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <title>MediaQueries Demo</title>
5. <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6. <link rel="stylesheet" media="screen" href="css/default.css" />
7. <link rel="stylesheet" media="screen" href="css/enhanced.css" />
8. </head>
9. <body>
10. <header>
11. <h1>RWD</h1>
12. </header>
13. <div id="content">
14. <div style="position: absolute; ">
15. <p><img src="images/house.jpeg"></p>
16. </div>
17. <ul id="features">
18. <li class="engage">
19. <h3>Fluid Layout</h3>
20. <p>SS3 Media queries enable you to style a page based on different display surface
21. factors such as width, height, orientation, resolution, etc. Media query listeners</p>
23. </li>
24. <li class="customize">
25. <h3>Flexible Image</h3>
26. <p>SS3 Media queries enable you to style a page based on different display surface
27. factors such as width, height, orientation, resolution, etc. Media query listeners</p>
29. </li>
30. <li class="integrate">
31. <h3>Media Queries</h3>
32. <p>SS3 Media queries enable you to style a page based on different display surface
33. factors such as width, height, orientation, resolution, etc. Media query listeners</p>
35. </li>
36. </ul>
37. </div>
39. <footer>
40. <p>Copyright 2020</p>
41. </footer>
42. </body>
43. </html>

The image can be downloaded [here](https://academy.onwingspan.com/common-content-store/Shared/Shared/Public/lex_auth_012985643372634112707_shared/web-hosted/assets/house.jpeg)

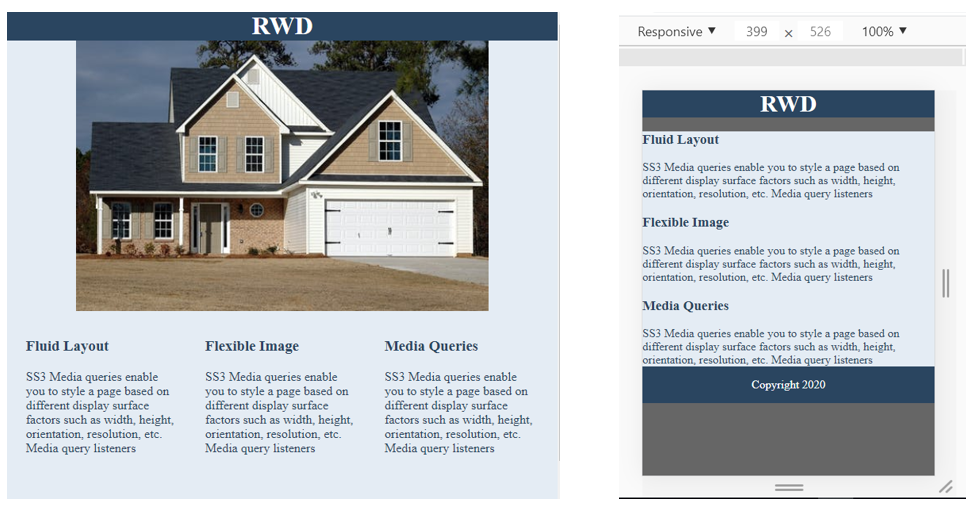
2. Create default.css file within the CSS folder and write the code as shown below

1. html {
2. background-color: *#666;*
3. }
4. body {
5. color: *#2a4560;*
6. }
7. header,
8. header a {
9. background-color: *#2a4560;*
10. color: *#fff;*
11. }
12. div {
13. background-color: *#e4ecf4;*
14. color: *#2a4560;*
15. }
16. footer {
17. background-color: *#2a4560;*
18. }
19. footer p {
20. color: *#fff;*
21. }
22. ul*#features > li a {*
23. color: red;
24. font-weight: bold;
25. text-align: center;
26. text-decoration: none;
27. }
28. header,
29. footer,
30. h1 {
31. text-align: center;
32. }
33. body,
34. header,
35. div,
36. footer,
37. fieldset,
38. ul,
39. li,
40. h1,
41. h2,
42. p,
43. img {
44. margin: 0;
45. padding: 0;
46. }
47. img {
48. width: 75%;
49. display: block;
50. margin-left: auto;
51. margin-right: auto;
52. }
53. *#features a {*
54. margin-top: 1em;
55. padding: 0.5em;
56. }
57. footer {
58. padding: 1em;
59. }

3. Create enhanced.css file within the CSS folder and write the code as shown below

1. @media (max-width: 400px) {
2. ul*#features li a {*
3. color: black;
4. }
5. img {
6. display: none;
7. }
8. }
9. @media (min-width: 600px) {
10. body > \* {
11. margin: 0 auto;
12. max-width: 56.25em;
13. }
14. ul*#features {*
15. text-align: center;
16. }
17. ul*#features li {*
18. display: inline-block;
19. margin: 2%;
20. min-height: 18em;
21. text-align: left;
22. width: 28%;
23. }
24. }
25. @media screen and (min-width: 900px) {
26. header {
27. height: 4em;
28. }
29. }

4. Observe the output



**NOTE:** The output shown above may vary depending on which browser you are using.