LAB # 04

Statement Purpose:

To familiarize the students with

- CSS Transition
- Animation, Responsive Webpages using Media Queries
- CSS Selectors
- Functions
- Animatable

CSS Transitions

CSS transitions allows you to change property values smoothly, over a given duration.

Mouse over the element below to see a CSS transition effect:

CSS

In this chapter you will learn about the following properties:

- transition
- transition-delay
- transition-duration
- transition-property
- transition-timing-function

How to Use CSS Transitions?

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

Note: If the duration part is not specified, the transition will have no effect, because the default value is 0.

The following example shows a 100px * 100px red element. The element has also specified a transition effect for the width property, with a duration of 2 seconds:

Example

```
div {
  width: 100px;
```

```
  height: 100px;
  background: red;
  transition: width 2s;
}
```

The transition effect will start when the specified CSS property (width) changes value.

Now, let us specify a new value for the width property when a user mouses over the element:

Example

```
div:hover {
  width: 300px;
}
```

Sample Code

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}

div:hover {
  width: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>
```

```
<p>Hover over the div element below, to see the transition effect:</p>
<div></div>


</body>
</html>
```

Change Several Property Values

The following example adds a transition effect for both the width and height property, with a duration of 2 seconds for the width and 4 seconds for the height:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s, height 4s;
}

div:hover {
  width: 300px;
  height: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>
<p>Hover over the div element below, to see the transition effect:</p>
<div></div>
</body>
```

</html>

Specify the Speed Curve of the Transition

The transition-timing-function property specifies the speed curve of the transition effect.

The transition-timing-function property can have the following values:

- ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
- linear - specifies a transition effect with the same speed from start to end
- ease-in - specifies a transition effect with a slow start
- ease-out - specifies a transition effect with a slow end
- ease-in-out - specifies a transition effect with a slow start and end
- cubic-bezier(n,n,n,n) - lets you define your own values in a cubic-bezier function

Sample Code

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}

#div1 {transition-timing-function: linear;}
#div2 {transition-timing-function: ease;}
#div3 {transition-timing-function: ease-in;}
#div4 {transition-timing-function: ease-out;}
#div5 {transition-timing-function: ease-in-out;}

div:hover {
  width: 300px;
```

```
}
</style>
</head>
<body>

<h1>The transition-timing-function Property</h1>

<p>Hover over the div elements below, to see the different speed curves:</p>

<div id="div1">linear</div><br>
<div id="div2">ease</div><br>
<div id="div3">ease-in</div><br>
<div id="div4">ease-out</div><br>
<div id="div5">ease-in-out</div><br>

</body>
</html>
```

CSS Animations

CSS allows animation of HTML elements without using JavaScript or Flash!

CSS

In this chapter you will learn about the following properties:

- @keyframes
- animation-name
- animation-duration
- animation-delay
- animation-iteration-count
- animation-direction
- animation-timing-function
- animation-fill-mode
- animation

What are CSS Animations?

An animation lets an element gradually change from one style to another.

You can change as many CSS properties you want, as many times as you want.

To use CSS animation, you must first specify some keyframes for the animation.

Keyframes hold what styles the element will have at certain times.

The @keyframes Rule

When you specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at certain times.

To get an animation to work, you must bind the animation to an element.

The following example binds the "example" animation to the element. The animation will last for 4 seconds, and it will gradually change the background-color of the element from "red" to "yellow":

Example

```
/* The animation code */
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}

/* The element to apply the animation to */
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

Sample Code

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
```

```
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes back to its original style.</p>

</body>
</html>

<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
```

```
  animation-duration: 4s;
}

@keyframes example {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: blue;}
  100% {background-color: green;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes back to its original style.</p>

</body>
</html>
```

Sample Code

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
```

```
  position: relative;
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes back to its original style.</p>

</body>
</html>
```

CSS Media Queries

CSS2 Introduced Media Types

The @media rule, introduced in CSS2, made it possible to define different style rules for different media types.

Examples: You could have one set of style rules for computer screens, one for printers, one for handheld devices, one for television-type devices, and so on.

Unfortunately these media types never got a lot of support by devices, other than the print media type.

Media Query Syntax

A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.

```
@media not|only mediatype and (expressions) {
  CSS-Code;
}
```

The result of the query is true if the specified media type matches the type of device the document is being displayed on and all expressions in the media query are true. When a media query is true, the corresponding style sheet or style rules are applied, following the normal cascading rules.

Unless you use the not or only operators, the media type is optional and the all type will be implied.

You can also have different stylesheets for different media:

```
<link rel="stylesheet" media="mediatype and|not|only
(expressions)" href="print.css">
```

Media Queries Simple Examples

One way to use media queries is to have an alternate CSS section right inside your style sheet.

The following example changes the background-color to lightgreen if the viewport is 480 pixels wide or wider (if the viewport is less than 480 pixels, the background-color will be pink):

Example

```
@media screen and (min-width: 480px) {
  body {
    background-color: lightgreen;
  }
}
```

Sample Code

```
<!DOCTYPE html>

<html>

<head>

<style>
```

```
body {
  background-color: pink;
}

@media screen and (min-width: 480px) {
  body {
    background-color: lightgreen;
  }
}
</style>
</head>
<body>

<h1>Resize the browser window to see the effect!</h1>
<p>The media query will only apply if the media type is screen and the viewport is 480px wide or wider.</p>
</body>
</html>
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
.wrapper {overflow: auto;}

#main {margin-left: 4px;}
```

```
#leftsidebar {
  float: none;
  width: auto;
}

#menulist {
  margin: 0;
  padding: 0;
}

.menuitem {
  background: #CDF0F6;
  border: 1px solid #d4d4d4;
  border-radius: 4px;
  list-style-type: none;
  margin: 4px;
  padding: 2px;
}

@media screen and (min-width: 480px) {
  #leftsidebar {width: 200px; float: left;}
  #main {margin-left: 216px;}
}
</style>
</head>
```

```
<body>

<div class="wrapper">

 <div id="leftsidebar">

   <ul id="menulist">

     <li class="menuitem">Menu-item 1</li>

     <li class="menuitem">Menu-item 2</li>

     <li class="menuitem">Menu-item 3</li>

     <li class="menuitem">Menu-item 4</li>

     <li class="menuitem">Menu-item 5</li>

   </ul>

 </div>

   <div id="main">

   <h1>Resize the browser window to see the effect!</h1>

   <p>This example shows a menu that will float to the left of the page if the viewport is 480 pixels wide or wider. If the viewport is less than 480 pixels, the menu will be on top of the content.</p>

  </div>

</div>

</body>

</html>
```

CSS Animatable

Definition and Usage

Some CSS properties are *animatable*, meaning that they can be used in animations and transitions.

Animatable properties can change gradually from one value to another, like size, numbers, percentage and color.

Sample Code

```
<!DOCTYPE html>
<html>
<head>
<style>
#myDIV {
  width: 300px;
  height: 200px;
  background: red;
  animation: mymove 5s infinite;
}

@keyframes mymove {
  from {background-color: red;}
  to {background-color: blue;}
}
</style>
</head>
<body>
<h1>Animation of background-color</h1>

<p>Gradually change the background-color from red to blue:<p>

<div id="myDIV"></div>

</body>
</html>
```