HTML5 and CSS3

Objectives

After completing this lesson, you should be able to:

- Apply CSS styles to HTML documents
- Use CSS3 features to add dynamic styles to elements with events
- Use media queries and media data to adapt styles to different screens
- Use JavaScript to add and remove styles from elements

CSS Basics

- CSS rule syntax:

```
selector {
    property_one: value_one;
    property_two: value_two;
}
```

- Applying a style:

```
/* Comments in a style sheet */
h1 {
    background-color: white;
    color: red;
}
```

selector

semicolon

new line

value

property

Adding CSS to HTML

1. **Inline style:**

```
<h1 style="color:red;">The Next Big Conference for
    Front-End Developers</h1>
```

2. **Embedded style:**

```
<head>
    <style>
      h1 {
      color: red;
      }
    </style>
</head>
```

3. Adding a style sheet:

```
<!DOCTYPE html>
<html>
    <head>
      <title>The Next Big Front-End Conference</title>
      <meta charset="UTF-8">
      <link rel="stylesheet" href="styles.css" />
      <script src="code.js"></script>
    </head>
    <body>
    </body>
</html>
```

```
<style>
    @import url(style.css);
</style>
```

- **Overriding a rule:**

```
h1 {
    background-color: red !important;
}
```

Selectors

1. Element type selectors:

```
p {
    color: blue;
    font-size: 1.4em;
}
```

- Multiple selectors:

```
p, h2 {
    text-transform: capitalize;
}
```

- Child selectors:

```
aside p {
    color: pink;
}
```

Only those <p> descendants of <aside>

2. Class selectors:

```
.copyright {
    font-size: smaller;
}
```

- Multiple classes:

```
.copyright.user-groups  {
    font-size: smaller;
}
```

3. ID selectors:

```
#section-1 {
  color: green;
}
```

Combinators

- Universal selectors:

```
* {
    color: blue;
}
```

- Descendant selector:

```
#section-2 p {
    font-weight: bold;
}
```

- Child selectors:

```
#section-2 > article > header > h1 {
    color: red;
}
```

Combinators

- Adjacent sibling selectors:

```
h1 + p {
    color: purple;
}
```

- General sibling selector:

```
header ~ p {
    color: orange;
}
```

Attribute Selectors

- Syntax:

```
E[attr]
E[attr=val]
```

- Example of attribute:

```
input[required] {
    background-color: yellow;
}
```

- Example of attribute value:

```
input[type="email"] {
    border: 1px dashed black;
}
```

Attribute Selectors

- Syntax:

```
E[attr^=val]                    // Starts with
E[attr*=val]                    // Contains
E[attr$=val]                    // Ends with
```

- *Starts with* example:

```
a[href^="#section-"] {
    color: red;
}
```

- *Contains an* example:

```
a[href*="tion-"] {
    color: orange;
}
```

Pseudo-Class Selectors

Pseudo-class	Matches
:link	Any link that the user has not visited yet
:visited	Any visited link
:hover	A link as the user passes the cursor over it
:active	An element being activated by the user
:focus	An element that has the focus
:enabled	Any element in enabled state
:checked	Radio option or check box in selected state

Pseudo-class	Matches
:target	Target elements
:first-child	The first child of an element
:last-child	The last child of an element
:nth-child(n)	The *n* child of an element
:nth-of-type(n)	The *n* child of a particular type, of its parent
:root	The element that is the root of the document
:not	An element not represented by its argument

Pseudo-Element Selectors

Pseudo-Element	Matches
`::first-line`	The first line of a given element
`::first-letter`	The first letter of an element
`::before`	Add content preceding a given element
`::after`	Add generated content after an element

Adding Dynamic Styles to Elements

- On hover:

```
input[type=submit]:hover {
    cursor: pointer;
}
```

- On focus:

```
input[type=text]:focus {
    background-color: lightyellow;
}
```

- On active:

```
a:active {
    font-size:150%;
}
```

CSS3 Properties: Fonts

generic font family

```
p { font-family: "Trebuchet MS", Verdana, sans-serif; }
```

- @font-face:

```
@font-face {
    font-family: "my paragraphtext";
    src: url(WebFont.woff) format("woff"),
         url(WebFont.ttf) format("truetype");
}
p {
    font-size: 12px;
    font-family: "my paragraphtext", Helvetica, sans-serif;
    font-weight: normal;
}
```

generic font family

CSS3 Properties: Colors

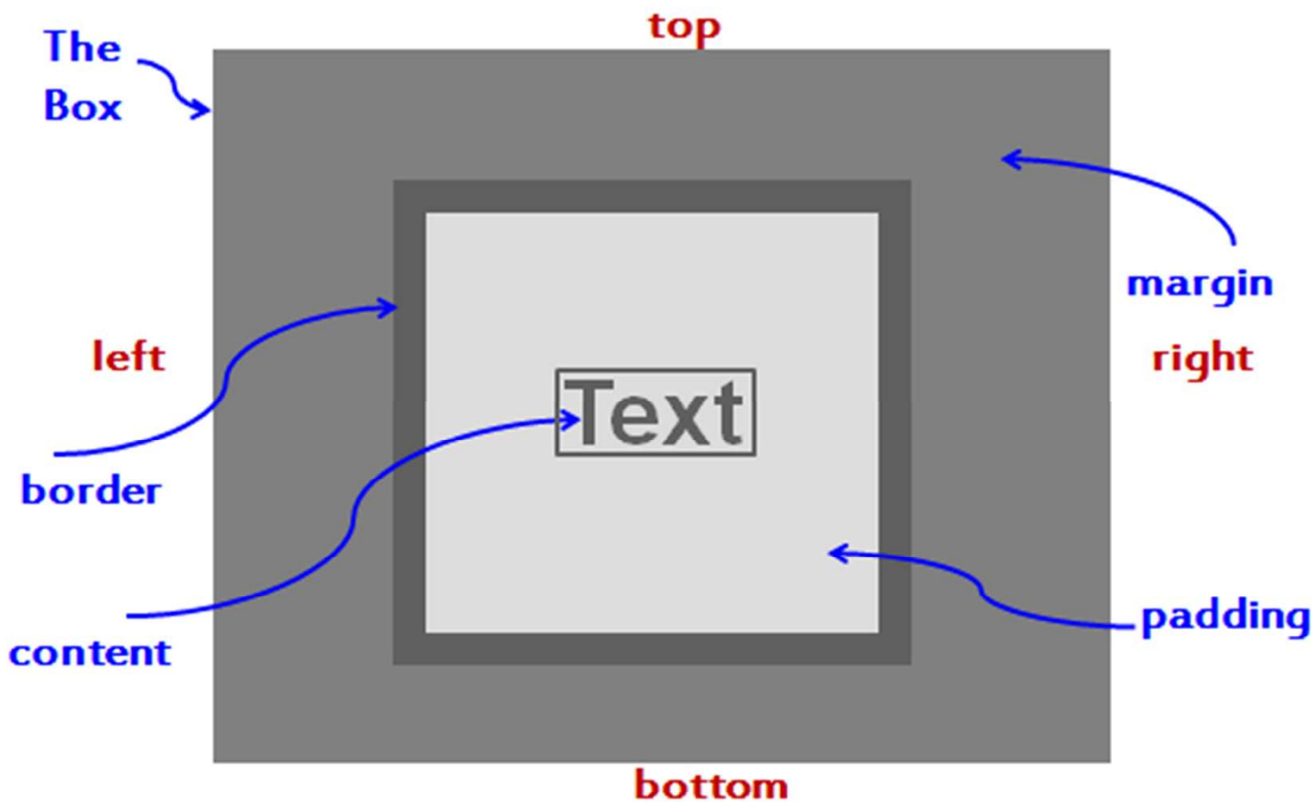
- RGB Color:

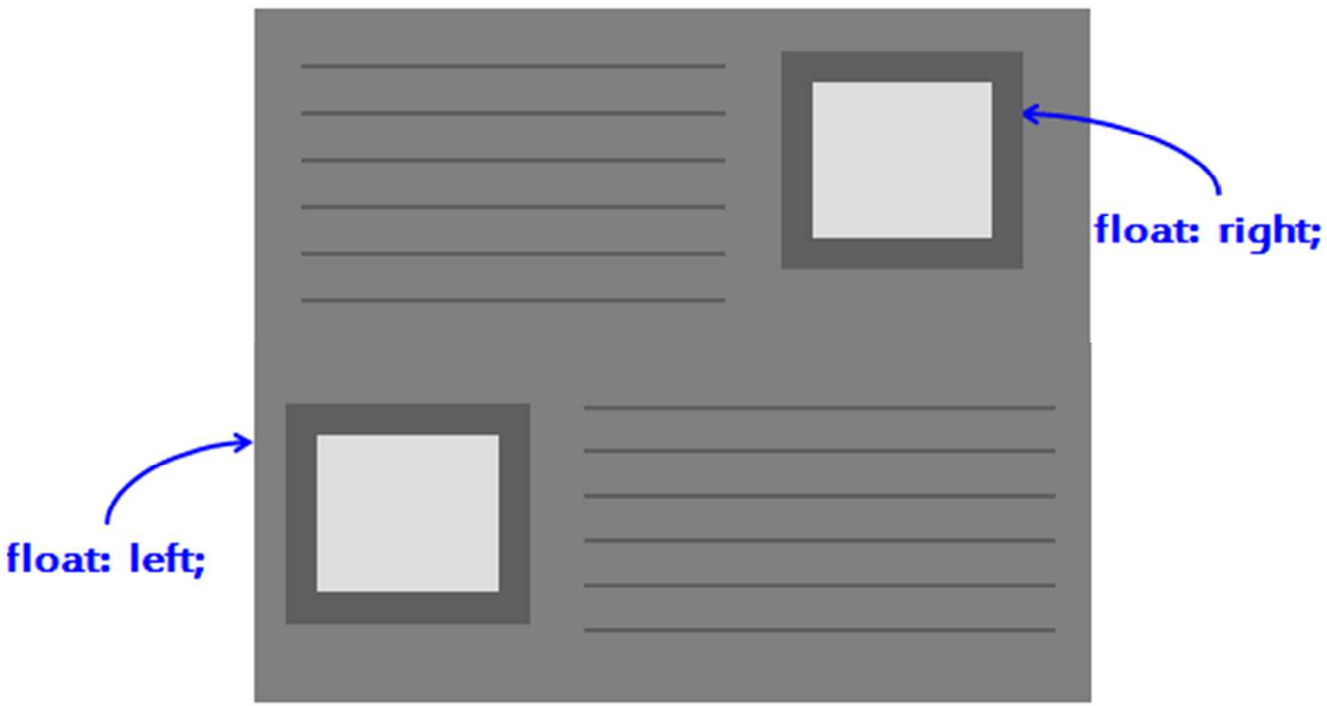
```
p {
    color: #800000;                 // or color: maroon;
    color: rgb(128, 0, 0);          // in functional notation
    color: rgba(128, 0, 0, 0.5);// rgb with transparency
}
```

- HSL:

```
p {
    color: hsl(0, 100%, 25%);          // maroon
    color: hsla(0, 100%, 25%, 0.5);  // maroon with
                                                    //
    transparency
 }
```

CSS Box Model




float: right;
float: left;

Vendor Prefixes

Prefix	Browser	Examples

-ms-	Microsoft	.mySelector {
		-moz-border-image:url(border.png) 30 30 repeat; /* Firefox */
		-webkit-border-image:url(border.png) 30 30 repeat; /* Safari */
		-o-border-image:url(border.png) 30 30 repeat; /* Opera */
		border-image:url(border.png) 30 30 repeat;
		}

-moz-	Mozilla	
-o-	Opera	
-webkit-	Safari and Chrome	

CSS Outline

CSS Outline Properties

- The CSS outline properties allow you to define an outline area around an element's box.
- An outline is a line that is drawn just outside the border edge of the elements. Outlines are generally used to indicate focus or active states of the elements such as buttons, links, form fields, etc.

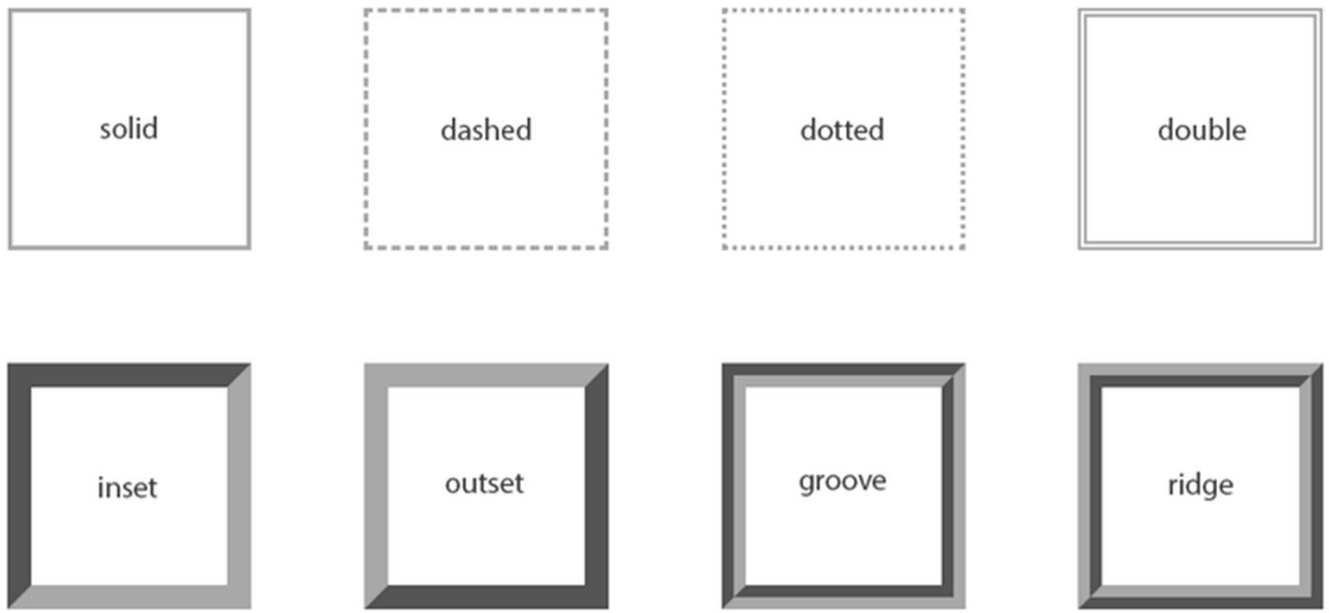
Outlines Vs Borders

An outline looks very similar to the border, but it differs from border in the following ways:

- Outlines do not take up space, because they always placed on top of the box of the element which may cause them to overlap other elements on the page.
- Unlike borders, outlines won't allow us to set each edge to a different width, or set different colors and styles for each edge. An outline is the same on all sides.
- Outlines do not have any impact on surrounding elements apart from overlapping.
- Unlike borders, outlines do not change the size or position of the element.
- Outlines may be non-rectangular, but you cannot create circular outlines.

Understanding the Different Outline Styles

- The outline-style property sets the style of an element's outline such as: solid, dotted, etc.
- The outline-style property can have one of the following values: none, solid, dashed, dotted, double, inset, outset, groove, and ridge.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>Example of CSS outline-style property</title>
    <style>
        p {
            outline-style: double;
            outline-width: 5px;
        }
    </style>
</head>
<body>
    <p>A Sample Outline View.</p>
</body>
</html>
```

Setting the Outline Width ,Color and Outline Shorthand Property

- The outline-width property specifies the width of the outline to be added on an element.

```
p {
    outline-style: dashed;
    outline-width: 10px;
}
```

```
p {
    outline-style: solid;
    outline-color: #0000ff;
}
```

```
p {
    outline: 5px solid  #ff00ff;
}
```

Opacity

CSS Opacity

- The opacity CSS property specifies the transparency of an element.
- Opacity is now a part of the CSS3 specifications, but it was present for a long time. However, older browsers have different ways of controlling the opacity or transparency.

```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS Opacity</title>
<style>
    p {
        opacity: 0.7;
        padding: 10px;
        background: #00ff00;
    }
</style>
</head>
<body>
    <p>This paragraph is 70% opaque (or 30% transparent). Play with <code>opacity</code> value to see how it works.</p>
</body>
</html>
```

CSS Image Opacity

- You can also make transparent images using CSS Opacity.
- The only differences between them are the level of their opacity.

CSS3 Borders

Using CSS3 Borders

- The CSS3 provides two new properties for styling the borders of an element in a more elegant way — the **border-image** property for adding the images to borders, and the **border-radius** property for making the rounded corners without using any images.
- **border-radius** property typically defines the shape of the corner of the outer border edge.

```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Rounded Corners</title>
<style>
    .box {
        width: 300px;
        height: 150px;
        background: #ffb6c1;
        border: 2px solid #f08080;
        border-radius: 20px;
    }
</style>
</head>
<body>
    <div class="box"></div>
</body>
</html>
```

CSS3 Text Effects

- CSS3 contains several new text features.
  - text-shadow
  - word-wrap

CSS3 Text Shadow

- You specify the horizontal shadow, the vertical shadow, the blur distance, and the color of the shadow:


```
h1
{
     text-shadow: 5px 5px 5px #FF0000;
}
```

CSS3 Word Wrapping

CSS3 Word Wrapping

If a word is too long to fit within an area, it expands outside:

This paragraph contains a very long word: thisisaveryveryveryveryveryverylongword. The long word will break and wrap to the next line.

In CSS3, the word-wrap property allows you to force the text to wrap - even if it means splitting it in the middle of a word:

This paragraph contains a very long word: thisisaveryveryveryveryveryveryverylongword. The long word will break and wrap to the next line.

```
p {word-wrap:break-word;}
```

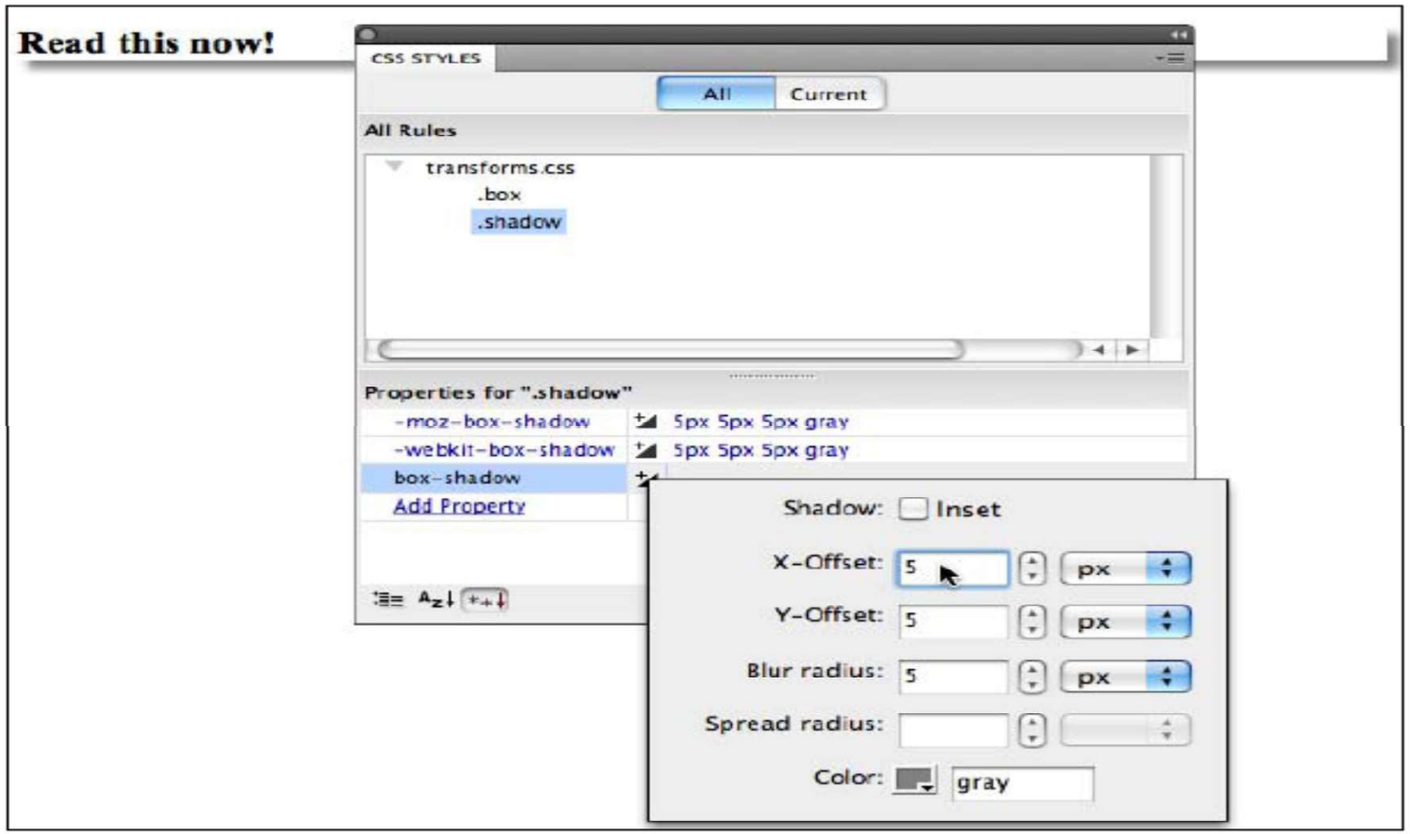
Shadows

- Drop-shadows may well be the most widely applied effect in graphic design. Don't quote me on that, but certainly shadows are a ubiquitous element in many designs, and now they are easy to apply to selected objects using CSS3.
  - Box Shadow
  - Text Shadow
- Both box-shadow and text-shadow effects are defined with a minimum of two parameters: x-offset (vertical distance), and y-offset (horizontal distance). In addition, they usually include a blur parameter (the thickness of the blur gradient) and a color

Box shadow

- As noted, box-shadow effects, are usually defined with four parameters: offset-x (horizontal distance), offset-y (vertical distance), blur (width in pixels), and the color of the shadow.

```
.shadow
{
    box-shadow: 5px 5px 5px gray;
    -webkit-box-shadow: 5px 5px 5px gray;
    -moz-box-shadow: 5px 5px 5px gray;
}
```


Read this now!
CSS STYLES
All
Current
All Rules
transforms.css
.box
.shadow
Properties for ".shadow"
-moz-box-shadow
5px 5px 5px gray
-webkit-box-shadow
5px 5px 5px gray
box-shadow
Add Property
Shadow:
Inset
X-Offset:
5
px
Y-Offset:
5
px
Blur radius:
5
px
Spread radius:
Color:
gray

CSS3 2D Transforms

- There are four main CSS translation functions:
  - Translate
  - Rotate
  - Scale
  - Skew
  - Matrix

CSS3 Transforms

- Using CSS3 transform, we can move, scale, turn, spin, and stretch elements.
- A transform is an effect that lets an element change shape, size and position.
- You can transform your elements using 2D or 3D transformation.

```
div
{
 transform: rotate(30deg);
 -ms-transform: rotate(30deg); /* IE 9 */
 -webkit-transform: rotate(30deg); /* Safari and Chrome */
 }
```

The translate() Method

- With the translate() method, the element moves from its current position, depending on the parameters given for the left (X-axis) and the top (Y-axis) position:


```
div
 {
 transform: translate(50px,100px);
 -ms-transform: translate(50px,100px); /* IE 9 */
 -webkit-transform: translate(50px,100px); /* Safari and Chrome */
 }
```

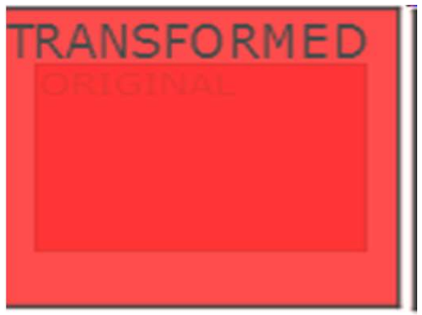
The rotate() Method

- With the rotate() method, the element rotates clockwise at a given degree. Negative values are allowed and rotates the element counter-clockwise


```
div
  {
  transform: rotate(30deg);
  -ms-transform: rotate(30deg); /* IE 9 */
  -webkit-transform: rotate(30deg); /* Safari and Chrome */
  }
```

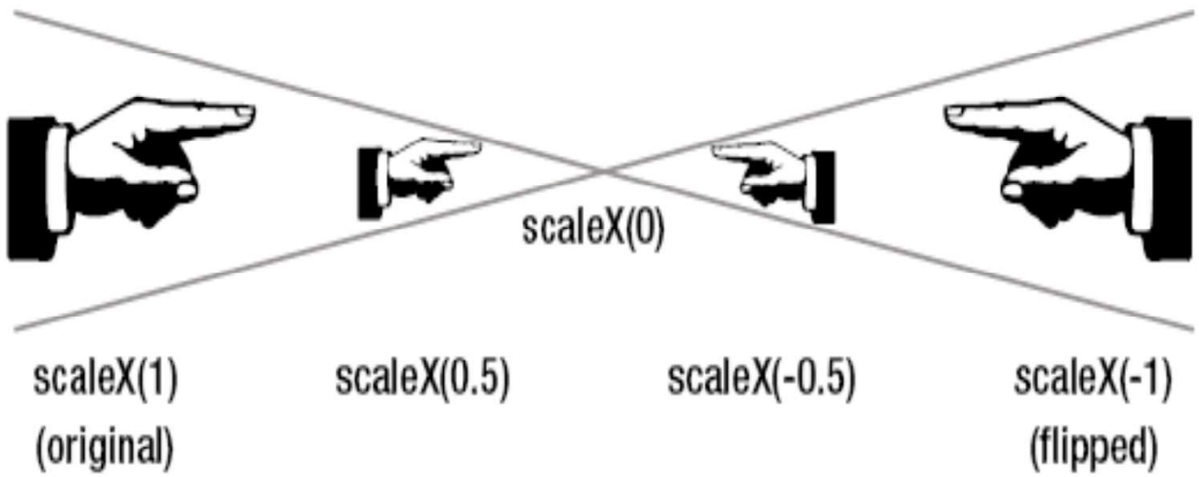
The scale() Method

- With the scale() method, the element increases or decreases the size, depending on the parameters given for the width (X-axis) and the height (Y-axis):



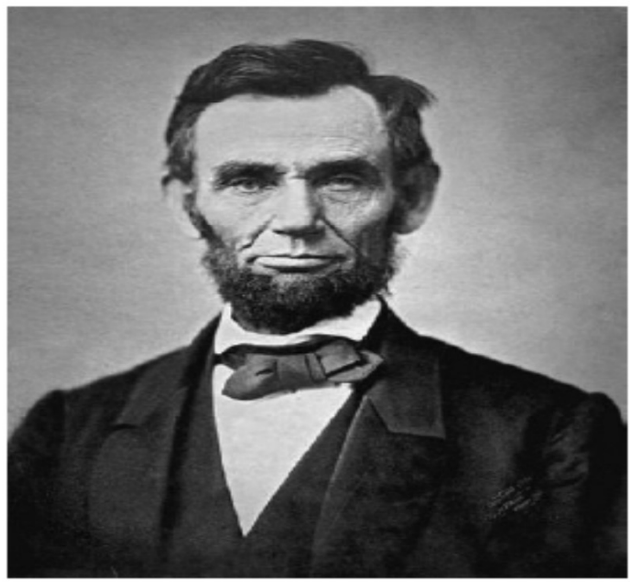

```
div
  {
  transform: scale(2,4);
  -ms-transform: scale(2,4); /* IE 9 */
  -webkit-transform: scale(2,4); /* Safari and Chrome */
  }
```

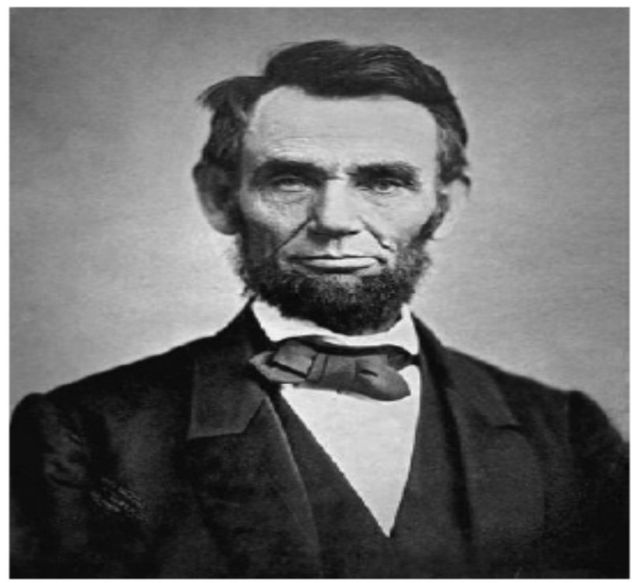
Flipping Images with scaleX



Flipping an Image with an Inline Transform Style

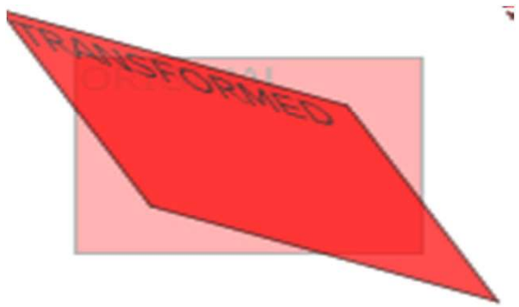
```
<img src="lincoln.jpg" alt="Abraham Lincoln, 1863" style="width:
389px; height: 480px; -moz-transform: scaleX(-1); -o-transform:
scaleX(-1); -ms-transform: scaleX(-1); -webkit-transform:
scaleX(-1); transform: scaleX(-1);">
```





The skew() Method

- With the skew() method, the element turns in a given angle, depending on the parameters given for the horizontal (X-axis) and the vertical (Y-axis) lines:




```
div
  {
  transform: skew(30deg,20deg);
  -ms-transform: skew(30deg,20deg); /* IE 9 */
  -webkit-transform: skew(30deg,20deg); /* Safari and Chrome */
  }
```

CSS3 3D Transforms

- CSS3 allows you to format your elements using 3D transforms.
  - rotateX()
  - rotateY()

The rotateX() Method

- With the rotateX() method, the element rotates around its X-axis at a given degree.


```
div
  {
  transform: rotateX(120deg);
  -webkit-transform: rotateX(120deg); /* Safari and Chrome */
  }
```

The rotateY() Method

- With the rotateY() method, the element rotates around its Y-axis at a given degree.

```
div
 {
 transform: rotateY(130deg);
 -webkit-transform: rotateY(130deg); /* Safari and Chrome */
 }
```

CSS3 Transitions

- With CSS3, we can add an effect when changing from one style to another, without using Flash animations or JavaScripts.

How It Works ?

- CSS3 transitions are effects that let an element gradually change from one style to another.
  - To do this, you must specify two things:
    - Specify the CSS property you want to add an effect to
    - Specify the duration of the effect
- The effect will start when the specified CSS property changes value. A typical CSS property change would be when a user mouse-over an element:

Single Effect and Multiple changes

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

- To add a transitional effect for more than one style, add more properties, separated by commas:

```
div
{
transition: width 2s, height 2s, transform 2s;
-webkit-transition: width 2s, height 2s, -webkit-transform 2s;
}
```

Transition Properties

Property	Description	CSS
transition	A shorthand property for setting the four transition properties into a single property	3
transition-property	Specifies the name of the CSS property to which the transition is applied	3
transition-duration	Defines the length of time that a transition takes. Default 0	3
transition-timing-function	Describes how the speed during a transition will be calculated. Default "ease"	3
transition-delay	Defines when the transition will start. Default 0	3

Media Queries

Media Queries

- CSS `@media` rule:

```
@media screen and (max-width: 750px){

    /* put your styles here */

}
```

media feature

media type

- `<link>` `media` attribute and in `@import`:

```
<link rel="stylesheet" media="screen and (max-width:
    468px)" href="mobile.css">
<link rel="stylesheet" media="screen and (min-width:
    768px)" href="screen.css">
@import url(smartphone.css) screen and (max-width:
    468px);
```

Media Queries

- Using the `only` keyword:

```
@media only screen and (min-width: 480px)
```

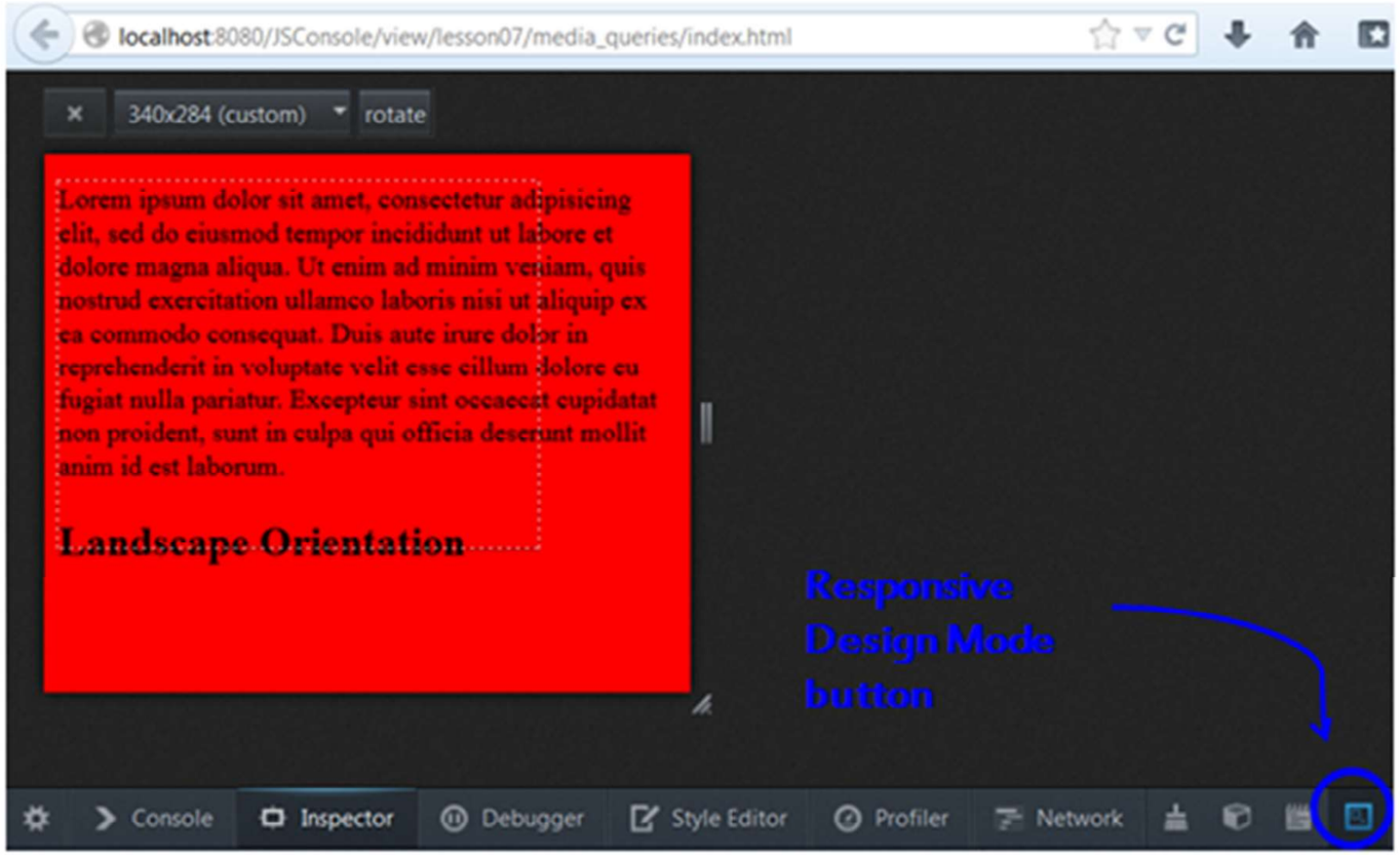
- Using the `not` keyword:

```
not screen and (max-width: 768px)
```

- Adjusting the viewport:

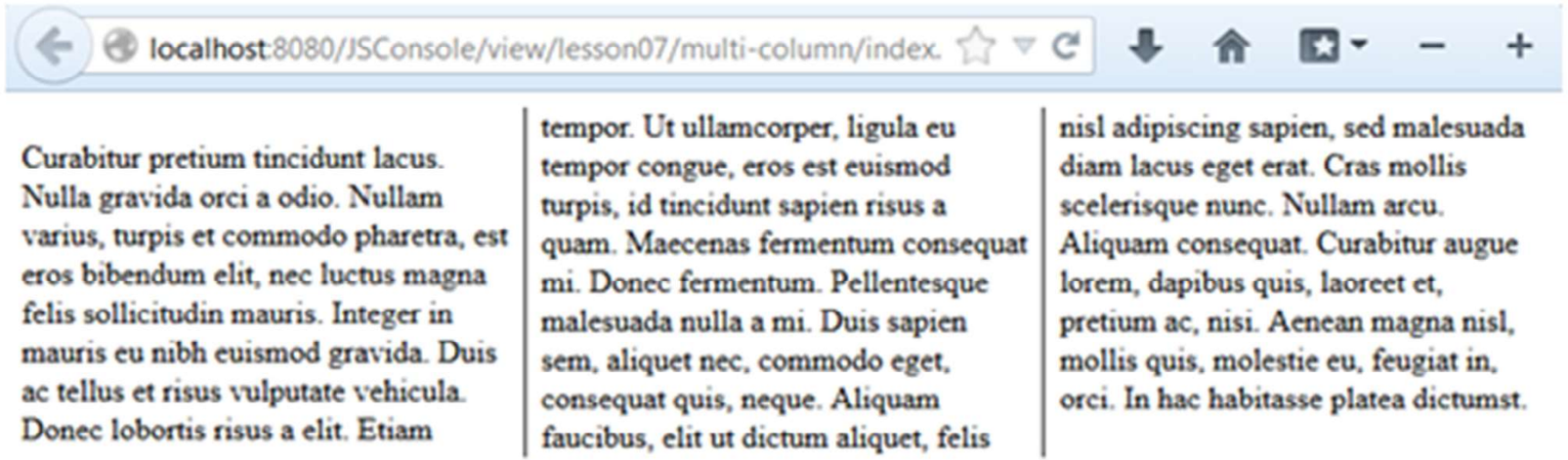
```
<meta name="viewport" content="width=device-width,
    initial-scale=1.0">
```

Example: Media Queries



Multi Column

Multicolumn




```
.columns {
    column-width: 13em;
    column-gap: 1em;
    column-rule: 1px solid #000;
}
```

width, style, and color

Modifying Styles with JS

Modifying Styles with JavaScript

- HTML:

```
<body>
    <button onclick="changeBox()">Change box style</button>
    <div id="box">
      <h1>I am in a box</h1>
    </div>
</body>
```

- JS function:

```
var changeBox = function (event) {
    console.log("changeBox function called");
    var box = document.getElementById("box");
    box.style.backgroundColor = "#ff9966";
    box.style.width = "280px";
    box.style.float = "left";
    box.style.borderStyle = "dotted";
};
```

Summary

In this lesson, you should have learned how to:

- Apply CSS styles to HTML documents
- Use CSS3 features to add dynamic styles to elements with events
- Use Media Queries and media data to adapt styles to different screens
- Use JavaScript to add and remove styles from elements