Introduction to CSS

What is CSS?

- CSS ("Cascading Style Sheets") determines how the elements in our XHTML documents are displayed and formatted.
- By using CSS, we separate the content of a web page from the presentation (format and styling) of that content.
- CSS enables us to make all pages of our website look similar and consistent.
- The power of CSS is that it allows us to make site-wide formatting changes by making edits to a single file.

In this course, we will be learning version 2.1 of CSS. A newer version, known as CSS3, is being increasingly adopted by web designers and web browser manufacturers.

What You Can Do with CSS

- There are lot more things you can do with CSS.
- You can easily apply same style rules on multiple elements.
- You can control the presentation of multiple pages of a website with a single style sheet.
- You can present the same page differently on different devices.
- You can style dynamic states of elements such as hover, focus, etc. that isn't possible otherwise.
- You can change the position of an element on a web page without changing the markup.

What You Can Do with CSS

- You can alter the display of existing HTML elements.
- You can transform elements like scale, rotate, skew, etc. in 2D or 3D space.
- You can create animations and transitions effects without using any JavaScript.
- You can create print friendly version of your web pages.
- The list does not end here, there are many other interesting things that you can do with CSS

How Browsers Process CSS

- A web browser will process all CSS code it encounters, even if it is from all three methods.
- For example, an external style sheet could define the font of a heading, an internal style sheet could specify the font size of the heading, and an inline style could italicize the heading. All three would be applied.
- Sometimes a browser will receive conflicting instructions from the CSS code. For example, what if each of the above CSS sources specified a different color for the heading text?

Browsers need a consistent way of settling these formatting conflicts in a consistent fashion. That is where the "cascade" of cascading style sheets comes into effect.

What Does "Cascading" Mean?

We use the term "cascading" because there is an established order of priority to resolve formatting conflicts:

1. Inline style (highest priority)
2. Internal style sheet (second priority)
3. External style sheet (third priority)
4. Web browser default (only if not defined elsewhere)

For each XHTML element, the browser will see which styles are defined inline and from internal and external style sheets. For any conflicts detected, it will use this priority system to determine which format to display on the page.

In the prior example, the heading text would display in the color specified by the inline style, which outranks all the others.

If multiple, conflicting styles are defined in the same style sheet, only the final one will be applied. Be careful, as this is another common mistake committed by beginners.

Three Ways to Use CSS

We can add CSS code in any combination of three different ways:

1. Inline Style - CSS code is placed directly into an XHTML element within the <body> section of a web page.
2. Internal Style Sheet - CSS code is placed into a separate, dedicated area within the <head> section of a web page.
3. External Style Sheet - CSS code is placed into a separate computer file and then linked to a web page.

Let's take a look now at examples of each of these methods.

Inline Style

To define an inline CSS style, we simply add the **style** attribute to an XHTML element with the CSS declaration as the attribute value:

```
<h2 style="color:red;">CAUTION: Icy Road Conditions</h2>
<h2>Please Slow Down!</h2>
```

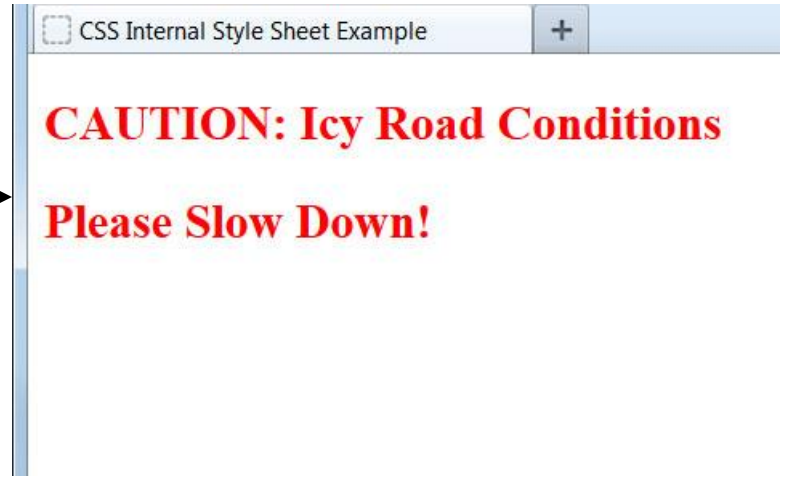
An inline style declaration is highly specific and formats just one element on the page. No other elements, including other <h2> elements on the page, will be affected by this CSS style.

Since inline styles have limited scope and do not separate content from presentation, their use is generally discouraged. We won't be using inline styles much in this class.

Internal Style Sheet

To use an internal CSS style sheet, we add a <style> section within the <head> of the page. All our CSS declarations go within this section:

```
<head>
  ...
  <style type="text/css">
    h2 {color:red;}
  </style>
</head>
<body>
  <h2>CAUTION: Icy Road Conditions</h2>
  <h2>Please Slow Down!</h2>
</body>
```



Styles declared in the internal style sheet affect all matching elements on the page. In this example, all <h2> page elements are displayed in the color red.

Since formatting declarations are entirely in the <head> section, away from the actual page content, internal CSS style sheets do a much better job than inline styles at separating content from presentation.

External Style Sheet

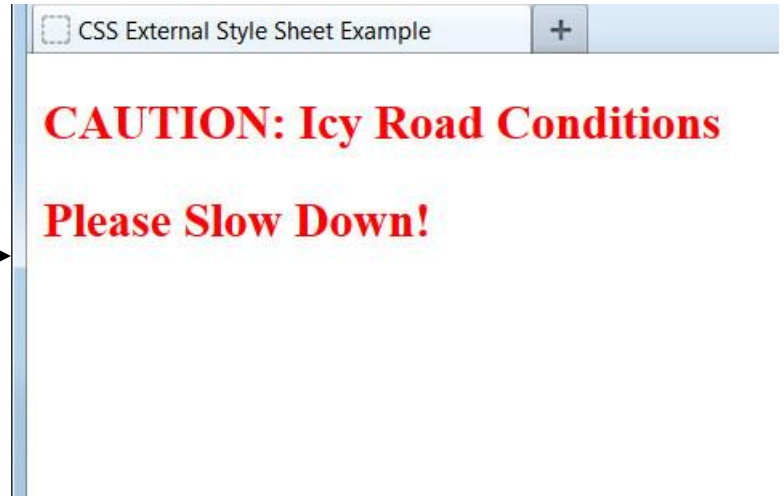
To use an external CSS style sheet, we create a new file (with a .css extension) and write our style declarations into this file. We then add a <link> element into our HTML file, right after the opening <head> tag:

style.css (separate file):

```
h2 {color:red;}
```

example.html file:

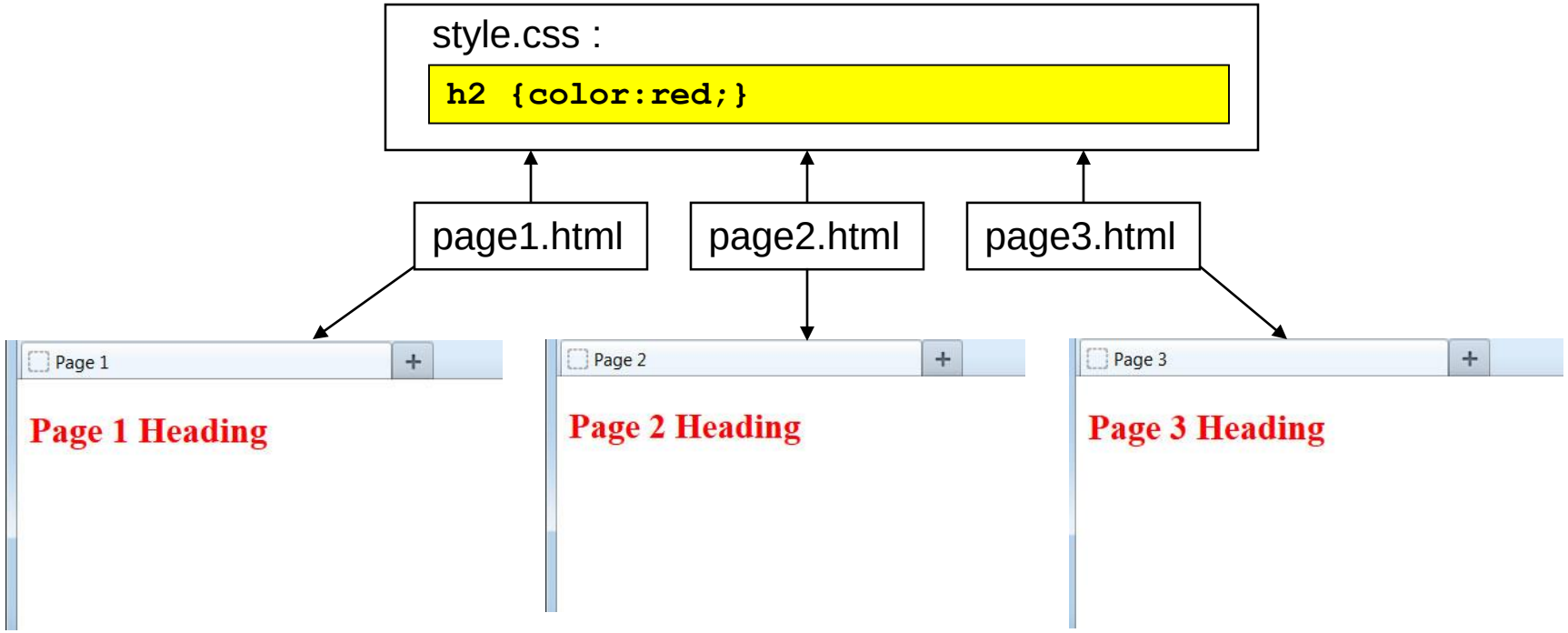
```
<head>
  <link rel="stylesheet" type="text/css"
   href="style.css" />
  ...
</head>
<body>
  <h2>CAUTION: Icy Road Conditions</h2>
  <h2>Please Slow Down!</h2>
</body>
```



The <link> element instructs the browser to load the external file specified by the href attribute and to apply the CSS style declarations contained there.

Benefit of External Style Sheet

The real power of using an external style sheet is that multiple web pages on our site can link to the same style sheet:



Styles declared in an external style sheet will affect all matching elements on all web pages that link to the style sheet. By editing the external style sheet, we can make site-wide changes (even to hundreds of pages) instantly.

Internal vs. External Style Sheets

Internal Style Sheets:

- are appropriate for very small sites, especially those that have just a single page.
- might also make sense when each page of a site needs to have a completely different look.

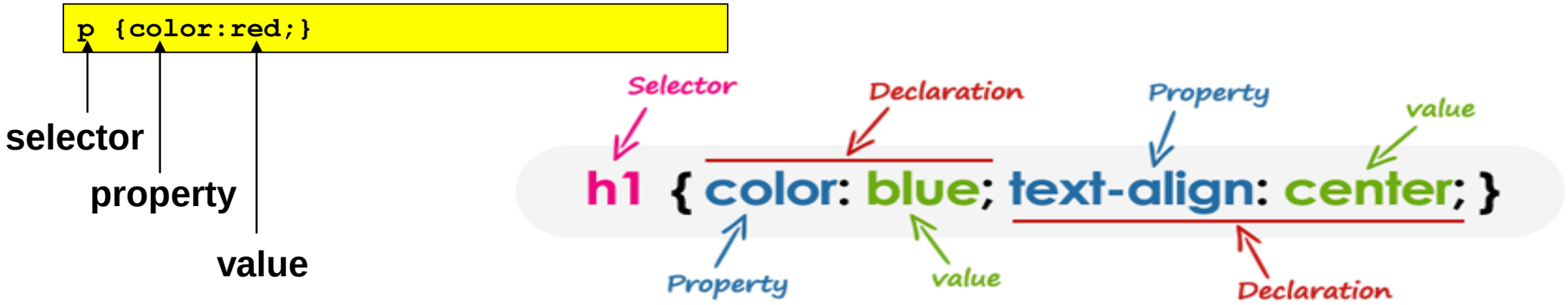
External Style Sheets:

- are better for multi-page websites that need to have a uniform look and feel to all pages.
- make for faster-loading sites (less redundant code).
- allow designers to make site-wide changes quickly and easily.

External style sheets create the furthest separation between content and presentation. For this reason - and the others listed above - we'll consider external style sheets to be the best option when creating a new site.

CSS Terminology and Syntax:

Now let's take a closer look at how we write CSS code. The correct syntax of a CSS declaration is: **selector {property:value;}**



Internal and external style sheets use this identical CSS syntax. Internal style sheets must use the opening and closing <style> tags to surround the CSS code, while external style sheets do not use the <style> element.

A semicolon must be placed after each CSS declaration. Omitting this semicolon is the single most common mistake made by those learning CSS.

What is Selector?

- A CSS selector is a pattern to match the elements on a web page. The style rules associated with that selector will be applied to the elements that match the selector pattern.
- Selectors are one of the most important aspects of CSS as they allow you to target specific elements on your web page in various ways so that they can be styled.
- Several types of selectors are available in CSS

- Universal Selector *** { margin: 0; padding: 0;}**
- Element Type Selector **p {color: blue;}**
- ID Selector **#error {color: red;}**
- Class Selector **.blue {color: blue;}**
- Descendant Selector **ul.menu li a {text-decoration: none;}**
- Child Selector **ul > li { list-style: square;}**
- Grouping Selector **h1, h2, h3 { font-weight: normal; }**

Universal Selector

- The universal selector, denoted by an asterisk (*), matches every single element on the page.

- The universal selector may be omitted if other conditions exist on the element. This selector is often used to remove the default margins and paddings from the elements for quick testing purpose.

```
<html>
<head>
<title>Example of CSS universal selector</title>
  <style>
    * {margin: 0; padding: 0;}
  </style>
</head>
<body>
  <h1>This is heading</h1>
  <p>This is a paragraph.</p>
</body>
</html>
```

Element Type Selector

- An element type selector matches all instance of the element in the document with the corresponding element type name.
- The style rules inside the p selector will be applied on every <p> element (or paragraph) in the document and color it blue, regardless of their position in the document tree

```
<html>
<head>
<title>Example of CSS element type selector</title>
  <style>
    h1 {color: red;}
    p {color: blue; }
  </style>
</head>
<body>
  <h1>This is heading</h1>
  <p>This is a paragraph.</p>
</body>
</html>
```

ID Selector

- The id selector is used to define style rules for a single or unique element.
- The id selector is defined with a hash sign (#) immediately followed by the id value.

```
<html>
<head>
<title>Example of CSS id selector</title>
  <style>
    #error {
        color: #ff0000;
    }
```

```
</style>
</head>
<body>
  <p id="error">This is a warning!</p>
</body>
</html>
```

Class Selector

- The class selectors can be used to select any HTML element that has a class attribute. All the elements having that class will be formatted according to the defined rule.
- The class selector is defined with a period sign (.) immediately followed by the class value.

```
<html>
<head>
<title>Example of CSS class selector</title>
  <style>
    .blue {
        color: #0000ff;
    }
  </style>
</head>
```

```
<body>
  <h1 class="blue">This is a heading</h1>
  <p class="blue">This is a paragraph.</p>
  <p>This is another paragraph.</p>
</body>
</html>
```

Descendant Selector

- You can use these selectors when you need to select an element that is the descendant of another element, for example, if you want to target only those anchors that are contained within an unordered list, rather than targeting all anchor elements

```
<html>
<head>
<title>Example of CSS Descendant Selectors</title>
<style>
    h1 em {color: green;}
    ul.menu {padding: 0; list-style: none;}
    ul.menu li{display: inline;}
    ul.menu li a {margin: 10px;
            text-decoration: none;}
</style></head><body>
        <h1>This is a <em>heading</em></h1>
        <ul class="menu">
            <li><a href="#">Home</a></li>
            <li><a href="#">About</a></li>
            <li><a href="#">Services</a></li>
            <li><a href="#">Contact</a></li>
        </ul>
</body></html>
```

Child Selector

- A child selector is used to select only those elements that are the direct children of some element.
- A child selector is made up of two or more selectors separated by a greater than symbol (>). You can use this selector, for instance, to select the first level of list elements inside a nested

```
<html>
<head>
<title>Example of CSS Child Selectors</title>
<style>
        ul > li {list-style: square;}
        ul > li ol {list-style: none;}
</style>
</head><body>
<ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">About</a></li>
    <li><a href="#">Services</a>
        <ol>
            <li><a href="#">Design</a></li>
            <li><a href="#">Development</a></li>
        </ol>
    </li>
    <li><a href="#">Contact</a></li>
</ul>
</body></html>
```

Grouping Selector

- Often several selectors in a style sheet share the same style rules declarations. You can group them into a comma-separated list to minimize the code in your style sheet. It also prevents you from repeating the same style rules over and over again.

```
<html >
<head>
<title>Example of CSS Grouping Selectors</title>
<style>
        h1, h2, h3 {font-weight: normal;}
        h1 {font-size: 36px;}
        h2 {font-size: 28px;}
        h3 {font-size: 22px;}
</style>
```

```
</head>
<body>
        <h1>This is a heading of level 1</h1>
        <h2>This is a heading of level 2</h2>
        <h3>This is a heading of level 3</h3>
</body>
</html>
```

Writing Comments in CSS

Comments are usually added with the purpose of making the source code easier to understand. It may help other developer (or you in the future when you edit the source code) to understand what you were trying to do with the CSS. Comments are significant to programmers but ignored by browsers.

A CSS comment begins with /*, and ends with */, as shown in the example below:

```
/* This is a CSS comment */
h1 { color: blue; text-align: center; }
 /* This is a multi-line CSS comment that spans across more than one line */
p { font-size: 18px; text-transform: uppercase; }
```

Case Sensitivity in CSS

CSS property names and many values are not case-sensitive. Whereas, CSS selectors are usually case-sensitive, for instance, the class selector .maincontent is not the same as .mainContent.

Therefore, to be on safer side, you should assume that all components of CSS rules are case-sensitive.

```
/* This is a CSS comment */
.maincontent { color: blue; text-align: center; }
 /* This is a multi-line CSS comment that spans across more than one line */
.mainContent { font-size: 18px; text-transform: uppercase; }
```

Setting Multiple Properties

We can define as many properties as we wish for a selector:

```
p {color:red;font-style:italic;text-align:center;}
```

In this example, all text within paragraph elements will show in red italics that is centered on the page.

```
p {
  color: red;
  font-style: italic;
  text-align: center;
}
```

Just as with HTML, browsers ignore space characters in CSS code. Many designers take advantage of this fact by placing the opening and closing curly brackets on their own dedicated lines. Each of the property and value pairings are placed on their own indented line, with a space after the colon. This makes the code far easier to read.

CSS Color

The color property defines the text color (foreground color in general) of an element.

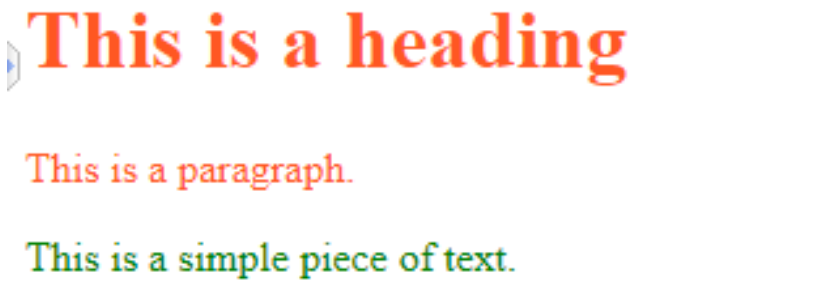
For instance, the color property specified in the body selector defines the default text color for the whole page.

Colors in CSS most often specified in the following formats:
a color keyword - like "red", "green", "blue", "transparent", etc.
a HEX value - like "#ff0000", "#00ff00", etc.
an RGB value - like "rgb(255, 0, 0)"

```
h1 { color: red; }
h1 { color: #ffa500; }
h1 { color: rgb(255, 165, 0); }
p { color: rgb(0, 255, 0); }
```

CSS Color Example

```
<html>
<head>
<title>Setting Text Color in CSS</title>
<style>
    body {
        color: #ff5722;
    }
    .text-green {
        color: #008000;
    }
</style>
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <div class="text-green">This is a simple piece of
   text.</div>
</body>
</html>
```



CSS Background Properties

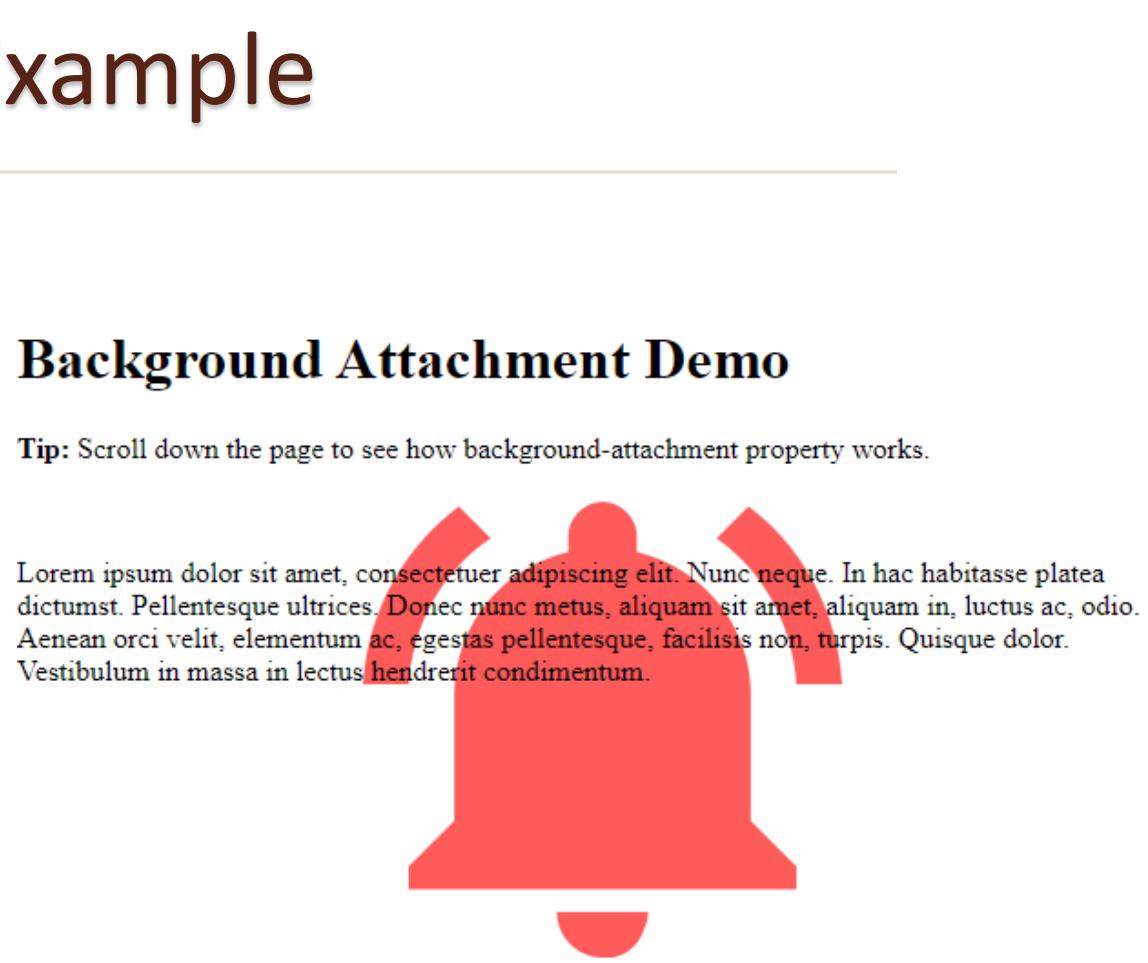
Background plays an important role in the visual presentation of a web page.

CSS provide several properties for styling the background of an element, including coloring the background, placing images in the background and managing their positioning

Property	Some Possible Values
background-color:	#ffccee;
background-image:	url("images/tile.png");
background-repeat:	repeat/repeat-x/repeat-y/no-repeat;
background-attachment:	scroll | fixed;
background-position:	top left/ top center / top right/ x-% y-%/x-pos y- pos
background-size:	length / % | auto | contain | cover

CSS Background Example

```
<html>
<head>
<title>Fixing background-attachment in CSS</title>
<style>
    body {
        background-image:
url("/examples/images/bell.png");
        background-repeat: no-repeat;
        background-position: center;
        background-attachment: fixed;
    }
</style>
</head>
<body>
……….
</body>
</html>
```



CSS font Properties

Choosing the right font and style is very crucial for the readability of text on a page.
CSS provide several properties for styling the font of the text, including changing their face, controlling their size and boldness, managing variant, and so on.

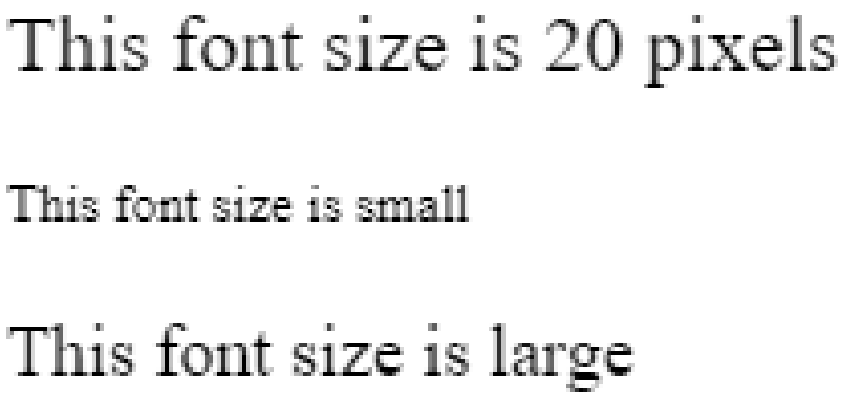
Property	Some Possible Values
font-family:	serif, sans-serif, monospace, cursive and fantasy
font-style:	normal | italic | oblique | inherit
font-weight:	normal | bold | bolder | lighter | 100 | 200 …..900
font-size:	--% | --em | --px | small | medium | large | larger | smaller
font-variant:	normal | small-caps | inherit

CSS font example

```
<html>
  <head>
  </head>
  <body>
    <p style = "font-size:20px;">
      This font size is 20 pixels
    </p>

    <p style = "font-size:small;">
      This font size is small
    </p>

    <p style = "font-size:large;">
      This font size is large
    </p>
  </body>
</html>
```



https://www.w3.org/TR/2018/REC-css-fonts-3-20180920/

CSS border Properties

The CSS border properties allow you to define the border area of an element's box.

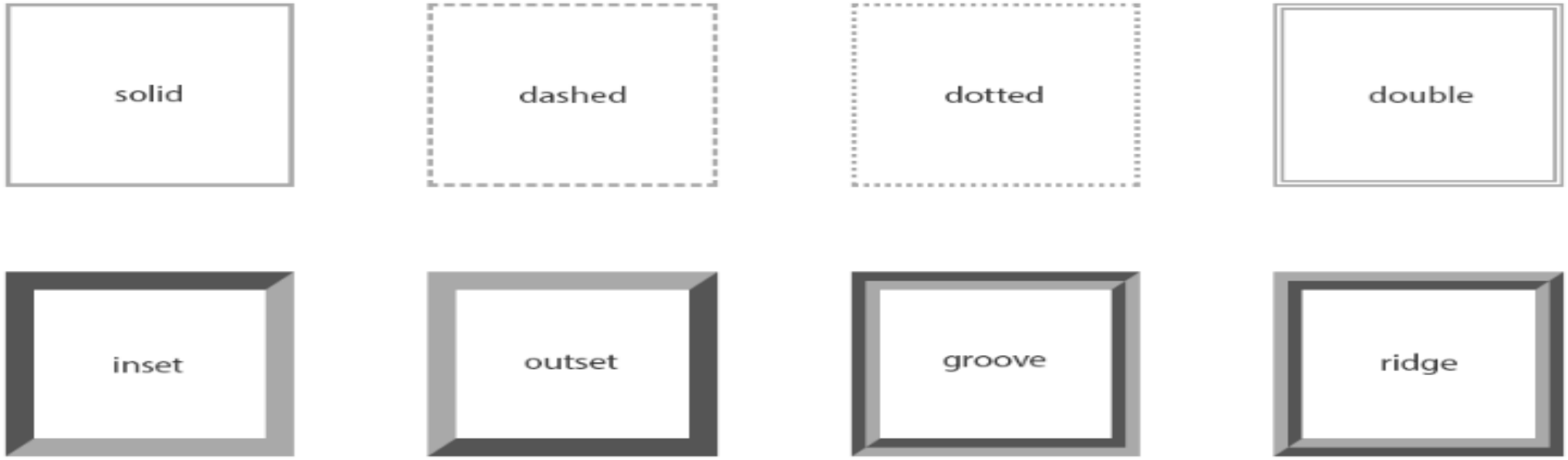
Borders appear directly between the margin and padding of an element. The border can either be a predefined style like, solid line, dotted line, double line, etc. or an image.

Property
border-width:
border-style:
Border-color:

Understanding the Border Styles

The border-style property sets the style of a box's border such as: solid, dotted, etc. It is a shorthand property for setting the line style for all four sides of the elements border.

The border-style property can have the following values: none, hidden, solid, dashed, dotted, double, inset, outset, groove, and ridge. Now, let's take a look at the following illustration, it gives you a sense of the differences between the border style types.



CSS border example

```
<html>
<head>
<style>
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid double;}
</style>
</head>
<body>
```

```
<h2>The border-style Property</h2>
<p>This property specifies what kind of border to
display:</p>

<p class="dotted">A dotted border.</p>
<p class="dashed">A dashed border.</p>
<p class="solid">A solid border.</p>
<p class="double">A double border.</p>
<p class="groove">A groove border.</p>
<p class="ridge">A ridge border.</p>
<p class="inset">An inset border.</p>
<p class="outset">An outset border.</p>
<p class="none">No border.</p>
<p class="hidden">A hidden border.</p>
<p class="mix">A mixed border.</p>

</body>
</html>
```

CSS border example

The border-style Property

This property specifies what kind of border to display:

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border.

A ridge border.

An inset border.

An outset border.

No border.

A hidden border.

A mixed border.

The Border Shorthand Property

The border CSS property is a shorthand property for setting one or more of the individual border properties border-width, border-style and border-color in a single rule.

```
p {
    border: 5px solid #00ff00;
}
```

```
p {
        border: 5px solid #00ff00;
        background: yellow;
        padding: 20px;
        margin: 20px;
    }
```

This is a paragraph with border.

CSS Rounded Borders

The border-radius property is used to add rounded borders to an element:

```
p.normal {
  border: 2px solid red;
}

p.round1 {
  border: 2px solid red;
  border-radius: 5px;
}

p.round2 {
  border: 2px solid red;
  border-radius: 8px;
}

p.round3 {
  border: 2px solid red;
  border-radius: 12px;
}
```

```
<body>

<h2>The border-radius Property</h2>
<p>This property is used to add rounded
borders to an element:</p>

<p class="normal">Normal border</p>
<p class="round1">Round border</p>
<p class="round2">Rounder border</p>
<p class="round3">Roundest border</p>

</body>
```

CSS Rounded Borders

The border-radius Property

This property is used to add rounded borders to an element:

Normal border

Round border

Rounder border

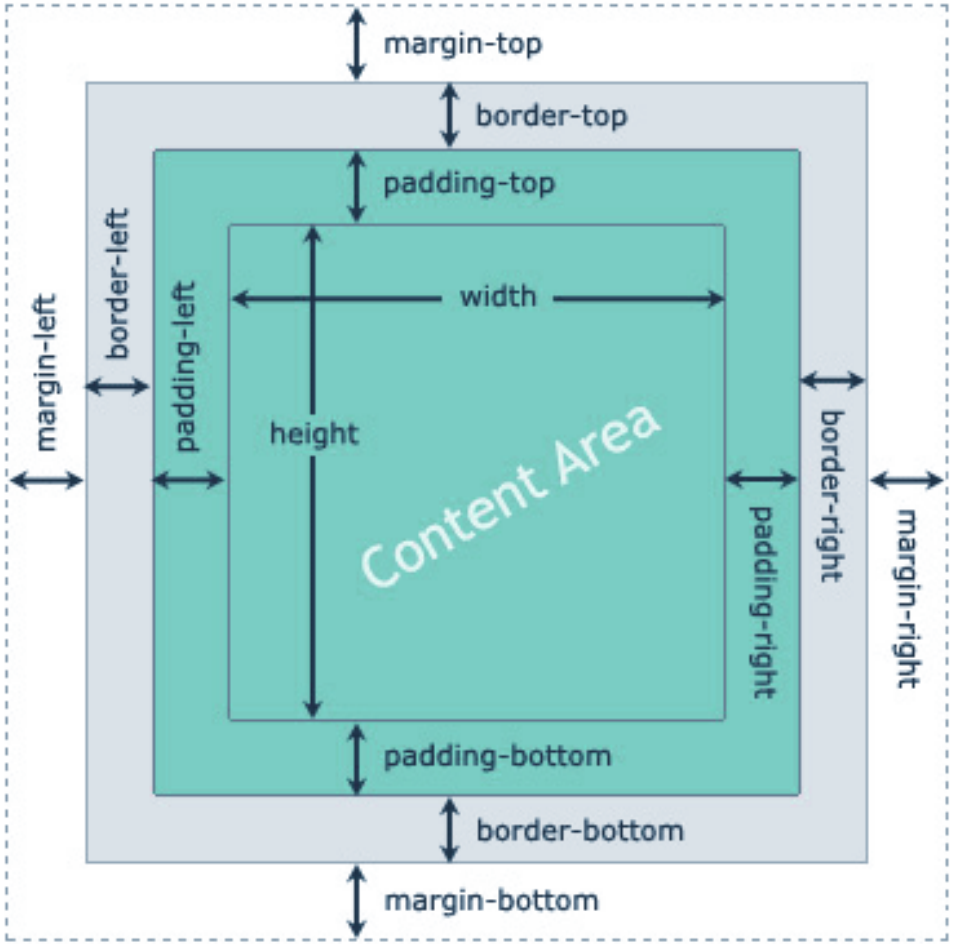
Roundest border

CSS Box Model

Every element that can be displayed on a web page is comprised of one or more rectangular boxes.

CSS box model typically describes how these rectangular boxes are laid out on a web page. These boxes can have different properties and can interact with each other in different ways, but every box has a ***content area*** and optional surrounding ***padding***, ***border***, and ***margin areas***.

The following diagram demonstrates how the width, height, padding, border, and margin CSS properties determines how much space an element can take on a web page.



CSS Box Model

```
<html>
<head>
<style>
div {
  background-color: lightgrey;
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
</style>
</head>
<body>

<h2>Demonstrating the Box Model</h2>
```

```
<p>The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.</p>
<div>This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</div>
</body>
</html>
```

CSS Box Model

Demonstrating the Box Model

The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.