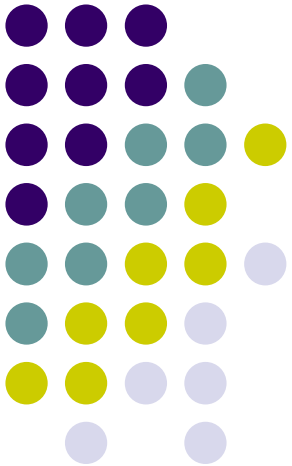
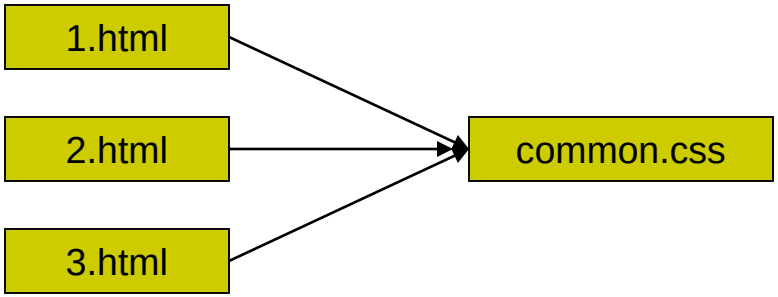
Cascading Style Sheets (CSS)

An Introduction



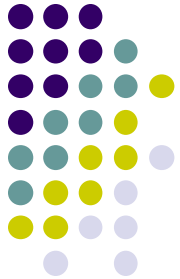
Style Sheets

- Style sheets are implemented by using Cascading Style Sheets (CSS)
- Keep all styling details separate and refer to this file from our HTML document.
- Multiple HTML documents can make use of the same CSS file, so that all of them can have same look-and-feel as defined in the CSS file.



Style Sheet Concept

Cascading Style Sheet an Introduction



- With CSS, your HTML documents can be displayed using different output styles and tells the browser how the element is to be presented to the user.
- It is a simple mechanism for adding style (e.g. fonts, colors, spacing) to Web documents
- Allows elements to “inherit” styles from parent elements.
- Styles are normally stored in **Style Sheets.**
- **External Style Sheets** can save you a lot of work
- External Style Sheets are stored in .**CSS files**

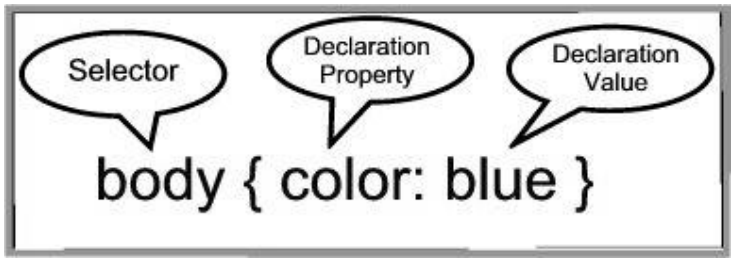
Why CSS

- By editing a single CSS file, you can make site wide design changes in seconds.
- CSS lets you output to multiple formats quickly.
- External CSS files are cached by browsers, improving load time.
- CSS eliminates the need for messy code -- namely font tags, spacer gifs and nested tables. This improves load time and makes developers' lives easier.
- CSS lets you do things normal HTML doesn't. Examples: better font control, absolute positioning, nifty borders.
- Practical use of CSS encourages proper HTML structure, which will improve accessibility and search engine placement.
- If you want valid XHTML Strict you have to use it anyway.

CSS Syntax

- The CSS syntax is made up of three parts: a selector, a property and a value:

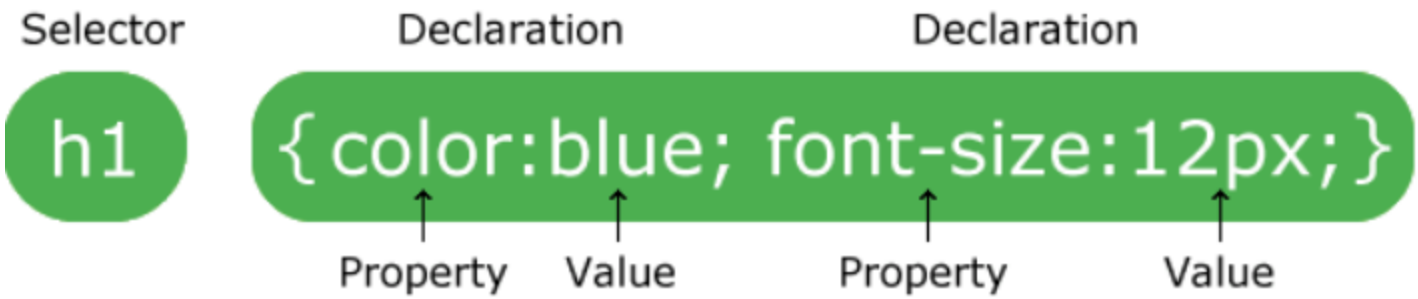
selector {property:value}



Eg. body {color: black}

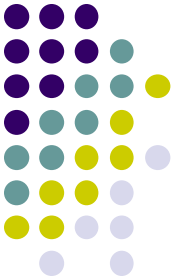
p {font-family:"sans serif"}

p {text-align:center;color:red}



All together

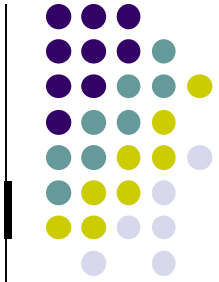
```
p
{
text-align: center;
color: black;
font-family: arial
}
```



- In the example below we have grouped all the header elements. All header elements will be displayed in green text color:

```
h1,h2,h3,h4,h5,h6 { color: green }
```

How to Insert a Style Sheet



- When a browser reads a style sheet, it will format the document according to it. There are three ways of inserting a style sheet:
  - External Style Sheet
  - Internal Style Sheet
  - Inline Styles
- If a document contains both external and internal style sheet then the values will be inherited from the more specific style sheet.

- An *external style* sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file.
  - Preferred - because two people can work independently.
- Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:

<head> <link rel="stylesheet" type="text/css" href="mystyle.css" /> </head>

- The browser will read the style definitions from the file '*mystyle.css*', and format the document according to it.
- An external style sheet can be written in any text editor. The file should not contain any html tags.

Example for external Style sheet

one.html

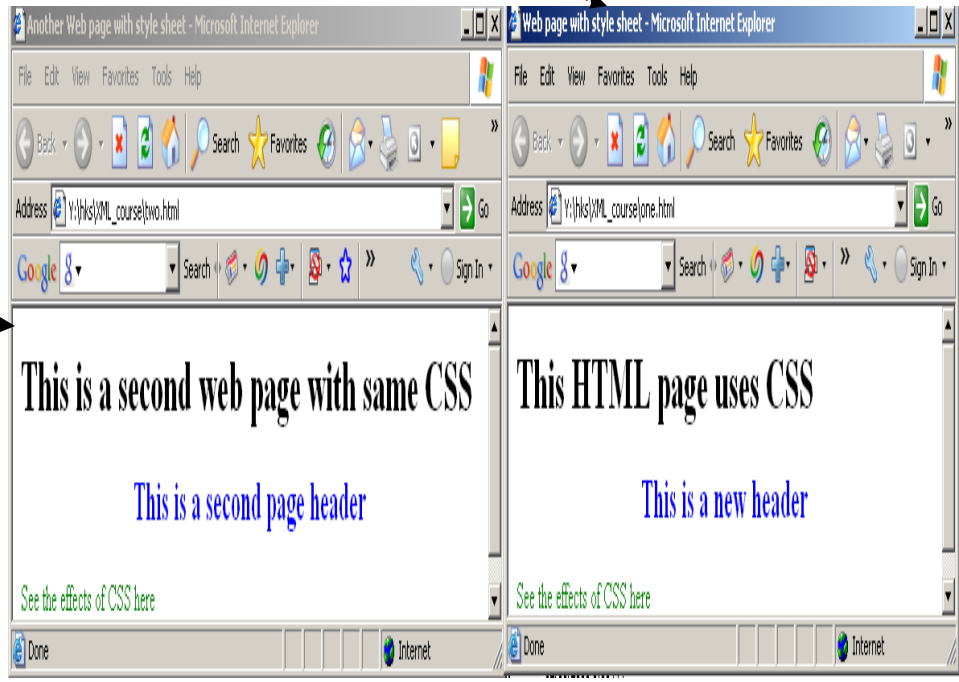
```
<html>
<head>
<title>Web page with style sheet</title>
</head>
<body>
<link href="one.css" rel="stylesheet" type="text/css" >
<h1>This HTML page uses CSS</h1>
<h2>This is  a new header </h2>
<p> See the effects of CSS here </p>
</body>
</html>
```

one.css

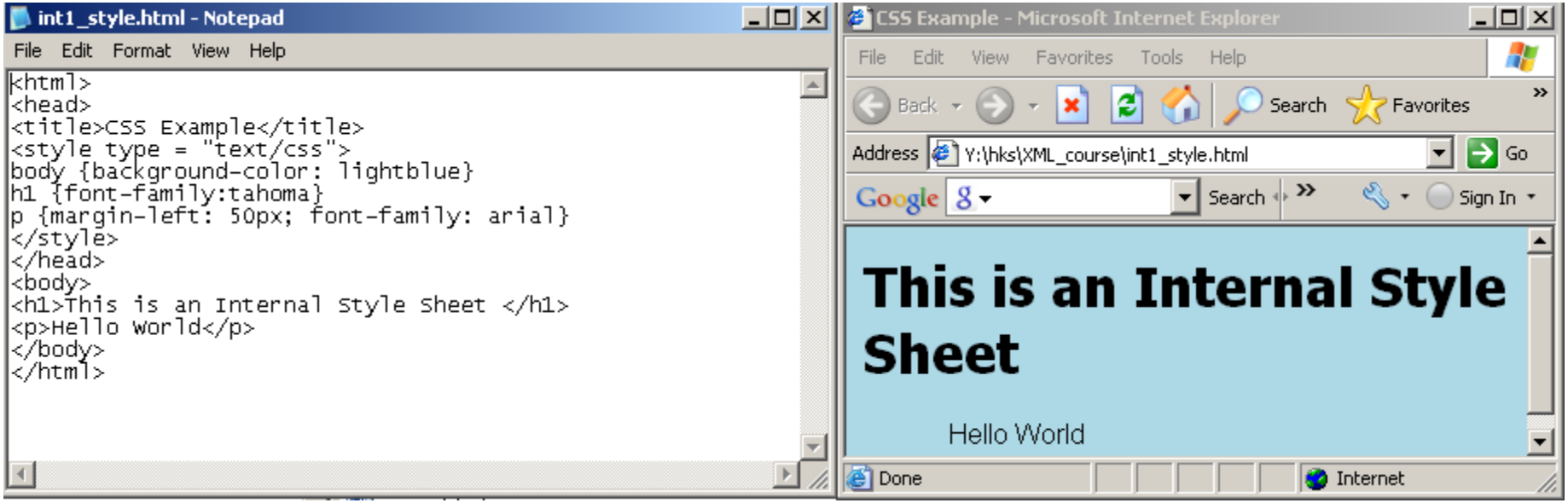
```
body { color:black}
h2 {text-align:center; color:blue;font-family:”verdana”}
p {font-family: “sans serif”; color:green}
```

two.html

```
<html>
<head>
<title>Another Web page with style sheet</title>
</head>
<body>
<link href="one.css" rel="stylesheet" type="text/css" >
<h1>This is a second web page with same CSS</h1>
<h2>This is  a second page header </h2>
<p> See the effects of CSS here </p>
</body>
</html>
```



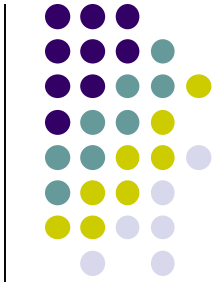
- An *internal or embedded style* sheet should be used when a single document has a unique style. You define internal styles in the head section by using the <style> tag
- The browser will read the style definitions, and format the document according to it.



- Using *inline style* it is possible to place CSS in the your HTML code.
- Inline CSS has the highest priority out of external, internal.
- To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a paragraph
- An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly, such as when a style is to be applied to a single occurrence of an element.

```
<p style="color: yellow;
margin-left: 20px"> This is a paragraph </p>
```

Class and ID Selectors



- You can also define your own selectors in the form of **class** and **ID** selectors.
- The benefit of this is that you can have the same HTML element, but present it differently depending on its class or ID
- In the CSS, a class selector is a name preceded by a **full stop** (“.”) and an ID selector is a name preceded by a **hash character** (“#”).

- Style the element with id="firstname"

```
<style>
#firstname {
  color: green;
}
</style>
```

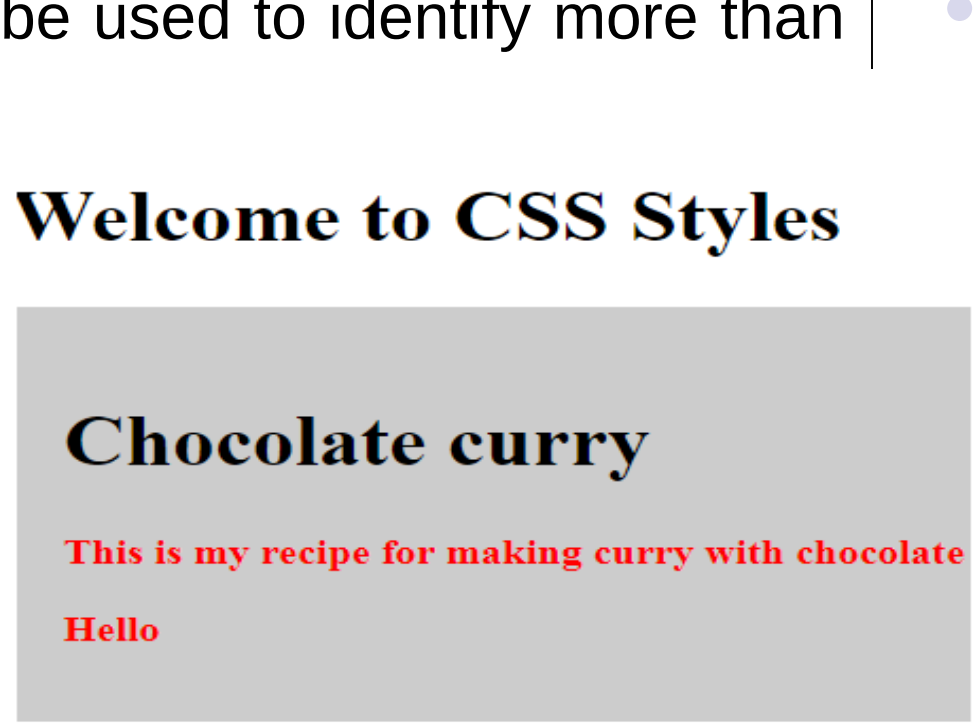
Note: An id name cannot start with a number!

Select and style all elements with class="intro"

```
<style>
.intro {
  background-color: yellow;
}
</style>
```

- The difference between an ID and a class is that an ID can be used to identify one element, whereas a class can be used to identify more than one.

```
<!DOCTYPE html>
<html>
<head>
<style>
#top {
    background-color: #ccc;
    padding: 20px
}
.intro {
    color: red;
    font-weight: bold;
}
</style>
</head>
<body>

<h1>Welcome to CSS Styles</h1>
<div id="top">
<h1>Chocolate curry</h1>
<p class="intro">This is my recipe for making curry with chocolate</p>
<p class="intro">Hello</p>
</div>
</body>
</html>
```



Generic Selectors

- A generic class can be defined if you want a style to apply to more than one kind of tag
- A generic class must be named, and the name must begin with a period.

```
.date {
  color: red;
}
```

- The .date class selector will target all HTML elements that have the class="date" attribute. So, the following HTML elements will **all** be styled:

HTML

```
<p class="date">
  Saturday Feb 21
</p>
<p>
  The event will be on <em class="date">Saturday</em>.
</p>
```

RESULT

Saturday Feb 21

The event will be on *Saturday*.

pseudo-elements and pseudo-classes

- CSS introduces the concepts of *pseudo-elements* and *pseudo-classes* to extend the addressing model and permit formatting based on information that lies outside the document tree.
- Pseudo-elements refer to sub-parts of an element's content (e.g., the first letter or first line of a paragraph, etc.).
- Pseudo-classes refer to elements that are grouped dynamically (e.g., all links that have been visited, all left-hand pages, etc.)
- Pseudo-classes are allowed anywhere in selectors while pseudo-elements may only appear as the last segment of a selector.

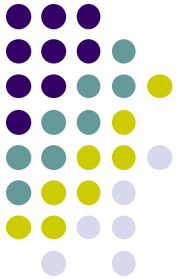
Pseudo Classes

- Pseudo classes are styles that apply when something happens, rather than because the target element simply exists
- Names begin with colons
- `hover` classes apply when the mouse cursor is over the element
- `focus` classes apply when an element has focus

The syntax of pseudo-classes:

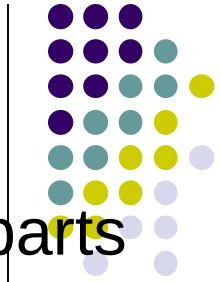
```
selector:pseudo-class {
  property:value;
}
```

Pseudo Class Example



```
<!-- pseudo.html -->
<html>
  <head> <title> Checkboxes </title>
    <style type = "text/css">
      input:hover {color: red;}
      input:focus {color: green;}
       a {   color: blue;  }
       a:hover { color: red;}
    </style>
  </head>
  <body>
    <form action = "">
      <p>
        Your name:
        <input type = "text" />
         <a href="#"> Click here </a>
      </p>
    </form>
  </body>
 </html>
```

pseudo-element



- A CSS pseudo-element is used to style specified parts of an element.
- **The :first-line pseudo-element**

The :first-line pseudo-element is used to apply special styles to the first formatted line. For instance:

P:first-line { font-style: small-caps }

- The above rule means "change the font style of the first line of every paragraph to small-caps". However, the selector "P:first-line" does not match any real HTML element. It does match a pseudo-element that conforming user agents will insert at the beginning of every paragraph.

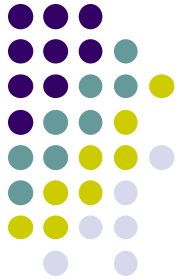
Pseudo element Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p:first-line {
  color: #ff0000;
  font-variant: small-caps;
}
</style>
</head>
<body>
<p>This is a somewhat long HTML paragraph that will
  be broken into several lines. The first line will be
  identified by a fictional tag sequence. The other lines will
  be treated as ordinary lines in the paragraph.</P>
</body>
</html>
```

The syntax of pseudo-elements:

```
selector::pseudo-element {
  property:value;
}
```

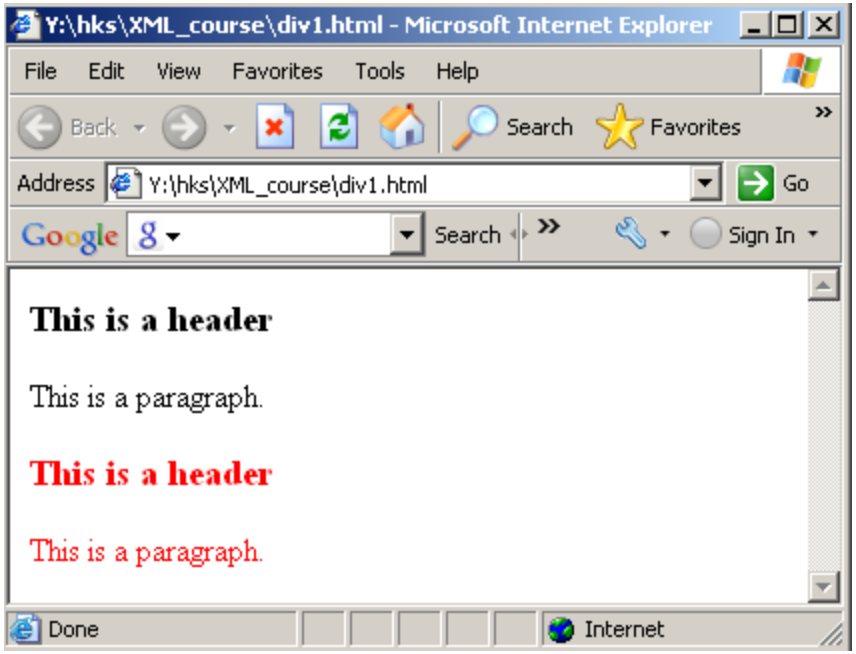
XHTML tag



- a tag defines sections of a Web page to make it easier to manage, style, and manipulate. You can use the tag when you want to *center a block of content or position a content block on the page*. The tag is a very powerful tool for Web developers.
- The div element is very often used with CSS to layout a web page.
- **Note:** Browsers usually place a line break before and after the div element.

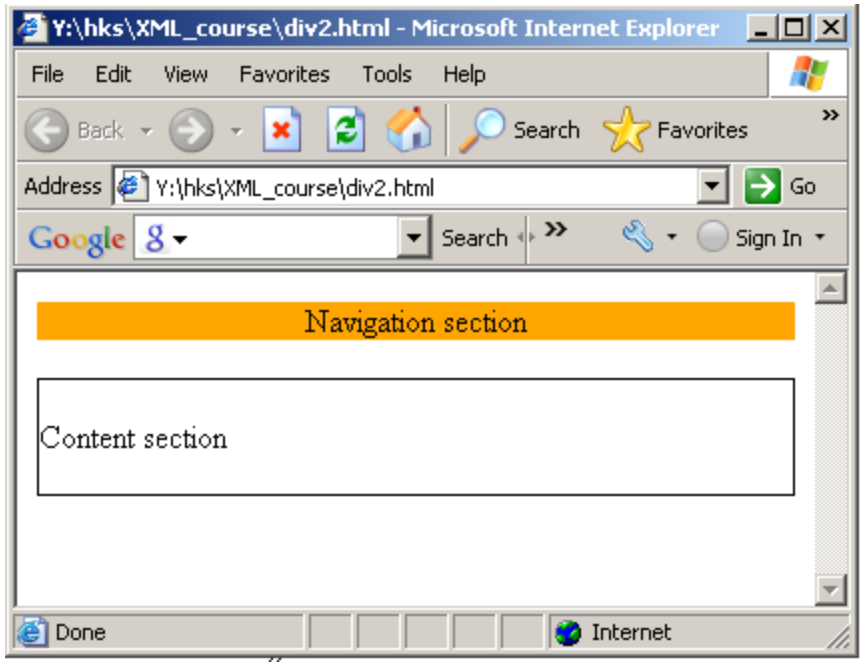
tag example-1

```
<html>
<body>

<h3>This is a header</h3>
<p>This is a paragraph.</p>

<div style="color:red">
  <h3>This is a header</h3>
  <p>This is a paragraph.</p>
</div>

</body>
</html>
```



tag example-2

```
<html>
<body>
<div style="background-
    color:orange;text-align:center">
    <p>Navigation section</p>
    </div>
<div style="border:1px solid
    black">
<p>Content section</p> </div>
</body>
</html>
```

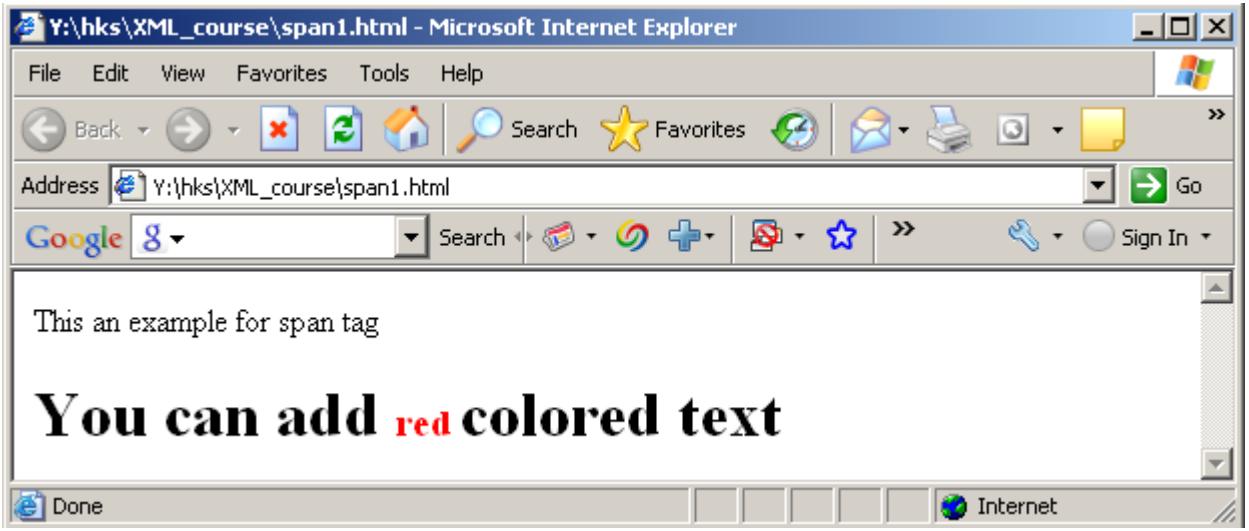


XHTML tag

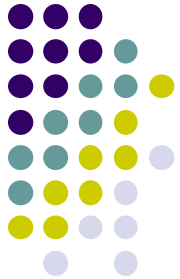
- The tag has very similar properties to the tag, in that it changes the style of the text it encloses.
- The difference is that span goes into a finer level, so we can span to format a single character if needed. There is no line feed after the tag.
- The tag won't change the enclosed items at without any style attributes.

tag example-1

```
<html>
<body>
<p>This an example for span tag</p>
<h1>You can add <span style="color: red; font-size: 14pt">red
    </span> colored text</h1>
</body>
</html>
```

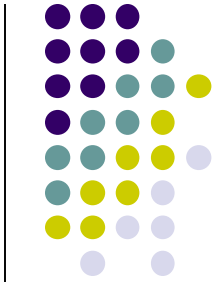


CSS Advantages



- Greater typography and page layout control.
- Style is separate from structure.
- Styles can be stored in a separate document and linked to from the web page.
- Potentially smaller documents.
- No need for <font> tags.
- Easier site maintenance.

CSS Disadvantages



- There is one large disadvantage -- Cascading Style Sheet technology is not yet uniformly supported by browsers.
- This disadvantage will be less of an issue in the future as the browsers comply with standards.
- The following properties can be used in CSS.

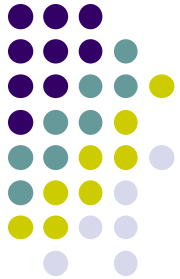
CSS Reference

Refer : for Values

http://www.w3schools.com/CSS/CSS_reference.asp

•Background •Border •Classification •Dimension •Font •Generated Content •List and Marker •Margin	•Outlines •Padding •Positioning •Table •Text

Hands on Exercise(25 mins)



1. Create document resume.htm file to hold your resume and create an external style sheet called “resume.css” that defines this formatting using CSS
   - *The background color*
   - *Text color*
   - Font with Italics and size
   - Define a style for H1 and/or H2 tags
2. Explore few pseudo elements and classes and use in the style sheet.