SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF INFORMATION TECHNOLOGY

WEB TECHNOLOGY

III YEAR - V SEM

UNIT 3 – SERVER-SIDE SCRIPTING

Topic 2 - Intrinsic Event Handling-Modifying Element Style

Intrinsic Event Handling

- An event is an occurrence of something potentially interesting to a script:
  - Ex: mouseover and mouseout events
- An HTML intrinsic event attribute is used to specify a script to be called when an event occurs
  - Ex: `onmouseover`
  - Name of attribute is `on` followed by event name

Intrinsic Event Handling

TABLE 5.1: HTML intrinsic event attributes.

Attribute	When Called
`onload`	Immediately after the body of document has been fully read and parsed by the browser (this attribute only pertains to `body` and `frameset`).
`onunload`	The browser is ready to load a new document in place of the current document (this attribute only pertains to `body` and `frameset`).
`onclick`	A mouse button has been clicked and released over the element.
`ondblclick`	The mouse has been double-clicked over the element.
`onmousedown`	The mouse has been clicked over the element.
`onmouseup`	The mouse has been released over the element.
`onmouseover`	The mouse has just moved over the element.
`onmousemove`	The mouse has moved from one location to another over the element.
`onmouseout`	The mouse has just moved away from the element.

Intrinsic Event Handling

`onfocus`	The element has just received the keyboard focus (this attribute only pertains to certain elements, including `a`, `label`, `input`, `select`, `textarea`, and `button`).
`onblur`	The element has just lost the keyboard focus (attribute pertains only to same elements as `onfocus`).
`onkeypress`	This element has the focus, and a key has been pressed and released.
`onkeydown`	This element has the focus, and a key has been pressed.
`onkeyup`	This element has the focus, and a key has been released.
`onsubmit`	This form element is ready to be submitted (only applies to `form` elements).
`onreset`	This form element is ready to be reset (only applies to `form` elements).
`onselect`	Text in this element has been selected (highlighted) in preparation for editing (applies only to `input` and `textarea` elements).
`onchange`	The value of this element has changed (applies only to `input`, `textarea`, and `select` elements).

Intrinsic Event Handling

```
<body onload="window.alert('Body loaded.');"
      onunload="window.alert('Unloading...');">
  <form action="http://www.example.org"
        onsubmit="window.alert('Submitting...');"
        onreset="window.alert('Resetting...');">
    <p>
      <input type="text" name="someText"
             onkeypress="window.alert('Text field got character.');"
             onselect="window.alert('Text selected.');" />
      <br />
      <input type="button" name="aButton" value="Click Me"
             onclick="window.alert('Button clicked.');" />
      <br />
      <input type="submit" name="aSubmit" value="Submit"
             onfocus="window.alert('Submit button got focus.');" />
      <input type="reset" name="aReset" value="Reset" />
    </p>
  </form>
</body>
```

Intrinsic Event Handling

- Intrinsic event attribute value is a script; what language is it written in?
- HTTP Content-Script-Type header field specifies default scripting language
- `meta` element allows document to specify values as if they were header fields

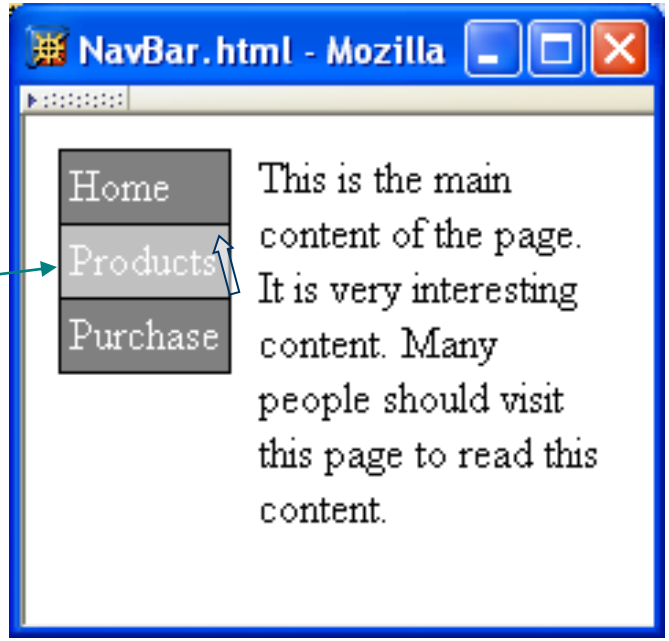
```
<meta http-equiv="Content-Script-Type" content="text/javascript" />
```

Header field name

Header field value

Modifying Element Style

Change background color of element containing cursor



Modifying Element Style

```
<td onmouseover="highlight(this);"
  onmouseout="lowlight(this);"><a
    href="http://www.example.org"
    >Products</a>
</td>
```

Modifying Element Style

Like rollover, style needs to be modified
both when entering and exiting the element.

```
<td onmouseover="highlight(this);"
  onmouseout="lowlight(this);"><a
    href="http://www.example.org"
    >Products</a>
</td>
```

Modifying Element Style

Reference to `Element` instance representing the `td` element

```
<td onmouseover="highlight(this);"
  onmouseout="lowlight(this);"><a
    href="http://www.example.org"
    >Products</a>
</td>
```

Modifying Element Style

```
function highlight(element) {
  element.style.backgroundColor = "silver";
  return;
}
```

Modifying Element Style

Reference to `Element` instance

```
function highlight(element) {
  element.style.backgroundColor = "silver";
  return;
}
```

Modifying Element Style

```
function highlight(element) {
  element.style.backgroundColor = "silver";
  return;
}
```

All `Element` instances have a `style` property with an Object value

Modifying Element Style

```
function highlight(element) {
  element.style.backgroundColor = "silver";
  return;
}
```

Properties of `style` object correspond to CSS style properties of the corresponding HTML element.

Modifying Element Style

- Rules for forming `style` property names from names of CSS style properties:
  - If the CSS property name contains no hyphens, then the `style` object's property name is the same
    - Ex: `color` → `color`
  - Otherwise, all hyphens are removed and the letters that immediately followed hyphens are capitalized
    - Ex: `background-color` → `backgroundColor`

Modifying Element Style

```
function highlight(element) {
  element.style.backgroundColor = "silver";
  return;
}
```

Net effect: "`silver`" becomes the specified value for CSS `background-color` property of `td` element; browser immediately modifies the window.

Modifying Element Style

- Alternative syntax (not supported in IE6/7/8):

```
function lowlight(element) {
  element.style.setProperty("background-color", "gray", "");
  return;
}
```

Modifying Element Style

- Alternate syntax (not supported in IE6/7/8):

```
function lowlight(element) {
  element.style.setProperty("background-color", "gray", "");
  return;
}
```

Every DOM2-compliant `style` object has a `setProperty()` method

Modifying Element Style

- Alternate syntax (not supported in IE6/7/8):

```
function lowlight(element) {
  element.style.setProperty("background-color", "gray", "");
  return;
}
```

CSS property value

CSS property name (unmodified)

Empty string or "important"

Modifying Element Style

- Advantages of `setProperty()` syntax:
  - Makes it clear that a CSS property is being set rather than merely a property of the `style` object
  - Allows CSS property names to be used as-is rather than requiring modification (which can potentially cause confusion)
- BUT lack of IE support makes it difficult to use (works with FF & Chrome)

Modifying Element Style

- Obtaining *specified* CSS property value:

```
if (element.style.backgroundColor == "gray") {
```

- Alternate DOM2 syntax (not supported by IE6/7/8):

```
if (element.style.getPropertyValue("background-color") == "gray") {
```