SOEN 287: WEB PROGRAMMING

Chapter 6
Dynamic Documents with JavaScript

Dynamic HTML document

- Up to now, our standard HTML document are static.
- Once displayed on the browser, a document does not change until the user initiates some activity like selecting a hyperlink with the mouse, or clicking on a button
- A ***dynamic HTML document*** is one whose tag attributes, tag contents, or element style properties can be changed after the document has been and is still being displayed by a browser.

Possible ‘dynamic’ changes

- When can they happen?
  - When explicitly requested by user
  - At regular time intervals
  - When a browser event occurs

- A few possible changes:
  - Element can be moved to a new location
  - Element can disappear and/or reappear
  - Colors of background and foreground (elements) can be changed
  - Text style, font, size and style can be changed
  - Content of an element can be changed
  - Elements can be stacked in a specific order
  - …

Positioning Elements

- CSS-P was released by W3C in 1997, completely supported by IE9, FX3, and Chrome

- The position of any element is dictated by the three style properties: `position`, `left`, and `top`

- The three possible values of position are `absolute`, `relative`, and `static`

Absolute Positioning

- When want an element to be placed in a specific position
- Not related to sequence of elements or position of other elements

- Example:
  Used to superimpose a text over a paragraph (like a watermark)

```
<p style = "position: absolute; left: 50px;
     top: 100px;">
```

→ `absPos.html`

Absolute Positioning

- If an element is *nested* inside another element and is absolutely positioned, the top and left properties are *relative to the enclosing element*

→ `absPos2.html`

Relative Positioning

- Setting the `top` and `left` property results in element being displaced the specified amount from where it would be placed

→ `relPos.html`

- If no `top` and `left` properties are specified, the element is placed exactly where it would have been placed if no `position` property were given
- **<u>But</u>** it can be moved later

Static Positioning

- Static positioning is the default value if `position` is <u>not</u> specified
- Element is placed as if no `top` and `left` properties are specified, the element is placed exactly where it would have been placed if no `position` property were given.
- Difference is that element <u>can't</u> be moved later.

Moving Elements

- If `position` is set to either `absolute` or `relative`, the element can be moved after it is displayed

- Just change the `top` and `left` property values with a script

→ `mover.html` & `mover.js` (example with absolute position)

Element Visibility

- The `visibility` property of an element controls whether it is displayed
- The values are `visible` and `hidden`

```
if (dom.visibility == "visible"
    dom.visibility = "hidden";
 else
    dom.visibility = "visible";
```

→ `showHide.html` & `showHide.js`

Changing Colors and Fonts

- Background color is controlled by the `backgroundColor` property
- Foreground color is controlled by the `color` property

Background color:

```
<input type = "text" size = "10"
       name = "background"
       onchange = "setColor('background',
                           this.value)">
```

→ `dynColors.html` & `dynColors.js`

Dynamic Colors and Fonts

- We can change the font properties of any element that contains text by using the `mouseover` and `mouseout` events to trigger a script that makes the changes

```
onmouseover = "this.style.color = 'blue';
       this.style.font = 'italic 16pt Times';"
onmouseout = "this.style.color = 'black';
       this.style.font = 'normal 16pt Times';”
```

Dynamic Colors and Fonts

- *JavaScript property names*:
  - For CSS attributes w/o hyphens – same
  - For CSS attributes w/hyphens – delete hyphen and capitalize the next letter
- For example:
  - `font-size` → `fontSize`
  - `background-color` → `backgroundColor`

→ `dynFont.html`

Dynamic Content

- The content of an HTML element is addressed with the `value` property of its associated JavaScript object

- Example:

  a help box for a form

→ `dynValue.html & dynValue.js`

Stacking Elements

- Have seen to date that top and left allows us to position an element
- 3rd dimension allows us to stack elements
- The `z-index` attribute determines which element is in front and which are covered by the front element
- The JavaScript property associated with the `z-index` attribute is `zIndex`
- `z-index` can be changed dynamically (by changing `zIndex`)

Stacking Elements

- To change stacking order, the handler function must change the `zIndex` property value of the element

- A call to the function from an element sets the `zIndex` value of the new top element to 1 and the `zIndex` value of the old top element to 0

- It also sets the current top to the new top

→`stacking.html` & `stacking.js`

You try

Write markup document and JS files for the following (based on exercises at the end of Chapter 6)

- **Exercise 1**:
  - Document must have at least 10 lines of text
  - Paragraph top: 100px and left: 100px
  - The width of the paragraph should be set to 330px
  - A light colored image must be superimposed over the center of the text as a nested element

You try

- **Exercise 2**:

  Modify document for exercise 1 to add four buttons labeled, Northwest, Northeast, Southwest & Southeast. When they are pressed the image must move to the specified corner. The image should start off in the northwest (upper left corner) of the text.

- **Exercise 3**:

  Modify document for exercise 2, to make the buttons toggle their respective copies of your image on and off so that at any time the document may include none, one, two three or four copies of the image. The initial document should have no images showing.