6.1 Introduction
- Dynamic HTML is not a new markup language
- 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
- We will discuss only W3C standard approaches
- All examples in this chapter, except the last, use the DOM 0 event model and work with both IE6 and NS6
- To make changes in a document, a script must be able to address the elements of the document using the DOM addresses of those elements

6.2 Element Positioning
- HTML tables can be used for element positioning, but they lack flexibility and are slow to render
- CSS-P was released by W3C in 1997

6.2 Element Positioning (continued)
- CSS-P allows us to place any element anywhere on the display, and move it later
- The position of any element can be dictated by the three style properties: position, left, and top
- The three possible values of position are absolute, relative, and static
- Absolute Positioning
  <p style = "position: absolute; left: 50px; top: 100 px;">
  SHOW absPos.html and Figure 6.1
  - If an element is nested inside another element and is absolutely positioned, the top and left properties are relative to the enclosing element
  SHOW absPos2.html and Figure 6.2

- Relative Positioning
  - 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
  - But it can be moved later
  - If top and left properties are given, they are offsets from where it would have placed without the position property being specified
  - If negative values are given for top and left, the displacement is upward and to the left
  - Can make superscripts and subscripts
  --> SHOW relPos.html & Figure 6.3

- Static Positioning
  - The default value if position is not specified
  - Neither top nor left can be initially set, nor can they be changed later

6.3 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
  --> SHOW mover.html & Figures 6.4 and 6.5

6.4 Element Visibility
- The visibility property of an element controls whether it is displayed
- The values are visible and hidden
- Suppose we want to toggle between hidden and visible, and the element’s DOM address is dom
  if (dom.visibility == "visible")
  dom.visibility = "hidden";
  else
  dom.visibility = "visible";
  --> SHOW showHide.html

6.5 Changing Colors and Fonts
- Background color is controlled by the backgroundColor property
- Foreground color is controlled by the color property
- Can use a function to change these two properties
- Let the user input colors through text buttons
- Have the text elements call the function with the element address (its name) and the new color
  Background color:
  <input type = "text" size = "10" name = "background" onchange = "setColor('background', this.value)">
  The actual parameter this.value works because at the time of the call, this is a reference to the text box (the element in which the call is made)
  - So, this.value is the name of the new color
  --> SHOW dynColors.html

6.5 Dynamic Colors and Fonts (continued)
- Changing fonts
  - We can change the font properties of a link by using the mouseover and mouseout events to trigger a script that makes the changes
  - In this case, we can assign the complete script to make the changes to the element’s attribute (in the HTML)
    onmouseover = "this.style.color = 'blue'; this.style.font = 'italic 16pt Times';"
    onmouseout = "this.style.color = 'black'; this.style.font = 'normal 16pt Times';"
  --> SHOW dynLink.html

6.6 Dynamic Content
- The content of an HTML element is addressed with the value property of its associated JavaScript object
  --> SHOW dynValue.html
6.7 Stacking Elements (continued)
- The `top` and `left` properties determine the position of an element on the display screen, which is a two-dimensional device.
- We can create the appearance of a third dimension by having overlapping elements, one of which covers the others (like windows).
- This is done with the `z-index` property, which determines which element is in front and which are covered by the front element.
- The JavaScript variable associated with the `z-index` property is `zIndex`.
- The stacking order can be changed dynamically.
- Make the elements anchors, so they respond to mouse clicking.
- The `href` attribute can be set to call a JavaScript function by assigning it the call, with `javascript:` attached to the call code.

6.8 Locating the Mouse Cursor (continued)
- If we want to locate the mouse cursor when the `mousedown` event occurs, we can use the `mousemove` event.
- We know how to move an element - just change its `top` and `left` properties.
- The DOM 2 event model is required (the `Event` object and its property, `currentTarget`).
- Example: magnetic poetry
  - The DOM 2 event model is required (the `Event` object and its property, `currentTarget`).
  - We can use `mousedown`, `mouseup`, `mousemove` events to grab, drag, and drop.
  - We know how to move an element - just change its `top` and `left` properties.

6.9 Reacting to a Mouse Click
- A mouse click can be used to trigger an action, no matter where the mouse cursor is in the display.
- Use event handlers for `mousedown` and `mouseup` that change the visibility attribute of the message.

6.10 Slow Movement of Elements (continued)
- Example: move a text element from its initial position (100, 100) to a new position (300, 300).
- If we want to move an element in a display that has more than one element, we must first determine which element the mouse cursor is over.
- We can get the `id` of an element on which an event occurs with the `scrElement` property of an event object; `scrElement` has a property named `id`.
  ```javascript
  event.srcElement.id
  ```

6.11 Dragging and Dropping an Element
- Drag and drop requires three processes:
  1. Get dom of the element to be moved when the mouse button is pressed down (`mousedown`)
  2. If we want to move an element in a display that has more than one element, we must first determine which element the mouse cursor is over.
  3. We can get the `id` of an element on which an event occurs with the `scrElement` property of an event object; `scrElement` has a property named `id`.

```
event.srcElement.id
```
6.11 Dragging and Dropping an Element (continued)

2. Move the element by changing its top and left properties of the element as the mouse cursor is moved (onmousemove)
   - Use event.x and event.y to track the mouse cursor

3. Dropping the element when the mouse button is released by undefining the dom used to carry out the move

--> SHOW dragDrop.html