JavaScript Events

CSE/ISE 102
(Intro to Web Design)

Stony Brook University

Event Flow

- Two models (IE vs. Netscape):
  - Event bubbling (IE)
  - Event capturing (Netscape)

Event Bubbling

- Event starts at most specific element and flows upward to least specific element
- e.g., `div` → `body` → `html` → `document`
- Supported by all modern browsers
  - IE9, Firefox, Chrome, and Safari continue up through `window` object

Event Capturing

- Least specific node gets the event first
  - event gradually reaches most specific node
  - e.g., document → html → body → div
- used only for special circumstances now

DOM Event Flow

- Three phases:

1. event capturing phase
2. at the target
3. event bubbling phase

Event Handlers

Event Handler Types

- HTML event handlers
- DOM level 0 event handlers
- DOM level 2 event handlers

HTML Event Handlers

- Assign events using an HTML attribute with the name of the event handler
  - value is JavaScript code to execute
    - either actual JS or a script defined elsewhere on the page

Examples

<input type="button" value="Click me" onclick="alert('Clicked!')" />

<input type="button" value="Click Me" onclick="showMessage()" />

Behind the Scenes

- A function is created to wrap the attribute value
  - contains a special local variable event
- Within the function, members of document and the element can be accessed as though they were local
  - form also includes parent form element

```
<form method="post" ...>

<input type="text" name="username"
value="">

<input type="button" value="Echo Username"
onclick="alert(username.value)">

</form>
```

Downsides

- Timing issues
  - HTML element may appear and be interacted with before the event handler code is ready
- Tightly couples JavaScript to the HTML

DOM 0 Event Handlers

- Each element has event handler properties like `onclick`
  - Start by retrieving reference to object
  - Assign a function to the property
  - event handler is considered to be a method of the element

Example

```
var btn = document.getElementByID("myBtn");
btn.onclick = function() { alert("Clicked"); };
```

DOM 2 Event Handlers

- Defines two methods: addEventListener() and removeEventListener()
- Each method takes 3 arguments:

1. event name to handle
2. event handler function
3. Boolean: whether to call event handler during capture phase (true) or bubble

Example

```
var btn = document.getElementByID("myBtn");
btn.addEventListener("click",
                     function() { alert(this.id); }, false);
// will be fired in the bubbling phase
```

Things to Consider

- Multiple event handlers can be added to the same element and event type
  - event handlers fire in order of addition
- Removal: use same arguments as original addition
  - **NOTE:** does ***NOT*** work for anonymous functions!

The `event` Object

The `event` Object

- Created when a DOM-related event is fired
- contains relevant information: source element, type of event, etc.
- In DOM-compliant browsers, passed in as sole argument to event handler

```
var btn =
document.getElementByID("myBtn");

btn.onclick = function(event)
{ alert(event.type); }; // "click"

btn.addEventListener("click",
function(event){ alert(event.type); },
false); // "click"
```

Common Event Properties

Property	Type	Read/Write	Description
bubbles	Boolean	Read only	does event bubble?
currentTarget	Element	Read only	whose event handler is currently handling event
eventPhase	Integer	Read only	1 = capture, 2 = at target, 3 = bubbling
target	Element	Read only	
type	String	Read only	type of event
view	AbstractView	Read only	`window` in which event occurred

Event Types

- Selected DOM Level 3 categories:
  - User Interface (UI) events
  - Focus events
  - Mouse events
  - Mutation events

User Interface Events

- load — activates when page is completely loaded
- error — fires on JS error, image loading problem, etc.
- select — fires on text selection in text box
- resize — fires on window resize
- Put these in the <body> tag

Focus Events

- blur — fires when an element loses focus
- focus — fires when an object gains focus

- These events do not bubble

Mouse Events

- click
- dblclick
- mouseover
- mouseout

- These events all bubble

Mutation Events

- Fire when part of the DOM has changed
- DOMSubtreeModified
- DOMNodeInserted
- DOMNodeRemoved

Next Time

- Scripting Forms