DOM (Document Object Model)

Nan Niu (nn@cs.toronto.edu)
CSC309 -- Summer 2008

What's DOM?

- W3C Standard
  - Implemented differently by every browser
- Interface between document displayed by browser and application programs
  - Object-oriented
- Platform-neutral and language-neutral collection of interfaces
  - Can program with a variety of languages (C, Java, VB)
- Documents have treelike structures
  - One interface for every XML node type
- Create documents, move around document structure (parse), and change, add, or delete elements.

DOM Representation of XML

```xml
<A>
  <B>text1</B>
  <C>
    <D>child of C</D>
    <E>another child of C</E>
  </C>
  <F>moreText</F>
</A>
```

DOM Structure for an XHTML

Language Binding

- A support language must have a binding to the DOM constructs
- Correspondence between constructs in the language and elements in the DOM
- JavaScript binding
  - Elements of a document are objects
  - Attributes are represented by properties

DOM Parts

- Core
  - Minimal set of objects and interfaces for accessing and manipulating document objects (mainly XML oriented)
- DHTML
  - Extends the core API to describe objects and methods specific to XHTML documents
  - Convenience methods and properties that more appropriate to script writers
  - These enhancements are not applicable to general XML documents
DHTML Object Model

Key Interfaces

- Document
- Element
- Event

Document

- The central interface is Document
- Create new elements, attributes and text nodes
- Access existing elements
  - `getElementsByTagName(stringName)`
  - `getElementById(stringId)`

Element

- Navigate document
- Change document tree structure

Example: DOM Tree

Event Driven Programming

Most, if not all, GUI systems and toolkits are designed to be event driven, meaning that the main flow of your program is not sequential from beginning to end. If you’ve never done GUI programming, this is one of the trickiest paradigm shifts.

— Robin Dunn, speaking on GUI programming at OSCON2004

Hollywood Principle: "Don’t call us; we’ll call you." ... You implement the interfaces, you get registered. You get called when the time is right. This requires a distinctly different way of thinking to that which is taught in introductory programming where the student dictates the flow of control.

Structured vs. Event Driven

Structured Program

Event Driven Architecture

Event Driven Execution
- JavaScript programs are typically event-driven
- Execution is triggered by various events that occur on the Web page, usually as a result of something the user does
  - `onClick`, `onDoubleClick`, `onKeyDown`, `onLoad`, `onMouseOver`, `onSubmit`, `onResize`, ...
- Events are associated with XHTML tag attributes
  - e.g., the `onclick` event can be associated with `<a>` and form `<input>` tags

Partial List of Events
- Clipboard
  - `oncopy`, `oncut`, `onpaste`
- Keyboard
  - `onkeydown`, `onkeyup`, `onkeypress`
- Mouse
  - `onmousedown`, `onmouseup`, `onmousemove`
- Other
  - `onfocus`, `onblur`, ...

Registering Event Handlers
- By assigning the event handler script to an event tag attribute
  - `<a id="myLink" href="..." onmouseover="popup();">...</a>`
- By assigning the event handler script to an event property of an object
  - `Document.getElementById("myLink").onmouseover = popup;`

 onload & timers
- `onload`
  - Fires when element (or all children) finish loading
  - Used in the `<body>` to execute script after page has been rendered
- Example: Count how many seconds have passed since page finish rendering

Announcements
- A1 is up
- Prerequisite waivers are due tonight
- Next week (May 28)
  - Office hour: 5-6pm in BA3289, A1-related questions ONLY
  - Tutorial starts at 6pm in WI1017; no lecture after the tutorial
- From week 4 (June 4) to the end of the term, we will meet in BA1200