topic:
• very quick and brief introduction to javascript

references:
• JavaScript: The Definitive Guide, 6th edition,
  by David Flanagan, O’Reilly Media, Inc., (c) 2011.
• on-line tutorial:
  http://www.w3schools.com/js/default.asp
• on-line reference guide:
  http://www.w3schools.com/jsref/default.asp

introduction to javascript

• JavaScript is a completely different language from Java
  — the syntax is like Java
  — but the functions are similar to Scheme
  — and the prototype-based inheritance is similar to Self
• works in tandem with HTML and CSS to implement web pages:
  — HTML → content
  — CSS → presentation
  — JavaScript → behavior

variables

• variables are declared with the var keyword
  var x; // declare a variable named x
• values are numbers, text strings, and boolean values
• some examples:
  var a = 1;
  var b = 0.23;
  var c = "hello";
  var d = 'world';
  var e = true;
  var f = false;
  var g = null;  // null means "no value"
  var h = undefined;  // undefined is like null
• JavaScript is loosely typed, so defining a variable does not mean specifying its data type;
  the data type is inferred from the value that is assigned to the variable; this can change
  during run-time

objects

• an object is another JavaScript data type
• an object literal defines an object and its contents
  var book = {
    genre: "mystery",
    pages: 180,
    name: "The Thin Man"
  } // end of object literal
  book.author = "Dashiell Hammett"; // add field by assigning a value
  book.getName = function() { return this.name; } // define a method
• empty objects can be created in two ways:
  var book1 = {};
  var book2 = new Object();
  // then put stuff in the empty objects:
  book1.genre = "mystery";
  book2.name = "The Thin Man";
arrays

- JavaScript also supports arrays
- Arrays can be defined and initialized like this:
  ```javascript
  var myarray = [1, 2, 3, 4, 5]; // create new array
  myarray[6] = 13; // add new entries to existing array
  ```
- Empty arrays can be created in two ways:
  ```javascript
  var myarray1 = new Array();
  var myarray2 = [];
  ```
- Arrays can contain any type of data, including objects

operators

- Arithmetic, logical and relational operators are just like Java:
  ```javascript
  +, -, *, /, %
  ||, &&, !
  >, >=, <, <=, ==, !=
  ```
- Javascript also has "strict" operators:
  - `===` means "strict equality"
  - `!==` means "strict inequality"
  - i.e., objects are identical (as opposed to the values they store being equal)

functions

- Functions can be defined inline, with parameters and with return values (both are optional)
- Functions defined within objects are called methods
- Functions can be defined like this:
  ```javascript
  function plusplus( x ) {
    return( x + 1 );
  }
  ```
- Or like this:
  ```javascript
  var plusplus = function( x ) { 
    return( x + 1 );
  }
  ```
- And invoked like this:
  ```javascript
  var y = plusplus( 3 ); // y is set to 4
  ```

client-side JavaScript

- Client-side JavaScript is embedded in HTML and is invoked in browser windows
- As opposed to server-side JavaScript, which implies any JavaScript run outside of a web browser
- Other JavaScript interpreters (other than web browsers) include:
  - **Rhino**
    - Free software from Mozilla:
      ```javascript
      http://www.mozilla.org/rhino
      ```
    - A JavaScript interpreter written in Java
    - Provides access to the full Java API
  - **Node**
    - Free software, under active development:
      ```javascript
      http://nodejs.org
      ```
    - A JavaScript interpreter written in C++, that sits on top of Google's V8 JavaScript engine
events

- event handlers are HTML attributes that begin with "on", e.g., "onclick"

- categories:
  - device-dependent input events
  - user interface events
  - state-change events
  - API-specific events
  - time and error handlers

- Legacy events
  - Form events
  - Window events: onload(), onerror()
  - Mouse events: clientX, clientY, button, which, altKey, ctrlKey, metaKey, shiftKey
  - Key events: keyCode, altKey, ctrlKey, metaKey, shiftKey

window object

- Timers: register a function to be invoked once or repeated after time has elapsed
  - setTimeout(): invoke once
  - setInterval(): invoke repeatedly
  - clearInterval(): cancels setInterval()

- Location object
  - window.location == document.location
  - refers to the URL of the web page
  - includes fields: protocol, host, hostname, port, pathname, search, hash
    * search is set to the URL elements after a '?' character in the URL
    * hash is set to the URL elements after a '#' character in the URL
  - methods:
    * loc.replace( URL ) or location = URL loads page

- History object
  - length is the number of entries in the history (browser page)
  - go( N ) is a function that returns the page to the previous N-th page in the browser history
  - for example, go( 2 ) is like hitting the "back" button twice in the browser

- Navigator object
  - contains browser and version number information
  - fields include appName, appVersion, userAgent (USER_AGENT content from HTTP header), platform (operating system)
  - methods:
    * online() returns true if connected or false otherwise
    * geolocation()
    * javaEnabled()
    * cookiesEnabled()

- Screen object
  - window.screen
  - fields include width, height, colorDepth
• Dialog boxes
  – alert()
  – confirm()
  – prompt()

• Error handling function
  – window.onerror = function( msg, url, line ) { ... }