JavaScript: Objects, BOM, and DOM

CS 4640
Programming Languages for Web Applications
Objects group variables and functions to create a model representing something you would recognize from the real world.

### Object type: Hotel

<table>
<thead>
<tr>
<th>Event</th>
<th>Happens when</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserve</td>
<td>reservation is made</td>
<td>Name: Awesome</td>
</tr>
<tr>
<td>Cancel</td>
<td>reservation is cancelled</td>
<td>Rating: 5</td>
</tr>
</tbody>
</table>

**Methods**

- `makeReservation()`: increases value of `bookings` property
- `cancelReservation()`: decreases value of `bookings` property
- `checkAvailability()`: subtracts value of `bookings` property from value of `rooms` property and returns number of rooms available

**Properties**

- Name: Awesome
- Rating: 5
- Rooms: 70
- Bookings: 56
- Pool: true
- Gym: true

### Object type: Car

<table>
<thead>
<tr>
<th>Event</th>
<th>Happens when</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerate</td>
<td>driver speeds up</td>
<td>Make: UVAI</td>
</tr>
<tr>
<td>Change</td>
<td>driver speeds up</td>
<td>currentSpeed: 30</td>
</tr>
<tr>
<td>Speed</td>
<td></td>
<td>Color: yellow</td>
</tr>
<tr>
<td>Fuel</td>
<td></td>
<td>Fuel: gasoline</td>
</tr>
</tbody>
</table>

**Methods**

- `changeSpeed()`: increases or decreases value of `currentSpeed` property

**Properties**

- Make: UVAI
- currentSpeed: 30
- Color: yellow
- Fuel: gasoline

Events are things or interactions that can happen to the objects. Methods represent tasks that are associated with the objects (or things we can do with the objects).

Properties tell us the characteristics of the objects.
JavaScript Objects

- JavaScript is an object-based language
  - It supports for object-oriented programming but not at the same level as other languages (ES6: introduced class – still lacks private property)

- Objects are represented as property-value pair
  - The property values can be data or functions (methods)

- A property is something that can be modified:
  - Data properties: primitive values or references to objects
  - Method properties: can be executed

- Objects can be created and their properties can be changed dynamically
  - JS is not really typed .. If it doesn’t care between a number and a string, why care between two kinds of objects?
Creating Objects

Create an object and assign variables and functions directly by using {} syntax

```javascript
var hotel = {
    name: "Awesome",
    rating: 5,
    rooms: 70,
    bookings: 56,
    pool: true,

    checkAvailability: function () {
        return this.rooms - this.bookings;
    }
};
```
Creating Objects with Constructors

function Hotel(name, rating, rooms, bookings, pool, gym) {
    this.name = name;
    this.rating = rating;
    this.rooms = rooms;
    this.bookings = bookings;
    this.pool = pool;
    this.gym = gym;
    this.checkAvailability = function() {
        return this.rooms - this.bookings;
    };
}

object

- name: "Awesome"
  rating: 5
  rooms: 70
  bookings: 56
  pool: true
  gym: true
  checkAvailability: function () {
    return this.rooms - this.bookings;
  }

object

- name: "Duh"
  rating: 3
  rooms: 45
  bookings: 27
  pool: false
  gym: false
  checkAvailability: function () {
    return this.rooms - this.bookings;
  }

Creating the constructor function and
revisit
Accessing Objects

- Access properties or methods of an object using dot notation

```javascript
var hotelName = hotel.name;
var roomsAvail = hotel.checkAvailability();
```

- Access properties or methods using square brackets

```javascript
var hotelName = hotel['name'];
var roomsAvail = hotel['checkAvailability']();
```
Updating Properties

• Update properties using dot notation

```javascript
hotel.name = 'VeryAwesome';
```

• Update properties using square brackets

```javascript
hotel['name'] = 'VeryAwesome';
```
Adding Properties

- Add a property using a dot notation

```javascript
hotel.shuttle = true;
```
Deleting Properties

• Delete a property using the `delete` keyword

```javascript
delete hotel.rating;
```
Web Browsers and Objects

Object type: Window
Properties
Location: http://www.cs.virginia.edu/~up3f/cs4640/syllabus.html

Object type: Document
Properties
URL: http://www.cs.virginia.edu/~up3f/cs4640/syllabus.html
lastModified: 01/14/2019 10:19:23
Title: CS4640 - WebPL
BOM: Browser Object Model

- **BOM** – collection of objects that the browser makes available to us for use with JavaScript

![Diagram of BOM objects]

- `window Object`
- `location Object`
- `history Object`
- `document Object`
- `navigator Object`
- `screen Object`
- `form Object`
- `link Object`
- `image Object`

DOM

(revisit)
DOM: Document Object Model

Object type: Document

Properties
- URL: http://www.cs.virginia.edu/~up3f/cs4640/syllabus.html
- lastModified: 01/14/2019 10:19:23
- Title: CS4640 - Web PL

Event
- Load: Happens when page and content have finished loading
- Click: user clicks the mouse over the page
- Keypress: user presses down on a key

Method
- write(): adds content to the document
- getElementByld(): accesses an element of a given id attribute
How A Browser See A Web Page

The browser receives an HTML page

It creates a model of the page and stores it in memory

It shows the page on screen using a rendering engine

Contact information
Instructor: Upsorn Praphamontripong
Office: Rice Hall 206
Office hours: TBD (until they are settled, whenever my door is open!)

TAs: TBD

Class hours
Tuesday, Thursday 2:00PM - 3:15PM, Clark Hall 108

Overview
The way web software is built has been rapidly changing. As use of the world wide web has shifted from information presentation to information gathering to direct customer sales (e-commerce) to enterprise applications, the amount and complexity of software has steadily been increasing. Many new technologies and frameworks have emerged everyday. Have you ever wondered what technologies, frameworks, or architectural styles you should use? What programming languages you should be familiar with? After you graduate, what languages will be popular or will be in great demand? How will you keep up with web development technologies? According to the UVA CS Industrial Advisory Board meeting, companies prioritize the foundation of web development over any specific technologies or frameworks.

This course will help you see how fundamental concepts of web development can be applied to develop reliable and usable web software regardless of the technologies or languages or frameworks. Although we put our emphasis on the concepts, you will develop dynamic web software with several commonly used programming languages and technologies.

You will work on user interface design, the front end development, back end development, and web-based information retrieval and processing. Over the semester, you will work on each programming assignment with a partner. Each assignment will be a step in creating a web application. You will be able to choose what to build, with the assignment constraining the features that must be used. By the end of the semester, you will have built a dynamic web application. With a large portion of the process that involves teamwork, interpersonal skills and conflict management.
# Using BOM Objects (Some Properties)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>window.screenX</td>
<td>X-coordinate of pointer, relative to top left corner of screen (in pixels)</td>
</tr>
<tr>
<td>window.screenY</td>
<td>Y-coordinate of pointer, relative to top left corner of screen (in pixels)</td>
</tr>
<tr>
<td>window.location</td>
<td>Current URL of window object</td>
</tr>
<tr>
<td>window.document</td>
<td>Reference to document object</td>
</tr>
<tr>
<td>window.history</td>
<td>Reference to history object for browser window or tab, which contains details of the pages that have been viewed in that window or tab</td>
</tr>
<tr>
<td>window.history.length</td>
<td>Number of items in history object</td>
</tr>
<tr>
<td>window.screen</td>
<td>Reference to screen object</td>
</tr>
<tr>
<td>window.screen.width</td>
<td>Accesses width property of screen object</td>
</tr>
<tr>
<td>window.screen.height</td>
<td>Accesses height property of screen object</td>
</tr>
</tbody>
</table>
# Using BOM Objects (Some Methods)

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>window.alert()</td>
<td>Create modal dialog box with message (user must click OK button to close it)</td>
</tr>
<tr>
<td>window.open(url)</td>
<td>Open new browser window with the specified URL</td>
</tr>
<tr>
<td>window.print()</td>
<td>Tell browser that user wants to print contents of current page (act like user has clicked a print option)</td>
</tr>
<tr>
<td>window.history.back()</td>
<td>Move backward through history</td>
</tr>
<tr>
<td>window.history.forward()</td>
<td>Move forward through history</td>
</tr>
<tr>
<td>window.history.go(step)</td>
<td>Move to specific page from session history (step specifies the number of pages, forward or backward)</td>
</tr>
<tr>
<td>history.pushState(state, title, url)</td>
<td>Create a new entry (or add a URL) at the top of the browser history</td>
</tr>
<tr>
<td>history.replaceState(state, title, url)</td>
<td>Modify the current entry (current URL at the top) of the browser history</td>
</tr>
</tbody>
</table>
Using DOM Objects

• Not part of HTML or JS

• Separate rules implemented by all major browser markers

• Two primary purposes:
  • Making a model of the HTML page
    • Specifies how browsers should create a model of an HTML page
  • Accessing and changing the HTML page
    • Specifies how JS can access and update the contents of a web page
DOM: Four Types of Nodes

- Document node
  - `<html>`
  - `<head>`
    - `<meta>`
    - `<title>`
    - `<link>`
  - `<body>`
    - `<a>`
    - `<table>`
      - `<tr>`
      - `<td>`
  - `<font>`
    - `size +2`
## Using DOM Objects
(Some Properties and Methods)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>document.title</code></td>
<td>Title of current document</td>
</tr>
<tr>
<td><code>document.lastModified</code></td>
<td>Date on which document was last modified</td>
</tr>
<tr>
<td><code>document.URL</code></td>
<td>String containing URL of current document</td>
</tr>
<tr>
<td><code>document.domain</code></td>
<td>Domain of current document</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>document.write()</code></td>
<td>Write text to document</td>
</tr>
<tr>
<td><code>document.getElementById(id)</code></td>
<td>Return element whose id attribute matches the specified id</td>
</tr>
<tr>
<td><code>document.querySelectorAll(selector)</code></td>
<td>Return list of elements that match the specified CSS selector</td>
</tr>
<tr>
<td><code>document.createElement(element)</code></td>
<td>Create new element</td>
</tr>
<tr>
<td><code>document.createTextNode(text)</code></td>
<td>Create new text node</td>
</tr>
</tbody>
</table>