JavaScript and Objects

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>
</tr>
</thead>
<tbody>
<tr>
<td>Reserve</td>
<td>reservation is made</td>
</tr>
<tr>
<td>Cancel</td>
<td>reservation is cancelled</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>makeReservation()</td>
<td>increases value of bookings property</td>
</tr>
<tr>
<td>cancelReservation()</td>
<td>decreases value of bookings property</td>
</tr>
<tr>
<td>checkAvailability()</td>
<td>subtracts value of bookings property from value of rooms property and returns number of rooms available</td>
</tr>
</tbody>
</table>

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

**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)**

<table>
<thead>
<tr>
<th>Method</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>changeSpeed()</td>
<td>increases or decreases value of currentSpeed property</td>
</tr>
</tbody>
</table>
JavaScript Objects

- JavaScript is an **object-based** language, not object-oriented
  - It has and uses objects, but does not support some features necessary for object-oriented languages (such as inheritance and polymorphism)

- 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
};
```

<table>
<thead>
<tr>
<th>Global frame</th>
<th>hotel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>name</th>
<th>&quot;Awesome&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>rating</td>
<td>5</td>
</tr>
<tr>
<td>rooms</td>
<td>70</td>
</tr>
<tr>
<td>bookings</td>
<td>56</td>
</tr>
<tr>
<td>pool</td>
<td>true</td>
</tr>
<tr>
<td>gym</td>
<td>true</td>
</tr>
</tbody>
</table>

```javascript
function checkAvailability() {
  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

<table>
<thead>
<tr>
<th>name</th>
<th>&quot;Awesome&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>rating</td>
<td>5</td>
</tr>
<tr>
<td>rooms</td>
<td>70</td>
</tr>
<tr>
<td>bookings</td>
<td>56</td>
</tr>
<tr>
<td>pool</td>
<td>true</td>
</tr>
<tr>
<td>gym</td>
<td>true</td>
</tr>
</tbody>
</table>

function() {
    return this.rooms - this.bookings;
}

object

<table>
<thead>
<tr>
<th>name</th>
<th>&quot;Duh&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>rating</td>
<td>3</td>
</tr>
<tr>
<td>rooms</td>
<td>45</td>
</tr>
<tr>
<td>bookings</td>
<td>27</td>
</tr>
<tr>
<td>pool</td>
<td>false</td>
</tr>
<tr>
<td>gym</td>
<td>false</td>
</tr>
</tbody>
</table>

function() {
    return this.rooms - this.bookings;
}
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';
```

![Diagram showing a hotel object with properties and a function for checking availability.](image)
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: 01/02/2017 10:19:23
Title: CS 4640 (Programming Languages for Web Apps)

CS 4640: Programming Languages for Web Applications
Syllabus — Spring 2018

Do users look at web apps the way they are? Or do users look at web apps the way they think?

Instructor contact information
Upsorn Praphamontipong
Office: Rice Hall 206
Email: upsorn
Office Hours: TBD

TA contact information
TBD

Class hours
Tuesday, Thursday 2:00-3:15

Prerequisite
CS 2150
Programming skills are required and students are expected to know HTML and Java or learn on their own.

Overview
This course presents the fundamentals of software development of web applications. The front end development, back end development, user interface design, and models for web-based information retrieval and processing are
BOM: Browser Object Model

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

- `window Object`
- `location Object`
- `history Object`
- `document Object`
- `navigator Object`
- `screen Object`
- `form Object`
- `link Object`
- `image Object`
- ...
DOM: Document Object Model

Object type: Document

Properties
- URL: http://www.cs.virginia.edu/~up3f/cs4640/syllabus.html
- lastModified: 01/02/2017 10:19:23
- Title: CS 4640 (Programming Languages for Web Apps)

Event
- Load: 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

Instructor contact information
- Upsorn Praphamontripong
- Office: Rice Hall 206
- Email: upsorn
- Office Hours: TBD

TA contact information
- TBD

Class hours
- Tuesday, Thursday 2:00pm - 3:15pm, Olin

Prerequisite
- CS 2150
- Programming skills are required and students should have knowledge of Java or learn on their own.

Overview
- This course presents the fundamentals of web applications. The front end development, back end development, user interface design, and models for web-based information retrieval and processing are
How A Browser See A Web Page

CS 4640: Programming Languages for Web Applications
Syllabus — Spring 2018

Do users look at web apps the way they are? Or do users look at web apps the way they think?

Instructor contact information
Upsorn Praphamontripong
Office: Rice Hall 206
Email: upsorn
Office Hours: TBD

TA contact information
TBD

Class hours
Tuesday, Thursday 2:00pm - 3:15pm, Olsson Hall 009

Prerequisite
CS 2150
Programming skills are required and students are expected to know HTML and Java or learn on their own.

Overview
This course presents the fundamentals of software development of web applications. The front end development, back end development, user interface design, and models for web-based information retrieval and processing are
# 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><code>window.alert()</code></td>
<td>Create modal dialog box with message (user must click OK button to close it)</td>
</tr>
<tr>
<td><code>window.open(url)</code></td>
<td>Open new browser window with the specified URL</td>
</tr>
<tr>
<td><code>window.print()</code></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><code>window.history.back()</code></td>
<td>Move backward through history</td>
</tr>
<tr>
<td><code>window.history.forward()</code></td>
<td>Move forward through history</td>
</tr>
<tr>
<td><code>window.history.go(step)</code></td>
<td>Move to specific page from session history (step specifies the number of pages, forward or backward)</td>
</tr>
<tr>
<td><code>history.pushState(state, title, url)</code></td>
<td>Create a new entry (or add a URL) at the top of the browser history</td>
</tr>
<tr>
<td><code>history.replaceState(state, title, url)</code></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 change 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>`
  - `<title>`
  - `<link>`
  - `<meta>`

- **Element node**
  - `<body>`
  - `<table>`
  - `<tr>`
  - `<td>`
  - `<a>`

- **Text node**
  - CS 4640 (Programming Language for Web Applications)

- **Attribute node**
  - `http-equiv` content-type
  - `rel` stylesheet href
  - `width` 75%
  - `align` center
  - `size` +2
  - `color` red

- **Other nodes**
  - `name top`
  - `align center width 100%`
  - `width 75% align center`
  - `Syllabus —`
  - `Spring 2018`
## Using DOM Objects
(Some Properties and Methods)

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

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