北京师范大学
香港浸会大学 联合国际学院
BEIJING NORMAL UNIVERSITY · HONG KONG BAPTIST UNIVERSITY
UNITED INTERNATIONAL COLLEGE

Java Script
6B Objects

Systems and Web Development Workshop
2025 Spring

Dr. Jefferson Fong

Announcements

Make-Up Test 1

- For students who missed Test 1:
  - You must have a valid reason to take the make-up test.
    - Get an absentee form from DCS office in T3-602-R2.
  - We will only have one make-up Test 1, both sections together.
    - The date and time will be determined after we get a list of students with valid reasons.

Announcements

Assignment 2

- Assignment 2 will be in two parts.
  - All topics for first part has been covered in class.
  - All topics for second part will have been covered after this lecture.
- Will be released before Reading Week (this Thu?)
- Is due Tue 8 Apr 2025, 11:55 pm.

Objects

- References
  - https://www.w3schools.com/js/js_object_definition.asp
  - https://www.w3schools.com/js/js_objects.asp
  - Mirror sites: replace w3schools.com with w3schools.cn or w3ccoo.com

- Most of you are taking Java, so you know what an object is.

Objects

- Java, along with C++, C#, and JavaScript are Object Oriented (OO)
  - Some people say Java is “Object Obsessed” because everything must be an object.
  - JavaScript is “Object Optional”; things in JS we’ve learned up to now didn’t use objects.
  - Originally JS was built to do the things we’ve taught you; objects were added to later versions of JS.
  - The syntax for objects in Java and JavaScript is quite different.
  - Contrast that with C and JS; similar things have the same syntax.
    - Recall “just like C” in the previous lectures.

Objects are Variables

- Objects are variables too. But objects can contain many values.

Example

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

Try it Yourself »

- A JavaScript object is a **collection** of **named values**

objectLiteral.html

www.w3schools.com/js/tryit.asp?filename=tryjs_object_object

Object Properties

- The named values, in JavaScript objects, are called **properties**.

```
var person = {
      firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"
};
```

Property	Value
firstName	John
lastName	Doe
age	50
eyeColor	blue

Object Methods

- Methods are **actions** that can be performed on objects.
- An **object method** is an object property containing a **function definition**.

Property	Value
firstName	John
lastName	Doe
age	50
eyeColor	blue
fullName	function() {return this.firstName + " " + this.lastName;}

- “this” refers to the “owner” of the function.
  - Here “this” is the person object that “owns” the fullname function.

Creating a JavaScript Object

- There are different ways to create new objects:
  1) Define and create a single object, using an object literal.
  2) Define and create a single object, with the keyword new.
  3) Define an object constructor, and then create objects of the constructed type.

1) Using an Object **Literal**

The easiest way to create a JavaScript Object.

- Using an object literal, you both define and create an object in one statement.

```
var person = {
    firstName:"John", lastName:"Doe", age:50,
    eyeColor:"blue"
};
```

Creating a JavaScript Object (literal).

John is 50 years old.

- The code creates a new JavaScript object with four properties:

objectLiteral.html

www.w3schools.com/js/tryit.asp?filename=tryjs_object_create_1

2) Using the Keyword **new**

- Creates a new JavaScript object with four properties

```
var person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";
```

- Both Literal and new Object() creates the same thing.
- Literal is preferred because it's simpler and runs faster.

Object Properties

- https://www.w3schools.com/js/js_object_properties.asp

- Properties are the values associated with a JavaScript object.
- A JavaScript object is a collection of unordered properties.
- Properties can usually be changed, added, and deleted, but some are read only.

Accessing Properties

- The syntax for accessing the property of an object:

```
objectName.property    // e.g. person.age
person.firstname + " is " + person.age + " years old.";
```

or

```
objectName["property"] // e.g. person["age"]
person["firstname"] + " is " + person["age"] + " years old."
```

or

```
objectName[expression] // e.g. x = "age"; person[x]
x = "age";
person["firstname"] + " is " + person[x] + " years old."
```

3) Object Constructors

- Examples from previous slides can only create single objects.

```
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
}
```

- The function Person() is an object constructor function.
  - It’s good practice for constructors to begin with upper-case.

objectConstructor.html

www.w3schools.com/js/tryit.asp?filename=tryjs_object_constructor

Object constructor function

- We can create new object of same type by calling the constructor function with the new keyword

```
const myFather
= new Person(“John", “Deer", 50, “black");

const myMother
= new Person(“Jane", “Doe", 48, "green");
```

- Use const because the object property values will not change after the object is created.
- Just like Java.

objectConstructor.html https://www.w3schools.com/js/tryit.asp?filename=tryjs_object_constructor1

Adding a Method to a Constructor

- Methods are actions that can be performed on objects.
- A **method** is a property containing a **function definition**.

Property	Value
firstName	John
lastName	Doe
age	50
eyeColor	blue
fullName	function() {return this.firstName + " " + this.lastName;}

Adding a Method to a Constructor

- Use **myFather.fullName ()** to output both first name and last name

```
document.getElementById("demo").innerHTML = "My father is " +
myFather.fullName();
```

Adding Method to Constructors

My father is John Doe

- Instead of :

```
document.getElementById("demo").innerHTML = "My father is " +
myFather. firstName + "  " + myFather.lastName ;
```

addMethodToConstructor.html

www.w3schools.com/js/tryit.asp?filename=tryjs_object_constructor6

Student Example

(A) An array to store objects.

(B) Constructor

```
11 <script>
12 var student =[];
13
14 // Constructor function for Person objects
15 function Student(first, last, sid, dob) {
16   this.firstName = first;
17   this.lastName = last;
18   this.studentId = sid;
19   this.dateOfBirth = dob;
20   this.age = function(){
21     today = new Date();
22     x = new Date(dob);
23     return today.getFullYear() - x.getFullYear()
24     }
25 }
```

studentExample.html in sample code folder “6b Object_samples”

Student Example

In studentExample.html:

(A) var ***student*** = [] // array to store objects

(B) function ***Student*** (first, last, sid, dob) // constructor

(C) student.push(new Student(…)) // adds objects to the array

(D) display1() – for loop though the array

(E) sort()

(F) display2() - for..in loop through the array

(G) Use Method 1 for function sort()

- You will learn about sort() in Algorithms class.

studentExample.html in sample code folder "6b Object_samples"

Student Example

(C) Add objects to array

(D) to (F)

(G)

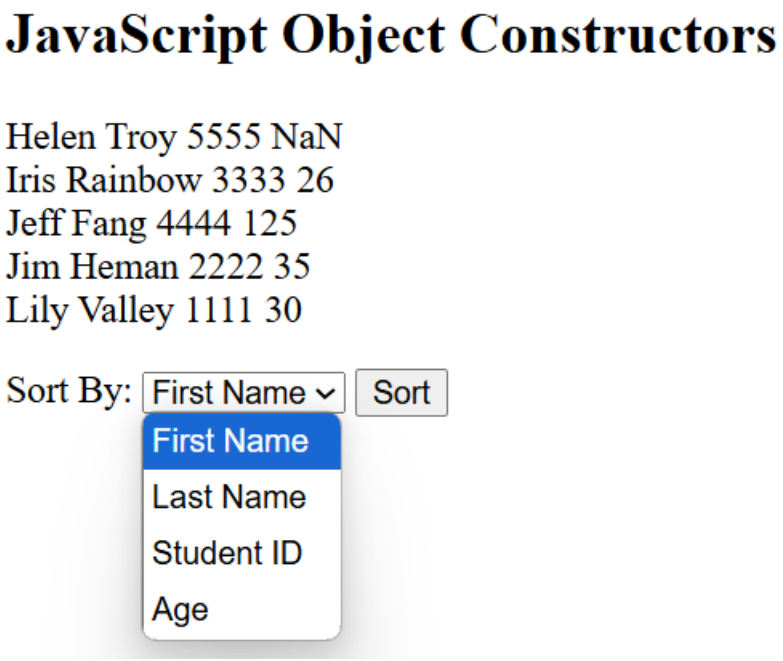
```
student.push(new Student("Lily", "Valley", 1111,
student.push(new Student("Helen", "Troy", 5555, "

display1();
sort();
display2();

// Method 1: Sorting String Attribute in Object
function sort(){
    student.sort(function(a, b) {
        return (a.lastName > b.lastName) ? 1 : ((
    });
}
```

studentExample.html in sample code folder “6b Object_samples”

Class Exercises: StudentExample2.html

StudentExample2.html

- Create a function to sort the display something like the figure at right.
- The user can sort by first name, last name, student ID or age.



1) Make a copy from StudentExampl.html

2) Create a drop-down list

- First create the code for Last Name, because you have that basic code in StudentExamp.html.

3) Create the "Sort" button to call the sort() function.

studentExample.html in sample code folder "6b Object_samples"

Class Exercises: StudentExample2.html

(4) Modify the sort() function

```
// Sorting String Attribute in Object -- Method1
function sort(){
    var k = document.getElementById("sort").value;
    switch(k) {
        case "lastName":
            student.sort(function(a, b) {
                return (a.lastName > b.lastName) ? 1 :
                    ((b.lastName > a.lastName) ? -1 : 0);
            });
            break;
        case "studentId":
```

(5) After you have sort by last name working, do the other options.

studentExample.html in sample code folder “6b Object_samples”