NASSCOM® FOUNDATION

Directorate General of Training

Powered by Microsoft

Session 2: OOP (Object Oriented Programming) and Object Type

Session 2: OOP (Object Oriented Programming) and Object Type

Objects in JavaScript

- JavaScript is designed on a simple object-based paradigm
- An object is a collection of properties, and a property is an association between a name or key and a value
- Objects in JavaScript, just like in many other programming languages, can be compared to objects in real life
- In JavaScript, an object is a standalone entity, with properties and type

Creating Objects in JavaScript

There are three ways of creating an object in JavaScript and they are as follows:

- Using an object initialiser
- Using an object constructor function
- Using the object dot create function

Creating Objects Using Object Initialiser

- Create an object using an object initialiser
- Object initialisers are expressions, and each object initialiser results in a new object being created
- Identical object initialisers create distinct objects that will not compare to each other as equal

Creating Objects Using Object Initialiser – Example

The example for creating an object using Object Initialiser is shown on the screen.

- Creates a student object with name, age and course as attributes and with their respective values
- As an object variable with the attributes and their respective values

Creating Objects Using the Constructor Function

JavaScript uses special functions called constructor functions to define and initialise objects and their features.

- There are two methods to create an object and store it in the variable person1:

Method 1

```
let person1 = new Object();
```

‘new’ is a reserved keyword

Object() is a class constructor function that is used to create objects.

The above code creates a blank object. We can then add properties and methods to this object using dot or bracket notation as shown.

```
person1.name = 'Chris';
person1['age'] = 38;
person1.greeting = function() {
alert('Hi! I\'m ' + this.name + '.');
```

Method 2

We can also pass an object literal to the Object() constructor as a parameter and prefill it with properties/methods.

```
let person1 = new Object({
  name: 'Chris',
  age: 38,
  greeting: function() {
    alert('Hi! I\'m ' + this.name + '.');
  }
});
```

Creating Objects Using the Constructor Function – Example

JavaScript compiler clearly invokes the constructor automatically even without explicitly calling it as shown in the example given below.

- **Constructor Function being automatically invoked**
- Creates a student object with name, age and course as properties and with their respective values using the constructor function

Program

```
function stud(name,age,course){
        this.name = name;
        this.age = age;
        this.course = course;
}
var student =
stud(“Swamy”,20,”Computer Science”)
```

Creating Objects Using Object.create() Function

- Objects can also be created using the Object.create() method
- The example for creating new objects using Object.create() method is shown

Program

```
// Animal properties and method
encapsulation
var Animal = {
   type: 'Invertebrates', // Default value of
properties
   displayType: function() {    // Method
which will display type of Animal
      console.log(this.type);
   }
};
// Create new animal type called animal1
var animal1 = Object.create(Animal);
animal1.displayType(); //
Output:Invertebrates
// Create new animal type called Fishes
var fish = Object.create(Animal);
fish.type = 'Fishes';
fish.displayType(); // Output:Fishes
```

Object Properties

The properties or attributes of an object define the characteristics of the object. We use a simple dot-notation to access the properties of an object.

The properties of the myCar object include the make, model and year.

```
var myCar = new Object();
myCar.make = 'Ford';
myCar.model = 'Mustang';
myCar.year = 1969;
```

<u>Property name is assigned to a variable</u>

```
var propertyName = 'make';
myCar[propertyName] = 'Ford';
propertyName = 'model';
myCar[propertyName] = 'Mustang';
```

Recap:

- Objects can contain related data and code, which represent information.
- We can also access properties by assigning the property name to a variable.