**JavaScript Objects**

In JavaScript, objects represent a fundamental and crucial data type, serving as the cornerstone for modern JavaScript development. Distinguished from primitive data types such as Number, String, Boolean, null, undefined, and symbol, which store a single value each, objects are more intricate. They can encompass a combination of primitive data types and reference data types.

Objects, classified as reference data types, involve variables assigned a reference or pointer to a value. This reference points to the memory location where the object is stored, and the variables don't directly store the value. Broadly defined, objects in JavaScript can be described as an unordered assortment of related data, comprising primitive or reference types, presented in "key: value" pairs. These keys, known as properties and methods for variables and functions, respectively, define the object's context.

For example, if an object represents a student, it could have properties like name, age, address, id, and methods like updateAddress, updateName, etc.

**Objects and Properties**

A JavaScript object associates properties with it, akin to variables attached to the object. Object properties resemble regular JavaScript variables but are distinct due to their association with objects. Accessing object properties involves using dot notation:

objectName.propertyName

Both the object name and property name are case-sensitive. Properties are defined by assigning values to them. Unassigned object properties are undefined, not null. Accessing or setting properties can also be achieved using bracket notation. Objects are sometimes referred to as associative arrays because each property is associated with a string value that serves as an access key. Bracket notation is useful for dynamic property determination and accessing properties with non-standard names.

**Creating Objects in JavaScript**

*Create JavaScript Object with Object Literal*

A straightforward method to create a JavaScript object is through object literals, defining properties and values within curly braces.

let bike = {name: 'SuperSport', maker:'Ducati', engine:'937cc'};

*Create JavaScript Object with Constructor*

Using constructors, which are essentially functions, and the new keyword allows the creation of multiple objects of the same type.

function Vehicle(name, maker) {

this.name = name;

this.maker = maker;

}

let car1 = new Vehicle(’Fiesta’, 'Ford’);

let car2 = new Vehicle(’Santa Fe’, 'Hyundai’)

console.log(car1.name); //Output: Fiesta

console.log(car2.name); //Output: Santa Fe

*Using the JavaScript Keyword new*

Creating a new JavaScript object with properties can also be accomplished using the new keyword.

var person = new Object();

person.firstName = “John”;

person.lastName = “Doe”;

person.age = 50;

person.eyeColor = “blue”;

*Using the Object.create method*

Objects can be instantiated using the Object.create() method, allowing the selection of a prototype object without defining a constructor function.

// 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