In JavaScript, an object is a composite data type that allows you to store and organize data. It consists of key-value pairs, where each key is a string (or symbol) and each value can be any data type, including other objects.

The Internal representation of an object in JavaScript may vary across different JavaScript engines (e.g., V8, SpiderMonkey). However, conceptually, objects typically have properties and an internal [[Prototype]] link.

Properties: These are the key-value pairs that define the characteristics of the object. You can access properties using dot notation (object.property) or bracket notation (object[“property”]).

[[Prototype]] Link: This is a reference to another object. It forms the basis for JavaScript’s prototypal inheritance. When you try to access a property on an object, and the property Is not found on the object itself, JavaScript looks for it in the object’s prototype chain.

Here's a simple example:

Javascript

Copy code

// Creating an object

Let person = {

Name: “John”,

Age: 30,

};

// Accessing properties

Console.log(person.name); // Output: John

Console.log(person[“age”]); // Output: 30

Understanding the internal workings of objects is crucial for effective JavaScript programming, especially when dealing with inheritance and object-oriented patterns.