

Object and Class

By CODEMIND Technology

Contact us 966 5044 698
966 5044 598

Object

- What is Object
- How to defined object
- Object methods
- Different ways to create Object
- Function as a property value
- Object inside Object
- How to traverse object using for in loop
- Assignments
- Class
- Constructor Function
- Prototype

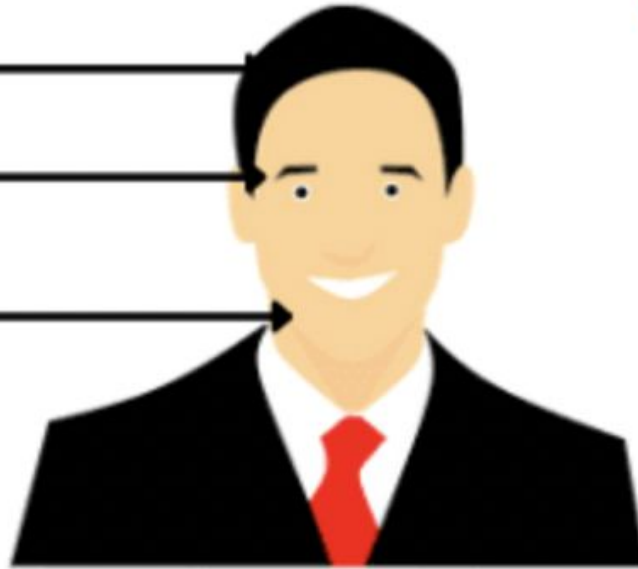
A real world example - Object

Object's properties

Black hair

Black eyes

Fair skin



Object's actions

Eat

Sleep

Walk

Play

Study

Sciencetech Easy

A person is an object

Fig: Example of an object in JavaScript

What is Object in JS

An object is a non-primitive data type in JavaScript which is used to store data in the form of key and value

There are different ways to create an object and few of them are

1. Using object literals
2. Using Class
3. Using function constructor

What is object in JS ?

An object is a non-primitive data type in JavaScript which is used to store data in the form of key and value

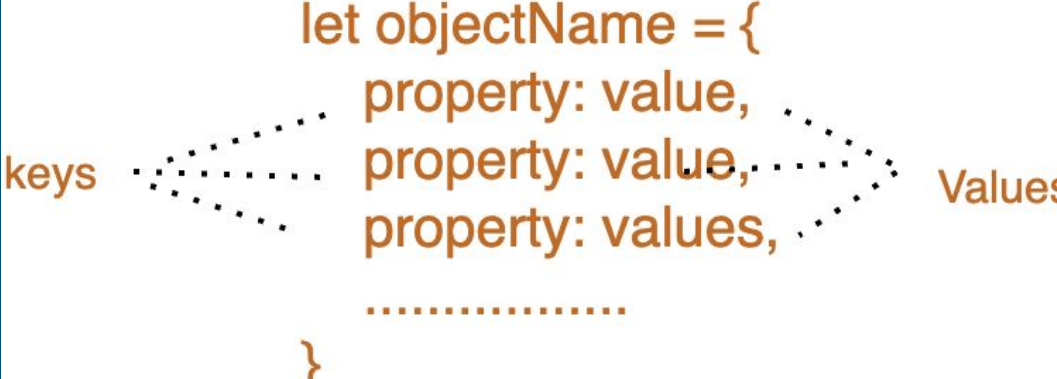
Object is similar to variable means objects are also named container only difference is that it used to store collection of data

Note: we can use let, const, var reserved word to create object

But best practise is to use const

Syntax: ----->

```
let objectName = {  
  property: value,  
  property: value,  
  property: values,  
  .....  
}
```



Accessing object properties

Using dot Notation

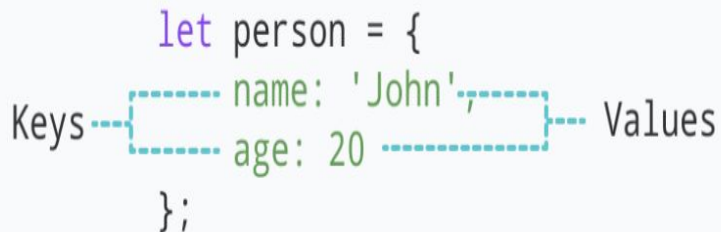
Syntax:

`objectName.property_name`

Using bracket Notation

Syntax:

`objectName["property_name"]`



```
let person = {  
  name: 'John',  
  age: 20  
};
```

The diagram shows a JavaScript object literal. A dashed box encloses the two key-value pairs, with the word 'Keys' to its left and 'Values' to its right. The keys 'name' and 'age' are highlighted in green, and the values 'John' and '20' are also highlighted in green.

Example:

`person.name;`

Example:

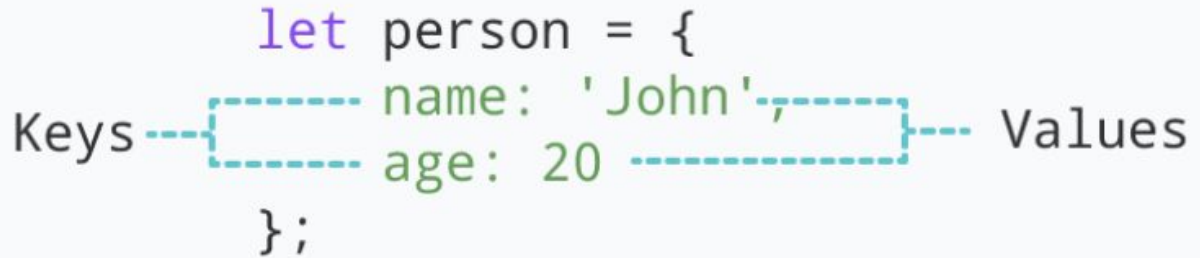
`person["name"]`

How to add properties in object

Syntax: `objectName.property_name = value;`

OR

`objectName["property_name"] = value;`



The diagram shows the JavaScript code `let person = { name: 'John', age: 20 };;` with annotations. A dashed blue box encloses the object literal `{ name: 'John', age: 20 }`. A dashed line connects the word "Keys" to the left side of the box, and another dashed line connects the word "Values" to the right side of the box. The code is color-coded: `let` is purple, `person` is black, `=` is black, `{` is black, `name` is green, `:` is black, `'John'` is green, `,` is black, `age` is green, `:` is black, `20` is green, `}` is black, and `;;` is black.

```
let person = {  
  name: 'John',  
  age: 20  
};;
```

Example →

`person.company = "CODEMIND Technology";`

OR

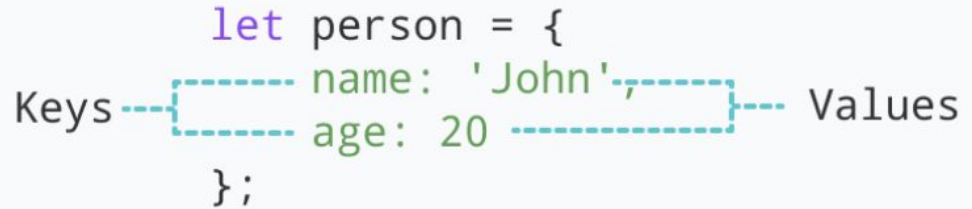
`Person["company"] = "CODEMIND Technology";`

Delete property from an object

Syntax: delete object_name.property_name;

Example →

delete person.age;



The diagram shows a code snippet: `let person = { name: 'John', age: 20 };`. A dashed blue box encloses the object's contents. To the left of the box is the word "Keys" with a dashed line pointing to the box. To the right of the box is the word "Values" with a dashed line pointing to the box. The box contains the text `name: 'John',` and `age: 20` on two lines.

How to update an object property:

Syntax: object_name.property_name = "new value";

Person.name = "Mohit Sharma";

Creating an empty object

Syntax: let objectName = { };

Example:

```
let employee = { };  
employee.name = "Mohit";
```

What if we use const instead of let, Can we add properties then?

An object overview with properties, values and methods

A user object with keys, values and methods

An object is a data type in JavaScript that is used to store a combination of data in a simple key-value pair. That's it.

Key

These are the keys in `user` object.

```
var user = {  
  name: "Aziz Ali",  
  yearOfBirth: 1988,  
  calculateAge: function(){  
    // some code to calculate age  
  }  
}
```

Value

These are the values of the respective keys in `user` object.

Method

If a key has a function as a value, it's called a method.

Object Methods in JS or Object method shorthand

In JavaScript, an object can also contain a function. For example,

```
const person = {  
  name: 'Sam',  
  age: 30,  
  // using function as a value  
  greet: function() { console.log('hello') }  
}  
  
person.greet(); // hello
```

Nested Objects

An object can also contain another object. For example,

```
// nested object
const student = {
  name: 'John',
  age: 20,
  marks: {
    science: 70,
    math: 75
  }
}

// accessing property of student object
console.log(student.marks); // {science: 70, math: 75}

// accessing property of marks object
console.log(student.marks.science); // 70
```

Assignment 0D: File→ 11_object_assigD.js

Create an object with name → professor

1. Think of all the properties that you could add (Add minimum 5 properties)
2. Also include nested object 'degrees' like `engineering: 'CSC', PHD: 'Adv Computing'` and few more.
3. Also add one array → 'certificates' with his certificates like 'Hacker Rank Participation', 'Certificate in IFE course', 'Certificate in Adv Programming'
4. Add function as a value which should concat all degrees in step 2, please return as string and log on console with - Teacher degrees are total degrees are:
5. Try to add new property like `totalExperience: "14 years"` and log on console
6. Modify any existing property and log complete object on console
7. Add one new certificate → "Oracle Certified" at the end of array → certificates
8. Log the last element of the array certificates.

Assignment 0E: File→ 11_object_assigE.js Object with data member and member function

Create an objects → sbiBank, axisBank, hdfcBank, yesBank with data members or properties such as

- A. Data members: bankName, location, accountNo, ifsc, interestRate
- B. Set the values separately according to their bank details
- C. Add a member function → showDetails(), In this object log the data members of that object in one line. Note: This function doesn't have return value and no args
- D. Invoke the function showDetails() on each object separately

How to get all entries from objects

```
Object.entries(person);
```

How to get all keys of object

```
Object.keys(person); // Returns an array of keys
```

How to get all values of objects

```
Object.values(person); // Returns an array of values
```

```
let person = {  
  name: 'John',  
  age: 20  
};
```

Keys --- { } --- Values

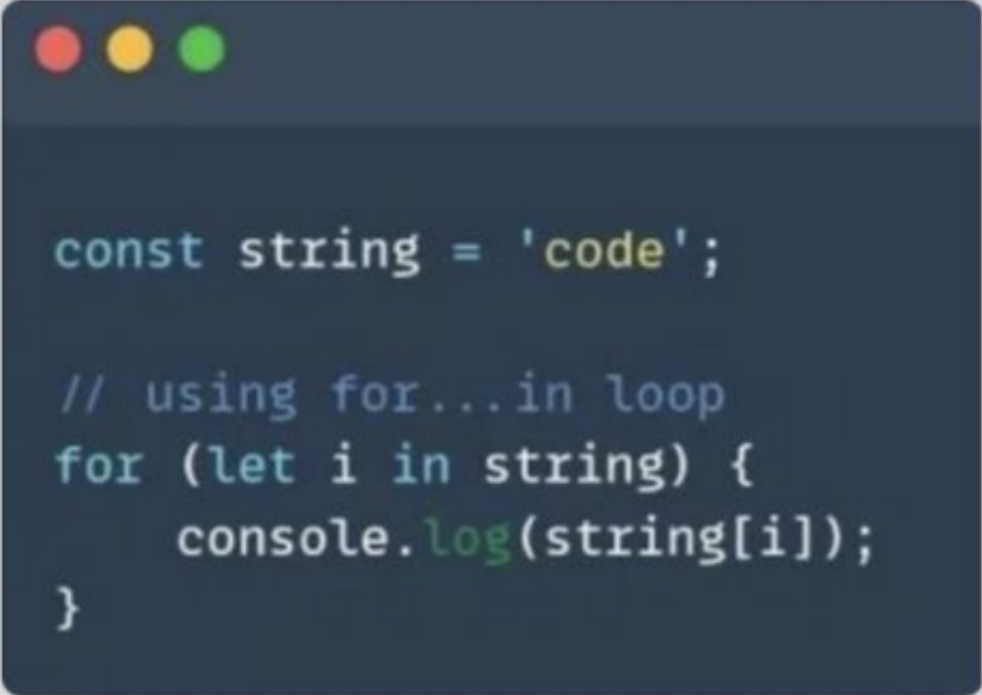
for in loop

- The for...in loop in JavaScript allows you to iterate over all property keys of an object.
- The syntax of the for...in loop is:

```
for (key in object) {  
    // body of for...in  
}
```

- In each iteration of the loop, a key is assigned to the key variable.

- for...in loop to iterate over string values.



```
const string = 'code';  
  
// using for...in loop  
for (let i in string) {  
    console.log(string[i]);  
}
```

How to traverse over object

The for...in allows you to access each property and value of an object without knowing the specific name of the property.

Syntax→

```
for( const key in object_name) {  
    // statements  
}
```

Example →

```
const student = {  
    name: 'Monica',  
    class: 7,  
    age: 12  
}  
  
// using for...in  
for ( let key in student ) {  
  
    // display the properties  
    console.log(`${key} => ${student[key]}`);  
}
```

'In' operator: To check if property exists or not in the object

Syntax: "property_name" **in** object_name;

Note: Return true if property exist in the object else returns false

Example:

"age" in person

```
let person = {  
  name: 'John',  
  age: 20  
};
```

Keys --- { } --- Values

Defining Object with 'const keyword'

When we define object as 'const' then after defining object that reference variable will not be changed to point any other object

```
const student = {  
  name: "Mohit",  
  rollNo: 1234  
}  
student = { // Not allowed: TypeError: Assignment to constant variable.  
  city: "Pune"  
}
```

Object Freezing

Freezing an object does not allow new properties to be added to an object and prevents from removing or altering the existing properties.

Object.freeze(): We can use this method for freezing objects and arrays.

Freezing an object:

Note: Freezing an object and defining object with const keyword are two different things.

```
let student = {  
  name: "Mohit",  
  rollNo: 1234  
}  
Object.freeze(student);  
student.name = "Mohit Sharma"; // Not allowed to update  
student.city = "Pune"; // Not allowed to add new property  
console.log(student);
```

Freezing an Array

Freezing an array does not allow new element to be added to an array and prevents from removing or altering the existing elements.

Object.freeze(): We can use this method for freezing objects array.

Freezing an Array:

```
let array = [2, 3, 4, 5];  
Object.freeze(array);  
array.push(6); // TypeError: Cannot add property 4, object is not extensible  
array.shift(); // TypeError: Cannot assign to read only property '0' of object '[object Array]'  
|
```

Note: When we declare array as const then that array reference variable can not point to any other array

Object.assign(): Syntax: Object.assign(target, ...sources);

This method is used for

1. To Clone an object

```
const emp = {  
  emp_name: "Anil",  
  company: "TCS"  
}  
  
// 1. Cloning an object  
const cloned_emp = Object.assign({}, emp);  
console.log(cloned_emp);
```

2. To merge an objects

```
const emp = {  
  emp_name: "Anil",  
  company: "TCS"  
}  
  
const emp_address = {  
  city: "Pune",  
  pin: 431202  
}  
  
// 2. Merge an Object  
const merged_obj = Object.assign({}, emp, emp_address);  
console.log(merged_obj);
```

Assignment A → Object cloning and Traversing, File → 11_objectCloneAssig.js

1. Create the object → 'bankSbi' using literals with at least 4 properties
2. Create the object → 'bankLocation' using literals with properties: street, city, pin
3. Clone the step 1 (bankSbi) and step 2 (bankLocation) objects using

- Object.assign()

Note: Log the cloned object details on console with meaning message using strings template

4. Create the object using literals → rateOfInterest with properties
 - homeLoanInterest, personalLoanInterest, dueInterest
5. Merge the step 1, step 2 and step 4 objects into new object namely → sbiDetails
Log all the properties that 'sbiDetails' got after merging with meaning message
6. Traverse this merged object - step 5 using for in loop and log details properly



Class and Object

Real word Example: Person

Different ways to create **Object**:

- Object literals
- Using class
- Constructor function

JS Class

Class

Person

**Data
Members**

OR Properties
OR States
OR Attributes

unique_id
name
age
city
gender

Methods

OR
Action OR
Behaviour

eat()
study()
sleep()
play()



name- John
age- 35
city- Delhi
gender- male



name- Dessy
age- 20
city- Pune
gender- female

Class in JavaScript

A class is a BLUEPRINT of an object. You can create an object from the class.

A class is introduced in ES6

A class contains:

- The set of attributes/properties
- The set of behavior or methods or actions
- Also class can have constructor

Object is the “instance” of a Class

Reserved keywords

- class
- new
- constructor
- this

Class and creating objects from it.

```
class Person {  
  name;  
  age;
```

```
  constructor( name, age ) {  
    this.name = name;  
    this.age = age;  
  }
```

```
  play() {  
    console.log("He love playing cricket");  
  }
```

```
  eat() {  
    console.log("He likes Pizza ");  
  }
```

```
}
```

Creating an Object from Class Person

```
let anilPerson = new Person("Anil", 25);
```

```
let rituPerson = new Person("Ritu", 23);
```

Accessing Properties and methods

```
anilPerson.name;  
anilPerson.play();
```

Naming Conventions

Class Name:

- First character should be capital
- Number is not allowed
- Special char _ is allowed

Object name:

- First letter should be small case
- Special char _ is allowed
- When more than word then name like 'anilPerson'

What is constructor()

The constructor() is a special method and is called automatically by new keyword to initialize the object.

Note: We can have constructor() with or without arguments

What is 'new'?

To create object we use 'new' keyword.

What is this?

This is a keyword which points to current object. When the class properties name and constructor or method arguments are with same name in that case we use the 'this' keyword to differentiate.

Assignment 0A: File→ 12_classAssigA.js

1. Define a class for Vehicle which should contains.

- Minimum 5 Properties or attributes or data members:
- Constructor

Create 5 objects from Vehicle class and add into array → arrayOfVehicles. Traverse it and log the details with meaning message and not as object.

2. Define a class for College which must contain

- Properties or attributes or data members: minimum 4
- Constructor

Create 4 objects from College class

3. Write a function → traverseObject() with one arg. such that it should traverse the complete given object using for in loop and log the output as [console.log(`\${key} \${element}`)]

3.1 Call this traverseObject() function by passing one by one object of College class

4. Find the given number is Prime Number or not ? Ex. 11

Different ways to create Objects

1. Using object literals
2. Using Class
3. Using function constructor

Constructor Function

function Person() is an object constructor function.

To create an object from a constructor function, we use the new keyword.

In JavaScript, a constructor function is used to create objects. For example,

```
// constructor function
function Person () {
  this.name = 'John',
  this.age = 23
}

// create an object
const person = new Person();
```

JavaScript Constructor Function Parameters

You can also create a constructor function with parameters. For example,

```
// constructor function
function Person (person_name, person_age, person_gender) {

    // assigning parameter values to the calling object
    this.name = person_name,
    this.age = person_age,
    this.gender = person_gender,

    this.greet = function () {
        return ('Hi' + ' ' + this.name);
    }
}

// creating objects
const person1 = new Person('John', 23, 'male');
const person2 = new Person('Sam', 25, 'female');

// accessing properties
console.log(person1.name); // "John"
console.log(person2.name); // "Sam"
```

Instanceof operator

Is used to check type of object

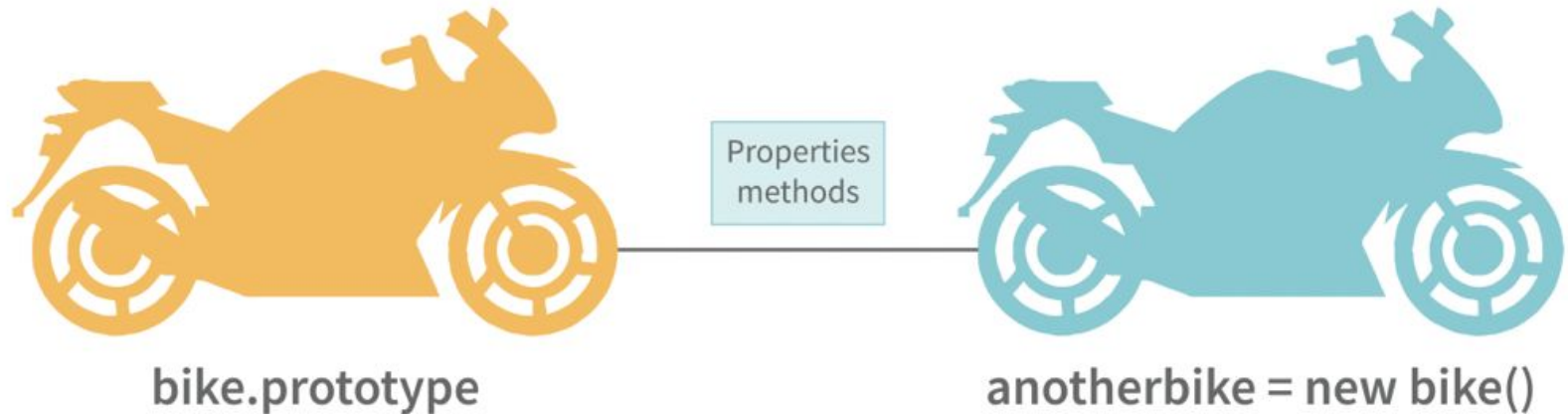
Syntax: object_name instanceof class_Name;

Example:

What is Blueprint ?

Orange color bike is the blueprint that is prototype and

Green color bike is the main object



Prototype in JavaScript

A prototype is a blueprint of an object. The prototype allows us to use properties and methods on an object even if the properties and methods do not exist on the current object.

The prototype object is special type of object to which additional properties can be attached to it which will be shared across all the instances of it's constructor function.

All javascript objects inherit properties from a prototype. For example,

- Array objects inherit properties from the Array prototype.
- Date objects inherit properties from the Date prototype
- On top of the chain is Object.prototype. Every prototype inherits properties and methods from the Object.prototype.

```
var arr = [];  
arr.push(2);  
  
console.log(arr); // Outputs [2]
```

IMP Example to explain in interview

- In the code above, as one can see, we have not defined any property or method called push on the array “arr” but the javascript engine does not throw an error.
- The reason is the use of prototypes. As we discussed before, Array objects inherit properties from the Array prototype.
- The javascript engine sees that the method push does not exist on the current array object and therefore, looks for the method push inside the Array prototype and it finds the method.

Note: Whenever the property or method is not found on the current object, the javascript engine will always try to look in its prototype and if it still does not exist, it looks inside the prototype's prototype and so on.


```
// constructor function
function Person () {
    this.name = 'John',
    this.age = 23
}

// creating objects
let person1 = new Person();
let person2 = new Person();

// adding new property to constructor function
Person.prototype.gender = 'Male';

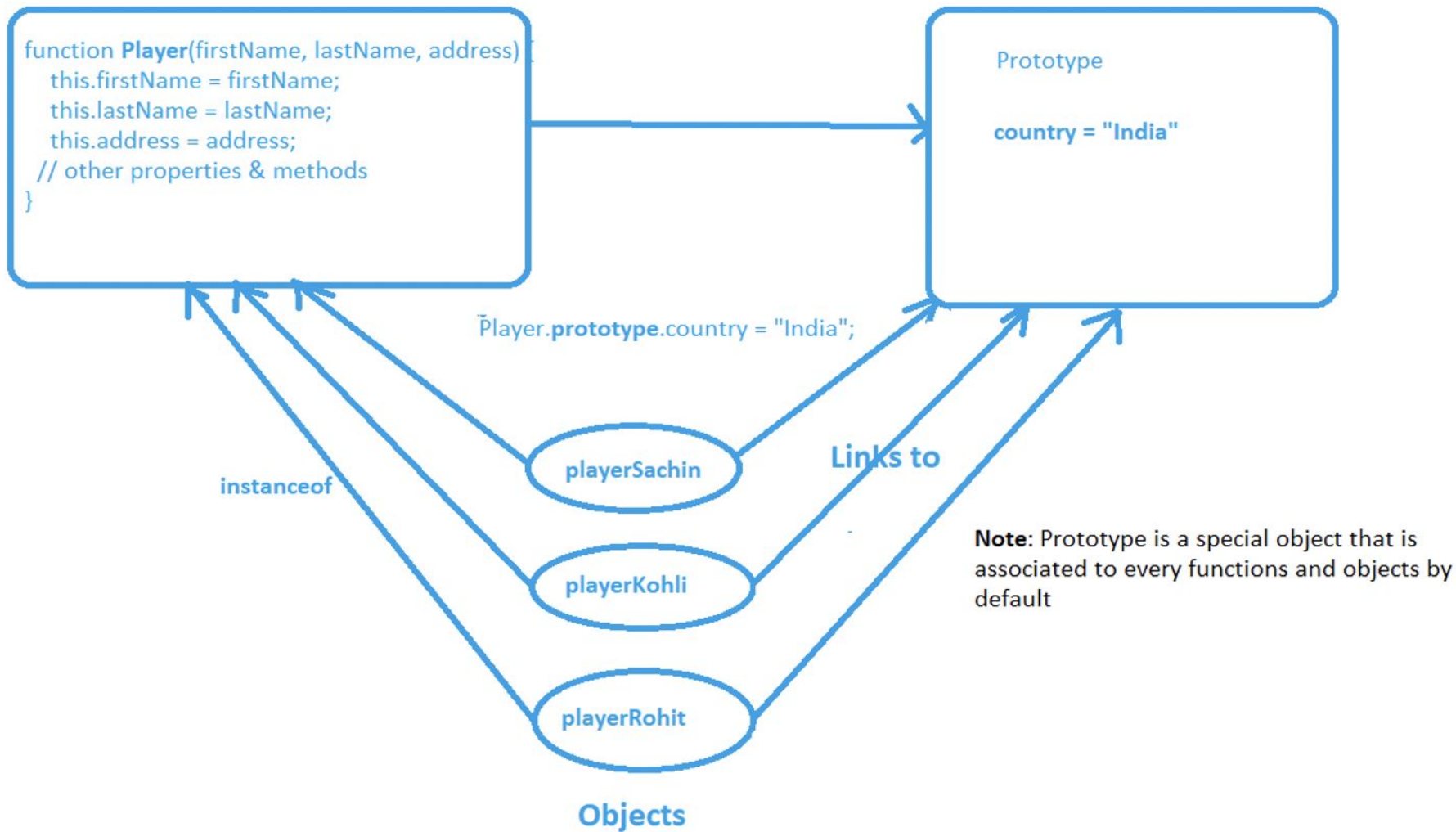
console.log(person1.gender); // Male
console.log(person2.gender); // Male
```

```
// Defining a Function Constructor
function Player(name, age, runs) {
  this.name = name;
  this.age = age;
  this.runs = runs;
  this.playerDetails = function() {
    return name + " " + age + " " + runs;
  }
}
```

The '**new**' keyword is used to create a new object(instance) from the constructor

Passing values to constructor arguments

```
// Using constructor
const virat = new Player("Virat Kohli", 31, 41000)
const rohit = new Player("Rohit Sharma Kohli", 32, 3000)
```



Built-in Objects in JavaScript

- JavaScript gives us a ton of useful built-in objects to make our lives easier. The Date and Math objects are very useful on a regular basis. Take a look at some of their features on the right.

Built-in Objects in JavaScript

JavaScript gives us a ton of useful built-in objects to make our lives easier. The **Date** very useful on a regular basis.

```
let date = new Date();

console.log(date);
console.log(date.getDate());
console.log(date.toString());
console.log(date.toTimeString());
var dt = new Date( "Aug 28, 2008 23:30:00");
console.log(dt);
const d = new Date("2015-03-25");
console.log(d);
const dd = new Date("03/25/2016");
console.log(dd);
const ddd= new Date("25 Mar 2015");
console.log(ddd);
```

List of methods used with Date and their description

Method	Description
Date()	Returns today's date and time
getDate()	Returns the day of the month for the specified date according to local_time.
getDay()	Returns the day of the week for the specified date according to local time.
getFullYear()	Returns the year of the specified date according to local time.
getHours()	Returns the hour in the specified date according to local time.
getMilliseconds()	Returns the milliseconds in the specified date according to local time.
getMinutes()	Returns the minutes in the specified date according to local time.
getMonth()	Returns the month in the specified date according to local time.
getSeconds()	Returns the seconds in the specified date according to local time.

Assignment: Constructor function, File→ 10_constructor_fun_assig.js

1. Create a constructor function with name→ Bank and add the data member such as
—bankName, location, ifscCode, branchCode
2. Create the objects such as yesBank, sbiBank, mahBank, axisBank and log the details with meaning msg as Bank Details is: \${bankName}, \${location}, \${ifscCode}, \${branchCode}
3. Add the data member → openTime = “9 AM IST” on prototype object
4. Add one more data member → closeTime =”6 PM IST” on prototype object
5. With meaningful msg Log the openTime and closeTime of object sbiBank
6. With meaningful msg Log the bankName and closeTime of object axisBank
7. With meaningful msg Log the bankName, branchCode and openTime of object yesBank

Thank you

