

JavaScript Objects { }

```
let myObject = { }
```



@adityauke



Object Definition

A pair of curly braces `{ }` containing comma `,` separated **key-value** pairs represent an object. It is a collection of unordered properties.

The key-value pair inside objects is called **property**.

Any property which contains a **function definition** is known as a **method**.

```
const myObj = {  
  firstName : 'Aditya',  
  lastName : 'Uke' // property  
  fullName : function () {  
    return `My name is ${this.firstName} ${this.lastName}`  
  } // method  
}  
console.log(myObj.fullName()) // My name is Aditya Uke
```

*The key must either be a **string** or a valid **identifier**.
Values can be of any type.*

Creating Objects

- Using object literal { }

```
const myObj = { name: 'Aditya', age: 21 }
```

- Using object constructor with new keyword

```
const myObj = new Object()  
myObj.name = 'Aditya'  
myObj.age = 21
```

Adding properties

- Dot Notation '.'

```
myObj.colorOne = 'blue'
```

- Bracket Notation '[]'

```
myObj['colorTwo'] = 'red'
```



Accessing properties

- Dot Notation '.'

```
myObj.colorOne // 'blue'
```

- Bracket Notation '[]'

```
myObj['colorTwo'] // 'red'
```

*Always write key inside **quotes** (' ') while using Bracket notation.*

Adding methods

- Dot Notation '.'

```
const myObj = { name: 'Aditya' }  
myObj.greet = function(){ return `Hello, ${this.name} this side` }
```

- Bracket Notation '[]'

```
const myObj = { name: 'Aditya' }  
myObj['greet'] = function(){return `Hello, ${this.name} this side`}
```

Accessing Methods

- Dot Notation '.'

```
const myObj = {  
  name: 'Aditya',  
  greet: function(){ return `HI, I am ${this.name}`}  
  // ES6 shorthand syntax for defining methods inside an object  
  job(){ return 'HI, I am a Developer' }  
}  
  
console.log(myObj.greet()) // HI, I am Aditya  
console.log(myObj.job())  // HI, I am Developer
```

*In ES6, shorthand syntax was introduced for defining methods inside an object. In this syntax **colon(:)** and **function** keyword are omitted.*

Accessing Methods

- Dot Notation '.'

```
const myObj = {  
  name: 'Aditya',  
  greet: function(){ return `HI, I am ${this.name}`}  
  // ES6 shorthand syntax for defining methods inside an object  
  job(){ return 'HI, I am a Developer' }  
}  
  
console.log(myObj.greet()) // HI, I am Aditya  
console.log(myObj.job())  // HI, I am Developer
```

*In ES6, shorthand syntax was introduced for defining methods inside an object. In this syntax **colon(:)** and **function** keyword are omitted.*

Getters and Setters

Getters => It is used to get the value of a property.
get keyword is used to define a getter method.

```
let user = {  
  name: "Aditya",  
  surname: "Uke",  
  
  get fullName() {  
    return `${this.name} ${this.surname}`;  
  }  
};  
  
console.log(user.fullName); // Aditya Uke
```

Getters and Setters

Setters => It is used to **set** the value of a property. set keyword is used to define a setter method.

```
let user = {  
  name: "Aditya",  
  surname: "Uke",  
  
  get fullName() {  
    return `${this.name} ${this.surname}`;  
  },  
  
  set fullName(value) {  
    [this.name, this.surname] = value.split(" ");  
  }  
};  
  
user.fullname = "Virat Kohli"  
console.log(user.fullname) // Virat Kohli
```