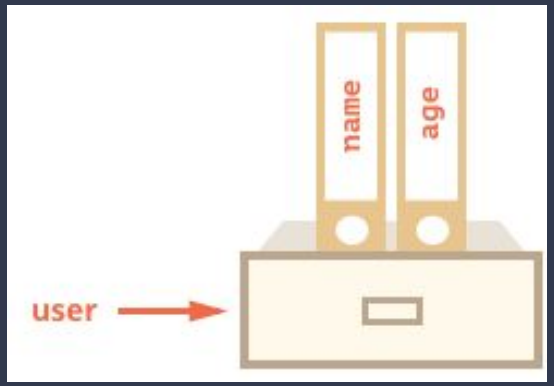
Lecture 8.1

Objects

Objects

JS Objects



The **Object** class represents one of JavaScript's data types. It is used to store various keyed collections and more complex entities.

In JavaScript, an object is a collection of properties, defined as a key-value pair.

Each property has a key and a value. The property key can be a string and the property value can be any valid value.

Creating an Object

To create an object, you use the object literal syntax. For example, the following snippet creates an empty object:

```
let empty = {};
```

The **Object literal** notation is basically an array of key:value pairs, with a colon separating the keys and values, and a comma after every key:value pair, except for the last, just like a regular array.

Creating an Object with Properties

To create an object with properties, you use the key : value syntax. For example, the following snippet creates a person object:

```
let person = {
    firstName: 'John',
    lastName: 'Doe'
};
```

The person object has two properties firstName and lastName with the corresponding values 'John' and 'Doe'.

Video - Object

Accessing Properties

1. The dot notation (.)

The following example shows how to use the dot notation to access a property of an object:

```
objectName.propertyName
```

For example, to access the firstName property of the person object, you use the following expression:

```
person.firstName
```

Accessing object with dot notation

2. Array-like notation

The following example shows how to use the array-like notation or (bracket notation) to access a property of an object:

```
objectName['propertyName'];
```

For example, to access the firstName and lastName property of the person object, you use the following expression:

```
console.log(person['firstName']);
console.log(person['lastName']);
```

When a property name contains spaces, you need to place it inside quotes. For example:

```
let address = {
    'building no': 3960,
    street: 'North 1st street',
    state: 'CA',
    country: 'USA'
};
```

To access the 'building no', you must use the array-like notation:

```
address['building no'];
```

Accessing Object via array-like notation

Modifying Object Properties

Changing the property's value

The following example shows how to use the array-like notation to access a property of an object:

```
let person = {
    firstName: 'John',
    lastName: 'Doe'
};

person.firstName = 'Jane';

console.log(person);
```

Adding a new property to an Object

Unlike objects in other programming languages such as Java and C#, you can add a property to an object after creating it.

The following statement adds the age property to the person object and assigns 73 to it:

```
person.age = 73;
```

Deleting property to an Object

To delete a property from an object, you use the delete operator:

```
delete objectName.propertyName;
```

The following example removes the age property from the person object:

```
delete person.age;
```

Check if property exists in Object

To check if a property exists in an object, you use the in operator:

```
propertyName in objectName
```

The following checks the person object for the property firstName:

```
let person = {
    firstName: 'John',
    lastName: 'Doe'
};

console.log('firstName' in person);
```

Iterate over Object properties

To iterate over all properties of an object without knowing property names, you use the for...in loop:

```
for(let key in object) {
    // ...
};
```

The following iterates over the person properties and outputs firstName:

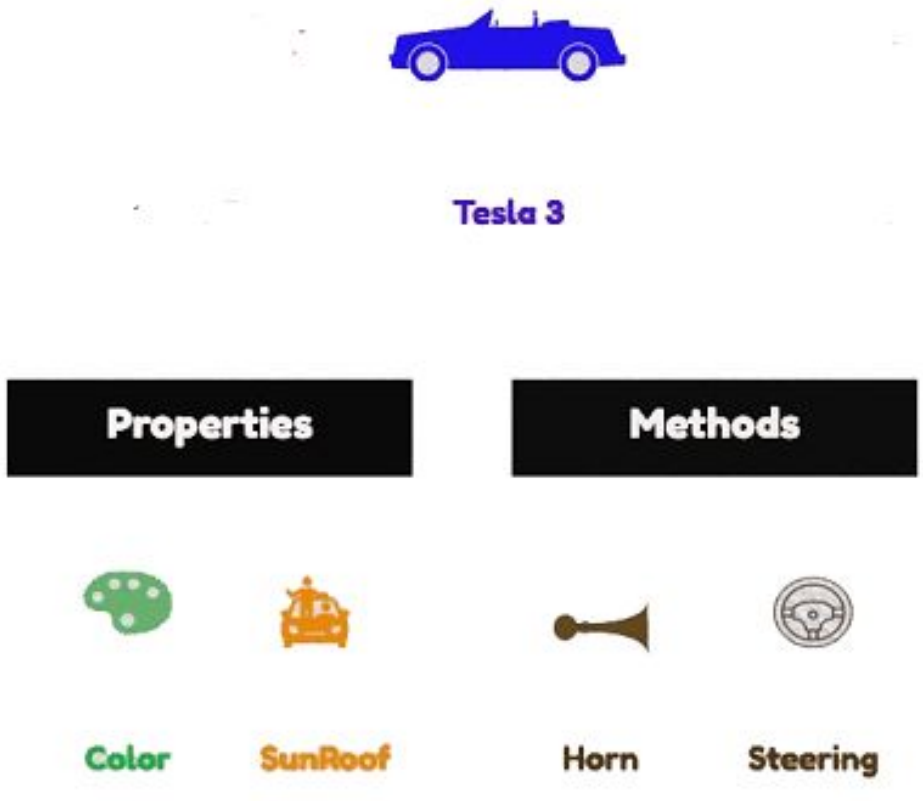
```
person = { firstName: 'Mike'};

for (const key in person) {
    console.log(person[key]);
}
```

Object Methods

Object Methods

An **Object** is a software unit of variables (properties) and methods (functions). These objects are often used to model the real-world objects that you find in everyday life. An Object's method provide the only way to access the data



Object Methods

Objects have actions. The actions are represented by functions

The following snippet adds the greet action to the person object:

```
person = {};

person.greet = function () {
    console.log('Hello, World!');
}

person.greet();
```

Method Shorthand

You can define methods using the object literal syntax:

```
let person = {
    firstName: 'John',
    lastName: 'Doe',
    greet: function () {
        console.log('Hello, World!');
    }
};
```

Video - This

The this value

Typically, methods need to access data stored in the object.

For example, you may want to develop a method that returns the full name of the person object by concatenating the first name and last name.

```
this.propertyName
```

The this value

Inside the method, the this value references the object that contains the method so you can access an object property using the dot notation:

```
getFullName: function () {
    return this.firstName + ' ' + this.lastName;
}
```

- An object is a collection of key-value pairs called properties. A property key is a string and value can be any valid value.
- Use the dot notation (.) or array-like notation ([]) to access an object property.
- The delete operator removes a property from an object.
- The in operator check if a property exists in an object.
- The for...in iterates over properties of an object.
- When functions are properties of an object, they are called methods.
- Use the this inside the method to access the object's properties.

Object Recap