Objects in detail

Creating an Object

```
const person = {
    firstName: "Shubham",
    lastName: "Kadam",
    age: 23,
    car: {
        brand: "Mercedes",
        year: 2020,
    },
};
```

The first block of code shows how to create a new object using object literal syntax. An object literal is a list of key-value pairs enclosed in curly braces `{}`. The keys are represented by strings, and the values can be any data type (strings, numbers, arrays, etc).

The Dot and Square Notation

```
// Dot notation
const person = {
    firstName: "Shubham",
};

person.dog = {
    name: "fluffy",
    age: 2,
};

person.age = 23; // Age is being added in person object
console.log(person.dog.age);  // 2

// Square bracket notation
const property = "age";

console.log(person[property]); // 23
```

The above code demonstrates two ways to access and modify the properties of an object: dot notation and square bracket notation.

With dot notation, you can access or set a property of an object using a dot (`.`) followed by the property name. For example, `person.age = 23` sets the `age` property of the `person` object to 23.

With square bracket notation, you can access or set a property using a string inside square brackets (`[]`). For example, `person['age'] = 23` sets the `age` property of the `person` object to 23.

```
const dog = {
    name: "Fluffy",
    bark: () => {
        console.log("woof,woof");
    },
};

dog.bark(); // woof,woof

// this keyword
const car = {
    name: "Lambo",
    model: 2019,
    Details: function () {
        console.log(this.name, this.model);
    },
};

car.Details(); // Lambo 2019
```

`this` notation is useful when you need to access a property whose name is stored in a variable.

Methods

```
 // Object.keys() creates an array containg the keys of an object.

// Intialize an object
const employee = {
    boss: "Shubham",
    secretary: "Samarth",
    sales: "John",
    accountant: "Oscar",
};

// Let's say we want to see all the positions(keys) in the employee object
const positions = Object.keys(employee);
console.log(positions); // [ 'boss', 'secretary', 'sales', 'accountant' ]

// Object.values() creates an array containg the values of an object.
const session = {
    id: 1,
    time: `26-July-2022`,
    device: "Mobile",
    browser: "Chrome",
};

const sessionInfo = Object.values(session);
console.log(sessionInfo); // [ 1, '26-July-2022', 'Mobile', 'Chrome' ]

// Object.entries() creates an array containg the key/value of an object.
const operatingSystem = {
    name: "Ubuntu",
    version: "20.04",
    license: "Open Source",
};

const entries = Object.entries(operatingSystem);

console.log(entries); // [
  [ 'name', 'Ubuntu' ],
  [ 'version', '20.04' ],
  [ 'license', 'Open Source' ]
]

// Intialize an object
const user = {
    username: "Shubham",
    password: "12345",
};

const admin = Object.freeze(user);

user.username = "Samarth"; // Trying to overwrite the object username
console.log(user.username); // Shubham (remains the same);

// Object.seal() prevents new properties
// from being added to an object,
// but allows the modification of existing properties
// Intialize an object

const user = {
    username: "Shubham",
    password: "12345",
};
```

```
const newUser = Object.seal(user);

newUser.username = "Samarth"; // The username will be changed
newUser.age = 23; // the age property will not be added because we applied Object.seal()

console.log(user);// { username: 'Samarth', password: '12345' }
```

The above block of code demonstrates four methods that are built into JavaScript: `Object.keys()`, `Object.values()`, `Object.entries()`, and `Object.freeze()`.

- `Object.keys()` creates an array containing the keys (property names) of an object.
- `Object.values()` creates an array containing the values of an object.
- `Object.entries()` is a built-in JavaScript method that returns an array of an object's own enumerable string-keyed property [key, value] pairs, in the same order as that provided by a for...in loop (the difference being that a for-in loop enumerates properties in the prototype chain as well).
- `Object.freeze()` is a method in JavaScript that prevents an object from being modified. It makes the object's properties and values read-only, and prevents new properties from being added, removed, or modified. Once an object is frozen, you can no longer change its properties or values, and you cannot add or remove properties from the object.