জাভাস্ক্রিপ্টে **অবজেক্ট** হলো একটি ডেটা স্ট্রাকচার যা কী-ভ্যালু পেয়ার (key-value pair) এর সংগ্রহ। এটি জাভাস্ক্রিপ্টের সবচেয়ে গুরুত্বপূর্ণ ডেটা টাইপগুলির মধ্যে একটি এবং জাভাস্ক্রিপ্টে প্রায় সবকিছুই অবজেক্ট (যেমন অ্যারে, ফাংশন, ডেট)। নিচে জাভাস্ক্রিপ্ট অবজেক্ট সম্পর্কে বিস্তারিত ব্যাখ্যা দেওয়া হলো:

**অবজেক্ট কি?**

অবজেক্ট হলো একটি সংগ্রহ যা প্রপার্টি (key-value pair) এবং মেথড (ফাংশন) ধারণ করে। প্রতিটি প্রপার্টির একটি নাম (key) এবং একটি মান (value) থাকে। উদাহরণস্বরূপ, একটি ব্যক্তির তথ্য (নাম, বয়স, ঠিকানা) একটি অবজেক্টে সংরক্ষণ করা যায়।

**অবজেক্ট তৈরি করা:**

জাভাস্ক্রিপ্টে অবজেক্ট তৈরি করার দুটি প্রধান উপায় রয়েছে:

**১. অবজেক্ট লিটারাল:**

এটি সবচেয়ে সহজ এবং সাধারণ পদ্ধতি।

javascript

Copy

let person = {

name: "John",

age: 30,

city: "Dhaka"

};

**২. new Object() সিনট্যাক্স:**

javascript

Copy

let person = new Object();

person.name = "John";

person.age = 30;

person.city = "Dhaka";

**অবজেক্টের প্রপার্টি অ্যাক্সেস করা:**

অবজেক্টের প্রপার্টি অ্যাক্সেস করার দুটি উপায় রয়েছে:

**১. ডট নোটেশন:**

javascript

Copy

console.log(person.name); // আউটপুট: John

**২. ব্র্যাকেট নোটেশন:**

javascript

Copy

console.log(person["name"]); // আউটপুট: John

**অবজেক্টের প্রপার্টি পরিবর্তন করা:**

আপনি অবজেক্টের প্রপার্টির মান পরিবর্তন করতে পারেন।

javascript

Copy

person.age = 31;

console.log(person.age); // আউটপুট: 31

**অবজেক্টে নতুন প্রপার্টি যোগ করা:**

আপনি অবজেক্টে নতুন প্রপার্টি যোগ করতে পারেন।

javascript

Copy

person.job = "Developer";

console.log(person.job); // আউটপুট: Developer

**অবজেক্ট থেকে প্রপার্টি ডিলিট করা:**

delete অপারেটর ব্যবহার করে আপনি অবজেক্ট থেকে একটি প্রপার্টি ডিলিট করতে পারেন।

javascript

Copy

delete person.city;

console.log(person.city); // আউটপুট: undefined

**অবজেক্টের মেথড:**

অবজেক্টের প্রপার্টি হিসেবে ফাংশন সংজ্ঞায়িত করা যায়। এই ফাংশনগুলিকে মেথড বলে।

javascript

Copy

let person = {

name: "John",

age: 30,

greet: function() {

return "Hello, " + this.name;

}

};

console.log(person.greet()); // আউটপুট: Hello, John

**this কিওয়ার্ড:**

অবজেক্টের মেথডের ভিতরে this কিওয়ার্ড ব্যবহার করে অবজেক্টের প্রপার্টি অ্যাক্সেস করা যায়। this অবজেক্টটিকে নির্দেশ করে যার মধ্যে মেথডটি ডিফাইন করা হয়েছে।

javascript

Copy

let person = {

name: "John",

greet: function() {

return "Hello, " + this.name;

}

};

console.log(person.greet()); // আউটপুট: Hello, John

**অবজেক্টের প্রোটোটাইপ:**

জাভাস্ক্রিপ্টে প্রতিটি অবজেক্ট একটি প্রোটোটাইপ অবজেক্ট থেকে প্রোপার্টি এবং মেথড ইনহেরিট করে। আপনি প্রোটোটাইপ ব্যবহার করে নতুন মেথড বা প্রপার্টি যোগ করতে পারেন।

javascript

Copy

function Person(name, age) {

this.name = name;

this.age = age;

}

Person.prototype.greet = function() {

return "Hello, " + this.name;

};

let john = new Person("John", 30);

console.log(john.greet()); // আউটপুট: Hello, John

**অবজেক্টের ইটারেশন:**

আপনি for...in লুপ ব্যবহার করে অবজেক্টের প্রপার্টি গুলি ইটারেট করতে পারেন।

javascript

Copy

for (let key in person) {

console.log(key + ": " + person[key]);

}

**অবজেক্টের স্ট্যাটিক মেথড:**

জাভাস্ক্রিপ্টে কিছু বিল্ট-ইন মেথড রয়েছে যা অবজেক্টের উপর কাজ করে। যেমন:

* Object.keys(): অবজেক্টের কী গুলি অ্যারে হিসেবে রিটার্ন করে।
* Object.values(): অবজেক্টের ভ্যালু গুলি অ্যারে হিসেবে রিটার্ন করে।
* Object.entries(): অবজেক্টের কী-ভ্যালু পেয়ার গুলি অ্যারে হিসেবে রিটার্ন করে।

**উদাহরণ:**

javascript

Copy

let person = {

name: "John",

age: 30

};

console.log(Object.keys(person)); // আউটপুট: ["name", "age"]

console.log(Object.values(person)); // আউটপুট: ["John", 30]

console.log(Object.entries(person)); // আউটপুট: [["name", "John"], ["age", 30]]

**সারসংক্ষেপ:**

| **বৈশিষ্ট্য** | **বর্ণনা** |
| --- | --- |
| **অবজেক্ট লিটারাল** | let obj = { key: value }; |
| **প্রপার্টি অ্যাক্সেস** | obj.key বা obj["key"] |
| **প্রপার্টি পরিবর্তন** | obj.key = newValue; |
| **নতুন প্রপার্টি যোগ** | obj.newKey = value; |
| **প্রপার্টি ডিলিট** | delete obj.key; |
| **মেথড** | obj.method = function() { ... }; |
| **this কিওয়ার্ড** | অবজেক্টের প্রপার্টি অ্যাক্সেস করতে ব্যবহৃত |
| **প্রোটোটাইপ** | Object.prototype |
| **ইটারেশন** | for (let key in obj) { ... } |
| **স্ট্যাটিক মেথড** | Object.keys(), Object.values(), Object.entries() |

অবজেক্ট ব্যবহার করে আপনি জাভাস্ক্রিপ্টে জটিল ডেটা স্ট্রাকচার তৈরি করতে পারেন এবং ডেটা ম্যানিপুলেট করতে পারেন। 😊

Insert the new method within Object

জাভাস্ক্রিপ্টে একটি অবজেক্টের মধ্যে নতুন মেথড (ফাংশন) যোগ করা খুবই সহজ। আপনি অবজেক্টের একটি প্রপার্টি হিসেবে ফাংশন অ্যাসাইন করে নতুন মেথড যোগ করতে পারেন। নিচে বিভিন্ন উপায়ে অবজেক্টে নতুন মেথড যোগ করার পদ্ধতি বাংলায় ব্যাখ্যা করা হলো:

### ****১. অবজেক্ট লিটারালে মেথড যোগ করা****:

যখন আপনি অবজেক্ট লিটারাল ব্যবহার করে অবজেক্ট তৈরি করেন, তখন সরাসরি মেথড যোগ করতে পারেন।

#### উদাহরণ:

javascript

Copy

let person = {

name: "John",

age: 30,

greet: function() {

return "Hello, " + this.name;

}

};

console.log(person.greet()); // আউটপুট: Hello, John

### ****২. ডট নোটেশন ব্যবহার করে মেথড যোগ করা****:

আপনি ডট নোটেশন ব্যবহার করে অবজেক্টে নতুন মেথড যোগ করতে পারেন।

#### উদাহরণ:

javascript

Copy

let person = {

name: "John",

age: 30

};

// নতুন মেথড যোগ

person.greet = function() {

return "Hello, " + this.name;

};

console.log(person.greet()); // আউটপুট: Hello, John

### ****৩. ব্র্যাকেট নোটেশন ব্যবহার করে মেথড যোগ করা****:

আপনি ব্র্যাকেট নোটেশন ব্যবহার করে অবজেক্টে নতুন মেথড যোগ করতে পারেন। এটি বিশেষভাবে উপকারী যখন মেথডের নাম ডাইনামিক হয়।

#### উদাহরণ:

javascript

Copy

let person = {

name: "John",

age: 30

};

// নতুন মেথড যোগ

person["greet"] = function() {

return "Hello, " + this.name;

};

console.log(person.greet()); // আউটপুট: Hello, John

### ****৪. অবজেক্টের প্রোটোটাইপে মেথড যোগ করা****:

যদি আপনি একটি কনস্ট্রাক্টর ফাংশন ব্যবহার করে অবজেক্ট তৈরি করেন, তাহলে প্রোটোটাইপে মেথড যোগ করতে পারেন। এটি সব ইনস্ট্যান্সে মেথডটি শেয়ার করে।

#### উদাহরণ:

javascript

Copy

function Person(name, age) {

this.name = name;

this.age = age;

}

// প্রোটোটাইপে নতুন মেথড যোগ

Person.prototype.greet = function() {

return "Hello, " + this.name;

};

let john = new Person("John", 30);

console.log(john.greet()); // আউটপুট: Hello, John

### ****৫. ES6 মেথড শর্টহ্যান্ড****:

ES6 এ আপনি অবজেক্ট লিটারালে মেথড ডিফাইন করার জন্য সংক্ষিপ্ত সিনট্যাক্স ব্যবহার করতে পারেন।

#### উদাহরণ:

javascript

Copy

let person = {

name: "John",

age: 30,

greet() {

return "Hello, " + this.name;

}

};

console.log(person.greet()); // আউটপুট: Hello, John

### ****৬.****Object.assign()****ব্যবহার করে মেথড যোগ করা****:

আপনি Object.assign() মেথড ব্যবহার করে একটি অবজেক্টে নতুন মেথড যোগ করতে পারেন।

#### উদাহরণ:

javascript

Copy

let person = {

name: "John",

age: 30

};

// নতুন মেথড যোগ

Object.assign(person, {

greet: function() {

return "Hello, " + this.name;

}

});

console.log(person.greet()); // আউটপুট: Hello, John

### ****সারসংক্ষেপ****:

| **পদ্ধতি** | **বর্ণনা** | **উদাহরণ** |
| --- | --- | --- |
| **অবজেক্ট লিটারাল** | অবজেক্ট লিটারালে সরাসরি মেথড যোগ করা | greet: function() { ... } |
| **ডট নোটেশন** | ডট নোটেশন ব্যবহার করে মেথড যোগ করা | person.greet = function() { ... } |
| **ব্র্যাকেট নোটেশন** | ব্র্যাকেট নোটেশন ব্যবহার করে মেথড যোগ করা | person["greet"] = function() { ... } |
| **প্রোটোটাইপ** | প্রোটোটাইপে মেথড যোগ করা | Person.prototype.greet = function() { ... } |
| **ES6 মেথড শর্টহ্যান্ড** | ES6 সংক্ষিপ্ত সিনট্যাক্স ব্যবহার করা | greet() { ... } |
| **Object.assign()** | Object.assign() ব্যবহার করে মেথড যোগ করা | Object.assign(person, { greet: function() { ... } }) |

এই পদ্ধতিগুলি ব্যবহার করে আপনি জাভাস্ক্রিপ্ট অবজেক্টে নতুন মেথড যোগ করতে পারেন এবং আপনার কোডকে আরও মডুলার ও পুনরায় ব্যবহারযোগ্য করতে পারেন। 😊