JavaScript: this, Constructor, Prototype, Inheritance

1. this Keyword

The 'this' keyword refers to the object it belongs to.

- In a method: refers to the owner object.

- Alone: refers to the global object (window in browser).

- In strict mode: undefined.

- In an event: refers to the element that received the event.

Example of 'this'

```
const person = {
  name: "Shekhar",
  greet: function() {
    console.log("Hello, " + this.name);
  }
};
person.greet(); // Hello, Shekhar
```

2. Constructor Function

Constructors are special functions used to create and initialize objects.

Constructor Example

```
function Person(name, age) {
  this.name = name;
  this.age = age;
}
const user = new Person("Taruna", 44);
console.log(user.name); // Taruna
```

3. Prototype in JavaScript

Every JavaScript function has a prototype property.

You can add properties and methods to objects via prototype.

Prototype Example

```
Person.prototype.greet = function() {
  return "Hi, I am " + this.name;
};
console.log(user.greet()); // Hi, I am Taruna
```

4. Inheritance using Prototype

Inheritance lets one object access the properties and methods of another using prototypes.

Inheritance Example

```
function Employee(name, id) {
  this.name = name;
```

```
  this.id = id;
}
Employee.prototype.getDetails = function() {
  return this.name + " (" + this.id + ")";
};
function Manager(name, id, dept) {
  Employee.call(this, name, id);
  this.department = dept;
}
Manager.prototype = Object.create(Employee.prototype);
const m1 = new Manager("Shekhar", 101, "IT");
console.log(m1.getDetails()); // Shekhar (101)
```