Phương thức tĩnh trong JavaScript

Trong JavaScript, **phương thức tĩnh** của một object hoặc class là các phương thức không gắn liền với một instance cụ thể của object hoặc class đó. Thay vào đó, chúng được gọi trực tiếp từ chính object hoặc class mà không cần tạo một instance.

---

1. **Phương thức tĩnh trên object**

Phương thức tĩnh được định nghĩa trực tiếp trong object mà không liên quan đến các thuộc tính hoặc phương thức instance của object đó.

Ví dụ:

```
const Calculator = {
  // Phương thức tĩnh
  add(a, b) {
    return a + b;
  },
  subtract(a, b) {
    return a - b;
  }
};

// Gọi phương thức tĩnh trực tiếp từ object
console.log(Calculator.add(5, 3));       // 8
console.log(Calculator.subtract(5, 3)); // 2
```

- **Điểm đặc biệt:** Các phương thức này không sử dụng từ khóa `this` để tham chiếu đến instance vì chúng không liên quan đến bất kỳ instance nào.

---

2. **Phương thức tĩnh trên class**

Phương thức tĩnh cũng được sử dụng trong các class, nhưng được định nghĩa bằng từ khóa `static`. Các phương thức này chỉ có thể được gọi trực tiếp từ class, không phải từ instance của class.

Ví dụ:

```
class MathUtils {
  // Định nghĩa phương thức tĩnh
  static multiply(a, b) {
    return a * b;
  }

  static divide(a, b) {
    if (b === 0) {
```

```
      throw new Error("Division by zero is not allowed");
    }
    return a / b;
  }
}

// Gọi phương thức tĩnh trực tiếp từ class
console.log(MathUtils.multiply(4, 3)); // 12
console.log(MathUtils.divide(10, 2)); // 5

// Không thể gọi phương thức tĩnh từ instance
const utils = new MathUtils();
try {
  console.log(utils.multiply(4, 3)); // Lỗi: utils.multiply is not a
function
} catch (error) {
  console.log(error.message);
}
```

---

3. **Khi nào sử dụng phương thức tĩnh?**

Phương thức tĩnh thường được sử dụng trong các trường hợp sau:

- Khi logic hoặc chức năng không phụ thuộc vào instance cụ thể của object hoặc class.
- Để nhóm các hàm tiện ích (utility functions) vào cùng một object hoặc class.
- Để tạo các hàm khởi tạo hoặc factory pattern.

Ví dụ ứng dụng thực tế:

- **Các tiện ích tính toán:**

```
class TemperatureConverter {
  static toCelsius(fahrenheit) {
    return (fahrenheit - 32) * 5 / 9;
  }

  static toFahrenheit(celsius) {
    return (celsius * 9 / 5) + 32;
  }
}

console.log(TemperatureConverter.toCelsius(98.6)); // 37
console.log(TemperatureConverter.toFahrenheit(37)); // 98.6
```

- **Factory pattern:**

```
class User {
  constructor(name, role) {
```

```
    this.name = name;
    this.role = role;
  }

  // Phương thức tĩnh để tạo người dùng mặc định
  static createAdmin(name) {
    return new User(name, "admin");
  }
}

const admin = User.createAdmin("Alice");
console.log(admin); // User { name: 'Alice', role: 'admin' }
```

---

4. So sánh phương thức tĩnh và phương thức instance

Đặc điểm	Phương thức tĩnh	Phương thức instance
Liên kết với	Class hoặc object	Instance của class hoặc object
Cách gọi	Gọi trực tiếp từ class hoặc object	Gọi từ instance cụ thể
Sử dụng từ khóa `this`	Không, trừ khi liên quan đến class	Có, để tham chiếu đến instance
Ứng dụng	Hàm tiện ích, factory method	Tương tác và làm việc với dữ liệu của instance

---

5. Tóm lại

- **Phương thức tĩnh** được định nghĩa trực tiếp trên object hoặc class và được gọi mà không cần tạo instance.
- Chúng rất hữu ích cho các chức năng không phụ thuộc vào instance, như các hàm tiện ích hoặc factory pattern.
- Phân biệt phương thức tĩnh với phương thức instance giúp tổ chức mã rõ ràng và dễ bảo trì hơn.