Bài 3
Lớp và đối tượng

Module: Advanced Programming with JavaScript

Mục tiêu

- Trình bày được mô hình lập trình hướng đối tượng
- Trình bày được các khái niệm lớp, đối tượng, phương thức, thuộc tính, hàm tạo (constructor)
- Trình bày được cú pháp khai báo lớp
- Trình bày được cú pháp khởi tạo đối tượng
- Trình bày được cách truy xuất thuộc tính, phương thức của lớp
- Tạo và sử dụng được các đối tượng đơn giản
- Mô tả được lớp bằng biểu đồ (Class Diagram)
- Sử dụng được Getter và Setter
- Sử dụng được từ khóa constructor
- Sử dụng được từ khóa this
- Trình bày được cơ chế Prototype trong JavaScript

Thảo luận

ECMAScript là gì?

Ý nghĩa của ECMAScript trong công nghệ JavaScript

Tổng quan về OOP

- Đối tượng có vai trò quan trọng trong ngôn ngữ lập trình
- Nhiều nhà phát triển coi cách tiếp cận của JavaScript để quản lý đối tượng là đơn giản so với cách quản lý đối tượng của Java, C # và các ngôn ngữ OOP khác.
- Nhiều nhà phát triển không coi JavaScript là một ngôn ngữ hướng đối tượng thực sự, mà chỉ là một ngôn ngữ có cách sử dụng đối tượng kỳ lạ.
- Phân loại một ngôn ngữ lập trình Hướng đối tượng dựa trên hai yêu cầu:
    - Khả năng mô hình hóa một vấn đề thông qua các đối tượng
    - Sự hỗ trợ của một số nguyên tắc cho phép sử dụng lại mã và mô-đun.

Mối quan hệ giữa các đối tượng

- Liên kết : Đây là khả năng của đối tượng để tham chiếu đến một đối tượng độc lập khác
- Tổng hợp : Đây là khả năng của đối tượng để nhúng một hoặc nhiều đối tượng độc lập
- Thành phần : Đây là khả năng của đối tượng để nhúng một hoặc nhiều đối tượng phụ thuộc

Tái sử dụng

- Tính đóng gói : Đây là khả năng tập trung vào một dữ liệu thực thể duy nhất và mã điều khiển nó, che giấu các chi tiết bên trong của nó
- Kế thừa : Đây là cơ chế mà một đối tượng có được một số hoặc tất cả các tính năng từ một hoặc nhiều đối tượng khác
- Tính đa hình : Đây là khả năng xử lý các đối tượng khác nhau dựa trên kiểu dữ liệu hoặc cấu trúc của chúng

JavaScript OOP và classical OOP

- JavaScript là ngôn ngữ kiểu động (dynamic)
- Ngay từ ban đầu, JavaScript không có khái niệm lớp (class) và không có từ khóa này.
  - Lưu ý: Phiên bản ES6 đã bổ sung từ khóa class để định nghĩa lớp
- Các lớp trong hầu hết các ngôn ngữ Hướng đối tượng đại diện cho **sự tổng quát hóa** của các đối tượng (nghĩa là, một mức độ trừu tượng bổ sung của các đối tượng).

Mô hình dựa trên đối tượng đơn giản

JavaScript được thiết kế trên một mô hình dựa trên đối tượng đơn giản:

- Một đối tượng là một tập hợp các thuộc tính và một thuộc tính là một liên kết giữa tên (hoặc khóa) và một giá trị.
- Giá trị của một thuộc tính có thể là một hàm. Trong trường hợp này, thuộc tính được gọi là một phương thức.
- Ngoài các đối tượng được định nghĩa sẵn trong trình duyệt, chúng ta có thể định nghĩa các đối tượng tùy biến.

Khởi tạo đối tượng

Có 3 cách khởi tạo đối tượng trong JavaScript:

- Sử dụng cú pháp khởi tạo
- Khởi tạo đối tượng bằng function
- Khởi tạo đối tượng từ class (ES6)

Demo

Khởi tạo đối tượng

Làm việc với đối tượng

Trong JavaScript, một đối tượng là một thực thể độc lập chứa các thuộc tính và kiểu.

- Truy cập các thuộc tính của một đối tượng bằng một ký hiệu dấu chấm
- Tên đối tượng và tên thuộc tính đều phân biệt chữ hoa chữ thường
- Sử dụng cú pháp dấu ngoặc nhọn ({}) để khởi tạo đối tượng
- Các thuộc tính chưa được gán của một đối tượng là undefined
- Tên thuộc tính đối tượng có thể là bất kỳ chuỗi hợp lệ nào

Demo

Làm việc với đối tượng

Prototype trong JavaScript

- Prototype là cơ chế giúp các đối tượng kế thừa các tính năng từ một đối tượng khác.
- JavaScript là một ngôn ngữ lập trình dựa trên prototype
- Prototype là khái niệm cốt lõi trong JavaScript.

Lưu ý:

- Trong JavaScript, một hàm (function) cũng là một đối tượng.
- Tất cả các đối tượng con tạo ra bởi hàm khởi tạo đều mang các giá trị trong thuộc tính prototype của hàm.
- Thuộc tính (prototype attribute) có giá trị trỏ tới đối tượng prototype mà nó kế thừa thuộc tính

Demo

Cách tạo prototype

Chuỗi prototype (prototype chain)

- Chuỗi prototype (prototype chain) là quá trình lặp lại khi tìm giá trị thuộc tính bên trong chính đối tượng.
  - Khi truy cập vào một thuộc tính của một đối tượng, JavaScript sẽ tìm thuộc tính đó bên trong chính đối tượng.
  - Nếu không tìm thấy, nó sẽ tiếp tục tìm lên trên Prototype của đối tượng và cứ tiếp tục như thế cho đến khi gặp Object.prototype thì dừng và cho ra kết quả.
  - Nếu vẫn không tìm thấy tại Object.prototype, kết quả sẽ là undefined.
- Prototype giúp chúng ta truy cập tới các thuộc tính và phương thức của đối tượng.
- Chuỗi prototype và thuộc tính prototype của hàm tạo nên cơ chế kế thừa prototype-based cho JavaScript.

Demo

Chuỗi prototype

Định nghĩa lớp trong ES6

Cú pháp khai báo một lớp:

```
class Class_name {}
```

Định nghĩa lớp có thể bao gồm những thành phần sau:

- Constructors (hàm tạo) - Chịu trách nhiệm cấp phát bộ nhớ cho các đối tượng của lớp.
- Functions (các hàm) - Các hàm thể hiện các hành động mà một đối tượng có thể thực hiện. Đôi khi chúng cũng được gọi là phương thức.

Các thành phần này kết hợp với nhau được gọi là thành viên dữ liệu của lớp.

Phương thức khởi tạo

Cú pháp:

```
class myClass {
  constructor(value1, value2) {
    this.property1 = value1;
    this.property2 = value2;
    this.property3 = "";
  }
}
```

- Phương thức khởi tạo của lớp là một phương thức có tên riêng là **constructor**.
- Không thể có nhiều hơn một phương thức khởi tạo trong một lớp.

Demo

Phương thức khởi tạo

Tóm tắt

Qua bài học này, chúng ta đã tìm hiểu:

- Mô hình lập trình hướng đối tượng
- Khái niệm lớp, đối tượng, phương thức, thuộc tính, hàm tạo (constructor)
- Cú pháp khai báo lớp
- Cú pháp khởi tạo đối tượng
- Cách truy xuất thuộc tính, phương thức của lớp
- Tạo và sử dụng được các đối tượng đơn giản
- Sử dụng được từ khóa constructor
- Sử dụng được từ khóa this
- Cơ chế Prototype trong JavaScript