LẬP TRÌNH WINDOWS

BÀI 9: DELEGATE VÀ SỰ KIỆN

Giảng viên: Lý Anh Tuấn

Email: tuanla@wru.vn

Delegate là gì?

- Delegate là kiểu dữ liệu đặc biệt, là biến kiểu tham chiếu, có khả năng lưu trữ một tham chiếu tới phương thức
- Delegate là một cơ chế hỗ trợ chung cho việc gọi phương thức gián tiếp trong khi chạy => delegate được hiểu là Ủy quyền

Khai báo delegate

- Khai báo delegate trong C# quyết định các phương thức có thể được tham chiếu bởi delegate đó
- Một delegate có thể tham chiếu tới một phương thức có cùng chữ ký với delegate đó

Ví dụ

- C# có sẵn một kiểu delegate có dạng sau:

```
public delegate void EventHandler(object sender, EventArgs e);
```

- Kiểu này được dùng để tham chiếu tới các phương thức có 2 tham số là object và EventArgs
- Sự kiện click của một nút là một thể hiện của kiểu delegate đó (delegate có tên EventHandler)

```
public event EventHandler Click;
```

Ví dụ

- Do vậy, khi tạo một sự kiện click cho một nút (VD nút btSo1) ta sẽ thấy sự kiện này tham chiếu tới một hàm có 2 tham số object và EvenArgs

```
this.btSo1.Click += new System.EventHandler(this.btSo1_Click);

private void btSo1_Click(object sender, EventArgs e)
{
}
```

Ứng dụng của delegate

- Trong lập trình C#, delegate được sử dụng chính trong thực thi sự kiện (event) và các phương thức gọi sau (call-back methods)
- Để thực thi delegate trong ứng dụng cần:
  - Khai báo delegates (khai báo kiểu, khai báo biến)
  - Tạo thể hiện delegates (cho biến delegate tham chiếu tới phương thức)
  - Sử dụng delegates

Khai báo delegate

- Khai báo kiểu delegate
  - Cú pháp:

delegate <kiểu_trả_về> <tên_delegate>(<tham_số>);

VD: delegate void Thongbao(string str); // tương tự khai báo
// phương thức sử dụng
// từ khoá delegate

- Khai báo biến delegate

VD: Thongbao thongbao1;

Tạo thể hiện cho deletage

- Khi một kiểu delegate được khai báo, một đối tượng delegate được tạo với từ khóa **new** và được liên kết với một phương thức cụ thể
- Khi tạo đối tượng delegate, tham số được truyền tới biểu thức **new** được viết tương tự như một lời gọi phương thức nhưng không có tham số kèm theo

Tạo thể hiện cho delegate

- Cú pháp:

  new <kiểu_delegate>(<đối tượng>.<phương_thức>);

  - VD: new Thongbao(myObj.SayHello);
  - Đối tượng có thể là *this* (và có thể bỏ qua)

    VD: new Thongbao(SayHello);
  - Phương thức có thể là *static*. Trong trường hợp này, tên của class phải được thay thế cho đối tượng.

    VD: new Thongbao (MyClass.StaticSayHello);

Tạo thể hiện cho delegate

- Yêu cầu: Chữ ký của phương thức phải trùng với chữ ký của <kiểu_delegate>
  - số lượng tham số
  - kiểu dữ liệu của tham số (bao gồm cả kiểu trả về)
  - kiểu truyền tham số (ref, out, value)

Tạo thể hiện cho delegate

- Tạo phương thức sẽ gán cho biến delegate

```
void SayHello(string str) //phương thức này phải có cùng
        //kiểu trả về và cùng tham số với delegate sẽ dùng nó
{
        Console.WriteLine("Hello from " + str);
}
```

- Tạo thể hiện cho biến delegate

```
thongbao1 = new Thongbao(SayHello);
```

Sử dụng delegate

- Sử dụng delegate tương tự như gọi một phương thức: gồm tên biến delegate kèm theo các đối số (nếu có)

Ví dụ:

```
thongbao1("John");
// gọi phương thức SayHello("John")
// => "Hello from John"
```

Sử dụng delegate

- Tất cả các phương thức phù hợp với delegate đều có thể được gán với biến delegate đó

VD:

```
void SayGoodBye(string str) {
    Console.WriteLine("Good bye from " + str);
}
thongbao1 = new Thongbao(SayGoodBye);
thongbao1("John");
// gọi SayGoodBye("John") => "Good bye from John"
```

Sử dụng delegate

- Lưu ý:
  - Biến delegate có thể được gán giá trị *null* (không có phương thức nào được gán cho nó).
  - Nếu biến delegate bằng *null* thì sẽ không được gọi

Các kiểu delegate

- Có hai kiểu delegate và tùy thuộc vào yêu cầu của ứng dụng mà lựa chọn kiểu delegate phù hợp
  - Single-cast delegate
  - Multicast delegate

Single-cast delegate

- Một single-cast delegate dẫn xuất từ lớp System.Delegate
- Nó chứa tham chiếu tới chỉ một phương thức tại một thời điểm

Multicast delegate

- Một multicast delegate dẫn xuất từ lớp System.MulticastDelegate
- Nó chứa lời gọi tới một danh sách phương thức
- Kiểu trả về của tất cả delegate này phải là giống nhau
- Khi một multicast delegate được gọi, nó sẽ xử lý tất cả phương thức theo thứ tự mà nó đã gán

Multicast delegate

- Thêm phương thức vào multicast delegate bằng cách sử dụng toán tử **+**
- Loại bỏ phương thức khỏi multicast delegate bằng cách sử dụng toán tử **-**

Multicast delegate: Ví dụ

- Biến multicast delegate có thể chứa nhiều giá trị cùng một thời điểm

```
Thongbao thongbao1;
thongbao1 = new Thongbao(SayHello);
thongbao1 += new Thongbao(SayGoodBye);
thongbao1("John");
// "Hello from John"   "Good bye from John"
```

Multicast delegate: Ví dụ

- Biến multicast delegate có thể chứa nhiều giá trị cùng một thời điểm

```
Thongbao thongbao1;
thongbao1 = new Thongbao(SayHello);
thongbao1("John");   // "Hello from John"
thongbao1 += new Thongbao(SayGoodBye);
thongbao1 -= new Thongbao(SayHello);
thongbao1("John");   // "Good bye from John"
```

Multicast delegate: Lưu ý

- Nếu multicast delegate là một hàm (phương thức có kiểu trả về khác void), lời gọi sẽ trả về giá trị của hàm được tham chiếu cuối cùng
- Vì khi biến multicast delegate là 1 biến tham chiếu đến nhiều hàm. Khi gọi biến multicast delegate thực hiện thì nó sẽ lần lượt thực thi các hàm mà nó tham chiếu đến => kết quả sẽ là kết quả của hàm cuối cùng

Multicast delegate: Lưu ý

- Nếu multicast delegate có tham số ref thì tham số đó sẽ được truyền qua tất cả các phương thức
- Nếu multicast delegate có tham số out, tham số của lời gọi cuối cùng sẽ được trả về

Bài tập

- Tạo 4 hàm cộng trừ nhân chia:
  - Tham số truyền vào là 2 số nguyên
  - Trả ra kết quả tương ứng với 2 số tham số truyền vào
- Tạo delegate
- Thực hiện tham chiếu đến 4 hàm trên
- Gọi và xem kết quả

Bài tập 2

- Tạo một lớp Nguoi có các thông tin:
  - Tên, tuổi, chiều cao, cân nặng, ….
  - Các phương thức so sánh tuổi, chiều cao, cân nặng giữa 2 người
- Thực hiện nhập thông tin 2 người
- Sử dụng delegate để khi gọi so sánh sẽ đưa ra kết quả của cả 3 phương thức so sánh tuổi, chiều cao, cân nặng của 2 người vừa nhập

SỰ KIỆN (EVENT)

Bắt sự kiện trên Winform

- Sự kiện là các tác động lên đối tượng trên form.
- Với mỗi tác động người dùng mong muốn thực hiện một nhiệm vụ cụ thể nào đó
- C# đã thiết kế sẵn cho mỗi sự kiện một kiểu delegate tương ứng
- Lập trình viên chỉ cần xử lý các nhiệm vụ bên trong hàm mà delegate của sự kiện đó gọi đến

Bắt sự kiện trên Winform

- Các delegate dùng để bắt sự kiện thường có dạng như sau:

  delegate void SomeEvent (object sender, SomeEventArgs e);

- Trong đó
  - Kiểu trả về: void
  - Tham số thứ nhất: Đối tượng gửi sự kiện (kiểu *object*)
  - Tham số thứ hai: sự kiện (một đối tượng của lớp *EventArgs* tương ứng)

Bắt sự kiện trên Winform

- Ví dụ: Tạo trình xử lý sự kiện ở thời điểm chạy
  - Bắt sự kiện click cho nút button1 thì cần khai báo:

```
button1.Click += new EventHandler(button1_Click);
```

  - Và định nghĩa hàm:

```
private void button1_Click(object sender, EventArgs e)
{
    // Thêm mã xử lý sự kiện ở đây
}
```

Bài tập

- Tạo 3 lớp: nhà, xe và người. Trong đó mỗi lớp được mô tả với ít nhất các thành phần như sau:
  - Lớp nhà: có thông tin về diện tích sàn, có phương thức so sánh 2 nhà và xuất ra nhà có diện tích lớn hơn
  - Lớp xe: có thông tin về giá xe, có phương thức so sánh 2 xe và xuất ra xe có giá cao hơn
  - Lớp người: có thông tin về chiều cao, có phương thức so sánh 2 người và xuất ra người cao hơn
- Tạo một form có menu cho phép chọn đối tượng làm việc:
  - Nếu chọn nhà thì hiển thị giao diện nhập thông tin 2 nhà
  - Nếu chọn xe thì hiển thị giao diện nhập thông tin 2 xe
  - Nếu chọn người thì hiển thị giao diện nhập thông tin 2 người
- Trên giao diện có một nút so sánh. Thực hiện so sánh 2 đối tượng đang làm việc trên giao diện bằng cách dùng delegate