**DELEGATE & EVENT**

# Delegate

## Delegate là gì?

Delegate là một kiểu dùng để mô tả phương thức, bao gồm có kiểu trả về và các tham số parametter của phương thức, nó có khả năng gọi đến phương thức có kiểu trả về và tham số parametter giống với nó.

Ví dụ trường hợp WinForm, nút Button là một class đã xây dựng sẵn, còn phương thức xư lý sự kiện bấm nút On Click là do người lập trình viết. Để gọi một phương thức khi phương thức đó chưa tồn tại hoặc chưa xác định thì làm sao xây dựng được class, cho nên sử dụng delegate trong trường hợp này.

A picture containing diagram

Description automatically generated

Syntax: [access modifier] delegate [return type] [delegate name]([parameters])

*Text

Description automatically generated*

Bản thân delegate chỉ là tham chiếu, được sử dụng để đại diện cho phương thức nào đó, chẳng hạn trong ví dụ trên thì delegate đại diện cho phương thức *sum* và *sub* hoặc bất kỳ phương thức nào có dạng 2 tham số là *int* và kiểu trả về là *int*

## Vai trò của delegate

Vai trò của delegate được sử dụng để giúp một class object tương tác ngươc trở lại với thực thể tạo ra và sử dụng class đó. Do có khả năng call back nên delegate được áp dụng trong lập trình hướng sự kiện (winforms và WPF). Delegate cũng được sử dụng phổ biến với mô hình lập trình bất đồng bộ ở dạng phương thức call – back hoặc lập trình đa luồng.

## Generic delegate

C# hỗ trợ người lập trình bằng generic delegate mà có thể trực tiếp sử dụng ngay để khai báo biến. Sử dụng generic delegate giúp bỏ qua giai đoạn khai báo kiểu delegate. Có 3 nhóm generic delegate: Actions, Funcs, Predicates.

### Actions

Actions là kiểu delegate tương ứng với các phương thức không trả về kiểu dữ liệu (đầu ra là void). Có thể truyền vào 16 tham số đầu vào cho kiểu Action.

Ex:

Text

Description automatically generated

### Funcs

Funcs là kiểu delegate tương ứng với các phương thức có trả về dữ liệu.

Ex:

Text

Description automatically generated

Định nghĩa kiểu funcs khác với actions ở chỗ, funcs luôn phải có điều kiện đặt ra ở vị trí cuối cùng trong danh sách kiểu generic (out Tresult)

### Predicate

Predicate là kiểu delegate tương ứng với các phương thức có 1 tham số đầu vào và trả về giá trị bool. Predicate được sử dụng trong các biểu thức so sánh.

Ex:

Predicate<int> predicate1 = (i) => i > 100;

Console.WriteLine($"is 10 > 100? it's {predicate1(10)}");

Predicate<string> predicate2 = (s) => s.Length > 10;

Biến predicate1 tham chiếu tới một hàm lambda thực hiện so sánh xem một số nguyên co lớ hơn 100 hay không, biến predicate2 tham chiếu tới một hàm lambda nhận một chuỗi ký tự và so sánh xem độ dài chuỗi đó có lớn hơn 10 không.

# event

## Event là gì?

Trong C#, Event là một đối tượng đặc biệt của Delegate, nó là nơi chứa các phương thức, các phương thức này sẽ được thực thi đồng loạt khi sự kiện xẩy ra. Bạn có thể thêm các phương thức sẽ được thực thi vào đối tượng Event của đối tượng phát ra sự kiện.

Ví dụ lớp Button mô phỏng một nút, nó định nghĩa ra một Event (sự kiện) để thông báo rằng nó vừa bị nhấn (Click). Khi button bị nhấn, event sẽ được thực thi những phương thức.

Một sự kiện được khai báo theo hai bước:

* Khai báo một delegate
* Khai báo biến của delegate với từ khó event

Syntax: pulblic delegate void Notify(); //delegate

public class Processing

{

Public event Notify ProcessCompleted; //event

}

Text

Description automatically generated

Kết quả in ra màn hình console:

Text

Description automatically generated

## Kích hoạt sự kiện

Sử dụng hàm .Invoke();

Có thể sử dụng thêm phép toán kiểm tra null “?” để kích hoạt event an toàn. Lý do là nếu không có subscriber nào, biến event sẽ nhận giá trị null. Mọi lệnh từ biến null sẽ đều dẫn đến NullException.

## Đăng ký/ Hủy đăng ký theo dõi event

Sự khác biệt của event so với delegate là ở cách sử dụng trong client code. Đối với event bạn chỉ có thể dùng phép toán += (đăng ký theo dõi) hoặc -= (hủy đăng ký), không thể sử dụng phép gán =.

Sở dĩ chỉ sử dụng được phép toán += hoặc -= là để đảm bảo rằng các handler (subscriber) không ảnh hưởng (ghi đè, xóa lãn nhau) đến nhau. Đây là cách event hạn chế tác dụng của delegate nhằm đảm bảo thực thi mô hình pub/sub.