

Bài: Delegate trong C#

Xem bài học trên website để ủng hộ Kteam: [Delegate trong C#](#)

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage [How Kteam](#) nhé!

Dẫn nhập

Ở bài học trước, chúng ta đã cùng nhau tìm hiểu về [LIST TRONG C#](#). Hôm nay chúng ta sẽ cùng tìm hiểu về **Delegate trong C#**.

Nội dung

Để đọc hiểu bài này tốt nhất các bạn nên có kiến thức cơ bản về các phần:

- [BIẾN](#) và [KIỂU DỮ LIỆU, TOÁN TỬ](#) trong C#
- [CÂU ĐIỀU KIỆN](#) trong C#
- Cấu trúc cơ bản của [VÒNG LẶP](#), [HÀM](#), [MẢNG](#) trong C#
- [LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG](#) trong C#

Trong bài học này, chúng ta sẽ cùng tìm hiểu các vấn đề:

- Delegate là gì?
- Khai báo Delegate trong C#
- Khởi tạo Delegate trong C#
- Multicast(đa hướng) một Delegate trong C#
- Cách dùng Delegate trong C#
- Dùng Delegate như một call-back function

Delegate là gì?

Delegate trong C# tương tự như con trỏ hàm trong C hoặc [C++](#).

Delegate là một **biến kiểu tham chiếu**(references) chứa tham chiếu **tới một phương thức**.

Tham chiếu của **Delegate** có thể **thay đổi runtime** (khi chương trình đang thực thi).

Delegate thường được dùng để **triển khai các phương thức hoặc sự kiện call-back**.

Bạn cứ hiểu **Delegate** là một **biến bình thường**, biến này chứa hàm mà bạn cần gọi. Sau này lỗi ra sai như hàm bình thường. Giá trị của biến **Delegate** lúc này là tham chiếu đến hàm. Có thể thay đổi runtime khi chương trình đang chạy.

Delegate được dẫn xuất từ lớp [System.Delegate](#) trong C#.

Khai báo Delegate trong C#

Khai báo **Delegate** trong C# sẽ tương tự như khai báo một biến. Nhưng cần thêm từ khóa **Delegate** để xác định đây là một Delegate. Đồng thời vì **Delegate** là để tham chiếu đến một hàm, nên cũng cần khai báo kèm **kiểu dữ liệu** trả về của và **tham số đầu vào** của **Delegate** tương ứng với hàm tham chiếu.

Công thức:

```
delegate <kiểu trả về> <tên delegate> (<danh sách tham số nếu có>);
```

Ví dụ:**C#:**

```
delegate int MyDelegate(string s);
```

Lưu ý: Chữ delegate viết thường

Lúc này chúng ta đã tạo một **Delegate** có tên là **MyDelegate**. **MyDelegate** có kiểu trả về là **int**, một tham số đầu vào là **string**.

MyDelegate lúc này có thể dùng làm kiểu dữ liệu cho mọi **Delegate** tới hàm tương ứng kiểu trả về và tham số đầu vào.

Khởi tạo và sử dụng Delegate trong C#

Khi kiểu **Delegate** được khai báo, đối tượng **Delegate** phải được tạo với từ khóa **new** và được **tham chiếu đến một phương thức cụ thể**. Phương thức này **phải cùng kiểu trả về và tham số đầu vào** với **Delegate** đã tạo.

Khi tạo một **Delegate**, tham số được truyền với biểu thức **new** được **viết tương tự** như một **lời gọi phương thức**, nhưng **không có tham số tới phương thức đó**. Tức là **chỉ truyền tên hàm** vào thôi. **Delegate** sẽ tự nhận định hàm được đưa vào có cùng kiểu dữ liệu trả ra và cùng tham số đầu vào hay không.

Ví dụ:**C#:**

```
class Program
{
    delegate int MyDelegate(string s);
    static void Main(string[] args)
    {
        Console.OutputEncoding = Encoding.Unicode;

        MyDelegate convertToInt = new MyDelegate(ConvertStringToInt);

        string numberSTR = "35";

        int valueConverted = convertToInt(numberSTR);

        Console.WriteLine("Giá trị đã convert thành int: " + valueConverted);

        Console.ReadLine();
    }

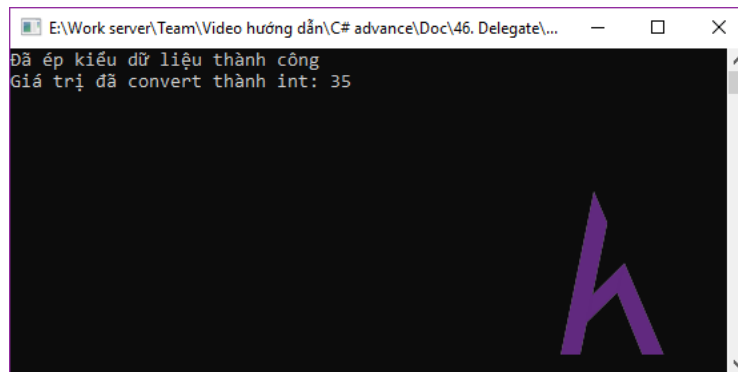
    static int ConvertStringToInt(string stringValue)
    {
        int valueInt = 0;

        Int32.TryParse(stringValue, out valueInt);

        Console.WriteLine("Đã ép kiểu dữ liệu thành công");

        return valueInt;
    }
}
```

Kết quả chạy chương trình:



Để các bạn hiểu rõ hơn về đoạn code trên thì mình sẽ giải thích một chút:

- Ở đây mình tạo một hàm `ConvertStringToInt` làm nhiệm vụ là chuyển kiểu dữ liệu của một số từ `string` sang `int`.
- Mình sử dụng `Delegate` bằng cách tạo một biến `convertToInt` có kiểu dữ liệu là `MyDelegate`. `convertToInt` này mình `new MyDelegate` với tham số đầu vào là tên hàm `ConvertStringToInt` (lưu ý chỉ tên hàm thôi).
- Mình có biến `numberSTR` kiểu `string` khởi tạo giá trị là `35`.
- Mình tạo một biến `valueConverted` kiểu `int` khởi tạo nó bằng kết quả gọi `Delegate convertToInt` với tham số truyền vào `Delegate` là biến `numberSTR`.
- Kết quả xuất ra màn hình Console là số `35`.

Nhận thấy `Delegate convertToInt` mình sử dụng tương tự như một hàm bình thường.

Do `MyDelegate` đã khởi tạo đồng nhất kiểu dữ liệu trả về và tham số đầu vào với hàm `ConvertStringToInt` nên `convertToInt` mới thỏa mãn điều kiện khởi tạo và sử dụng của hàm `ConvertStringToInt` này.

Vậy `Delegate` bản chất **chỉ là một biến thay thế cho hàm**, biến này tham chiếu đến hàm nó muốn tham chiếu để thay thế khi dùng. Cách dùng y như gọi một hàm.

Vì sao cần `Delegate`? Khi bạn **cần dùng một hàm như một biến** ví dụ như tham số truyền vào của một hàm, hàm call-back, event...

Multicast(đa hướng) một Delegate trong C#

Khi bạn cần **thực hiện một chuỗi hàm với cùng kiểu trả về và cùng tham số đầu vào** mà không muốn gọi nhiều hàm tuần tự (**chỉ gọi 1 hàm 1 lần duy nhất**). Lúc này bạn sẽ cần dùng đến **Multicast Delegate**.

Bản chất bạn có thể làm một chuỗi `Delegate` cùng kiểu `Delegate` bằng cách dùng toán tử `+`. Lúc này khi bạn gọi `Delegate` sẽ thực hiện tuần tự các `Delegate` được cộng vào với nhau.

Bạn có thể loại bỏ `Delegate` trong multicast bằng toán tử `-`.

Ví dụ:

C#:

```
class Program
{
    delegate int MyDelegate(string s);
    static void Main(string[] args)
    {
        Console.OutputEncoding = Encoding.Unicode;

        MyDelegate convertToInt = new MyDelegate(ConvertStringToInt);

        MyDelegate showString = new MyDelegate(ShowString);

        MyDelegate multicast = convertToInt + showString;

        string numberSTR = "35";

        int valueConverted = convertToInt(numberSTR);

        Console.WriteLine("Giá trị đã convert thành int: " + valueConverted);

        Console.WriteLine("Kết quả khi gọi multicast Delegate");
        multicast(numberSTR);

        Console.ReadLine();
    }

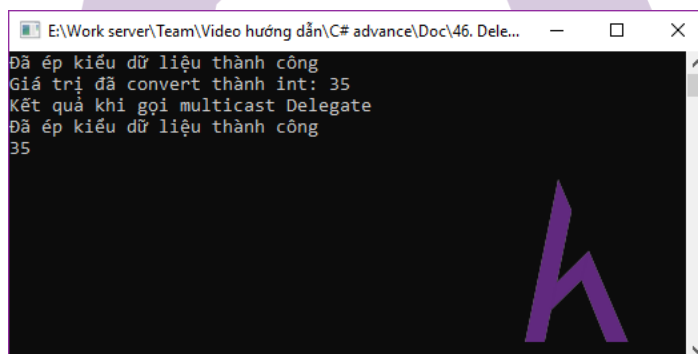
    static int ConvertStringToInt(string stringValue)
    {
        int valueInt = 0;

        Int32.TryParse(stringValue, out valueInt);
        Console.WriteLine("Đã ép kiểu dữ liệu thành công");

        return valueInt;
    }

    static int ShowString(string stringValue)
    {
        Console.WriteLine(stringValue);
        return 0;
    }
}
```

Kết quả: khi chạy chương trình



```
E:\Work server\Team\Video hướng dẫn\C# advance\Doc\46. Dele...
Đã ép kiểu dữ liệu thành công
Giá trị đã convert thành int: 35
Kết quả khi gọi multicast Delegate
Đã ép kiểu dữ liệu thành công
35
```

Dùng lại ví dụ của phần trước.

Mình tạo thêm hàm [ShowString](#) với mục đích là xuất ra màn hình Console chuỗi truyền vào. Mình tạo thêm 2 [Delegate](#) là [showString](#) tham chiếu tới hàm [ShowString](#) và [multicast](#) là kết quả cộng của 2 [Delegate](#) [convertToInt](#) và [showString](#).

Mình gọi [Delegate multicast](#) để thực hiện 1 lần 2 [Delegate](#) tuần tự là [convertToInt](#) và [showString](#).

C#:

```
Console.WriteLine("Kết quả khi gọi multicast Delegate");  
multicast(numberSTR);
```

Khi cần loại bỏ [Delegate](#) trong [multicast](#) bạn chỉ việc trừ [Delegate](#) ra

C#:

```
multicast = multicast - showString;
```

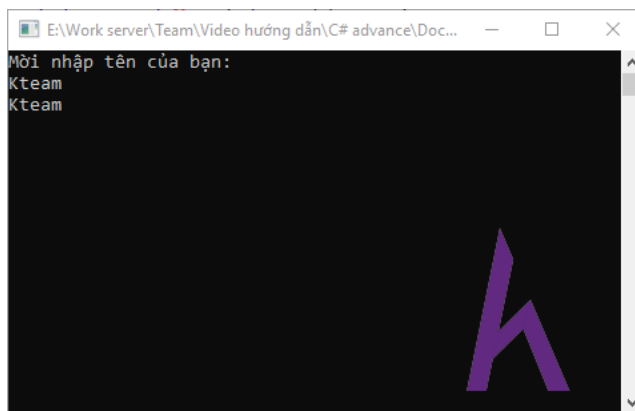
Dùng Delegate cho call-back function

Như mình đã nói ở trên, [Delegate](#) cũng là một biến. Vậy nên mình có thể truyền [Delegate](#) vào hàm làm **parameter** như biến bình thường. Lúc này [Delegate](#) này sẽ được gọi là **call-back function**. Mục đích của việc này là hàm nhận call-back function là param có thể gọi [Delegate](#) được đưa vào khi nào cần như ví dụ sau:

C#:

```
delegate int MyDelegate(string s);  
static void Main(string[] args)  
{  
    Console.OutputEncoding = Encoding.Unicode;  
  
    MyDelegate showString = new MyDelegate(ShowString);  
  
    NhapVaShowTen(showString);  
  
    Console.ReadLine();  
}  
  
static void NhapVaShowTen(MyDelegate showTen)  
{  
    Console.WriteLine("Mời nhập tên của bạn:");  
    string ten = Console.ReadLine();  
    showTen(ten);  
}  
  
static int ShowString(string stringValue)  
{  
    Console.WriteLine(stringValue);  
    return 0;  
}
```

Kết quả: Khi chạy chương trình:



Như bạn thấy, mình đã sử dụng [Delegate](#) làm call-back function thành công.

Ý nghĩa của ví dụ là mỗi khi người dùng nhập vào tên của mình thì sẽ gọi [Delegate ShowString](#) để hiển thị tên người dùng vừa nhập vào ra màn hình console. Vậy lúc này hàm [ShowString](#) này hoàn toàn có thể được định nghĩa do người dùng mà không cần can thiệp vào code của hàm [NhapVaShowTen](#).

Kết luận

Nội dung bài này giúp các bạn nắm được:

- Delegate là gì?
- Khai báo Delegate trong C#
- Khởi tạo Delegate trong C#
- Multicast(đa hướng) một Delegate trong C#
- Cách dùng Delegate trong C#
- Dùng Delegate như một call-back function

Bài học sau chúng ta sẽ cùng tìm hiểu về EVENT TRONG C#.

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên “**Luyện tập – Thử thách – Không ngại khó**”.