

## Bài 13

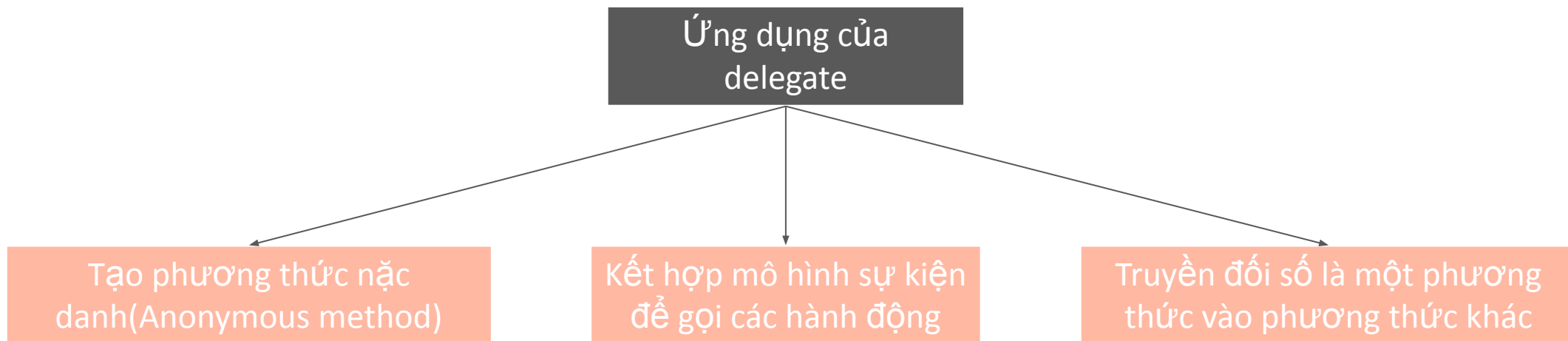
# Delegate và Event

# Mục tiêu

- Giới thiệu về Delegate
- Tạo và sử dụng Delegate
- Multicast Delegate
- Giới thiệu về sự kiện
- Tạo và sử dụng sự kiện

# Giới thiệu Delegate (ủy nhiệm)

- Delegate là một **kiểu tham chiếu**, nó định nghĩa lên **phương thức mẫu** (gồm tên, kiểu trả về, danh sách tham số và kiểu tham số). Thể hiện của delegate dùng để **tham chiếu (trỏ)** tới phương thức thật lúc run-time **có cùng mẫu với nó**.
- Chúng ta có thể thực thi phương thức thật thông qua thể hiện của delegate.



# Khai báo và sử dụng

## CÚ PHÁP:

`access_modifier delegate return_type Delegate_name([parameters])`

```
//Định nghĩa delegate
public delegate int Calculation(int a, int b);
public class Utility
{
    public static int Add(int a, int b)
    {
        return a + b;
    }
    public static int Sub(int a, int b)
    {
        return a - b;
    }
    public static double Div(double a, double b)
    {
        return a / b;
    }
}
```

```
public class Program
{
    static void Main(string[] args)
    {
        //tham chiếu delegate tới phương thức Add cùng mẫu
        Calculation cal = new Calculation(Utility.Add); //= Utility.Add;
        //thực thi phương thức
        int tong2so = cal(3, 5);
        Console.WriteLine(tong2so); //=8 nha
        //tham chiếu tới phương thức Sub cùng mẫu
        cal = Utility.Sub;
        int hieu2so = cal(3, 5);
        Console.WriteLine(hieu2so); //= -2 nha
        //tham chiếu tới phương thức Div khác mẫu -> lỗi
        cal = Utility.Div;
    }
}
```

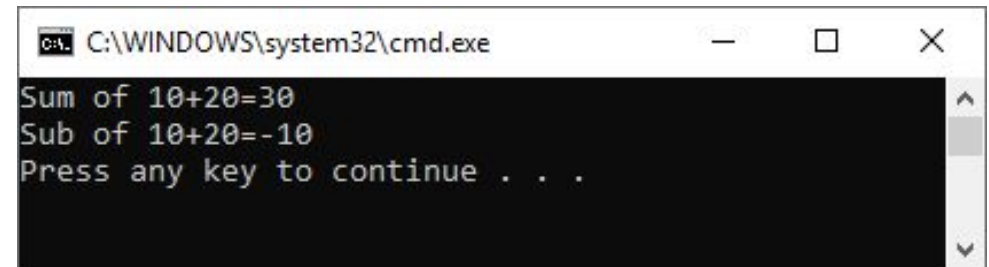
# Multicast Delegates

- Một delegate tham chiếu tới nhiều phương thức tại cùng 1 thời điểm được gọi là Multicast delegate
- Khi delegate thực thi thì tất cả các phương thức nó tham chiếu tới đều được thực thi.
- Các phương thức mà delegate tham chiếu tới phải có kiểu trả về là **void** nếu không trình biên dịch sẽ báo lỗi.
- Toán tử '+=' sẽ được sử dụng để tạo multicast delegate.

# Multicast Delegates

```
//Định nghĩa delegate
public delegate void Calculation(int a, int b);
public class Utility
{
    public static void Add(int a, int b)
    {
        int sum = a + b;
        Console.WriteLine("Sum of {0}+{1}={2}",a,b,sum);
    }
    public static void Sub(int a, int b)
    {
        int sub = a - b;
        Console.WriteLine("Sub of {0}+{1}={2}", a, b, sub);
    }
}
```

```
public class Program
{
    static void Main(string[] args)
    {
        //tham chiếu tới phương thức Add
        Calculation cal = Utility.Add;
        //tham chiếu nối tiếp với phương thức Sub
        cal += Utility.Sub;
        //gọi delegate -> cả 2 Add và Sub thực thi
        cal(10, 20);
    }
}
```

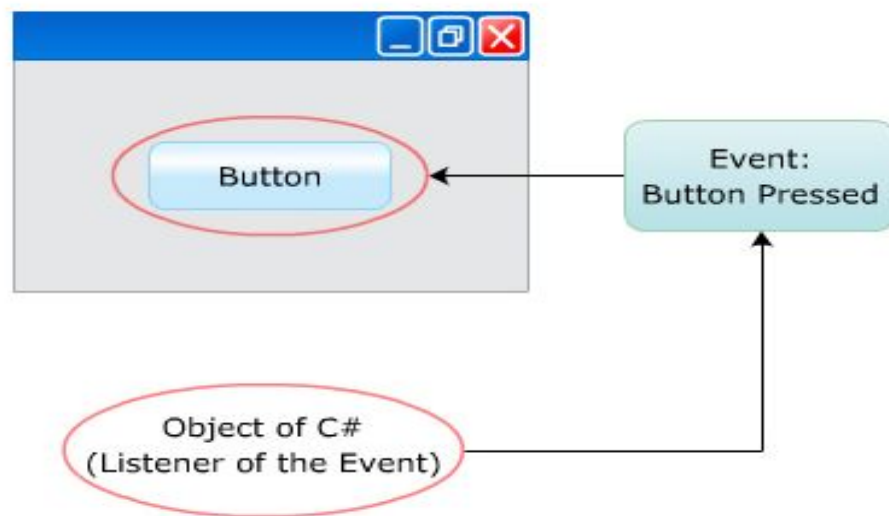


The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The output of the program is displayed as follows:

```
Sum of 10+20=30
Sub of 10+20=-10
Press any key to continue . . .
```

# Sự kiện (Events)

- Trong thực tế sự kiện là các hành động của người dùng như nhấn phím, click, di chuyển chuột, ... Các Application cần phản hồi các sự kiện này khi chúng xuất hiện. Ví dụ, các ngắt (interrupt). Các sự kiện (Event) được sử dụng để giao tiếp bên trong tiến trình.



# Đặc điểm của sự kiện

- Một sự kiện là một hành động được sinh ra bởi người dùng hoặc hệ thống, nó cho phép các đối tượng được yêu cầu thông báo tới các đối tượng khác hoặc lớp khác để xử lý sự kiện

They can be declared in classes and interfaces.

They can be declared as abstract or sealed.

They can be declared as virtual.

They are implemented using delegates.



# Tạo và sử dụng sự kiện

- Để tạo và sử dụng sự kiện chúng ta thực hiện theo 4 bước sau



- Các sự kiện sử dụng các delegate để gọi các phương thức trong các đối tượng đã đăng ký sự kiện
- Khi một sự kiện chứa đựng một loạt các đăng ký được sinh ra, nhiều delegate sẽ được gọi.

# Tạo và sử dụng sự kiện

- Cú pháp

## **Khai báo delegate**

```
<access_modifier> delegate <return_type> <DelegateName>(parameters);
```

## **Khai báo event**

```
<access_modifier> event <DelegateName> EventName;
```

# Tạo và sử dụng sự kiện

```
//khai báo delegate
delegate void HandleDisplay();
class Program
{
    //khai báo sự kiện
    event HandleDisplay Display;
    //định nghĩa phương thức
    public void Show()
    {
        Console.WriteLine("Hello world!");
    }
    static void Main(string[] args)
    {
        //tạo đối tượng
        Program p = new Program();
        //đăng ký sự kiện
        p.Display += new HandleDisplay(p.Show);
        //phát ra sự kiện
        p.Display();
    }
}
```

# HỎI ĐÁP





# TRẢI NGHIỆM THỰC HÀNH