**USER CONTROL**

Giảng Viên hướng dẫn: Thầy Lê Thanh Trọng

NHÓM 19:

Bùi Đình Lộc Thọ 13520844

Lê Tấn Thịnh 13520772

**PHỤ LỤC**

[GIỚI THIỆU USER CONTROL](#User control có thể hiểu là control)

[TẠO USER CONTROL TRONG C#](#Mở Visual studio và tạo)

[THÊM PROPERTY CHO USER CONTROL](#User Control được kế thừa từ class UserControl)

[MỘT SỐ HÀM CẦN THIẾT CHO THIẾT KẾ USER CONTROL](#Một số cài đặt cho User Control:)

THIẾT KẾ MỘT USER CONTROL ĐƠN GIẢN

**GIỚI THIỆU USER CONTROL**

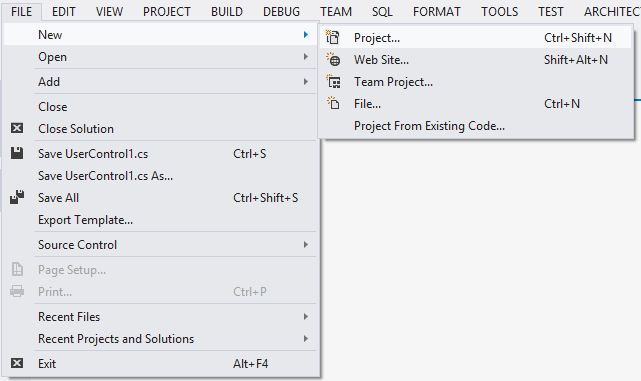
* User control có thể hiểu là control do người dùng tạo ra để phục vụ những vấn đề mà những control mặc định của hệ thống không làm được, khó thao tác, bị trùng lặp thao tác nhiều hoặc là để có thêm nhiều chức năng hơn đôi khi là muốn cho giao diện của phần mềm đẹp hơn và mới lạ hơn.
* Và một trong những công cụ cần thiết để làm được User Control đó là GDI+ hỗ trợ cho việc vẽ và thiết kế giao diện cho User Control. Và các công cụ nằm trong **Namespace System.Drawing**

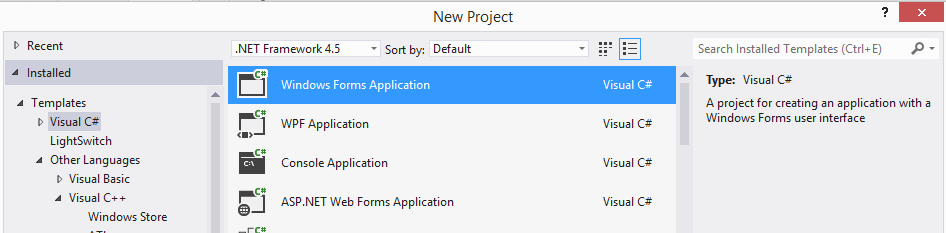
**VD:**

* + Khi bạn sử dụng phần mềm, có một số Button bạn thấy có hiệu ứng khi rê chuột vào thì sáng dần lên rất đẹp,…. Nhưng Button mặc định thì chỉ có một kiểu khi đó bạn muốn làm những Button hay thậm chí những control khác đẹp hơn thì phải làm bằng User Control.

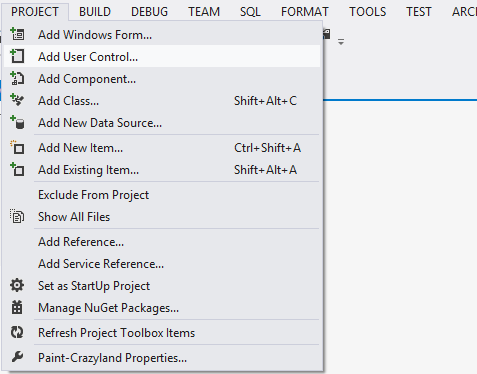
**TẠO USER CONTROL TRONG C#**

* **Bước 1**: Mở Visual studio và tạo 1 project Windows Application (C#).





* **Bước 2**: sau khi tạo project xong bạn add user control bằng cách vào menu **Project > Add User control….** Sau đó đặt tên cho User Control xong chọn OK. Sau đó màn hình làm việc của User Control sẽ được hiện lên





Như vậy chỉ với 2 bước đơn giản là có thể tạo được 1 User control.

* Bạn có thể nhấp đúp vào vùng Region của User control để xem code:



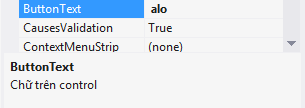
* Như ta thấy thì class UserControl1 kế thừa từ class UserControl và nó chỉ có hàm khởi tạo Component và chúng ta sẽ code trong này để làm thành 1 control cho riêng mình.
* Vì kế thừa từ class UserControl nên User Control của ta sẽ có các thuộc tính và sự kiện của class này.

**THÊM PROPERTY CHO USER CONTROL**

* User Control được kế thừa từ class UserControl nên nó sẽ có những thuộc tính cơ bản của class này (như BackColor, Size, Location ,…..).
* Chúng ta hoàn toàn có thể thêm các thuộc tính khác để làm cho Control của chúng ta thêm đa dạng hơn. **VD**: như User control không hỗ trợ sẵn thuộc tính Text nên nếu ta làm 1 button thì cần thêm thuộc tính này để có thể hiện Text lên form (thực ra thuộc tính Text ở User Control bị ẩn đi chúng ta phải override và thêm các attributes cần thiết để hiện lên).
* Cách thêm thuộc tính trong User Control cũng giống như thêm 1 thuộc tính trong class tuy nhiên nó có thêm một số thành phần là các attributes được khai báo trong ngoặc vuông phía trên thuộc tính muốn thêm các mô tả:

//hiển thị thể loại và nội dung của User Control ở hộp Properties

[Description("Chữ trên control"), Category("Demo")]

string m\_buttonText;

public string ButtonText

{

get { return m\_buttonText; }

set { m\_buttonText = value; }

}

* Sau khi thêm một Properties thành công thì bạn có thể qua Form để thêm User Control vào form (nhớ rebuild solotion, sau khi rebuild thì User Control sẽ nằm ở ToolBox) . ta có thể thay đổi thuộc tính của User Control ngay ở hộp Properties (chuột phải vào User Control chọn Properties).

**MỘT SỐ HÀM CẦN THIẾT CHO THIẾT KẾ USER CONTROL**

* Một số cài đặt cho User Control:
  + this.DoubleBuffered = true;  
    cho phép tăng gấp đôi bộ đệm (chống giật hình)
* Một số hàm cần thiết khi làm User Control:  
  + **Hàm SetStyle(ControlStyles flag, bool Value)**  
    các ControlStyles có thể ngăn cách nhau bằng dấu gạch đứng “|”, biến bool cho phép kích hoạt chức năng đó hay không, hàm này thường được gọi ở Construct của User Control.  
    - this.SetStyle(ControlStyles.OptimizedDoubleBuffer | ControlStyles.SupportsTransparentBackColor, true);  
        
      ở đây là cho phép cải thiện DoubleBuffer và cho phép làm màu nền trong suốt khi vẽ form ngoài ra còn nhiều options khác.
  + **Hàm OnPaint(PaintEventArgs e)**

protected override void OnPaint(PaintEventArgs e)

{

base.OnPaint(e);  
}  
  
ta override lại hàm OnPaint đây là hàm vẽ User Control mọi thiết kế đồ họa của ta sẽ vẽ trong hàm này. Hàm này có 1 tham số truyền vào có chứa 1 Graphics dùng cung cấp các công cụ để vẽ lên User Control (e.Graphics). Hàm này được gọi thi ta gọi hàm Invalidate().

* + **Các hàm Override cử chỉ của Chuột:**  
      
    //hàm này được gọi khi chuột di chuyển trên Region của User Control

protected override void OnMouseMove(MouseEventArgs e)

{

base.OnMouseMove(e);

}

//hàm này được gọi khi có sự kiện nhấn chuột xuống Region của User Control

protected override void OnMouseDown(MouseEventArgs e)

{

base.OnMouseDown(e);

}

//hàm này được gọi khi có sự kiện MouseUp trên Region của User Control

protected override void OnMouseUp(MouseEventArgs e)

{

base.OnMouseUp(e);

}

//hàm này được gọi khi chuột nằm trên Region của Control

protected override void OnMouseHover(EventArgs e)

{

base.OnMouseHover(e);

}

//hàm này được gọi khi chuột rời khỏi Region User Control

protected override void OnMouseLeave(EventArgs e)

{

base.OnMouseLeave(e);

}

**THIẾT KẾ MỘT USER CONTROL ĐƠN GIẢN**