Trường ĐH Công Thương TP. HCM Khoa Công nghệ Thông tin Bộ môn Kỹ thuật phần mềm **Công nghệ .NET**	**BÀI 2. STYLE, TEMPLATE VÀ RESOURCE**	

A. MỤC TIÊU

- Thiết kế được form giao diện thực tế.
- Tạo giao diện thân thiện, dễ đọc, dễ thao tác

B. DỤNG CỤ - THIẾT BỊ THỰC HÀNH CHO MỘT SINH VIÊN

STT	Thiết bị - Phần mềm – Thư viện	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	
2	Microsoft Visual Studio 2013 trở lên	1	1	

C. NỘI DUNG THỰC HÀNH

1. Tóm tắt lý thuyết

I. Resource

Resource là dữ liệu thiết kế có thể tái sử dụng, được khai báo một lần và dùng lại cho nhiều control.

Resource giúp giảm trùng lặp mã XAML, dễ thay đổi giao diện và hỗ trợ xây dựng giao diện thống nhất trong toàn bộ ứng dụng.

Phạm vi (Scope) của Resource:

- Resource cục bộ: khai báo trong control và chỉ control đó sử dụng.

```
<Button>
    <Button.Resources>
        <SolidColorBrush x:Key="MyBrush" Color="LightBlue"/>
    </Button.Resources>
</Button>
```

- Resource trong Window/UserControl: Dùng cho tất cả phần tử con chứa bên trong:

```
<Window.Resources>
    <SolidColorBrush x:Key="PrimaryColor" Color="#3498db"/>
</Window.Resources>
```

- Resource toàn ứng dụng (App.xaml): áp dụng cho tất cả phần tử (control) trong ứng dụng:

```
<Application.Resources>
  <SolidColorBrush x:Key="AppBackground" Color="#f0f0f0"/>
</Application.Resources>
```

- ResourceDictionary: tách resource ra file riêng và khai báo trong App.xaml:

```
<ResourceDictionary>
  <ResourceDictionary.MergedDictionaries>
    <ResourceDictionary Source="Styles/ButtonStyle.xaml"/>
    <ResourceDictionary Source="Styles/TextBoxStyle.xaml"/>
  </ResourceDictionary.MergedDictionaries>
</ResourceDictionary>
```

II. Style

Style là tập hợp các Setter dùng để thiết lập các thuộc tính giao diện cho control, giúp tạo giao diện đồng nhất và dễ bảo trì.

```
<Window.Resources>
  <Style x:Key="PrimaryButton" TargetType="Button">
    <Setter Property="Background" Value="#3498db"/>
    <Setter Property="Foreground" Value="White"/>
    <Setter Property="Padding" Value="10"/>
    <Setter Property="FontWeight" Value="Bold"/>
  </Style>
</Window.Resources>
```

Gọi Style:

```
<Button Style="{StaticResource PrimaryButton}" Content="Lưu"/>
```

Nếu không chỉ định Key, Style tự động áp dụng cho toàn bộ control cùng loại. Ví dụ sau, tất cả TextBox đều dùng chung style:

```
<Window.Resources>
  <Style TargetType="TextBox">
    <Setter Property="Margin" Value="5"/>
    <Setter Property="Padding" Value="6"/>
  </Style>
</Window.Resources>
```

Ngoài ra, có thể Style mới dựa trên Style sẵn có:

```
<Style x:Key="DangerButton" TargetType="Button"
                        BasedOn="{StaticResource PrimaryButton}">
  <Setter Property="Background" Value="Red"/>
</Style>
```

III. ControlTemplate

ControlTemplate: định nghĩa lại cấu trúc hiển thị (visual tree) của một Control. Ví dụ: biến Button thành hình tròn, gradient, thêm icon…

```
<ControlTemplate TargetType="Button">
        <Border CornerRadius="8"
                Background="LightBlue" Padding="10">
            <ContentPresenter HorizontalAlignment="Center"
                              VerticalAlignment="Center" />
        </Border>
</ControlTemplate>
```

Trong đó:

- TargetType="Button": Áp dụng cho Button
- Border: Vẽ giao diện mới
- ContentPresenter: Nơi hiển thị Content của Button (Text, Image…). Không có ContentPresenter thì Button không hiển thị nội dung

Áp dụng ControlTemplate cho Button:

1. Gán trực tiếp:

```
<Button Content="Đăng nhập">
    <Button.Template>
        <ControlTemplate TargetType="Button">
            <Border CornerRadius="8"
                    Background="LightBlue" Padding="10">
                <ContentPresenter HorizontalAlignment="Center"
                                  VerticalAlignment="Center" />
            </Border>
        </ControlTemplate>
    </Button.Template>
</Button>
```

2. Đặt trong Style:

```
<Style x:Key="RoundedButtonStyle" TargetType="Button">
  <Setter Property="Template">
    <Setter.Value>
      <ControlTemplate TargetType="Button">
        <Border Background="#2D89EF"
                CornerRadius="10" Padding="8">
            <ContentPresenter  HorizontalAlignment="Center"
                               VerticalAlignment="Center"
                               Foreground="White"/>
        </Border>
      </ControlTemplate>
    </Setter.Value>
  </Setter>
</Style>
```

Sử dụng Style:

```
<Button Content="Lưu"
        Style="{StaticResource RoundedButtonStyle}" />
```

IV. DataTemplate

DataTemplate: Dùng để thiết kế giao diện hiển thị dữ liệu: ListBox, ComboBox, ListView / GridView, DataGrid.

```
<ListBox.ItemTemplate>
  <DataTemplate>
    <StackPanel Orientation="Horizontal">
      <Image Source="{Binding Avatar}" Width="40" Height="40"/>
      <TextBlock Text="{Binding Name}" Margin="10 0"/>
    </StackPanel>
  </DataTemplate>
</ListBox.ItemTemplate>
```

V. Trigger trong Style và Template

Trigger cho phép thay đổi thuộc tính giao diện khi trạng thái thay đổi.

- Property Trigger

```
<Trigger Property="IsMouseOver" Value="True">
  <Setter Property="Background" Value="#5dade2"/>
</Trigger>
```

Giải thích: Khi chuột di chuyển vào control (IsMouseOver = True): đổi màu nền (Background) của control thành "#5dade2".

Sử dụng:

```
<Style TargetType="Button">
  <Style.Triggers>
    <Trigger Property="IsMouseOver" Value="True">
      <Setter Property="Background" Value="#5dade2"/>
    </Trigger>
  </Style.Triggers>
</Style>
```

- DataTrigger (so sánh với dữ liệu)

```
<DataTrigger Binding="{Binding IsValid}" Value="False">
   <Setter Property="Background" Value="Red"/>
</DataTrigger>
```

Giải thích: Khi giá trị IsValid (từ DataContext) của control = False: đổi Background của control thành màu đỏ.

VI. StaticResource và DynamicResource

Loại	Đặc điểm	Khi sử dụng
StaticResource	Load 1 lần khi khởi tạo	Style, Brush không thay đổi
DynamicResource	Cập nhật khi resource thay đổi	Theme, ngôn ngữ, màu sắc runtime

```
Background="{DynamicResource ThemeColor}"
```

2. Bài tập thực hành trên lớp

Bài tập có hướng dẫn

Bài 1. Viết ứng dụng kiểm tra lỗi (giao diện WPF).

Yêu cầu:

- Thiết kế Form theo mẫu:
  - Tô nền gradient
  - Sử dụng layout: Grid/ StackPanel/ WrapPanel

- Sử StaticResource thiết kế Style đồng nhất cho 2 textBox (tùy chọn định dạng màu nền, viền, màu chữ)
- Sử StaticResource thiết kế Style đồng nhất cho 3 nút Button (nền xanh, chữ trắng, khoảng cách lề 10, in đậm)

- Kiểm tra dữ liệu khi mất tiêu điểm (sự kiện LostFocus):
  - TextBox họ tên: Nếu không được nhập dữ liệu thì hiện thông báo lỗi (chuỗi thông báo) bên phải TextBox
  - TextBox năm sinh: năm sinh không phải là số hoặc năm sinh là số âm hoặc lớn hơn năm hiện tại thì thông báo lỗi (chuỗi thông báo) bên phải TextBox
- Chức năng:
  - Nút Hiển thị: hiển thị thông tin nhập vào Messagebox bao gồm: tên, tuổi (năm hiện tại – năm sinh).
  - Nút Xóa: xóa thông tin đã nhập trên các TextBox, đặt con trỏ văn bản vào Textbox Nhập họ tên.
  - Nút Exit: xác nhận người dùng có thực sự muốn thoát khỏi chương trình không? Yes: thoát, No: không

Hướng dẫn:

Bước 1: Tạo mới ứng dụng trên Visual Studio (BTM1.xaml)

Bước 2: Thiết kế giao diện

Code XAML:

```
<!-- ===== KHAI BÁO RESOURCE ===== -->
<Window.Resources>
  <!-- Style TextBox (dùng cho 2 TextBox) -->
  <Style x:Key="InputTextBox" TargetType="TextBox">
    <Setter Property="Width" Value="240"/>
    <Setter Property="Height" Value="28"/>
    <Setter Property="Margin" Value="10,0,0,0"/>
    <Setter Property="Padding" Value="5"/>
    <Setter Property="Background" Value="White"/>
    <Setter Property="BorderBrush" Value="#6C8EBF"/>
    <Setter Property="Foreground" Value="Black"/>
  </Style>

  <!-- Style Button (dùng cho 3 Button) -->
  <Style x:Key="MainButton" TargetType="Button">
    <Setter Property="Width" Value="90"/>
    <Setter Property="Height" Value="35"/>
    <Setter Property="Margin" Value="10"/>
    <Setter Property="Background" Value="#3498DB"/>
    <Setter Property="Foreground" Value="White"/>
    <Setter Property="FontWeight" Value="Bold"/>
  </Style>
</Window.Resources>
```

```
<!-- ===== LAYOUT ===== -->
<Grid Margin="10">
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto"/>
    <RowDefinition Height="*"/>
  </Grid.RowDefinitions>
```

```
<!-- Tiêu đề -->
<TextBlock Text="Thông tin"
        FontSize="22"
        FontWeight="Bold"
        HorizontalAlignment="Center"
        Margin="0,10"
        Grid.Row="0"/>
<!-- Nội dung -->
<Border Grid.Row="1"
      CornerRadius="12"
      BorderThickness="2"
      BorderBrush="#9DB5E4"
      Padding="20">
   <!-- Gradient nền -->
   <Border.Background>
      <LinearGradientBrush StartPoint="0,0" EndPoint="0,1">
         <GradientStop Color="White" Offset="0"/>
         <GradientStop Color="#D6EAF8" Offset="1"/>
      </LinearGradientBrush>
   </Border.Background>
   <StackPanel>
      <!-- Nhập họ tên -->
      <StackPanel Orientation="Horizontal" Margin="0,10">
         <TextBlock Text="Nhập họ tên:"
                Width="90" VerticalAlignment="Center"/>
         <TextBox Style="{StaticResource InputTextBox}"/>
      </StackPanel>
      <!-- Năm sinh -->
      <StackPanel Orientation="Horizontal" Margin="0,10">
         <TextBlock Text="Năm sinh:"  Width="90"
                    VerticalAlignment="Center"/>
```

```
            <TextBox Style="{StaticResource InputTextBox}"/>
        </StackPanel>
        <!-- Button -->
        <WrapPanel HorizontalAlignment="Center" Margin="0,20">
            <Button Content="Hiển thị"
                    Style="{StaticResource MainButton}"/>
            <Button Content="Xóa"
                    Style="{StaticResource MainButton}"/>
            <Button Content="Thoát"
                    Style="{StaticResource MainButton}"/>
        </WrapPanel>
    </StackPanel>
  </Border>
</Grid>
```

Bước 3: Viết mã lệnh C#

- TextBox Họ tên mất tiêu điểm (LostFocus)

```
private void txtName_LostFocus(object sender, RoutedEventArgs e)
{
    if (string.IsNullOrWhiteSpace(txtName.Text))
        txtNameError.Text = "Họ tên không được để trống!";
    else
        txtNameError.Text = string.Empty;
}
```

- TextBox Năm sinh mất tiêu điểm (LostFocus)

```
private void txtYear_LostFocus(object sender, RoutedEventArgs e)
{
    int year;
    int currentYear = DateTime.Now.Year;
    if (!int.TryParse(txtYear.Text, out year))
        txtYearError.Text = "Năm sinh phải là số!";
```

```
    else
    {
        if (year < 0)
            txtYearError.Text = "Năm sinh không là số âm!";
        else if (year > currentYear)
            txtYearError.Text = "Năm sinh không \n lớn hơn năm hiện tại!";
        else
            txtYearError.Text = string.Empty;
    }
}
```

- Hiển thị thông tin theo yêu cầu khi click vào Button Show:

```
private void btnShow_Click(object sender, RoutedEventArgs e)
{
    int year = Convert.ToInt32(txtYear.Text);
    int age = DateTime.Now.Year - year;

    string message = "Họ tên: " + txtName.Text + "\n"
                + "Tuổi: " + age;

    MessageBox.Show(message, "Thông tin",
                MessageBoxButton.OK,
                MessageBoxImage.Information);
}
```

- Button Clear thực hiện lệnh xóa các TextBox:

```
private void btnClear_Click(object sender, EventArgs e)
{
    txtName.Clear();
    txtYear.Clear();
    txtName.Focus();
}
```

- Thoát chương trình khi click vào Button Exit:

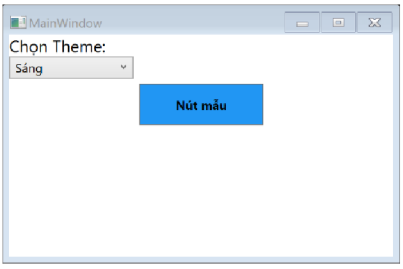
```
private void btnExit_Click(object sender, EventArgs e)
{
    this.Close();
}
```

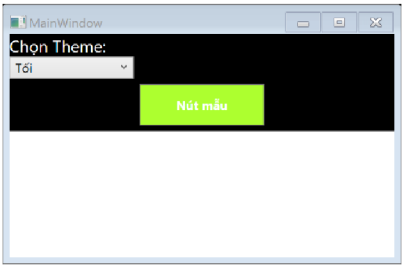
- Sự kiện Window_Closing

```
private void Window_Closing(object sender,
                            System.ComponentModel.CancelEventArgs e)
{
   MessageBoxResult result = MessageBox.Show(
                            "Bạn có chắc chắn muốn thoát không?",
                            "Xác nhận",
                             MessageBoxButton.YesNo,
                             MessageBoxImage.Question);

   if (result == MessageBoxResult.No)
      e.Cancel = true;
}
```

Bài 2. Tạo Theme Light/Dark cho ứng dụng





Yêu cầu:

– Tạo 2 Resource Dictionary: ThemeLight.xaml và ThemeDark.xaml. Mỗi theme định nghĩa: Brush nền (Background), Brush chữ (Foreground), Brush button (PrimaryButtonColor)

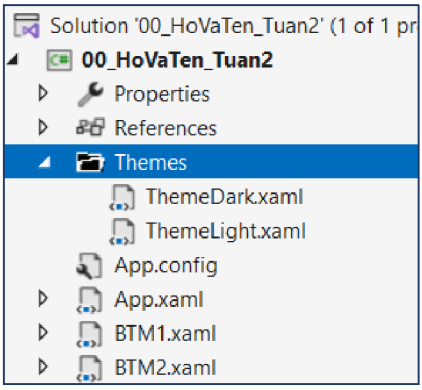
– Mỗi theme định nghĩa màu nền, màu chữ, màu button.

– Cho phép người dùng chuyển đổi theme bằng ComboBox

Hướng dẫn:

Bước 1: Tạo 2 file Resource Dictionary trong thư mục **Themes** theo cấu trúc cây thư mục:

- Tạo folder Themes
- Right click Themes → Add → New Item → Resource Dictionary



Trong đó:

- ThemeLight.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <!-- Nền -->
  <SolidColorBrush x:Key="BackgroundColor" Color="White"/>
  <!-- Chữ -->
  <SolidColorBrush x:Key="ForegroundColor" Color="Black"/>
  <!-- Button -->
  <SolidColorBrush x:Key="PrimaryButtonColor" Color="#3498DB"/>
</ResourceDictionary>
```

- ThemeDark.xaml

```
<ResourceDictionary
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <!-- Nền -->
  <SolidColorBrush x:Key="BackgroundColor" Color="#1E1E1E"/>
```

```
    <!-- Chữ -->
    <SolidColorBrush x:Key="ForegroundColor" Color="White"/>
    <!-- Button -->
    <SolidColorBrush x:Key="PrimaryButtonColor" Color="#2ECC71"/>
</ResourceDictionary>
```

Bước 2: Thiết kế giao diện chính có ComboBox để chuyển theme và nút nhấn Button.

```
<Window Title="MainWindow" Height="450" Width="800">
    <StackPanel Background="{DynamicResource BackgroundColor}"
                VerticalAlignment="Top">
        <TextBlock Text="Chọn Theme:"
            Foreground="{DynamicResource ForegroundColor}"
            FontSize="16"/>
        <ComboBox x:Name="cboTheme" HorizontalAlignment="Left"
                  VerticalAlignment="Top" Width="120"
                  SelectionChanged="cboTheme _SelectionChanged">
            <ComboBoxItem> Sáng </ComboBoxItem>
            <ComboBoxItem> Tối </ComboBoxItem>
        </ComboBox>
        <Button Content="Nút mẫu"
            Width="120" Height="40"
            Background="{DynamicResource ButtonColor}"
            Foreground="{DynamicResource ForegroundColor}"
            FontWeight="Bold" Margin="5"/>
    </StackPanel>
</Window>
```

Bước 3: Thiết lập theme mặc định (Light) (App.xaml)

```
<Application.Resources>
   <ResourceDictionary>
      <ResourceDictionary.MergedDictionaries>
         <!-- Theme mặc định -->
         <ResourceDictionary Source="Themes/ThemeLight.xaml"/>
      </ResourceDictionary.MergedDictionaries>
</ResourceDictionary>
</Application.Resources>
```

Bước 4: Viết xử lý chuyển theme trong BTM2.xaml.cs

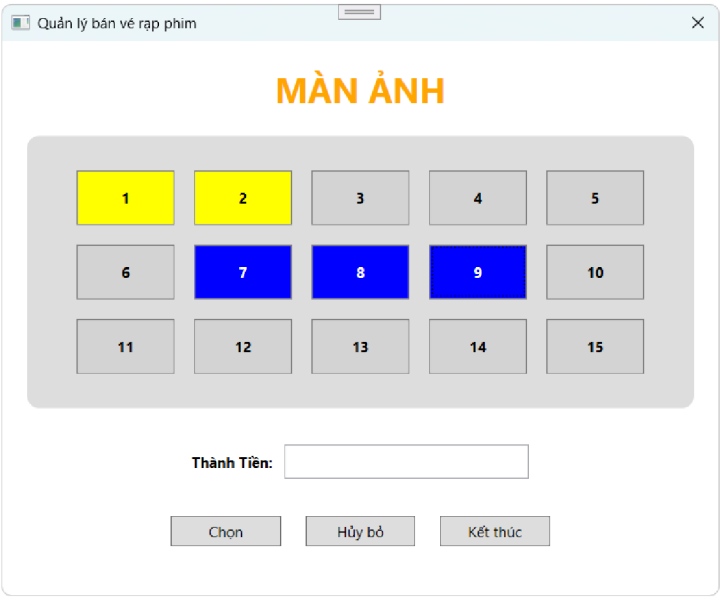
```
private void ChangeTheme(string themeName)
{
   ResourceDictionary dict = new ResourceDictionary();
   dict.Source = new Uri($"Themes/{themeName}.xaml",
                                                UriKind.Relative);

   Application.Current.Resources.MergedDictionaries.Clear();
   Application.Current.Resources.MergedDictionaries.Add(dict);
}
```

```
private void cboTheme_SelectionChanged(object sender,
                                        SelectionChangedEventArgs e)
{
   if (cboTheme.SelectedIndex == 0)
      ChangeTheme("ThemeLight");
   else
      ChangeTheme("ThemeDark");
}
```

Bài tập sinh viên tự làm

Bài 3. Quản lý bán vé rạp chiếu phim



Yêu cầu:

- Tạo một project giúp quản lý vé cho rạp chiếu phim.
- **Giao diện:**
  - Tiêu đề MÀN ẢNH (màu cam)
  - Rạp có 15 ghế, được bố trí: 3 hàng × 5 cột
  - Các ghế được canh đều và canh giữa Form
  - Mỗi ghế được biểu diễn bằng Button
  - Ghế được đánh số thứ tự từ 1 đến 15, theo thứ tự từ trái sang phải, từ trên xuống dưới.
  - Mỗi ghế có 3 trạng thái, được phân biệt bằng màu sắc:
    - Ghế trống (chưa bán) - Màu xám
    - Ghế đang chọn - Màu xanh
    - Ghế đã bán - Màu vàng
  - TextBox / Label “Thành tiền”: hiển thị tổng số tiền cần thanh toán
  - Ba nút chức năng: Chọn, Hủy bỏ, Kết thúc.

- **Chức năng:**
  - **Xử lý khi chọn ghế:**

    Khi người sử dụng click chuột tại một vị trí ghế trên sơ đồ thì:

    - Nếu là ghế chưa bán (màu xám) thì đổi màu ghế sang màu xanh (đánh dấu vị trí ghế đang được chọn).
    - Nếu là ghế đang chọn (có màu xanh) thì đổi màu của ghế trở về màu xám (hủy chọn).
    - Nếu là ghế đã bán (màu vàng) thì xuất hiện một MessageBox thông báo "Vị trí này đã được bán, vui lòng chọn ghế khác!" cho người sử dụng biết.
  - **Xử lý nút CHỌN**
    - Đổi màu tất cả các ghế đang chọn (màu xanh) trên sơ đồ sang màu vàng (đây là vị trí đã bán).
    - Tính tổng số tiền phải thanh toán
    - Hiển thị tổng tiền vào ô Thành tiền
    - Giá vé được tính theo từng lô ghế:
      1. Giá vé lô A (hàng 1): 1000/vé
      2. Giá vé lô B (hàng 2): 1500/vé
      3. Giá vé lô C (hàng 3): 2000/vé
  - **Xử lý nút HỦY BỎ:**
    - Đổi màu các vị trí đang chọn (màu xanh) về trạng thái ghế trống.
    - Thành Tiền hiển thị giá trị 0.
  - **Xử lý nút Kết thúc:** thực hiện xác nhận người dùng và đóng form.

3. Bài tập về nhà

Bài 4. Thiết kế giao diện hỗ trợ giải phương trình bậc nhất và phương trình bậc hai.

- Trạng thái ban đầu khi Form được load:
  - Nút Giải ở trạng thái không khả dụng (IsEnabled = false).
  - TextBox Nhập c ở trạng thái mờ / không khả dụng.

- **Xử lý khi chọn *Phương trình bậc nhất*:**
  - TextBox Nhập c bị ẩn hoặc vô hiệu hóa.
  - Người dùng phải nhập đầy đủ dữ liệu vào a và b.
  - Khi dữ liệu hợp lệ:
    - Nút Giải được kích hoạt.
  - Khi nhấn nút **Giải**:
    - Chương trình giải phương trình bậc nhất
    - Hiển thị kết quả vào TextBox Kết quả.
    - Sau khi hiển thị kết quả, nút Giải trở về trạng thái mờ (không khả dụng).

- **Xử lý khi chọn *Phương trình bậc hai:***
  - Hiển thị đầy đủ 3 TextBox a, b, c.

- Người dùng phải nhập đầy đủ dữ liệu vào các ô cần thiết.
- Khi dữ liệu hợp lệ:
  - Nút Giải được kích hoạt.
- Khi nhấn nút **Giải**:
  - Chương trình giải phương trình bậc hai
  - Hiển thị kết quả vào TextBox Kết quả.
  - Sau khi hiển thị kết quả, nút Giải trở về trạng thái mờ (không khả dụng).

- **Kiểm tra dữ liệu nhập**
  - Kiểm tra các TextBox a, b, c:
    - Không được để trống.
    - Phải là giá trị số hợp lệ.
  - Trường hợp dữ liệu không hợp lệ:
    - Hiển thị thông báo lỗi phù hợp cho người dùng.
    - Không cho phép thực hiện chức năng Giải.

- **Xử lý nút Thoát**
  - Khi người dùng nhấn nút Thoát hoặc đóng Form:
    - Hiển thị hộp thoại xác nhận:

      "Bạn có chắc chắn muốn thoát chương trình không?"
    - Yes: Đóng chương trình.
    - No: Quay lại Form.