Giao diện người dùng: Làm việc với XAML để tạo giao diện người dùng

Thiết kế giao diện người dùng là một phần quan trọng của quá trình phát triển ứng dụng phần mềm. Giao diện người dùng cho
các ứng dụng di động có thể được phát triển bằng cách sử dụng mã chương trình bằng ngôn ngữ lập trình (ví dụ: C Sharp, Java, Kotlin, Swift) hoặc sử dụng ngôn ngữ đánh dấu XAML

Mã XAML có một số ưu điểm so với
với mã chương trình tương đương, ví dụ: bằng ngôn ngữ C Sharp

- XAML ngắn gọn và dễ đọc hơn mã C Sharp tương đương.

- Hiển thị trực quan hơn về hệ thống phân cấp cha-con của các đối tượng giao diện người dùng.

Cũng có những nhược điểm, chủ yếu liên quan đến những hạn chế được tích hợp trong ngôn ngữ đánh dấu:

- Mã XAML không thể chứa mã xử lý sự kiện: tất cả các trình xử lý sự kiện phải được xác định trong tệp mã C Sharp có phần mở rộng xaml.cs.

- Mã XAML không thể chứa các vòng lặp cho
xử lý dữ liệu lặp đi lặp lại
- Mã XAML không được chứa các điều kiện
toán tử
- Mã XAML thường không thể tạo đối tượng
(các thể hiện của lớp) sử dụng hàm tạo
- Mã XAML không thể gọi các phương thức lớp

Tệp có phần mở rộng xaml chứa phần đánh dấu giao diện trực quan của trang và là tệp xml. Dòng đầu tiên là định nghĩa tệp xml tiêu chuẩn. Định nghĩa phần tử gốc ContentPage kết nối
bốn vùng tên bằng thuộc tính xmlns.

Không gian tên http://xamarin.com/schemas/2014/forms xác định hầu hết các loại từ Biểu mẫu Xamarin được sử dụng để xây dựng GUI.

Không gian tên thứ hai, http://schemas.microsoft.com/winfx/2009/xaml, xác định các kiểu và kiểu XAML cho thời gian chạy ngôn ngữ chung. Vì chỉ có một không gian tên có thể là không gian tên cơ sở nên không gian tên đó được sử dụng với tiền tố x: xmlns:x.
Điều này có nghĩa là các thuộc tính phần tử có trong không gian tên này sẽ được sử dụng với tiền tố x - x:Name hoặc x:Class.
Đây là những không gian tên chính.

Hai không gian tên sau đây chỉ được sử dụng trong quá trình phát triển trong Visual Studio.

Biểu thức mc:Ignorable="d" có nghĩa là tất cả các phần tử có tiền tố d sẽ bị bỏ qua trong quá trình thực thi ứng dụng và sẽ chỉ được sử dụng trong quá trình phát triển.

Sau khi kết nối các không gian tên sẽ có thuộc tính x:Class="ProjectName.MainPage", trỏ đến lớp đại diện cho trang này trong không gian tên "ProjectName".

Do đó, XAML cung cấp một khuôn khổ đơ n giản để xác định các thành phần giao diện người dùng và các thuộc tính của chúng. Mỗi phần tử

phải có thẻ mở và thẻ đóng, chẳng hạn như
tại điều khiển Nhãn đã thảo luận trước đó.

Thẻ được mở bằng khung góc mở và
được đóng bằng dấu gạch chéo và dấu ngoặc nhọn.

```
<Label
  Text="Xamarin.Forms"            HorizontalOptions="Center"
  VerticalOptions="StartAndExpand"  BackgroundColor="#EEEA0C"
  FontSize="30"                   TextColor="#000000"
  HeightRequest="50" />
```

Trong XAML, thuộc tính lớp thường được đặt làm thuộc tính XML:

```
<Label Text="Xin chào, XAML!"
    VerticalOptions="Trung tâm"
    FontAttribut="Đậm"
    FontSize="Lớn"
    TextColor="Aqua"

<Nhãn
      Text="Giao diện người dùng"
      HorizontalOptions="Center" VerticalOptions="StartAndExpand"
      BackgroundColor="#0F29E9" FontSize="20"
      FontAttribut="Đậm"            TextColor="#E3DE0C"
      Chiều caoRequest="40" />
```

Có một cách khác để đặt thuộc tính trong XAML.

Để làm điều này, bạn cần thêm thẻ mở và đóng vào mã chương trình, bao gồm tên lớp và tên thuộc tính, cách nhau bằng dấu chấm. Tuy nhiên, cú pháp phần tử thuộc tính có thể được sử dụng cho nhiều thuộc tính:

```
<Label Text="Xin chào, XAML!"
    VerticalOptions="Trung tâm">
  <Label.FontAttribution>
     In đậm
  </Label.FontAttribution>
  <Nhãn.Kích thước phông chữ>
     Lớn
  </Label.FontSize>
</Nhãn>
```

Mỗi phần tử trong XAML đại diện cho một đối tượng của một lớp C cụ thể
Sắc nét và thuộc tính của các phần tử tương ứng với thuộc tính của các lớp này.

Ví dụ: phần tử ContentPage thực sự sẽ đại diện cho một đối tượng
của lớp ContentPage cùng tên và phần tử Label là một đối tượng của lớp Label.

Lưu ý rằng các thuộc tính của lớp có thể lấy các giá trị thuộc nhiều loại khác
nhau: chuỗi, double, int, v.v.

Trong mã XAML, các thuộc tính phần tử tương ứng
có ý nghĩa văn bản.

```
<Label
    Text="Пользовательский интерфейс"
    HorizontalOptions="Center"   VerticalOptions="StartAndExpand"
    BackgroundColor="#0F29E9"  FontSize="20"
    FontAttributes="Bold"        TextColor="#E3DE0C"
    HeightRequest="40" />
```

Mã XAML có thể sử dụng các thuộc tính đính kèm, được nhận dạng trong các tệp
XAML dưới dạng thuộc tính chứa
tên lớp và thuộc tính, cách nhau bằng dấu chấm.

Chúng được gọi là thuộc tính đính kèm vì chúng được xác định
một lớp nhưng được gắn với các đối tượng khác.

Nhiệm vụ 1

Hãy mô tả cách sử dụng thuộc tính đính kèm bằng ví dụ về bố cục
Bố cục tuyệt đối.

Hãy tạo dự án Cờ vua cho phần mềm đa nền tảng
Ứng dụng Xamarin.Forms sử dụng mẫu Trống.

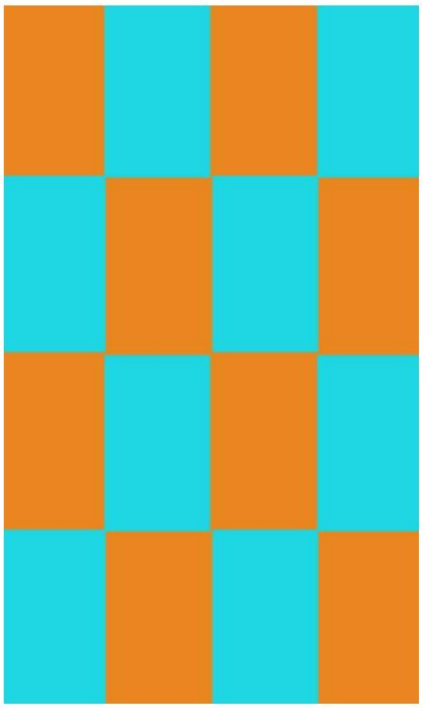
Hãy mở tệp MainPage.xaml và nhập mã chương trình vào đó. Mã sử dụng

bố cục AbsotutLayout và tám điều khiển BoxView để triển khai bàn cờ bằng cách
sử dụng các tính năng định vị theo tỷ lệ của điều khiển BoxView. Là thuộc tính đính
kèm cho điều khiển BoxView

Các thuộc tính LayoutBounds và LayoutFlags của bố cục Tuyệt đối được sử dụng.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
       xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
       x:Class="Chess.MainPage"
       Title="Trang demo tuyệt đối">

  <AbsoluteLayout BackgroundColor="#FF8080">
    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="0,33, 0, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="1, 0, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="0, 0,33, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="0,67, 0,33, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="0,33, 0,67, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="1, 0,67, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="0, 1, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" />

    <BoxView Color="#8080FF"
         Tuyệt đốiLayout.LayoutBounds="0,67, 1, 0,25, 0,25"
         Tuyệt đốiLayout.LayoutFlags="Tất cả" /
 > </AbsoluteLayout>
</ContentPage>
```

Khi bạn khởi động ứng dụng phần mềm, người dùng giao diện tươ ng ứng với năm chươ ng trình gõ XAML



Trong các ứng dụng phần mềm, để tránh tình trạng trang hiển thị chồng lên thanh trạng thái trên màn hình thiết bị di động khi sử dụng nền tảng iOS, thuộc tính
Phần đệm cho một trang bằng mã XAML. Ngoài ra, để
tính đến nền tảng mà ứng dụng phần mềm chạy trên đó, lớp OnPlatform được sử dụng để hoạt động với thuộc tính Padding.

Lớp OnPlatform có thuộc tính Nền tảng. Yếu tố bật
đặt các giá trị của thuộc tính Nền tảng, cũng như giá trị của thuộc tính Giá trị để đặt Độ dày Điều này được thực hiện bằng cách đặt giá trị của thuộc tính Độ dày bằng thuộc tính x:TypeArguments.

Các giá trị cho tham số Độ dày xác định độ rộng của khoảng trắng cho thuộc tính Padding.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
       xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
       x:Lớp="...">


  <ContentPage.Padding>
    <OnPlatform x:TypeArguments="Độ dày">
      <OnPlatform.Platforms>
        <Trên nền tảng="iOS" Value="0, 20, 0, 0" />
        <Trên nền tảng="Android" Value="0, 0, 0, 0" />
      </OnPlatform.Platforms>
    </OnPlatform>
  </ContentPage.Padding>
...
```

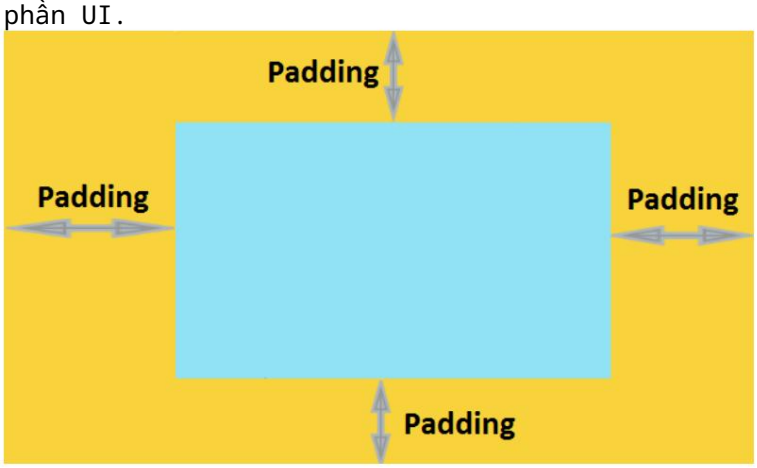
</Trang nội dung>

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="...">

  <ContentPage.Padding>
    <OnPlatform x:TypeArguments="Thickness">
      <OnPlatform.Platforms>
        <On Platform="iOS" Value="0, 20, 0, 0" />
        <On Platform="Android" Value="0, 0, 0, 0" />
      </OnPlatform.Platforms>
    </OnPlatform>
  </ContentPage.Padding>
  ...
</ContentPage>
```

Thuộc tính Padding chỉ định lượng không gian trống trên đường viền của thành phần UI. Ba tùy chọn Độ dày có thể được sử dụng để đặt thuộc tính Đệm. Độ dày với một tham số áp dụng cho các cạnh
trái, trên, phải và dưới cùng của thành phần giao diện người dùng;

Độ dày có hai giá trị (ngang và dọc) được áp dụng tương ứng cho bên trái và bên phải cũng như cho các cạnh trên và dưới của thành phần UI;

Độ dày bốn giá trị được áp dụng cho các cạnh trái, trên, phải và dưới của thành phần UI.



Nhiệm vụ 2

Mã XAML thường sử dụng các tiện ích mở rộng đánh dấu cho phép bạn đặt thuộc tính trên các đối tượng được các đối tượng khác trong mã của bạn tham chiếu gián tiếp. Do đó, tiện ích mở rộng đánh dấu XAML được thiết kế để chia sẻ các đối tượng trên mã.

Hãy xem xét việc sử dụng tiện ích mở rộng đánh dấu.

Để thực hiện việc này, hãy tạo một dự án MarkExt cho ứng dụng đa nền tảng Xamarin.Forms bằng cách sử dụng mẫu "Trống". Hãy mở tệp MainPage.xaml và nhập mã chương trình vào đó.

Mã để triển khai tiện ích mở rộng đánh dấu sử dụng thuộc tính ResourceDictionary của thuộc tính Resources của ContentPage để lưu trữ các giá trị thuộc tính để chia sẻ
bằng cách tham chiếu chúng từ mã XAML.

Để sử dụng từ điển tài nguyên trên một trang, mã sẽ sử dụng một cặp thẻ trong thuộc tính Tài nguyên của ContentPage. Thẻ ResourceDictionary được sử dụng để chỉ định từ điển tài nguyên.

Các thành phần giao diện người dùng và giá trị thuộc tính được thêm vào từ điển tài nguyên. Mỗi phần tử được cấp một khóa từ điển, được chỉ định bằng thuộc tính x:Key.

Hai phần tử được chỉ định, là các đối tượng thuộc loại LayoutOptions, mỗi phần tử có một khóa và tập thuộc tính duy nhất.

Thuộc tính x:Key chỉ định một phần tử từ điển borderWidth (để đặt độ rộng của đường viền của các điều khiển), là giá trị thuộc loại double và có giá trị là 3.

Thuộc tính x:Key chỉ định một phần tử từ điển spinningAngle (để chỉ định góc xoay của điều khiển), là một giá trị kép có giá trị là -15 độ.

Thuộc tính x:Key chỉ định một mục từ điển cho thuộc tính fontSize (để đặt kích thước phông chữ), là một giá trị kép có giá trị là 30.

Thuộc tính x:Key đặt thành phần từ điển cho thuộc tính textColor (màu của trang tùy thuộc vào nền tảng mà ứng dụng phần mềm đang chạy). Với mục đích này, từ điển tài nguyên sử dụng lớp
OnPlatform để xác định giá trị của thuộc tính textColor cho các nền tảng khác nhau.

Trong giao diện người dùng, bên trong một trang thuộc loại ContentPage có bố cục kiểu StackLayout, bên trong có hai nút điều khiển. Để đặt giá trị thuộc tính điều khiển, sáu đối

tượng từ từ điển tài nguyên được tham chiếu bằng tiện ích mở rộng đánh dấu StaticResource. Phần mở rộng đánh dấu StaticResource được phân tách bằng dấu ngoặc nhọn và trỏ đến một đối tượng trong từ điển.

Tiện ích mở rộng đánh dấu StaticResource được sử dụng để truy cập các đối tượng từ từ điển chỉ một lần khi tạo các điều khiển trên trang.

Tiện ích mở rộng đánh dấu DynamicResource được thiết kế để truy cập các đối tượng từ điển có giá trị có thể thay đổi trong quá trình thực thi ứng dụng phần mềm.

```
ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
        x:Class="MarkExt.MainPage"
        Title="Tiện ích mở rộng đánh dấu"
        BackgroundColor="Bisque">


  <ContentPage.Resources>
    <ResourceDictionary>
      <LayoutOptions x:Key="horzOptions"
              Căn chỉnh="Trung tâm" />

      <LayoutOptions x:Key="vertOptions"
              Căn chỉnh="Trung tâm"
              Mở rộng="Đúng" />

      <x:Double x:Key="borderWidth">3</x:Double>

      <x:Double x:Key="rotationAngle">-15</x:Double>

      <x:Double x:Key="fontSize">24</x:Double>

      <OnPlatform x:Key="textColor"
            x:TypeArguments="Color">
        <Trên nền tảng="iOS" Value="Red" />
        <Trên nền tảng="Android" Value="Aqua" />
        <Trên nền tảng="UWP" Value="#80FF80" />
      </OnPlatform>

    </ResourceDictionary> </
  ContentPage.Resources>

  <StackLayout>
    <Button Text="Tiện ích mở rộng XAML"
        widthRequest="200" BackgroundColor="San hô"
```

```
            HeightRequest="100" BorderColor="Đen"

            HorizontalOptions="{StaticResource hozOptions}"
            VerticalOptions="{StaticResource vertOptions}"
            BorderWidth="{StaticResource borderWidth}"
            Rotation="{StaticResource spinningAngle}"
            TextColor="{StaticResource textColor}"
            FontSize="{StaticResource fontSize}" />

        <Button Text="Từ điển tài nguyên"
            widthRequest="200" BackgroundColor="Nâu"
            HeightRequest="100" BorderColor="Vàng"

            HorizontalOptions="{StaticResource hozOptions}"
            VerticalOptions="{StaticResource vertOptions}"
            BorderWidth="{StaticResource borderWidth}"
            Rotation="{StaticResource spinningAngle}"
            TextColor="{StaticResource textColor}"
            FontSize="{StaticResource fontSize}" />

    </StackLayout>
</Trang nội dung>
```

Khi bạn khởi động ứng dụng phần mềm, người dùng giao diện tươ ng ứng với năm chươ ng trình XAML đã gõ.

Nhiệm vụ 3

Liên kết dữ liệu cho phép bạn liên kết các thuộc tính của hai đối tượng để việc thay đổi một trong số chúng sẽ thay đổi đối tượng kia. Liên kết dữ liệu kết nối các thuộc tính của hai điều khiển, được gọi là
nguồn dữ liệu và đối tượng đích.

Mã XAML yêu cầu hai bước để liên kết dữ liệu:

1. Gán nguồn dữ liệu cho điều khiển đối tượng đích bằng thuộc tính BindingContext. Việc đặt BindingContext của đối tượng đích được thực hiện bằng cách sử dụng tiện ích mở rộng đánh dấu ở dạng đối số x:Reference.

2. Sử dụng lớp Binding để liên kết thuộc tính nguồn
đối tượng đối với một thuộc tính của đối tượng mục tiêu.

Hãy xem xét việc sử dụng liên kết dữ liệu bằng XAML. Để thực hiện điều này, hãy tạo một dự án DatBind cho ứng dụng đa nền tảng Xamarin.Forms bằng mẫu Trống. Hãy mở tệp MainPage.xaml và nhập mã chương trình vào đó.

Mã XAML tương ứng với một ContentPage chứa StackLayout chứa điều khiển Thanh trượt và hai điều khiển Nhãn. Điều khiển Nhãn đầu tiên được xoay theo góc được chọn bởi điều khiển Thanh trượt. Một điều khiển Nhãn khác hiển thị giá trị của góc được chọn bởi điều khiển Thanh trượt.

Định nghĩa của điều khiển Thanh trượt chứa thuộc tính x:Name, được tham chiếu trong mã bởi hai điều khiển Nhãn bằng cách sử dụng tiện ích mở rộng đánh dấu x:Reference.

Tiện ích mở rộng đánh dấu x:Reference chỉ định tên của nguồn dữ liệu được gán bằng thuộc tính x:Name. Do đó, thanh trượt đóng vai trò là nguồn dữ liệu.

Mã sử dụng lớp Liên kết để liên kết thuộc tính Xoay của điều khiển Nhãn đầu tiên, được đặt tên là "ROTATION", với thuộc tính Giá trị của điều khiển Thanh trượt và để liên kết thuộc tính Văn bản của điều khiển Nhãn thứ hai với thuộc tính Giá trị của điều khiển Thanh trượt .

```
ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
        x:Class="DatBind.MainPage"
        Title="Liên kết thanh trượt"
        BackgroundColor="Orange">

  <StackLayout>
    <Văn bản nhãn="XOAY"
       VerticalTextAlignment="Trung tâm"
```

```
            HorizontalTextAlignment="Trung tâm"
            widthRequest="200" BackgroundColor="Nâu"
            HeightRequest="80" TextColor="Vàng"
            BindingContext="{x:Tên tham chiếu=slider}"
            Rotation="{Đường dẫn liên kết=Value}"
            FontAttribut="Đậm"
            Kích thước phông chữ="30"
            HorizontalOptions="Trung tâm"
            VerticalOptions="CenterAndExpand" />

        <Thanh trượt x:Tên="thanh trượt"
             Tối đa="360"
             VerticalOptions="CenterAndExpand" />
        <Label BindingContext="{x:Thanh trượt tham chiếu}"
            VerticalTextAlignment="Trung tâm"
            HorizontalTextAlignment="Trung tâm"
            widthRequest="400" BackgroundColor="XanhVàng"
            HeightRequest="100" TextColor="Đỏ"
           Text="{Giá trị liên kết, StringFormat='Góc quay {0:F0} độ'}"
            FontAttribut="Đậm"
            Kích thước phông chữ="26"
            HorizontalOptions="Trung tâm"
            VerticalOptions="CenterAndExpand" />
    </StackLayout>
</Trang nội dung>
```

Hãy khởi chạy ứng dụng phần mềm. Hãy chứng minh sự truyền dữ liệu từ điều khiển Thanh trượt đến điều khiển Nhãn