## Làm việc với Bố cục

Bố cục cho phép bạn sắp xếp và nhóm các điều khiển trong các trang trong giao diện người dùng của ứng dụng phần mềm. Việc chọn bố cục khi thiết kế giao diện người dùng yêu cầu người thiết kế phải biết cách định vị các điều khiển trong các trang được hiển thị khi ứng dụng phần mềm đang chạy.

Bài học này sẽ bao gồm các bố cục sau: 1. Bố cục dạng lưới

2. Bố cục khung

3. Bố cục ngăn xếp
4. Bố cục xem nội dung

5. Bố cục ScrollView

6. Bố cục tuyệt đối

7. Bố cục tương đối

8. Bố cục linh hoạt

Hãy xem xét các tính năng khi làm việc với bố cục Lưới, cho phép bạn đặt các điều khiển trong các ô của bảng được tạo bên trong bố cục. Bằng cách này, mỗi ô có thể chứa nhiều điều khiển hoặc bố cục.

Bố cục kiểu lưới có các thuộc tính sau:

1. Thuộc tính Column xác định số cột trong bảng. Đánh số cột bắt đầu từ 0. Giá trị mặc định là 0.

2. Thuộc tính ColumnDefinitions xác định các thuộc tính cho cột mới được tạo trong bảng.

3. Thuộc tính ColumnSpacing xác định khoảng cách giữa các cột các bảng. Giá trị mặc định của thuộc tính này là 6 đơn vị.

4. Thuộc tính ColumnSpan xác định số lượng cột được nối trong một dòng. Giá trị mặc định của thuộc tính này là 1.

5. Thuộc tính Row xác định số hàng trong bảng. Đánh số dòng bắt đầu từ 0. Giá trị mặc định của thuộc tính là 0.

6. Thuộc tính RowDefinitions xác định các thuộc tính cho hàng mới được tạo trong bảng.

7. Thuộc tính RowSpacing xác định khoảng cách giữa các hàng các bảng. Giá trị mặc định của thuộc tính này là 6 đơn vị.

8. Thuộc tính RowSpan xác định số lượng hàng được hợp nhất trong một cột. Giá trị mặc định của thuộc tính này là 1.

Để xem xét các tính năng khi làm việc với bố cục Lưới, hãy tạo dự án GridDemo cho ứng dụng phần mềm đa nền tảng bằng cách sử dụng mẫu “Rỗng” . Bạn cần vào cửa sổ Solution Explorer, chọn và mở file MainPage.xaml rồi gõ mã XAML bên dưới. Mã chương trình cung cấp cho việc hình thành bố cục kiểu Lưới, trong đó các điều khiển BoxView và Label được đặt ở cả ba hàng, mỗi hàng có 2 cột. Mỗi cột có

cùng chiều rộng. Hàng thứ ba kết hợp hai cột. Điều khiển BoxView trên hàng thứ ba ghi lại cả hai cột. Một số điều khiển có thể được đặt trong một ô.

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

    <Grid.RowDefinitions>
      <RowDefinition Height="2*" />
      <RowDefinition />
      <RowDefinition Height="100" /> </
    Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
      <ColumnDefinition />
      <ColumnDefinition /> </
    Grid.ColumnDefinitions>
    <BoxView Color="Orange" />
    <Label Text="Hàng 0
            Cột 0"
        HorizontalTextAlignment="Trung tâm"
        FontSize="20" TextColor="Đen"
        FontAttribut="Đậm" HorizontalOptions="Trung tâm"
        VerticalOptions="Center" />
    <BoxView Grid.Column="1" Color="Yellow" /> <Label
    Grid.Column="1"
        Text="Hàng 0
            Cột 1"
        HorizontalTextAlignment="Trung tâm"
        FontSize="20" TextColor="Đen"
        FontAttribut="Đậm"
        HorizontalOptions="Trung tâm"
        VerticalOptions="Center" />
    <BoxView Grid.Row="1" Color="Bisque" /> <Label
    Grid.Row="1"
        Text="Hàng 1
               Cột 0"
        HorizontalTextAlignment="Trung tâm"
        FontSize="20" TextColor="Đen"
        FontAttribut="Đậm"
        HorizontalOptions="Trung tâm"
        VerticalOptions="Center" />
    <BoxView Grid.Row="1"
         Grid.Column="1" Color="Aqua" /> <Nhãn
    Grid.Row="1"
        Lưới.Column="1"
```

```
                Text="Hàng 1
                Cột 1"
                HorizontalTextAlignment="Trung tâm"
                FontSize="20" TextColor="Đen"
                FontAttribut="Đậm"
                HorizontalOptions="Trung tâm"
                VerticalOptions="Trung tâm" />
            <BoxView Grid.Row="2" Grid.ColumnSpan="2"
                 Color="LightGray" />
            <Nhãn Grid.Row="2" Grid.ColumnSpan="2"
                Text="Hàng 2, Cột 0 và 1"
                HorizontalTextAlignment="Trung tâm"
                FontSize="20" TextColor="Đen"
                FontAttribut="Đậm"
                HorizontalOptions="Trung tâm"
                VerticalOptions="Trung tâm" />
        </Grid>
    </Trang nội dung>
```

Để minh họa cách sử dụng bố cục Lưới, trước tiên bạn phải xây dựng rồi chạy ứng dụng phần mềm để minh họa cách sử dụng bố cục Lưới. Trang hiển thị các điều khiển BoxView và Label nằm trong các ô của bảng được tạo bằng cách sử dụng bố cục Lưới.


Tiếp theo, chúng ta hãy chuyển sang nghiên cứu các tính năng làm việc với bố cục của các loại Frame và StackLayout. Bố cục khung được sử dụng để tạo đường viền xung quanh điều khiển có trong bố cục. Bố cục kiểu Khung có các thuộc tính sau:

1. Thuộc tính BorderColor xác định màu đường viền của bố cục kiểu Khung.
2. Thuộc tính CornerRadius xác định bán kính của góc bố cục.

3. Thuộc tính HasShadow xác định xem có bóng xung quanh bố cục hay không.

Bố cục StackLayout cho phép bạn đặt các điều khiển hoặc bố cục bên trong nó, theo chiều dọc bên dưới nhau hoặc theo chiều ngang cạnh nhau, tùy thuộc vào giá trị của thuộc tính Định hướng. Thuộc tính Spacing kiểm soát khoảng cách giữa các điều khiển trong bố cục và có giá trị mặc định là 6.

Để tìm hiểu cách làm việc với bố cục Frame và StackLayout, hãy tạo dự án StackLayoutDemo cho ứng dụng phần mềm đa nền tảng bằng cách sử dụng mẫu Trống. Bạn cần vào cửa sổ Solution Explorer, chọn và mở tệp MainPage.xaml, sau đó nhập mã chương trình bên dưới. Mã này hướng dẫn bạn cách tạo StackLayout có ba điều khiển Nhãn và sáu điều khiển Khung xếp chồng lên nhau theo chiều dọc. Mỗi bố cục Khung chứa một StackLayout bên trong, chứa điều khiển BoxView nằm ngang hiển

thị màu và điều khiển Nhãn hiển thị tên của các màu trong điều khiển BoxView. Các điều chỉnh đã được thực hiện đối với màu nền, màu sắc, cỡ chữ, kiểu viết và vị trí văn bản trong các điều khiển Nhãn cũng như vị trí của các điều khiển trong bố cục. Ngoài ra, bằng cách sử dụng thuộc tính BorderColor của bố cục Khung, chúng tôi đã điều chỉnh độ dày của đường viền xung quanh bố cục.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
         x:Class="StackLayoutDemo.MainPage"
         BackgroundColor="Bisque"
         Title="Bản trình diễn StackLayout">

   <Lề StackLayout="20">
      <Label Text="Bản trình diễn StackLayout"
           Kích thước phông chữ="40"
           FontAttribut="Đậm"
           TextColor="San hô"/>
      <Label Text="Màu cơ bản"
           Kích thước phông chữ="40"
           FontAttribut="Đậm"
           TextColor="Đen"/>
      <Frame BorderColor="Đen"
           Phần đệm="5">
        <StackLayout Orientation="Ngang"
                Khoảng cách="15">
          <BoxView Color="Đỏ" />
          <Văn bản nhãn="Đỏ"
                Kích thước phông chữ="30"
                TextColor="Đen"
```

```
            VerticalOptions="Trung tâm" />
    </StackLayout>
</Khung>
<Frame BorderColor="Đen"
       Phần đệm="5">
    <StackLayout Orientation="Ngang"
                 Khoảng cách="15">
        <BoxView Color="Vàng" />
        <Văn bản nhãn="Vàng"
               Kích thước phông chữ="30"
               TextColor="Đen"
               VerticalOptions="Trung tâm" />
    </StackLayout>
</Khung>
<Frame BorderColor="Đen"
       Phần đệm="5">
    <StackLayout Orientation="Ngang"
                 Khoảng cách="15">
        <BoxView Color="Blue" />
        <Văn bản nhãn="Màu xanh"
               Kích thước phông chữ="30"
               TextColor="Đen"
               VerticalOptions="Trung tâm" />
    </StackLayout>
</Khung>
<Label Text="Màu phụ"
       Kích thước phông chữ="40"
       FontAttribut="Đậm"
       TextColor="Đen"/>
<Frame BorderColor="Đen"
       Phần đệm="5">
    <StackLayout Orientation="Ngang"
                 Khoảng cách="15">
        <BoxView Color="Green" />
        <Văn bản nhãn="Xanh"
               Kích thước phông chữ="30"
               TextColor="Đen"
               VerticalOptions="Trung tâm" />
    </StackLayout>
</Khung>
<Frame BorderColor="Đen"
       Phần đệm="5">
    <StackLayout Orientation="Ngang"
                 Khoảng cách="15">
        <BoxView Color="Orange" />
```

```
                <Văn bản nhãn="Cam"
                    Kích thước phông chữ="30"
                    TextColor="Đen"
                    VerticalOptions="Trung tâm" />
            </StackLayout>
        </Khung>
        <Frame BorderColor="Đen"
            Phần đệm="5">
            <StackLayout Orientation="Ngang"
                Khoảng cách="15">
                <BoxView Color="Tím" />
                <Văn bản nhãn="Tím"
                    Kích thước phông chữ="30"
                    TextColor="Đen"
                    VerticalOptions="Trung tâm" />
            </StackLayout>
        </Khung>
    </StackLayout>
</Trang nội dung>
```

Để minh họa cách làm việc với bố cục Frame và StackLayout Trước tiên bạn cần xây dựng và sau đó chạy ứng dụng phần mềm.

Giao diện người dùng hiển thị ba điều khiển Nhãn và sáu điều khiển Khung bên trong StackLayout, mỗi điều khiển xác định một đường dọc theo đường viền của StackLayout bên trong bố cục Khung. Mỗi StackLayout bên trong bố cục Frame chứa các điều khiển BoxView và Label được bố trí theo chiều ngang.

Tiếp theo, chúng ta sẽ xem xét các chi tiết cụ thể khi làm việc với bố cục loại ContentView, chứa một phần tử con duy nhất và thường được dùng để tạo mẫu cho các điều khiển tùy chỉnh và có thể tái sử dụng.

Để làm việc với bố cục kiểu ContentView, hãy tạo một dự án ContViewDemo dành cho ứng dụng phần mềm đa nền tảng Xamarin.Forms sử dụng mẫu Trống. Tiếp theo, bằng cách sử dụng chuỗi hành động được mô tả trước đó, một đối tượng "Trang nội dung" có tên Thẻ sẽ được tạo. Tiếp theo, bạn mở tệp Card.xaml đã tạo và nhập mã XAML sau. Sử dụng mã XAML, bạn tạo mẫu kiểm soát bản đồ tùy chỉnh là một hình chữ nhật bo tròn có màu ở bên trái và văn bản ở bên phải.


Trong mã, bố cục ContentView chứa mẫu điều khiển tùy chỉnh và đặt giá trị thuộc tính x:Name thành giá trị này. Giá trị này có thể được sử dụng để truy cập điều khiển bằng Binding sẽ được tạo từ mẫu.

Mẫu điều khiển, được gọi là Thẻ, khai báo các thuộc tính sau:

thuộc tính CardTitle chỉ định văn bản được hiển thị trong điều khiển;

thuộc tính CBColor chỉ định màu đường viền của phần tử mẫu kiểm soát và màu sắc của đường phân cách;

thuộc tính TarColor chỉ định màu của hình vuông được hiển thị bởi điều khiển BoxView ở phía bên trái của điều khiển mẫu;

thuộc tính CardColor chỉ định màu nền của điều khiển mẫu;

thuộc tính ColText chỉ định màu của văn bản trong điều khiển mẫu.

Bố cục ContentView chứa bố cục Khung, là một hình chữ nhật có các cạnh được bo tròn, màu nền do một thuộc tính đặt CardColor và màu đường viền được chỉ định bởi thuộc tính CBColor. Ngoài ra, các khoảng lệch khỏi ranh giới bố cục được chỉ định và các giá trị thuộc tính cho sự hiện diện của bóng xung quanh bố cục cũng được chỉ định. Bên trong bố cục Khung là bố cục Lưới. Bên trong bố cục Lưới là một bảng có một hàng và ba cột. Cột đầu tiên chứa điều khiển Frame của điều khiển BoxView. Bằng cách đặt giá trị Margin, bạn có thể tạo đường viền xung quanh điều khiển BoxView. Màu mà điều khiển BoxView điền vào được chỉ định bởi thuộc tính TarColor. Toàn bộ cột thứ hai được chiếm bởi điều khiển BoxView, hoạt động như một dấu phân cách giữa các cột. Màu của điều khiển này được đặt bằng cách sử dụng

Thuộc tính CBColor. Ô thứ ba chứa điều khiển Nhãn hiển thị văn bản được đặt bằng thuộc tính CardTitle. Màu của văn bản trong tùy chọn kiểm soát Nhãn được đặt bởi thuộc tính ColText. Liên kết cho phép bạn liên kết các thuộc tính của bố cục Khung và các điều khiển BoxView và Nhãn với các thuộc tính

ThẻTitle, CBColor, TarColor, CardColor, ColText.

```
      <ContentView xmlns="http://xamarin.com/schemas/2014/forms"
            xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
            xmlns:d="http://xamarin.com/schemas/2014/forms/design"
            xmlns:mc="http://schemas.openxmlformats.org/markup-
                                                  khả năng tương thích/2006"
             x:Tên="cái này"
             x:Class="ContViewDemo.Card">
        <Frame BindingContext="{x:Tham khảo cái này}"
                 BackgroundColor="{Binding CardColor,
                                       FallbackValue='Vàng'}"
                 BorderColor="{Liên kết CBColor,
                                  FallbackValue='Đen'}"
            GócRadius="10"
            HasShadow="Đúng"
            Phần đệm="10"
            VerticalOptions="Trung tâm"
            HorizontalOptions="Trung tâm">
          <Lưới>
            <Grid.RowDefinitions>
              <Chiều cao định nghĩa hàng="75" />
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
              <Chiều rộng định nghĩa cột="75" />
              <Chiều rộng định nghĩa cột ="10" />
              <Chiều rộng định nghĩa cột="200" />
            </Grid.ColumnDefinitions>
            <Frame IsClippedToBounds="Đúng"
                BorderColor="{Binding CBColor,
                                FallbackValue='Black'}"
                Chiều caoYêu cầu="70"
                Chiều rộngYêu cầu="70"
                HasShadow="Sai"
                HorizontalOptions="Trung tâm"
                VerticalOptions="Trung tâm">
              <BoxView BackgroundColor ="{Binding TarColor,
                                           FallbackValue='Đen'}"
                Ký quỹ="-10"
                Chiều rộngYêu cầu="50"
                Chiều caoYêu cầu="50"/>
```

```
            </Khung>
            <BoxView Grid.Column="1"
                  BackgroundColor="{Binding CBColor,
                                         FallbackValue='Black'}"
                  Chiều rộngYêu cầu="10"
                  VerticalOptions="Điền" />
            <Nhãn Grid.Column="2"
                 Text="{Tiêu đề thẻ ràng buộc,
                                 FallbackValue='Tiêu đề thẻ'}"
                 TextColor="{Liên kết ColText,
                                        FallbackValue='Đỏ'}"
                 FontAttribut="Đậm"
                 Kích thước phông chữ="24"
                 VerticalTextAlignment="Trung tâm"
                 HorizontalTextAlignment="Trung tâm" />
         </Grid>
      </Khung>
   </ContentView>
```

Tiếp theo, bạn cần mở tệp Card.xaml.cs và nhập mã C Sharp được đưa ra bên dưới. Mã chương trình tạo các thuộc tính CardTitle, CBColor, TarColor, CardColor, ColText cho điều khiển Mẫu thẻ. Bằng cách sử dụng BindableProperty, bạn có thể liên kết các thuộc tính của điều khiển được tạo dựa trên mẫu Thẻ với các thuộc tính của mẫu này.

Bạn sử dụng các phương thức GetValue và SetValue để lấy và đặt giá trị của các thuộc tính có thể liên kết cho một điều khiển được tạo từ mẫu Thẻ.

```
sử dụng System.ComponentModel;
sử dụng Xamarin.Forms;
không gian tên ContViewDemo
{
   [DesignTimeVisible(true)]
   Thẻ lớp một phần công khai: ContentView
   {
      BindableProperty tĩnh công khai chỉ đọc
         CardTitleProperty = BindableProperty.Create(
                    nameof(CardTitle),
                    typeof(string),
                    typeof(Card));
      BindableProperty tĩnh công khai chỉ đọc
         CBColorProperty = BindableProperty.Create(
                     nameof(CBColor),
                     typeof(Color),
                     typeof(Card),
```

```
                Màu sắc.Đen);
BindableProperty tĩnh công khai chỉ đọc
   TarColorProperty =




BindableProperty.Create( nameof(TarColor), typeof(Color), typeof(Card), Color.Blac
   CardColorProperty =




BindableProperty.Create( nameof(CardColor), typeof(Color), typeof(Card), Color.Whi
   ColTextProperty =




BindableProperty.Create( nameof(ColText), typeof(Color), typeof(Card), Color.Red);
   get => (string)GetValue(Card.CardTitleProperty); set
   => SetValue(Card.CardTitleProperty, value);
}

Màu công khai CBColor {

   get => (Color)GetValue(Card.CBColorProperty); set =>
   SetValue(Card.CBColorProperty, value);

} Màu công khai TarColor
{
   get => (Color)GetValue(Card.TarColorProperty); set =>
   SetValue(Card.TarColorProperty, value);

} Màu thẻ công khai Màu {

   get => (Color)GetValue(Card.CardColorProperty); set =>
   SetValue(Card.CardColorProperty, value);

} ColText màu công khai
{
   get => (Color)GetValue(Card.ColTextProperty); set =>
   SetValue(Card.ColTextProperty, value);
}
```

```
        thẻ công cộng()
        {
          Khởi tạoComponent();
        }
      }
    }
```

Tiếp theo, chọn tệp MainPage.xaml trong hộp thoại Solution Explorer, mở tệp đó và nhập mã XAML sau. Trong mã chương trình cho một trang thuộc loại ContentPage, một liên kết đến các

điều khiển được thêm vào - không gian tên của điều khiển người dùng, nghĩa là đối với mẫu điều khiển, mã chương trình được cung cấp trong tệp Card.xaml.

MainPage chứa StackLayout chứa điều khiển Nhãn để hiển thị tiêu đề trang cũng như ba điều khiển tùy chỉnh là phiên bản của mẫu Thẻ.

Mỗi lần truy cập mẫu, mã chương trình sẽ chuyển các giá trị của thuộc tính bị ràng buộc CardTitle, CBColor, TarColor, CardColor, ColText vào mẫu. Các thuộc tính này được liên kết với mẫu trước đó trong mã chương trình được cung cấp trong tệp Card.xaml và Card.xaml.cs.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
       xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
       xmlns:d="http://xamarin.com/schemas/2014/forms/design"
       xmlns:mc="http://schemas.openxmlformats.org/markup-
                                                khả năng tương thích/2006"
        xmlns:controls="clr-namespace:ContViewDemo"
        x:Class="ContViewDemo.MainPage"
        Phần đệm="10, 50"
        BackgroundColor="Azure">
  <StackLayout>
    <Nhãn
       Ký quỹ="20"
       Text="Bản demo xem nội dung"
       FontSize="30" FontAttribut="Đậm"
       TextColor="Đen"
       HorizontalOptions="Trung tâm"
       HorizontalTextAlignment="Trung tâm"
       VerticalTextAlignment="Trung tâm"/>

    <controls:Card CardTitle="Đỏ"
             CBColor="Đen"
             TarColor="Đỏ"
```

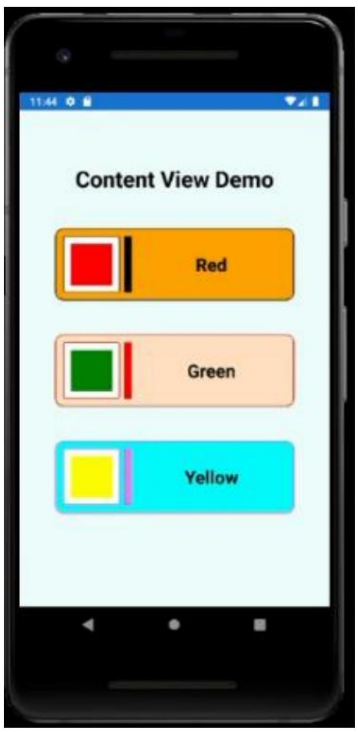
```
                CardColor="Cam"
                ColText="Đen"/>
        <controls:Card CardTitle="Xanh"
                CBColor="Đỏ"
                TarColor="Xanh"
                CardColor="Bisque"
                ColText="Đen"/>
        <controls:Card CardTitle="Vàng"
                CBColor="Tím"
                TarColor="Vàng"
                CardColor="Aqua"
                ColText="Đen"/>
    </StackLayout>
</Trang nội dung>
```

Để minh họa cách làm việc với bố cục ContentView, trước tiên bạn phải xây dựng và sau đó chạy ứng dụng phần mềm.

Sau khi khởi chạy, giao diện người dùng sẽ hiển thị ba điều khiển dựa trên mẫu Thẻ được triển khai bằng bố cục ContentView. Khi giao diện người dùng được xây dựng, mã sẽ truy cập vào mẫu điều khiển ba

lần với các giá trị thuộc tính có thể liên kết mới.

Kết quả là, ba
các trường hợp khác nhau của điều khiển mẫu.




Tiếp theo, hãy xem xét các tính năng khi làm việc với bố cục kiểu ScrollView, nhờ đó bạn có thể cuộn các điều khiển nằm bên trong bố cục này. Rất thường xuyên bên trong một bố cục như

ScrollView lưu trữ bố cục kiểu StackLayout. Trong trường hợp này, không thể hiển thị trên màn hình thiết bị tất cả các điều khiển nằm bên trong bố cục kiểu StackLayout. Để đặt hướng cuộn (ngang, dọc hoặc theo cả hai hướng), bạn phải đặt giá trị thích hợp cho thuộc tính Orientation của bố cục kiểu ScrollView.

Để xem chi tiết cụ thể khi làm việc với bố cục ScrollView, hãy tạo dự án ScrViewDemo cho ứng dụng phần mềm đa nền tảng bằng cách sử dụng mẫu Trống. Bạn cần vào hộp thoại Solution Explorer, chọn và mở tệp MainPage.xaml rồi nhập đoạn mã sau vào đó. Trong mã, bố cục ScrollView chứa bố cục StackLayout. StackLayout chứa bốn bố cục Khung, tạo ra các ranh giới của bố cục Lưới bên trong nó. Mỗi bố cục Khung có màu nền, màu đường viền, bán kính góc và sự hiện diện của bóng xung quanh bố cục. Bố cục lưới được đặt thành ba hàng và hai cột. Cột đầu tiên của hàng đầu tiên chứa biểu tượng hình tròn, hai màu được tạo bằng cách sử dụng bố cục Khung và điều khiển BoxView. Cột thứ hai của hàng đầu tiên chứa điều khiển Nhãn có văn bản cần hiển thị. Toàn bộ hàng thứ hai, trong đó các cột được kết hợp, được điều khiển BoxView chiếm giữ, hoạt động như một dấu phân cách cho hàng đầu tiên và hàng thứ ba. Hàng thứ ba, kết hợp hai cột, chứa điều khiển Nhãn cũng hiển thị văn bản.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
       xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
       xmlns:d="http://xamarin.com/schemas/2014/forms/design"
       xmlns:mc="http://schemas.openxmlformats.org/markup-
                                                  khả năng tương thích/2006"
        xmlns:controls="clr-namespace: ScrViewDemo "
        x:Class="ScrViewDemo.MainPage"
        Title="ScrollView Demo"
        Phần đệm="10">
  <Chế độ xem cuộn>
    <StackLayout>
      <Frame
        BackgroundColor="Cá hồi"
        BorderColor="Xanh" CornerRadius="5"
        HasShadow="True" Padding="8"
        VerticalOptions="Trung tâm"
        HorizontalOptions="Trung tâm">
        <Lưới>
          <Grid.RowDefinitions>
            <Chiều cao định nghĩa hàng="75" />
            <RowDefinition Chiều cao="4" />
            <RowDefinition Height="Tự động" />
          </Grid.RowDefinitions>
```

```
      <Grid.ColumnDefinitions>
        <ColumnDefinition width="75" />
        <ColumnDefinition width="200" /> </
      Grid.ColumnDefinitions> <Frame
      BorderColor="Black"
            BackgroundColor="Vàng"
            CornerRadius="38" HeightRequest="60"
            Chiều rộngYêu cầu="60"
            HorizontalOptions="Trung tâm"
            HasShadow="Sai"
            VerticalOptions="Trung tâm">
        <BoxView
            Lề="-10" CornerRadius="38"
            BackgroundColor="Violet"/> </
      Frame>
      <Label Grid.Column="1"
            Text="Alexander Ovechkin"
            FontAttribut="Đậm"
            FontSize="24" TextColor="Đen"
            VerticalTextAlignment="Trung tâm"
            HorizontalTextAlignment="Center" />
      <BoxView Grid.Row="1"
             Grid.ColumnSpan="2"
             BackgroundColor="Xanh"
             Chiều caoYêu cầu="2"
             HorizontalOptions="Fill" />
      <Nhãn Grid.Row="2" Grid.ColumnSpan="2"
            Text="Vận động viên khúc côn cầu.
                  Đội trưởng câu lạc bộ khúc côn cầu
                  NHL Washington Capitals
            FontAttribut="Đậm"
            FontSize="24" TextColor="Đen"
            HorizontalTextAlignment="Trung tâm"
            VerticalTextAlignment="Bắt đầu"
            VerticalOptions="Điền"
            HorizontalOptions="Điền" />
    </Grid>
  </Khung>
  <Khung
      BackgroundColor="XanhVàng"
      BorderColor="Xanh" CornerRadius="5"
      HasShadow="True" Padding="8"
      VerticalOptions="Trung tâm"
      HorizontalOptions="Trung tâm">
    <Lưới>
```

```
<Grid.RowDefinitions>
  <RowDefinition Height="75" />
  <RowDefinition Height="4" />
  <RowDefinition Height="Auto" /> </
Grid.RowDefinitions>
<Grid.ColumnDefinitions>
  <ColumnDefinition Height="75" />
  <ColumnDefinition width="200" /> </
Grid.ColumnDefinitions>
<Frame BorderColor="Black"
       BackgroundColor="Màu xanh"
       GócRadius="38"
       Chiều caoYêu cầu="60"
       Chiều rộngYêu cầu="60"
       HorizontalOptions="Trung tâm"
       HasShadow="Sai"
       VerticalOptions="Trung tâm">
  <BoxView
      Ký quỹ="-10"
      GócRadius="38"
      BackgroundColor="Yellow"/> </
Frame>
<Label Grid.Column="1"
       Text="Pavel Datsyuk"
       FontAttribut="Đậm"
       Kích thước phông chữ="24"
       TextColor="Đen"
       VerticalTextAlignment="Trung tâm"
       HorizontalTextAlignment="Center" />
<BoxView Grid.Row="1"
        Grid.ColumnSpan="2"
        BackgroundColor="Màu xanh"
        Chiều caoYêu cầu="2"
        HorizontalOptions="Fill" />
<Nhãn Grid.Row="2"
       Grid.ColumnSpan="2"
       Text="Cầu thủ khúc côn cầu. Đội trưởng câu lạc bộ khúc côn
             cầu KHL Avtomobileist"
       FontAttribut="Đậm"
       Kích thước phông chữ="24"
       TextColor="Đen"
       HorizontalTextAlignment="Trung tâm"
       VerticalTextAlignment="Bắt đầu"
       VerticalOptions="Điền"
       HorizontalOptions="Điền" />
```

```
    </Grid>
</Frame>
<Frame
      BackgroundColor="Silver"
      BorderColor="Đỏ"
      GócRadius="5"
      HasShadow="Đúng"
      Phần đệm="8"
      VerticalOptions="Trung tâm"
      HorizontalOptions="Center">
    <Grid>
      <Grid.RowDefinitions>
        <RowDefinitions Height="75" />
        <RowDefinition Height="4" />
        <RowDefinition Height="Auto" /> </
      Grid.RowDefinitions>
      <Grid. Định nghĩa cột>
        <ColumnDefinition width="75" />
        <ColumnDefinition width="200" /> </
      Grid.ColumnDefinitions> <Frame
      BorderColor="Black"
            BackgroundColor="Ô liu"
            GócRadius="38"
            Chiều caoYêu cầu="60"
            Chiều rộngYêu cầu="60"
            HorizontalOptions="Trung tâm"
            HasShadow="Sai"
            VerticalOptions="Trung tâm">
        <BoxView
            Ký quỹ="-10"
            GócRadius="38"
            BackgroundColor="Orange"/> </
      Frame>
      <Label Grid.Column="1"
            Text="Sergey Mozyakin"
            FontAttribut="Đậm"
            Kích thước phông chữ="24"
            TextColor="Đen"
            VerticalTextAlignment="Trung tâm"
            HorizontalTextAlignment="Center" />
      <BoxView Grid.Row="1"
             Grid.ColumnSpan="2"
             BackgroundColor="Đỏ"
             Chiều caoYêu cầu="2"
             HorizontalOptions="Điền" />
```

```
      <Nhãn Grid.Row="2"
             Grid.ColumnSpan="2"
             Text="Cầu thủ khúc côn
              cầu. Cầu thủ ghi bàn xuất sắc nhất trong
              lịch sử khúc côn cầu Nga"
             FontAttribut="Đậm"
             Kích thước phông chữ="24"
             TextColor="Đen"
             HorizontalTextAlignment="Trung tâm"
             VerticalTextAlignment="Bắt đầu"
             VerticalOptions="Điền"
             HorizontalOptions="Fill" /> </
    Grid> </
  Frame>
  <Frame
       BackgroundColor="Aqua"
       BorderColor="Nâu"
       GócRadius="5"
       HasShadow="Đúng"
       Phần đệm="8"
       VerticalOptions="Trung tâm"
       HorizontalOptions="Center">
    <Grid>
      <Grid.RowDefinitions>
        <RowDefinition Height="75" />
        <RowDefinition Height="4" />
        <RowDefinition Height="Auto" />
      </Grid.RowDefinitions>
      <Grid.ColumnDefinitions>
        <ColumnDefinition width="75" />
        <ColumnDefinition width="200" /> </
      Grid.ColumnDefinitions> <Frame
      BorderColor="Black"
             BackgroundColor="Đỏ"
             GócRadius="38"
             Chiều caoYêu cầu="60"
             Chiều rộngYêu cầu="60"
             HorizontalOptions="Trung tâm"
             HasShadow="Sai"
             VerticalOptions="Trung tâm">
        <BoxView
            Ký quỹ="-10"
            GócRadius="38"
            BackgroundColor="Vàng"/>
      </Khung>
```

```
                    <Nhãn Grid.Column="1"
                           Text="Vladislav Tretyak"
                           FontAttribut="Đậm"
                           Kích thước phông chữ="24"
                           TextColor="Đen"
                           VerticalTextAlignment="Trung tâm"
                           HorizontalTextAlignment="Trung tâm" />
                    <BoxView Grid.Row="1"
                            Grid.ColumnSpan="2"
                            BackgroundColor="Đỏ"
                            Chiều caoYêu cầu="2"
                            HorizontalOptions="Điền" />
                    <Nhãn Grid.Row="2"
                           Grid.ColumnSpan="2"
                           Text="Cầu thủ khúc côn
                            cầu. Thủ môn xuất sắc nhất trong lịch
                            sử khúc côn cầu thế giới"
                           FontAttribut="Đậm"
                           Kích thước phông chữ="24"
                           TextColor="Đen"
                           HorizontalTextAlignment="Trung tâm"
                           VerticalTextAlignment="Bắt đầu"
                           VerticalOptions="Điền"
                           HorizontalOptions="Điền" />
                </Grid>
            </Khung>
        </StackLayout>
    </ScrollView>
</Trang nội dung>
```

Để minh họa cách làm việc với bố cục ScrollView, bạn cần xây dựng và chạy một ứng dụng phần mềm. Khi khởi chạy ứng dụng phần mềm, bạn có thể thấy cả bốn bố cục Loại khung đều không vừa với màn hình của thiết bị di động. Bố cục ScrollView với tính năng cuộn cho phép bạn sắp xếp việc xem thông tin trong tất cả các bố cục loại Khung.


Phần sau đây thảo luận về các tính năng khi làm việc với bố cục Tuyệt đối, được thiết kế để đặt các điều khiển bên trong bố cục biểu thị các giá trị tuyệt đối của tọa độ và kích thước. Ngoài ra, để đặt các điều khiển bên trong bố cục, bạn có thể chỉ định các giá trị tỷ lệ thuận với giá trị của các tham số đặc trưng cho kích thước và vị trí của bố cục. Bố cục Tuyệt đối có các thuộc tính sau:

1. Thuộc tính LayoutBounds xác định vị trí và kích thước của điều khiển được đặt bên trong bố cục. Giá trị mặc định của thuộc tính này là (0, 0, AutoSize, AutoSize).

2. Thuộc tính LayoutFlags xác định việc sử dụng các giá trị thuộc tính đặc trưng cho kích thước và vị trí của bố cục để có được các tỷ lệ khác nhau khi đặt các điều khiển trong bố cục. Thuộc tính này có thể nhận các giá trị sau:

Không có gì chỉ định rằng các giá trị chiều rộng và chiều cao của bố cục sẽ được hiểu là giá trị tuyệt đối (đây là giá trị mặc định của thuộc tính này);

giá trị XProportal chỉ ra rằng giá trị của x sẽ là được hiểu là tỷ lệ thuận;

giá trị YProportal chỉ ra rằng giá trị y sẽ là được hiểu là tỷ lệ thuận;

giá trị Tỷ lệ chiều rộng cho biết rằng giá trị chiều rộng sẽ là được hiểu là tỷ lệ thuận;

giá trị HeightProportal cho biết giá trị chiều cao sẽ là được hiểu là tỷ lệ thuận;

giá trị Tỷ lệ vị trí cho biết giá trị x và y sẽ là được hiểu là tỷ lệ thuận;

giá trị SizeProportal chỉ định rằng các giá trị chiều rộng và chiều cao sẽ là được hiểu là tỷ lệ thuận;

giá trị All chỉ ra rằng tất cả các giá trị (x, y, width, Height) sẽ được hiểu là tỷ lệ.

Để làm việc với bố cục kiểu Tuyệt đối, hãy tạo một dự án AbsLayoutDemo cho ứng dụng phần mềm đa nền tảng sử dụng mẫu Trống. Bạn cần vào hộp thoại Solution Explorer, chọn và mở tệp MainPage.xaml rồi nhập đoạn mã sau.

Nửa đầu của mã bao gồm việc tạo biểu tượng từ bốn đường giao nhau, được tạo bằng bốn điều khiển BoxView và điều khiển Nhãn. Vị trí của mỗi điều khiển BoxView được xác định bằng cách sử dụng hai giá trị tuyệt đối đầu tiên được chỉ định làm tham số trong thuộc tính LayoutBounds. Kích thước của mỗi điều khiển được xác định bằng cách sử dụng tham số thứ ba và thứ tư trong thuộc tính LayoutBounds.

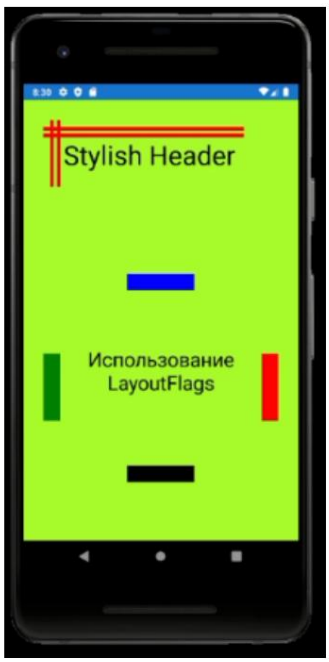
Trong nửa sau của mã, bốn điều khiển BoxView và một điều khiển Nhãn được đặt trong bố cục bằng cách sử dụng các giá trị tỷ lệ (thuộc tính LayoutFlags được đặt thành LocationProportal) và các điều khiển được định kích thước bằng cách sử dụng các giá trị tuyệt đối. Do đó, hai giá trị đầu tiên được chỉ định trong thuộc tính LayoutBounds cho điều khiển BoxView và Label xác định bố cục của điều khiển bằng cách sử dụng các giá trị tỷ lệ.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
       xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
       x:Class="AbsLayoutDemo.MainPage"
        Title="Bản trình diễn AbsolutLayout"
        Phần đệm="10"
        BackgroundColor="XanhVàng">
  <Lề tuyệt đốiLayout="20">
    <BoxView Color="Đỏ"
         Tuyệt đốiLayout.LayoutBounds="0, 10, 300, 5" />
    <BoxView Color="Đỏ"
         Tuyệt đốiLayout.LayoutBounds="0, 20, 300, 5" />
    <BoxView Color="Đỏ"
         Tuyệt đốiLayout.LayoutBounds="10, 0, 5, 100" />
    <BoxView Color="Đỏ"
         Tuyệt đốiLayout.LayoutBounds="20, 0, 5, 100" />
    <Label Text="Tiêu đề đầy phong cách"
        Kích thước phông chữ="40"
        TextColor="Đen"
```

```
            Tuyệt đốiLayout.LayoutBounds="30, 25" />
        <BoxView Color="Xanh"
            Tuyệt đốiLayout.LayoutBounds="0,5,0,4,100,25"
            Tuyệt đốiLayout.LayoutFlags="PositionProportal" />
        <BoxView Color="Xanh"
            Tuyệt đốiLayout.LayoutBounds="0,0.7,25,100"
            Tuyệt đốiLayout.LayoutFlags="PositionProportal" />
        <BoxView Color="Đỏ"
            Tuyệt đốiLayout.LayoutBounds="1,0.7,25,100"
            Tuyệt đốiLayout.LayoutFlags="PositionProportal" />
        <BoxView Color="Đen"
            Tuyệt đốiLayout.LayoutBounds="0,5,0,9,100,25"
            Tuyệt đốiLayout.LayoutFlags="PositionProportal" />
        <Label Text="Sử dụng cờ bố cục"
            Kích thước phông chữ="30"
            TextColor="Đen"
            HorizontalOptions="Trung tâm"
            HorizontalTextAlignment="Trung tâm"
            Tuyệt đốiLayout.LayoutBounds="0,5,0,65,300,80"
            Tuyệt đốiLayout.LayoutFlags="PositionProportal" />
    </AbsoluteLayout>
</Trang nội dung>
```

Để minh họa cách làm việc với bố cục thuộc loại AbsolulLayout, một ứng dụng phần mềm sẽ được khởi chạy. Sau khi khởi chạy ứng dụng phần mềm, logo kèm dòng chữ hiển thị ở đầu giao diện người dùng

"Tiêu đề sành điệu" sử dụng vị trí tuyệt đối của các điều khiển và bốn hình chữ nhật có nhãn "Sử dụng cờ bố cục" ở cuối giao diện người dùng được lấy bằng cách sử dụng vị trí tỷ lệ.



Tiếp theo, chúng ta xem xét cách làm việc với bố cục loại InteractiveLayout, được dùng để đặt các điều khiển trong bố cục bằng cách sử dụng các giá trị tương đối hoặc tuyệt đối của các tham số đặc trưng cho kích thước của các điều khiển. Trong trường hợp này, để có được các giá trị tham số, các giá trị đặc trưng cho kích thước của bố cục.

liên quan đến được sử dụng

Bố cục thuộc loại InteractiveLayout có các thuộc tính sau để đặt các giá trị đặc trưng cho vị trí và kích thước của các điều khiển:

thuộc tính XConstraint chỉ định vị trí tương đối dọc theo trục X cho điều khiển bên trong bố cục, có tính đến kích thước của bố cục;

thuộc tính YConstraint chỉ định vị trí tương đối dọc theo trục Y cho điều khiển bên trong bố cục, có tính đến kích thước của bố cục;

thuộc tính widthConstraint chỉ định chiều rộng tương đối cho điều khiển nằm trong bố cục, có tính đến kích thước của bố cục;

thuộc tính HeightConstraint chỉ định giá trị độ cao tương đối cho điều khiển nằm bên trong bố cục, có tính đến kích thước của bố cục;

thuộc tính BoundsConstraint chỉ định vị trí và kích thước tương đối cho điều khiển nằm trong bố cục, có tính đến kích thước

cách trình bày.

XAML sử dụng tiện ích mở rộng đánh dấu ConstraintExpression để sắp xếp bố cục tuyệt đối của các điều khiển. Tiện ích mở rộng đánh dấu này được sử dụng để liên kết vị trí và kích thước của một phần tử với các giá trị thuộc tính đặc trưng cho bố cục hoặc điều khiển InteractiveLayout mà nó được định vị tương đối. Trong trường hợp này, lớp ConstraintExpression có các thuộc tính sau:

Thuộc tính Constant xác định thành phần hằng số để biểu thị vị trí hoặc kích thước của điều khiển;

thuộc tính ElementName xác định tên của thành phần giao diện người dùng tương ứng với thành phần giao diện người dùng khác được đặt;

thuộc tính Yếu tố xác định giá trị của hệ số tỷ lệ khi xác định vị trí và kích thước của điều khiển được đặt so với điều khiển được chỉ định trong thuộc tính

ElementName (theo mặc định thuộc tính này có giá trị là 1);

thuộc tính Thuộc tính chỉ định tên của thuộc tính trong phần tử Nguồn, được sử dụng khi đặt điều khiển;

thuộc tính ConstraintType xác định kiểu sắp xếp tương đối phần tử điều khiển.

Thuộc tính ConstraintType có thể nhận các giá trị sau:

giá trịRelativeToParent cho biết rằng điều khiển được đặt tương ứng với phần tử “cha” chứa điều khiển đó;

giá trị InteractiveToView chỉ ra rằng – điều khiển được đặt tương đối với một số điều khiển khác, không phải điều khiển gốc;

Giá trị không đổi chỉ ra rằng có một thành phần không đổi đối với việc lựa chọn vị trí hoặc kích thước điều khiển.

Để xem xét các tính năng khi làm việc với bố cục thuộc loại Bố cục tương đối, hãy tạo dự án RelatLayoutDemo cho ứng dụng phần mềm đa nền tảng bằng cách sử dụng mẫu "Rỗng" . Sau khi tạo xong project các bạn cần vào cửa sổ Solution Explorer, chọn và mở file
MainPage.xaml rồi nhập mã XAML bên dưới.

Trong phần đầu tiên của mã chương trình, sử dụng các giá trị tuyệt đối của các tham số đặc trưng cho vị trí và kích thước của các điều khiển, các đường ngang của khung kép được hình thành dọc theo ranh giới trang thông qua việc sử dụng bốn điều khiển BoxView.

Trong phần thứ hai của mã chương trình, bằng cách sử dụng các giá trị tuyệt đối của các tham số đặc trưng cho vị trí và kích thước của các điều khiển, một điều khiển BoxView với các góc được làm tròn sẽ được tạo ra, cũng như điều khiển Nhãn nằm ở trên cùng với dòng chữ "AbsolutLayoutDemo" .

Vị trí của từng điều khiển BoxView cũng như điều khiển Nhãn được xác định bằng cách sử dụng các giá trị tuyệt đối được chỉ định trong thuộc tính XConstraint và YConstraint. Kích thước của mỗi điều khiển BoxView được xác định bằng cách sử dụng các giá trị tuyệt đối được chỉ định trong thuộc tính HeightConstraint và HeightConstraint.

Trong phần thứ ba của mã chương trình, sử dụng các giá trị tương đối của các tham số đặc trưng cho vị trí và kích thước của các điều khiển, các đường thẳng đứng của khung kép được hình thành dọc theo ranh giới của trang. Các dòng được hình thành bằng cách sử dụng bốn điều khiển BoxView. Trong trường hợp này, việc đặt các giá trị tương đối đặc trưng cho vị trí và kích thước của các điều khiển BoxView được thực hiện bằng cách sử dụng tiện ích mở rộng đánh dấu ConstraintExpression.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
         x:Class="RelatLayoutDemo.MainPage"
         BackgroundColor="WhiteSmoke">
   <RelativeLayout Margin="20">
      <BoxView Color="Cam"
            Tương đốiLayout.XConstraint="0"
            Tương đốiLayout.YConstraint="10"
            Tương đốiLayout.WidthConstraint="400"
            Tương đốiLayout.HeightConstraint="10" />
      <BoxView Color="Cam"
            Tương đốiLayout.XConstraint="0"
            Tương đốiLayout.YConstraint="40"
            Tương đốiLayout.WidthConstraint="400"
```

```
      Tương đốiLayout.HeightConstraint="10" />
<BoxView Color="Cam"
      Tương đốiLayout.XConstraint="0"
      Tương đốiLayout.YConstraint="520"
      Tương đốiLayout.WidthConstraint="400"
      Tương đốiLayout.HeightConstraint="10" />
<BoxView Color="Cam"
      Tương đốiLayout.XConstraint="0"
      Tương đốiLayout.YConstraint="550"
      Tương đốiLayout.WidthConstraint="400"
      Tương đốiLayout.HeightConstraint="10" />
<BoxView Color="Vàng"
      Tương đốiLayout.XConstraint="80"
      Tương đốiLayout.YConstraint="150"
      Tương đốiLayout.WidthConstraint="220"
      Tương đốiLayout.HeightConstraint="250"
      CornerRadius="90"/>
<Label Text="RelativeLayoutDemo"
    HorizontalTextAlignment="Trung tâm"
    FontSize="30" TextColor="Đen"
    FontAttribut="Đậm"
    Tương đốiLayout.XConstraint="5"
    Tương đốiLayout.YConstraint="250" />
<BoxView Color="Cam"
      Tương đốiLayout.XConstraint=
      "{ConstraintExpression
         Loại=RelativeToParent,
         Thuộc tính=Chiều
         rộng, Hằng số=-360}"
      Tương đốiLayout.YConstraint=
      "{ConstraintExpression
         Loại=RelativeToParent,
         Thuộc tính=Chiều
         cao, Hằng số=-630}"
      Tương đốiLayout.WidthConstraint=
      "{ConstraintExpression
         Loại=RelativeToParent,
         Thuộc tính=Chiều rộng,
         Hằng số=-360}"
      Tương đốiLayout.HeightConstraint=
      "{ConstraintExpression
         Loại=RelativeToParent,
         Thuộc tính=Chiều cao,
         Hằng số=-30}"/>
<BoxView Color="Cam"
```

```
        Tương đốiLayout.XConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều
           rộng, Hằng số=-340}"
        Tương đốiLayout.YConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều cao,
           Hằng số=-630}"
        Tương đốiLayout.WidthConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều
           rộng, Hằng số=-360}"
        Tương đốiLayout.HeightConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều
           cao, Hằng số=-30}"/>
    <BoxView Color="Cam"
        Tương đốiLayout.XConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều
           rộng, Hằng số=-20}"
        Tương đốiLayout.YConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều cao,
           Hằng số=-630}"
        Tương đốiLayout.WidthConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều
           rộng, Hằng số=-360}"
        Tương đốiLayout.HeightConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
           Thuộc tính=Chiều
           cao, Hằng số=-30}"/>
    <BoxView Color="Cam"
        Tương đốiLayout.XConstraint=
        "{ConstraintExpression
           Loại=RelativeToParent,
```

```
                    Thuộc tính=Chiều
                    rộng, Hằng số=-40}"
                Tương đốiLayout.YConstraint=
                "{ConstraintExpression
                    Loại=RelativeToParent,
                    Thuộc tính=Chiều cao,
                    Hằng số=-630}"
                Tương đốiLayout.WidthConstraint=
                "{ConstraintExpression
                    Loại=RelativeToParent,
                    Thuộc tính=Chiều rộng,
                    Hằng số=-360}"
                Tương đốiLayout.HeightConstraint=
                "{ConstraintExpression
                    Loại=RelativeToParent, Thuộc
                    tính=Chiều cao, Hằng
                    số=-30}"/>
        </RelativeLayout>
    </Trang nội dung>
```

Để minh họa cách làm việc với bố cục kiểu InteractiveLayout, bạn cần xây dựng giải pháp và sau đó, nếu không có lỗi, hãy chạy ứng dụng phần mềm. Vị trí và kích thước của bốn đường khung ngang được lấy bằng cách sử dụng các giá trị tuyệt đối của các thuộc tính đặc trưng cho các điều khiển BoxView. Vị trí và kích thước của bốn đường khung dọc được lấy bằng cách sử dụng các giá trị thuộc tính tương đối của các điều khiển BoxView, phụ thuộc vào các giá trị của thuộc tính bố cục InteractiveLayout. Vị trí và kích thước cho các điều khiển BoxView và Label xuất hiện ở giữa trang được lấy bằng cách sử dụng các giá trị tuyệt đối của các thuộc tính đặc trưng cho các điều khiển BoxView và Label.

Tiếp theo, chúng ta chuyển sang nghiên cứu các tính năng khi làm việc với bố cục thuộc loại FlexLayout, giống như bố cục của loại StackLayout, có thể sắp xếp việc sắp xếp các điều khiển mà nó chứa theo chiều ngang và chiều dọc. Đồng thời, bố cục kiểu FlexLayout cũng có thể sắp xếp vị trí của các điều khiển nếu chúng không vừa với một hàng (cột). Trong trường hợp này, kích thước, vị trí và căn chỉnh của các điều khiển được kiểm soát. Ngoài ra, bố cục kiểu FlexLayout có thể điều chỉnh việc hiển thị các điều khiển cho phù hợp với các kích thước màn hình khác nhau.

FlexLayout có các thuộc tính sau:

1. Thuộc tính Direction xác định hướng đặt các điều khiển trong bố cục. Giá trị thuộc tính mặc định là Hàng, có nghĩa là các điều khiển được sắp xếp theo chiều ngang (theo hàng) trong bố cục. Trong trường hợp này, trục chính là trục hoành và trục phụ là trục tung. Việc đặt thuộc tính thành Cột sẽ khiến các điều khiển được sắp xếp theo chiều dọc (theo cột) trong bố cục. Nếu các điều khiển được sắp xếp theo cột thì trục chính của bố cục là dọc và trục phụ là

trục ngang.

Vì vậy thuộc tính Direction có thể nhận những thông tin sau
giá trị:

Giá trị cột chỉ định vị trí của các điều khiển ở trên cùng
xuống theo chiều dọc bên trong bố cục;

giá trị ColumnReverse chỉ định vị trí của các điều khiển
từ dưới lên trên theo chiều dọc bên trong bố cục;

giá trị Hàng chỉ định cách sắp xếp theo chiều ngang của các điều khiển từ trái sang phải trong bố cục;

giá trị RowReverse chỉ định vị trí của các điều khiển
từ phải sang trái theo chiều ngang bên trong bố cục.

2. Thuộc tính AlignItems cho biết thứ tự sắp xếp các phần tử. Thuộc tính AlignItems có thể nhận các giá trị sau:

giá trị Kéo dài đặt điều khiển được kéo dài theo chiều dọc hoặc chiều ngang tùy theo hướng của trục nhỏ (giá trị này được đặt theo mặc định);

giá trị Center đặt vị trí của điều khiển ở giữa
so với trục nhỏ;

giá trị Bắt đầu chỉ định vị trí của điều khiển ở bên trái (ngang) hoặc trên cùng (dọc) tùy thuộc vào hướng của trục nhỏ;

giá trị Kết thúc chỉ định vị trí của điều khiển ở bên phải (ngang) hoặc dưới cùng (dọc) tùy thuộc vào hướng của trục nhỏ.

3. Thuộc tính AlignContent tương tự như thuộc tính AlignItems nhưng được dùng để làm việc với các hàng hoặc cột nói chung và không hoạt động

với các điều khiển riêng biệt. Thuộc tính này có thể nhận các giá trị sau:

giá trị Center chỉ định rằng nhóm hàng sẽ ở giữa bố cục;

giá trị End chỉ định rằng nhóm hàng sẽ được đặt ở cuối phần tử cha;

SpaceAround chỉ định rằng sẽ có khoảng cách bằng nhau giữa các hàng và bằng một nửa khoảng cách giữa các hàng trên cùng và dưới cùng cũng như các cạnh trên và dưới của phần tử cha;

giá trị SpaceBetween chỉ định rằng các dòng trên cùng và dưới cùng sẽ lần lượt được định vị dọc theo các cạnh trên và dưới của phần tử cha và các dòng còn lại sẽ cách đều nhau;

giá trị SpaceEvenly chỉ ra rằng sẽ có cùng một khoảng cách giữa tất cả các dòng và cùng một khoảng trắng trước dòng đầu tiên và sau dòng cuối cùng;

giá trị Bắt đầu chỉ định rằng nhóm hàng sẽ được đặt ở đầu phần tử cha;

giá trị Kéo dài cho biết nhóm hàng sẽ được kéo dài ngay từ đầu
đến cuối phần tử cha.

4. Thuộc tính JustifyContent cho biết thứ tự hiển thị các phần tử.
trục chính. Thuộc tính JustifyContent có thể có các giá trị sau:

giá trị Bắt đầu tương ứng với vị trí của điều khiển ở bên trái (ngang) hoặc trên cùng (dọc) tùy theo hướng của trục chính (giá trị này được đặt theo mặc định);

giá trị Cuối tương ứng với vị trí của điều khiển ở bên phải (ngang) hoặc dưới cùng (dọc), tùy thuộc vào hướng của trục chính;

giá trị Center tương ứng với vị trí của điều khiển theo chiều ngang hoặc chiều dọc, tùy thuộc vào hướng của trục chính;

giá trị SpaceBetween đặt khoảng cách giữa các điều khiển;

giá trị SpaceAround đặt khoảng trắng xung quanh mỗi điều khiển;

Giá trị SpaceEvenly - đặt khoảng cách bằng nhau giữa các điều khiển và trước phần tử đầu tiên và cuối cùng trong một hàng hoặc phía trên phần tử đầu tiên và sau phần tử cuối cùng trong một cột.

5. Thuộc tính Wrap khiến điều khiển được ngắt dòng sang hàng hoặc cột tiếp theo (tùy thuộc vào bố cục của các điều khiển) nếu nó không vừa với hàng hoặc cột. Thuộc tính có thể nhận các giá trị sau:

giá trị NoWrap (tùy chọn mặc định) tương ứng với việc không bao bọc điều khiển vào đầu hàng hoặc cột tiếp theo (điều này làm giảm kích thước của các điều khiển để chúng vừa với hàng hoặc cột);

giá trị Gói tương ứng với việc điều khiển có được gói trong đầu hàng hoặc cột tiếp theo;

giá trị Reverse (“wrap-reverse” trong XAML) tương ứng với sự hiện diện di chuyển điều khiển đến cuối hàng hoặc cột tiếp theo;

6. Thuộc tính Order cho phép bạn thay đổi thứ tự đặt các điều khiển trong bố cục. Thông thường, các điều khiển được đặt trong bố cục theo thứ tự chúng được chỉ định trong mã bằng thuộc tính Children. Giá trị mặc định của thuộc tính là 0. Nếu thuộc tính Thứ tự của điều khiển đầu tiên trong bố cục được đặt thành giá trị nhỏ hơn các điều khiển khác trong bố cục thì điều khiển đó sẽ xuất hiện dưới dạng thành phần đầu tiên trong hàng hoặc cột. Tương tự, một điều khiển sẽ xuất hiện cuối cùng trong một hàng hoặc cột nếu thuộc tính Thứ tự của điều khiển đó được đặt thành giá trị lớn hơn giá trị của các điều khiển khác.

7. Thuộc tính Cơ sở chỉ định chiều rộng của các điều khiển khi chúng được đặt theo chiều ngang hoặc chiều cao của các điều khiển khi chúng được đặt theo chiều dọc. Giá trị thuộc tính có thể được chỉ định theo giá trị tuyệt đối hoặc dưới dạng phần trăm của kích thước bố cục. Giá trị mặc định của thuộc tính Cơ sở là FlexBasis.Auto, có nghĩa là chiều rộng hoặc chiều cao được chỉ định trước đó của điều khiển sẽ được sử dụng.

8. Thuộc tính Grow chỉ định cách phân bổ không gian giữa các điều khiển. Thuộc tính Grow có giá trị mặc định là 0. Nếu thuộc tính được đặt thành giá trị dương, khoảng trống trên trục chính sẽ được phân bổ cho điều khiển và các điều khiển khác có thuộc tính Grow được đặt thành giá trị dương. Không gian dọc theo trục chính sẽ được phân bổ tương ứng với các giá trị thuộc tính mà các điều khiển có.

9. Thuộc tính Shrink chỉ định phần tử con nào được ưu tiên khi hiển thị kích thước đầy đủ của chúng. Giá trị mặc định của thuộc tính là trừ 1, nhưng giá trị của thuộc tính phải lớn hơn hoặc bằng 0. Thuộc tính Shrink được bao gồm nếu giá trị thuộc tính Wrap là NoWrap và tổng chiều rộng của hàng hoặc cột chứa các điều khiển là lớn hơn chiều rộng hoặc chiều cao của FlexLayout. Khi thuộc tính được đặt thành 0, kích thước đầy đủ của điều khiển sẽ được hiển thị trong bố cục. Nếu giá trị thuộc tính lớn hơn 0, kích thước của điều khiển sẽ bị nén khi chúng hiển thị trên màn hình thiết bị.

Để xem lại các chi tiết cụ thể khi làm việc với bố cục FlexDemo, hãy tạo một dự án cho ứng dụng phần mềm đa nền tảng Xamarin.Forms bằng mẫu Trống. Trước tiên, bạn cần vào hộp thoại Solution Explorer và sử dụng trình tự đã thảo luận trước đó để tạo lớp EnumPicker. Tiếp theo, bạn cần mở tệp EnumPicker.cs đã tạo và nhập mã C Sharp được đưa ra bên dưới.

Mã này tạo một mẫu điều khiển EnumPicker được tham chiếu bằng mã trong tệp MainPage.xaml (sẽ được tạo sau). Điều khiển EnumPicker là con của điều khiển Picker, được sử dụng để chọn một giá trị từ danh sách.

Thuộc tính EnumType được tạo và tham chiếu bằng mã trong tệp MainPage.xaml. Mã chương trình mô tả cách làm việc với thuộc tính
EnumType khi giá trị trong điều khiển EnumPicker mà người dùng hiện đang làm việc thay đổi.

Các thuộc tính oldValue và newValue của điều khiển EnumPicker được liên kết với thuộc tính bố cục FlexLayout được chuyển qua các giá trị cũ và mới được chọn thu được từ một trong các điều khiển EnumPicker (các phần tử này được xác định trong tệp MainPage.xaml và một trong số chúng người dùng sẽ đang làm việc tại thời điểm hiện tại).

Nguồn dữ liệu chứa các giá trị của thuộc tính oldValue và newValue được chỉ định bằng cách sử dụng thuộc tính ItemsSource của đối tượng bộ chọn. Đối tượng bộ chọn nhận dữ liệu từ điều khiển Bộ chọn được xác định trong tệp MainPage.xaml mà người dùng hiện đang làm việc.

Mã này sử dụng các câu lệnh có điều kiện để kiểm tra các giá trị cũ và mới nhận được từ các điều khiển EnumPicker được cung cấp trong tệp MainPage.xaml nếu giá trị của thuộc tính oldValue và newValue thay đổi. Thuộc tính EnumType cũng được khai báo trong mã chương trình.

```
sử dụng Hệ thống;
sử dụng System.Reflection;
sử dụng Xamarin.Forms;
không gian tên FlexDemo
{ lớp EnumPicker: Bộ chọn
   {
   tĩnh công khai chỉ đọc BindableProperty EnumTypeProperty =
        BindableProperty.Create("EnumType",
                                    typeof(Type),
                                    typeof(EnumPicker),
          propertyChanged: (có thể liên kết, oldValue, newValue) =>
          {
             Bộ chọn EnumPicker = (EnumPicker)bindable;
             nếu (oldValue != null)
             {
                picker.ItemsSource = null;
             }
             nếu (newValue != null)
             {
                if (!((Type)newValue).GetTypeInfo().IsEnum)
                   ném ngoại lệ đối số mới
                  ("EnumPicker: Thuộc tính EnumType phải là
```

```
                                                    kiểu liệt kê");
            picker.ItemsSource = Enum.GetValues((Type)newValue);
                 }
               });
          loại công khai EnumType
          {
            set => SetValue(EnumTypeProperty, value);
            lấy => (Loại)GetValue(EnumTypeProperty);
          }
        }
      }
```

Tiếp theo, bạn cần mở tệp MainPage.xaml và nhập mã XAML sau vào đó. Mã chương trình bên trong một trang thuộc loại

ContentPage chứa bố cục kiểu Lưới, bên trong có một bảng gồm hai hàng. Dòng đầu tiên chứa bố cục lưới. Dòng thứ hai chứa bố cục

FlexLayout mà bố cục được liên kết bằng cách sử dụng liên kết BindingContext Lưới từ dòng đầu tiên. Các điều khiển Nhãn được đặt trong bố cục được tạo kiểu bằng cách sử dụng các phần tử Kiểu và Setter sao cho vị trí dọc của điều khiển Nhãn được căn giữa.

Bên trong bố cục Lưới ở hàng đầu tiên là một bố cục Lưới bên trong khác tạo ra sáu hàng và hai cột.

Trong hàng đầu tiên của bố cục Lưới bên trong ở cột đầu tiên có nhãn hiển thị dòng chữ "Nhãn số lượng:". Cột thứ hai của hàng đầu tiên của bố cục này chứa StackLayout với bố cục điều khiển theo chiều ngang là Nhãn và Stepper. Điều này liên kết điều khiển Nhãn với điều khiển Stepper bằng cách sử dụng liên kết BindingContext. Giá trị được chọn trong điều khiển Stepper sẽ được chuyển đổi thành giá trị văn bản và hiển thị trong điều khiển Nhãn. Giá trị được chọn bởi điều khiển Stepper cho biết số lượng điều khiển Nhãn được đặt trong FlexLayout. Khi giá trị được chọn bởi điều khiển Stepper thay đổi, trình xử lý sự kiện OnStepperChanged sẽ được kích hoạt, được hiển thị trong mã C Sharp trong tệp MainPage.xaml.cs sẽ được tạo

Sau đó.

Ở hàng thứ hai của bố cục Lưới bên trong, ở cột đầu tiên, có nhãn hiển thị tên của thuộc tính “Direction:” . Cột thứ hai của hàng thứ hai chứa phần tử EnumPicker, dựa trên lớp EnumPicker từ tệp EnumPicker.cs. Trong trường hợp này, phần tử
EnumPicker được liên kết với thuộc tính Direction của FlexLayout bằng cách sử dụng Binding.

Ở hàng thứ ba của bố cục Lưới bên trong, ở cột đầu tiên, có nhãn hiển thị tên của thuộc tính “Wrap:” . TRONG

Cột thứ hai của hàng thứ hai chứa phần tử EnumPicker, được liên kết với thuộc tính Wrap của bố cục FlexLayout bằng cách sử dụng Binding.

Ở hàng thứ tư của bố cục Lưới bên trong, ở cột đầu tiên, có nhãn hiển thị tên thuộc tính “JustifyContent:” . Trong cột thứ hai của hàng thứ hai có một phần tử
Một EnumPicker được liên kết bằng cách sử dụng Binding với thuộc tính JustifyContent của FlexLayout.

Ở hàng thứ năm của bố cục Lưới bên trong, ở cột đầu tiên có nhãn hiển thị tên thuộc tính “AlignItems:” . Cột thứ hai của hàng thứ hai chứa phần tử EnumPicker được liên kết với thuộc tính AlignItems của bố cục FlexLayout.

Ở hàng thứ sáu của bố cục Lưới bên trong, ở cột đầu tiên, có nhãn hiển thị tên của thuộc tính “AlignContent:” . Cột thứ hai của hàng thứ hai chứa phần tử EnumPicker được liên kết với thuộc tính AlignContent của bố cục FlexLayout.

Sử dụng điều khiển EnumPicker, danh sách các giá trị của thuộc tính bố cục FlexLayout mà bạn hiện đang làm việc sẽ được hiển thị (đây là các thuộc tính Direction, Wrap, JustifyContent, AlignItems, AlignContent).

Do đó, trong tệp này, năm điều khiển EnumPicker được tạo để hiển thị các giá trị của thuộc tính bố cục FlexLayout. Đồng thời, quy định số lượng điều khiển Nhãn hiển thị trong giao diện người dùng khi khởi chạy ứng dụng

phần mềm là ba. Các giá trị ban đầu của thuộc tính bố cục FlexLayout cũng được đặt, giá trị này sẽ có hiệu lực khi ứng dụng phần mềm được khởi chạy.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
       xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
        xmlns:local="clr-namespace:FlexDemo"
        x:Class="FlexDemo.MainPage">
  <Lề lưới="10, 0">
    <Grid.RowDefinitions>
      <RowDefinition Height="Tự động" />
      <Chiều cao định nghĩa hàng="*" />
    </Grid.RowDefinitions>
    <!-- Bảng điều khiển -->
    <Grid BindingContext="{x:Reference flexLayout}"
        Lưới.Row="0">
      <Grid.RowDefinitions>
        <RowDefinition Height="Tự động" />
        <RowDefinition Height="Tự động" />
        <RowDefinition Height="Tự động" />
        <RowDefinition Height="Tự động" />
        <RowDefinition Height="Tự động" />
```

```
   <RowDefinition Height="Tự động" />
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
   <ColumnDefinition width="Auto" /> <ColumnDefinition
   width="*" /> </Grid.ColumnDefinitions>
<Grid.Resources> <ResourceDictionary>
<Style TargetType="Label">
   <Setter
      Property="VerticalOptions" Value="
         Center" /
         > </Style> </ResourceDictionary> </Grid.Resources> <Label
      Text="Số
   nhãn:"


     Grid.Row="0" Grid.Column="0" />
<StackLayout Orientation="Ngang"
         Grid.Row="0" Grid.Column="1">


   <Nhãn
    Text="{Nguồn liên kết={x:Số tham chiếuStepper},
                 Path=Value,
                 StringFormat='{0:F0}'}" />
   <Stepper x:Name="numberStepper"
         Tối thiểu="0"
         Tối đa="99"
         Tăng = "1"
         Giá trị="3"
         ValueChanged="OnStepperChanged" /> </
StackLayout>
<Label Text="Direction:"
     Grid.Row="1" Grid.Column="0" />
<local:EnumPicker EnumType="{x:Type FlexDirection}"
            SelectedItem="{Hướng liên kết}"
            Grid.Row="1" Grid.Column="1" /> <Label
Text="Wrap:"
     Grid.Row="2" Grid.Column="0" />
<local:EnumPicker EnumType="{x:Type FlexWrap}"
            SelectedItem="{Binding Wrap}"
            Grid.Row="2" Grid.Column="1" />
<Label Text="JustifyContent:"
     Grid.Row="3" Grid.Column="0" />
<local:EnumPicker EnumType="{x:Type FlexJustify}"
            SelectedItem="{Binding JustifyContent}"
            Grid.Row="3" Grid.Column="1" />
```

```
            <Label Text="AlignItems:"
                Grid.Row="4" Grid.Column="0" />
            <local:EnumPicker EnumType=
                                    "{x:Gõ FlexAlignItems}"
                      SelectedItem="{Binding AlignItems}"
                      Grid.Row="4" Grid.Column="1" />
            <Label Text="Căn chỉnh nội dung:"
                Grid.Row="5" Grid.Column="0" />
            <local:EnumPicker EnumType=
                                  "{x:Gõ FlexAlignContent}"
                      SelectedItem="{Binding AlignContent}"
                      Grid.Row="5" Grid.Column="1" />
        </Grid>
        <!--Bố cục FlexLayout -->
        <FlexLayout x:Name="flexLayout"
               BackgroundColor="AliceBlue"
               Grid.Row="1" />
    </Grid>
</Trang nội dung>
```

Tiếp theo, bạn cần mở tệp MainPage.xaml.cs và nhập mã C Sharp được đưa ra bên dưới.

Mã chỉ định một mảng màu để chọn màu văn bản trong các điều khiển Nhãn được đặt trong FlexLayout. Mảng chữ số và chữ thập được thiết kế để hiển thị văn bản được đánh số trong điều khiển Nhãn.

Cung cấp trình xử lý cho sự kiện OnStepperChanged, được tham chiếu từ mã trong tệp MainPage.xaml. Tham số đầu vào là số lượng điều khiển Nhãn được chọn bởi điều khiển Stepper có tên numberStepper đã được xác định trước đó trong tệp MainPage.xaml.

Là đối tượng mà trình xử lý sẽ hành động

OnStepperChanged là bố cục FlexLayout. Tham số numberStepper.Value tạo một giá trị đếm chỉ định số lượng điều khiển Nhãn trong FlexLayout. Vị trí của các điều khiển Nhãn được sắp xếp bằng thuộc tính Trẻ em và sử dụng vòng lặp.

```
sử dụng Hệ thống;
sử dụng System.Collections.Generic;
sử dụng System.Linq;
sử dụng System.Text;
sử dụng System.Threading.Tasks;

sử dụng Xamarin.Forms;
sử dụng Xamarin.Forms.Xaml;
```

```
không gian tên FlexDemo
{
   lớp công khai MainPage : ContentPage { Màu tĩnh

   [] màu = {Color.Red, Color.Magenta,
             Color.Blue, Color.Cyan, Color.Green, Color.Yellow };
      chuỗi tĩnh[] chữ sốText = { "", "Một", "Hai", "Ba",
      "Bốn", "Năm", "Sáu", "Bảy", "Tám", "Chín", "Mười",
      "Mười một", "Mười hai", "Mười ba", "Mười bốn",
      "Mười lăm", "Mười sáu", "Mười bảy", "Mười tám", "Mười chín"}; chuỗi
      tĩnh[] thập kỷText = { "", "", "Hai mươi", "Ba mươi",
        "Bốn mươi", "Năm mươi", "Sáu mươi", "Bảy mươi", "Tám mươi", "Chín mươi"};
      công khai MainPage()

      { Khởi tạoComponent();
         OnStepperChanged(flexLayout, new
        ValueChangedEventArgs(0, numberStepper.Value)); } void

      OnStepperChanged(người gửi đối tượng,
                               ValueChangedEventArgs args)
      {
         int count = (int)args.NewValue;
         while (flexLayout.Children.Count > count) {

    flexLayout.Children.RemoveAt(flexLayout.Children.Count - 1);

         } while (flexLayout.Children.Count < count) {

            số int = flexLayout.Children.Count + 1; chuỗi văn
            bản = ""; nếu (số
            < 20) {

               văn bản = chữ sốText[số];

            }

            else { text = decimText[số / 10] + (số % 10
                   == 0 ? "" : "-") + chữ sốText[số % 10];

            }
            Nhãn nhãn = Nhãn mới {

               Văn bản =
               văn bản, FontSize = 16 + 4 * ((số - 1) %
               4), TextColor = màu sắc [(số - 1) % màu sắc. Độ dài],
```

```
                BackgroundColor = Color.LightGray
            };
            flexLayout.Children.Add(nhãn);
        }
    }
}
```

Để minh họa cách sử dụng bố cục FlexLayout, trước tiên bạn phải xây dựng ứng dụng phần mềm rồi chạy nó. Sau khi khởi chạy ứng dụng phần mềm, bảng điều khiển hiển thị ở phía trên giao diện người dùng và bố cục hiển thị ở phía dưới

FlexLayout. Bảng điều khiển hiển thị điều khiển Stepper cho phép bạn thay đổi số lượng điều khiển Nhãn được đặt trong FlexLayout. Bảng điều khiển cũng chứa năm điều khiển EnumPicker cho phép bạn đặt giá trị của thuộc tính bố cục kiểu FlexLayout đã thảo luận trước đó. Khi bạn khởi chạy một ứng dụng phần mềm, ba điều khiển Nhãn sẽ được hiển thị theo mặc định. Trong trường hợp này, các giá trị thuộc tính của bố cục kiểu FlexLayout hiển thị khi khởi động ứng dụng phần mềm được đặt trước đó trong mã chương trình trong tệp MainPage.xaml. Sử dụng bảng điều khiển, bạn có thể chọn số lượng thành phần Nhãn được đặt bên trong bố cục. Tiếp theo, bằng cách chọn các giá trị thuộc tính bố cục, bạn kiểm soát vị trí của các điều khiển Nhãn trên giao diện người dùng.