CENTRO PARA EL DESARROLLO DE LA INDUSTRIA DEL SOFTWARE | UANL

Xamarin

Desarrollo Móvil con Xamarin

2 – Layouts, recursos y estilos

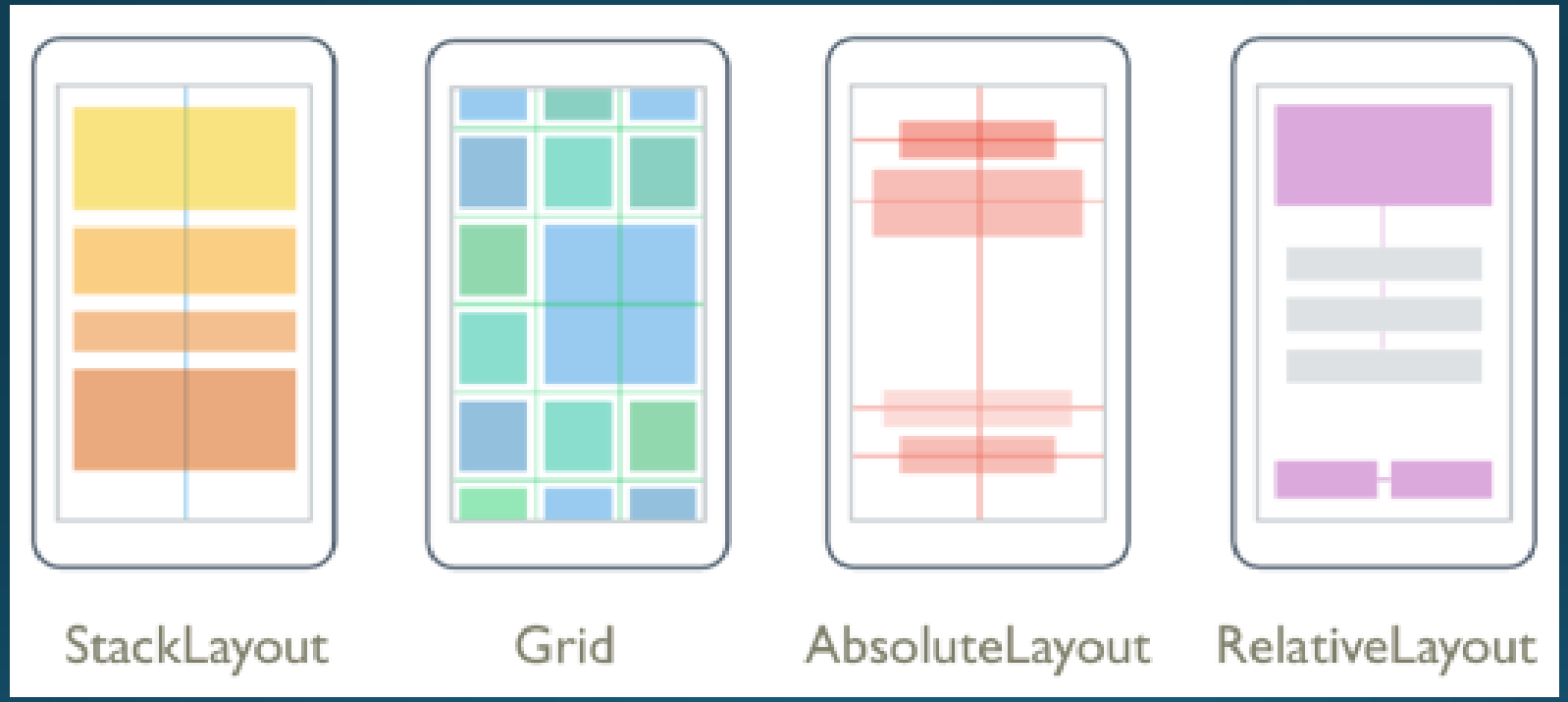
Contenido

- Layouts
- StackLayout
  - Expandir elementos
- Grid
- Recursos
- Estilos

Layouts

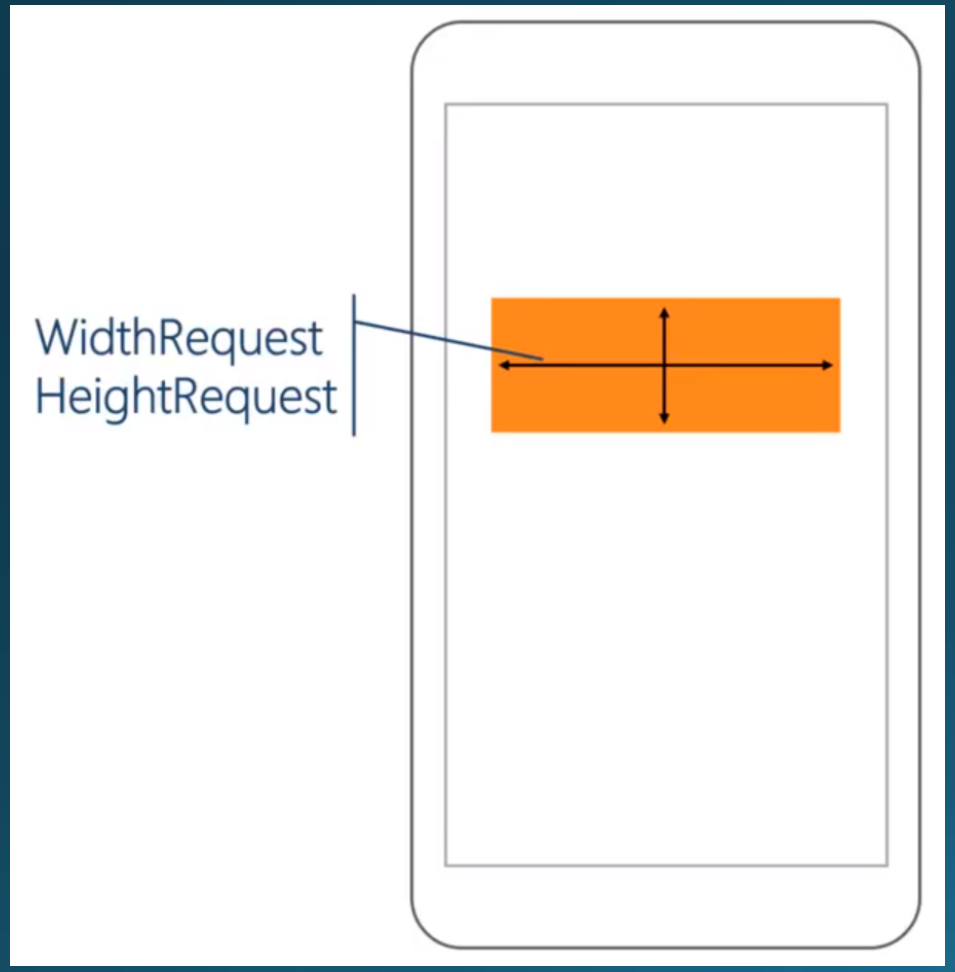
- El tamaño y la posición de los controles en una aplicación de Xamarin.Forms están definidos, principalmente, por su contenedor (layout).

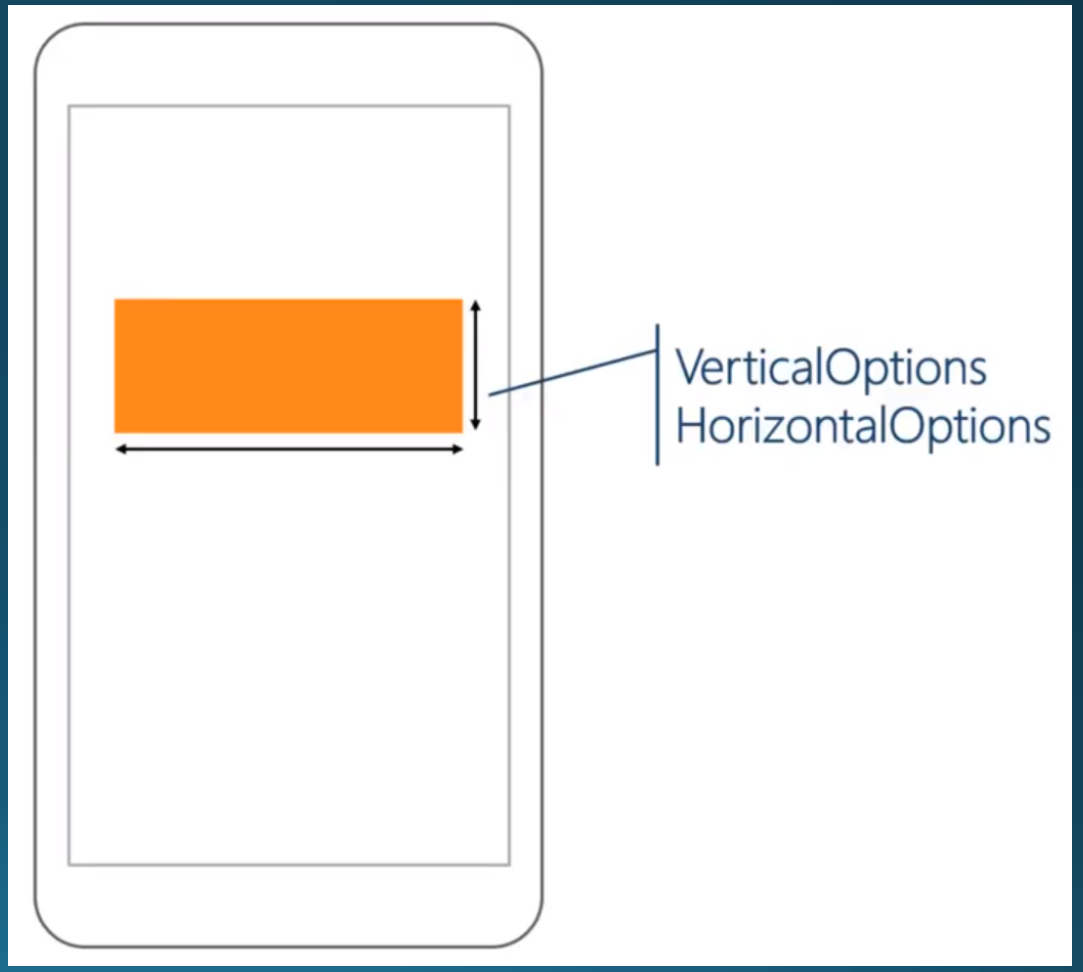
- Los controles, o vistas, definen el tamaño que les gustaría tener, pero el layout que los contiene es el que decide este valor.


StackLayout
Grid
AbsoluteLayout
RelativeLayout

- Los controles (vistas) comunican al layout sus preferencias mediante las propiedades:
  - Tamaño: **WidthRequest** y **HeightRequest**.
  - Posición: **VerticalOptions** y **HorizontalOptions**.

- Las cuatro propiedades son **peticiones**, la decisión final para el tamaño y la posición de cada vista es del layout que los contiene.


WidthRequest
HeightRequest

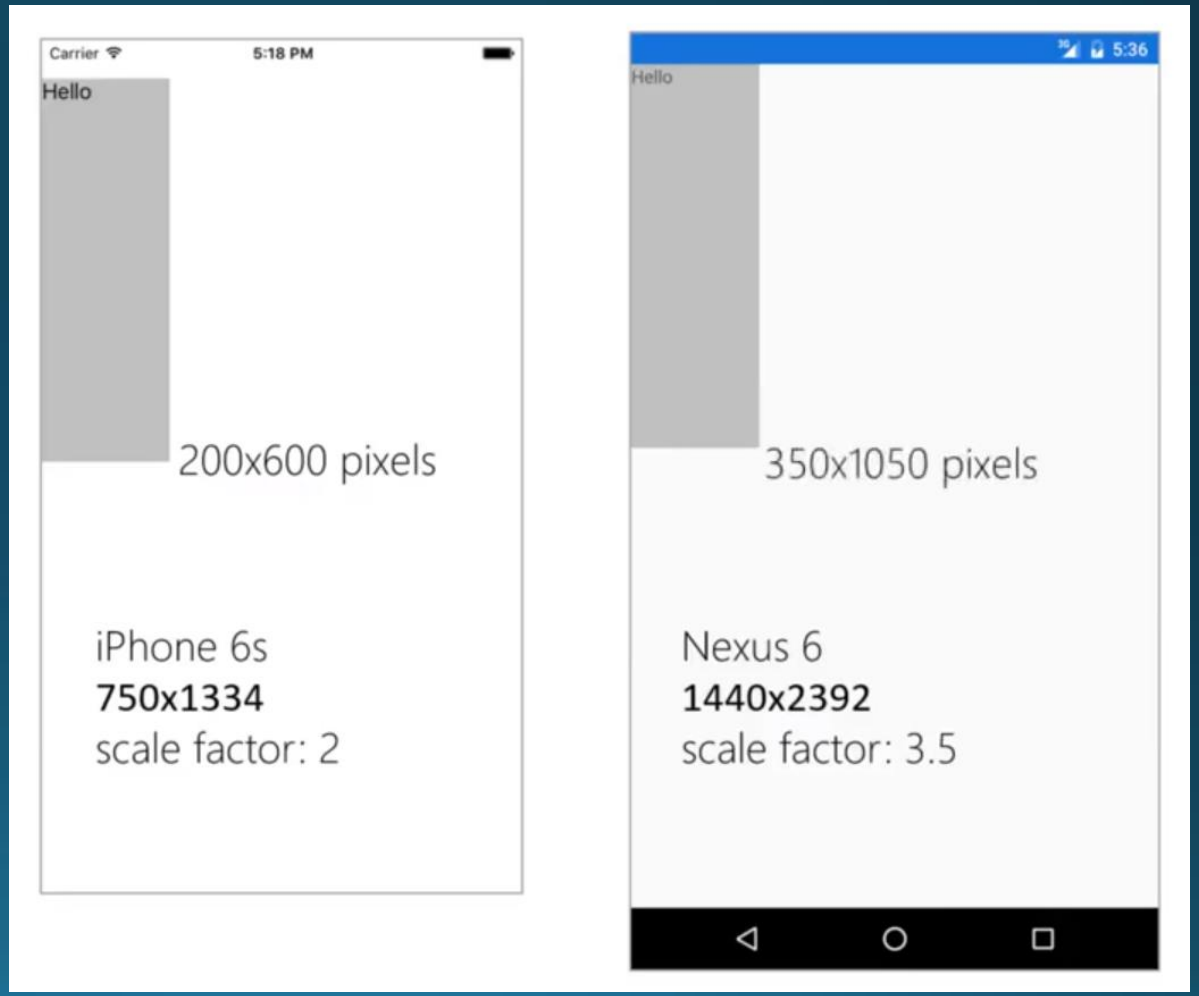

VerticalOptions
HorizontalOptions

```
<Label
    Text="Hello"
    WidthRequest="100"
    HeightRequest="300"
    BackgroundColor="Silver" />
```

- Los valores de **WidthRequest** y **HeightRequest** son simplemente doubles, cada plataforma interpreta lo que significan esos valores.
  - iOS: Points
  - Android: Density-independent pixels

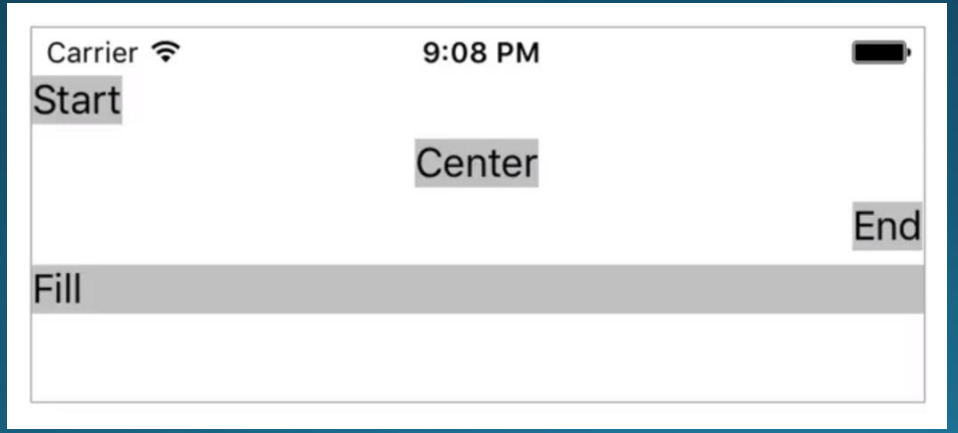
- Las plataformas aplican un factor de escala para determinar los pixeles físicos de cada vista.

```
<Label
    Text="Hello"
    WidthRequest="100"
    HeightRequest="300"
    BackgroundColor="Silver" />
```



- Las propiedades para el posicionamiento pueden tomar cuatro valores: **Start, Center, End** y **Fill**.

```
<StackLayout>
    <Label Text="Start"  HorizontalOptions="Start"  BackgroundColor="Silver" />
    <Label Text="Center" HorizontalOptions="Center" BackgroundColor="Silver" />
    <Label Text="End"    HorizontalOptions="End"    BackgroundColor="Silver" />
    <Label Text="Fill"   HorizontalOptions="Fill"   BackgroundColor="Silver" />
</StackLayout>
```



StackLayout

- El contenedor **StackLayout** acomoda a sus elementos hijos en una columna, de arriba abajo, o en una fila, de izquierda a derecha.

- La orientación de **StackLayout** define este acomodo. Por default, la orientación es vertical.

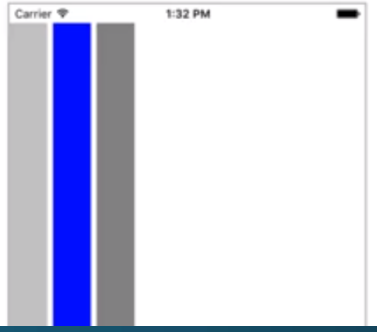
- Por default, también se asigna algo de espacio entre los elementos hijos. La propiedad **Spacing** define este espacio.

Orientación vertical

```
<StackLayout>
    <BoxView Color="Silver" />
    <BoxView Color="Blue" />
    <BoxView Color="Gray" />
</StackLayout>
```

Orientación horizontal

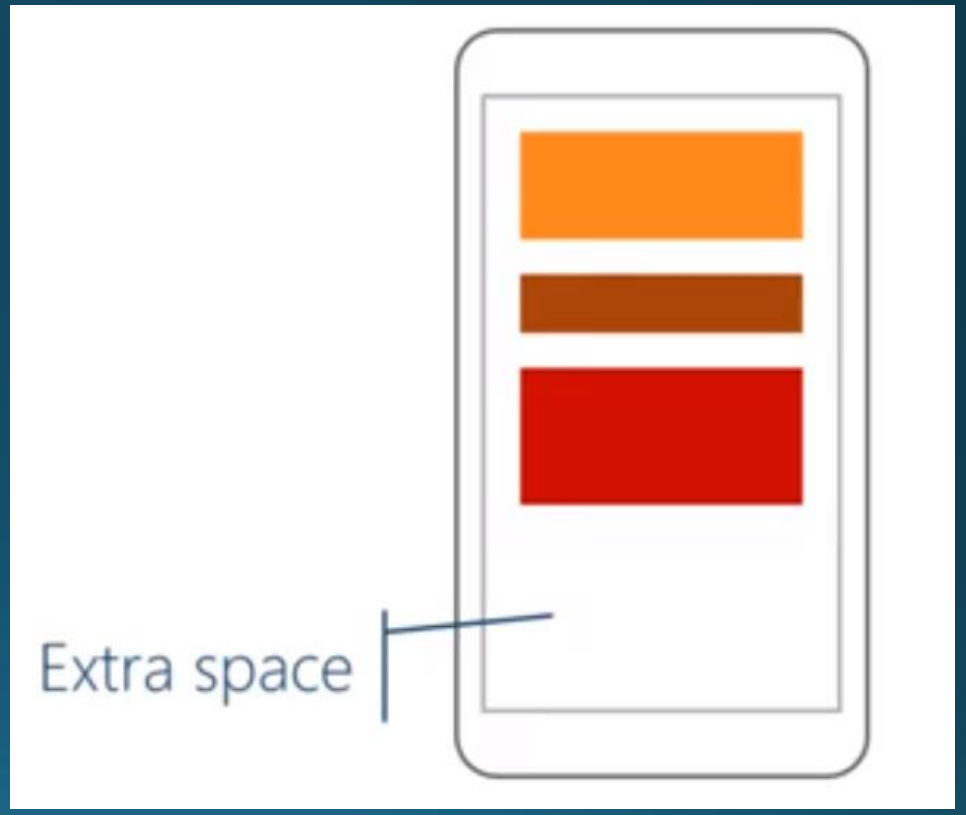
```
<StackLayout Orientation="Horizontal">
    <BoxView Color="Silver" />
    <BoxView Color="Blue"   />
    <BoxView Color="Gray"   />
</StackLayout>
```



Expandir elementos

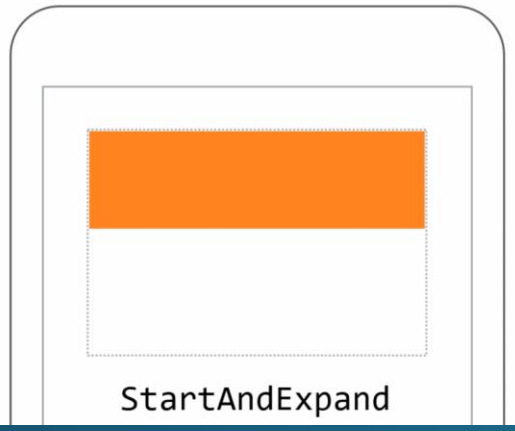
- En ocasiones, los elementos hijos no van a ocupar todo el espacio disponible en el contenedor de **StackLayout**.

- **StackLayout** ofrece una característica para que sus elementos puedan ocupar ese espacio extra: **Expand**.

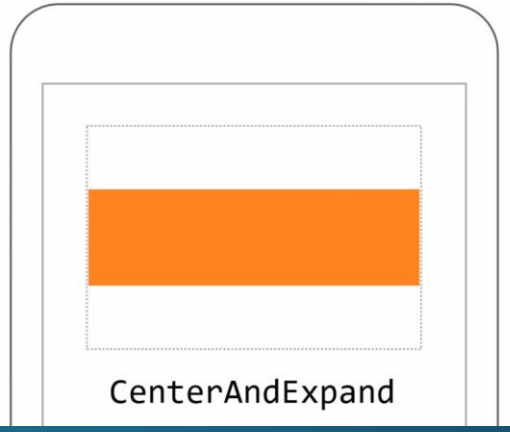


- Sólo el valor de **FillAndExpand** para el posicionamiento vertical y horizontal modifica el tamaño de los controles.

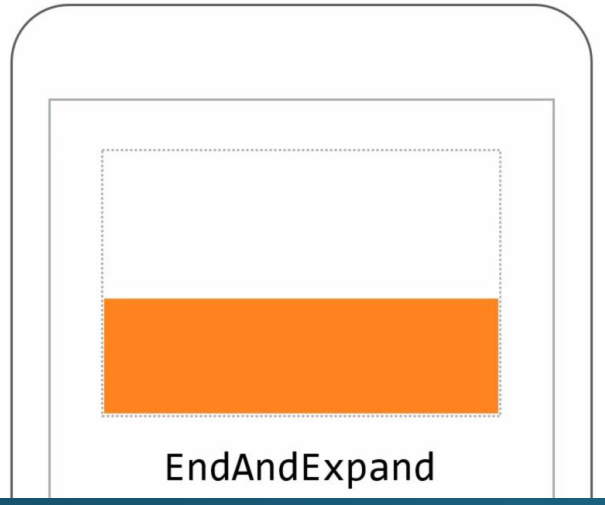
```
<StackLayout Orientation="Vertical">
   <Label ... VerticalOptions="StartAndExpand" />
   <Label ... VerticalOptions="CenterAndExpand" />
   <Label ... VerticalOptions="EndAndExpand" />
   <Label ... VerticalOptions="FillAndExpand" />
</StackLayout>
```



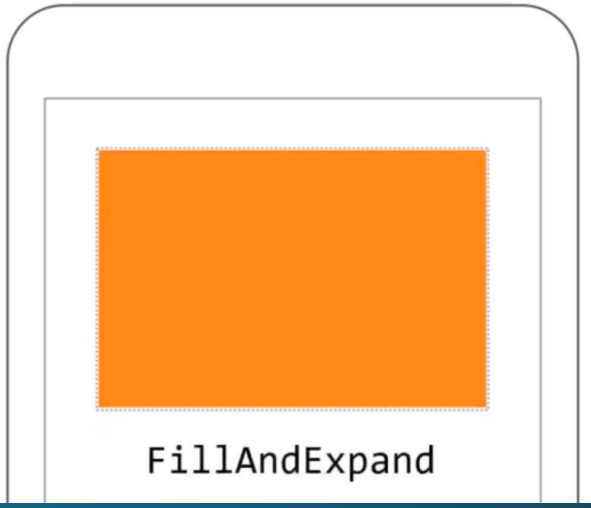

```
<StackLayout Orientation="Vertical">
   <Label ... VerticalOptions="StartAndExpand" />
   <Label ... VerticalOptions="CenterAndExpand" />
   <Label ... VerticalOptions="EndAndExpand" />
   <Label ... VerticalOptions="FillAndExpand" />
</StackLayout>
```



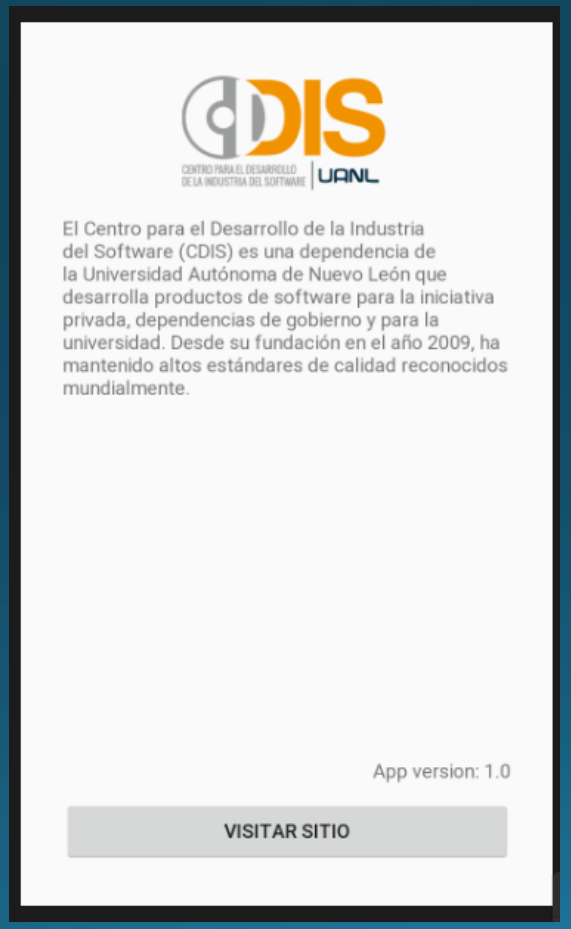
```
<StackLayout Orientation="Vertical">
   <Label ... VerticalOptions="StartAndExpand" />
   <Label ... VerticalOptions="CenterAndExpand" />
   <Label ... VerticalOptions="EndAndExpand" />
   <Label ... VerticalOptions="FillAndExpand" />
</StackLayout>
```




```
<StackLayout Orientation="Vertical">
   <Label ... VerticalOptions="StartAndExpand" />
   <Label ... VerticalOptions="CenterAndExpand" />
   <Label ... VerticalOptions="EndAndExpand" />
   <Label ... VerticalOptions="FillAndExpand" />
</StackLayout>
```



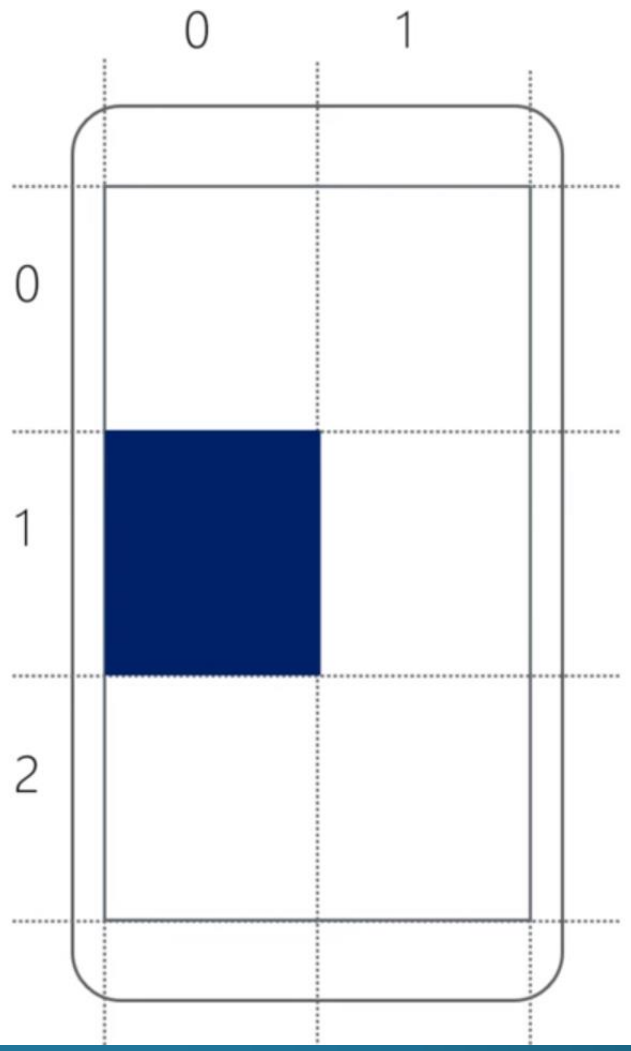
- Ejercicio #1 – Construir UI con StackLayout



Grid

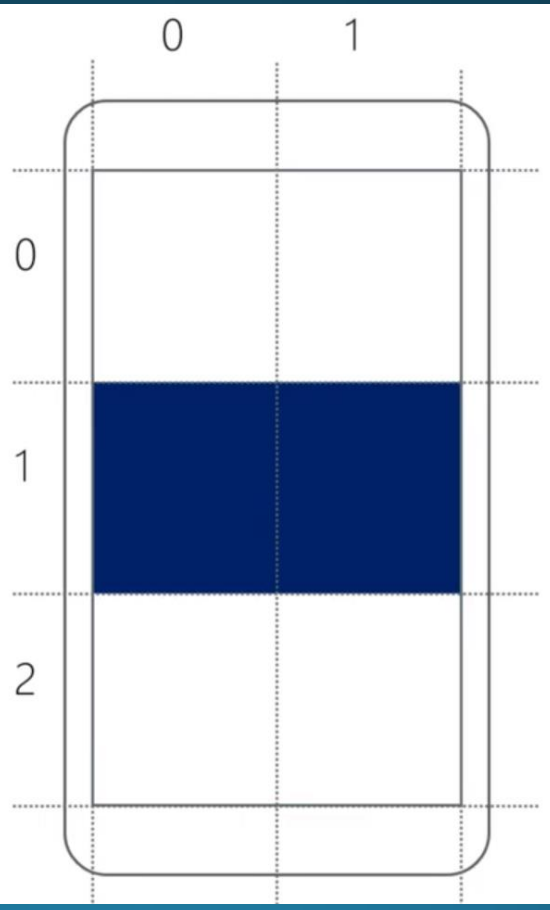
- **Grid** es un contenedor más versátil que **StackLayout**.
- Permite que se puedan definir filas y columnas en las pantallas de la aplicación. Se pueden asignar tamaños para cada fila y/o columna del **Grid**.
- Para posicionar controles (vistas) en un **Grid**, se deben usar las propiedades **Grid.Column** y **Grid.Row**.

```
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition />
        <RowDefinition />
        <RowDefinition />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>

    <BoxView Grid.Row="1" Grid.Column="0"
             BackgroundColor="Navy" />


</Grid>
```



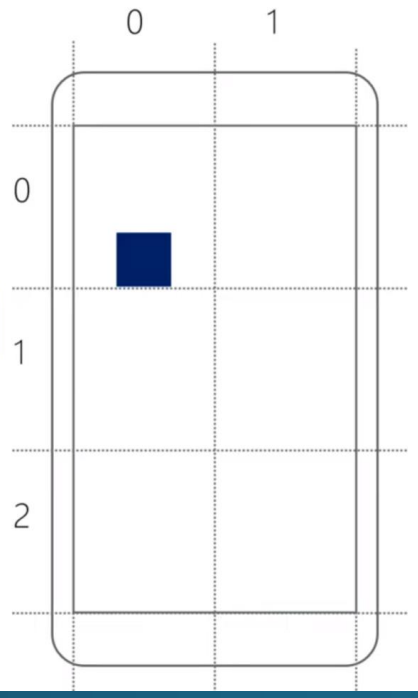
- Las propiedades **Grid.RowSpan** y **Grid.ColumnSpan** proporcionan más de una celda del **Grid** para algún elemento.

```
<Grid>
   <Grid.RowDefinitions>
      <RowDefinition />
      <RowDefinition />
      <RowDefinition />
   </Grid.RowDefinitions>
   <Grid.ColumnDefinitions>
      <ColumnDefinition />
      <ColumnDefinition />
   </Grid.ColumnDefinitions>

   <BoxView Grid.Row="1" Grid.Column="0"
            Grid.ColumnSpan="2"
            BackgroundColor="Navy" />

</Grid>
```

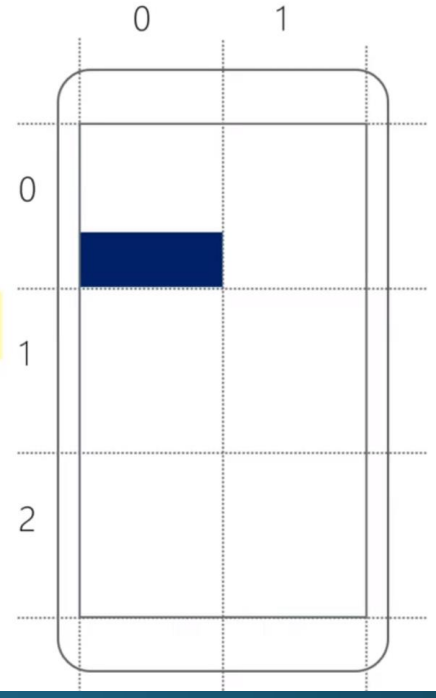


- También es posible alinear elementos dentro de su celda asignada en el **Grid**.

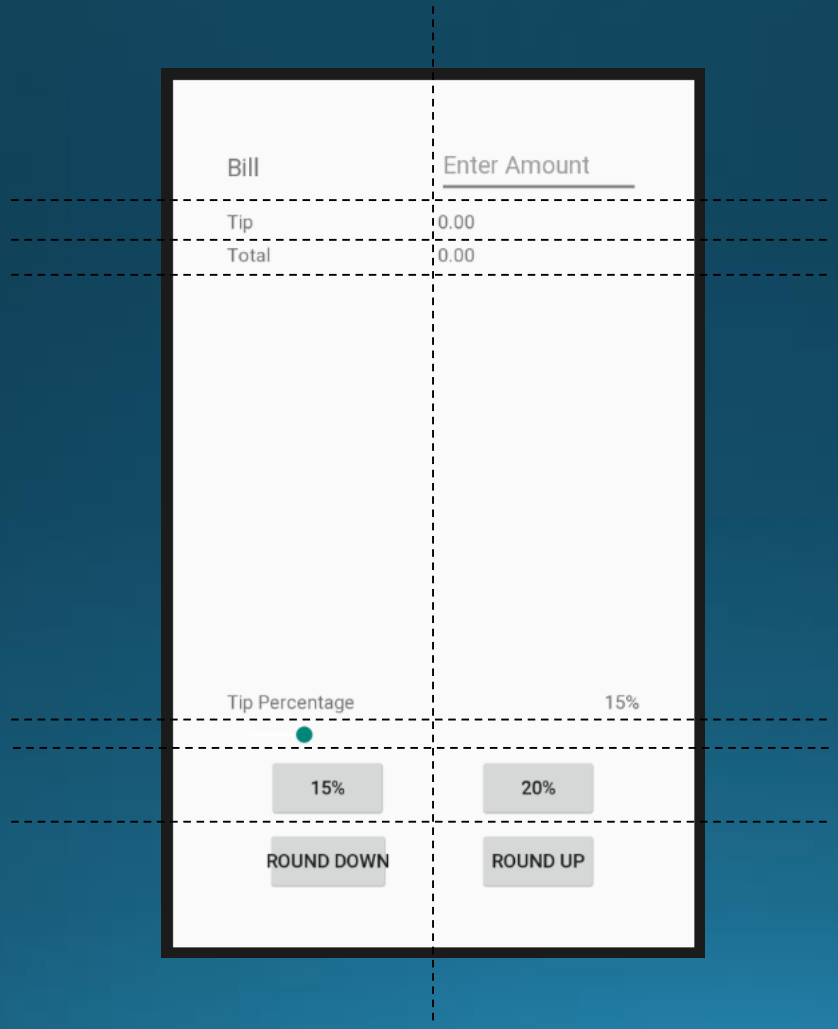
```
<Grid>
    ...
    <BoxView HorizontalOptions="Center"
             VerticalOptions="End"
             BackgroundColor="Navy"
             WidthRequest="50"
             HeightRequest="50"  />
</Grid>
```




```
<Grid>
    ...
    <BoxView HorizontalOptions="Fill"
             VerticalOptions="End"
             BackgroundColor="Navy"
             WidthRequest="50"
             HeightRequest="50"  />
</Grid>
```



- Ejercicio #2 – Construir UI con Grid



Recursos

- Cuando se diseñan las interfaces de usuario (UI), es normal definir los mismos valores para los tipos de letra, los colores, y demás características visuales de los controles de la aplicación.

- Utilizar los mismos valores en toda la aplicación provee una apariencia y experiencia consistente para el usuario.

- Xamarin.Forms provee una manera para definir estos valores en un solo lugar, y no en cada control que los requiera.

- Los recursos son objetos que se pueden compartir a través de la UI.
- Con los recursos, definimos un valor en específico en un solo lugar, y le asignamos un nombre a este valor.
- En XAML, los recursos se almacenan en un diccionario de recursos (**Page.Resources**).

- Ejercicio #3 – Usar Recursos en XAML

- Para actualizar los valores de los recursos definidos en un diccionario en tiempo de ejecución, se debe utilizar la propiedad **DynamicResource** en XAML, y la clase **Resources** en C#.

```
<ResourceDictionary>
  <Color x:Key="bg">Blue</Color>
</ResourceDictionary>

<StackLayout BackgroundColor="{DynamicResource bg}">
  ...
</StackLayout>

void OnChangeColor()
{
  this.Resources["bg"] = Color.Green;
}
```

- Ejercicio #4 – Actualizar Recursos desde C#

Estilos

- Los estilos definen un conjunto de propiedades para vistas (controles) en particular.

- Como con los recursos, a los estilos se les asigna un nombre. En caso de que no tengan un nombre, se aplican a todas las vistas.

```
<Style TargetType="Button">
  <Setter Property="BackgroundColor" Value="#2A84D3" />
  <Setter Property="BorderColor"     Value="#1C5F9B" />
  <Setter Property="BorderRadius"    Value="10" />
  <Setter Property="BorderWidth"     Value="3" />
  <Setter Property="TextColor"       Value="White" />
</Style>
```

- Ejercicio #5 – Crear estilos

- Es posible definir recursos que estén disponibles para todas las pantallas de la aplicación.

- En este caso, se debe definir un diccionario de recursos (**Application.Resources**) a nivel de la aplicación, es decir, en el archivo **App.xaml**.

- Ejercicio #6 – Crear estilos a nivel de la aplicación