**Ejercicio #1 – Construir UI con StackLayout**

1. En raíz del proyecto, crear pantalla **AboutPage**.
   1. En XAML: Dentro de **StackLayout** con **Padding** y **Spacing**:
      1. Un control Image (agregar imagen de carpeta **Recursos**), dos labels y un botón.
   2. En codebehind:
      1. En constructor, crear handler para evento **Clicked** del botón.
      2. Agregar librería **Xamarin.Essentials**.
      3. En handler, llamar método **OpenAsync** de **Browser** para mostrar un sitio en el navegador (ver diferentes **BrowserLaunchMode’s**).
2. Modificar codebehind de **App**, para que la **MainPage** ahora sea **AboutPage**.
3. Correr la app (se muestra el contenido y el sitio web al dar clic al botón).
4. Continuar con la presentación.

**Ejercicio #2 – Construir UI con Grid**

1. En carpeta nueva **TipCalculator**, crear la pantalla **TipGridPage**.
2. En XAML, utilizar **Grid**:
   1. Agregar un **Grid** con **Padding**.
   2. Definir las 7 filas y dos columnas (comentar alturas: **Auto** y **Star**) …
   3. Posicionar las vistas (controles) en cada celda (**Slider** ocupa las dos columnas) …
      1. Asignar **VerticalOptions** a controles de primera fila …
      2. Asignar **VerticalOptions** y **HorizontalOptions** a labels de 4ta fila …
      3. Asignar **HorizontalOptions** a botones …
3. En codebehind, agregar el código de la clase **TipCalculator**, de carpeta **Recursos**.
4. Correr la app (vista de **Grid** correcta).
5. Continuar con la presentación.

**Ejercicio #3 – Usar Recursos en XAML**

1. En XAML de **TipGridPage**, crear la etiqueta **ContentPage.Resources**; definir dos colores: **bgColor** (gris claro) y **lblColor** (azul oscuro).
2. Usar **StaticResource** para acceder a los recursos:
   1. Asignar el color gris claro a **BackgroundColor** del **Grid**.
   2. Asignar el color azul oscuro a **TextColor** de cada **Label**.
3. Correr la app o ver el cambio en la ventana **Preview**.
4. Continuar con la presentación.

**Ejercicio #4 – Actualizar recursos desde C#**

1. Crear el botón **changeColors**:
   1. Crear una nueva fila para el **Grid**. Asignar el botón a esa fila y que ocupe ambas columnas.
   2. Modificar la fila de todos los demás controles (sumar uno).
2. En codebehind, crear handler para el evento **Clicked** de **changeColors**:
   1. Crear variable global **initialColor**, inicializarla a cero en el constructor.
   2. En el handler, si **initialColor** es cero, cambiar los valores del color de los recursos y aumentar a uno; si no es cero, cambiar a los valores originales y disminuir en uno.
3. En XAML, cambiar todas las llamadas a **StaticResource** por **DynamicResource**.
4. Correr la app (colores cambian).
5. Continuar con la presentación.

**Ejercicio #5 – Crear estilos**

1. Asignar atributos **FontSize(Large)** y **FontAttributes(Bold)** a cada Label. \*Comentar que hay mismos atributos para los labels\*
2. Crear un estilo para labels con un nombre que contenga tres setters, para **FontSize**, **FontAttributes** y **TextColor**.
   1. El valor de **TextColor** sigue siendo el recurso correspondiente de su color, con **DynamicResource**.
3. Asignar a cada Label ese estilo (con **DynamicResource**) y quitar los atributos.
4. Crear un estilo para botones (sin nombre) que contenga tres setters: **BackgroundColor**, **TextColor** y **CornerRadius**.
   1. El valor de **BackgroundColor** es el recurso de color azul, y **TextColor** es el recurso de color gris claro; se asignan con **DynamicResource** (el botón **changeColors** invertirá su color).
5. Correr la app o ver en la ventana **Preview**.
6. \*Comentar un nuevo requerimiento: que sólo las cantidades sean **Bold**\*.
7. Crear un nuevo estilo con un setter para **FontAtributes**; agregar el atributo **BasedOn**, con el valor del estilo inicial (con **StaticResource**).
8. Quitar el setter de **FontAttributes** del estilo inicial.
9. Cambiar el valor de **Style** de los Labels de cantidades (seguir usando **DynamicResource**).
10. Correr la app o ver en la ventana **Preview**.
11. Continuar con la presentación.

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

1. En **TipGridPage.xaml**, cortar el contenido de **ContentPage.Resources** (dos colores y tres estilos).
2. En **App.xaml**, pegar el contenido dentro de **Application.Resources**.
3. Correr la app o ver en la ventana **Preview**.