

## PRÁCTICA 2 - EXPLORANDO LA ARQUITECTURA DE .NET MAUI

---

### INFORMACIÓN GENERAL

**Asignatura:** Introducción al Desarrollo de Aplicaciones Móviles

**Período:** 2016-C1

**Semana:** 2

**Instructor:** Michael Grullon

**Valor:** 2 puntos (de 8 puntos totales de prácticas del Primer Período)

**Fecha de entrega:** Usualmente 7 días después de la clase

**Modalidad de entrega:** A través de Moodle

---

### OBJETIVOS DE LA PRÁCTICA

Al completar esta práctica demostrarás que:

- ✓ Comprendes la arquitectura y estructura de .NET MAUI
  - ✓ Puedes identificar los componentes principales de un proyecto
  - ✓ Sabes modificar recursos globales de la aplicación
  - ✓ Entiendes el ciclo de vida de las aplicaciones
  - ✓ Puedes implementar navegación básica entre páginas
- 

### DESCRIPCIÓN DE LA PRÁCTICA

Esta práctica profundiza en la arquitectura interna de .NET MAUI. Explorarás los archivos clave del proyecto, modificarás recursos, implementarás eventos del ciclo de vida y crearás navegación entre páginas.

---

### INSTRUCCIONES DETALLADAS

#### PARTE 1: Exploración de la Estructura del Proyecto (0.5 puntos)

##### 1.1 Abrir tu proyecto de la Semana 1

1. Abre Visual Studio 2022
2. Abre el proyecto que creaste en la Práctica 1: Practica1\_MAUI\_[TuNombre]

3. Si no lo tienes, crea un nuevo proyecto .NET MAUI App

## 1.2 Documentar la estructura

Crea un documento Word o PDF y agrega una sección llamada **"Estructura del Proyecto"**

**Documenta lo siguiente:**

### 1. Archivos principales en la raíz:

- ¿Qué hace MauiProgram.cs?
- ¿Qué hace App.xaml?
- ¿Qué hace App.xaml.cs?
- ¿Qué hace MainPage.xaml?

### 2. Carpeta Resources:

- Lista las subcarpetas que contiene
- ¿Qué tipo de archivos va en cada subcarpeta?

### 3. Carpeta Platforms:

- Lista las carpetas de plataformas que ves
- ¿Para qué sirve cada una?

**CAPTURA DE PANTALLA 1:** Solution Explorer mostrando toda la estructura del proyecto expandida

---

## PARTE 2: Modificar Recursos Globales (0.75 puntos)

### 2.1 Modificar colores de la aplicación

1. Abre el archivo Resources/Styles/Colors.xaml
2. Identifica los colores definidos (Primary, Secondary, Tertiary, etc.)
3. **Modifica los siguientes colores a tu preferencia:**
  - Primary: Cambia a tu color favorito (ejemplo: #FF6B35 para naranja)
  - Secondary: Cambia a un color complementario
  - Background: Mantén un color claro o el que prefieras

**CAPTURA DE PANTALLA 2:** Archivo Colors.xaml mostrando tus colores modificados

## **2.2 Agregar una fuente personalizada (OPCIONAL - BONUS +0.25)**

1. Descarga una fuente gratuita de Google Fonts (ejemplo: Roboto, Montserrat)
2. Copia el archivo .ttf a Resources/Fonts/
3. Abre MauiProgram.cs
4. Agrega el registro de la fuente en ConfigureFonts:

```
.ConfigureFonts(fonts =>
```

```
{
```

```
    fonts.AddFont("OpenSans-Regular.ttf", "OpenSansRegular");
```

```
    fonts.AddFont("OpenSans-Semibold.ttf", "OpenSansSemibold");
```

```
    fonts.AddFont("TuFuente.ttf", "TuFuenteNombre"); // Nueva línea
```

```
});
```

**CAPTURA BONUS 1:** Código mostrando la fuente agregada en MauiProgram.cs

---

## **PARTE 3: Implementar Eventos del Ciclo de Vida (0.5 puntos)**

### **3.1 Modificar App.xaml.cs**

1. Abre el archivo App.xaml.cs
2. Agrega los siguientes métodos dentro de la clase App:

```
protected override void OnStart()
```

```
{
```

```
    base.OnStart();
```

```
    // La aplicación se inició
```

```
    System.Diagnostics.Debug.WriteLine("App OnStart - Aplicación iniciada");
```

```
}
```

```
protected override void OnSleep()
```

```

{
    base.OnSleep();

    // La aplicación pasó a segundo plano

    System.Diagnostics.Debug.WriteLine("App OnSleep - Aplicación en segundo
plano");
}

protected override void OnResume()
{
    base.OnResume();

    // La aplicación regresó del segundo plano

    System.Diagnostics.Debug.WriteLine("App OnResume - Aplicación restaurada");
}

```

3. Ejecuta la aplicación en el emulador
4. Observa la ventana **Output** en Visual Studio
5. Minimiza la app en el emulador (botón Home)
6. Regresa a la app
7. Observa los mensajes en Output

**CAPTURA DE PANTALLA 3:** Archivo App.xaml.cs mostrando los métodos del ciclo de vida implementados

**CAPTURA DE PANTALLA 4:** Ventana Output mostrando los mensajes de Debug.WriteLine

## **PARTE 4: Crear Segunda Página y Navegación (0.75 puntos)**

### **4.1 Crear una nueva página**

1. En Solution Explorer, clic derecho en tu proyecto
2. Selecciona **Add → New Item**

3. Busca **".NET MAUI ContentPage (XAML)"**

4. Nombra la página: DetallesPage.xaml

5. Haz clic en "Add"

#### 4.2 Diseñar la segunda página

1. Abre DetallesPage.xaml

2. Reemplaza el contenido dentro de <ContentPage> con:

```
<VerticalStackLayout Padding="30" Spacing="25">
```

```
<Label
```

```
    Text="Página de Detalles"
```

```
    FontSize="32"
```

```
    FontAttributes="Bold"
```

```
    HorizontalOptions="Center" />
```

```
<Label
```

```
    Text="Esta es una segunda página en tu aplicación MAUI"
```

```
    FontSize="18"
```

```
    HorizontalOptions="Center" />
```

```
<Label
```

```
    Text="Arquitectura: .NET MAUI permite navegar entre páginas fácilmente"
```

```
    FontSize="14"
```

```
    TextColor="Gray"
```

```
    HorizontalOptions="Center" />
```

```
<Button
```

```
    x:Name="btnRegresar"
```

```

        Text="← Regresar"
        FontSize="18"
        Clicked="OnRegresarClicked"
        HorizontalOptions="Center"
        WidthRequest="200" />
</VerticalStackLayout>

```

3. Abre DetallesPage.xaml.cs

4. Agrega el método del evento:

```

private async void OnRegresarClicked(object sender, EventArgs e)
{
    await Navigation.PopAsync();
}

```

**CAPTURA DE PANTALLA 5:** Código XAML de DetallesPage.xaml

#### 4.3 Agregar navegación desde MainPage

1. Abre MainPage.xaml

2. Agrega un botón nuevo antes del botón "Click me" existente:

```

<Button
    x:Name="btnDetalles"
    Text="Ir a Detalles →"
    FontSize="18"
    Clicked="OnDetallesClicked"
    HorizontalOptions="Center"
    Margin="0,0,0,20" />

```

3. Abre MainPage.xaml.cs

4. Agrega el método del evento:

```

private async void OnDetallesClicked(object sender, EventArgs e)

```

```
{  
    await Navigation.PushAsync(new DetallesPage());  
}
```

**CAPTURA DE PANTALLA 6:** Código XAML de MainPage mostrando el botón agregado

#### 4.4 Configurar NavigationPage

1. Abre App.xaml.cs
2. En el constructor, modifica la línea de MainPage:

**ANTES:**

```
MainPage = new MainPage();
```

**DESPUÉS:**

```
MainPage = new NavigationPage(new MainPage());
```

#### 4.5 Ejecutar y probar

1. Ejecuta la aplicación en el emulador
2. Haz clic en el botón "Ir a Detalles"
3. Verifica que navegue a la segunda página
4. Haz clic en "Regresar"
5. Verifica que regrese a la página principal

**CAPTURA DE PANTALLA 7:** Emulador mostrando MainPage con el nuevo botón

**CAPTURA DE PANTALLA 8:** Emulador mostrando DetallesPage después de navegar

---

### PARTE 5: Documentar MauiProgram.cs (0.5 puntos)

#### 5.1 Analizar y documentar

1. Abre MauiProgram.cs
2. En tu documento, crea una sección "**Análisis de MauiProgram.cs**"
3. Responde las siguientes preguntas:

**Preguntas:**

1. **¿Qué es MauiApp.CreateBuilder()?**

- Respuesta esperada: Crea un constructor que permite configurar la aplicación MAUI

2. **¿Para qué sirve .UseMauiApp<App>()?**

- Respuesta esperada: Especifica la clase App como punto de entrada de la aplicación

3. **¿Qué hace .ConfigureFonts()?**

- Respuesta esperada: Registra las fuentes que la aplicación puede usar

4. **¿Por qué es importante builder.Build()?**

- Respuesta esperada: Construye y retorna la aplicación configurada, lista para ejecutarse

**CAPTURA DE PANTALLA 9:** Archivo MauiProgram.cs completo

---

**PARTE 6: Explorar Carpeta Platforms (OPCIONAL - BONUS +0.25)**

**6.1 Identificar archivos específicos de plataforma**

1. Expande la carpeta Platforms/Android/
2. Abre AndroidManifest.xml
3. Identifica el nombre del paquete (package)
4. Expande Platforms/Windows/
5. Observa los archivos de configuración

**En tu documento, crea una sección "Código Específico de Plataforma":**

- ¿Qué información contiene AndroidManifest.xml?
- ¿Para qué sirve el archivo Package.appxmanifest en Windows?
- ¿Por qué necesitamos archivos específicos si MAUI es multiplataforma?

**CAPTURA BONUS 2:** AndroidManifest.xml mostrando configuración del paquete

---

**FORMATO DE ENTREGA**



Debes entregar un **documento PDF** que contenga:

**Portada:**

- Nombre de la asignatura
- Título: "Práctica 2 - Explorando la Arquitectura de .NET MAUI"
- Tu nombre completo
- Matrícula
- Fecha de entrega

**Contenido Obligatorio:**

**Sección 1: Estructura del Proyecto**

- Captura 1: Solution Explorer expandido
- Documentación de archivos principales y carpetas

**Sección 2: Modificación de Recursos**

- Captura 2: Colors.xaml modificado
- Explicación de los colores elegidos

**Sección 3: Ciclo de Vida**

- Captura 3: App.xaml.cs con métodos implementados
- Captura 4: Output con mensajes de Debug
- Explicación de qué hace cada método

**Sección 4: Navegación**

- Captura 5: DetallesPage.xaml
- Captura 6: MainPage.xaml con botón agregado
- Captura 7: MainPage en emulador
- Captura 8: DetallesPage en emulador

**Sección 5: Análisis de MauiProgram.cs**

- Captura 9: MauiProgram.cs
- Respuestas a las 4 preguntas

### Contenido BONUS (Opcional):

#### BONUS 1: Fuente Personalizada (+0.25)

- Captura mostrando fuente agregada en MauiProgram.cs

#### BONUS 2: Código Específico (+0.25)

- Captura de AndroidManifest.xml
- Respuestas sobre archivos específicos de plataforma

---

### CRITERIOS DE EVALUACIÓN




| Criterio                           | Puntos     | Descripción   |
|------------------------------------|------------|---|
| Exploración de estructura          | 0.5        | Documentación completa de la estructura del proyecto      |
| Modificación de recursos           | 0.5        | Colores modificados correctamente en Colors.xaml          |
| Ciclo de vida                      | 0.5        | Métodos implementados y funcionando (evidencia en Output) |
| Navegación entre páginas           | 0.75       | Segunda página creada y navegación funcionando            |
| Análisis de MauiProgram.cs         | 0.25       | Respuestas correctas a las preguntas                      |
| <b>TOTAL BASE</b>                  | <b>2.5</b> |   |
| <b>BONUS: Fuente personalizada</b> | +0.25      | Fuente agregada y registrada correctamente                |
| <b>BONUS: Código específico</b>    | +0.25      | Análisis de archivos específicos de plataforma            |
| <b>TOTAL MÁXIMO</b>                | <b>3.0</b> |   |

**Nota para aprobar:** 1.4 puntos (70% de 2.0 base)




---

## RÚBRICA DETALLADA




### Exploración de estructura (0.5 puntos)

-  **Excelente (0.5):** Documentación completa y correcta de todos los archivos y carpetas
-  **Regular (0.25):** Documentación parcial o con errores menores
-  **Insuficiente (0):** Sin documentación o incorrecta




### Modificación de recursos (0.5 puntos)

-  **Excelente (0.5):** Colores modificados, captura clara, app ejecuta con nuevos colores
-  **Regular (0.25):** Colores modificados pero captura poco clara o errores menores
-  **Insuficiente (0):** No modificó colores o no hay evidencia




### Ciclo de vida (0.5 puntos)

-  **Excelente (0.5):** Métodos implementados correctamente, captura de Output muestra mensajes
-  **Regular (0.25):** Métodos implementados pero falta evidencia de Output
-  **Insuficiente (0):** No implementó métodos o no funcionan

### Navegación entre páginas (0.75 puntos)

-  **Excelente (0.75):** Navegación funciona en ambas direcciones, capturas claras
-  **Regular (0.4):** Navegación parcialmente funcional o capturas incompletas
-  **Insuficiente (0):** Navegación no funciona o no hay evidencia

### Análisis de MauiProgram.cs (0.25 puntos)

-  **Excelente (0.25):** 4 respuestas correctas y completas
-  **Regular (0.15):** 2-3 respuestas correctas
-  **Insuficiente (0):** 0-1 respuestas correctas

---

## PREGUNTAS FRECUENTES

**P: ¿Puedo usar el proyecto de la Práctica 1?** R: Sí, es recomendado. Esta práctica continúa sobre ese proyecto.

**P: No encuentro la carpeta Platforms/** R: Asegúrate de hacer clic en "Show All Files" en el Solution Explorer.

**P: Los mensajes de Debug.WriteLine no aparecen** R: Verifica que la ventana Output esté configurada en "Debug" y que estés ejecutando en modo Debug (no Release).

**P: El NavigationPage me da error** R: Asegúrate de usar `new NavigationPage(new MainPage())` en `App.xaml.cs`.

**P: ¿Puedo cambiar otros recursos además de colores?** R: Sí, puedes experimentar con estilos en `Resources/Styles/Styles.xaml`, pero no es obligatorio.

---

## CONSEJOS Y RECOMENDACIONES

### Para explorar el proyecto:

- Usa Ctrl+Click en archivos para abrirlos rápidamente
- Expande todas las carpetas para ver la estructura completa
- Lee los comentarios en el código, te ayudarán a entender

### Para modificar colores:

- Usa un color picker online para elegir colores en formato hex
- Prueba diferentes combinaciones hasta que te guste
- Recuerda que Primary se usa en los botones y elementos principales

### Para el ciclo de vida:

- La ventana Output debe mostrar "Debug" en el dropdown
- Minimiza y restaura la app varias veces para ver todos los eventos
- Si no ves mensajes, verifica que hayas guardado `App.xaml.cs`

### Para la navegación:

- Asegúrate de que `NavigationPage` esté configurado primero
- La barra de navegación debe aparecer en la parte superior
- El botón de regresar aparece automáticamente en iOS/Android

💡 **Organización del documento:**

- Usa encabezados claros para cada sección
- Agrega números a tus capturas (Captura 1, Captura 2...)
- Incluye explicaciones breves bajo cada captura

---

## **PROBLEMAS COMUNES Y SOLUCIONES**

### **Problema 1: "Navigation is null"**

#### **Solución:**

- Verifica que hayas envuelto `MainPage` en `NavigationPage`
- El código debe ser: `MainPage = new NavigationPage(new MainPage());`

### **Problema 2: No veo los cambios de color**

#### **Solución:**

- Limpia y reconstruye la solución: Build → Clean Solution, luego Build → Rebuild
- Verifica que guardaste `Colors.xaml` antes de ejecutar
- Reinicia el emulador

### **Problema 3: Error al crear `DetallesPage`**

#### **Solución:**

- Asegúrate de elegir `".NET MAUI ContentPage (XAML)"`, no solo `"ContentPage"`
- Verifica que el namespace sea correcto en `DetallesPage.xaml.cs`

### **Problema 4: `Debug.WriteLine` no muestra nada**

#### **Solución:**

- Abre View → Output (o Ctrl+Alt+O)
- En el dropdown "Show output from:", selecciona "Debug"

- Verifica que estés ejecutando en modo Debug, no Release
- 

## RECURSOS ADICIONALES

### Documentación oficial:

- <https://learn.microsoft.com/dotnet/maui/fundamentals/>
- <https://learn.microsoft.com/dotnet/maui/fundamentals/app-lifecycle>

### Tutoriales recomendados:

- Microsoft Learn: MAUI App Lifecycle
- YouTube: ".NET MAUI Navigation Tutorial"

### Soporte:

- Foro de Moodle de la asignatura
  - Consultas en horario de clase
- 

## CHECKLIST ANTES DE ENTREGAR

Antes de subir tu práctica a Moodle, verifica:

- ☐ Tengo las 9 capturas de pantalla obligatorias
- ☐ Documenté la estructura del proyecto completamente
- ☐ Modifiqué los colores y la app ejecuta correctamente
- ☐ Implementé los 3 métodos del ciclo de vida
- ☐ La navegación funciona en ambas direcciones
- ☐ Respondí las 4 preguntas sobre MauiProgram.cs
- ☐ Mi documento tiene portada con todos mis datos
- ☐ El documento está en formato PDF
- ☐ El nombre del archivo es: Practica2\_[MiNombre].pdf
- ☐ He revisado la ortografía y redacción
- ☐ El tamaño del archivo es menor a 10 MB

- [ ] (Opcional) Agregué fuente personalizada
  - [ ] (Opcional) Analicé archivos específicos de plataforma
- 

## FECHA DE ENTREGA



**Fecha límite:** Especificar - generalmente 7 días después de la clase



**Hora límite:** 11:59 PM



**Plataforma:** Moodle - Sección "Prácticas" → "Práctica 2"

---

## CRITERIOS DE PENALIZACIÓN

Se aplicarán las siguientes penalizaciones:

- **Entrega tardía:** -0.5 puntos por día de retraso (máximo 2 días)
  - **Formato incorrecto:** -0.25 puntos (si no es PDF)
  - **Capturas ilegibles:** -0.1 puntos por captura
  - **Plagio detectado:** 0 puntos en la práctica
- 

**Nota importante:** Esta práctica es **individual**. Puedes ayudar a tus compañeros a resolver problemas técnicos, pero cada estudiante debe realizar su propia implementación y entregar sus propias capturas de pantalla.

---

¡Éxito en tu segunda práctica de .NET MAUI!

Esta práctica te ayudará a comprender profundamente cómo funciona .NET MAUI internamente, lo cual es fundamental para el resto del curso.

**Michael Grullon**

Instructor - Introducción al Desarrollo Móvil

Período 2016-C1