

17. Estructura de un proyecto .NET MAUI

Proyecto único

- ❖ **El proyecto único** de .NET MAUI toma las experiencias de desarrollo específicas de cada plataforma y las abstrae en un único proyecto compartido que pueda tener como destino **Android, iOS, macOS y Windows**.
- ❖ Un proyecto único de .NET MAUI proporciona una experiencia de desarrollo multiplataforma simplificada y coherente, independientemente de las plataformas destinadas. El proyecto único de .NET MAUI proporciona las siguientes características:
 - Un único proyecto compartido que puede tener como destino Android, iOS, macOS, Tizen y Windows.
 - Selección simplificada de destino de depuración para ejecutar las aplicaciones .NET MAUI.
 - Archivos de recursos compartidos dentro del único proyecto.
 - Un único manifiesto de aplicación que especifica el título, el identificador y la versión de la aplicación.
 - Acceso a las API y herramientas específicas de la plataforma cuando sea necesario.
 - Un único punto de entrada de la aplicación multiplataforma.

Recursos

Recursos

El proyecto único permite almacenar archivos de **recursos** en una sola ubicación mientras se consumen en cada plataforma. Esto incluye fuentes, imágenes, el icono de la aplicación, la pantalla de presentación, los recursos sin procesar (RAW) y los archivos CSS para aplicar estilos a aplicaciones .NET MAUI.

Normalmente, los archivos de recursos deben colocarse en la carpeta Resources del proyecto de aplicación MAUI de .NET o en las carpetas secundarias de la carpeta Resources y deben tener establecida correctamente su acción de compilación.

Resource	Acción de compilación
Icono de la app	Mauicon
Fuentes	MauiFont
Imágenes	Mauimage
Pantalla de presentación	Mauisplashscreen
Recursos sin procesar	Mauiasset
Archivos CSS	Mauicss

Recursos

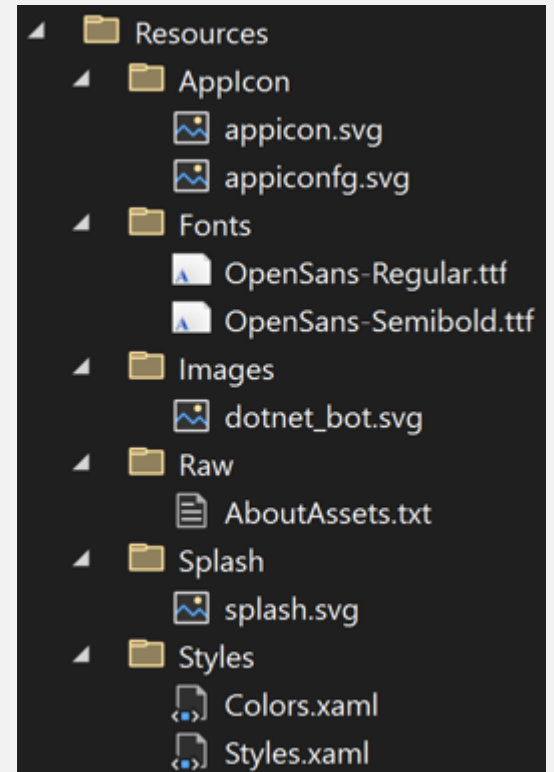
La acción de compilación de un archivo de recursos se establecerá correctamente, si el recurso se ha agregado a la carpeta secundaria Resources.

También podemos editar manualmente cada opción en la configuración del proyecto.

```
<ItemGroup>
  <!-- Images -->
  <MauiImage Include="Resources\Images\*" />

  <!-- Fonts -->
  <MauiFont Include="Resources\Fonts\*" />

  <!-- Raw assets -->
  <MauiAsset Include="Resources\Raw\*" />
</ItemGroup>
```



Icono de la aplicación

Se puede agregar un icono de aplicación al proyecto de aplicación arrastrando una imagen a la carpeta Resources\AppIcon del proyecto, donde su acción de compilación se establecerá automáticamente en **MauiIcon**. Esto crea una entrada correspondiente en el archivo del proyecto:

```
<MauiIcon Include="Resources\AppIcon\appicon.svg" />
```

En tiempo de compilación, el icono de la aplicación se puede cambiar de tamaño a los tamaños correctos para la plataforma y el dispositivo de destino. Los iconos de la aplicación cuyo tamaño se ha cambiado se agregan al paquete de la aplicación. Los iconos de la aplicación se cambian de tamaño a varias resoluciones porque tienen varios usos, incluidos los que se usan para representar la aplicación en el dispositivo y en la tienda de aplicaciones.

Imágenes

Se puede agregar una imagen al proyecto de aplicación arrastrándola a la carpeta Resources\Images del proyecto, donde su acción de compilación se establecerá automáticamente en **MauiImage**. Esto crea una entrada correspondiente en el archivo del proyecto:

```
<MauiImage Include="Resources\Images\logo.svg" />
```

Fuentes

Se puede agregar un formato de tipo verdadero (TTF) o fuente de tipo abierto (OTF) al proyecto de aplicación arrastrándolo a la carpeta Resources\Fonts del proyecto, donde su acción de compilación se establecerá automáticamente en **MauiFont**. Esto crea una entrada correspondiente por fuente en el archivo del proyecto:

```
<MauiFont Include="Resources\Fonts\OpenSans-Regular.ttf" />
```


Fuentes

Se puede agregar un formato de tipo verdadero (TTF) o fuente de tipo abierto (OTF) al proyecto de aplicación arrastrándolo a la carpeta Resources\Fonts del proyecto, donde su acción de compilación se establecerá automáticamente en **MauiFont**. Esto crea una entrada correspondiente por fuente en el archivo del proyecto:

```
<MauiFont Include="Resources\Fonts\OpenSans-Regular.ttf" />
```

SplashScreen

Se puede agregar una pantalla de presentación al proyecto de aplicación arrastrando una imagen a la carpeta Resources\Splash del proyecto, donde su acción de compilación se establecerá automáticamente en **MauiSplashScreen**. Esto crea una entrada correspondiente en el archivo del proyecto:

```
<ItemGroup>  
  <MauiSplashScreen Include="Resources\Splash\splashscreen.svg" />  
</ItemGroup>
```

En tiempo de compilación, la imagen de la pantalla de presentación se cambia de tamaño al tamaño correcto para la plataforma y el dispositivo de destino. La pantalla de presentación cuyo tamaño se ha cambiado se agrega al paquete de la aplicación.

Recursos sin procesar

Un archivo de recursos sin procesar, como HTML, JSON y vídeo, se puede agregar al proyecto de aplicación arrastrándolo a la carpeta Resources\Raw del proyecto, donde su acción de compilación se establecerá automáticamente en **MauiAsset**. Esto crea una entrada correspondiente por recurso en el archivo del proyecto:

```
<MauiAsset Include="Resources\Raw\index.html" />
```

Los controles pueden consumir activos sin procesar, según sea necesario:

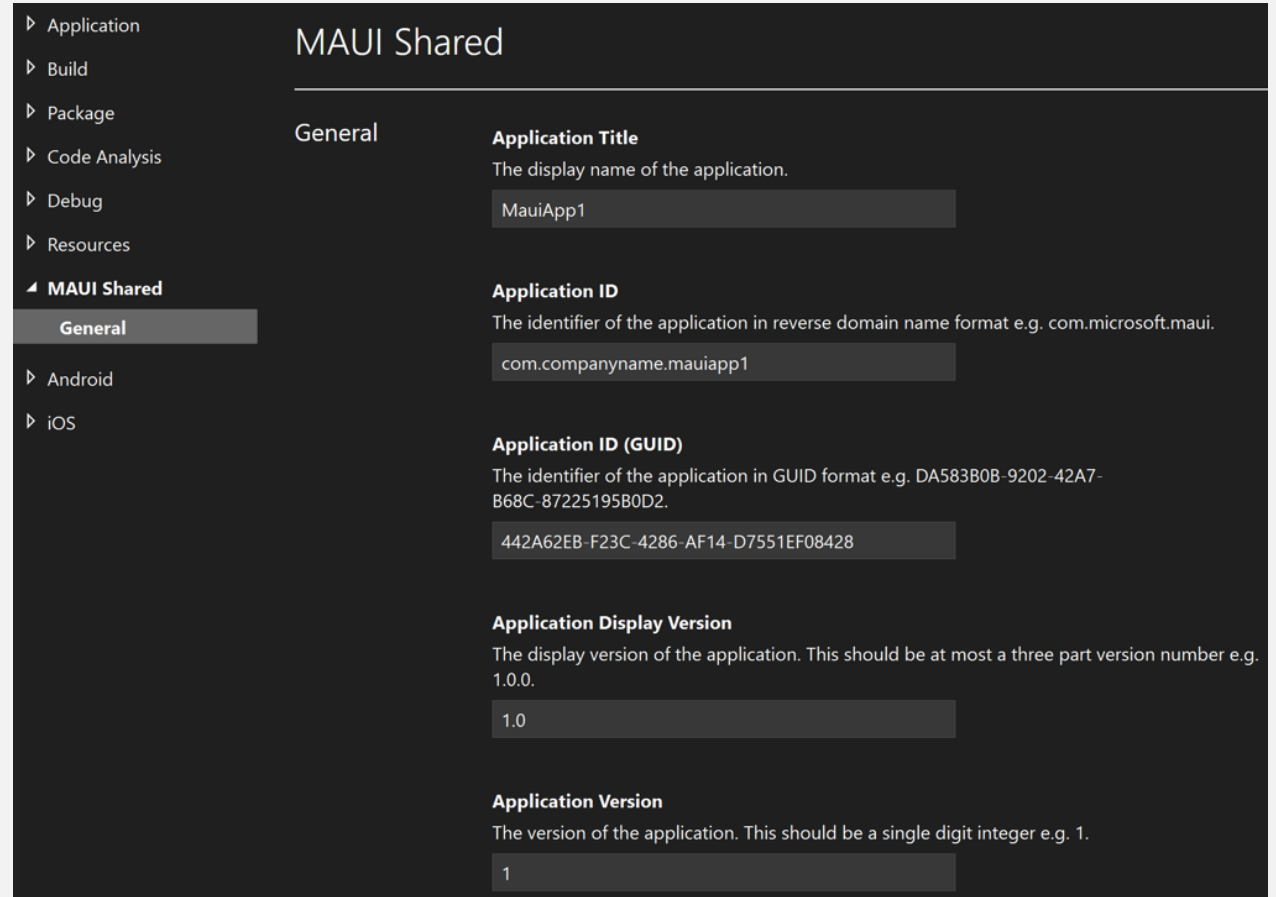
```
<WebView Source="index.html" />
```

Archivo de manifiesto

Manifiesto

Cada plataforma usa su propio archivo de manifiesto de aplicación nativa para especificar información como el título de la aplicación, el identificador, la versión, etc. El proyecto único de .NET MAUI permite especificar estos datos comunes de la aplicación en una sola ubicación en el archivo del proyecto (.csproj).

Para especificar los datos del manifiesto de la aplicación compartida para un proyecto, abra el menú contextual del proyecto en Explorador de soluciones y, a continuación, elija Propiedades. El título, el identificador y la versión de la aplicación se pueden especificar en MAUI Shared > General:



The screenshot shows the 'MAUI Shared' section of the project properties in Visual Studio. The 'General' tab is selected. The following fields are visible:

- Application Title:** The display name of the application. Value: MauiApp1
- Application ID:** The identifier of the application in reverse domain name format e.g. com.microsoft.maui. Value: com.companyname.mauiapp1
- Application ID (GUID):** The identifier of the application in GUID format e.g. DA583B0B-9202-42A7-B68C-87225195B0D2. Value: 442A62EB-F23C-4286-AF14-D7551EF08428
- Application Display Version:** The display version of the application. This should be at most a three part version number e.g. 1.0.0. Value: 1.0
- Application Version:** The version of the application. This should be a single digit integer e.g. 1. Value: 1

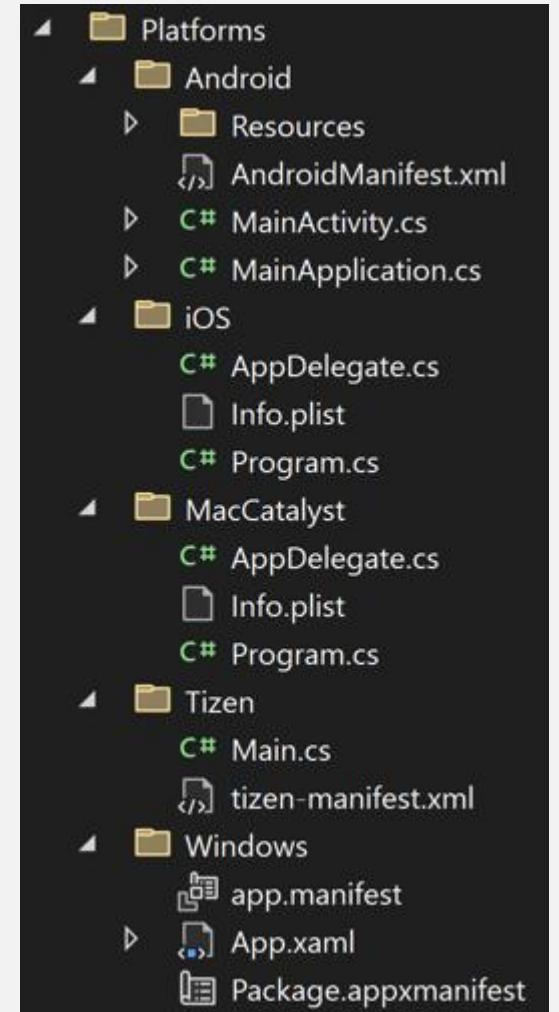
Código específico por plataforma

Código específico de la plataforma

Un proyecto de aplicación .NET MAUI contiene una carpeta **Platforms**, con cada carpeta secundaria que representa una plataforma que .NET MAUI puede tener como destino.

Las carpetas de cada plataforma contienen recursos específicos de la plataforma y código que inicia la aplicación en cada plataforma.

En tiempo de compilación, el sistema de compilación solo incluye el código de cada carpeta al compilar para esa plataforma específica.



Código específico de la plataforma

El destino múltiple también se puede combinar con la compilación condicional para que el código esté destinado a plataformas específicas:

```
#if ANDROID
    handler.NativeView.SetBackgroundColor(Colors.Red.ToNative());
#elif IOS
    handler.NativeView.BackgroundColor = Colors.Red.ToNative();
    handler.NativeView.BorderStyle = UIKit.UITextBorderStyle.Line;
#elif WINDOWS
    handler.NativeView.Background = Colors.Red.ToNative();
#endif
```


Punto de entrada de la aplicación

Punto de entrada de la aplicación

Aunque las carpetas Platforms contienen código específico de la plataforma que inicia la aplicación en cada plataforma, las aplicaciones .NET MAUI tienen un único punto de entrada de aplicación multiplataforma. Cada punto de entrada de plataforma llama a un `CreateMauiApp` método en la clase estática `MauiProgram` del proyecto de aplicación y devuelve un `MauiApp`, que es el punto de entrada de la aplicación.

```
namespace MyMauiApp;

public static class MauiProgram
{
    public static MauiApp CreateMauiApp()
    {
        var builder = MauiApp.CreateBuilder();
        builder
            .UseMauiApp<App>();

        return builder.Build();
    }
}
```

Punto de entrada de la aplicación

La App clase deriva de la clase **Application**:

```
namespace MyMauiApp;

public class App : Application
{
    public App()
    {
        InitializeComponent();

        MainPage = new AppShell();
    }
}
```