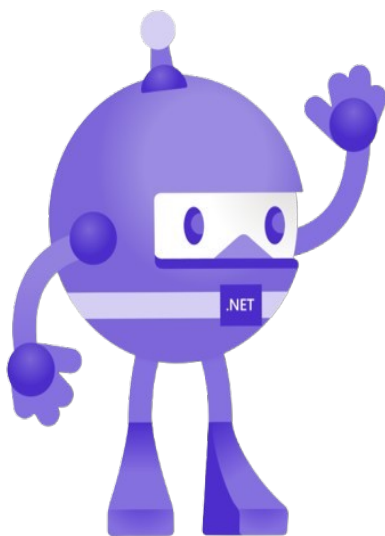


# DESARROLLO DE INTERFACES

## Interfaz con Firebase Gestión de Tareas



Marcos Zahonero

## Índice

<b>Introducción.....</b>	<b>4</b>
<b>Tiempo utilizado.....</b>	<b>5</b>
<b>Material utilizado.....</b>	<b>6</b>
Mi PC de casa.....	6
<b>Objetivos de la práctica.....</b>	<b>7</b>
<b>Desarrollo de la Actividad.....</b>	<b>8</b>
Página - MainPage.....	8
Añadir tareas.....	9
Visual.....	9
Código.....	10
Error al no introducir un nombre (Visual / Código).....	10
Tarea seleccionada.....	11
Código.....	11
Tarea.cs.....	11
Actualizar tarea.....	12
Visual.....	12
Código.....	12
Error al no introducir nombre ni seleccionar.....	13
Eliminar usuario.....	13
Visual.....	13
Código.....	14
Error al no seleccionar una tarea.....	14
CollectionView (OcTareas).....	15
Visual.....	15
Código.....	15
Tarea.cs.....	16
CollectionView (Logs).....	17
Visual.....	17
Código.....	17
Log.cs.....	18
Resultado Final – Vista Móvil.....	19
Resultado Final – Vista Ordenador.....	19
GitHub.....	20
<b>Conclusión.....</b>	<b>21</b>
<b>Problemas y/o sugerencias en la actividad.....</b>	<b>22</b>
1# - Mi extra a la actividad.....	22
<b>Webgrafía.....</b>	<b>23</b>

## Introducción

En esta actividad veremos como podemos incorporar a una interfaz en MAUI una base de datos en Firebase, es decir, con una base de datos online, no local como la anterior actividad, con su tabla, sus campos y sus datos, haremos que se vea el contenido desde la propia interfaz y facilitaremos la forma de añadir/modificar y borrar los datos mediante botones que programaremos para que funcione correctamente, avisándonos de avisos en caso de dar problemas en las operaciones.

## **Tiempo utilizado**

He utilizado 5 horas, y para hacer el documento y vídeos aproximadamente 1 hora más, en total serían 6 horas, esta actividad fue más accesible pero me gasto algo más de tiempo por dedicarle tiempo a la estructura de la interfaz y hacerla atractiva de ver.

## Material utilizado

- El portátil de clase, con mi ordenador personal de casa.
- Visual Studio

## Mi PC de casa

Componente	Nombre
Sistema Operativo	Windows 11 Pro
Procesador	AMD Ryzen 5 2600X
Tarjeta gráfica	NVIDIA GeForce GTX 1650 4GB
Memoria RAM	16 GB
Placa base	B450M-A PRO MAX
Disco duro 1#	KIOXIA EXCERIA G2 SSD 1 TB
Disco duro 2#	ST1000DM010-2EP102 HDD 1 TB

## Objetivos de la práctica

- Realizar una interfaz con Firebase en MAUI
- Entender el funcionamiento de como conectar con la base de datos
- Entender el funcionamiento de los Binding y utilizarlos
- Diseñar una interfaz interesante de ver
- Conocer Firebase para entender como aplicarlo en distintas ocasiones

## Desarrollo de la Actividad

Esta aplicación consta de una única página, con los siguientes elementos:

- 1 Label
- 1 Entry
- 3 Buttons
- 2 CollectionView
- 2 ScrollView

### Página - MainPage

Al iniciar la aplicación aparece así, en Entry vacío y una vista actualizada de la base de datos actual, en la cual ahora mismo no tiene datos:

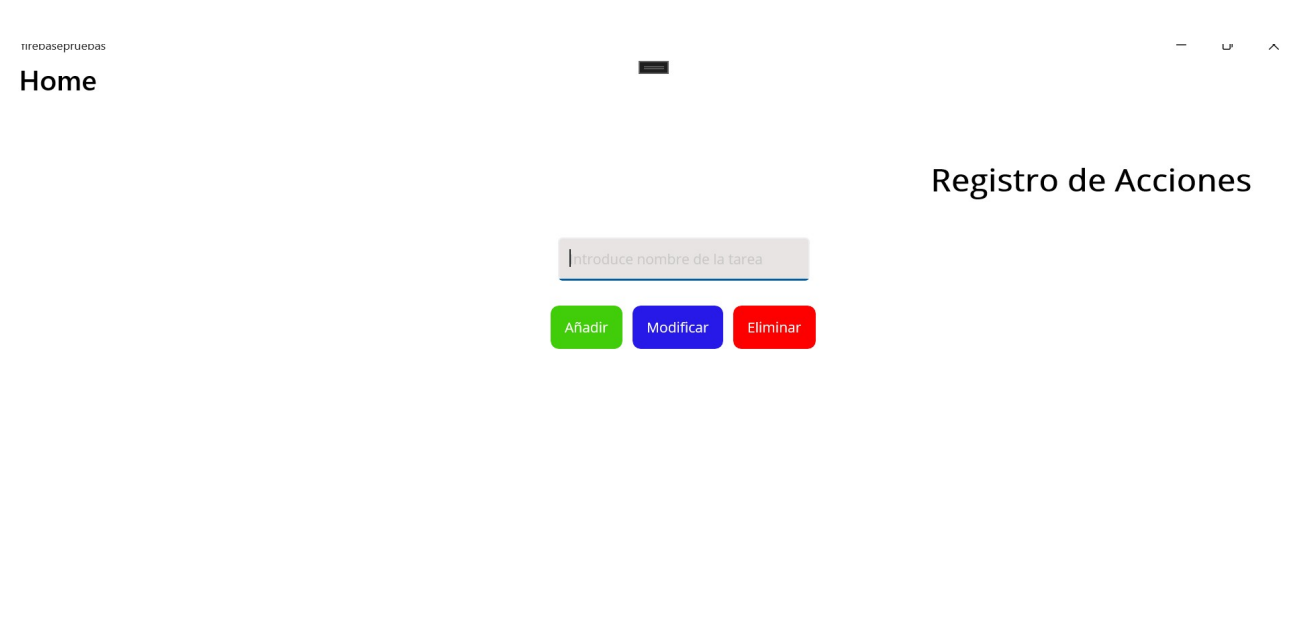


Figura 1: Página inicial de la MainPage

En este caso he puesto algunas tareas y he eliminado una para que se vea correctamente en el registro de acciones.

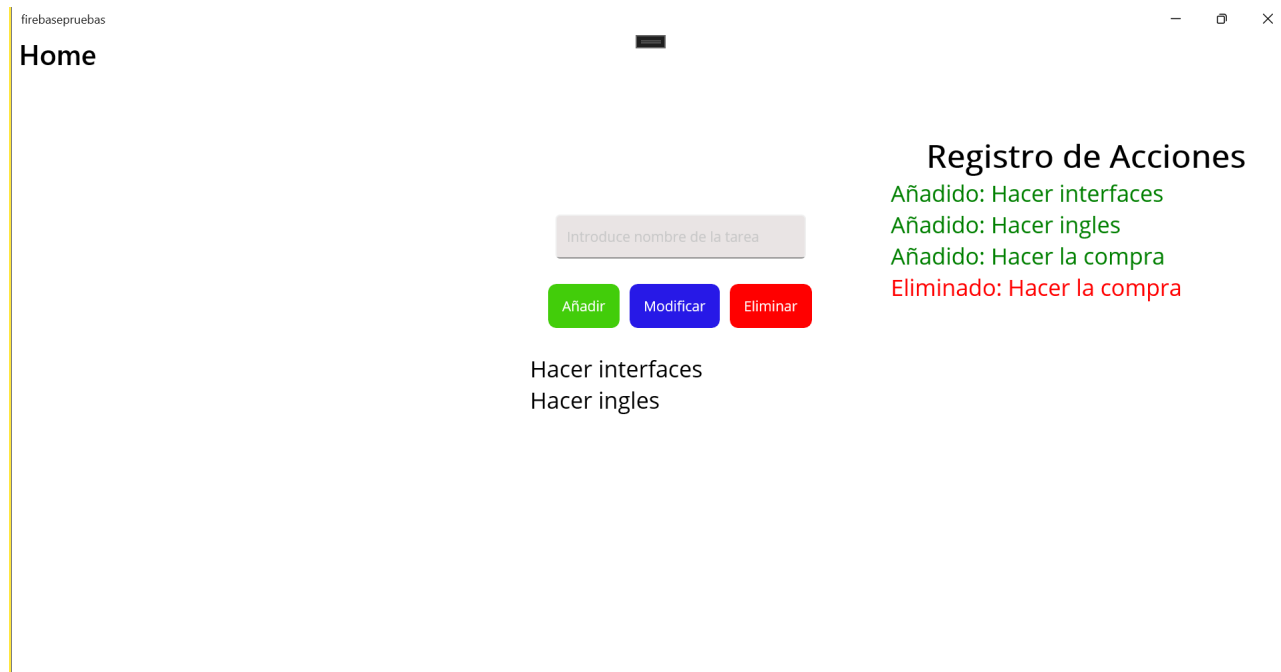


Figura 2: Página MainPage con algunos datos introducidos

## Añadir tareas

Para añadir usuarios a la base de datos tenemos el único Entry que tenemos en toda la interfaz y el botón de “Añadir” que al hacer clic se añadirá a la base de datos y también se verá gráficamente mediante un CollectionView.

## Visual

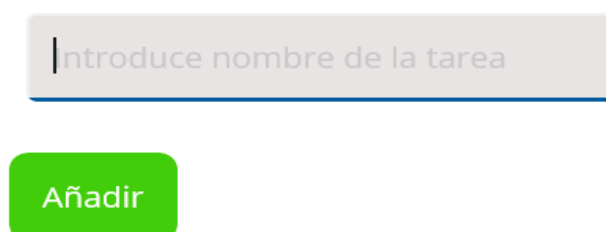


Figura 3: MainPage (Entry y BtnAñadir)



## Código

```
private async void btnAdd_Clicked(object sender, EventArgs e)
{
    if (!string.IsNullOrEmpty(eNombre.Text))
    {
        Tarea tarea = new Tarea
        {
            IdTarea = string.Empty, // Inicialmente vacío
            NombreTarea = eNombre.Text
        };

        // Publicar la tarea en Firebase y obtener la respuesta
        var response = await firebaseClient.Child("Tareas").PostAsync(tarea);
        string key = response.Key; // Obtener la clave generada

        if (!string.IsNullOrEmpty(key))
        {
            // Actualizar el ID de la tarea en Firebase
            await firebaseClient.Child("Tareas").Child(key).PutAsync(new Tarea
            {
                IdTarea = key,
                NombreTarea = tarea.NombreTarea
            });

            // Agrega la acción al registro
            Logs.Add(new Log
            {
                Descripcion = $"Añadido: {eNombre.Text}",
                Tipo = "Añadir"
            });
        }

        // Recargar la base de datos para actualizar la lista
        reloadDatabase();

        // Limpiar el campo de entrada
        eNombre.Text = string.Empty;
    }
}
```

Figura 4: MainPage.xaml.cs del método del botón Añadir

## Error al no introducir un nombre (Visual / Código)

Al no cumplir la condición de `if (!string.IsNullOrEmpty(eNombre.Text))` saltará este aviso:

**¡Error!**

Tienes que poner un nombre.



Figura 5: DisplayAlert visual

```
else
{
    await DisplayAlert("¡Error!", "Tienes que poner un nombre.", "Vale");
}
```

Figura 6: Código del DisplayAlert

## Tarea seleccionada

No hay forma visual de ver la tarea seleccionada

## Código

Al seleccionar un elemento (tarea) de la lista se guardará automáticamente en la variable `selected`, que es un objeto de tipo `Tarea`, en el cual podrás acceder a su `id` y `nombreTarea`.

```
<CollectionView
    ItemsSource="{Binding OcTareas}"
    SelectionMode="Single"
    SelectedItem="{Binding Selected}">
```

Figura 8: Selected XAML

```
private Tarea _selected;
11 referencias
public Tarea Selected
{
    get { return _selected; }
    set
    {
        if (value != null)
        {
            _selected = value;
            OnPropertyChanged();
        }
    }
}
```

Figura 7: Selected C#

## Tarea.cs

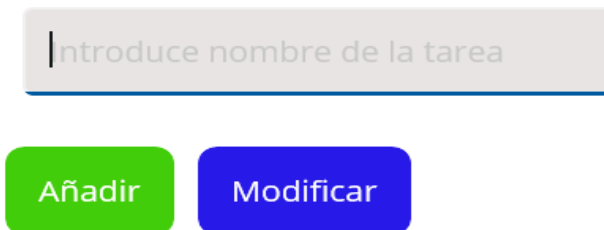
```
public class Tarea : INotifyPropertyChanged
{
    private string _IdTarea;
    private string _NombreTarea;
    7 referencias
    public string IdTarea
    {
        get { return _IdTarea; }
        set
        {
            _IdTarea = value;
            OnPropertyChanged();
        }
    }
    13 referencias
    public string NombreTarea
    {
        get { return _NombreTarea; }
        set
        {
            _NombreTarea = value;
            OnPropertyChanged();
        }
    }

    public event PropertyChangedEventHandler PropertyChanged;
    2 referencias
    protected void OnPropertyChanged([CallerMemberName] string name = null)
    {
        PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(name));
    }
}
```

Figura 9: Clase Tarea con sus atributos

## Actualizar tarea

### Visual



### Código

Al seleccionar una tarea como he dicho antes saco la Tarea y gracias a eso puedo aquí saber cual quiero reemplazar, así que creo un nuevo registro por una parte para marcar los registros que comentaré después y consulto la base de datos y usando la ID del selected y cambio la Tarea con la actual.

```
private async void btnUpdate_Clicked(object sender, EventArgs e)
{
    // Verificar que haya una tarea seleccionada
    if (Selected != null)
    {
        // Verificar que haya texto en el eNombre
        if (!string.IsNullOrEmpty(eNombre.Text))
        {
            // Agrega la acción a la lista
            Log registro = new Log
            {
                Descripción = $"Actualizado: {Selected.NombreTarea} -> {eNombre.Text}",
                Tipo = "Actualizar"
            };

            if (registro.Descripción != _ultimoLog.Descripción && registro.Tipo != _ultimoLog.Tipo)
            {
                Logs.Add(registro);
            }

            registro = _ultimoLog;

            // Actualizar el nombre de la tarea seleccionada
            Selected.NombreTarea = eNombre.Text;

            // Actualizar la tarea en Firebase
            await firebaseClient.Child("Tareas").Child(Selected.IdTarea).PutAsync(Selected);

            // Refrescar la lista
            reloadDatabase();

            // Limpiar el campo de entrada
            eNombre.Text = string.Empty;
        }
    }
}
```

### Error al no introducir nombre ni seleccionar

Al ser la variable `if` (`Selected != null`) saldrá un `DisplayAlert` mostrando texto como avisando de lo siguiente en la primera tarea, y si si tienes algo seleccionado pero no cumples la condición de `if` (`!string.IsNullOrEmpty(eNombre.Text)`) te salta la segunda alerta de abajo:

```
else
{
    await DisplayAlert("¡Error!", "El campo de nombre no puede estar vacío.", "Vale");
}
else
{
    await DisplayAlert("¡Error!", "Tienes que seleccionar una tarea primero.", "Vale");
}
```

Figura 10: (MainPage.xaml.cs) `DisplayAlerts` actualizando tareas

### Eliminar usuario

#### Visual

Añadir

Modificar

Eliminar

## Código

Seleccionamos de nuevo a un usuario y lo borramos sabiendo su ID como también dije antes.

```
private async void btnRemove_Clicked(object sender, EventArgs e)
{
    if (Selected != null)
    {
        Debug.WriteLine(Selected.NombreTarea);
        var collection = firebaseClient
        //Esborra l'item de Firebase amb la clau itemseleccionado.id
        .Child("Tareas").Child(Selected.IdTarea).DeleteAsync();

        // Agrega la acción a la lista
        Log registro = new Log
        {
            Descripcion = $"Eliminado: {Selected.NombreTarea}",
            Tipo = "Eliminar"
        };

        Logs.Add(registro);

        // Limpiar el campo de entrada
        eNombre.Text = string.Empty;

        // Volvemos a actuar la base de datos para que se actualice la tabla
        reloadDatabase();
    }
}
```

Figura 11: (MainPage.xaml.cs) Función borrar

## Error al no seleccionar una tarea

Al ser la variable `selected == null` saldrá el así:

**¡Error!**

Tienes que seleccionar antes una tarea.

Vale

## CollectionView (OcTareas)

### Visual

La parte inferior que tiene los 3 elementos:



Hacer Interfaces

Hacer Ingles

Hacer PSP

### Código

Tiene un ScrollView para que puedas hacer bajar y subir en cuanto hayan muchos más trabajadores de lo esperado y así que no estropee el resto de interfaz.

Tiene un Binding que hace que se vuelve la información de la lista OcTareas y cuando seleccionamos una tarea ocurre lo que decía antes, coge esa tarea y lo establece en una variable del código, aquí los Bindings:

### XAML:

```
<CollectionView
    ItemsSource="{Binding OcTareas}"
    SelectionMode="Single"
    SelectedItem="{Binding Selected}">

    <CollectionView.ItemTemplate>
        <DataTemplate>
            <StackLayout Orientation="Horizontal">
                <Label Text="{Binding NombreTarea}"
                    FontSize="Medium"/>
            </StackLayout>
        </DataTemplate>
    </CollectionView.ItemTemplate>
</CollectionView>
```

**Tarea.cs**

```
public class Tarea : INotifyPropertyChanged
{
    private string _IdTarea;
    private string _NombreTarea;
    public string IdTarea
    {
        get { return _IdTarea; }
        set
        {
            _IdTarea = value;
            OnPropertyChanged();
        }
    }
    public string NombreTarea
    {
        get { return _NombreTarea; }
        set
        {
            _NombreTarea = value;
            OnPropertyChanged();
        }
    }

    public event PropertyChangedEventHandler PropertyChanged;
    protected void OnPropertyChanged([CallerMemberName] string name = null)
    {
        PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(name));
    }
}
```

## CollectionView (Logs)

### Visual

La parte inferior que tiene los 3 elementos, que contiene además de esos tipos contiene el de actualizar que aquí no aparece pero en el vídeo sí aparece:

## Registro de Acciones

Añadido: Patata

Añadido: Hacer interfaces

Eliminado: Patata

### Código

```
<CollectionView ItemsSource="{Binding Logs}">
  <CollectionView.ItemTemplate>
    <DataTemplate>
      <Label Text="{Binding Descripcion}" FontSize="Medium">
        <Label.Triggers>
          <DataTrigger TargetType="Label" Binding="{Binding Tipo}"
            Value="Añadir">
            <Setter Property="TextColor" Value="Green" />
          </DataTrigger>
          <DataTrigger TargetType="Label" Binding="{Binding Tipo}"
            Value="Actualizar">
            <Setter Property="TextColor" Value="#2719E7" />
          </DataTrigger>
          <DataTrigger TargetType="Label" Binding="{Binding Tipo}"
            Value="Eliminar">
            <Setter Property="TextColor" Value="Red" />
          </DataTrigger>
        </Label.Triggers>
      </Label>
    </DataTemplate>
  </CollectionView.ItemTemplate>
</CollectionView>
```



**Log.cs**

Contiene la descripción que es el “Añadido: Hacer Interfaces” y el tipo es para saber que tipo es para colorearlo a su forma.

```
public class Log
{
    public string Descripcion { get; set; }
    public string Tipo { get; set; }
}
```

## Resultado Final – Vista Móvil

Introduce nombre de la tarea

Añadir

Modificar

Eliminar

Hacer Interfaces  
Hacer Ingles  
Hacer PSP

## Registro de Acciones

## Resultado Final – Vista Ordenador

Home

Introduce nombre de la tarea

Añadir

Modificar

Eliminar

Hacer Interfaces  
Hacer Ingles  
Hacer PSP

## Registro de Acciones

## GitHub

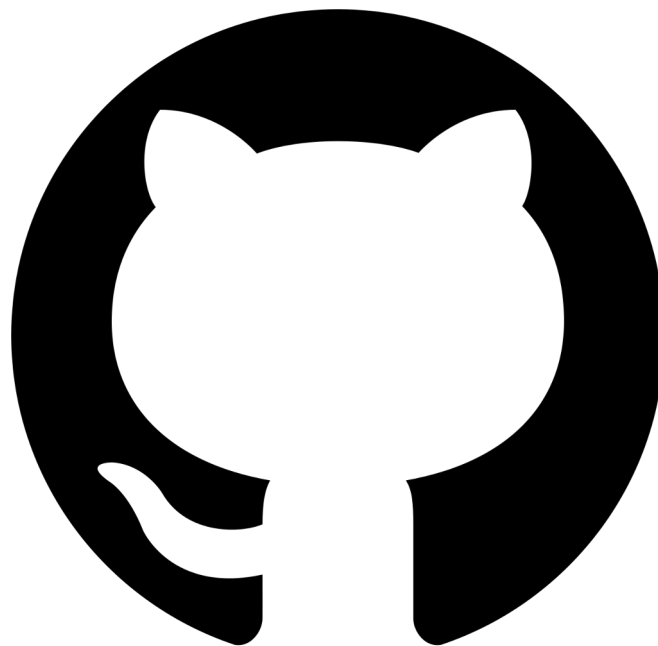


**He cambiado el nombre del repositorio así que si has estado tenido problemas con mis actividades al corregir será por eso, en esta actividad está corregido para el resto del curso.**



En esta actividad he añadido GitHub para añadir el proyecto completo, junto a una descripción en el [README.md](#) para añadir una breve explicación y preview de cada actividad.

Enlace: [Maek0s/2DAM\\_DesarrolloInterfaces](#)



## **Conclusión**

En conclusión, firebase tiene más errores que una base de datos normal sin ningún tipo de dudas, incluso hay veces aleatorias que ocurre sin ningún motivo.

## Problemas y/o sugerencias en la actividad

### 1# – Mi extra a la actividad

Aquí estamos de nuevo, 2 vídeos, uno de su funcionamiento, añadiendo, actualizando, borrando y viendo los posibles errores que da y como reacciona la interfaz para avisarte y poco más, el otro vídeo de explicativo se basará sobre todo en el código principal donde hablaré de como he gestionado para seleccionar el usuario y así poder hacer las funciones que solicita el ejercicio.



Vídeo funcionamiento: <https://youtu.be/aUg2jU0Fsz8>

Video explicativo de código: <https://youtu.be/lePUUrnidU>

## Webgrafía

- <https://visualstudio.microsoft.com/es/vs/community/>
- <https://dotnet.microsoft.com/es-es/apps/maui>

- **CollectionsView:**

<https://learn.microsoft.com/es-es/dotnet/maui/user-interface/controls/collectionview/>