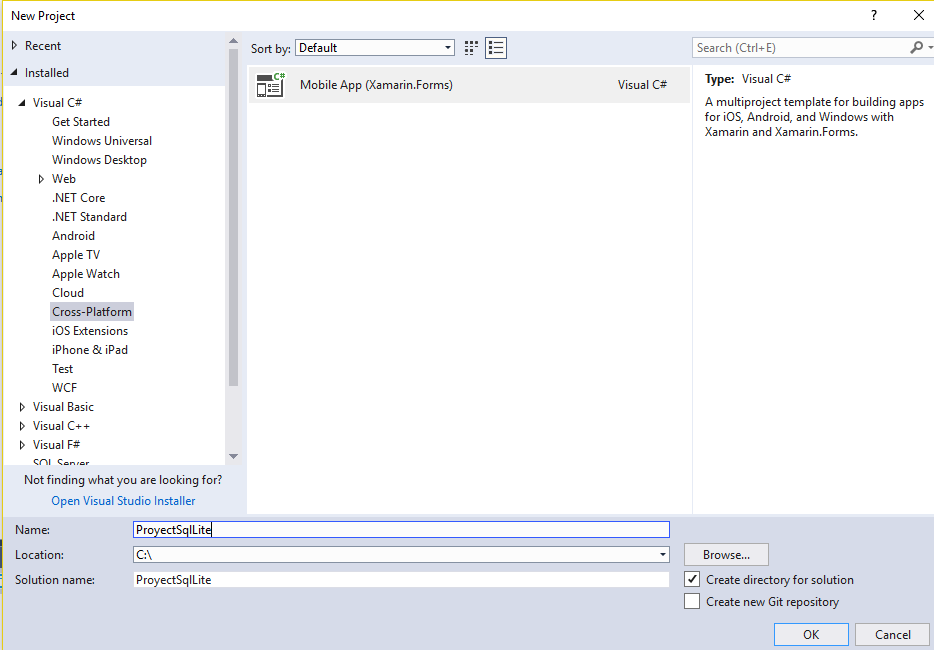
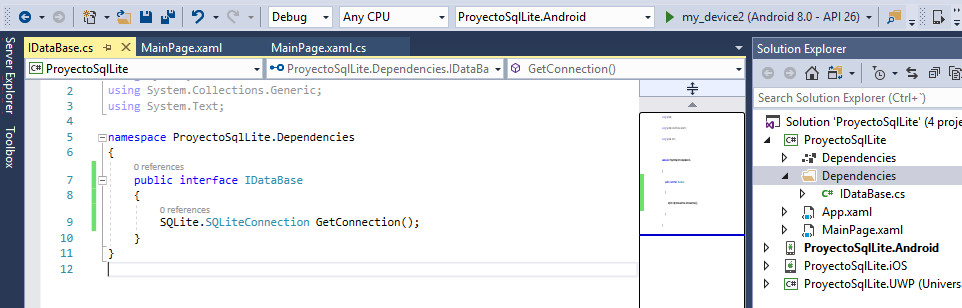
Creamos un proyecto Cross-Platform:

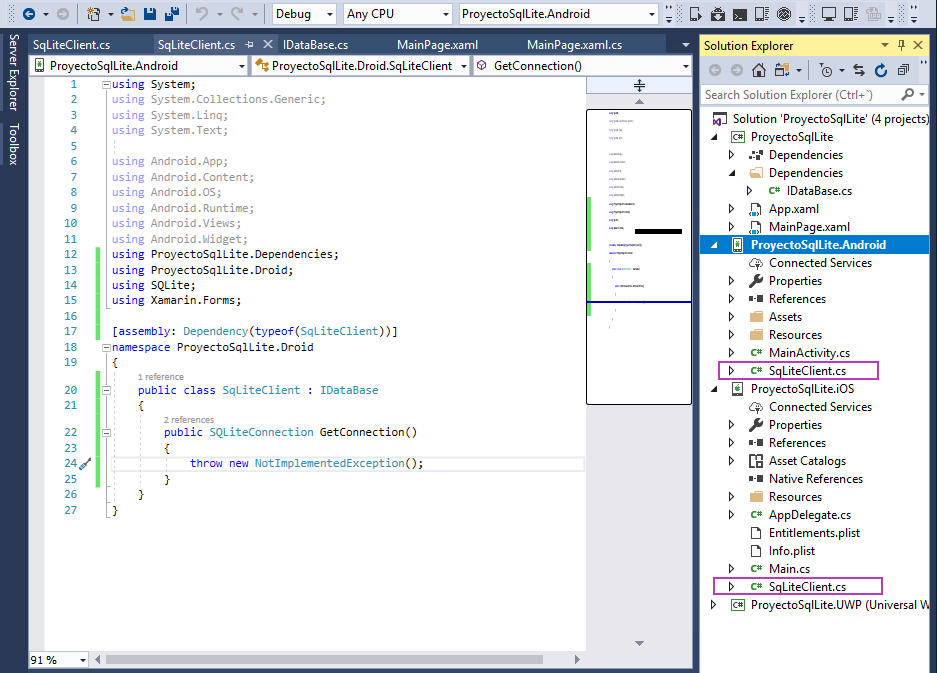


Lo primero que debemos hacer es agregar el nuget: “sqlite-net-pcl” sobre la solución.

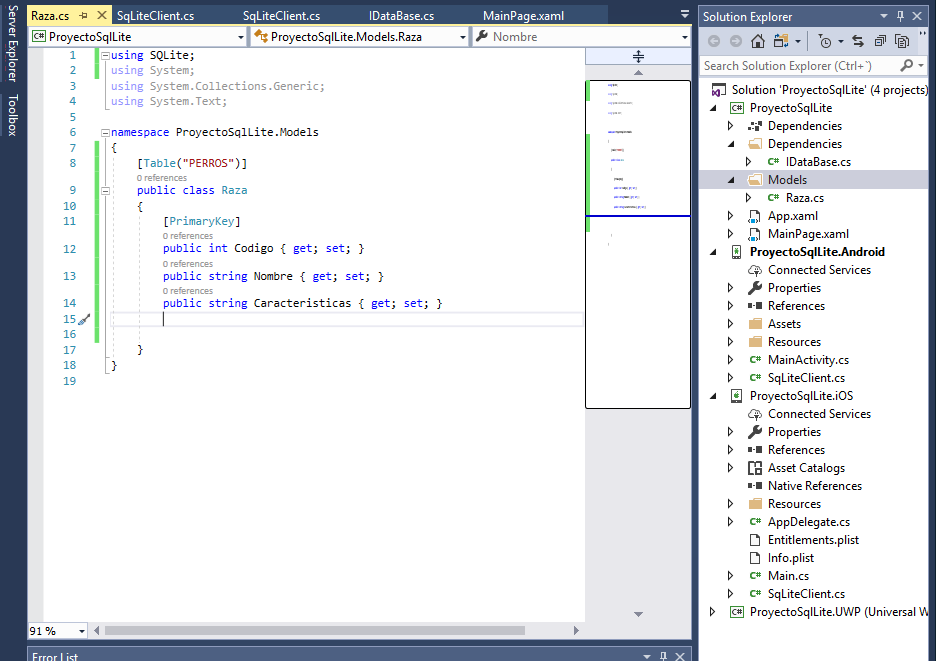
Creamos la carpeta Dependencies y dentro la interface IDataBase, que nos devolverá la conexión a una plataforma o a otra.



Ahora haremos el Servicio de Dependencias: sobre el proyecto de Android y de IOS crearemos una clase llamada SQLiteClient. Al hacer el using sobre Dependency, en [assembly: Dependency(typeof(SqLiteClient))] elegimos Xamarin.Forms. Y heredará de IDataBase. Lo que hará será recuperar de las carpetas.



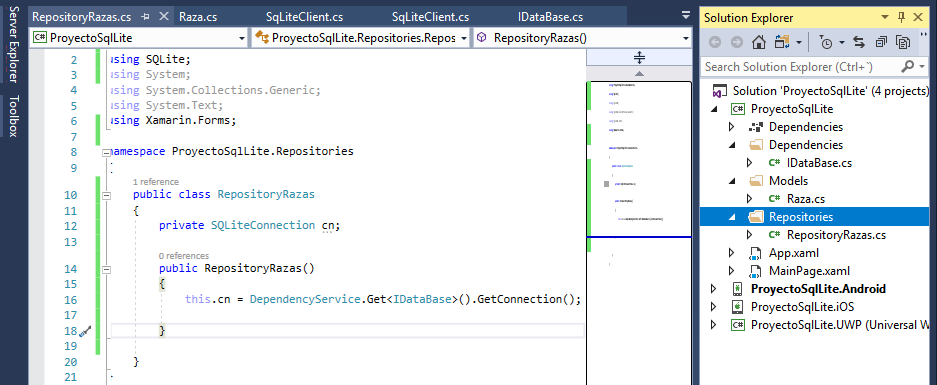
Creamos la carpeta Models y la clase Raza con sus propiedades.



A continuación, creamos el Repository con la clase RepositoryRazas. Todas las acciones se realizan a través de la conexión SQLiteConnection.

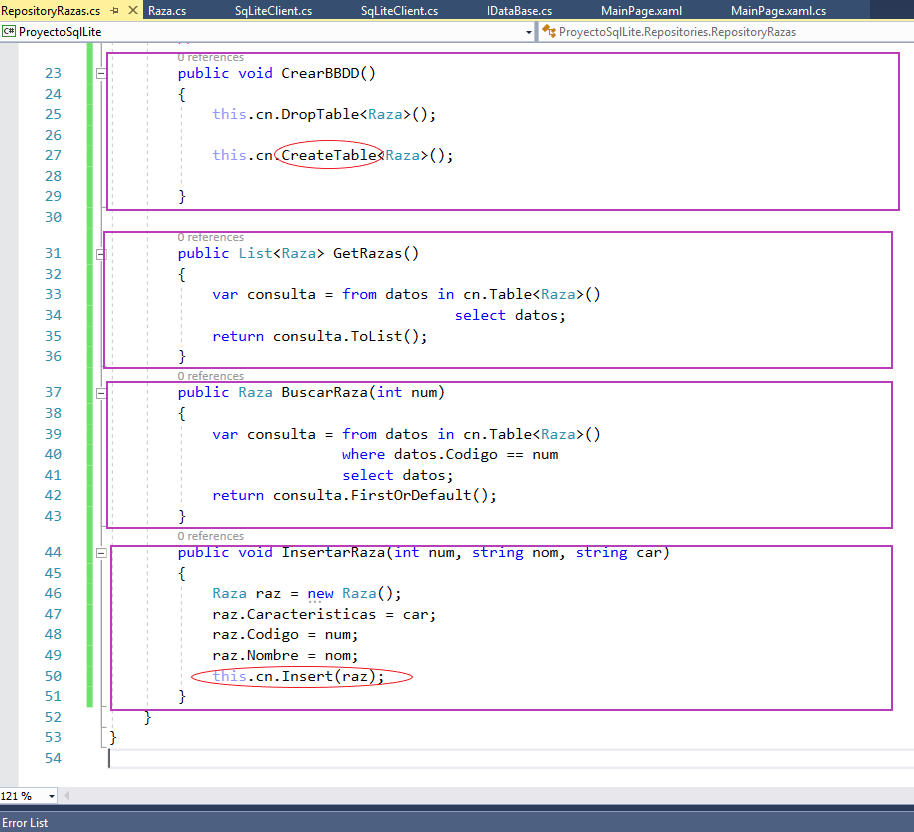
Indicamos en el constructor de donde sacará dicha conexión. Android e IOS lo recuperará de una manera distinta. Internamente leerá si estás en Android o de IOS.

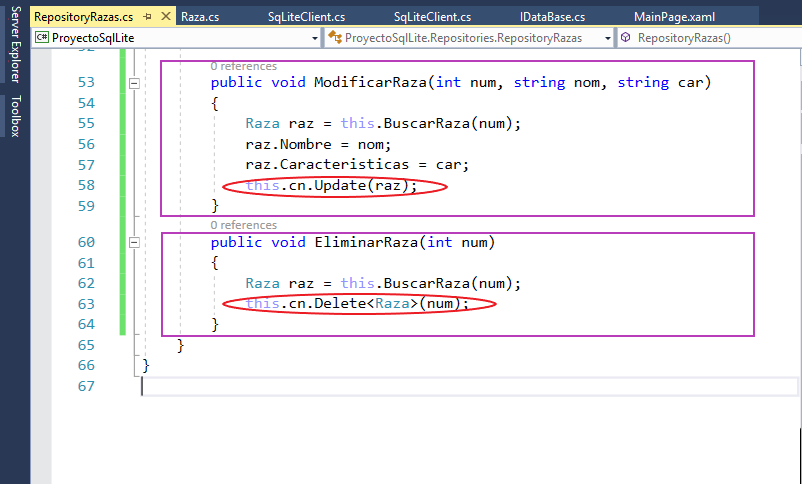
DependencyService.Get<IDataBase>().GetConnection();



Creamos un método para crear la BBDD, todo se crea a partir de la conexión.

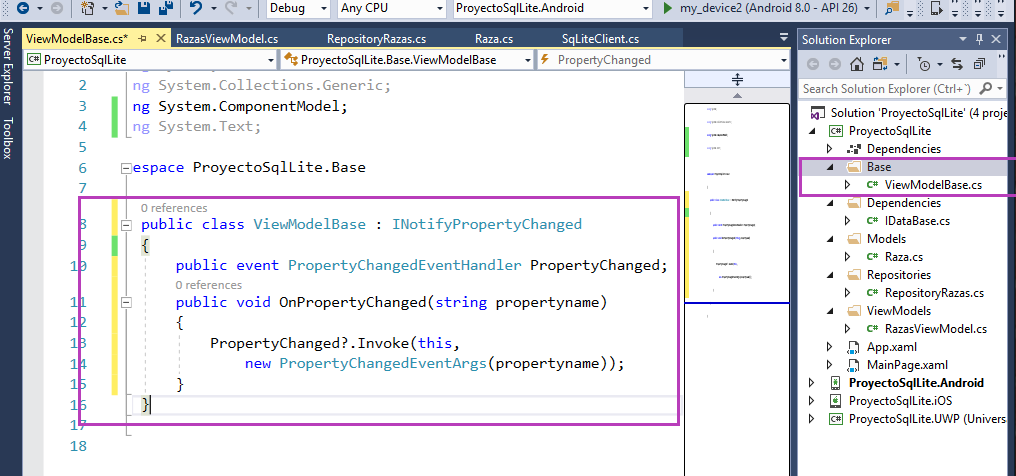
Otro método para la búsqueda, insertar eliminar y modificar:





Creamos la carpeta Base con la clase ViewModelBase, heredará de INotifyPropertyChanged.

Hacemos el método con OnPropertyChanged para que se actualicen las vistas.



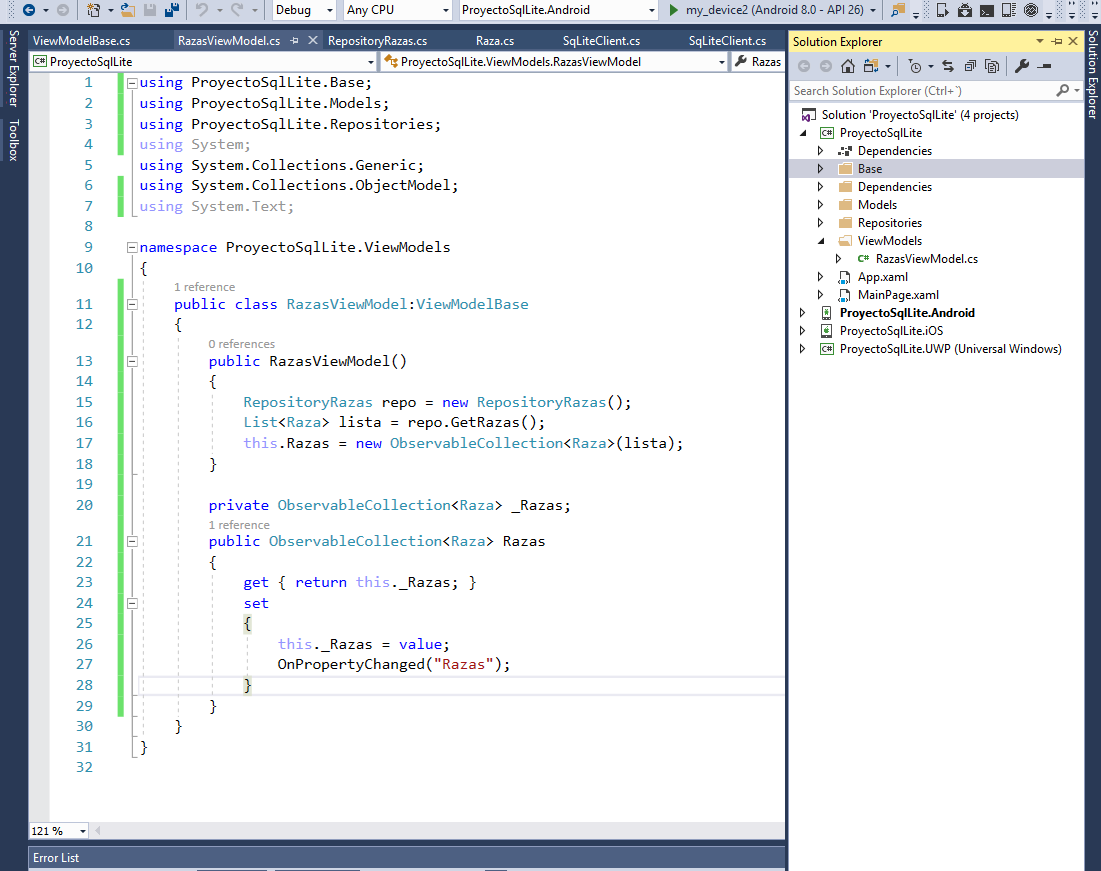
A continuación, vamos a visualizar los elementos.

* ListView para mostrar todos los datos.
* ViewModel para mostrar el conjunto que se llamará RazasViewModel
* ViewModel para el objeto Raza para insertar,modificar,eliminar que se llamará RazaViewModel.

Creamos la carpeta ViewModels y la clase RazasViewModel. Heredará de ViewModelBase.

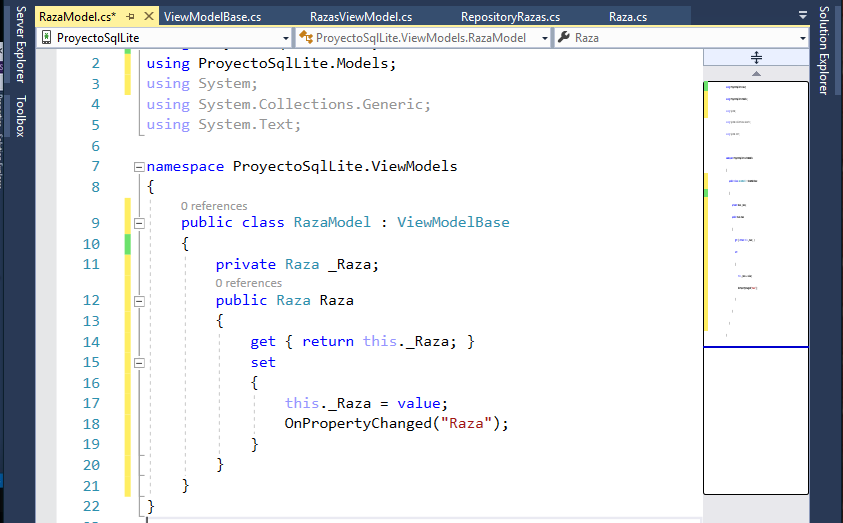
Esta clase tendrá una colección de razas de perros para poder representarlo en cualquier pantalla. Al ser colección en Xamarin lo llamamos ObservableCollection<>. Y creamos la propiedad extendida para que si realizamos algún cambio, podamos verlo con OnPropertyChanged().

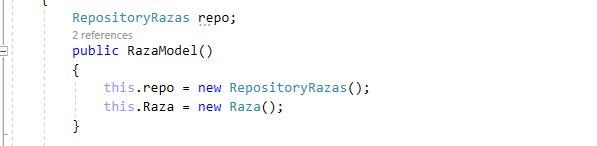
En el constructor cargo todas las razas de los perros y creo la colección para poder visualizarlo.



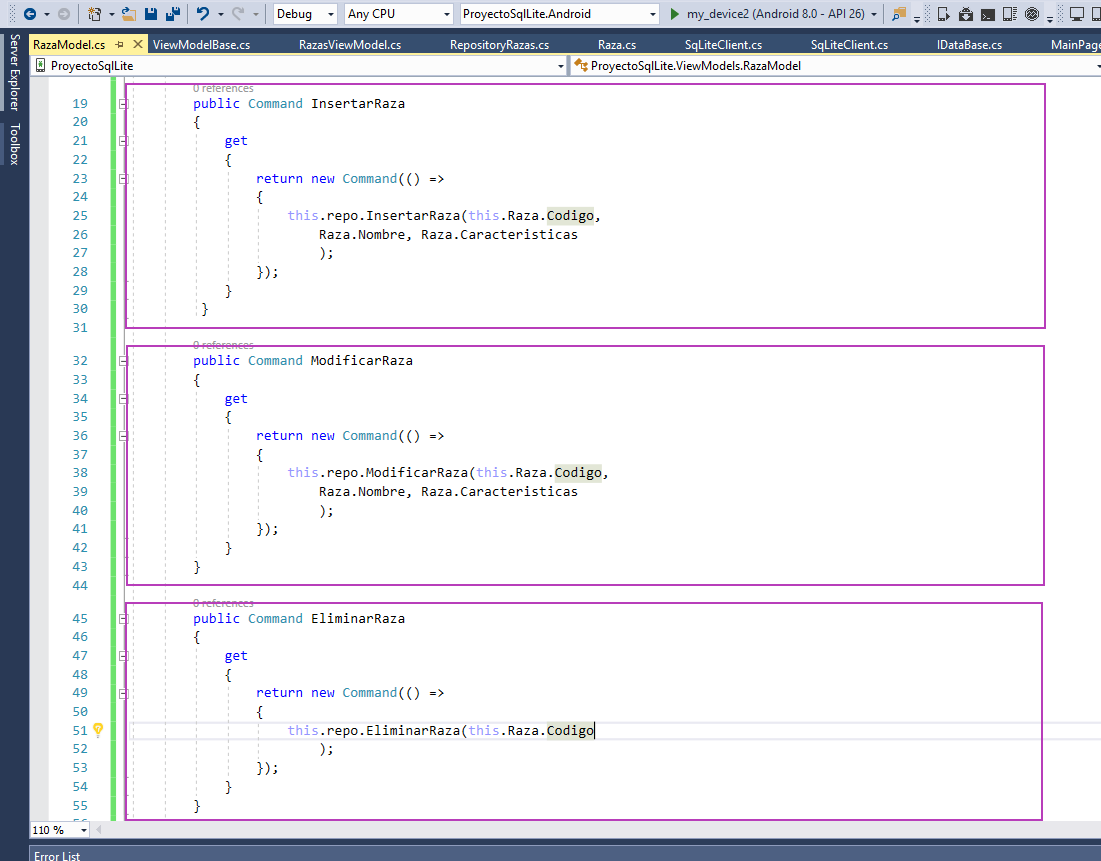
Ahora vamos a crear el ViewModel para una única raza, es decir, vamos a representar eliminar,modificar,insertar.

Creamos la clase RazaModel, que tendrá una propiedad de la clase Raza, será extendida.





A continuación, como queremos insertar, eliminar y modificar, vamos a crear los Command para realizar la acciones en el ViewModel RazaModel.



Ahora vamos a crear las vistas.

Creamos una carpeta Views y una vista InsertarRaza.xaml, irá enlazado con Binding al ViewModel RazaModel.

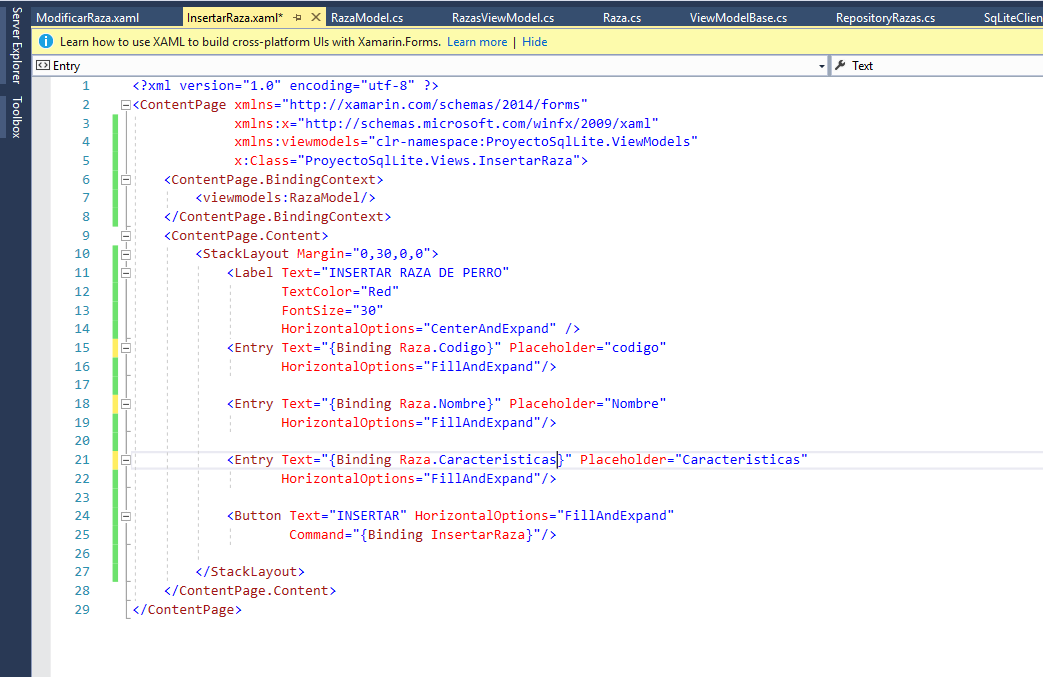
Esta vista creará una nueva raza de perros, entonces creamos el espacio de nombre para viewmodels: xmlns:viewmodels="clr-namespace:ProyectoSqlLite.ViewModels"

y creamos el objeto RazaModels:

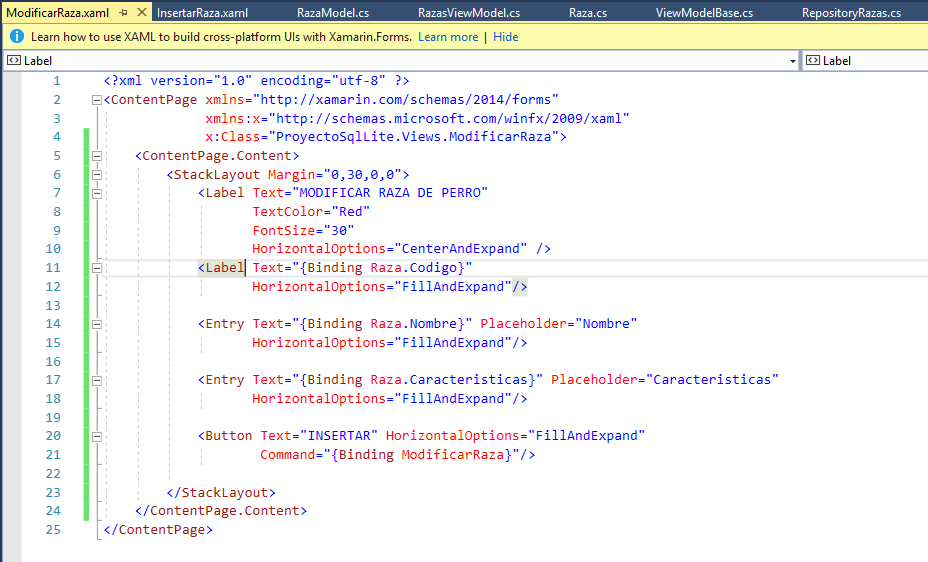
<ContentPage.BindingContext>

<viewmodels:RazaModel/>

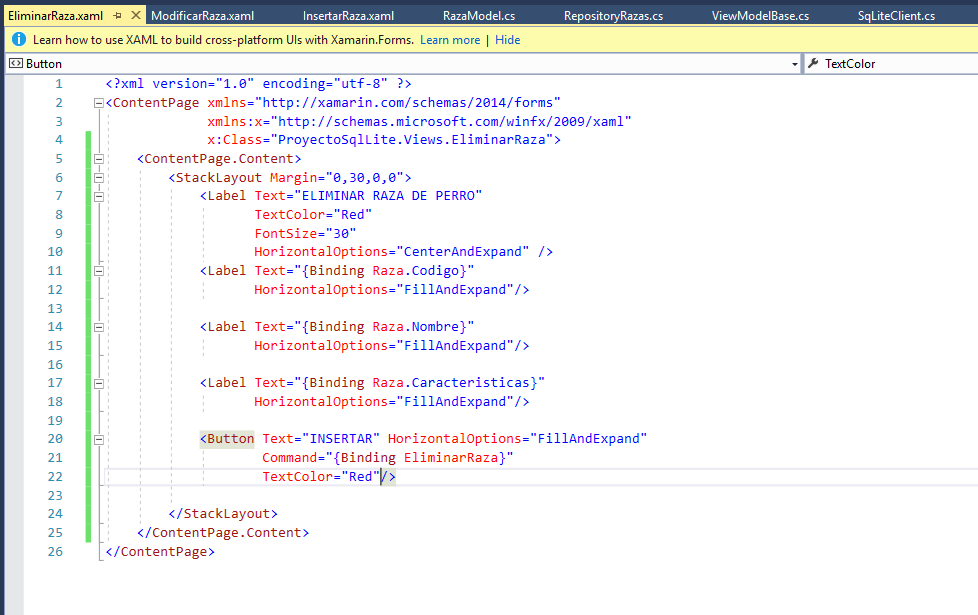
</ContentPage.BindingContext>



A continuación creamos la vista de ModificarRaza.xaml.



Y la vista de EliminarRaza.xaml



Ahora creamos la vista de todas las razas, se llamará RazasView.xaml.

Esta vista irá enlazada a RazasViewModel :

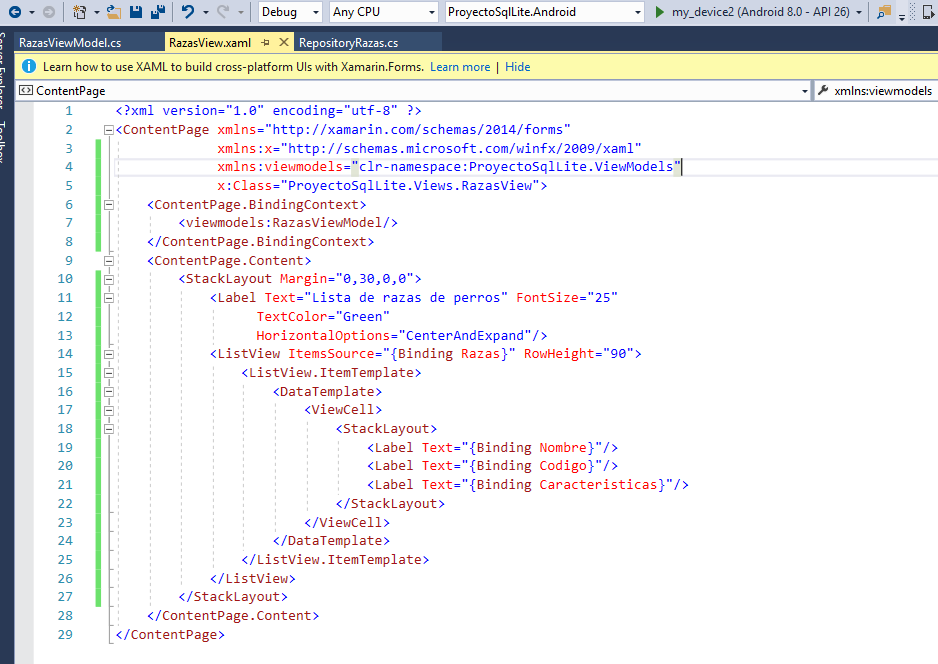
xmlns:viewmodels="clr-namespace:ProyectoSqlLite.ViewModels"

con su respectivo ContentPage.

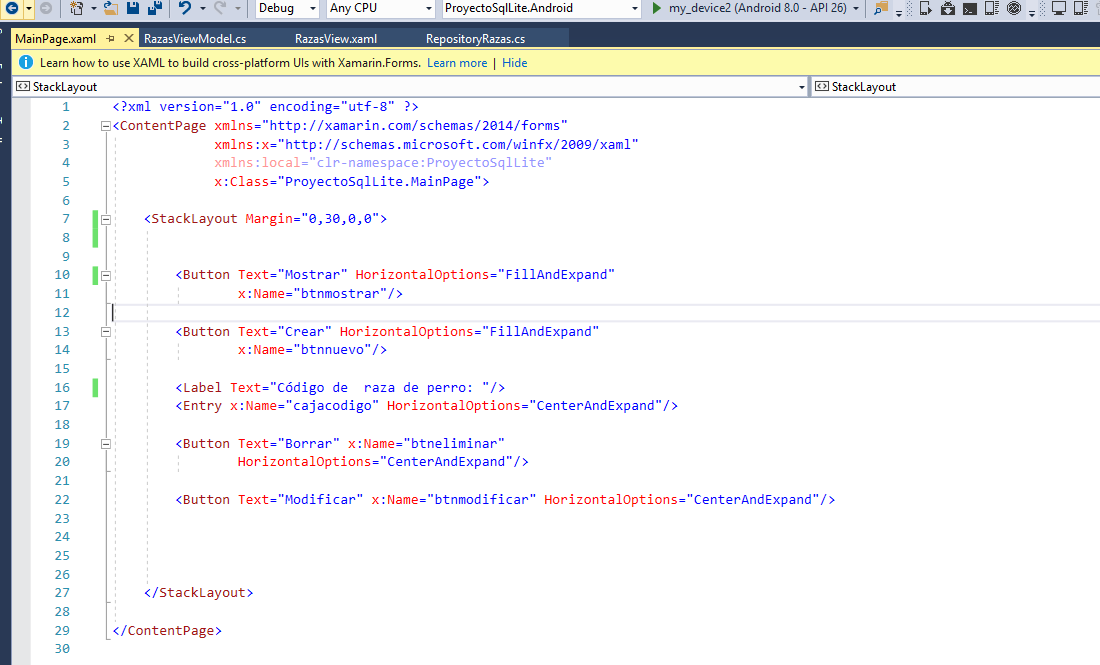
<ContentPage.BindingContext>

<viewmodels:RazasViewModel/>

</ContentPage.BindingContext>



Ahora haremos la llamada desde MainPage.xaml con sus botones.



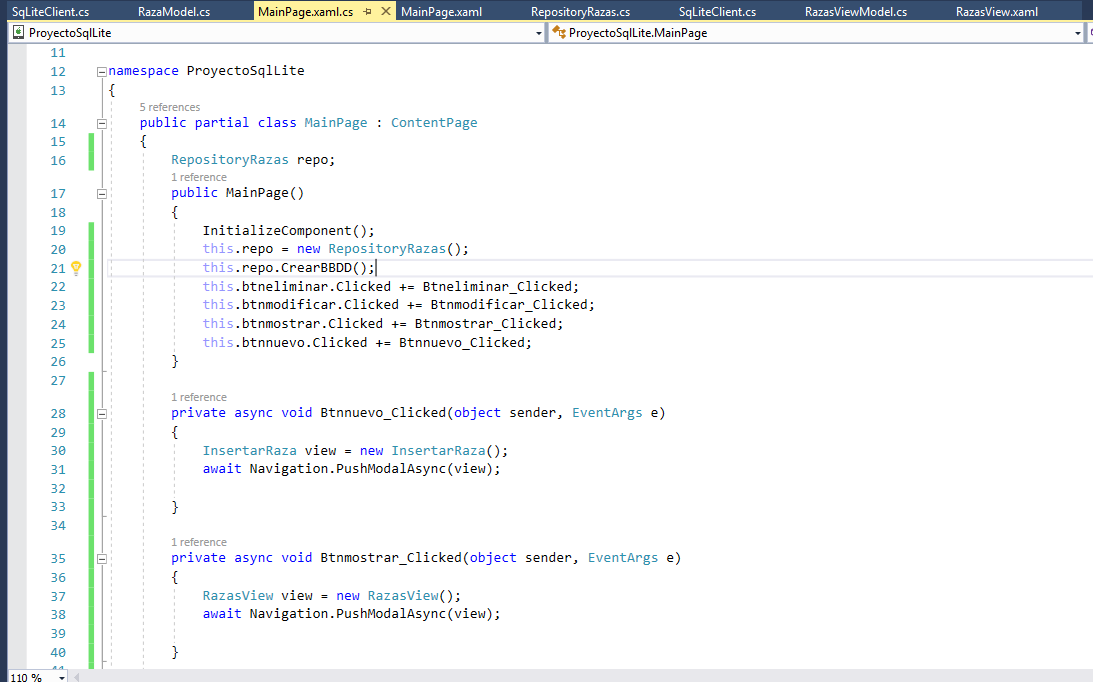
Ahora nos vamos a MainPage.xaml.cs y llamamos a los botones:

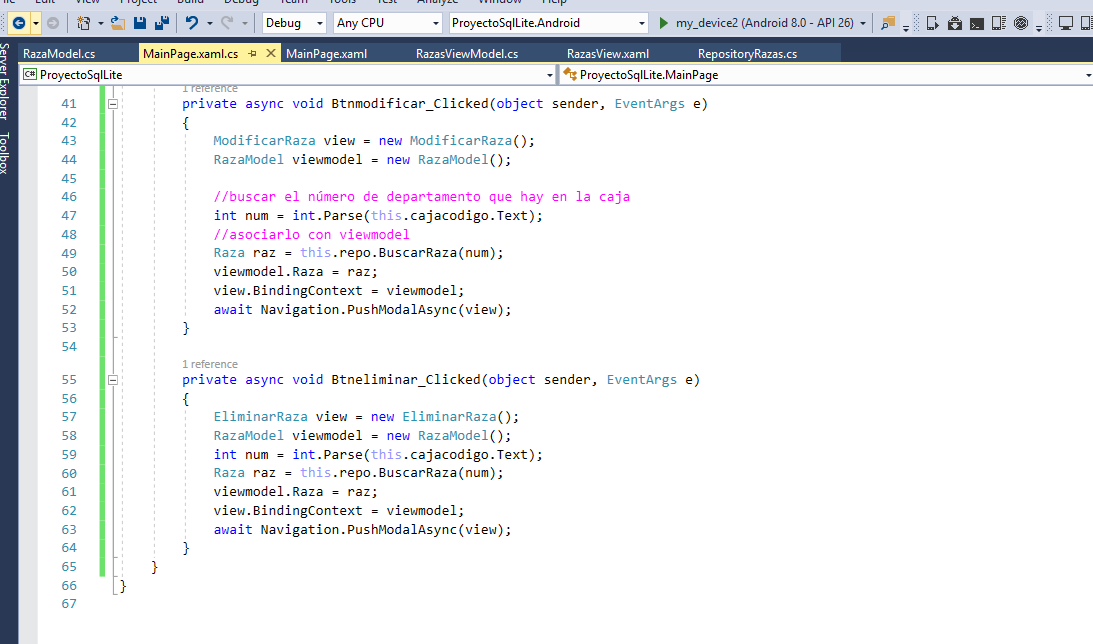
this.btneliminar.Clicked += Btneliminar\_Clicked;🡪 Necesito saber el código

this.btnmodificar.Clicked += Btnmodificar\_Clicked; Necesito saber el código

this.btnmostrar.Clicked += Btnmostrar\_Clicked;🡪mostrará la vista de Razas.

this.btnnuevo.Clicked += Btnnuevo\_Clicked; 🡪mostrará la vista



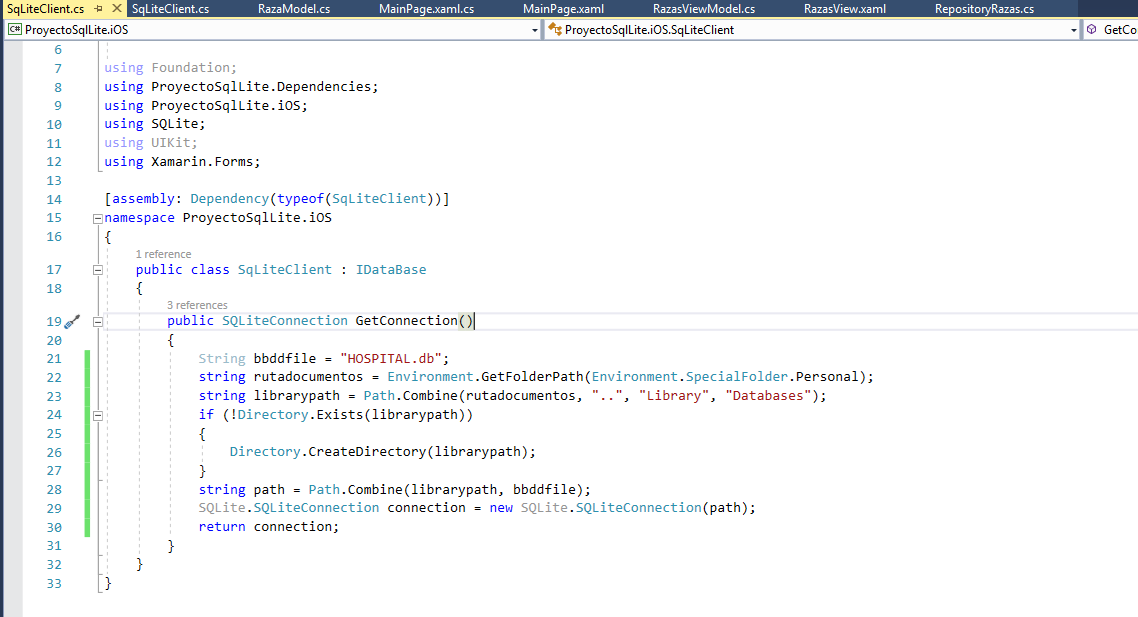


A continuación nos vamos a SqlCliente de Android y de IOS para indicar cómo recuperamos la conexión:

ANDROID:



IOS:



Resultado final:

