Основные операции с SQLite.NET

Последнее обновление: 16.09.2016

Вначале определим класс, объекты которого будут храниться в базе данных. Добавим в главный проект следующий класс Friend:

```
using SQLite;

namespace SQLiteApp
{
    [Table("Friends")]
    public class Friend
    {
        [PrimaryKey, AutoIncrement, Column("_id")]
        public int Id { get; set; }

        public string Name { get; set; }
        public string Email { get; set; }
        public string Phone { get; set; }
    }
}
```

Класс Friend выступает в качестве модели приложения. Этот класс использует атрибуты, которые позволяют настроить его отображение на таблицу в бд. Для настройки мы можем использовать следующие атрибуты:

- **[PrimaryKey]**: применяется к свойству типа `int` и указывает, что столбец в таблице, который соответствует этому свойству, будет выполнять роль первичного ключа. Составные ключи не поддерживаются
- **[AutoIncrement]**: применяется к свойству типа `int` и указывает, что столбец в таблице, который соответствует этому свойству, будет инкрементировать значение на единицу при добавлении нового элемента
- **[Column(name)]**: задает сопоставление свойства со столбцом в таблице, который имеет имя name
- **[Table(name)]**: устанавливает название таблицы, которая будет соотвествовать данному классу
- **[MaxLength(value)]**: устанавливает максимальную длину для строковых свойств

- **[Ignore]**: указывает, что свойство будет игнорироваться. Это может быть полезно, если значение данного свойства не надо хранить в базе данных, и данное свойство не должно сопоставляться со столбцами из таблицы в бд
- **[Unique]**: гарантирует, что столбец, который соответствует свойству с этим атрибутом, будет иметь уникальные неповторяющиеся значения

При запросах к базе данных будет происходить автоматическое сопоставление типов данных из SQLite с типами данных из C#. Сопоставление типов можно описать следующей таблицей:

C#	SQLite
int, long	integer, bigint
bool	integer (1 = true)
enum	integer
float	real
double	real
decimal	real
string	varchar, text
DateTime	numeric, text

byte[]

blob

Класс репозитория

Также создадим класс репозитория, через который будут идти все операции с данными:

```
using System.Collections.Generic;
using System.Linq;
using SQLite;
using Xamarin.Forms;

namespace SQLiteApp
{
    public class FriendRepository
    {
        SQLiteConnection database;
        public FriendRepository(string filename)
        {
            string databasePath = DependencyService.Get<ISQLite>().GetDatabasePath(filename);
            database = new SQLiteConnection(databasePath);
            database.CreateTable<Friend>();
        }
        public IEnumerable<Friend> GetItems()
        {
            return (from i in database.Table<Friend>() select i).ToList();

        }
        public Friend GetItem(int id)
        {
            return database.Get<Friend>(id);
        }
        public int DeleteItem(int id)
        {
            return database.Delete<Friend>(id);
        }
        public int SaveItem(Friend item)
        {
            if (item.Id != 0)
            {
                database.Update(item);
                return item.Id;
            }
            else
            {
                return database.Insert(item);
            }
        }
    }
}
```

В конструкторе класса происходит создание подключения и базы данных (если она отсутствует). Поскольку на конкретных платформах логика создания будет отличаться, то здесь используется метод `DependencyService.Get<ISQLite>()`, позволяющий в зависимости от платформы применить определенную реализацию интерфейса ISQLite.

Для всех операций с данными используются методы, определенные в классе SQLiteConnection:

- **Insert**: добавляет объект в таблицу

- **Get<T>**: позволяет получить элемент типа T по id
- **Table<T>**: возвращает все объекты из таблицы
- **Delete<T>**: удаляет объект по id
- **Update<T>**: обновляет объект
- **Query<T>**: выполняет SQL-выражение и возвращает строки из таблицы в виде объектов типа T (относится к выражениям SELECT)
- **Execute**: выполняет SQL-выражение, но ничего не возвращает (относится к операциям. где не надо возвращать результат - UPDATE, INSERT, DELETE)

Если предполагается, что к базе данных может обращаться сразу несколько потоков, то для блокирования одновременных операций с бд в классе репозитории можно использовать блок **lock** с заглушкой, например:

```
SQLiteConnection database;
static object locker = new object();
//.....................

public int DeleteItem(int id)
{
    lock(locker)
    {
        return database.Delete<Friend>(id);
    }
}
```

Создаваемое подключение будет общим для всего приложения, поэтому изменим файл **App.xaml.cs** следующим образом:

```
using Xamarin.Forms;

namespace SQLiteApp
{
    public partial class App : Application
    {
        public const string DATABASE_NAME = "friends.db";
        public static FriendRepository database;
        public static FriendRepository Database
        {
            get
            {
                if (database == null)
                {
                    database = new FriendRepository(DATABASE_NAME);
                }
                return database;
            }
        }
        public App()
        {
            InitializeComponent();
            MainPage = new NavigationPage(new MainPage());
        }

        protected override void OnStart() { }

        protected override void OnSleep() { }

        protected override void OnResume() { }
```

```
    }
}
```

Статический объект репозитория, создаваемый при создании главной страницы приложения, будет доступен из любого места приложения.

Поскольку в нашем приложении мы будем переходить по страницам - к странице добавления или просмотра, то в качестве главной страницы устанавливается объект `NavigationPage`.

Добавление страниц приложения

Теперь на главной странице MainPage.xaml следующий код:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="SQLiteApp.MainPage" Title="Список друзей">
  <StackLayout>
    <ListView x:Name="friendsList" ItemsSource="{Binding}" ItemSelected="OnItemSelected">
      <ListView.ItemTemplate>
        <DataTemplate>
          <ViewCell>
            <ViewCell.View>
              <StackLayout Orientation="Horizontal">
                <Label Text="{Binding Name}" FontSize="Medium" />
              </StackLayout>
            </ViewCell.View>
          </ViewCell>
        </DataTemplate>
      </ListView.ItemTemplate>
    </ListView>
    <Button Text="Добавить" Clicked="CreateFriend" />
  </StackLayout>
</ContentPage>
```

Элемент ListView будет выводить список объектов, а при нажатии на элемент списка, будет срабатывать обработчик `OnItemSelected`. И также для добавления нового объекта определена кнопка.

И изменим файл кода **MainPage.xaml.cs**:

```
using System;
using Xamarin.Forms;

namespace SQLiteApp
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }
        protected override void OnAppearing()
        {
            friendsList.ItemsSource = App.Database.GetItems();
            base.OnAppearing();
        }
        // обработка нажатия элемента в списке
        private async void OnItemSelected(object sender, SelectedItemChangedEventArgs e)
        {
            Friend selectedFriend = (Friend)e.SelectedItem;
            FriendPage friendPage = new FriendPage();
            friendPage.BindingContext = selectedFriend;
            await Navigation.PushAsync(friendPage);
        }
```

```
        // обработка нажатия кнопки добавления
        private async void CreateFriend(object sender, EventArgs e)
        {
            Friend friend = new Friend();
            FriendPage friendPage = new FriendPage();
            friendPage.BindingContext = friend;
            await Navigation.PushAsync(friendPage);
        }
    }
}
```

При переходе на любую страницу у нее вызывается метод `OnAppearing()`, поэтому тут мы можем установить привязку и настроить другие начальные данные.

Остальные оба обработчика - OnItemSelected и CreateFriend предусматривают переход на страницу FriendPage, которая будет отвечать за работу с одним объектом из бд. Через свойство `BindingContext` мы можем установить полученный объект в качестве контекста страницы, а на самой странице использовать выражения привязки для изменения значений свойств объекта.

Теперь добавим эту страницу **FriendPage**. В ее коде xaml пропишем следующий интерфейс:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="SQLiteApp.FriendPage" Title="Информация о друге">
  <StackLayout>
    <Label Text="Имя" />
    <Entry Text="{Binding Name}" />
    <Label Text="Email" />
    <Entry Text="{Binding Email}" />
    <Label Text="Телефон" />
    <Entry Text="{Binding Phone}" />
    <StackLayout Orientation="Horizontal">
      <Button Text="Сохранить" Clicked="SaveFriend" />
      <Button Text="Удалить" Clicked="DeleteFriend" />
      <Button Text="Отмена" Clicked="Cancel" />
    </StackLayout>
  </StackLayout>
</ContentPage>
```

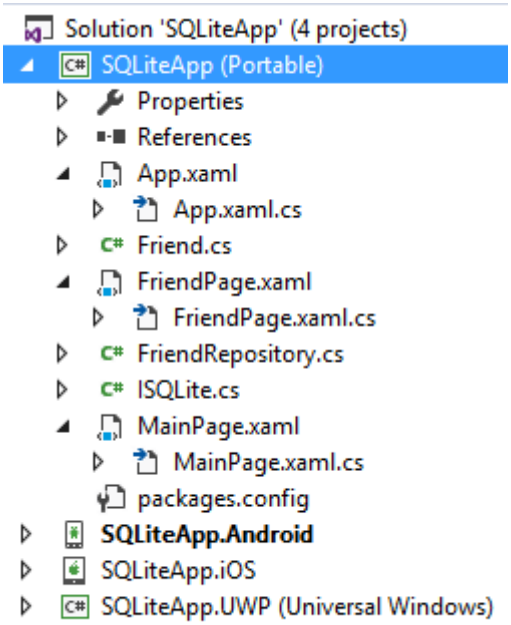
А в файле связанного кода **FriendPage.xaml.cs** добавим обработчики нажатия кнопок:

```
using System;
using Xamarin.Forms;

namespace SQLiteApp
{
    public partial class FriendPage : ContentPage
    {
        public FriendPage()
        {
            InitializeComponent();
        }

        private void SaveFriend(object sender, EventArgs e)
        {
            var friend = (Friend)BindingContext;
            if (!String.IsNullOrEmpty(friend.Name))
            {
                App.Database.SaveItem(friend);
            }
            this.Navigation.PopAsync();
        }
        private void DeleteFriend(object sender, EventArgs e)
```

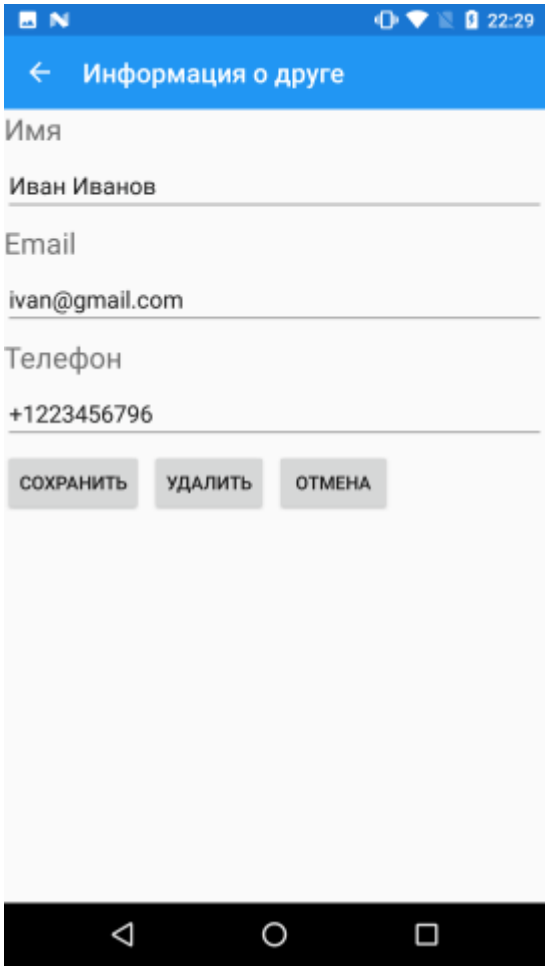
```
        {
            var friend = (Friend)BindingContext;
            App.Database.DeleteItem(friend.Id);
            this.Navigation.PopAsync();
        }
        private void Cancel(object sender, EventArgs e)
        {
            this.Navigation.PopAsync();
        }
    }
}
```

Обработчики используют методы репозитория для сохранения и удаления объекта и после этого осуществляют переход назад на главную страницу.

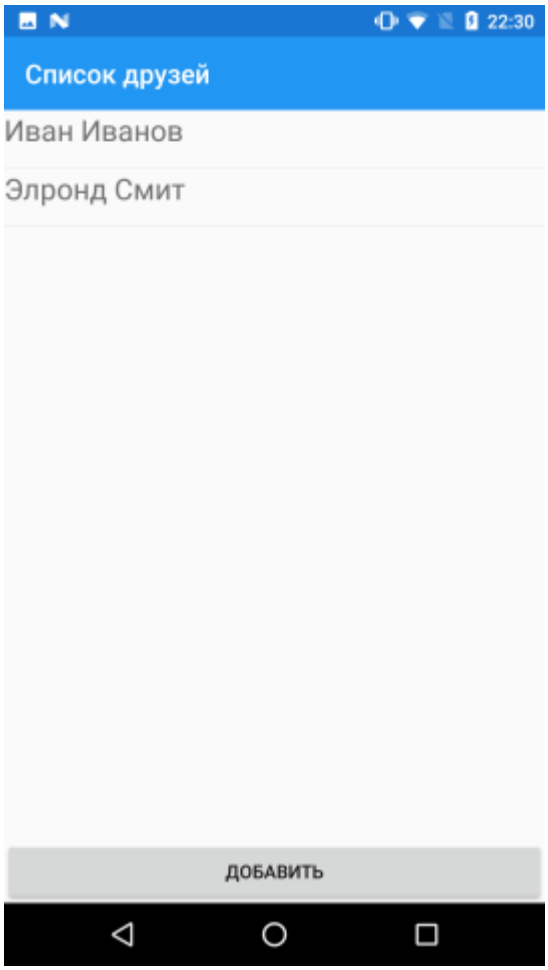
В итоге главный проект будет выглядеть следующим образом:



Теперь с главной страницы мы можем попасть на страницу добавления и создать там новые объекты:



После добавления все объекты будут отображаться в списке на главной странице:



Назад Содержание Вперед

32 Комментариев metanit.com 1 Войти

Рекомендовать Поделиться Лучшее в начале

Присоединиться к обсуждению...

ВОЙТИ С ПОМОЩЬЮ ИЛИ ЧЕРЕЗ DISQUS ?

Имя

Сергій Матвієнко • 2 года назад

Напишите более интересный пример, связь многие ко многим и т.д.
Да и вообще по ксамарин формс очень мало интересных примеров, одни хелло ворды...
Хотелось бы увидеть больше кастомного рендеринга и интересных примеров работы с MVVM

2 • Ответить • Поделиться ›

The Amazing • месяц назад

Unhandled Exception:

System.NullReferenceException: Object reference not set to an instance of an object.

в строке

string databasePath = DependencyService.Get<isqlite>().GetDatabasePath(filename); что не так?

• Ответить • Поделиться ›

Metanit Модератор ➚ The Amazing • месяц назад

а классы, которые реализуют интерфейс ISQLite, вы в каждый проект добавляли, как описано в прошлой теме?

• Ответить • Поделиться ›

The Amazing ➚ Metanit • месяц назад

Да, добавил в каждый

• Ответить • Поделиться ›

nepsy • 7 месяцев назад

Добрый день. С чем может быть связанна данная ошибка?

Error CS0104 'Environment' is an ambiguous reference between 'Android.OS.Environment' and 'System.Environment' SQLiteApp.Android c:\users\rasa.xwsnet\documents\visual studio 2017\Projects\SQLiteApp\SQLiteApp\SQLiteApp.Android\SQLite_Android.cs Line 23 Active

• Ответить • Поделиться ›

Metanit Модератор ➚ nepsy • 7 месяцев назад

в файле подключено два пространства имен 'Android.OS' and 'System', поэтому VS не может определить, из какого пространства брать класс Environment. Уберите подключение Android.OS

1 • Ответить • Поделиться ›

nepsy ➚ Metanit • 7 месяцев назад

Спасибо! И отдельное за Ваш ресурс!

• Ответить • Поделиться ›

Иван Кузьмук • год назад

При создании подключения в строке database = new SQLiteConnection(databasePath); появляется исключение System.TypeInitializationException: The type initializer for 'SQLite.SQLiteConnection' threw an exception.Приложение делаю под андроид и databasePath = /data/data/RepeatingWords. Droid/files/repeatwords.db. В чем может быть причина ошибки? Необходимые пакеты добавлены. И БД я найти не могу, т.е. её по этому пути нет, и нет вообще папки data!

• Ответить • Поделиться ›

Данил Прокопчук ➚ Иван Кузьмук • год назад

как решил данную проблему?

• Ответить • Поделиться ›

Данил Прокопчук ➚ Данил Прокопчук • год назад

Сам отвечу на свой вопрос)) Помогло дабавления пакета" sqlite-net-pcl" в стартап проект (project.Droid).

• Ответить • Поделиться ›

Metanit Модератор → Иван Кузьмук • год назад

попробуйте в качестве пути оставить только непосредственное название файла базы данных

^ v • Ответить • Поделиться ›

Ярослав Орлов • год назад

Добрый вечер, как все это реализовать при помощи ICommand как это было в предыдущих статьях? Пытаюсь сделать, но Id всегда ноль и из за этого при редактирвоании или удалении не передает обьект в команду. Подскажите в чем беда. + В БД почему-то не инкрементятся айдишники.

^ v • Ответить • Поделиться ›

Metanit Модератор → Ярослав Орлов • год назад

неправильна установлена привязка к команде или к параметру команды

^ v • Ответить • Поделиться ›

Ярослав Орлов → Metanit • год назад

Добрый вечер, когда я вытягиваю все записи с БД, то у них у всех ID по 0 непойму почему. Подскажите в какую сторону копать если знаете.

^ v • Ответить • Поделиться ›

Ярослав Орлов → Ярослав Орлов • год назад

Странно, убрав атрибут Column("_Id") начало инкрементиться адекватно и вытягивать записи с бд с нормальными айдишниками.

^ v • Ответить • Поделиться ›

Daria Barysheva • год назад

Добрый день! Большое спасибо за Ваш ресурс. Подскажите, пожалуйста, если будет возможность, следующее. Нет ли какого-то иного способа синхронизировать таблицу БД SQLite с отображающим ее списком на форме в Xamarin, кроме как вызывая при каждой отрисовке страницы LINQ-выражение по извлечению данных из БД (как в Вашем примере: friendsList.ItemsSource = App.Database.GetItems();)? В Руководстве по WPF на Вашем сайте в разделе SQLite в примерах не требовалось повторного извлечения данных в список после изменения БД, но там использовался Entity Framework (и метод db.SaveChanges();). Нельзя ли что-то аналогичное использовать и в Xamarin? Или доступны только либо полная перезагрузка списка, либо ручное изменение нужных элементов в нем?

^ v • Ответить • Поделиться ›

Metanit Модератор → Daria Barysheva • год назад

список объектов (ObservableCollection) надо выделить во ViewModel и к нему установить привязку. Соответственно все операции с бд вынести во ViewModel. При удачном добавлении добавленный элемент добавляется в список во ViewModel. И перезагружать уже тогда не надо.

^ v • Ответить • Поделиться ›

Daria Barysheva → Metanit • год назад

Спасибо за быстрый ответ! Но я уже пробовала ObservableCollection. Все

методы работы с БД у меня во ViewModel. У Вас в примере модель Friend, у меня Writer. Соответственно, вместо IEnumerable<friend> у меня привязка идет к ObservableCollection<writer>. Первый раз при создании ViewModel я получаю список Writers так: Writers = new ObservableCollection<writer>((from i in database.Table<writer>() select i).ToList()). Но даже в таком случае после выполнения операций в БД список сам не обновляется, если его не перегрузить (опять-таки обратившись к БД), хотя в БД данные обновляются (после перезахода в приложение все изменения становятся видны).

Ответить • Поделиться ›

Daria Barysheva ➚ Daria Barysheva • год назад

Понятно, что если после успешного обновления в БД я сама выполню нужное мне добавление или изменение элемента списка - список обновится и перезагружать его полностью из БД уже не надо. Но это единственный способ? Получается все равно нужно и БД менять, и сам список, нет возможности одним изменением это сделать?

Ответить • Поделиться ›

Metanit Модератор ➚ Daria Barysheva • год назад

а как вы еще хотите, как ObservabeCollection узнает, что в базу данных был добавлен объект? Никак. Он узнает, только если эту же ObservableCollection будет также добавлен объект. А добавить его можно только вручную добавить в ObserColl
В случае с WPF и SQLite - там просто операции все шли через кеш.

Ответить • Поделиться ›

Daria Barysheva ➚ Metanit • год назад

Понятно.Спасибо!

Ответить • Поделиться ›

Николай • год назад

Подскажите,пожалуйста, как получить из базы юзера по имени, а не по ID?

Метод в классе-репозитории:
public UserData GetUserByName(string name) { return database.Get<userdata>(name); }

Вызов: UserData user = App.Database.GetUserByName(name);

Так выдает исключение System.InvalidOperationException: Sequence contains no elements.
Передается корректное имя, существующее в базе.

Ответить • Поделиться ›

Metanit Модератор ➚ Николай • год назад

User someUser = df.Get<User>(u => u.Name == "Саша");

Ответить • Поделиться ›

Aleksei Yagelo • год назад

Статью надо обновить

Ответить • Поделиться ›

Metanit Модератор → Aleksei Yagelo • год назад

что именно обновить?

^ | v • Ответить • Поделиться ›

Aleksei Yagelo → Metanit • год назад

Начиная отсюда
"Создаваемое подключение будет общим для всего приложения, поэтому изменим класс App следующим образом:"

Например, OnSleep()
https://developer.xamarin.c...

+ код работы с формами FriendPage.xaml.cs и MainPage.xaml.cs

^ | v • Ответить • Поделиться ›

Makarkin & Partners / Макаркин • 2 года назад

а как правильно реализовать код: При первом запуске юзер выбирает определенное значение и система подгружает нужные настройки (для его выбора) PS: настройки для каждого выбора заранее забиты в config.xml (на сколько безопасно хранить их тут)? А при повторном запуске, система уже НЕ спрашивает а запоминает выбор

^ | v • Ответить • Поделиться ›

Eugene Zarozhny • 3 года назад

и если создаётся база в приложении, то в какой директории она будет находится? в папке с приложением нету

^ | v • Ответить • Поделиться ›

Metanit Модератор → Eugene Zarozhny • 3 года назад

а папку мы в прошлой теме определяли для каждой ос отдельно

^ | v • Ответить • Поделиться ›

Eugene Zarozhny • 3 года назад

а если надо готовую базу подключить, куда её скинуть, в ресурсы?

^ | v • Ответить • Поделиться ›

Metanit Модератор → Eugene Zarozhny • 3 года назад

да, база данных добавляется в ресурсы, а потом при обращении к бд мы смотрим, есть ли файл, если нет, то копируем опять же по тому же пути из ресурсов

^ | v • Ответить • Поделиться ›

Volodymyr → Metanit • 3 года назад

Есть ли у Вас готовый пример? Необходимо подключить БД и выполнить SQL запрос. Запрос должен вернуть 1 ячейку. Значение, хранимое в этой