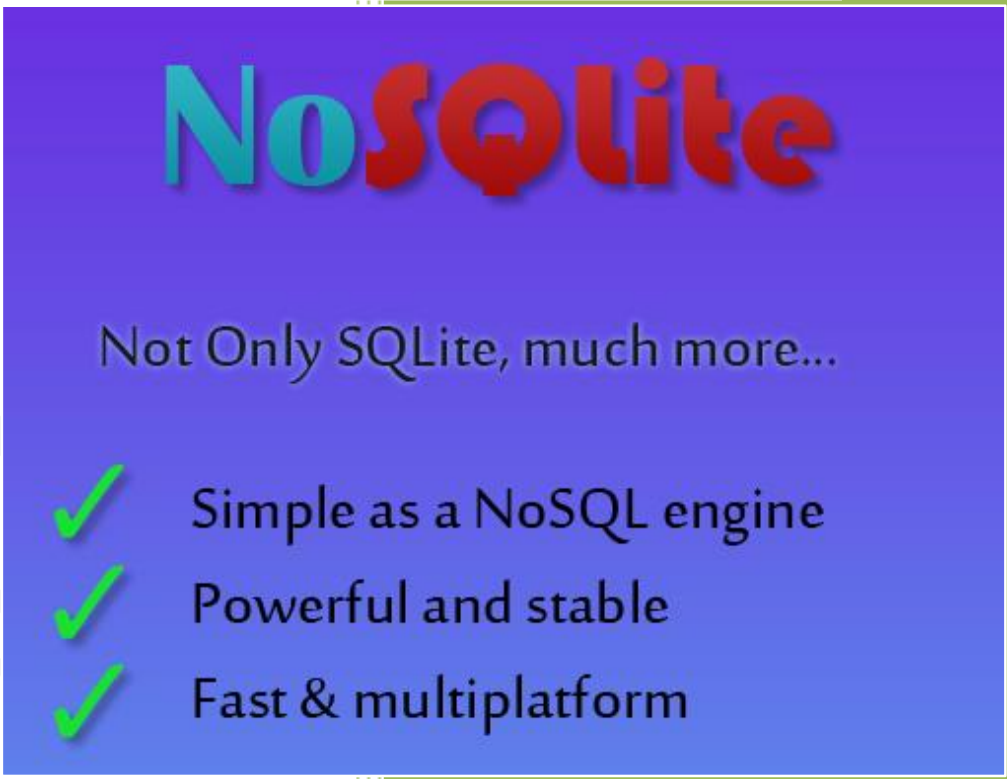
Not Only SQLite

User manual



Asset Repository

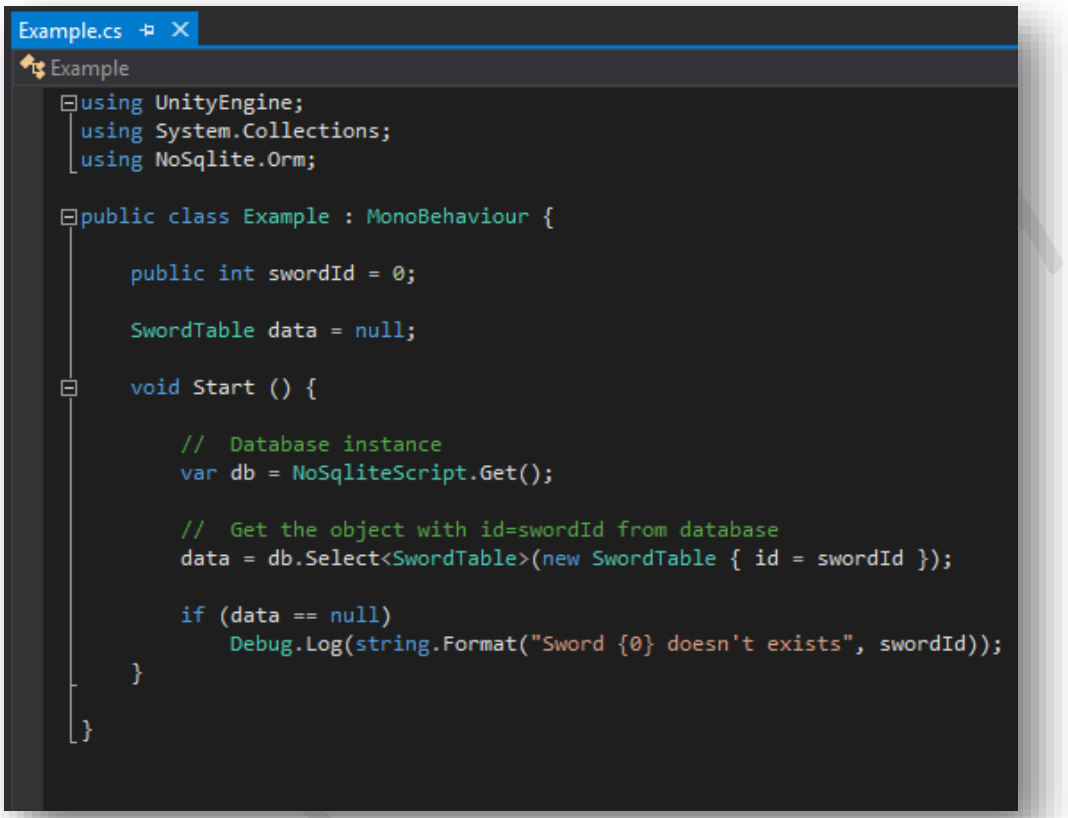
2018-04-26

Index

Introduction

1. Description

NoSQLite is a SQLite ORM for Unity 3D. This tool allows you to work with databases in a simple way, like if you're using NoSQL or object-oriented databases. You can store or get entire objects only with two lines of code.



2. Features

- Works with Unity & Unity Pro.
- Saves entire objects into a SQLite database.
- Auto-creates your object's table if you save one and its table doesn't exists.
- Supports multiple SQLite databases.
- You can execute SQL queries if you want, you are not limited.
- SQLite 3 format, so you can edit your databases in every moment.

3. Compatibility

- **Android & iOS**: Yes
- **Windows, Linux, Mac**: Yes
- **WebGL**: Yes *(*)*
- **Windows Store & Phone**: No

(*) The database in Webplayer/WebGL is executed on memory, so the changes will not be saved.

4. Folders

- **Doc**: Manual and API as PDF. *The PDF version of API is not recommended, to see the last API please go to the API section of this manual.*
- **Examples:** Useful examples for better understanding.
- **Scripts:** Scripts for implement NoSqlite.
- **Sources:** Extensions for NoSqlite.
- **Plugins**: Core of the package.

5. Why SQLite?

SQLite is fast, robust, multiplatform, and open source.

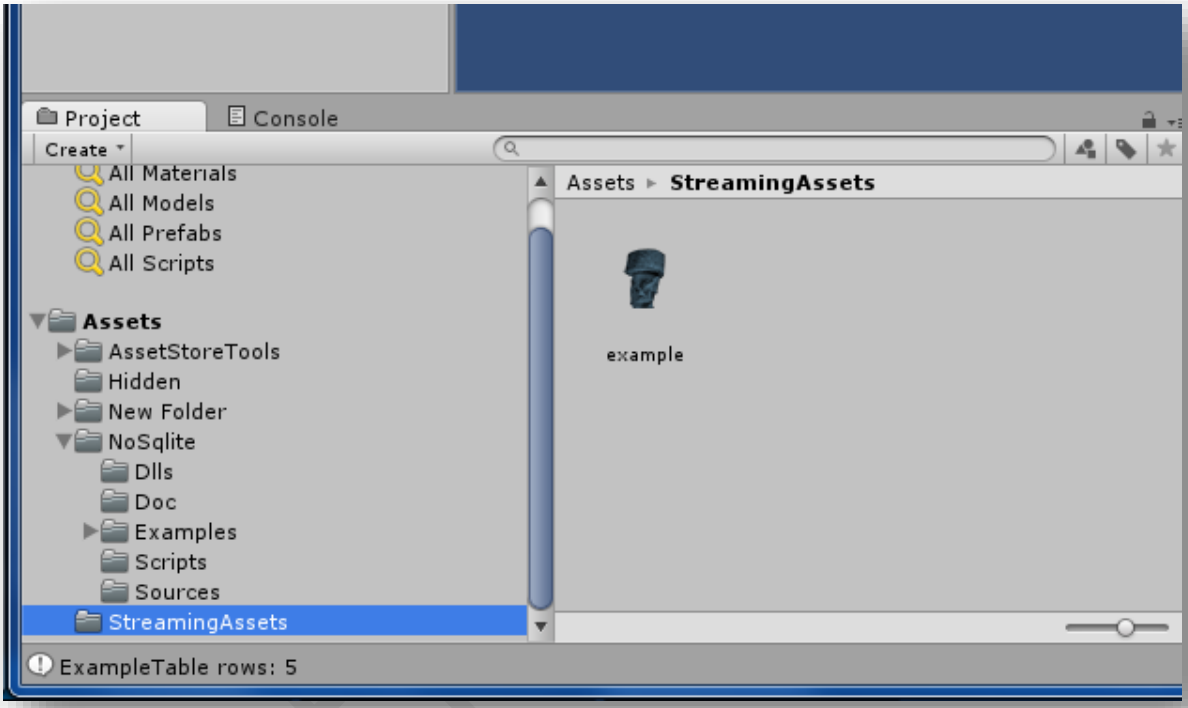
"SQLite is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world. The source code for SQLite is in the public domain." - http://www.sqlite.org

Getting started

1. Create your SQLite database (Optional)

First you need to create a database for SQLite 3. There are a lot of free SQLite editors. My favorite is "SQLite Studio". You can get it here: http://sqlitestudio.pl/

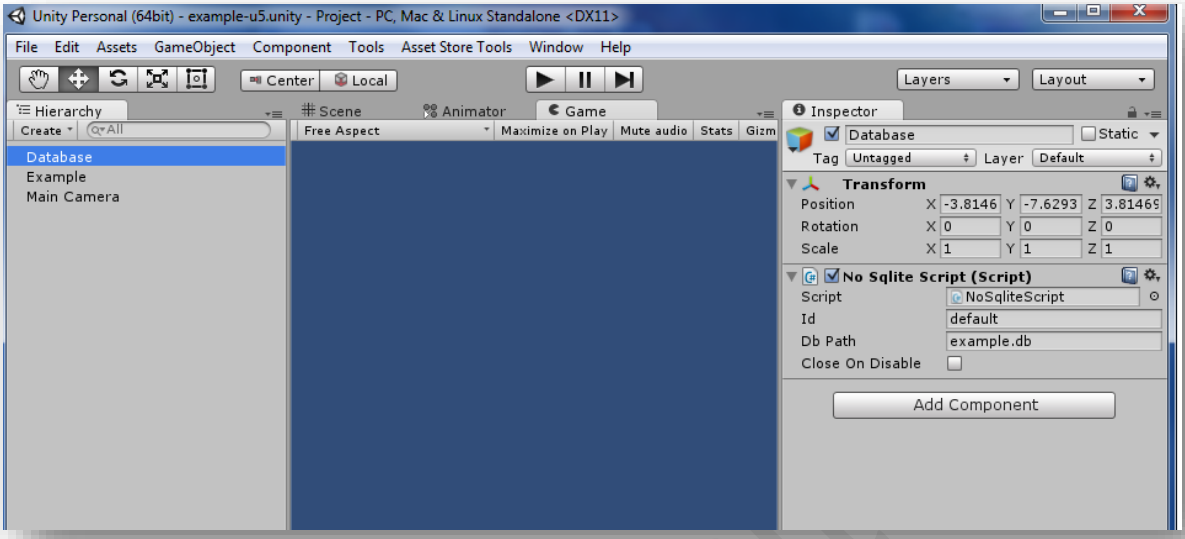
After you've created the database, copy it to /Assets/StreamingAssets (create the directory if doesn't exists).



This step is optional because if the table doesn't exists when you store an object, it is created automatically.

2. Add the Script "NoSqliteScript" to a GameObject

Create a GameObject for the database and then drop the "NoSqliteScript" inside.



3. Create the Entities for the Tables.

Note: for a complete Attributes list go to the API section.

Example 1: simple entity

```
using UnityEngine;
using NoSqlite.Attributes;
using System;

public class SimpleEntityExample {

    [PK]
    public string name;
    public DateTime birthDay;
    public int age;
    public string description;

}
```

The SQLite table will be "SimpleEntityExample", with one Primary Key (PK): "name". Also will have 3 more attributes: birthDay, age and description.

Example 2: complex entity

```
public class ComplexEntityExample
{
    [PK(true)]
    public int id;                      //<     Autoincremental

    [NotNull]
    public SColor color;                //<     Saving a nested object

    public SColor secundaryCollor;      //<     Saving a nested object
                                        //      (Can be null)

    public Status status;               //<     Enum
}
```

The entity "ComplexEntityExample" contains two objects inside (color, secundaryColor) serialized as strings, and one enum.

To save objects as string, that objects must implement the interface: ISerializable.

```
#region ISerializable implementation

public string Serialize()
{
    return string.Join(",", new string[] {
    ((int)Math.Round(color.r*255)).ToString(),
    ((int)Math.Round(color.g*255)).ToString(),
    ((int)Math.Round(color.b*255)).ToString(),
    ((int)Math.Round(color.a*255)).ToString()
});
}

public object Deserialize(string str)
{
    if (string.IsNullOrEmpty(str))
        return new SColor(Color.black);

    string[] c = str.Split(',');
    return new SColor(new Color(
        (int.Parse(c[0])) / 255f,
        (int.Parse(c[1])) / 255f,
        (int.Parse(c[2])) / 255f,
        (int.Parse(c[3])) / 255f
    ));
}

#endregion
```

```
public class SColor : ISerializable
{
    public Color color;

    public SColor(Color c)
    {
        color = c;
    }
    public SColor()
    {
    }

    ISerializable implementation
}
```

Finally, here's the enum.

```
public enum Status
{
    Disabled = 0,
    Enabled = 1,
    Deleted = 2
}
```

API

You can find the API here.

Credits

Asset Repository

by @lcnvdl

Do you need a feature? Do you need support?

Contact us! ☺

http://unity.lucianorasente.com