KIRUPA.COM

TUTORIALS FORUMS

Question of the Week

What non-geek hobbies do you have?

Flash
- Basics / Animation
- ActionScript
- Special Effects

HTML5
- CSS / HTML Basics
- Animation
- JavaScript
- Common Tasks

Windows Phone

Get Help

SQLite in VB.net - Page 1

by Austin Andrews aka Templarian | 3 May 2008

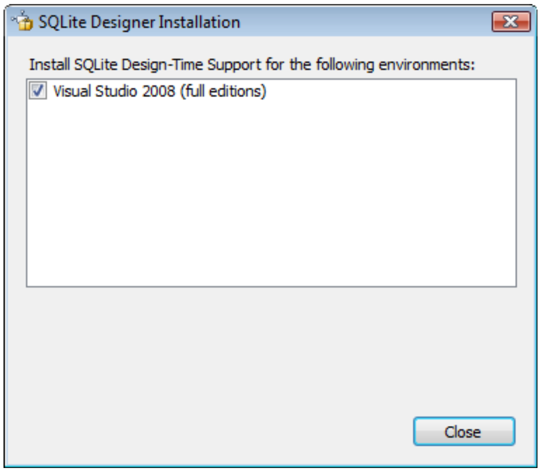
SQLite is a very flexible relational database that allows users to store rows of data for their programs. Because it is highly used, database format wrappers are found in almost all languages for it. This tutorial will focus on the basics of installing and using the most popular Visual Basic .NET wrapper System.Data.SQLite - created and maintained by Robert Simpson. If you are familiar with ADO.NET you will find this very similar. Don't worry if you aren't though, for it it is quite simple to use.

Installation of System.Data.SQLite

For this step you will need any version of Microsoft Visual Studio 2005/2008. The tutorial is written with Visual Studio 2008 in mind, but all steps are the same on either version. Please follow the link below to download System.Data.SQLite off the official site: http://sqlite.phxsoftware.com/

> **Note**
>
> Make sure to download the latest non-Alpha version, and you will want to download the “SQLite-X.X.XX.X-setup.exe” as this is the quickest method of installation.

Click next through the installation wizard till you get to this screen:



SQLite Designer Installation

Install SQLite Design-Time Support for the following environments:

☑ Visual Studio 2008 (full editions)

For those using VS 2008 Express Design-Time support does not work, so do not install it. This is not a glitch, it is disabled by Microsoft in the Express edition of 2008, but does work in the 2005 Express Edition. This is not used in the tutorial so it will not matter either way if you have it installed it.

Installing the SQLite Design-Time Support is optional for those that need to test queries. It is rather easy to use and understand, and I’m sure you won’t need it after this tutorial or if you know the basics of SQL already.

Create a VB.NET New Project

Let’s start up Visual Studio 2005/2008 and create a New Visual Basic » Windows Form Application. I created this with .NET 2.0 as it is the lowest version of .NET that is supported.

Creating a Database

Check out Kirupa's new book!!!

First understand that a database is a completely empty file when it's first created, and if the database isn't found where you open the connection it will create a new file.
First create a button and name it **btn_createdb** and double click the button to bring up the code viewer.

The code that you will be placing into the button basically opens up the Windows save dialog so that you can simply save your database where you want:

```
Private Sub btn_createdb_Click(ByVal sender As System.Object, ByVal e
        'Save Dialog Box
        Dim f As New SaveFileDialog
        f.Filter = "SQLite 3 (*.db3)|*.db3|All Files|"
        f.ShowDialog()
        'Create Database
        Dim SQLconnect As New SQLite.SQLiteConnection()
        'Database Doesn't Exist so Created at Path
        SQLconnect.ConnectionString = "Data Source=" & f.FileName & "
        SQLconnect.Open()
        SQLconnect.Close()
    End Sub
```

When saving a database, the standard extension is .db3, but if you are using this for back up purposes, you may choose your own extension.

Remember that in a real example, you will usually never create an empty database.

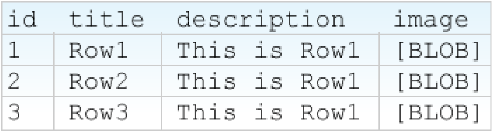
Creating a Table

The first thing to understand is your basic data types that you have to work with. SQLite, being a very compact database, has 4 main (not counting NULL) such data types. This makes it really simple to know which type your data should be in.

- **INTEGER**
  The value is a signed integer, stored in 1, 2, 3, 4, 6, or 8 bytes depending on the magnitude of the value.
- **REAL**
  The value is a floating point value, stored as an 8-byte IEEE floating point number.
- **TEXT**
  The value is a text string, stored using the database encoding (UTF-8, UTF-16BE or UTF-16-LE).
- **BLOB**
  The value is a blob of data, stored exactly as it was input.

(Source: http://www.sqlite.org/datatype3.html)

What you will do is add some code to create a table. Before delving into the code, let's take a look at what your table will look like first:



id	title	description	image
1	Row1	This is Row1	[BLOB]
2	Row2	This is Row1	[BLOB]
3	Row3	This is Row1	[BLOB]

Now that you know what we will be creating, let's add it. First start by creating a button called **btn_createtable**. Double click the button to be taken to the code view where you will be inside this button's event handler.

Copy and paste the following code:

```
    Dim f As New OpenFileDialog
    f.Filter = "SQLite 3 (*.db3)|*.db3|All Files|*.*"
    If f.ShowDialog() = DialogResult.OK Then
        Dim SQLconnect As New SQLite.SQLiteConnection()
        Dim SQLcommand As SQLiteCommand
        SQLconnect.ConnectionString = "Data Source=" & f.FileName & "
        SQLconnect.Open()
        SQLcommand = SQLconnect.CreateCommand
        'SQL query to Create Table
        SQLcommand.CommandText = "CREATE TABLE foo(id INTEGER PRIMARY
        SQLcommand.ExecuteNonQuery()
        SQLcommand.Dispose()
        SQLconnect.Close()
    End If
```

You created a table called foo, and if you were to visualize it, it looks just like the image of the table I showed you earlier. Ok, let's take a small break and continue on the next page.

Onwards to the next page!

Previous | Next

1 | 2

SQLite in VB.net - Page 2

by Austin Andrews aka Templarian | 3 May 2008

In the previous page, we started looking at some simple database operations you can do using SQLite. In this page, let's continue doing more of the same.

Inserting a Record (Update/Delete)

Just like in all SQL, if you know how to insert a record, you can change the SQL query to update and delete a record. Create another button called **btn_insert** and add the fallowing code:

```
    Dim f As New OpenFileDialog
    f.Filter = "SQLite 3 (*.db3)|*.db3|All Files|*.*"
    If f.ShowDialog() = DialogResult.OK Then
        Dim SQLconnect As New SQLite.SQLiteConnection()
        Dim SQLcommand As SQLiteCommand
        SQLconnect.ConnectionString = "Data Source=" & f.FileName & "
        SQLconnect.Open()
        SQLcommand = SQLconnect.CreateCommand
        'Insert Record into Foo
        SQLcommand.CommandText = "INSERT INTO foo (title, description
        'Update Last Created Record in Foo
        SQLcommand.CommandText = "UPDATE foo SET title = 'New Title',
        'Delete Last Created Record from Foo
        SQLcommand.CommandText = "DELETE FROM foo WHERE id = last_ins
        SQLcommand.ExecuteNonQuery()
        SQLcommand.Dispose()
        SQLconnect.Close()
    End If
```

Reading Records into a List

What's the use of putting all this data into a database if you cannot show it off? First create a button called "btn_readrecord", and a listbox called "lst_records", place the code below into the button:

```
Dim f As New OpenFileDialog
    f.Filter = "SQLite 3 (*.db3)|*.db3|All Files|*.*"
    If f.ShowDialog() = DialogResult.OK Then
        Dim SQLconnect As New SQLite.SQLiteConnection()
        Dim SQLcommand As SQLiteCommand
        SQLconnect.ConnectionString = "Data Source=" & f.FileName & "
        SQLconnect.Open()
        SQLcommand = SQLconnect.CreateCommand
        SQLcommand.CommandText = "SELECT * FROM foo"
        Dim SQLreader As SQLiteDataReader = SQLcommand.ExecuteReader(
        lst_records.Items.Clear()
        While SQLreader.Read()
            lst_records.Items.Add(String.Format("ID = {0}, Title = {1
            'Optionally use the variable names to identify columns
            'De lst_records.Items.Add(String.Format("ID = {0}, Title =
        End While
        SQLcommand.Dispose()
        SQLconnect.Close()
    End If
```

Reading Tables into a Drop Down

Most projects require multiple tables to store all the relational information. And most

of the times the tables are used to organize categories, so a way to find these tables is needed as shown with the code following code example.

But, first, create a button called **btn_tables**, and a Combo Box called **cmb_tables.** Once you have done that, then copy and paste the following code:

```
    Dim f As New OpenFileDialog
    f.Filter = "SQLite 3 (*.db3)|*.db3|All Files|*.*"
    If f.ShowDialog() = DialogResult.OK Then
        Dim SQLconnect As New SQLite.SQLiteConnection()
        Dim SQLcommand As SQLiteCommand
        SQLconnect.ConnectionString = "Data Source=" & f.FileName & "
        SQLconnect.Open()
        SQLcommand = SQLconnect.CreateCommand
        Dim SchemaTable = SQLconnect.GetSchema(SQLiteMetaDataCollecti
        For int As Integer = 0 To SchemaTable.Rows.Count - 1
            If SchemaTable.Rows(int)!TABLE_TYPE.ToString = "table" Th
           cmb_tables.Items.Add(SchemaTable.Rows(int)!TABLE_NAME.ToSt
            End If
        Next
        SQLcommand.Dispose()
        SQLconnect.Close()
    End If
```

Writing/Reading a BLOB Image into the Table

I have left actually writing and reading an image out of the examples thus far because it’s optional and easier to explain out of context. With that said, it is hard enough that one would be mad if I left it out. If you are interested to see this included into the other source, it is a highly recommended to download the source example I've provided at the end of this tutorial that is based on the sample code you see here.

For this, simply create a button called **btn_insertimage**, a button called **btn_updateimage**, a button called **btn_viewimage**, and a picture box called **pic_viewer**. This example creates a database with a table called foo with 2 columns and image BLOB. In the table, one record is inserted and viewable. The image is always entered into where id is equal to 1.

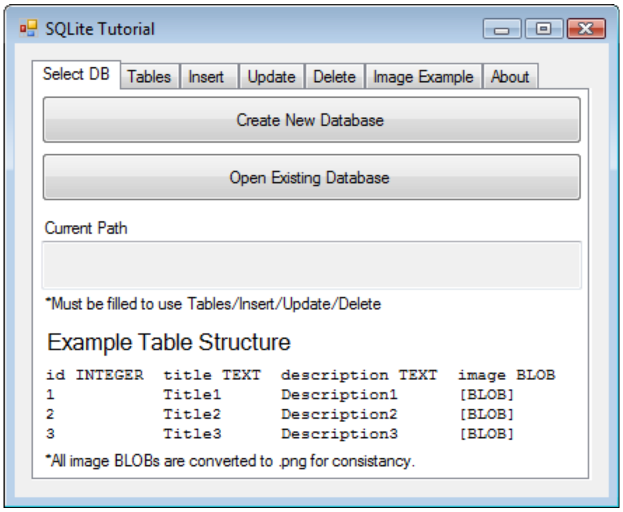
The code is:

```
Imports System.Data.SQLite
Imports System.IO

Public Class sqlitetutorial

    'Image BLOB Functions
    Private Function BlobToImage(ByVal blob)
        Dim mStream As New System.IO.MemoryStream
        Dim pData() As Byte = DirectCast(blob, Byte())
        mStream.Write(pData, 0, Convert.ToInt32(pData.Length))
        Dim bm As Bitmap = New Bitmap(mStream, False)
        mStream.Dispose()
        Return bm
    End Function

    Public Overloads Function ImageToBlob(ByVal id As String, ByVal f
        Dim fs As FileStream = New FileStream(filePath, FileMode.Open
        Dim br As BinaryReader = New BinaryReader(fs)
        Dim bm() As Byte = br.ReadBytes(fs.Length)
        br.Close()
        fs.Close()
        'Create Parm
        Dim photo() As Byte = bm
        Dim SQLparm As New SQLiteParameter("@image", photo)
        SQLparm.DbType = DbType.Binary
        SQLparm.Value = photo
        Return SQLparm
    End Function

    'NOT USED IN THE SOURCE.
    Public Overloads Function ImageToBlob(ByVal id As String, ByVal i
        Dim ms As New MemoryStream()
        image.Save(ms, System.Drawing.Imaging.ImageFormat.Png)
        'Create Parm
        Dim photo() As Byte = ms.ToArray()
        Dim SQLparm As New SQLiteParameter("@image", photo)
        SQLparm.DbType = DbType.Binary
        SQLparm.Value = photo
        Return SQLparm
```

Summary

Hopefully with all these code examples and short explanations, you should be better prepared for your future projects that use SQLite.

Download the source for this project. The source example is a fully working database editor for a simple database with an id, title, description, and image fields:



*Must be filled to use Tables/Insert/Update/Delete

Example Table Structure

```
id INTEGER  title TEXT  description TEXT  image BLOB
1           Title1      Description1      [BLOB]
2           Title2      Description2      [BLOB]
3           Title3      Description3      [BLOB]
```

*All image BLOBs are converted to .png for consistancy.

All of which are stored in the databases. It is highly recommended to read through the source of the example. If you have any questions, please post on the forums.

Austin Andrews aka Templarian

Previous | Next

1 | 2

Home | About kirupa.com | Meet the Moderators | Advertise | Link to Us | Credits