Compiling and using SQLite with Windows Phone 8

This article explains how to compile and use public SQLite with Windows Phone 8.

Introduction

This wiki explains how to use SQLite in Windows Phone 8 applications. It explains how to modify the open source SQLite source code and compile it for use in your Windows Phone 8 applications.

This wiki takes you through the following steps:

1. Compiling the SQLite source code for ARM and x86 targets
2. Creating an Extension SDK targeted to Windows Phone 8
3. Using SQLite with your Windows Phone 8 applications

Pre-requisites

Before you begin, you will need:

- Visual Studio 2012 (RTM) or later.
- Visual Studio 2012 SDK
- Tcl 8.5 installed to "C:\Tcl".
- All these packages (Add these to your path).
- Fossil (source control for SQLite - like Git). Add fossil to your path.

Compiling SQLite

Following the instructions in the sections below compile SQLite for ARM and x86 architectures.

Building binaries for an ARM target

1. Open **Visual Studio 2012 ARM Phone Tools Command Prompt** in administrator mode.
2. Navigate to where you want to put the source or create a directory:

```
mkdir c:\sqlite-arm
cd c:\sqlite-arm
```

3. Get the latest SQLite source code:

```
fossil clone http://www.sqlite.org/cgi/src sqlite3.fossil
```

4. Open the repository you just created:

```
fossil open sqlite3.fossil
```

5. Type in the commands below in order:

```
SET NCRTLIBPATH=%ProgramFiles(x86)%\Microsoft Visual Studio
11.0\VC\WPSDK\WP80\lib

SET NSDKLIBPATH=%ProgramFiles(x86)%\Windows Phone Kits\8.0\lib\x86

SET USE_RC=0

SET OPTS=-DWINAPI_FAMILY=WINAPI_FAMILY_PHONE_APP -DSQLITE_OMIT_LOAD_EXTENSION=1
-DSQLITE_WIN32_FILEMAPPING_API=1 -DSQLITE_HAS_CODEC=1

nmake /F Makefile.msc sqlite3.c XCOMPILE=1 USE_NATIVE_LIBPATHS=1 FOR_WINRT=1
NO_TCL=1

nmake -f Makefile.msc sqlite3.dll /I XCOMPILE=1 USE_NATIVE_LIBPATHS=1
FOR_WINRT=1 NO_TCL=1
```


(C) Copyright Nokia 2014. All rights reserved.

opcodes.h	12/13/2013 3:44 PM	H File	11 KB
parse.c	12/13/2013 3:44 PM	C File	152 KB
parse.h	12/13/2013 3:44 PM	H File	8 KB
parse.h.temp	12/13/2013 3:44 PM	TEMP File	7 KB
parse.out	12/13/2013 3:44 PM	OUT File	381 KB
parse.y	12/12/2013 11:17 ...	Y File	50 KB
README	12/12/2013 11:17 ...	File	2 KB
see.c	10/17/2013 6:54 PM	C File	52 KB
shell.c	12/12/2013 11:17 ...	C File	109 KB
spec.template	12/12/2013 11:17 ...	TEMPLATE File	2 KB
sqlite.pc.in	12/12/2013 11:17 ...	IN File	1 KB
sqlite3.1	12/12/2013 11:17 ...	1 File	7 KB
sqlite3.c	12/13/2013 3:45 PM	C File	5,076 KB
sqlite3.def	12/18/2013 8:56 PM	DEF File	5 KB
sqlite3.dll	12/13/2013 2:27 PM	Application extens...	843 KB
sqlite3.exp	12/13/2013 3:44 PM	Exports Library File	30 KB
sqlite3.fossil	12/12/2013 11:18 ...	FOSSIL File	43,433 KB
sqlite3.h	12/13/2013 3:36 PM	H File	346 KB
sqlite3.lib	12/13/2013 3:44 PM	Object File Library	50 KB
sqlite3.lo	12/13/2013 3:44 PM	LO File	2,914 KB
sqlite3.pc.in	12/12/2013 11:17 ...	IN File	1 KB
sqlite3.pdb	12/13/2013 3:44 PM	Program Debug D...	19 KB
sqlite3ext.h	12/12/2013 11:17 ...	H File	26 KB
sqlite3rc.h	12/18/2013 8:56 PM	H File	1 KB
vc110.pdb	12/18/2013 8:56 PM	Program Debug D...	356 KB
VERSION	12/12/2013 11:17 ...	File	1 KB

Final binaries

When you're done, the `c:\sqlite-arm` source directory should contain `sqlite3.dll`, `sqlite3.h`, `sqlite3.lib` and `sqlite3.pdb` along with a lot of other generated files.

Building binaries for a x86 target

1. Open the **Visual Studio 2012 x86 Phone Tools Command Prompt** in administrator mode.
2. Create a source directory and navigate to it.

```
mkdir c:\sqlite-x86
cd c:\sqlite-x86
```

3. Follow steps 3, 4 and 5 building binaries for ARM target.

opcodes.h	12/13/2013 3:44 PM	H File	11 KB
parse.c	12/13/2013 3:44 PM	C File	152 KB
parse.h	12/13/2013 3:44 PM	H File	8 KB
parse.h.temp	12/13/2013 3:44 PM	TEMP File	7 KB
parse.out	12/13/2013 3:44 PM	OUT File	381 KB
parse.y	12/12/2013 11:17 ...	Y File	50 KB
README	12/12/2013 11:17 ...	File	2 KB
see.c	10/17/2013 6:54 PM	C File	52 KB
shell.c	12/12/2013 11:17 ...	C File	109 KB
spec.template	12/12/2013 11:17 ...	TEMPLATE File	2 KB
sqlite.pc.in	12/12/2013 11:17 ...	IN File	1 KB
sqlite3.1	12/12/2013 11:17 ...	1 File	7 KB
sqlite3.c	12/13/2013 3:45 PM	C File	5,076 KB
sqlite3.def	12/18/2013 8:56 PM	DEF File	5 KB
sqlite3.dll	12/13/2013 2:27 PM	Application extens...	843 KB
sqlite3.exp	12/13/2013 3:44 PM	Exports Library File	30 KB
sqlite3.fossil	12/12/2013 11:18 ...	FOSSIL File	43,433 KB
sqlite3.h	12/13/2013 3:36 PM	H File	346 KB
sqlite3.lib	12/13/2013 3:44 PM	Object File Library	50 KB
sqlite3.lo	12/13/2013 3:44 PM	LO File	2,914 KB
sqlite3.pc.in	12/12/2013 11:17 ...	IN File	1 KB
sqlite3.pdb	12/13/2013 3:44 PM	Program Debug D...	19 KB
sqlite3ext.h	12/12/2013 11:17 ...	H File	26 KB
sqlite3rc.h	12/18/2013 8:56 PM	H File	1 KB
vc110.pdb	12/18/2013 8:56 PM	Program Debug D...	356 KB
VERSION	12/12/2013 11:17 ...	File	1 KB

When you're done, the `c:\sqlite-x86` source directory should contain `sqlite3.dll`, `sqlite3.h`, `sqlite3.lib` and `sqlite3.pdb` along with a lot of other generated files.

Building the Extension SDK

Now that we have all the required binaries, we will create an Extension SDK targeted to Windows Phone 8.

- Create a new project by selecting **File -> New -> Project -> Visual C# -> Extensibility -> VSIX** project
- Create a new file `your_proj_name.props` and copy the source code below into it. (Make sure to replace *your_proj_name* and *version* accordingly.)

```
<?xml version="1.0" encoding="utf-8"?>
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <PropertyGroup>
    <PackageConfiguration Condition="'$(Configuration)' ==
''">Debug</PackageConfiguration>
    <PackageConfiguration Condition="'$(Configuration)' ==
'Debug'">Debug</PackageConfiguration>
    <PackageConfiguration Condition="'$(Configuration)' ==
'Release'">Retail</PackageConfiguration>
  </PropertyGroup>
```


(C) Copyright Nokia 2014. All rights reserved.

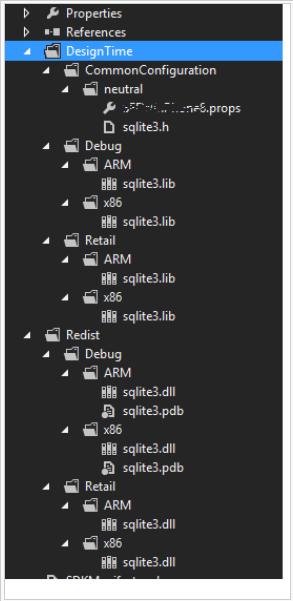
```
  <PropertyGroup>
    <IncludePath>$(FrameworkSDKRoot)\..\..\Windows
Phone\v8.0\ExtensionSDKs\your_proj_name\version\DesignTime\CommonConfiguration\Neutral;$(IncludePath)</IncludePath>
    <LibraryPath>$(FrameworkSDKRoot)\..\..\Windows
Phone\v8.0\ExtensionSDKs\your_proj_name\version\DesignTime\$(PackageConfiguration)\$(PlatformTarget);$(LibraryPath)</LibraryPath>
    <PropertySheetDisplayName> your_proj_name, version</PropertySheetDisplayName>
  </PropertyGroup>
  <ItemDefinitionGroup>
    <Link>

<AdditionalDependencies>sqlite3.lib;%(AdditionalDependencies)</AdditionalDependencies>
    </Link>
  </ItemDefinitionGroup>
</Project>
```

- Create a new file called `SDKManifest.xml` in the project root, and copy the XML below into it. (Make sure to replace *your_proj_name* accordingly.)

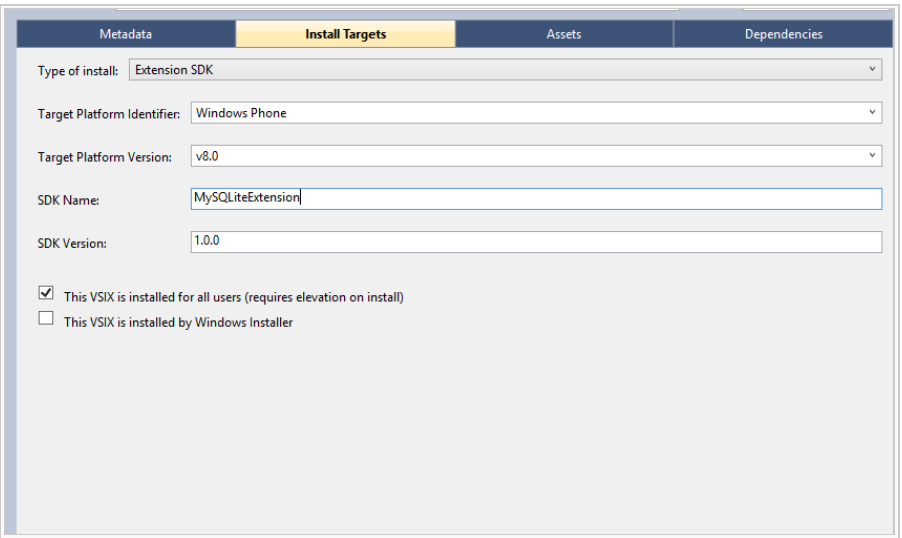
```
<?xml version="1.0" encoding="utf-8" ?>
<FileList
    DisplayName="MySQLite for Windows Phone"
    ProductFamilyName="your_proj_name"
    MoreInfo="http://www.sqlite.org/"
    MinVSVersion="11.0"
    SupportsMultipleVersions="Error"
    SupportedArchitectures="x86;ARM">
</FileList>
```

- Create the hierarchy below in the application source, and add the files that were generated during build process to the project. Files from `sqlite-x86` go into `x86` folders. Files from `sqlite-arm` go into `ARM` folders.



Extension SDK hierarchy

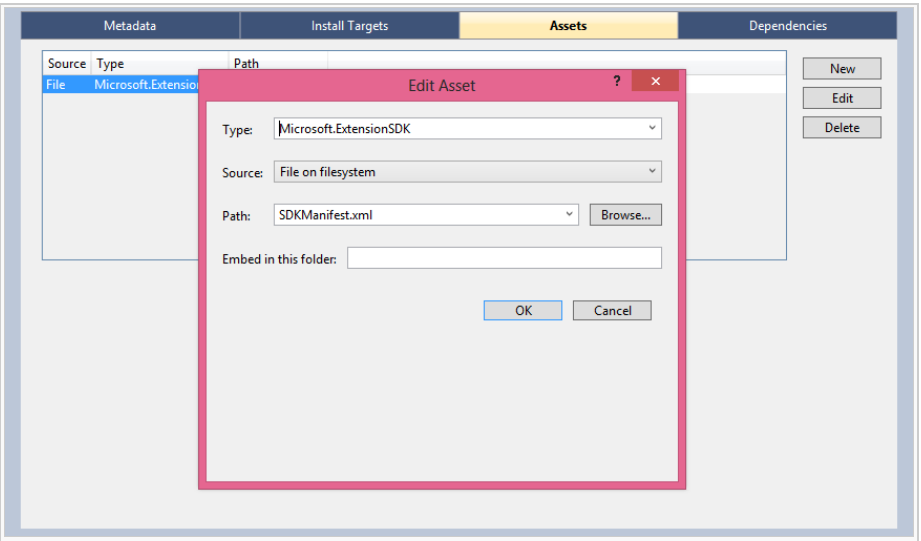
- Right click on file, click on **Properties** and change *Build Action* to *Content*, *Copy to output directory* to *Copy always* and *Include in VSIX* to *True*. Repeat this for all the added files.
- Double click on `source.extension.vsixmanifest`.
- Input the options below on the **install targets** tab.



install targets tab options

(C) Copyright Nokia 2014. All rights reserved.

- Input the options below in the **asset** tab.



assets tab options

- Now build the project. A `.VSIX` file should be generated in the `bin/{config}` folder.

SQLite Usage

- Install the MySQLite Extension SDK you just created on your computer and restart Visual Studio.
- Create a new Windows phone application.
- Download sqlite-net-wp8 and add the `Sqlite.vcxproj` project to your solution by selecting **File -> Add -> Existing Project**.
- Add the required compilation symbols as follows:

  Right click on **Windows Phone Project -> Properties**
  Click on the **Build** tab.
  In the **Configuration** dropdown, select **All configurations**.
  In the **Platform** dropdown, select **All platforms**.
  Add `USE_WP8_NATIVE_SQLITE;WINDOWS_PHONE;SILVERLIGHT` to the **Conditional compilation symbols** textbox.

- Add a reference to the SQLite extension to the sqlite project as follows:

  Right click on **SQLite Project -> Properties**.
  Click on the **Common properties** tab.
  Remove **SQLite.WP80** from the references.
  Click on **Add new reference**
  Select **Windows Phone -> Extensions -> SQLite extension**

- Add the **sqlite-net** nuget package to the Windows Phone project.
- Once this is done, you can use the database as you wish.

(C) Copyright Nokia 2014. All rights reserved.