QGis + SpatiaLite data provider

build notes

a step-by-step guide

---

a very quick intro

In order to build an experimental version of QGis 1.1 supporting the SpatiaLite data provider you can freely choose one of the following ways:

- ➔ alternative 1:
  - get the latest QGis [unstable] development sources from the SVN
  - download the `qgis_spatialite_patches.tar.gz` tarball
  - then manually apply to the QGis sources any patch required in order to implement the SpatiaLite data provider
- ➔ *alternative* 2:
  - download the `qgis_spatialite.tar.gz` tarball [this representing QGis rev.10264 and already including any SpatiaLite related patch]

Before starting the QGis build you need to build SpatiaLite. see:
http://www.gaia-gis.it/spatialite-2.3/

Once you've got the patched QGis sources (and built SpatiaLite), you can build QGis as usual:

```
cmake ..
make
sudo make install
```

A1) patching the sources by yourself

step A1-1: getting the GGis sources from the SVN

```
svn co https://svn.osgeo.org/qgis/trunk/qgis qgis_spatialite
```

step A1-2: downloading the `qgis_spatialite_patches.tar.gz` tarball and extracting the files

```
gunzip qgis_spatialite_patches.tar.gz
tar xvf qgis_spatialite_patches.tar
```

step A1-3: applying the SpatiaLite related patches to QGis source [using a shell script; it assumes that both `qgis_spatialite` and `qgis_spatialite_patches` are within the same mother directory]

```
cd qgis_spatialite_patches
./apply_patches
```

rationale: full patches list

file	notes
`~/CMakeLists.txt`	mod: added a search directive for spatialite
`~/cmake/FindSPATIALITE.cmake`	new: searching rules for spatialite's headers and libs
`~/images/themes/default/mActionAddSpatiaLiteLayer.png`	new: a menu/toolbar icon representing a spatialite data source
`~/src/app/CMakeLists.txt`	mod: included the spatialite source selection added stuff
`~/src/app/qgisapp.h`	mod: added menu and toolbar support for spatialite source selection
`~/src/app/qgisapp.cpp`	
`~/src/app/qgsspatialitesourceselect.h`	new: implementing the spatialite source selection dialog and closely related stuff
`~/src/app/qgsspatialitesourceselect.cpp`	
`~/src/app/qgsspatialitetablemodel.h`	
`~/src/app/qgsspatialitetablemodel.cpp`	
`~/src/app/qgsspatialitefilterproxymodel.h`	
`~/src/app/qgsspatialitefilterproxymodel.cpp`	
`~/src/ui/qgsspatialitesourceselectbase.ui`	new: spatialite source selection dialog template
`~/src/providers/CMakeLists.txt`	mod: included the spatialite subdir
`~/src/providers/spatialite/CMakeLists.txt`	new: data provider implementation
`~/src/providers/spatialite/qgsspatialiteprovider.h`	
`~/src/providers/spatialite/qgsspatialiteprovider.cpp`	

A2) using the pre-patched sources

step A2-1: downloading the `qgis_spatialite.tar.gz` tarball and extracting the files it contains

```
gunzip qgis_spatialite.tar.gz
tar xvf qgis_spatialite.tar
```

B) building

I tested all this on Ubuntu 8.04.1
I'm not sure if it will work on some different Linux …

Important notice: any required dependency has to be resolved **before** starting the actual build [GEOS, PROJ, GDAL …] and don't forget**, SpatiaLite itself !!!**

step B-1: creating a build dir

```
cd qgis_spatialite
mkdir build_test
cd build_test
```

step B-2: running CMake

```
cmake ..
…
-- Generating done
-- Build files have been written to: blah blah
```

Critical: there is some major conflict still to be solved:

1. **sqlite3**: the standard headers and libraries on Ubuntu (the ones you found on `/usr/include` and `/usr/lib`) are incredibly obsolete. SpatiaLite absolutely requires the latest v.3.6.10 (or v.3.6.11): this is shipped within SpatiaLite, but (at least on my Ubuntu box) they are located on `/usr/local/include` and `/usr/local/lib`
2. **proj.4**: identical problem. Ubuntu ships v.4.6.0 (on `/usr/include` and `/usr/lib`), SpatiaLite requires the latest v.4.6.1 (on `/usr/local/include` and `/usr/local/lib`)
3. **geos**: some as above. Ubuntu ships an obsolete v.2.2.3 (on `/usr/include` and `/usr/lib`), SpatiaLite requires the latest v.3.0.3 (on `/usr/local/include` and `/usr/local/lib`)

So, when running CMake I got lots and lots of messages like this one:

```
CMake Warning at src/providers/grass/CMakeLists.txt:52 (ADD_LIBRARY):
  Cannot generate a safe runtime search path for target grassprovider because
  files in some directories may conflict with libraries in implicit
  directories:
    runtime library [libproj.so.0] in /usr/lib may be hidden by files in:
      /usr/local/lib
    runtime library [libsqlite3.so.0] in /usr/lib may be hidden by files in:
      /usr/local/lib
  Some of these libraries may not be found correctly.
```

I fixed this issue (yes I know by myself: this is an horrible and tricky workaround, not really a solution …) simply hand-editing the `CMakeCache.txt` file, replacing:

1. any occurrence of: `/usr/lib/libsqlite3.so` with: `/usr/local/lib/libsqlite3.so`
2. any occurrence of: `/usr/lib/libproj.so` with: `/usr/local/lib/libproj.so`

After applying this, I then relaunched CMake again another time …

```
cmake ..
-- Found Proj: /usr/local/lib/libproj.so
-- Found Sqlite3: /usr/local/lib/libsqlite3.so
-- Found GEOS: /usr/local/lib/libgeos_c.so
-- Found GDAL: /usr/lib/libgdal1.4.0.so
-- Found SpatiaLite: /usr/local/lib/libspatialite.so
-- Found PostgreSQL: /usr/lib/libpq.so
-- Found Expat: /usr/lib/libexpat.so
-- Using GSL from /usr
-- Found GRASS: /usr/lib/grass (6.2.2)
-- Python libraries found
-- Python bindings enabled
-- Configuring done
-- Generating done
-- Build files have been written to: blah blah
```

All right, we have no more warning messages …

This specific point surely needs to be carefully handled; a more general and elegant solution is absolutely due.
Any useful suggestion will be really appreciated.

<u>step B-3:</u> building and installing

```
make
sudo make install
```