Subject: How to share a schema .sch with multiple header and source files
Posted by MonkeyH on Wed, 07 Aug 2019 12:54:39 GMT
View Forum Message <> Reply to Message

Hello,
I'm new to U++ and I'm making a generic desktop application wich read and write data to an sqlite database.

Planning for multiple functionalities, I guess the structure of this application could be like this:

Main Application (main window definition and methods)
  |
  | Db schema definition
  |
  +------> functionality class 1 (some data manipulation and user interface)
  |
 (...)
  |
  +------> functionality class N (some data manipulation and user interface)

How I can share the database schema with all the header and source files, avoiding multiple definitions and link errors?

Thanks.

Subject: Re: How to share a schema .sch with multiple header and source files
Posted by mirek on Fri, 09 Aug 2019 08:56:47 GMT
View Forum Message <> Reply to Message

Actually, SQL_Sqlite3 example has it already right:

This goes to header:

```
#define SCHEMADIALECT <plugin/sqlite3/Sqlite3Schema.h>
#define MODEL <SQL_Sqlite3/simple.sch>
#include "Sql/sch_header.h"
```

(hint, hint, see "_header" :)

This goes to some C++ file:

```
#ifdef _DEBUG
#include <Sql/sch_schema.h>
#endif
```

```
#include <Sql/sch_source.h>
```

Now above provides two things. "_schema" part creates script generation routines that create sql scripts to actually create the schema in the database. Usually we are using script with "alter table add column" commands so that incremental changes can be added to database (we just ignore "thing already exists" errors when performing that script). In the example, this coresponds to

```
#ifdef _DEBUG
 SqlSchema sch(SQLITE3);
 All_Tables(sch);
 if(sch.ScriptChanged(SqlSchema::UPGRADE))
  SqlPerformScript(sch.Upgrade());
 if(sch.ScriptChanged(SqlSchema::ATTRIBUTES)) {
  SqlPerformScript(sch.Attributes());
 }
 if(sch.ScriptChanged(SqlSchema::CONFIG)) {
  SqlPerformScript(sch.ConfigDrop());
  SqlPerformScript(sch.Config());
 }
 sch.SaveNormal();
#endif
```

(SaveNormal saves scripts files to the disk, so that they can be used e.g. for the production machine).

Not that this is usually excluded in release (#ifdef _DEBUG) version of application... (but not necessary).

"_source" part creates code to support S_TABLE structures (e.g. mapping between column names and C++ variables) and is always needed.

Mirek
