Subject: Re: How to use multiple schemas and databases?
Posted by jarchalex on Mon, 07 Nov 2011 16:39:22 GMT
View Forum Message <> Reply to Message

Now this code is already compilable)) But causes SQL error "no such table: FLOWCONVERSIONUNITS"

```
#include <CtrlLib/CtrlLib.h>
#include <GridCtrl/GridCtrl.h>
#include <Painter/Painter.h>
#include <SqlCtrl/SqlCtrl.h>
#include <plugin/sqlite3/Sqlite3.h>
using namespace Upp;

#define MODEL <studyDraw_01/db.sch>
#define SCHEMADIALECT <plugin/sqlite3/Sqlite3Schema.h>
#include "Sql/sch_header.h"
#include "Sql/sch_schema.h"
#include "Sql/sch_source.h"

#define LAYOUTFILE <studyDraw_01/gui.lay>
#include <CtrlCore/lay.h>

struct Tab_Settings {
 Sqlite3Session sqlite3;

 Tab_Settings() { Init();}
 void Init() {
  if(!sqlite3.Open(ConfigFile("settings.conf"))) {
   Exclamation("Can't create or open database file\n");
   return;
  }
  SQL = sqlite3;
  SqlSchema sch(SQLITE3);
  sqlite3.SetTrace();
  All_Tables(sch);
  if(sch.ScriptChanged(SqlSchema::UPGRADE))
   Sqlite3PerformScript(sch.Upgrade());
  if(sch.ScriptChanged(SqlSchema::ATTRIBUTES))
   Sqlite3PerformScript(sch.Attributes());
  if(sch.ScriptChanged(SqlSchema::CONFIG)) {
   Sqlite3PerformScript(sch.ConfigDrop());
   Sqlite3PerformScript(sch.Config());
  }
  sqlite3.SetTrace();
 }
};
```

```
#define MODEL <studyDraw_01/db_info.sch>
#define SCHEMADIALECT <plugin/sqlite3/Sqlite3Schema.h>
#include "Sql/sch_header.h"
#include "Sql/sch_source.h"

struct Tab_FlowConversionUnits {
 Sqlite3Session sqlite3;

 Tab_FlowConversionUnits() { Init(); }
 void Init() {
  if(!sqlite3.Open(ConfigFile("settings_2.conf"))) {
   Exclamation("Can't create or open database file\n");
   return;
  }
  SqlSchema sch(SQLITE3);
  sqlite3.SetTrace();
  All_Tables(sch);
  if(sch.ScriptChanged(SqlSchema::UPGRADE))
   Sqlite3PerformScript(sch.Upgrade());
  if(sch.ScriptChanged(SqlSchema::ATTRIBUTES))
   Sqlite3PerformScript(sch.Attributes());
  if(sch.ScriptChanged(SqlSchema::CONFIG)) {
   Sqlite3PerformScript(sch.ConfigDrop());
   Sqlite3PerformScript(sch.Config());
  }
  sqlite3.SetTrace();
 }
};

struct MyDraw: Ctrl {
 virtual void Paint(Draw& w) {

  Size sz = GetSize();
  ImageBuffer ib(sz);
  BufferPainter sw(ib);
  DoPaint(sw);
  w.DrawImage(0, 0, ib);

 }
 void DoPaint(Painter& sw) {
  Size sz = GetSize();
  sw.DrawRect(0, 0, sz.cx, sz.cy, White());
  sw.Rectangle(sz.cx*0.05,sz.cy*0.05,sz.cx*0.9,sz.cy*0.9);
  sw.Fill(LtBlue());

  sw.Scale(0.5);
  sw.Translate(0, 50);
```

```
  const char *txt = "GRADIENT TEXT";
  Font fnt = Arial(100).Bold();
  Size tsz = GetTextSize(txt, fnt);
  sw.Text(100, 100, txt, fnt)
    .Stroke(4, 100, 100, Blue(), 100 + tsz.cx, 100, LtRed());
 }

 void RightDown(Point, dword) {
  CallbackArgTarget<int> result;
  MenuBar rMenu;
  for(int i = 0; i < 10; i++)
   rMenu.Add(AsString(i), result[i]);
  rMenu.Execute();
  if(!IsNull(result))
   PromptOK("You have selected " + AsString((int)result));
 }
};
struct MyWindow: TopWindow {
 MenuBar menu;
 StatusBar status;
 SqlArray table1;
 SqlArray tab_settings;
 GridCtrl table2;
 GridCtrl table3;
 TabCtrl tabs;
 Splitter spl, set;
 EditString name;
 EditInt val;
 EditDouble valD;
 EditString parameter;
 EditString value;
 MyDraw draw;
 Tab_Settings tab_set;
 Tab_FlowConversionUnits tab_flow;

 void Exit() {
  if(PromptOKCancel("Exit?"))
   Break();
 }

 void SubMenu(Bar& bar) {
  bar.Add("Exit", THISBACK(Exit))
   .Help("Exit application");
 }

 void MainMenu(Bar& bar) {
  bar.Add("Menu", THISBACK(SubMenu));
 }
```

```
typedef MyWindow CLASSNAME;

MyWindow() {
 Title("DB test");
 AddFrame(menu);
 AddFrame(status);


 table1.SetSession(tab_flow.sqlite3);
 table1.SetTable(FLOWCONVERSIONUNITS);
 table1.AddKey(ID);
 table1.AddColumn(UNIT, t_("Units")).Edit(parameter);
 table1.AddColumn(FACTOR, t_("Conversion factor")).Edit(value);
 table1.Appending().Removing();
 table1.SetOrderBy(ID, UNIT);


 table2.AddIndex();
 table2.AddColumn(0, t_("One"));
 table2.AddColumn(t_("Two"));
 table2.AddColumn(t_("Three"));
 table2.Appending().Removing().Editing().Accepting().Canceling();
 table2.RejectNullRow();
 table2.SetToolBar();
 table3.AddColumn(0, t_("One"));
 table3.SetToolBar();


 tabs.Add(table1.SizePos(), "table1");

 tabs.Add(table2.SizePos(), "table2");
 tabs.Add(table3.SizePos(), "table3");
 tabs.Set(0);


 tab_settings.SetSession(tab_set.sqlite3);
 tab_settings.SetTable(SETTINGS);
 tab_settings.AddKey(ID);
 tab_settings.AddColumn(PARAMETER, t_("Parameter")).Edit(parameter);
 tab_settings.AddColumn(VALUE, t_("Value")).Edit(value);
 tab_settings.Appending().Removing();
 tab_settings.SetOrderBy(ID, PARAMETER);

 spl.Vert();
 spl.Add(tab_settings);
 spl.Add(draw);
 spl.Add(tabs);
```

```
  spl.SetPos(0,0);
  Add(spl);
  menu.Set(THISBACK(MainMenu));
  menu.WhenHelp = status;
  tab_settings.Query();
 }
};

GUI_APP_MAIN
{
 MyWindow w;
 w.Sizeable().MinimizeBox().MaximizeBox();
 w.SetRect(0, 0, 600, 500);
 w.Run();
}
```
