Subject: Sql object
Posted by mubeta on Sun, 07 Feb 2021 17:07:39 GMT
View Forum Message <> Reply to Message

Hi everyone,
before before writing this post I read several posts on this topic, but I came to the conclusion that I had to change the code of the U ++ library.

The Sql object seems for now unable to be hooked at the free choice of the programmer to different DB connections. So that if the choice of a SQLite or MySQL type DB for example, depended on configuration choices to be left to the end user of the resulting app, it would be impossible to use the same U++ code to write an application, in fact for now, the SQL object must be built clearly indicating a connection since its compilation.
Sql sql1(myconnection);

After hours and hours of testing, research, etc., I came to the unwanted decision to make the umpteenth change to the U++ library, moving the Sql.SetSession() method from private to public, and now I have finally managed to write a code that works on several DB chosen by the end user.

I do not understand why the presence of such a method has been left as private. If there are concerns about possible change errors at runtime, different mechanisms could be identified, but with a much more flexible object.

Subject: Re: Sql object
Posted by dolik.rce on Mon, 08 Feb 2021 05:42:20 GMT
View Forum Message <> Reply to Message

Hi mubeta,

Years ago, I've been trying something similar. I've come with a solution, where the sql code was compiled as dynamic library, so the final program didn't even require mysql libraries to run, unless it was configured to use mysql. You can find the code on my github, if you're intrested:
https://github.com/dolik-rce/thewatchdog/tree/master/src/Dyn amicSql

Note however few important things:
The code is pretty old, ~7 years, it might not work with current U++. I do not use the code anymore, and don't maintain it, so I cannot guarantee or support it in anyway.It also needed little patch of U++ sources: https://github.com/dolik-rce/thewatchdog/blob/master/patch/M ySql.patchOnly MySql and SQLite are implemented, but adding more options should be easyActually the worst part was not to allow dynamic selection of engine, but to write queries in a way that is compatible with both sqlite and mysql. In theory there is ANSI SQL, which should be a common subset of all SQL languages, but reality is much different :)

Anyway, I hope this helps, or at least inspires you (or someone else in future) :)

Best regards,
Honza