Subject: L
Posted by mirek on Sun, 08 Feb 2009 07:16:03 GMT
View Forum Message <> Reply to Message

White_Owl wrote on Thu, 05 February 2009 14:261) I am more used to a "driver-behind-connection" way of database interfaces. For example, such interfaces like ODBC, ADO, or even QT, do not work with actual connections, but delegate all work to some driver. Application programmer will use some kind of universal connection object. The actual driver is selected in run-time by supplying this universal object with driver's name right before the establishing the actual connection.
What I see in U++ so far is that relationship between the universal connection object and the driver is reversed. To create a connection to database XYZ I have to make an offspring of SqlSession class and use it for actual connection object. That means I have to know in the design-time what kind of database I will use...
Am I correct so far?

Yep. Note however that any real application will need to know that too, as SQL dialects differ significantly.

Moreover, you are actually not using SqlSession in most applications after the connection.

At least, SqlExp is able to iron out most differences in SQL... So in the end, U++ code is far more "cross-DB" than anything else.

Quote:
2) Why PostgreSQL, Oracle and other packages are located in the root of uppsrc, but sqlite3 package is in the plugin subfolder? Should not they be in some common uppsrc/plugin/db/ subfolder?

Maybe Well, the (poor) explanation is that "plugin" is intended for source code "imported" from other projects. Sqlite3 package contains the whole Sqlite3 C sources, therefore it is in "plugin". Oracle etc... are just interfaces using external libs.

Mirek