Windows embedded

The embedded server is a fully functional server linked as a dynamic library that is distributed with the name fbembed.dll. It has exactly the same features as the usual Superserver and its client part exports the standard Firebird API entrypoints.

The embedded server acts as a true local server for a single client accessing databases on a local machine. It can also act as a remote gateway that redirects all network calls to other hosts, just as the regular client library does.

Firebird Embedded for Windows comes only as a zip kit, since it is only a component of the embedded system that you will build around it. However, you should take care to unpack the kit in the structure under which it was packed, since many parts of an embedded setup rely on finding one another within that directory tree.

Registry

Any Firebird Registry entries are ignored. The root directory of the embedded server is the one where the embedded library binary (fbembed.dll, usually renamed to fbclient.dll) is located.

Database access

Client access can be only via the local (XNET) protocol, i.e. NOT a TCP/IP local loopback connection string that includes the server name localhost or the IP address 127.0.0.1. The embedded server supports only the local connect to an absolute database file path without a server name.

The client program gets exclusive access to the database file after a successful connect. If another Firebird server already has a client attached to the database, the client program will be denied access. This is intentional.

Do not try to connect to a database on any mapped location! The database MUST be on a local partition that is controlled by the machine that is hosting your embedded server and its surrounding application.

back to top of page

Authentication and security

The security database (security2.fdb) is not used in connecting to the embedded server. Hence it is not required. Any user is able to attach to any database. Since both the server and the client run in the same address space, security becomes just an agreement between the accessor and the accessed, which can be easily compromised.

Note: SQL privileges are still checked and enforced. Users that are assigned privileges in a Firebird database are not dependent on the existence of the user in the security database. Applications may still validly pass a user name in the database connection attributes and should do so, to make their

user known to the database's access control list.

Compatibility

You may run any number of applications with the embedded server without any conflicts. Having a full Firebird or InterBase® server running on the same machine is not a problem, either.

However, be aware that you cannot access a single database from a number of servers simultaneously, regardless of whether they be embedded or full servers. An embedded server has the Superserver architecture and hence exclusively locks any database it attaches to. This is intentional.

Installing an Embedded Server application

MS Visual C/C++ runtimes

For v.2.1.x the MS runtime libraries msvcp80.dll and msvcr80.dll must be available in the embedded library's path. You can extract copies of these libraries from the zip kit version of the full Firebird build if they are not already present on your system.

If you have skipped over the earlier notes concerning the MCVC8 runtime libraries, it is recommended that you review them now.

Application root

Just copy fbembed.dll, icudt30.dll, icuin30.dll and icuuc30.dll into the directory with your application exectable.

You should also copy firebird.msg and firebird.conf (if necessary) to the same directory.

Note: You will need firebird.conf only if it is necessary to set some non-default configuration parameters for the embedded server.

If external libraries are required for your application, such as INTL support (fbintl.dll and fbintl.conf) or UDF libraries, create subdirectories beneath the application root for them, emulating the Firebird server ones, e.g. /intl or /udf, respectively.

Rename fbembed.dll

Rename fbembed.dll to either fbclient.dll or gds32.dll, according to which is required by your database connectivity software.

Start your application

Now start your application and it will use the embedded server as a both a client library and a server and will be able to access local datasases via the XNET network emulation protocol.

back to top of page

Installation structure examples

```
c:\my_app\app.exe
c:\my_app\gds32.dll
c:\my_app\ib_util.dll
c:\my_app\icudt30.dll
c:\my_app\icuin30.dll
c:\my_app\icuuc30.dll
c:\my_app\firebird.conf
c:\my_app\firebird.msg
c:\my_app\intl\fbintl.dll
c:\my_app\intl\fbintl.conf
c:\my_app\udf\fbudf.dll
```

Suppose you want to place the Firebird files (excluding the renamed fbembed.dll) in another directory. In that case, you need to modify your firebird.conf and set RootDirectory to the Firebird directory tree that is parent to the Firebird files.

Example

```
c:\my_app\app.exe
c:\my_app\gds32.dll
c:\my_app\ib_util.dll
c:\my_app\icudt30.dll
c:\my_app\icuin30.dll
c:\my_app\icuuc30.dll
c:\my_app\firebird.conf
d:\fb\firebird.msg
d:\fb\intl\fbintl.dll
d:\fb\intl\fbintl.conf
d:\fb\udf\fbudf.dll
```

In firebird.conf:

```
RootDirectory = d:\fb
```