Firebird 2 Migration & Installation

Helen Borrie (Collator/Editor)

11 January 2008 - Document v. mi210_03 - for Firebird 2.1 RC 1

Firebird 2 Migration & Installation

11 January 2008 - Document v. mi210_03 - for Firebird 2.1 RC 1

Helen Borrie (Collator/Editor)

Table of Contents

Chapter 1

Known Compatibility Issues

D. Yemanov

This chapter is intended as a set of alerts to those who are migrating Firebird 1.0 or 1.5 databases to Firebird 2.0. It should be studied before attempting to install any servers.

The FIREBIRD Variable

FIREBIRD is an optional environment variable that provides a system-level pointer to the root directory of the Firebird installation. If it exists, it is available everywhere in the scope for which the variable was defined.

The FIREBIRD variable is NOT removed by scripted uninstalls and it is not updated by the installer scripts. If you leave it defined to point to the root directory of a v.1.5.x installation, there will be situations where the Firebird engine, command-line tools, cron scripts, batch files, installers, etc., will not work as expected.

If the Windows installer program finds a value for %FIREBIRD% it will make that path the default location that it offers, instead of `c:\Program Files\Firebird\Firebird_2_0`.

Unless you are very clear about the effects of having a wrong value in this variable, you should remove or update it before you begin installing Firebird 2.0. After doing so, you should also check that the old value is no longer visible in the workspace where you are installing Firebird--use the `SET FIREBIRD` command in a Windows shell or `printenv FIREBIRD` in a POSIX shell.

Security in Firebird 2 (All Platforms)

Be aware of the following changes that introduce incompatibilities with how your existing applications interface with Firebird's security:

Direct connections to the security database are no longer allowed
Apart from the enhancement this offers to server security, it also isolates the mechanisms of authentication from the implementation.

- User accounts can now be configured only by using the Services API or the *gsec* utility.
- For backing up the security database, the Services API is now the only route. You can employ the `-se[rvice] hostname:service_mgr` switch when invoking the *gbak* utility for this purpose.

Non-SYSDBA users no longer can see other users' accounts in the security database
A non-privileged user can retrieve or modify only its own account and it can change its own password.

Remote attachments to the server without a login and password are now prohibited

- For attachments to Superserver, even root trying to connect locally without “localhost:” in the database file string, will be rejected by the remote interface if a correct login is not supplied.
-

Embedded access without login/password works fine. On Windows, authentication is bypassed. On POSIX, the Unix user name is used to validate access to database files.

The security database is renamed to `security2.fdb`

If you upgrade an existing installation, be sure to upgrade the security database using the provided script in order to keep your existing user logins.

Before you begin the necessary alterations to commission an existing security database on the Firebird 2.0 server, you should create a *gbak* backup of your old security.fdb (from v.1.5) or isc4.gdb (from v.1.0) using the old server's version of *gbak* and then restore it using the Firebird 2.0 *gbak*.

> **Important**
>
> You must make sure that you restore the security database to have a page size of at least 4 Kb. The new security2.fdb will not work with a smaller page size.

> **Warning**
>
> A simple `'cp security.fdb security2.fdb'` will make it impossible to attach to the firebird server !

For more details see the notes in the chapter on security in the accompanying Release Notes. Also read the file `security_database.txt` in the *upgrade* directory beneath the root directory of your installation.

SQL Migration Issues

DDL

Views made updatable via triggers no longer perform direct table operations

In former versions, a naturally updatable view with triggers passed the DML operation to the underlying table and executed the triggers as well. The result was that, if you followed the official documentation and used triggers to perform a table update (inserted to, updated or deleted from the underlying table), the operation was done twice: once executing the view's trigger code and again executing the table's trigger code. This situation caused performance problems or exceptions, particularly if blobs were involved.

Now, if you define triggers for a naturally updatable view, it becomes effectively like a non-updatable view that has triggers to make it updatable, in that a DML request has to be defined on the view to make the operation on the underlying table happen, viz.

1. if the view's triggers define a DML operation on the underlying table, the operation in question is executed once and the table triggers will operate on the outcome of the view's triggers

2. if the view's triggers do not define any DML request on the underlying table then no DML operation will take place in that table

> **Important**
>
> Some existing code may depend on the assumption that requesting a DML operation on an updatable view with triggers defined would cause the said operation to occur automatically, as it does for an updatable view with no triggers. For example, this “feature” might have been used as a quick way to write records to a log table en route to the “real” update. Now, it will be necessary to adjust your view trigger code in order to make the update happen at all.

New Reserved Words (Keywords)

A number of new reserved keywords are introduced. The full list is available in a chapter of its own in the accompanying Release Notes and also in Firebird's CVS tree in /doc/sql.extentions/README.keywords. You must ensure that your DSQL statements and procedure/trigger sources do not contain those keywords as identifiers.

> **Note**
>
> In a Dialect 3 database, such identifiers can be redefined using the same words, as long as the identifiers are enclosed in double-quotes. In a Dialect 1 database there is no way to retain them: they must be redefined with new, legal words.

CHECK Constraint Change

Formerly, CHECK constraints were not SQL standard-compliant in regard to the handling of NULL. For example, `CHECK (DEPTNO IN (10, 20, 30))` should allow NULL in the DEPTNO column but it did not.

In Firebird 2.0, if you need to make NULL invalid in a CHECK constraint, you must do so explicitly by extending the constraint. Using the example above:

```
CHECK (DEPTNO IN (10, 20, 30) AND DEPTNO IS NOT NULL)
```

DML

Changed Ambiguity Rules in SQL

A. Brinkman

In summary, the changes are:

1. When an alias is present for a table, that alias, and not the table identifier, must be used to qualify columns; or no alias is used. Use of an alias makes it invalid to use the table identifier to qualify a column.

2. Columns can now be used without qualifiers in a higher scope level. The current scope level is checked first and ambiguous field checking is done at scope level.

Examples

a) 1. When an alias is present it must be used or no alias at all must be used.

This query was allowed in FB1.5 and earlier versions:

```
SELECT
```

```
  RDB$RELATIONS.RDB$RELATION_NAME
FROM RDB$RELATIONS R
```

Now, the engine will correctly report an error that the field “RDB$RELATIONS.RDB $RELATION_NAME” could not be found.

Use this (preferred):

```
SELECT
  R.RDB$RELATION_NAME
FROM RDB$RELATIONS R
```

or this statement:

```
SELECT
  RDB$RELATION_NAME
FROM
  RDB$RELATIONS R
```

a) 2. The next statement will now use the appropriate FieldID correctly from the subquery and from the updating table:

```
UPDATE TableA
  SET
    FieldA = (SELECT SUM(A.FieldB) FROM TableA A
    WHERE A.FieldID = TableA.FieldID)
```

> **Note**
>
> Although it is possible in Firebird to provide an alias in an update statement, many other database vendors do not support it. These SQL statement syntaxes provide better interchangeability with other SQL database products.

a) 3. This example ran incorrectly in Firebird 1.5 and earlier:

```
SELECT
  RDB$RELATIONS.RDB$RELATION_NAME,
  R2.RDB$RELATION_NAME
FROM RDB$RELATIONS
JOIN RDB$RELATIONS R2 ON
 (R2.RDB$RELATION_NAME = RDB$RELATIONS.RDB$RELATION_NAME)
```

If RDB$RELATIONS contained 90 rows, it would return 90 * 90 = 8100 rows, but in Firebird 2.0 it will correctly return 90 rows.

b) 1. This would fail in Firebird 1.5, but is possible in Firebird 2.0:

```
SELECT
  (SELECT RDB$RELATION_NAME FROM RDB$DATABASE)
FROM RDB$RELATIONS
```

b) 2. Ambiguity checking in subqueries

This would run on Firebird 1.5 without reporting an ambiguity, but will report it in Firebird 2.0:

```
SELECT
  (SELECT FIRST 1 RDB$RELATION_NAME
   FROM RDB$RELATIONS R1
   JOIN RDB$RELATIONS R2 ON
     (R2.RDB$RELATION_NAME = R1.RDB$RELATION_NAME))
FROM RDB$DATABASE
```

Multiple Hits to Same Column Now Illegal

It is no longer allowed to make multiple "hits" on the same column in an INSERT or UPDATE statement. Thus, a statement like

```
INSERT INTO T(A, B, A) ...
```

or

```
UPDATE T SET A = x, B = y, A = z
```

will be rejected in Firebird 2.n, even though it was tolerated in InterBase and previous Firebird versions.

Query Plans

Stricter validation of user-specified plans

User-specified plans are validated more strictly than they were formerly. If you encounter an exception related to plans, e.g. `Table T is not referenced in plan`, it will be necessary to inspect your procedure and trigger sources and adjust the plans to make them semantically correct.

> **Important**
>
> Such errors could also show up during the restore process when you are migrating databases to the new version. It will be necessary to correct these conditions in original database before you attempt to perform a backup/restore cycle.

Plan must refer to all tables in query

Using a plan without a reference to all tables in query is now illegal and will cause an exception. Some previous versions would accept plans with missing references, but it was a bug.

PSQL

Restrictions on assignment to context variables in triggers

- Assignments to the OLD context variables are now prohibited for every kind of trigger.

- Assignments to NEW context variables in AFTER-triggers are also prohibited.

> **Tip**
>
> If you get an unexpected error `Cannot update a read-only column` then violation of one of these restrictions will be the source of the exception.

Reference to "current of <cursor>" outside scope of loop

In Firebird 1.5 and earlier, referring to "current of <cursor>" outside the scope of the cursor loop was accepted by the PSQL parser, allowing the likelihood of run-time occurring as a result. Now, it will be rejected in the procedure or trigger definition.

NULLS are now “lowest” for sorts

NULL is now treated as the lowest possible value for ordering purposes and sets ordered on nullable criteria are sorted accordingly. Thus:

- for ascending sorts NULLs are placed at the beginning of the result set
- for descending sorts NULLs are placed at the end of the result set

> **Important**
>
> In former versions, NULLs were always at the end. If you have client code or PSQL definitions that rely on the legacy NULLs placement, it will be necessary to use the NULLS LAST option in your ORDER BY clauses for ascending sorts.

CURRENT_TIMESTAMP now returns milliseconds by default

The context variable CURRENT_TIMESTAMP now returns milliseconds by default, while it truncated sub-seconds back to seconds in former versions. If you need to continue receiving the truncated value, you will now need to specify the required accuracy explicitly, i.e. specify `CURRENT_TIMESTAMP(0)`.

*ORDER BY <ordinal-number> now causes SELECT * expansion*

When columns are referred to by the “ordinal number” (degree) in an ORDER BY clause, when the output list uses `SELECT * FROM ...` syntax, the column list will be expanded and taken into account when determining which column the number refers to.

This means that, now, `SELECT T1.*, T2.COL FROM T1, T2 ORDER BY 2` sorts on the second column of table T1, while the previous versions sorted on T2.COL.

> **Tip**
>
> This change makes it possible to specify queries like `SELECT * FROM TAB ORDER BY 5`.

Configuration Parameters

Configuration parameter DeadThreadsCollection is deprecated

The parameter DeadThreadsCollection for Superserver in firebird.conf is deprecated and will be ignored if set. Firebird version 2 efficiently cleans up dead threads straight away.

Command-line Tools

Change to gbak -R Semantics

An important change has been done to prevent accidental database overwrites as the result of users mistakenly treating "-R" as an abbreviation for "restore". *gbak -R* was formerly a shortcut for "-REPLACE_DATABASE". Now the -R switch no longer restores a database by overwriting an existing one, but instead reports an error.

If you actually *want* the former behaviour, you have two alternatives:

- Specify the full syntax `gbak -REPLACE_DATABASE`. There is a new shortcut for the -REPLACE_DATABASE switch: `gbak -REP`

  OR

- Use the new command `-R[ECREATE_DATABASE] OVERWRITE`. The `-R` shortcut now represents the -R[ECREATE_DATABASE] switch and the OVERWRITE keyword must be present in either the full or the abbreviated form.

> **Warning**
>
> If you use the full syntax, you are expected to know what this restore mode actually means and have some recovery strategy available if the backup subsequently turns out to be unrestorable.

Performance

The following changes should be noted as possible sources of performance loss:

Existence Predicates NOT IN and ALL May Be Slow
: Firebird and, before that, InterBase, have produced incorrect results for the logical existence predicates ALL and NOT IN for many years. That problem has bee corrected in Firebird 2.0, but the change means that indexes on the inner tables cannot be used and performance may be slow compared to the same query's performance in V.1.5. "Inner tables" are the tables used in the subquery argument inside an ALL or NOT IN expression.

> **Note**
>
> NOT EXISTS is approximately equivalent to NOT IN and will allow Firebird to use indexes.

Superserver garbage collection changes
: Formerly, Superserver performed only background garbage collection. By contrast, Classic performs "co-operative" GC, where multiple connections share the performance hit of GC.

  Superserver's default behaviour for GC is now to combine cooperative and background modes. The new default behaviour generally guarantees better overall performance as the garbage collection is performed online, curtailing the growth of version chains under high load.

It means that some queries may be slower to start to return data if the volume of old record versions in the affected tables is especially high. ODS10 and lower databases, having ineffective garbage collection on indices, will be particularly prone to this problem.

The GCPolicy parameter in firebird.conf allows the former behaviour to be reinstated if you have databases exhibiting this problem.

Firebird API

Note the following changes affecting the API

isc_interprete is deprecated
isc_interprete() is deprecated as dangerous. Use fb_interpret() instead.

Events callback routine declaration corrected
The new prototype for `isc_callback` reflects the actual callback signature. Formerly, it was:

```
typedef void (* isc_callback) ();
ISC_STATUS isc_que_events(
  ISC_STATUS *, isc_db_handle *, ISC_LONG *, short,
  char *, isc_callback, void *);
```

In the Firebird 2.0 API it is:

```
typedef void (*ISC_EVENT_CALLBACK)
  (void*, ISC_USHORT, const ISC_UCHAR*);
ISC_STATUS isc_que_events(
  ISC_STATUS*, isc_db_handle*, ISC_LONG*, short,
  const ISC_SCHAR*, ISC_EVENT_CALLBACK, void*);
```

It may cause a compile-time incompatibility, as older event handling programs cannot be compiled if they use a bit different signature for a callback routine (e.g., `void*` instead of `const char*` as the last parameter).

Windows-Specific Issues

For installing, configuring and connecting to Windows servers, be aware of the following issues:

Windows Local Connection Protocol with XNet

The transport internals for the local protocol have been reimplemented (XNET instead of IPServer). With regard to the local protocol, the new client library is therefore incompatible with older servers and older client libraries are incompatible with the Firebird 2 servers.

If you need to use the local protocol, please ensure your server and client binaries have exactly the same version numbers.

Client Impersonation No Longer Works

WNET (a.k.a. NetBEUI, Named Pipes) protocol no longer performs client impersonation. For more information, refer to the chapter about new features in the accompanying Release Notes.

Interactive Option Added to instsvc.exe

D. Yemanov

The optional switch -i[nteractive] has been implemented in `instsvc.exe` to enable an interactive mode for LocalSystem services.

For v.1.5, it was required (as *Allow service to interact with desktop*) to run the local IPC protocol, as it used a windows message to connect the server. In v.2.0, it is no longer necessary and the server itself does not need this option.

However, some custom UDFs may use the Win32 messaging facilities and this option allows them to work as expected.

> **Note**
>
> `instsvc.exe` is a command-line utility for installing and uninstalling the Firebird service. It does not apply to Windows systems that do not have the ability to run services (Win9x, WinME).
>
> For detailed usage instructions, refer to the document `README.instsvc` in the `doc` directory of your Firebird installation.

Chapter 2

INSTALLATION NOTES

Please read the previous chapter, Known Compatibility Issues before you set out to install Firebird 2.0.

Choosing a Server Model

Classic, Superserver and Embedded are all the same Firebird engine. The differences are in the ways the server module uses machine and network resources. Briefly:

- Classic runs a stub process named **fb_inet_server.exe** on Windows that listens for connection requests from network clients. On POSIX systems, the *[x]inetd daemon* performs this listening role. The stub process or daemon creates a separate **fb_inet_server** process for each successful connection. Each process has its own private page cache and, on 32-bit systems, can use up to 2 GB of RAM.
- Superserver is a process named **fbserver.exe** on Windows, **fbserver** on POSIX. Like the Classic server stub it also listens for connection requests. Unlike Classic, Superserver is a *threaded application*. It starts (or assigns) one or more threads for each successful connection request. All of these connection threads share a common page cache. On 32-bit systems, Superserver is limited to using a maximum of 2 GB of RAM.
- In discussions, Classic and Superserver are often referred to as “full servers” because Firebird also supports an *embedded* server on both Windows and POSIX. In this model, intended for “stand-alone” deployments, a local client and a server are rolled together into one shared object library.
  - On Windows, the embedded server library is named **fbembed.dll**. It is a temporary file name that must be changed in order to be used. The Windows embedded server is an instance of Superserver.
  - On POSIX, the embedded library is distributed with the Classic kits. Its name is **libfbembed.so** and its manner of resource usage is like Classic.

Database Compatibility Among Models

There are no issues that make databases created by one server model incompatible with another server model. Your ultimate choice of which server model to deploy to user sites will be determined by comparing the performance of one with another in your test lab. You don't have to do anything to a database in order to make it work under a different server model.

Full Servers

At install time, your choice is most likely to be whether to use Classic or Superserver. For development, there's nothing in it. For testing and deployment, the choice may be more important. Superserver's shared page cache, properly configured, can be beneficial for performance where many users are working concurrently. On the other hand, Superserver for Windows does not “play nice” with multiple CPUs on most rigs and has to be set for affinity with just one CPU.

Classic can be a good choice if the host server has multiple CPUs and plenty of RAM.

> **Note**
>
> There is more discussion about the Classic/Superserver decision in the Quick Start Guide. A more detailed paper on the subject can be found in the IBPhoenix documentation archives.

Embedded

Treat the embedded server as a deployment option. It is intended to be used by one and only one system user exclusively, which makes it impossible to use in many integrated application development environments.

Windows Installs

On Windows, you have three server models to choose from: Superserver, Classic and Embedded Server. This means you have some decisions to make before installing Firebird 2.1.

Choosing an Installation Method

Under almost all circumstances you should use the binary installer to install Firebird on Windows. If you are new to Firebird you should certainly use the binary installer until you are familiar with a standard Firebird installation. It covers all common usage cases and is typically 'click-through' by accepting the default values.

Cases when you might wish to consider installing Firebird manually from a zip file include:

- You need to run multiple versions of Firebird concurrently
- You wish to run the Firebird service as a special user
- You need to deploy a specific configuration for your application

> **Note**
>
> The Embedded Server can only be installed from a zip-file. It is intended to be integrated with the installer of your own application.

READ THIS FIRST!

General notes applicable to all Windows installs

- Make sure you are logged in as Administrator (doesn't apply on Win9x or ME)
- Check to make sure that there is no FIREBIRD environment variable defined that is visible to Administrator-level users or to the LocalSystem user—see the section called “The FIREBIRD Variable” at the start of the previous chapter.
- If you already have an earlier version of Firebird or InterBase® on your server and you think you might want to go back to it, set up your fall-back position before you begin:
  - Use the existing version of **gbak** to back up your database files in transportable format. Do this before you uninstall the old version.

- If migrating from a Firebird version earlier than version 2.0 you should use **gbak** to back up your old security database. It is named **security.fdb** for Firebird 1.5 or **isc4.gdb** for a Firebird 1.0 installation. You can restore it later as **security2.fdb**, using the directions in the chapter entitled “Security” in the accompanying Release Notes.

- Go to your System directory and make backup copies of **fbclient.dll** and/or **gds32.dll** if you have applications that rely on finding those libraries there. You might want to name the backup “gds32.dll.fb15” or “gds32.dll.fb103” or something similarly informative; or hide it in another directory.

- STOP ANY FIREBIRD OR INTERBASE SERVER THAT IS RUNNING.

  At install time, the installer will try to detect if an existing version of Firebird or InterBase is installed and/or running. However this detection is not foolproof. For a non-installer install, you are on your own!

  > **Important**
  >
  > A new installation of Firebird will not work correctly if an old version is still running. The uninstaller will usually stop an existing server as part of the uninstallation process, so you probably don't need worry about this. However, if you do have problems later this is something to go back and check.

- Before installing Firebird 2.1 it is recommended that you uninstall a previous version of Firebird (if it exists). If you have special settings in your existing **firebird.conf** (**ibconfig**, for Firebird 1.0) there may be some values that you want to transfer to equivalent parameters in the new **firebird.conf**. The uninstaller for all versions of Firebird will preserve certain configuration files. See below for more details.

  > **Note**
  >
  > If you are upgrading from Firebird 1.0.x, you should review the release notes for Firebird 1.5.5. You will find details of the correlation between settings in **ibconfig** and **firebird.conf**. Study the notes about **firebird.conf** to work out what can be copied directly and what parameters require new syntax.
  >
  > If this document is not in the documentation directory after installation, you can read or download it from the Release Notes section of the Firebird Documentation Index.

- When reinstalling Firebird 2.1, certain configuration files in the installation directory will be preserved if you run the installer. These files are:

  ```
  aliases.conf
  firebird.conf
  firebird.log
  security2.fdb
  ```

  - The default Aliases.conf is just a place holder file, so if it already exists it will be left as it is. If it doesn't exist it will be generated.

  - If firebird.conf exists the installer will generate a firebird.conf.default file with the default values and leave the existing file untouched.

  - the firebird.log file is generated automatically by the server when required. An empty log file is not created at installation time.

  -

If the security2.fdb database exists it will be used. If it doesn't exist an empty, default database will be installed.

- It is assumed that:

  1. you understand how your network works
  2. you understand why a client/server system needs both a server and clients
  3. you have read the accompanying Release Notes—or at least realise that you need to read them if something seems to have gone wrong
  4. you know to go to the Firebird lists index to find a suitable support list if you encounter a problem.

- Provided you do not have a FIREBIRD environment variable defined, the default root location of Firebird 2.1 will be `C:\Program Files\Firebird\Firebird_2_1`. For Firebird 2.0 it will be `C:\Program Files\Firebird\Firebird_2_0`.

Naming databases on Windows

Note that the recommended extension for database files on Windows ME and XP is ".fdb" to avoid possible conflicts with "System Restore" feature of these Windows versions. Failure to address this issue on these platforms will give rise to the known problem of delay on first connection to a database whose primary file and/or secondary files are named using the ".gdb" extension that used to be the Borland convention for suffixing InterBase database file names.

The issue is described in more detail in Other Win32 Issues at the end of the Windows installation notes.

Other Pre-installation Issues

Installation of Microsoft system libraries

The problems associated with installing different versions of Microsoft system libraries are so notorious that it has acquired the name 'DLL Hell'. And as each new generation of Microsoft operating systems are released the policy for dealing with this issue changes. Sometime this can lead to even more hell.

The binary installer will determine the host operating system and try to install system libraries appropriately for that O/S. In most cases there will be no problems. However, early versions of WinXP and Windows 2003 that have not used Windows Update will not have the correct version of the Windows Installer required to install the side-by-side assemblies of the run-time libraries. If this occurs there are two possible solutions:

- Run Windows Update and then re-install Firebird; OR

- Install the `vcredist.exe` package available from Microsoft

Zip kit install

To achieve this when installing from the zip kit requires more work from the user:

- If the host O/S is pre-WinXP then the msvc 80 runtime libraries can be copied from the Firebird \bin\ directory into the Windows \system32\ directory.

- If the host O/S is WinXP or later then you will need to check the version of the Windows installer. Run `msiexec.exe` from a console prompt and a help screen will be displayed. If it is earlier than v.3.0 you must upgrade.

  Once you have `msiexec.exe` v3.0 or later you can then install the vccrt MSI file located in the \system32\ directory of the zip kits.

Running Firebird as a service with a special user name

Firebird can be made more secure on a Windows system if it is run as a service with its own user name. If you want to make use of the secure login feature, create a “firebird service user” on the system with any name and password you like. It must have the privileges for a normal user.

More information is in the document named `README.instsvc.txt`. If you have a zip kit, you will find it in the /doc/ directory of the zipfile's root. If you don't have a zip kit available, the file won't be available until after the installation. You can read the same document at this URL.

Installing Multiple Servers

Firebird 2.1 makes it easier than ever to run multiple servers simultaneously. However the second and subsequent servers must be installed manually. You can read more about this in the file `install_windows_manually.txt` available in the doc directory after installation, or it can be found at this URL.

Installing under 64-bit versions of Windows

The 64-bit binary installer includes a 32-bit client kit so that everything will work 'out of the box'. On the other hand, the zip kits are platform specific, so don't forget to install the 32-bit MS C runtime msi, along with the 32-bit client library if you need to use 32-bit applications on the server.

Simultaneous installation of 32-bit and 64-bit versions of Firebird is possible, but at least one must be installed and configured manually. Note that under these circumstances the FIREBIRD environment variable must NOT be defined at the system level.

Installation of fbclient.dll

Since Firebird 1.5, gds32.dll is not the “native” name of the client library. It is now called fbclient.dll. Given the problems that Microsoft have had with DLL hell it wouldn't make much sense if we continued to store the Firebird client library in the system directory. Furthermore, as we want to allow multiple engines to run simultaneously we would be creating our own DLL hell if we continued the practice of using the system directory for the client library.

So, from Firebird 1.5 on, the client library resides in the \bin directory along with all the other binaries. The installer provides the option (unchecked) to copy the client to the system directory for those who have applications that require to load them from there.

Registry Key

A Registry key has been added and all Firebird 2.0 compliant applications should use this key if they need to read a Registry key to locate the correct version of Firebird that they wish to use. The new key is:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Firebird Project\Firebird Server\Instances
```

Firebird will guarantee that one entry under this key always exists. It will be known as

```
"DefaultInstance"
```

and will store the path to the root directory of (yes, you've guessed it) the default installation. Those that don't care about particular installations can always use the default instance to locate the fbclient.dll.

Future versions of Firebird will see other entries under Instances. Applications will be able to enumerate the Registry entries to determine which Server instance they wish to load.

Supporting legacy applications and drivers

Traditionally, applications that use InterBase or Firebird have expected to load the gds32.dll client library from the system directory. Firebird 2.0 ships with a tool named 'instclient.exe' that can install a clone of fbclient.dll to the Windows System directory. This clone gets patched on the fly so that its file version information begins with "6.3", to provide compatibility for old applications that check the GDS32.DLL file version and can not make sense of a number string such as "2.0".

InstClient.exe Tool

This 'instclient.exe' tool can also install the FBCLIENT.DLL itself in the Windows system directory, if required. This will take care of tools or applications that need to load it from there.

The instclient.exe utility should be located in the 'bin' directory of your Firebird installation and must be run from there in a command shell.

Usage of instclient.exe:

```
instclient i[nstall] [ -f[orce] ] library
           q[uery] library
           r[emove] library
```

where library is: `fbclient | gds32`

`'-z'` can be used with any other option, prints version.

Version information and shared library counts are handled automatically. You may provide the `-f[orce]` option to override version checks.

> **Caution**
>
> If you `-f[orce]` the installation, it could break another Firebird or InterBase® version already installed. You might have to reboot the machine in order to finalize the copy.

For more details, see the document README.Win32LibraryInstallation.txt which is located in ..\doc.

Cleaning up release candidate installs

It should be noted that the installer removes fbclient.dll from the <system> directory if the file is found there. The installer also removes any deprecated HKLM\Software\Firebird* Registry keys.

Using the Win32 Firebird Installer

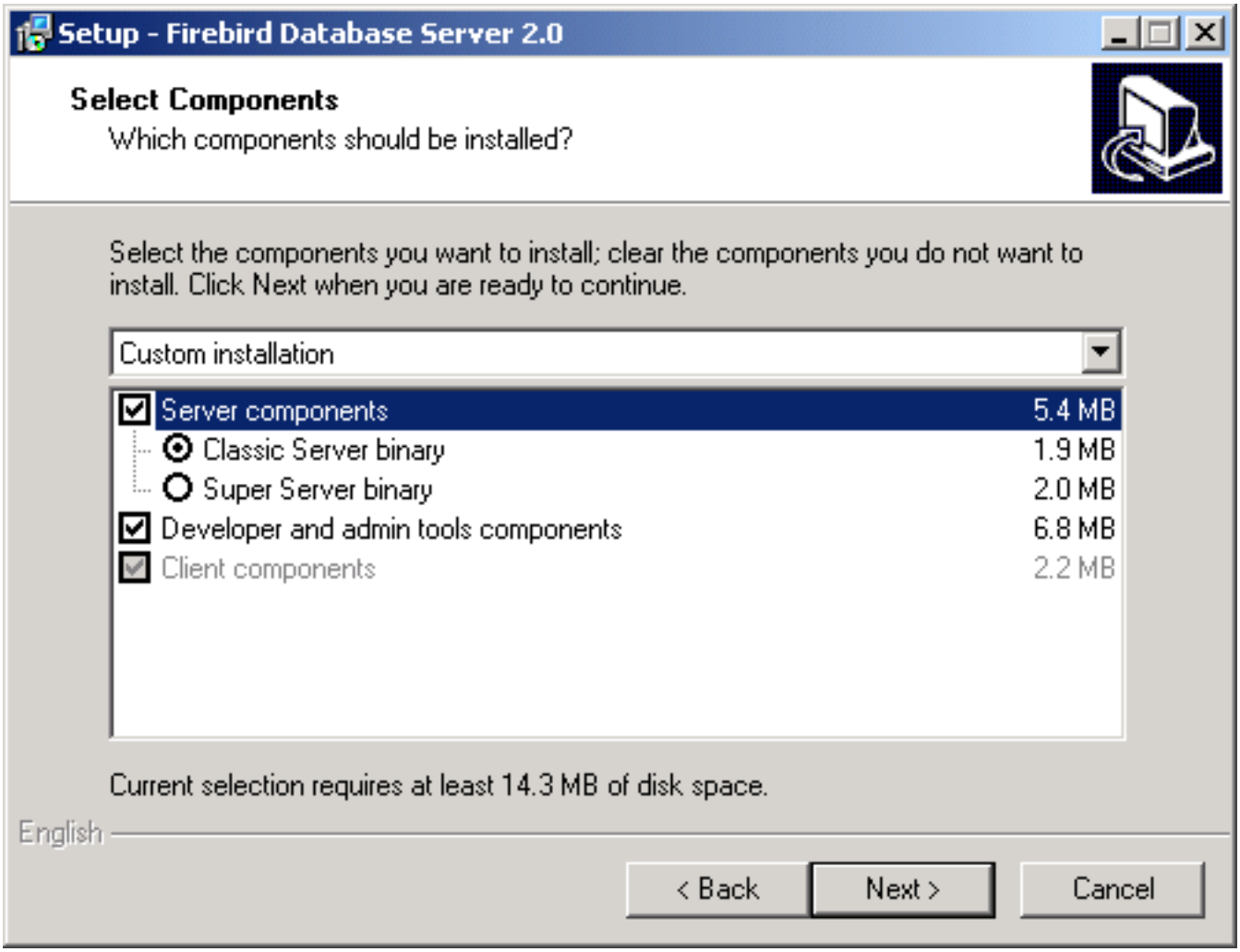
> **Important**
>
> Don't overlook the need to have the Microsoft® Visual C and Visual C++ runtimes (msvcrt.dll and msvcpp60.dll, respectively) present in the system directory of all Windows servers and clients, including Windows Embedded installations. For your convenience, copies of these libraries will be placed in the \bin directory of the Firebird install. However, you should check first whether later versions of these libraries are already present. Don't overwrite later versions.

This is really the easy part: the actual install. Just run the executable and respond to the dialogs. After you have answered a few dialogs about licensing and installation notes, you should see one where you decide on the location of the Firebird root directory.

Installation (Root) directory
: The installer should be showing “c:\Program Files\Firebird\Firebird_2_0” by default. If you decide not to use the default root location, browse to a location you have pre-created; or just type in the full path and let the installer find it. The path you type in doesn't have to exist: the installer will prompt you and create it if it doesn't exist.

  Here you can also opt not to have the installer create Startup Menu icons by checking off the option. If you are installing on Windows 9x or WinMe, or you plan to run the server as an application in another Win32 environment, keep the icons option checked on.

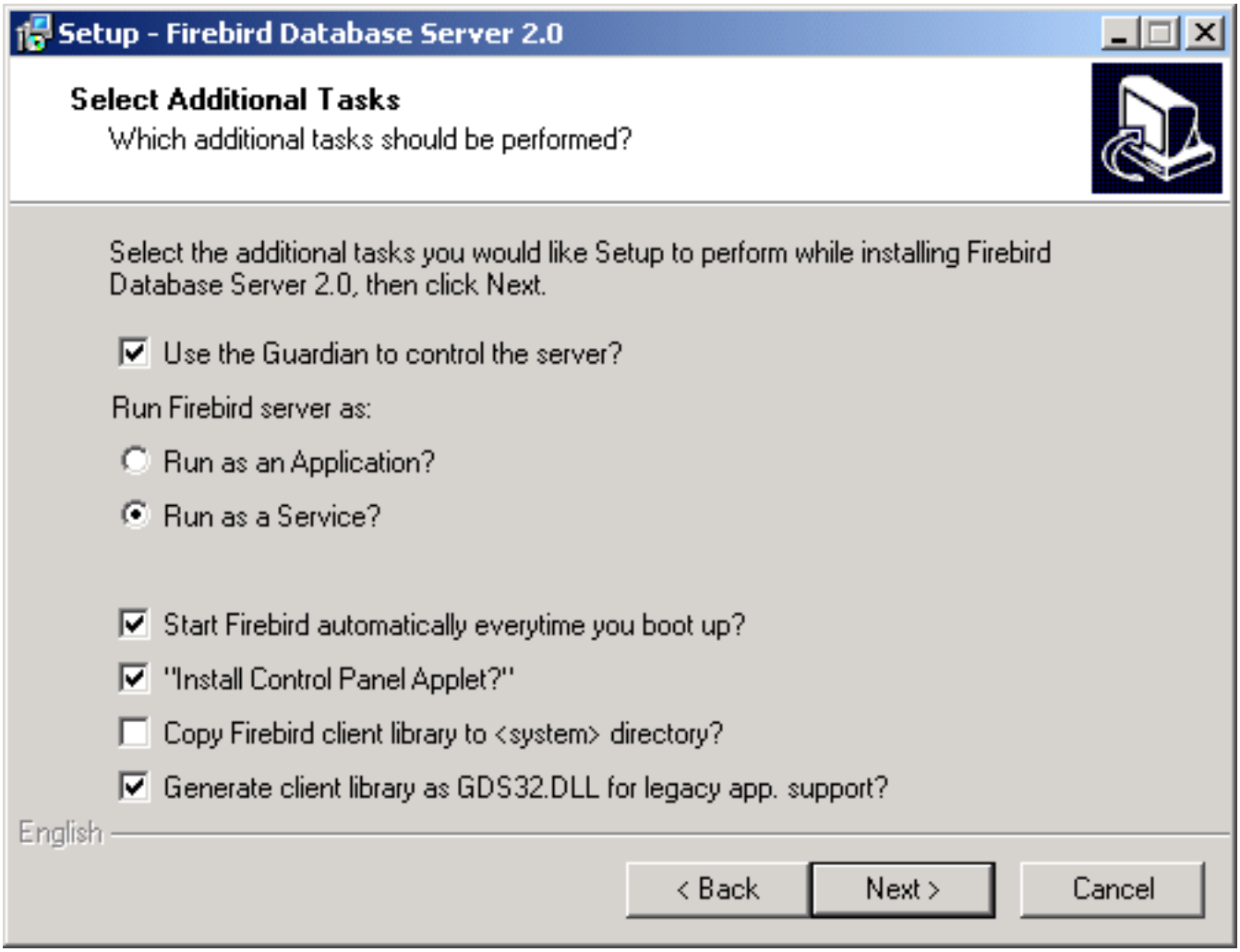
Next, you should see a screen where you choose the installation you want:



Choose the installation you want and hit the "Next" button to carry on responding to dialogs.

> **Note**
>
> If you're installing a server, you should choose Superserver (preselected by the installer) or Classic (as seen in the image above). Leave “Server components” and “Developer and admin tools components” checked on.
>
> For a client-only install, check off “Server components”, leaving “Client components” and, optionally, “Developer and admin tools components” checked on.
>
> There is also a drop-down for a custom installation which new users can safely ignore.

The next screen of interest enables you to set up how you want the server to run.



Choose the options you want, according to your choice of server model.

Use the Guardian...

Guardian is a utility than can run "over the top" of Superserver and restart it if it crashes for any reason. If you chose the Classic server, the Guardian option won't appear.

For deployment of Superserver on Win9x, WinME and WinNT 4.0, using Guardian can avoid the situation where the server stops serving and nobody can find the DBA to restart it. On other Win32 platforms, you can set the operating system to restart the service instead and not bother with the Guardian.

Service or application?

If you select to install Superserver or Classic, and your OS version supports services, you will be asked to choose whether to run Firebird as a service or as an application. Unless you have a compelling need to run the server as an application, choose service.

Manual or automatic?

With the automatic option, Firebird will start up whenever you boot the host machine. With the manual option you can start the server on demand from the Services applet in the Settings/Control Panel/ Administration Tools selection.

Use Control Panel Applet (Superserver only)

If Superserver is being installed, you will see an option to “Install Control Panel applet?”. It's a good idea to keep this as it places an applet in the Control Panel from which you can stop and [re]start the server.

Eventually, the dialogs will stop, you will press “Install” and the server will either silently start the server (if you requested it) or prompt you for permission to reboot. Reboot will be needed if the installer was unable to update a DLL due to its being already loaded when the installer started up.

Uninstallation

If you are going to uninstall Firebird, first shut down all connections to databases and then, if you are running Superserver, shut down the server. The Firebird uninstall routine (run from Add/Remove Programs in the Control Panel) preserves and renames the following key files:

preserves security2.fdb or renames it to security2.fbnnnn
preserves firebird.log
preserves firebird.conf or renames it to firebird.confnnnn
preserves aliases.conf or renames it to aliases.confnnnn

"nnnn" is the build number of the old installation.

No attempt is made to uninstall files that were not part of the original installation.

Shared files such as fbclient.dll and gds32.dll will be deleted if the share count indicates that no other application is using them.

The Registry keys that were created will be removed.

Installing Superserver from a zip kit

The installation of FB 2.0 is similiar in principle to previous versions. If you don't have a special setup program (it's distributed separately) the steps are the following:

- unzip the archive into a new directory
- change the current directory to $FIREBIRD\bin (here and below, $FIREBIRD refers to the directory where the v.2.0 files are located)
- run instreg.exe:

```
instreg.exe install
```

  It causes the installation path of the directory above to be written into the registry (HKLM\Software\Firebird Project\Firebird Server\Instances\DefaultInstance)

- if you want to register a service, also run instsvc.exe:

```
instsvc.exe install
```

- optionally, you may need to run instclient.exe to copy fbclient.dll or a specially-generated clone as gds32.dll to the OS system directory

Installing Classic Server from a zip kit

To install the CS engine, the only difference is the additional switch for instsvc.exe:

```
instsvc.exe install -classic
```

> **Important**
>
> Notice that this means that you may have only one architecture of the engine--either fbserver.exe (Superserver) or fb_inet_server.exe (the parent process for Classic)--installed as a service.
>
> The Control Panel applet is not installed with Classic--deliberately. Don't try to install and use it. The concept of terminating a service does not apply to the Classic model.

Simplified setup

If you don't need a registered service, then you may avoid running both instreg.exe and instsvc.exe. In this case you should just unzip the archive into a separate directory and run the server as an application:

```
fbserver.exe -a
```

It should treat its parent directory as the root directory in this case.

Uninstallation

To remove Firebird 2.0.1 without a Windows Uninstaller you should:

- stop the server
- run "instreg.exe remove"
- run "instsvc.exe remove"
- delete installation directory
- delete fbclient.dll and gds32.dll from the OS system directory

Other Win32 Issues

Winsock2
: Firebird requires WinSock2. All Win32 platforms should have this, except for Win95. A test for the Winsock2 library is made during install. If it is not found the install will fail. To find out how to go about upgrading, visit this link.

Windows ME and XP

Windows ME and XP (Home and Professional editions) there is a feature called *System Restore*, that causes auto-updating (backup caching?) of all files on the system having a ".gdb" suffix. The effect is to slow down InterBase/Firebird database access to a virtual standstill as the files are backed up every time an I/O operation occurs. (On XP there is no System Restore on the .NET Servers).

A file in the Windows directory of ME, c:\windows\system\filelist.xml, contains "protected file types". ".gdb" is named there. Charlie Caro originally recommended deleting the GDB extension from the "includes" section of this file. However, since then, it has been demonstrated that WinME might be rebuilding this list. In XP, it is not possible to edit filelist.xml at all.

On ME, the permanent workarounds suggested are one of:

- use FDB (Firebird DB) as the extension for your primary database files--RECOMMENDED
- move databases to C:\My Documents, which is ignored by System Restore
- switch off System Restore entirely (consult Windows doc for instructions).

On Windows XP Home and Professional editions you can move your databases to a separate partition and set System Restore to exclude that volume.

Windows XP uses smart copy, so the overhead seen in Windows ME may be less of an issue on XP, for smaller files at least. For larger files (e.g. Firebird database files, natch!) there doesn't seem to be a better answer as long as you have ".gdb" files located in the general filesystem.

Updated Notes for Windows Embedded

Some changes between Firebird 1.5 and Firebird 2.0 mean the existing docs are slightly out-of-date. For convenience, the following are the updated notes.

The embedded server is a fully functional server linked as a dynamic library (fbembed.dll). It has exactly the same features as the usual Superserver and 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.

Registry

The Firebird Registry entries are ignored. The root directory of the embedded server is the same directory as the one where the embedded library binary 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 successful connect.

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.

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 embedded servers simultaneously, regardless of whether they be embedded or full servers. An embedded server has the SuperServer architecture and hence exclusively locks attached databases.

Installing an Embedded Server Application

Application Root

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

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 parameter 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.

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
```

POSIX Platforms

(Originally by Mark O'Donohue, revised for 2.0)

The Firebird server comes in two forms, Classic, which runs as a service, and SuperServer, which runs as a background daemon. Classic is the more traditional UNIX service, while Superserver uses threads, rather than processes. For the user just starting out with Firebird, either will do, although the Classic server is likely to prove a better platform for initially experimenting with Firebird.

READ THIS FIRST

- You will need to be root user to install Firebird.
- Installation on Linuxen requires a glibc package installed that is equal to or greater than glibc-2.2.5 and a libstdc++.so equal to or greater than libstdc++-5.0.

> **Note**
>
> Some higher distros, e.g. Mandriva 10.2, might fail to complete the password-setting script at the end of a Classic installation because the local client, libfbembed.so needs libstdc++.so.5 and the installed version of this runtime is missing. An "impure" solution that should solve the immediate problem, at least, is to Google for “compat-libstdc++” and find one that was built for your kernel version.
>
> The *pure* solution of course is to compile Firebird on the same system that you are going to run it on! This might be necessary, anyway, if the compatibility runtimes cause problems with other applications.

- Do not try to use *rpm --update* to bring any existing Firebird package installation up to date. *The Firebird packages do not support it.*
- If you are installing Superserver on a Linux that supports the “new POSIX threading library ” (NPTL) then choose the NPTL build of Firebird. Most distros with the 2.6 kernel are built with NPTL enabled; some with later 2.4 kernels also enabled it, but it may be wise to prepare to revert to the regular build and set up to export the LD_ASSUME_KERNEL=2.2.5 variable if the 2.4 implementation of the NPTL causes problems. Details for doing this follow below.
- 64-bit builds are available for both Classic and Superserver. These should be installed only on a 64-bit Linux system. NPTL support is native on 64-bit Linux.

Setting Linux to Use the Old Threading Model

If the NPTL causes problems for SuperServer and locally compiled programs, including utilities such as gbak throwing a *Broken Pipe* error, you can try to solve the problem by forcing Linux to use the old threading model. To fix.-

1. In /etc/init.d/firebird

   ```
   LD_ASSUME_KERNEL=2.2.5
   export LD_ASSUME_KERNEL
   ```

   That takes care of the server instance.

2. You need to have the `LD_ASSUME_KERNEL` environment variable set up within the local environment as well, so add the following to /etc/profile, to ensure every user picks it up for the command line utilities.

   after

   ```
   HISTSIZE=1000
   ```

   add

   ```
   LD_ASSUME_KERNEL=2.2.5
   ```

   On the following line, export it (this is all in one line):

```
export PATH USER LOGNAME MAIL HOSTNAME
    HISTSIZE INPUT_RC LD_ASSUME_KERNEL
```

Installing on Linux

The following instructions describe the Classic installation. For installation of Superserver the "CS" in the package name is replaced by "SS". For example, the package `FirebirdCS-2.0.0-nnnnn.i686.rpm` is replaced by `FirebirdSS-2.0.0-nnnnn.i686.rpm`.

> **Note**
>
> For those who, in the past, have had trouble installing Firebird on Slackware, the good news is that the installers in this version *do* include Slackware support.

Log in as root, or open a root shell. In the example filenames, replace *nnnnn* with the build number of the kit you actually have.

RPM Installer

For the RPM installer, type:

```
$rpm -ivh FirebirdCS-2.0.0-nnnnn.i686.rpm
```

Installing the Tarball

To install the tarball, place the ".tar.gz" file and type:

```
$tar -xzf FirebirdCS-2.0.0-nnnnn.tar.gz
$cd FirebirdCS-2.0.0-nnnnn.i686
$./install.sh
```

What the Linux install scripts will do

The Linux install scripts will

1. Attempt to stop any currently running server
2. Add the user 'firebird' and the group 'firebird' if they do not already exist.
3. Install the software into the directory /opt/firebird and create links for libraries in /usr/lib and header files in /usr/include
4. Automatically add gds_db for port 3050 to /etc/services if the entry does not already exist
5. Automatically add localhost.localdomain and HOSTNAME to /etc/gds_hosts.equiv

6. a. SuperServer only installs a /etc/rc.d/init.d/firebird server start script.

   b. Classic server installs a /etc/xinetd.d/firebird start script or, for older inetd systems, adds an entry to the /etc/inetd file

7. Specific to SuSE, a new rcfirebird link is created in /usr/bin for the init.d script and an /etc/rc.config Firebird entry is created.

8. Starts the server/service. Firebird should start automatically in runlevel 2, 3 or 5

9. Generates and sets a new random SYSDBA password and stores it in the file /opt/firebird/SYSDBA.password.

10. Adds an entry to aliases.conf for the sample database, employee.fdb.

Testing your Linux installation

Step 1 - Accessing a database

In a shell:

```
$cd /opt/firebird/bin
$./isql -user sysdba -password <password>[1]

SQL>connect localhost:employee.fdb /* this is an aliased path */

SQL>select * from sales;
SQL>select rdb$relation_name from rdb$relations;
SQL>help;

SQL>quit;
```

> **Note**
>
> [1]A password has been generated for you on installation. It can be obtained from the /opt/firebird/SYSDBA.password file, located in the Firebird root directory.

Step 2 - Creating a database

The Firebird server runs by default as the user 'firebird'. While this has always been the recommended configuration, the previous default was for the server to run as 'root' user. When running as root user, the server had quite wide-ranging ability to read, create and delete database files anywhere on the POSIX filesystem.

For security reasons, the service should have a more limited ability to read/delete and create files.

While the new configuration is better from a security perspective, it requires some special considerations to be taken into account for creating new databases:

1. the user 'firebird' has to have write permission to the directory in which you want to create the database.

2. the recommended value of the DatabaseAccess attribute in the /opt/firebird/firebird.conf file should be set to None, to permit access only through entries in the aliases.conf file.

3. use entries in aliases.conf to abstract users from the physical locations of databases.

Procedures for creating a new database can vary with different configurations but the following configuration and steps are recommended:

1. If a directory that is owned by the user 'firebird' does not exist, then change to root user and create the directory:

```
$su - root
$mkdir -p /var/firebird
$chown firebird:firebird /var/firebird
```

2. Create a new physical database and set up an alias entry to point to it. As root or firebird user, run the following script:

```
$cd /opt/firebird/bin
$./createAliasDB.sh test.fdb /var/firebird/test.fdb
```

   (Usage is: createAliasDB.sh <dbname> <pathtodb>)

3. As an alternative (for step 2) the steps in the createAliasDB.sh script can be performed manually by:

```
$vi /opt/firebird/aliases.conf
```

   and add the line at the end of the file:

```
test.fdb /var/firebird/test.fdb
```

4. Then create the database:

```
$/opt/firebird/bin/isql -u sysdba -p <password>
SQL>create database 'localhost:test.fdb';
SQL>quit;
```

5. If the *DatabaseAccess* value in /opt/firebird/firebird.conf is set to Full or a restricted path value (for example: DatabaseAccess=/var/firebird) another alternative to step 2 is to create the physical database file directly, using the absolute path with the filename:

```
$/opt/firebird/bin/isql -u sysdba -p <password>
SQL>create database '/var/firebird/test.fdb';
SQL>quit;
```

   If you use this configuration, the database file can also be directly accessed without an entry in the aliases file:

```
$/opt/firebird/bin/isql -u sysdba -p <password>
SQL>connect '/var/firebird/test.fdb';
SQL>quit;
```

Utility Scripts

In addition to the standard install files the following scripts are provided in the bin directory of this release.-

changeDBAPassword.sh
: Change the Firebird SYSDBA user password. For Superserver, this script will change the init script /etc/rc.d/init.d/firebird to use the new password as well.

createAliasDB.sh
: Usage: `createAliasDB.sh <dbname> <dbpath>`

  This script creates a new physical database and adds an entry in the aliases.conf file.

fb_config
: A script that can be used in makefiles to generate the required include paths and lib include directives for the installed version of Firebird. `fb_config -help` will give a complete list of options.

changeGdsLibraryCompatibleLink.sh
: Classic only-Change the client library link for libgds.so between the multithreaded libfbclient.so and the single threaded libfbembed.so library that allows an embedded direct open of the db file. For compatibility with previous installs, libgds.so by default points to libfbembed.so.

Linux Server Tips

"Embedded" or direct access to database files

The Classic install offers an "embedded" mode of access that allows programs to open database files directly. To operate in this mode, a database-enabled user requires privileged access to some of the Firebird configuration and status files.

Now that it is the 'firebird' user (not root) that is the default user to run the software, you need to know how to get a user into the firebird group to enable direct access to databases. It is documented in the readme notes, but the following steps should get you where you need to be.

To add a user (e.g. skywalker) to the firebird group, the root user needs to do:

```
$ usermod -G firebird skywalker
```

Next time 'skywalker' logs on, he can start working with firebird databases.

To list the groups that a user belongs to, type the following at the command line:

```
$ groups
```

> **Warning**
>
> We have been informed of a “gotcha” with the *usermod* syntax in the Debian family of Linux platforms (including Ubuntu). The switches for this command are non-standard and the above usage will remove the user from all other groups.
>
> Please study the online documentation for your distro to work out the syntax you need to add a user to a group in Debian.

Uninstalling on Linux

If you need to uninstall, do it as root user. The following examples use Classic server but the same holds true for SuperServer by replacing the CS with SS.

Uninstalling an RPM package

For rpm packages:

```
$rpm -e FirebirdCS-2.0.0
```

Uninstalling a tarball installation

for the .tar.gz install:

```
$/opt/firebird/bin/uninstall.sh
```

Solaris

Install Firebird Classic & SuperServer on Solaris 2.7 Sparc, not currently available. Please refer older releasenotes as a reference to 2.0 installations.

MacOS X

Install Firebird Classic on MacOS X / Darwin, not currently available. Please refer to older releasenotes as a reference to 2.0 installations.

FreeBSD

Not currently available. Please refer to older releasenotes as a reference to 2.0 installations.

Debian

Not currently available. Please refer to the relevant pages at the Debian site for your Debian version and Firebird 2.0 build.