

□ : / / .

□ Robustness (): 가 , , (recoverability)

□ DB (DBA)가 : 가 admin

(relational)

hierarchical (network model, XML model)

(: record ,

(application)

가 .)

Open mind, open world!!
www.openwith.net

Embedded MySQL

MySQL embedded DB server library `libmysqld`
MySQL server (embed)
(- 가
) MySQL (standalone application)
embedded server application 2 가 가
1. embedded server library 가
 Source build configure `--with-embedded-server`
option
 Binary distribution Max distribution . (
`libmysqld` .)
 RPM embedded server RPM
2. DB start up shut down
가 (client library (-
`mysqlclient`) embedded server library `libmysqld` .)
가
embedded server call calling sequence stubs(dummy routine) 가
client library

Embedded MySQL

MySQL 가 MySQL GPL
 libmysqld link GPL free
 software MySQL AB
 Openwith.net MySQL AB () MySQL porting,
hky@openwith.net (02)3443-4774

Embedded MySQL Server Library

MySQL embedded client/server API 가 (old threaded application) MySQL embedded library

mysql_server_init()	function (가 main() function).
mysql_server_end()	
mysql_thread_init()	MySQL access thread .
mysql_thread_end()	pthread_exit()

libmysqld

libmysqld library MySQL configure --with-embedded-server
 option libmysqld system-specific pthread

libraries . (mysql_config --libmysqld-libs .)

Embedded Server Option

- [server] section . embedded client-server MySQL
- client/server option [mysqld] section .
- embedded MySQL option [embedded] section .
- application option [ApplicationName_SERVER] section .

Embedded MySQL

level `test_libmysqld.c' `GNUmakefile' `test_libmysqld' directory mysql-4.0 GNU `make'

```
`test_libmysqld.c'
/*
 * A simple example client, using the embedded MySQL server library
 */

#include <mysql.h>
#include <stdarg.h>
#include <stdio.h>
#include <stdlib.h>
```

```
MYSQL *db_connect(const char *dbname);
void db_disconnect(MYSQL *db);
void db_do_query(MYSQL *db, const char *query);

const char *server_groups[] = {
    "test_libmysqld_SERVER", "embedded", "server", NULL
};

int
main(int argc, char **argv)
{
    MYSQL *one, *two;

    /* mysql_server_init() must be called before any other mysql
     * functions.
     *
     * You can use mysql_server_init(0, NULL, NULL), and it will
     * initialize the server using groups = {
     *   "server", "embedded", NULL
     * }.
     *
     * In your $HOME/.my.cnf file, you probably want to put:

[test_libmysqld_SERVER]
language = /path/to/source/of/mysql/sql/share/english

     * You could, of course, modify argc and argv before passing
     * them to this function. Or you could create new ones in any
     * way you like. But all of the arguments in argv (except for
     * argv[0], which is the program name) should be valid options
     * for the MySQL server.
     *
     * If you link this client against the normal mysqlclient
```

```
* library, this function is just a stub that does nothing.
*/
mysql_server_init(argc, argv, (char **)server_groups);

one = db_connect("test");
two = db_connect(NULL);

db_do_query(one, "SHOW TABLE STATUS");
db_do_query(two, "SHOW DATABASES");

mysql_close(two);
mysql_close(one);

/* This must be called after all other mysql functions */
mysql_server_end();

exit(EXIT_SUCCESS);
}

static void
die(MYSQL *db, char *fmt, ...)
{
    va_list ap;
    va_start(ap, fmt);
    vfprintf(stderr, fmt, ap);
    va_end(ap);
    (void)putc('\n', stderr);
    if (db)
        db_disconnect(db);
    exit(EXIT_FAILURE);
}

MYSQL *
```

```
db_connect(const char *dbname)
{
    MYSQL *db = mysql_init(NULL);
    if (!db)
        die(db, "mysql_init failed: no memory");
    /*
     * Notice that the client and server use separate group names.
     * This is critical, because the server will not accept the
     * client's options, and vice versa.
     */
    mysql_options(db, MYSQL_READ_DEFAULT_GROUP, "test_libmysqld_CLIENT");
    if (!mysql_real_connect(db, NULL, NULL, NULL, NULL, dbname, 0, NULL, 0))
        die(db, "mysql_real_connect failed: %s", mysql_error(db));

    return db;
}

void
db_disconnect(MYSQL *db)
{
    mysql_close(db);
}

void
db_do_query(MYSQL *db, const char *query)
{
    if (mysql_query(db, query) != 0)
        goto err;

    if (mysql_field_count(db) > 0)
    {
        MYSQL_RES *res;
        MYSQL_ROW row, end_row;
```

```
int num_fields;

if (!(res = mysql_store_result(db)))
    goto err;

num_fields = mysql_num_fields(res);
while ((row = mysql_fetch_row(res)))
{
    (void)fputs(">> ", stdout);

    for (end_row = row + num_fields; row < end_row; ++row)
        (void)printf("%s \t", row ? (char*)*row : "NULL");

    (void)fputc(' \n', stdout);
}
(void)fputc(' \n', stdout);
mysql_free_result(res);
}
else
    (void)printf("Affected rows: %lld \n", mysql_affected_rows(db));

return;

err:
    die(db, "db_do_query failed: %s [%s]", mysql_error(db), query);
}

`GNUmakefile`

# This assumes the MySQL software is installed in /usr/local/mysql
inc      := /usr/local/mysql/include/mysql
lib      := /usr/local/mysql/lib

# If you have not installed the MySQL software yet, try this instead
#inc     := $(HOME)/mysql-4.0/include
#lib     := $(HOME)/mysql-4.0/libmysqld

CC      := gcc
```

```
CPPFLAGS := -I$(inc) -D_THREAD_SAFE -D_REENTRANT
CFLAGS := -g -W -Wall
LDFLAGS := -static
# You can change -lmysqld to -lmysqlclient to use the
# client/server library
LDLIBS = -L$(lib) -lmysqld -lz -lm -lcrypt

ifneq (, $(shell grep FreeBSD /COPYRIGHT 2>/dev/null))
# FreeBSD
LDFLAGS += -pthread
else
# Assume Linux
LDLIBS += -lpthread
endif

# This works for simple one-file test programs
sources := $(wildcard *.c)
objects := $(patsubst %c,%o,$(sources))
targets := $(basename $(sources))

all: $(targets)

clean:
rm -f $(targets) $(objects) *.core
```

Embedded MySQL ()

<Deeply Embedded>

- 3COM
- Avery Dennison
- LeapFrog SchoolHouse
- NEC
- NetIQ
- Quest Software
- RLX Technologies
- [S2 Security Corporation](#)
- SAS
- SS8 Networks
- Standard Networks
- Symantec
- Monta Vista
-

<Bundled>

- Active Voice
- American Education Corporation
- BMC
- Deutsch Post
- F5 Networks
- Hyperion
- McAfee
- Motorola
- Right Now Technologies
- [Sterling Commerce](#)
- Tekelec
- UPS
- Veritas