NAME

db5.3_codegen — generate application code for Berkeley DB

SYNOPSIS

db5.3_codegen [**-Vv**] [**-a** *api*] [**-i** *file*] [**-o** *prefix*]

DESCRIPTION

The **db5.3_codegen** utility generates application code to create and configure Berkeley DB database environments and databases based on a simple description language and writes it to one or more output files. The generated code may need modification, in the case of complicated applications, but will usually significantly reduce the time required to create Berkeley DB applications.

The options are as follows:

-a *api*
Generate code for the specified API (currently, only "c" is accepted).

-i *file*
Specify an input *file*; by default, standard input is used.

-o *prefix*
Specify an output file *prefix*; by default, "application" is used.

-V Write the library version number to standard output and exit.

-v Run in verbose mode.

The **db5.3_codegen** utility exits 0 on success, and >0 if an error occurs.

C Language Specific Information

By default, when the **db5.3_codegen** utility generates C-language code, the output file is named "application.c". The output filename can be specified with the **-o** option.

At the beginning of the output file is a list of public database environment (*DB_ENV*) handles and database (*DB*) handles, as specified by the description language. The database environment handle variables are named "*XXX*_dbenv", where "*XXX*" is the name of the environment in the input specification. For databases associated with a database environment, the database handle variables are named "*XXX_YYY*", where "*XXX*" is the name of the environment, and "*YYY*" is the name of the database. For standalone databases, the database handle variables are named "*XXX*", where "*XXX*" is the name of the database.

There are two public functions in the output file: **bdb_startup**() and **bdb_shutdown**(). The **bdb_startup**() function should be called to create and configure the database environments and databases, and the **bdb_shutdown**() function should be called to gracefully shut down the environments and databases.

Specification Language

The **db5.3_codegen** uses a simple description language:

- Lines in the input consist of white-space separated tokens.
- Tokens are case-insensitive.
- Empty lines and lines where the first non-space character is a hash mark ("#") are ignored. In addition, hash marks may appear in lines, in which case the content of the line from the hash mark to the end of the line is ignored.

There are two top-level objects: "environment" and "database", which correspond to database environments and databases, respectively. These top-level objects can be associated with keywords to describe their configuration and relationships.

For example, the following input would create two standalone databases:

```
database data_one {
    type btree
}
database data_two {
    type btree
}
```

In this case, there would be no *DB_ENV* handle, and the public *DB* handles would be:

```
DB      *data_one;
DB      *data_two;
```

For example, the following input would create a database environment which contains three databases:

```
environment myenv {
    database data_one {
        type btree
    }
    database data_two {
        type btree
    }
    database data_three {
        type btree
    }
}
```

In this case, the public *DB_ENV* and *DB* handles would be:

```
DB_ENV  *myenv_dbenv;
DB      *myenv_data_one;
DB      *myenv_data_two;
DB      *myenv_data_three;
```

A variety of keywords can be specified for the databases and the environments. For example, the cache size can be specified for the database environment, and the page size can be specified for the database, as well as for secondary relationships:

```
environment myenv {
    cachesize 2 0 10
    database data_one {
        type btree
        pagesize 1024
    }
    database data_two {
        primary data_one
        secondary_offset 10 15
        type btree
        pagesize 32768
    }
    database data_three {
        type btree
        pagesize 512
    }
}
```

Environment Keywords

environment Start a database environment block.

There must be three tokens on the line: the keyword, the name of the environment and an opening brace ("{").

home Specify the database environment home directory.

There must be two tokens on the line: the keyword and the home directory.

cachesize Specify the database environment cache size.

There must be two tokens on the line: the keyword, the gigabytes of cache, the bytes of cache, and the number of caches (the number of underlying physical areas into which the cache is logically divided).

private Specify the database environment is private.

There must be one token on the line: the keyword by itself.

} End the database environment block.

There must be one token on the line: the keyword by itself.

Database Keywords

database Start a database block.

There must be three tokens on the line: the keyword, the name of the database and an opening brace ("{").

custom Specify a custom key-comparison routine. This is used when the Btree database requires a specific sort that **db5.3_codegen** cannot generate. A stub key comparison routine will be created and configured for the database which should be modified as necessary. See the "**key_type**" keyword for more information.

There must be one token on the line: the keyword by itself.

dupsort Configure the database to support sorted duplicates.

There must be one token on the line: the keyword by itself.

extentsize Configure the size of the Queue database extent files.

There must be two tokens on the line: the keyword and the extent file size, as a number of pages.

key_type Configure a integral type key-comparison routine. This is used when the Btree database key is an integral type (such as "*unsigned int*" or "*u_int32_t*"). Any C-language integral type may be specified. See the "**custom**" keyword for more information. A Btree comparison routine based on the type of the key will be created and configured.

There must be two tokens on the line: the keyword and the type.

pagesize Configure the database page size.

There must be two tokens on the line: the keyword and the page size in bytes.

primary Configure the database as a secondary index. A stub secondary callback routine will be created and configured for the database, which should be modified as necessary. See the "**secondary_offset**" keyword for more information.

There must be two tokens on the line: the keyword and the name of the primary database for which this database is a secondary.

recnum Configure the Btree database to support record number access.

There must be one token on the line: the keyword by itself.

re_len Configure the record length for a Queue database or a fixed-length Recno database.

There must be two tokens on the line: the keyword and the length of a record, in bytes.

secondary_offset Configure a secondary callback routine based on a byte string found in the primary database's data item.

There must be three tokens on the line: the keyword, the byte offset from the beginning of the primary data item where the secondary key occurs, and the length of the secondary key in bytes.

transaction Configure the database (and, by extension, the database environment), to be transactional.

There must be one token on the line: the keyword by itself.

type Configure the database type.

There must be two tokens on the line: the keyword and the type, where the type is one of "btree", "hash", "queue" or "recno".

} End the database environment block.

There must be one token on the line: the keyword by itself.

AUTHORS

Thorsten Glaser <tg@debian.org> wrote this manual page for the Debian project (but may be used by others) after the original HTML format documentation