Redis with C

To use Redis with C, you need a C Redis client library. Here, you can learn how to use hiredis to connect to a Redis database from an application written in C.

Additional C client libraries are available. To learn more, see the C section of the Redis Clients page.

Install hiredis

Download the latest hiredis release from the GitHub repository.

Connect to Redis

The following code creates a connection to Redis using the hiredis synchronous API:

```c
#include "hiredis.h"

redisContext *c = redisConnect("hostname", port);
if (c != NULL && c->err) {
    printf("Error: %s\n", c->errstr);
    // handle error
} else {
    printf("Connected to Redis\n");
}

redisReply *reply;
reply = redisCommand(c, "AUTH password");
freeReplyObject(reply);
...
redisFree(c);
```

To adapt this example to your code, replace the following values with your database’s values:

- In line 1, set the hostname of redisConnect to your database’s hostname or IP address
- In line 1, set the port of redisConnect to your database’s port
- In line 6, replace “password” with your database’s password

Example code for Redis commands

Once connected to Redis, you can read and write data with Redis command functions.

The following code snippet assigns the value bar to the Redis key foo, reads it back, and prints it:
// open a connection to Redis
...

redisReply *reply;

reply = redisCommand(c, "SET %s %s", "foo", "bar");
freeReplyObject(reply);

reply = redisCommand(c, "GET %s", "foo");
printf("%s
", reply->str);
freeReplyObject(reply);

Example output:

$ gcc example_hiredis.c -o example_hiredis
$ ./example_hiredis
Connected to Redis
bar

SSL

The hiredis client supports SSL natively as of version 1.0.0. For older versions of hiredis, you can use stunnel to secure the connection.

Updated: November 2, 2021