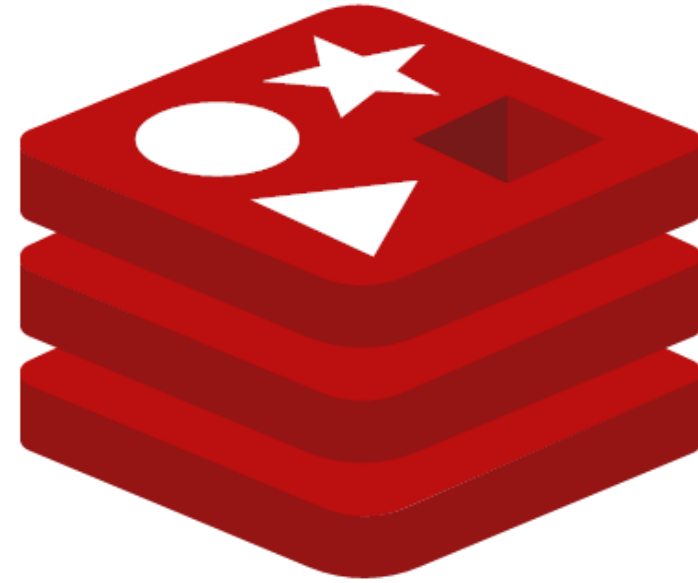


REDIS



JAVIER MARTÍNEZ ÁLVAREZ - UO258092

ISMAEL CADENAS ALONSO – UO251025

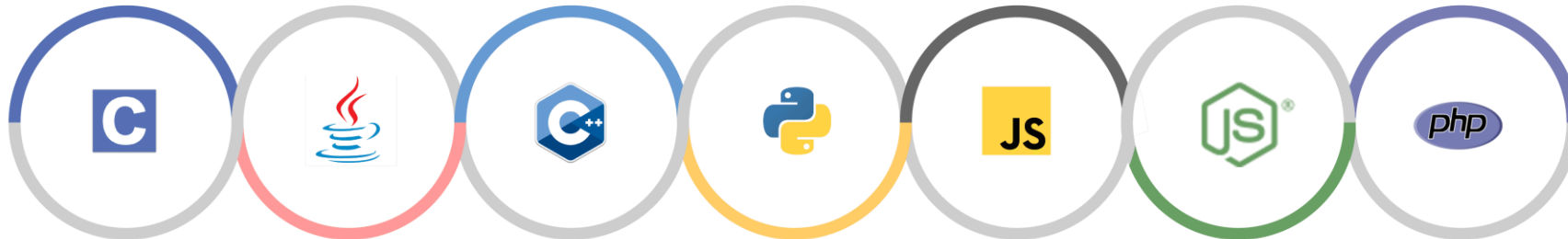
IGNACIO BERMEJO ÁLVAREZ – UO240279

CHRISTIAN PELÁEZ FERNÁNDEZ - UO258764

Introducción

¿Qué es?

- **Redis** es un motor de base de datos en memoria, basado en el almacenamiento en tablas de hashes (clave/valor).
- El nombre Redis significa **RE**mote **D**ictionary **S**erver (servidor de diccionario remoto).
- Entre los casos de uso principales de Redis se encuentran el almacenamiento en caché, la administración de sesiones, pub/sub y las clasificaciones.
- Tiene licencia BSD, está escrito en código C optimizado y admite numerosos lenguajes de desarrollo.
 - ActionScript, C, C++, C#, Clojure, CommonLisp, Erlang, Go, Haskell, haXe, Io, Java, JavaScript (Node.js), Lua, Objective-C, Perl, PHP, Pure Data, Python, Ruby, Scala, Smalltalk y Tcl.



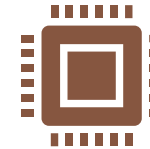
¿Cómo funciona?



Incorpora un conjunto de estructuras de datos en memoria versátiles que le permiten crear con facilidad diversas aplicaciones personalizadas.



Permite el uso de estructuras más complejas y flexibles que abren una serie de posibilidades ante las distintas necesidades de aplicaciones de negocio.



Proporciona acceso a estructuras de datos mutables a través de un conjunto de comandos, que se envían utilizando un modelo servidor-cliente con sockets TCP y un protocolo simple.



Ofrece 5 estructuras de datos con los que es posible modelar la solución más adecuada para cubrir las necesidades de un proyecto.

¿Cómo funciona?

```
> LPUSH lista Mirai
(integer) 1
> LPUSH lista BiGeek
(integer) 2
> LPUSH lista Tadaima
(integer) 3
> LRANGE lista 0 5
1) "Mirai"
2) "BiGeek"
3) "Tadaima"
```

← LISTS

↓ SETS

STRINGS



```
> SET ClaveM Mirai is my buddy
OK
> GET ClaveM
"Mirai is my buddy"
```

```
> SADD MiSet Mirai
(integer) 1
> SADD MiSet BiGeek
(integer) 1
> SADD MiSet Tadaima
(integer) 1
> SADD MiSet BiGeek
(integer) 0
> SMEMBERS MiSet
1) "Mirai"
2) "BiGeek"
3) "Tadaima"
```

```
> ZADD MiSetOrdenado 1 Mirai
(integer) 1
> ZADD MiSetOrdenado 2 BiGeek
(integer) 1
> ZADD MiSetOrdenado 3 Tadaima
(integer) 1
> ZADD MiSetOrdenado 4 Mirai
(integer) 1
> ZADD MiSetOrdenado 5 Mirai
(integer) 1
> ZRANGE MiSetOrdenado 0 10 WITHSCORES
1) "BiGeek"
2) "2"
3) "Tadaima"
4) "3"
5) "Mirai"
6) "5"
```



SORTED SETS

```
> HMSET empleado empid "99" nombre "Juan" apaterno "Perez"
fnac "10071980" Salario "10000"
OK
```

HASHES



```
> HGET empleado empid
"99"
> HGET empleado salario
"10000"
> HGETALL empleado
empid
"99"
nombre
"Juan"
apaterno
"Perez"
fnac
"10071980"
Salario
"10000"
```

Ventajas y desventajas

Fácil
configuración

Una variedad de
tipos de datos



Una velocidad muy
por rápida gracias a
su almacenamiento
en memoria

Posibilidad de persistir
datos en disco para
recuperación ante
fallas

Alta disponibilidad

El método de
persistencia RDB
consume mucho I/O
(escritura en disco)

Curva de
aprendizaje
baja

Extensible
usando LUA
scripting

Compatibilidad
con gran variedad
de lenguajes

Todos los datos
trabajados deben
encajar en la memoria
(en caso de no usar
persistencia física)

Atributos de calidad



RENDIMIENTO



EXTENSIBILIDAD



USABILIDAD

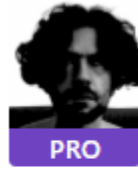


COMPATIBILIDAD



FIABILIDAD

Stakeholders



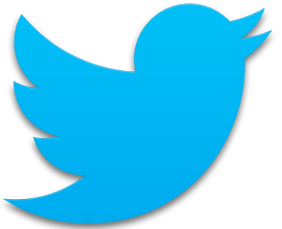
Salvatore Sanfilippo antirez
Computer programmer based in Sicily, Italy. I mostly write OSS software. Born 1977. Not a puritan.
📍 Campobello di Licata, Sicily, Italy



Owns this repository



Committed to this repository in the past day



redislabs
HOME OF REDIS



Socios



Colaboradores

Clientes

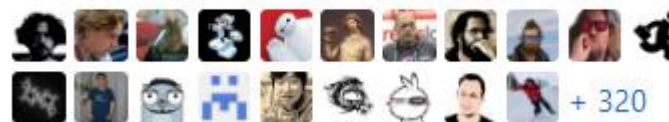


Pieter Noordhuis pietern

📍 NL



340 direct contributors



Committed to this repository

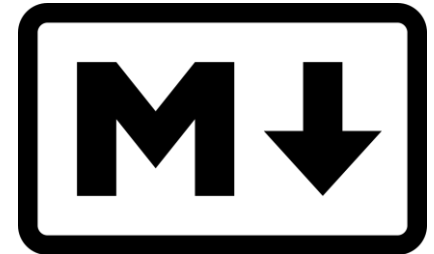
GIT

Repositorios


Documentación -> <https://github.com/antirez/redis-doc>




Página web -> <https://github.com/antirez/redis-io>








Desarrollo -> <https://github.com/antirez/redis>









Desarrollo

 antirez / **redis**


 Watch ▾ 2.8k  Star 42.1k  Fork 16.4k







 Code  Issues 1,757  Pull requests 836  Actions  Projects 0  Security  Insights

Redis is an in-memory database that persists on disk. The data model is key-value, but many different kind of values are supported: Strings, Lists, Sets, Sorted Sets, Hashes, Streams, HyperLogLogs, Bitmaps. <http://redis.io>

 9,129 commits  67 branches  0 packages  229 releases  339 contributors  BSD-3-Clause

Branch: unstable ▾ [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download ▾](#)

 antirez Merge branch 'faster-rdb-loading' into unstable ✓ Latest commit 3072498 2 hours ago

 .github/workflows	change CI to build and run the module api tests	last month
 deps	Update linenoise.	28 days ago
 src	Merge branch 'faster-rdb-loading' into unstable	2 hours ago
 tests	Merge branch 'lcs' into unstable	3 days ago
 utils	Merge pull request #6052 from jtru/better-systemd-integration-v2	4 months ago
 .gitignore	fix comments in latency.c	28 days ago

Integración continua

GitHub Actions



Don't allow empty spaces in ACL usernames ...

antirez committed 5 hours ago ✓

Don't allow empty spaces in ACL k

antirez committed 5 hours ago ✓

Merge branch 'unstable' of github

antirez committed 6 hours ago

Fix HELLO reply in Sentinel mode,

antirez committed 6 hours ago

Merge pull request #7078 from karelrooted/fix/geo-hash-edge-case ...

antirez committed 11 hours ago ✓

All checks have passed

3 successful checks

- ✓ CI / test-ubuntu-latest (push) [Details](#)
- ✓ CI / build-ubuntu-old (push) [Details](#)
- ✓ CI / build-macos-latest (push) [Details](#)

<> Code 1,759 Issues 835 Pull requests 835 Actions Projects 0 Security Insights

✓ Merge pull request #7074 from hayleeliu/unstable

unstable 767977c

CI

on: push

- ✓ test-ubuntu-latest
- ✓ build-ubuntu-old
- ✓ build-macos-latest

CI

This run Workflow file

```
.github/workflows/ci.yml
1 name: CI
2
3 on: [push, pull_request]
4
5 jobs:
6   test-ubuntu-latest:
7     runs-on: ubuntu-latest
8     steps:
9       - uses: actions/checkout@v1
10      - name: make
11        run: make
12      - name: test
13        run: |
14          sudo apt-get install tc18.5
15          ./runtest --clients 2 --verbose
16      - name: module api test
17        run: ./runtest-moduleapi --clients 2 --verbose
18
19   build-ubuntu-old:
20     runs-on: ubuntu-16.04
21     steps:
22       - uses: actions/checkout@v1
23       - name: make
24         run: make
```

Comunidad

Formas de contribuir:

- Añadir funcionalidad a la aplicación
- Arreglar fallos o reportarlos mediante issues
- Mejorar la documentación o la página web
- Mantener y añadir librerías de clientes



Arquitectura y patrones



Kashish Aggarwal

Today at 02:27

Hi Christian,

This Support Platform is for Redis Enterprise customers. I believe you need to know more about Open Source Redis (the architecture for Enterprise and Open Source Redis is not exactly the same).

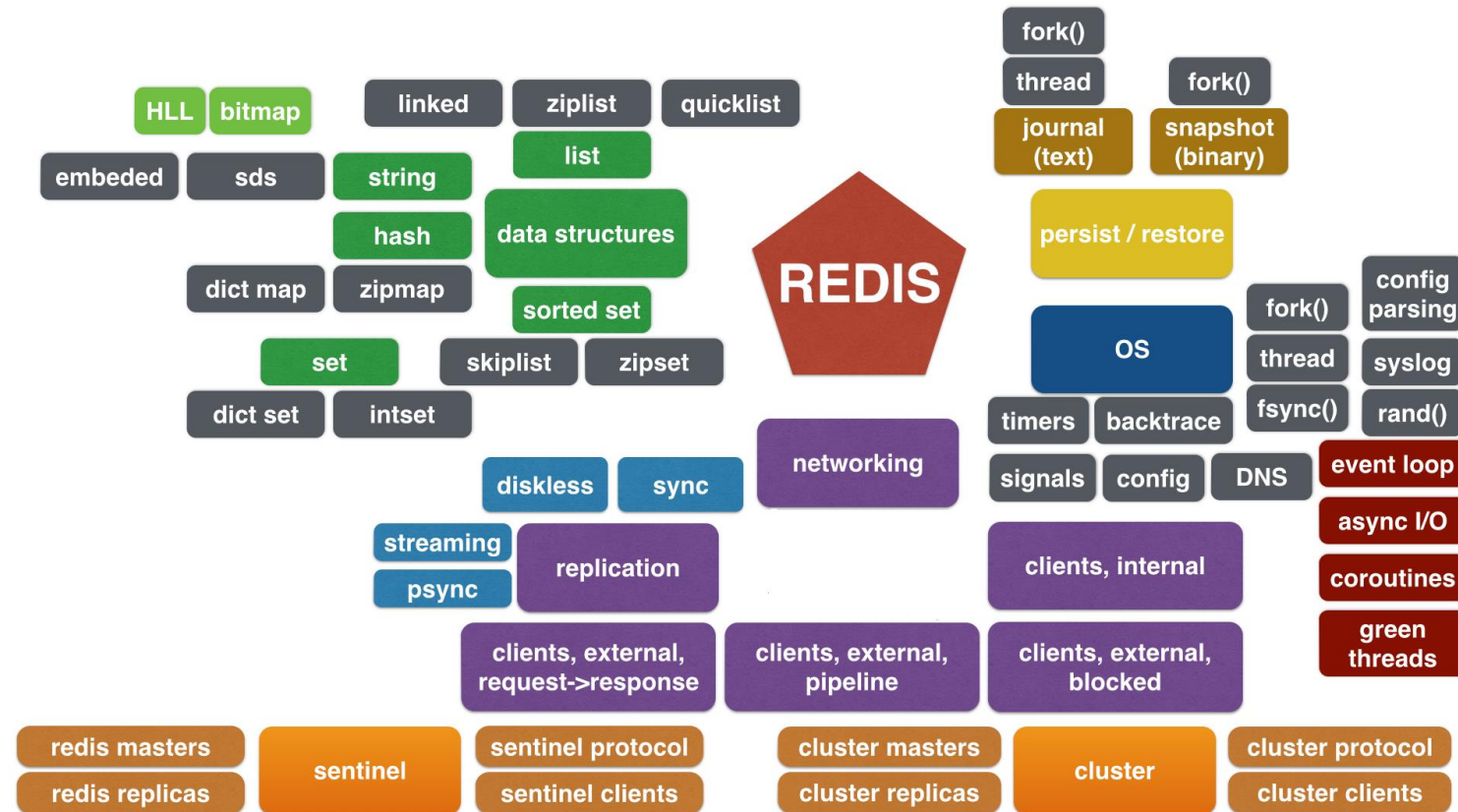
Redis is an in-memory server running on a host listening to the client's request on port 6379 (default; although it is configurable). It is capable of serving hundreds of millions of operations per second with sub-millisecond latency. Unfortunately, there isn't any diagram that demonstrates the architecture for open-source Redis on the official website (www.redis.io). You can access the official documentation [here](#) and the source code on the Github page [here](#). Some detailed information about different Redis components are mentioned in the readme.md on the Github page I shared above.

Let me know if this helps you or if you have any more questions.

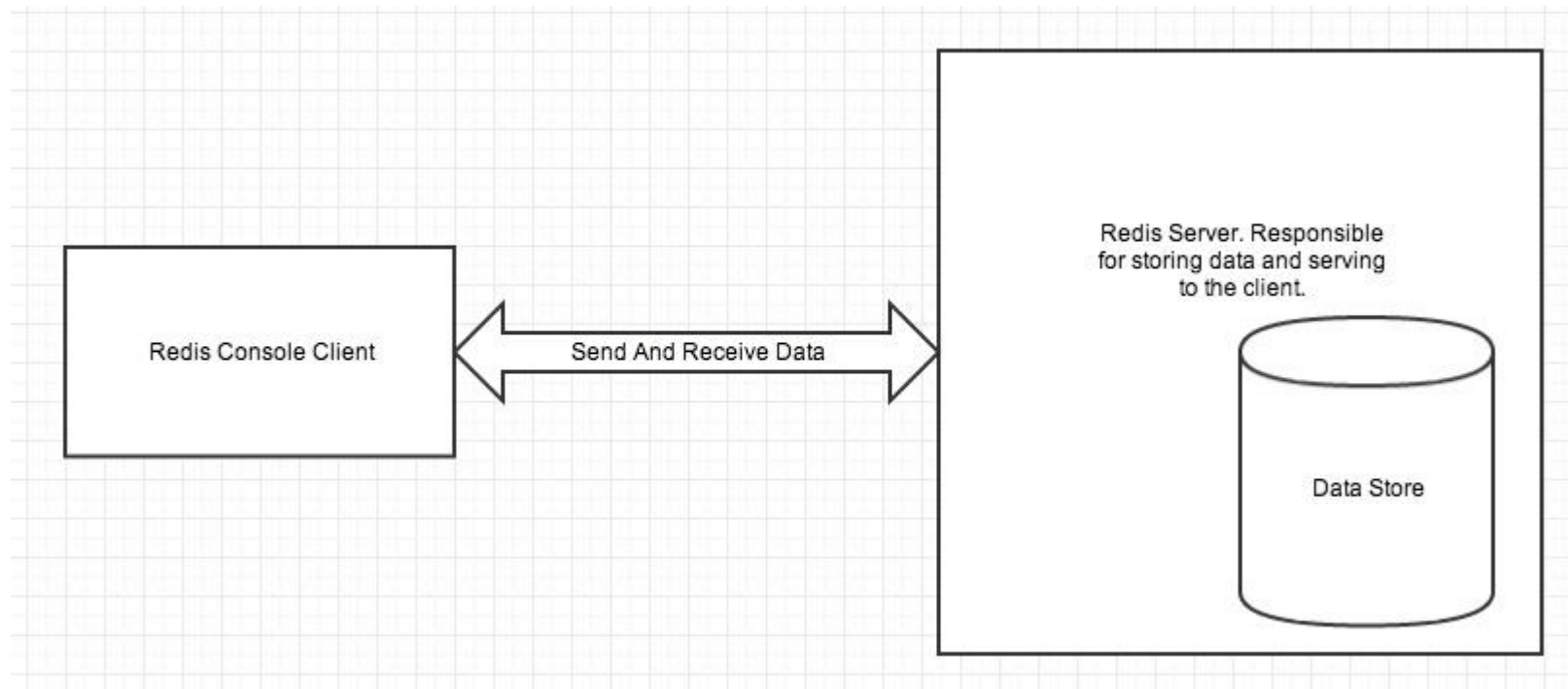
Good luck with your presentation :)

Best Regards

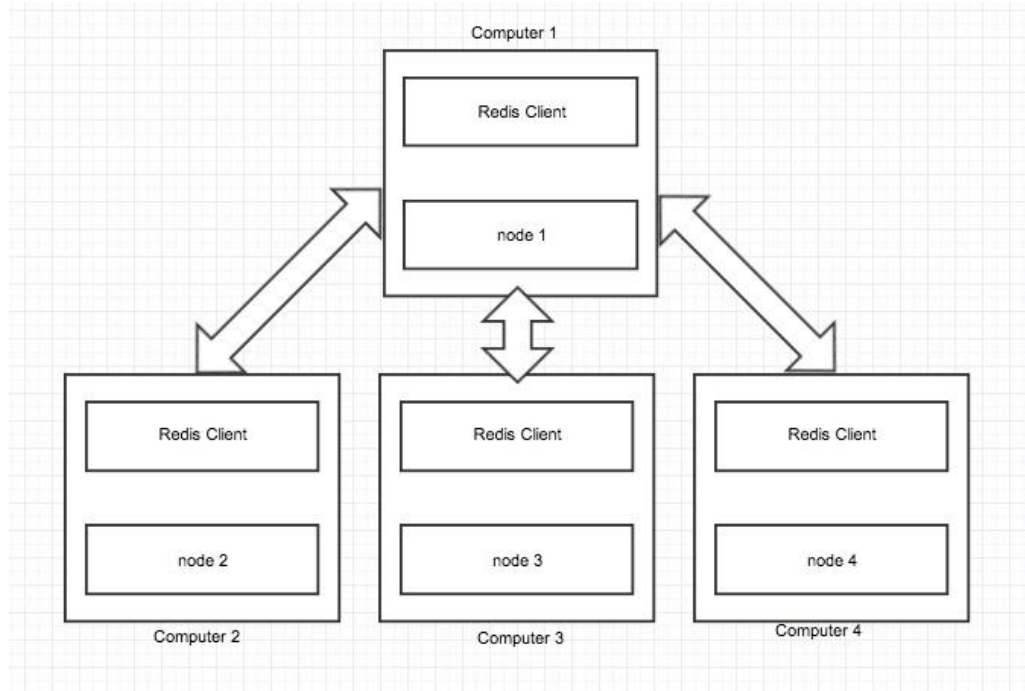
Arquitectura



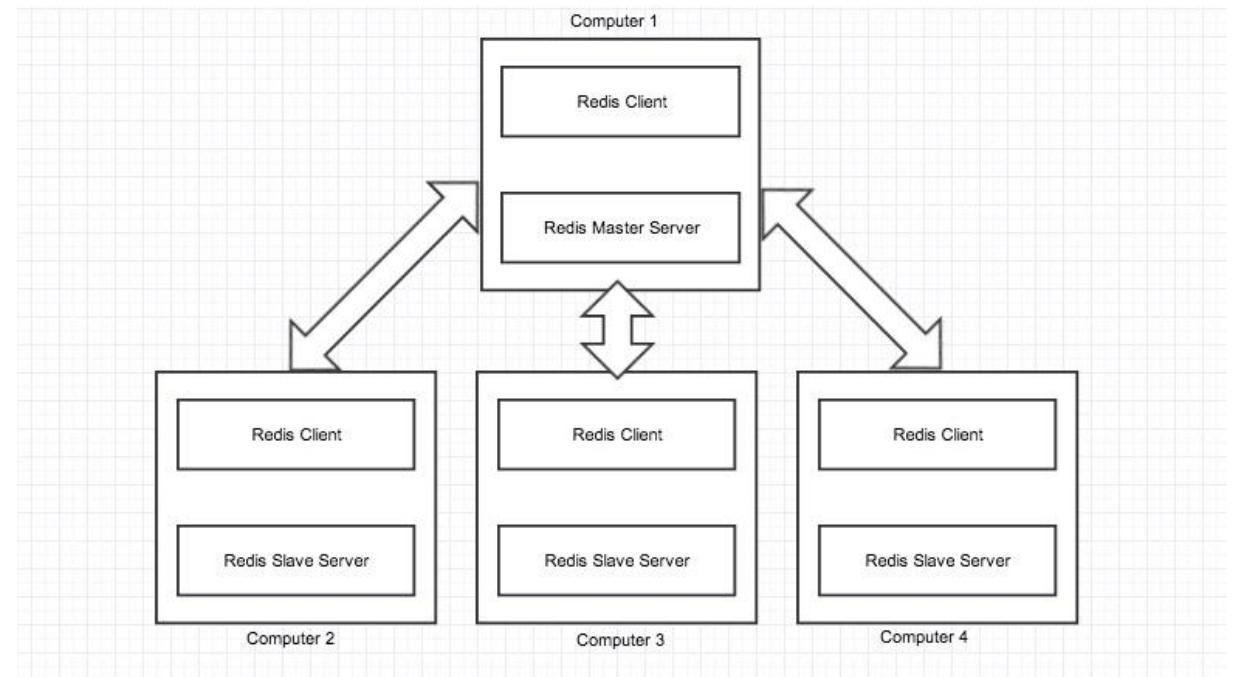
Única instancia



Múltiples instancias



Clústering



Replicación

Pruebas

COMPILACIÓN Y PRUEBAS DE REDIS

Compilación

PREREQUISITOS

- Linux, OSX, OpenBSD, NetBSD o FreeBSD
- GCC
- make

CLONAR ÚLTIMA VERSIÓN ESTABLE (SRC)

```
$ curl -O http://download.redis.io/redis-stable.tar.gz
```

```
$ tar xvzf redis-stable.tar.gz
```

```
$ cd redis-stable
```

COMPILAR

```
$ make
```

COMPROBAR QUE SE HA COMPILADO CORRECTAMENTE

```
$ make test
```

```
CC lazyfree.o
CC module.o
CC evict.o
CC expire.o
CC geohash.o
CC geohash_helper.o
CC childinfo.o
CC defrag.o
CC siphash.o
CC rax.o
CC t_stream.o
CC listpack.o
CC localtime.o
CC lolwut.o
CC lolwut5.o
LINK redis-server
INSTALL redis-sentinel
CC redis-cli.o
LINK redis-cli
CC redis-benchmark.o
LINK redis-benchmark
INSTALL redis-check-rdb
INSTALL redis-check-aof

Hint: It's a good idea to run 'make test' ;)

make[1]: Leaving directory '/mnt/c/Users/javie/Desktop/redis/redis-stable/src'
jaluma@JaviPC ➤ /mnt/c/Users/javie/Desktop/redis/redis-stable
```

Finalizando el proceso de compilación

Ejecución

```
$ cd src
```

ARRANCAR SERVIDOR (Configuración *DEFAULT*)

```
$ ./redis-server
```

ARRANCAR SERVIDOR (Configuración *CUSTOM*)

```
$ ./redis-server <<config-file.conf>>
```

```
jaluma@JaviPC /mnt/c/Users/javie/Desktop/redis/redis-stable > cd src
jaluma@JaviPC /mnt/c/Users/javie/Desktop/redis/redis-stable/src > ./redis-server
2610:C 12 Apr 2020 16:35:58.443 # 000000000000 Redis is starting 000000000000
2610:C 12 Apr 2020 16:35:58.443 # Redis version=5.0.8, bits=64, commit=00000000, modified=0, pid=2610, just started
2610:C 12 Apr 2020 16:35:58.443 # Warning: no config file specified, using the default config. In order to specify a co
nfig file use ./redis-server /path/to/redis.conf
2610:M 12 Apr 2020 16:35:58.444 * Increased maximum number of open files to 10032 (it was originally set to 1024).

                                     _____
                                    /  _  /
                                   /  /  /
                                  /  /  /
                                 /  /  /
                                /  /  /
                               /  /  /
                              /  /  /
                             /  /  /
                            /  /  /
                           /  /  /
                          /  /  /
                         /  /  /
                        /  /  /
                       /  /  /
                      /  /  /
                     /  /  /
                    /  /  /
                   /  /  /
                  /  /  /
                 /  /  /
                /  /  /
               /  /  /
              /  /  /
             /  /  /
            /  /  /
           /  /  /
          /  /  /
         /  /  /
        /  /  /
       /  /  /
      /  /  /
     /  /  /
    /  /  /
   /  /  /
  /  /  /
 /  /  /
/  /  /

Redis 5.0.8 (00000000/0) 64 bit

Running in standalone mode
Port: 6379
PID: 2610

http://redis.io

2610:M 12 Apr 2020 16:35:58.446 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core
/somaxconn is set to the lower value of 128.
```

Ejecutando el servidor

Pruebas

Lista de comandos disponibles: <https://redis.io/commands>

CONECTARSE A REDIS CLI

```
$ cd src
```

```
$ ./redis-cli
```

EJEMPLO COMANDOS REDIS CLI:

- Variable (set y get)

```
$ set <<variable-name>> <<value>>
```

```
$ get <<variable-name>>
```

- Incrementar un contador

```
$ incr <<counter-name>>
```

```
jaluma@JaviPC > /mnt/c/Users/javie/Desktop/redis/redis-stable/src > ./redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> set subject arquisoft
OK
127.0.0.1:6379> get subject
"arquisoft"
127.0.0.1:6379> incr day
(integer) 1
127.0.0.1:6379> incr day
(integer) 2
127.0.0.1:6379> get day
"2"
127.0.0.1:6379>
```

*Pruebas usando **Redis CLI***

Ejemplo usando .NET Core

```
private static void Main(string[] args)
{
    var redis = ConnectionMultiplexer.Connect(configuration: "localhost");
    var db = redis.GetDatabase();

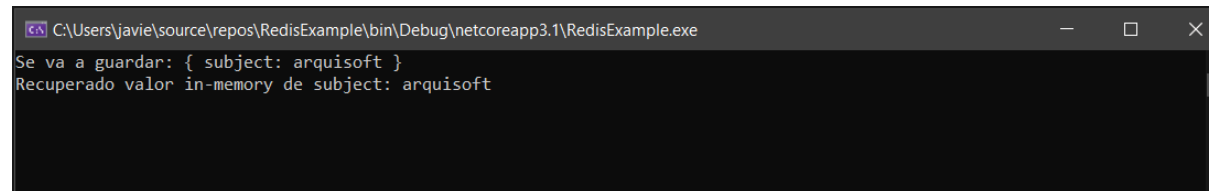
    const string key = @"subject";
    const string value = @"arquisoft";
    Console.WriteLine($"Se va a guardar: {{ {key}: {value} }}");

    db.StringSet(key, value);

    string cacheValue = db.StringGet(key);
    Console.WriteLine($"Recuperado valor in-memory de {key}: {cacheValue}");

    Console.ReadKey();
}
```

Ejemplo usando .NET Core y StackExchange.Redis

A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Users\javie\source\repos\RedisExample\bin\Debug\netcoreapp3.1\RedisExample.exe. The window contains two lines of text: "Se va a guardar: { subject: arquisoft }" and "Recuperado valor in-memory de subject: arquisoft".

```
C:\Users\javie\source\repos\RedisExample\bin\Debug\netcoreapp3.1\RedisExample.exe
Se va a guardar: { subject: arquisoft }
Recuperado valor in-memory de subject: arquisoft
```

Salida del programa



¿Alguna pregunta?
