**一：简介**

Redis是一个开源的高性能key-value数据库。Redis是Remote DIctionary Server（远程字典服务器）的缩写，它以字典结构存储数据，并允许其他应用通过TCP协议读写字典中的内容。Redis字典中的value支持的数据类型有：字符串，散列，列表，集合，有序集合。

Redis数据库中的所有数据都存储在内存中，内存的读写速度远快于硬盘，因此Redis在性能上对比其他基于硬盘存储的数据库有非常明显的优势。但是将数据存储在内存中也有问题，程序退出后内存中的数据会丢失。不过Redis提供了持久化的功能，可以将内存中的数据异步写入到硬盘中，同时不影响继续提供服务。

Redis虽然是作为数据库开发的，但由于其提供了丰富的功能，越来越多的人将其用作缓存、队列。

Redis可以为每个键设置生存时间，生存时间到期后键会自动被删除，这一功能配合出色的性能让Redis可以作为缓存系统来使用，而且由于Redis支持持久化和丰富的数据类型，使其成为了另一个非常流行的缓存系统Memcached的有力竞争者。Redis是单线程模型，而Memcached支持多线程，所以在多核服务器上后者的性能更高一些。然而， Redis在绝大部分场合下其性能都不会成为瓶颈。

作为缓存系统，Redis还可以限定数据占用的最大内存空问，在数据达到空间限制后可以按照一定的规则自动淘汰不需要的键。

除此之外，Redis的列表类型键可以用来实现队列，并支持阻塞式读取，可以很容易地实现一个高性能的优先级队列。同时在更高层面上，Redis还支持“发布/订阅”的消息模式。

Redis提供了一百多个命令，但是常用的却只有十几个，并且每个命令都很容易记忆，Redis的命令比SQL语句要简单很多。

Redis提供了几十种不同编程语言的客户端库，这些库都很好地封装了Redis的命令，使得在程序中与Redis进行交互变得更容易。

Redis使用C语言开发，代码量只有3万多行。这降低了用户通过修改Redis源代码来使之更适合自己项目需要的门槛。

**二：准备**

1：安装

Redis约定次版本号（即第一个小数点后的数字）为偶数的版本是稳定版（如2.4版、2.6版），奇数版本是非稳定版（如2.5版、2.7版），推荐使用稳定版本进行开发和在生产环境使用。

Redis兼容大部分POSIX系统，包括Linux，BSD等，在这些系统中，推荐直接下载源代码编译安装以获得最新的稳定版本。下载源码包后解压即可使用make命令完成编译，完整的命令如下：

wget http://download.redis.io/releases/redis-3.0.5.tar.gz

tar xzf redis**-**3.0.5.tar.gz

cd redis**-**3.0.5

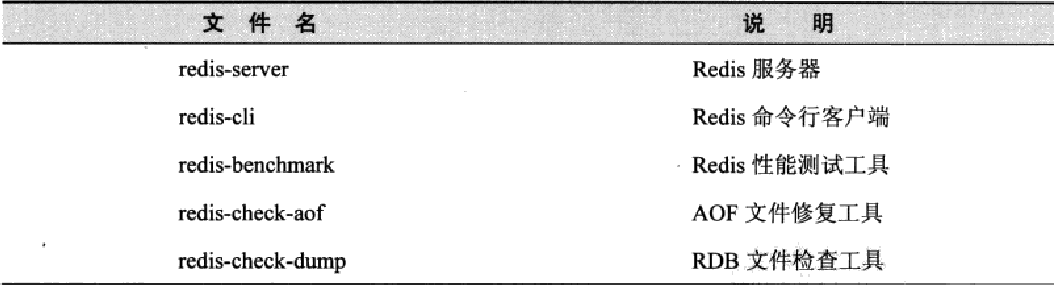
make

make install

Redis没有其他外部依赖，安装过程很简单。make后在源代码目录的src子目录中可以找到若干可执行程序，执行make install命令后会将这可执行程序复制到/usr/local/bin/目录下，以便以后执行程序时可以不用输入完整的路径。

在实际运行Redis前推荐使用make test命令测试Redis是否编译正确。

下表列出了Redis的可执行文件的名称以及对应的说明，如果在编译后执行了make install命令，这些程序会被复制到/usr/local/bin/目录内，所以在命令行中直接输入程序名称即可执行：



最常使用的两个程序是redis-server和redis-cli，其中redis-server是Redis的服务器，启动Redls即运行redis-server；redis-cli是Redis自带的Redis命令行客户端。

2：启动

启动Redis有直接启动和通过初始化脚本启动两种方式，分别适用于开发环境和生产环境。直接运行redis-server即可启动Redis：

# redis-server

3733**:**C 28 Nov 15**:**01**:**20.735 # Warning**:** no config file specified**,** using the **default** config**.** In order to specify a config file use redis**-**server **/**path**/**to**/**redis**.**conf

**...**

3733**:**M 28 Nov 15**:**01**:**20.754 **\*** DB loaded from disk**:** 0.004 seconds

3733**:**M 28 Nov 15**:**01**:**20.755 **\*** The server is now ready to accept connections on port 6379

Redis服务器默认会使用6379端口（6379在手机按键上是MERZ对应的号码，而MERZ取自意大利女明星Alessia Merz的名字。MERZ长期以来被Redis作者antirez及其朋友当作愚蠢的代名词。后来Redis作者在开发Redis时就选用了这个端口。具体见http://oldblog.antirez.com/post/redis-as-LRU-cache.html）。通过--port参数可以自定义端口号：

# redis-server --port 6666

在Linux系统中可以通过初始化脚本启动Redis，使Redis能随系统自动运行，在生产环境中推荐使用此方法运行Redis。在Redis源代码目录的utils文件夹中有一个名为redis\_init\_script的初始化脚本文件，内容如下：

*#!/bin/sh*

*#*

*# Simple Redis init.d script conceived to work on Linux systems*

*# as it does use of the /proc filesystem.*

REDISPORT**=**6379

EXEC**=/**usr**/**local**/**bin**/**redis-server

CLIEXEC**=/**usr**/**local**/**bin**/**redis-cli

PIDFILE**=/**var**/**run**/**redis\_${REDISPORT}**.**pid

CONF**=**"/etc/redis/${REDISPORT}.conf"

**case** "$1" **in**

**start)**

**if** **[** **-f** **$PIDFILE** **]**

**then**

**echo** "$PIDFILE exists, process is already running or crashed"

**else**

**echo** "Starting Redis server..."

**$EXEC** **$CONF**

**fi**

**;;**

**stop)**

**if** **[** **!** **-f** **$PIDFILE** **]**

**then**

**echo** "$PIDFILE does not exist, process is not running"

**else**

PID**=$(cat $PIDFILE)**

**echo** "Stopping ..."

**$CLIEXEC** -p **$REDISPORT** shutdown

**while** **[** **-x** **/**proc**/**${PID} **]**

**do**

**echo** "Waiting for Redis to shutdown ..."

**sleep** 1

**done**

**echo** "Redis stopped"

**fi**

**;;**

**\*)**

**echo** "Please use start or stop as first argument"

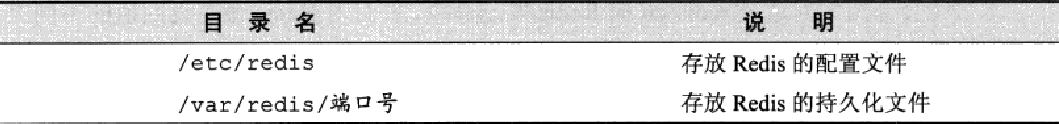
**;;**

**esac**

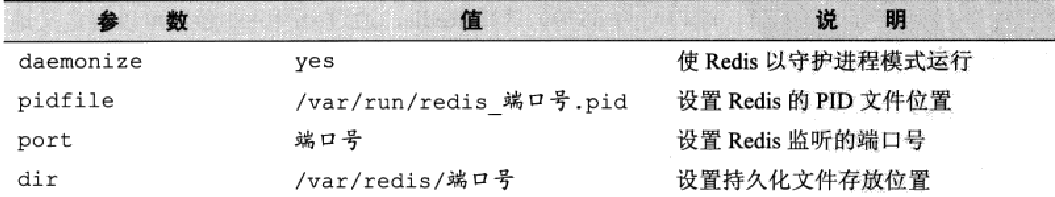
需要配置Redis的运行方式和持久化文件、日志文件的存储位置等，具体步骤如下：

1)：配置初始化脚本。首先将初始化脚本复制到/etc/init.d目录中，文件名为redis\_*port*，其中*port*表示Redis监听的端口号，客户端通过该端口连接Redis。然后修改脚本第6行的REDISPORT变量的值为同样的端口号。

2)：建立需要的文件夹。建立下表列出的目录：



3)：修改配置文件。首先将配置文件模板复制到/etc/redis目录中，以端口号命名（如6379.conf），然后按照下表对其中的部分参数进行编辑：



然后就可以使用/etc/init.d/redis\_port start命令来启动Redis了。

3：终止

考虑到Redis有可能正在将内存中的数据同步到硬盘中，强行终止Redis进程可能会导致数据丢失。正确停止Redis的方式是向Redis发送shutdown命令：

# redis-cli shutdown

当Redis收到shutdown命令后，会先断开所有客户端连接，然后根据配置执行持久化，完成后退出。

Redis可以妥善处理SIGTERM信号，所以使用“kill redis pid”也可以正常结束Redis，效果与发送shutdown命令一样。

4：发送命令

redis-cli是Redis自带的命令行客户端，通过redis-cli向Redis发送命令有两种方式：

一是将命令作为redis-cli的参数执行，比如上面用过的redis-cli shutdown。redis-cli执行时会自动按照默认配置（服务器地址为127.0.0.1，端口号为6379）连接Redis。可以通过-h和-p参数自定义地址和端口号：

# redis-cli -h 127.0.0.1 -p 6379

Redis提供了PING命令来测试客户端与Redis的连接是否正常，如果连接正常会收到回复PONG。如：

# redis-cli PING

PONG

第二种方式是不附带参数运行redis-cli，这样会进入交互模式。可以自由输入命令，例如：

# redis-cli

127.0.0.1**:**6379**>** ping

PONG

5：配置

除了可以配置端口号之外，Redis还支持其他配置选项，如是否开启持久化、日志级别等。Redis支持通过配置文件来设置这些选项。启用配置文件的方法是在启动时将配置文件的路径作为启动参数传递给redis-server，如：

# redis-server /path/to/redis.conf

通过启动参数传递同名的配置选项会覆盖配置文件中相应的参数，就像这样：

# redis-server /path/to/redis.conf --loglevel warning

Redis提供了一个配置文件的模板redis.conf。它位于源代码目录的根目录中。

除此之外，还可以在Redis运行时通过”CONFIG SET”命令，在不重启Redis的情况下动态修改部分Redis配置。就像这样：

127.0.0.1**:**6379**>** config set loglevel warning

OK

并不是所有的配置都可以使用”CONFIG SET”命令修改。在运行的时候也可以使用”CONFIG GET”命令获得Redis当前的配置情况：

127.0.0.1**:**6379**>** config get loglevel

1**)** "loglevel"

2**)** "warning"

6：多数据库

Redis是一个字典结构的存储服务器，一个Redis实例提供了多个用来存储数据的字典，客户端可以指定将数据存储在哪个字典中。这与我们熟知的在一个关系数据库实例中可以创建多个数据库类似，所以可以将其中的每个字典都理解成一个独立的数据库。

每个数据库对外都是以一个从0开始的递增数字命名，Redis默认支持16个数据库，可以通过配置参数databases来修改这一数字。客户端与Redis建立连接后会自动选择0号数据库，不过可以随时使用SELECT命令更换数据库，如要选择1号数据库：

127.0.0.1**:**6379**>** select 1

OK

然而这些以数字命名的数据库又与我们理解的数据库有所区别。首先Redis不支持自定义数据库的名字，每个数据库都以编号命名，开发者必须自己记录哪些数据库存储了哪些数据。另外Redis也不支持为每个数据库设置不同的访问密码，所以一个客户端要么可以访问全部数据库，要么连一个数据库也没有权限访问。最重要的是多个数据库之间并不是完全隔离的，比如”FLUSHALL”命令可以清空一个Redis实例中所有数据库中的数据。

因此，这些数据库更像是一种命名空间，而不适宜存储不同应用的数据。不同的应用应该使用不同的Redis实例存储数据。由于Redis非常轻量级，一个空Redis实例占用的内存只有1 MB左右，所以不用担心多个Redis实例会额外占用很多内存。