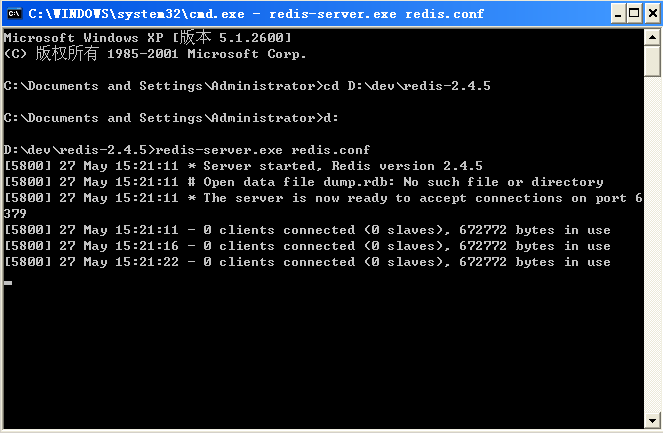
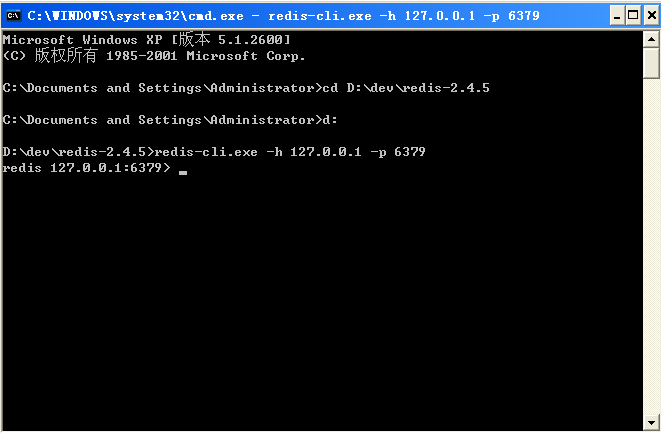
# **[windows下安装redis](http://www.cnblogs.com/linjiqin/archive/2013/05/27/3101694.html)**

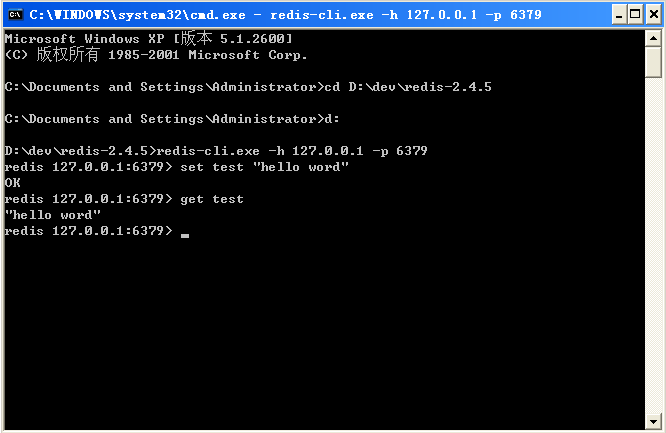
**1、redis简介**  
redis是一个key-value存储系统。和Memcached类似，它支持存储的value类型相对更多，包括string(字符串)、list(链表)、set(集合)、zset(sorted set --有序集合)和hashs（哈希类型）。这些数据类型都支持push/pop、add/remove及取交集并集和差集及更丰富的操作，而且这些操作都是原子性的。在此基础上，redis支持各种不同方式的排序。与memcached一样，为了保证效率，数据都是缓存在内存中。区别的是redis会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，并且在此基础上实现了master-slave(主从)同步。

Redis 是一个高性能的key-value数据库。 redis的出现，很大程度补偿了memcached这类key/value存储的不足，在部分场合可以对关系数据库起到很好的补充作用。它提供了Python，Ruby，Erlang，PHP客户端，使用很方便。

**2、windows下安装redis**  
下载地址[https://github.com/dmajkic/redis/downloads](https://github.com/dmajkic/redis/downloads" \t "http://www.cnblogs.com/linjiqin/archive/2013/05/27/_blank)。下载到的Redis支持32bit和64bit。根据自己实际情况选择，我选择32bit。把32bit文件内容拷贝到需要安装的目录下,比如：D:\dev\redis-2.4.5。

打开一个cmd窗口，使用cd命令切换到指定目录（D:\dev\redis-2.4.5）运行 redis-server.exe redis.conf 。运行以后出现如下界面。  
  
这就说明Redis服务端已经安装成功。

重新打开一个cmd窗口，使用cd命令切换到指定目录（D:\dev\redis-2.4.5）运行 redis-cli.exe -h 127.0.0.1 -p 6379，其中 127.0.0.1是本地ip，6379是redis服务端的默认端口。运行成功如下图所示。  
这样，Redis windows环境下搭建已经完成，是不是很简单。  
  
这样，Redis windows环境下搭建已经完成，是不是很简单。

环境已经搭建好，总得测试下吧。比如：存储一个key为test，value为hello word的字符串，然后获取key值。  
  
正确输出 hell word，测试成功！

# **Redis本地环境搭建**

## **Windows 下环境搭建**

### **1. 设置hosts**

set duapphosts=127.0.0.1 sqld.duapp.com

set redisduapphosts=127.0.0.1 redis.duapp.com

echo %duapphosts% >> C:\Windows\System32\drivers\etc\hosts

echo %redisduapphosts% >> C:\Windows\System32\drivers\etc\hosts

### **2. 下载Redis-Windows版本**

Redis官网下载页面: <http://redis.io/download>

Windows下Redis项目: <https://github.com/MSOpenTech/redis>

在releases页面找到并下载最新的ZIP包: <https://github.com/MSOpenTech/redis/releases>

### **3. 解压安装**

加压下载后的文件 redis-2.8.17.zip 到 redis-2.8.17 目录. 例如: D:\DevlopPrograms\redis-2.8.17.

如果需要简单测试一下, 鼠标双击 redis-server.exe即可,如果没错, 稍后会弹出命令行窗口显示执行状态.

如果不是 Administrator用户,则可能需要以管理员身份运行. 或者参考 [Windows 7 启用超级管理员administrator账户的N种方法](http://tieba.baidu.com/p/1262871133)

简单测试,则使用 redis-cli.exe 即可, 打开后会自动连接上本机服务器. 可以输入 info 查看服务器信息.

如果要进行基准测试,可以启动服务器后,在cmd中运行 redis-benchmark.exe 程序.

### **4. 启动与注册服务**

如果准备长期使用,则需要注册为系统服务.

进入CMD,切换目录:

D:

cd D:\DevlopPrograms\redis-2.8.17

注册服务,可以保存为 service-install.bat 文件:

redis-server.exe --service-install redis.windows.conf --loglevel verbose

redis-server --service-start

卸载服务, 可以保存为 uninstall-service.bat 文件.:

redis-server --service-stop

redis-server --service-uninstall

可以在注册服务时,通过 –service-name redisService1 参数直接指定服务名,适合安装多个实例的情况,卸载也是同样的道理.

启动redis服务器时也可以直接指定配置文件,可以保存为 startup.bat 文件:

redis-server.exe redis.windows.conf

当然,指定了配置文件以后,可能会碰到启动失败的问题.此时,请修改配置文件,指定 maxheap 参数.

### **5. 修改配置文件**

修改配置文件redis.windows.conf,如果有中文,请另存为UTF-8编码.

# 修改端口号

# port 6379

port 80

# 指定访问密码

# requirepass foobared

requirepass 6EhSiGpsmSMRyZieglUImkTr-eoNRNBgRk397mVyu66MHYuZDsepCeZ8A-MHdLBQwQQVQiHBufZbPa

# 设置最大堆内存限制,两者设置一个即可

# maxheap <bytes>

maxheap 512000000

# 设置最大内存限制, 两者设置一个即可

# maxmemory <bytes>

# maxmemory 512000000

此时,如果用客户端来访问,使用如下cmd命令,可以保存为 client.bat 文件:

redis-cli.exe -h redis.duapp.com -p 80 -a 6EhSiGpsmSMRyZieglUImkTr-eoNRNBgRk397mVyu66MHYuZDsepCeZ8A-MHdLBQwQQVQiHBufZbPa

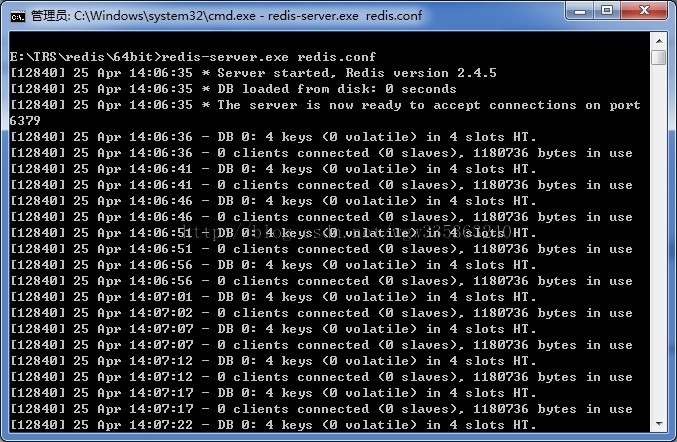
### **6. 其他附加**

管理工具: RedisStudio: <https://github.com/cinience/RedisStudio>

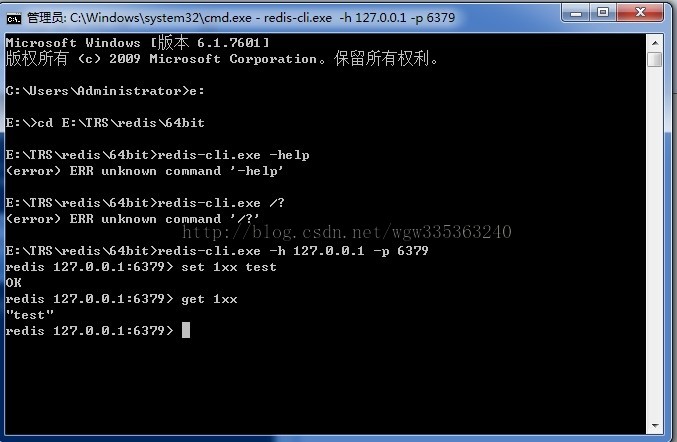
当然,目录里面也有一些word文档, 有兴趣可以读一读.

更多信息,请参考: [renfufei的专栏-Redis: http://blog.csdn.net/renfufei/article/category/2470713](http://blog.csdn.net/renfufei/article/category/2470713)

感觉这东西跟mongodb差不多，安装和布置挺简单，  
下载地址：https://github.com/dmajkic/redis/downloads 下载下来的包里有两个，  
一个是32位的，一个是64位的。根据自己的实情情况选择，我的是64bit，  
把这个文件夹复制到其它地方，比如E:\TRS\redis目录下。  
打开一个cmd窗口  使用cd命令切换目录到E:\TRS\redis 运行 redis-server.exe redis.conf   
如果想方便的话，可以把redis的路径加到系统的环境变量里，这样就省得再输路径了，后面的那个redis.conf可以省略，如果省略，会启用默认的。输入之后，会显示如下界面：



这时候别启一个cmd窗口，原来的不要关闭，不然就无法访问服务端了  
切换到redis目录下运行 redis-cli.exe -h 127.0.0.1 -p 6379 出现下图：



这时候，就已经完成配置了，现在说下它的的redis.conf配置文件。下面是相关项的说明，

**[html]** [view plain](http://blog.csdn.net/wgw335363240/article/details/24469159" \o "view plain) [copy](http://blog.csdn.net/wgw335363240/article/details/24469159" \o "copy" \t "http://blog.csdn.net/wgw335363240/article/details/_blank) [IMG_258](https://code.csdn.net/snippets/311576)[IMG_259](https://code.csdn.net/snippets/311576/fork)

1. # Redis configuration file example


5. # Note on units: when memory size is needed, it is possible to specifiy
7. # it in the usual form of 1k 5GB 4M and so forth:
9. #
11. # 1k =**>** 1000 bytes
13. # 1kb =**>** 1024 bytes
15. # 1m =**>** 1000000 bytes
17. # 1mb =**>** 1024\*1024 bytes
19. # 1g =**>** 1000000000 bytes
21. # 1gb =**>** 1024\*1024\*1024 bytes
23. #
25. # units are case insensitive so 1GB 1Gb 1gB are all the same.


29. # By default Redis does not run as a daemon. Use 'yes' if you need it.
31. # Note that Redis will write a pid file in /var/run/redis.pid when daemonized.
33. daemonize no
35. Redis默认不是以守护进程的方式运行，可以通过该配置项修改，使用yes启用守护进程


39. # When running daemonized, Redis writes a pid file in /var/run/redis.pid by
41. # default. You can specify a custom pid file location here.
43. pidfile /var/run/redis.pid
45. 当Redis以守护进程方式运行时，Redis默认会把pid写入/var/run/redis.pid文件，可以通过pidfile指定
47. # Accept connections on the specified port, default is 6379.
49. # If port 0 is specified Redis will not listen on a TCP socket.
51. port 6379
53. 指定Redis监听端口，默认端口为6379
55. # If you want you can bind a single interface, if the bind option is not
57. # specified all the interfaces will listen for incoming connections.
59. #
61. # bind 127.0.0.1
63. 绑定的主机地址
65. # Specify the path for the unix socket that will be used to listen for
67. # incoming connections. There is no default, so Redis will not listen
69. # on a unix socket when not specified.
71. #
73. # unixsocket /tmp/redis.sock
75. # unixsocketperm 755


79. # Close the connection after a client is idle for N seconds (0 to disable)
81. timeout 0
83. 当 客户端闲置多长时间后关闭连接，如果指定为0，表示关闭该功能
85. # Set server verbosity to 'debug'
87. # it can be one of:
89. # debug (a lot of information, useful for development/testing)
91. # verbose (many rarely useful info, but not a mess like the debug level)
93. # notice (moderately verbose, what you want in production probably)
95. # warning (only very important / critical messages are logged)
97. loglevel verbose
99. 指定日志记录级别，Redis总共支持四个级别：debug、verbose、notice、warning，默认为verbose
101. # Specify the log file name. Also 'stdout' can be used to force
103. # Redis to log on the standard output. Note that if you use standard
105. # output for logging but daemonize, logs will be sent to /dev/null
107. logfile stdout
109. 日志记录方式，默认为标准输出，如果配置Redis为守护进程方式运行，而这里又配置为日志记录方式为标准输出，则日志将会发送给/dev/null
111. # To enable logging to the system logger, just set 'syslog-enabled' to yes,
113. # and optionally update the other syslog parameters to suit your needs.
115. # syslog-enabled no


119. # Specify the syslog identity.
121. # syslog-ident redis


125. # Specify the syslog facility.  Must be USER or between LOCAL0-LOCAL7.
127. # syslog-facility local0


131. # Set the number of databases. The default database is DB 0, you can select
133. # a different one on a per-connection basis using SELECT **<dbid>** where
135. # dbid is a number between 0 and 'databases'-1
137. databases 16
139. 设置数据库的数量，默认数据库为0，可以使用SELECT **<dbid>**命令在连接上指定数据库id
141. ################################ SNAPSHOTTING  #################################
143. #
145. # Save the DB on disk:
147. #
149. #   save **<seconds>** **<changes>**
151. #
153. #   Will save the DB if both the given number of seconds and the given
155. #   number of write operations against the DB occurred.
157. #
159. #   In the example below the behaviour will be to save:
161. #   after 900 sec (15 min) if at least 1 key changed
163. #   after 300 sec (5 min) if at least 10 keys changed
165. #   after 60 sec if at least 10000 keys changed
167. #
169. #   Note: you can disable saving at all commenting all the "save" lines.


173. save 900 1
175. save 300 10
177. save 60 10000
179. 分别表示900秒（15分钟）内有1个更改，300秒（5分钟）内有10个更改以及60秒内有10000个更改。
181. 指定在多长时间内，有多少次更新操作，就将数据同步到数据文件，可以多个条件配合
183. # Compress string objects using LZF when dump .rdb databases?
185. # For default that's set to 'yes' as it's almost always a win.
187. # If you want to save some CPU in the saving child set it to 'no' but
189. # the dataset will likely be bigger if you have compressible values or keys.
191. rdbcompression yes
193. 指定存储至本地数据库时是否压缩数据，默认为yes，Redis采用LZF压缩，如果为了节省CPU时间，可以关闭该选项，但会导致数据库文件变的巨大
195. # The filename where to dump the DB
197. dbfilename dump.rdb
199. 指定本地数据库文件名，默认值为dump.rdb
201. # The working directory.
203. #
205. # The DB will be written inside this directory, with the filename specified
207. # above using the 'dbfilename' configuration directive.
209. #
211. # Also the Append Only File will be created inside this directory.
213. #
215. # Note that you must specify a directory here, not a file name.
217. dir ./
219. 指定本地数据库存放目录
221. ################################# REPLICATION #################################


225. # Master-Slave replication. Use slaveof to make a Redis instance a copy of
227. # another Redis server. Note that the configuration is local to the slave
229. # so for example it is possible to configure the slave to save the DB with a
231. # different interval, or to listen to another port, and so on.
233. #
235. # slaveof **<masterip>** **<masterport>**
237. slaveof **<masterip>** **<masterport>** 设置当本机为slav服务时，设置master服务的IP地址及端口，在Redis启动时，它会自动从master进行数据同步
239. # If the master is password protected (using the "requirepass" configuration
241. # directive below) it is possible to tell the slave to authenticate before
243. # starting the replication synchronization process, otherwise the master will
245. # refuse the slave request.
247. #
249. # masterauth **<master-password>**
251. masterauth **<master-password>** 当master服务设置了密码保护时，slav服务连接master的密码
253. # When a slave lost the connection with the master, or when the replication
255. # is still in progress, the slave can act in two different ways:
257. #
259. # 1) if slave-serve-stale-data is set to 'yes' (the default) the slave will
261. #    still reply to client requests, possibly with out of data data, or the
263. #    data set may just be empty if this is the first synchronization.
265. #
267. # 2) if slave-serve-stale data is set to 'no' the slave will reply with
269. #    an error "SYNC with master in progress" to all the kind of commands
271. #    but to INFO and SLAVEOF.
273. #
275. slave-serve-stale-data yes


279. # Slaves send PINGs to server in a predefined interval. It's possible to change
281. # this interval with the repl\_ping\_slave\_period option. The default value is 10
283. # seconds.
285. #
287. # repl-ping-slave-period 10


291. # The following option sets a timeout for both Bulk transfer I/O timeout and
293. # master data or ping response timeout. The default value is 60 seconds.
295. #
297. # It is important to make sure that this value is greater than the value
299. # specified for repl-ping-slave-period otherwise a timeout will be detected
301. # every time there is low traffic between the master and the slave.
303. #
305. # repl-timeout 60


309. ################################## SECURITY ###################################


313. # Require clients to issue AUTH **<PASSWORD>** before processing any other
315. # commands.  This might be useful in environments in which you do not trust
317. # others with access to the host running redis-server.
319. #
321. # This should stay commented out for backward compatibility and because most
323. # people do not need auth (e.g. they run their own servers).
325. #
327. # Warning: since Redis is pretty fast an outside user can try up to
329. # 150k passwords per second against a good box. This means that you should
331. # use a very strong password otherwise it will be very easy to break.
333. #
335. # requirepass foobared
337. requirepass foobared 设置Redis连接密码，如果配置了连接密码，客户端在连接Redis时需要通过AUTH **<password>**命令提供密码，默认关闭
339. # Command renaming.
341. #
343. # It is possilbe to change the name of dangerous commands in a shared
345. # environment. For instance the CONFIG command may be renamed into something
347. # of hard to guess so that it will be still available for internal-use
349. # tools but not available for general clients.
351. #
353. # Example:
355. #
357. # rename-command CONFIG b840fc02d524045429941cc15f59e41cb7be6c52
359. #
361. # It is also possilbe to completely kill a command renaming it into
363. # an empty string:
365. #
367. # rename-command CONFIG ""


371. ################################### LIMITS ####################################


375. # Set the max number of connected clients at the same time. By default there
377. # is no limit, and it's up to the number of file descriptors the Redis process
379. # is able to open. The special value '0' means no limits.
381. # Once the limit is reached Redis will close all the new connections sending
383. # an error 'max number of clients reached'.
385. #
387. # maxclients 128
389. maxclients 128 设置同一时间最大客户端连接数，默认无限制，Redis可以同时打开的客户端连接数为Redis进程可以打开的最大文件描述符数，如果设置 maxclients 0，表示不作限制。当客户端连接数到达限制时，Redis会关闭新的连接并向客户端返回max number of clients reached错误信息
391. # Don't use more memory than the specified amount of bytes.
393. # When the memory limit is reached Redis will try to remove keys with an
395. # EXPIRE set. It will try to start freeing keys that are going to expire
397. # in little time and preserve keys with a longer time to live.
399. # Redis will also try to remove objects from free lists if possible.
401. #
403. # If all this fails, Redis will start to reply with errors to commands
405. # that will use more memory, like SET, LPUSH, and so on, and will continue
407. # to reply to most read-only commands like GET.
409. #
411. # WARNING: maxmemory can be a good idea mainly if you want to use Redis as a
413. # 'state' server or cache, not as a real DB. When Redis is used as a real
415. # database the memory usage will grow over the weeks, it will be obvious if
417. # it is going to use too much memory in the long run, and you'll have the time
419. # to upgrade. With maxmemory after the limit is reached you'll start to get
421. # errors for write operations, and this may even lead to DB inconsistency.
423. #
425. # maxmemory **<bytes>**
427. maxmemory **<bytes>**指定Redis最大内存限制，Redis在启动时会把数据加载到内存中，达到最大内存后，Redis会先尝试清除已到期或即将到期的Key，当此方法处理 后，仍然到达最大内存设置，将无法再进行写入操作，但仍然可以进行读取操作。Redis新的vm机制，会把Key存放内存，Value会存放在swap区
429. # MAXMEMORY POLICY: how Redis will select what to remove when maxmemory
431. # is reached? You can select among five behavior:
433. #
435. # volatile-lru -**>** remove the key with an expire set using an LRU algorithm
437. # allkeys-lru -**>** remove any key accordingly to the LRU algorithm
439. # volatile-random -**>** remove a random key with an expire set
441. # allkeys-**>**random -**>** remove a random key, any key
443. # volatile-ttl -**>** remove the key with the nearest expire time (minor TTL)
445. # noeviction -**>** don't expire at all, just return an error on write operations
447. #
449. # Note: with all the kind of policies, Redis will return an error on write
451. #       operations, when there are not suitable keys for eviction.
453. #
455. #       At the date of writing this commands are: set setnx setex append
457. #       incr decr rpush lpush rpushx lpushx linsert lset rpoplpush sadd
459. #       sinter sinterstore sunion sunionstore sdiff sdiffstore zadd zincrby
461. #       zunionstore zinterstore hset hsetnx hmset hincrby incrby decrby
463. #       getset mset msetnx exec sort
465. #
467. # The default is:
469. #
471. # maxmemory-policy volatile-lru


475. # LRU and minimal TTL algorithms are not precise algorithms but approximated
477. # algorithms (in order to save memory), so you can select as well the sample
479. # size to check. For instance for default Redis will check three keys and
481. # pick the one that was used less recently, you can change the sample size
483. # using the following configuration directive.
485. #
487. # maxmemory-samples 3


491. ############################## APPEND ONLY MODE ###############################


495. # By default Redis asynchronously dumps the dataset on disk. If you can live
497. # with the idea that the latest records will be lost if something like a crash
499. # happens this is the preferred way to run Redis. If instead you care a lot
501. # about your data and don't want to that a single record can get lost you should
503. # enable the append only mode: when this mode is enabled Redis will append
505. # every write operation received in the file appendonly.aof. This file will
507. # be read on startup in order to rebuild the full dataset in memory.
509. #
511. # Note that you can have both the async dumps and the append only file if you
513. # like (you have to comment the "save" statements above to disable the dumps).
515. # Still if append only mode is enabled Redis will load the data from the
517. # log file at startup ignoring the dump.rdb file.
519. #
521. # IMPORTANT: Check the BGREWRITEAOF to check how to rewrite the append
523. # log file in background when it gets too big.


527. appendonly no
529. appendonly no指定是否在每次更新操作后进行日志记录，Redis在默认情况下是异步的把数据写入磁盘，如果不开启，可能会在断电时导致一段时间内的数据丢失。因为 redis本身同步数据文件是按上面save条件来同步的，所以有的数据会在一段时间内只存在于内存中。默认为no
531. # The name of the append only file (default: "appendonly.aof")
533. # appendfilename appendonly.aof
535. appendfilename appendonly.aof指定更新日志文件名，默认为appendonly.aof
537. # The fsync() call tells the Operating System to actually write data on disk
539. # instead to wait for more data in the output buffer. Some OS will really flush
541. # data on disk, some other OS will just try to do it ASAP.
543. #
545. # Redis supports three different modes:
547. #
549. # no: don't fsync, just let the OS flush the data when it wants. Faster.
551. # always: fsync after every write to the append only log . Slow, Safest.
553. # everysec: fsync only if one second passed since the last fsync. Compromise.
555. #
557. # The default is "everysec" that's usually the right compromise between
559. # speed and data safety. It's up to you to understand if you can relax this to
561. # "no" that will will let the operating system flush the output buffer when
563. # it wants, for better performances (but if you can live with the idea of
565. # some data loss consider the default persistence mode that's snapshotting),
567. # or on the contrary, use "always" that's very slow but a bit safer than
569. # everysec.
571. #
573. # If unsure, use "everysec".


577. # appendfsync always
579. appendfsync everysec
581. # appendfsync no
583. 指定更新日志条件，共有3个可选值：
585. no：表示等操作系统进行数据缓存同步到磁盘（快）
587. always：表示每次更新操作后手动调用fsync()将数据写到磁盘（慢，安全）
589. everysec：表示每秒同步一次（折衷，默认值）
591. # When the AOF fsync policy is set to always or everysec, and a background
593. # saving process (a background save or AOF log background rewriting) is
595. # performing a lot of I/O against the disk, in some Linux configurations
597. # Redis may block too long on the fsync() call. Note that there is no fix for
599. # this currently, as even performing fsync in a different thread will block
601. # our synchronous write(2) call.
603. #
605. # In order to mitigate this problem it's possible to use the following option
607. # that will prevent fsync() from being called in the main process while a
609. # BGSAVE or BGREWRITEAOF is in progress.
611. #
613. # This means that while another child is saving the durability of Redis is
615. # the same as "appendfsync none", that in pratical terms means that it is
617. # possible to lost up to 30 seconds of log in the worst scenario (with the
619. # default Linux settings).
621. #
623. # If you have latency problems turn this to "yes". Otherwise leave it as
625. # "no" that is the safest pick from the point of view of durability.
627. no-appendfsync-on-rewrite no


631. # Automatic rewrite of the append only file.
633. # Redis is able to automatically rewrite the log file implicitly calling
635. # BGREWRITEAOF when the AOF log size will growth by the specified percentage.
637. #
639. # This is how it works: Redis remembers the size of the AOF file after the
641. # latest rewrite (or if no rewrite happened since the restart, the size of
643. # the AOF at startup is used).
645. #
647. # This base size is compared to the current size. If the current size is
649. # bigger than the specified percentage, the rewrite is triggered. Also
651. # you need to specify a minimal size for the AOF file to be rewritten, this
653. # is useful to avoid rewriting the AOF file even if the percentage increase
655. # is reached but it is still pretty small.
657. #
659. # Specify a precentage of zero in order to disable the automatic AOF
661. # rewrite feature.


665. auto-aof-rewrite-percentage 100
667. auto-aof-rewrite-min-size 64mb


671. ################################## SLOW LOG ###################################


675. # The Redis Slow Log is a system to log queries that exceeded a specified
677. # execution time. The execution time does not include the I/O operations
679. # like talking with the client, sending the reply and so forth,
681. # but just the time needed to actually execute the command (this is the only
683. # stage of command execution where the thread is blocked and can not serve
685. # other requests in the meantime).
687. #
689. # You can configure the slow log with two parameters: one tells Redis
691. # what is the execution time, in microseconds, to exceed in order for the
693. # command to get logged, and the other parameter is the length of the
695. # slow log. When a new command is logged the oldest one is removed from the
697. # queue of logged commands.


701. # The following time is expressed in microseconds, so 1000000 is equivalent
703. # to one second. Note that a negative number disables the slow log, while
705. # a value of zero forces the logging of every command.
707. slowlog-log-slower-than 10000


711. # There is no limit to this length. Just be aware that it will consume memory.
713. # You can reclaim memory used by the slow log with SLOWLOG RESET.
715. slowlog-max-len 1024


719. ################################ VIRTUAL MEMORY ###############################


723. ### WARNING! Virtual Memory is deprecated in Redis 2.4
725. ### The use of Virtual Memory is strongly discouraged.


729. ### WARNING! Virtual Memory is deprecated in Redis 2.4
731. ### The use of Virtual Memory is strongly discouraged.


735. # Virtual Memory allows Redis to work with datasets bigger than the actual
737. # amount of RAM needed to hold the whole dataset in memory.
739. # In order to do so very used keys are taken in memory while the other keys
741. # are swapped into a swap file, similarly to what operating systems do
743. # with memory pages.
745. #
747. # To enable VM just set 'vm-enabled' to yes, and set the following three
749. # VM parameters accordingly to your needs.


753. vm-enabled no
755. 指定是否启用虚拟内存机制，默认值为no，简单的介绍一下，VM机制将数据分页存放，由Redis将访问量较少的页即冷数据swap到磁盘上，访问多的页面由磁盘自动换出到内存中（在后面的文章我会仔细分析Redis的VM机制）
757. # vm-enabled yes


761. # This is the path of the Redis swap file. As you can guess, swap files
763. # can't be shared by different Redis instances, so make sure to use a swap
765. # file for every redis process you are running. Redis will complain if the
767. # swap file is already in use.
769. #
771. # The best kind of storage for the Redis swap file (that's accessed at random)
773. # is a Solid State Disk (SSD).
775. #
777. # \*\*\* WARNING \*\*\* if you are using a shared hosting the default of putting
779. # the swap file under /tmp is not secure. Create a dir with access granted
781. # only to Redis user and configure Redis to create the swap file there.
783. vm-swap-file /tmp/redis.swap
785. 虚拟内存文件路径，默认值为/tmp/redis.swap，不可多个Redis实例共享
787. # vm-max-memory configures the VM to use at max the specified amount of
789. # RAM. Everything that deos not fit will be swapped on disk \*if\* possible, that
791. # is, if there is still enough contiguous space in the swap file.
793. #
795. # With vm-max-memory 0 the system will swap everything it can. Not a good
797. # default, just specify the max amount of RAM you can in bytes, but it's
799. # better to leave some margin. For instance specify an amount of RAM
801. # that's more or less between 60 and 80% of your free RAM.
803. vm-max-memory 0
805. 将所有大于vm-max-memory的数据存入虚拟内存,无论vm-max-memory设置多小,所有索引数据都是内存存储的(Redis的索引数据 就是keys),也就是说,当vm-max-memory设置为0的时候,其实是所有value都存在于磁盘。默认值为0
807. # Redis swap files is split into pages. An object can be saved using multiple
809. # contiguous pages, but pages can't be shared between different objects.
811. # So if your page is too big, small objects swapped out on disk will waste
813. # a lot of space. If you page is too small, there is less space in the swap
815. # file (assuming you configured the same number of total swap file pages).
817. #
819. # If you use a lot of small objects, use a page size of 64 or 32 bytes.
821. # If you use a lot of big objects, use a bigger page size.
823. # If unsure, use the default :)
825. vm-page-size 32
827. Redis swap文件分成了很多的page，一个对象可以保存在多个page上面，但一个page上不能被多个对象共享，vm-page-size是要根据存储的 数据大小来设定的，作者建议如果存储很多小对象，page大小最好设置为32或者64bytes；如果存储很大大对象，则可以使用更大的page，如果不 确定，就使用默认值
829. # Number of total memory pages in the swap file.
831. # Given that the page table (a bitmap of free/used pages) is taken in memory,
833. # every 8 pages on disk will consume 1 byte of RAM.
835. #
837. # The total swap size is vm-page-size \* vm-pages
839. #
841. # With the default of 32-bytes memory pages and 134217728 pages Redis will
843. # use a 4 GB swap file, that will use 16 MB of RAM for the page table.
845. #
847. # It's better to use the smallest acceptable value for your application,
849. # but the default is large in order to work in most conditions.
851. vm-pages 134217728
853. 设置swap文件中的page数量，由于页表（一种表示页面空闲或使用的bitmap）是在放在内存中的，，在磁盘上每8个pages将消耗1byte的内存。
855. # Max number of VM I/O threads running at the same time.
857. # This threads are used to read/write data from/to swap file, since they
859. # also encode and decode objects from disk to memory or the reverse, a bigger
861. # number of threads can help with big objects even if they can't help with
863. # I/O itself as the physical device may not be able to couple with many
865. # reads/writes operations at the same time.
867. #
869. # The special value of 0 turn off threaded I/O and enables the blocking
871. # Virtual Memory implementation.
873. vm-max-threads 4
875. 设置访问swap文件的线程数,最好不要超过机器的核数,如果设置为0,那么所有对swap文件的操作都是串行的，可能会造成比较长时间的延迟。默认值为4
877. ############################### ADVANCED CONFIG ###############################


881. # Hashes are encoded in a special way (much more memory efficient) when they
883. # have at max a given numer of elements, and the biggest element does not
885. # exceed a given threshold. You can configure this limits with the following
887. # configuration directives.
889. hash-max-zipmap-entries 512
891. hash-max-zipmap-value 64
893. 指定在超过一定的数量或者最大的元素超过某一临界值时，采用一种特殊的哈希算法
895. # Similarly to hashes, small lists are also encoded in a special way in order
897. # to save a lot of space. The special representation is only used when
899. # you are under the following limits:
901. list-max-ziplist-entries 512
903. list-max-ziplist-value 64


907. # Sets have a special encoding in just one case: when a set is composed
909. # of just strings that happens to be integers in radix 10 in the range
911. # of 64 bit signed integers.
913. # The following configuration setting sets the limit in the size of the
915. # set in order to use this special memory saving encoding.
917. set-max-intset-entries 512


921. # Similarly to hashes and lists, sorted sets are also specially encoded in
923. # order to save a lot of space. This encoding is only used when the length and
925. # elements of a sorted set are below the following limits:
927. zset-max-ziplist-entries 128
929. zset-max-ziplist-value 64


933. # Active rehashing uses 1 millisecond every 100 milliseconds of CPU time in
935. # order to help rehashing the main Redis hash table (the one mapping top-level
937. # keys to values). The hash table implementation redis uses (see dict.c)
939. # performs a lazy rehashing: the more operation you run into an hash table
941. # that is rhashing, the more rehashing "steps" are performed, so if the
943. # server is idle the rehashing is never complete and some more memory is used
945. # by the hash table.
947. #
949. # The default is to use this millisecond 10 times every second in order to
951. # active rehashing the main dictionaries, freeing memory when possible.
953. #
955. # If unsure:
957. # use "activerehashing no" if you have hard latency requirements and it is
959. # not a good thing in your environment that Redis can reply form time to time
961. # to queries with 2 milliseconds delay.
963. #
965. # use "activerehashing yes" if you don't have such hard requirements but
967. # want to free memory asap when possible.
969. activerehashing yes


973. ################################## INCLUDES ###################################


977. # Include one or more other config files here.  This is useful if you
979. # have a standard template that goes to all redis server but also need
981. # to customize a few per-server settings.  Include files can include
983. # other files, so use this wisely.
985. #
987. # include /path/to/local.conf
989. # include /path/to/other.conf