下载redis安装包：

[root@root ~]# cd /usr/local/src

[root@root src]# wget http://download.redis.io/releases/redis-2.8.3.tar.gz

[root@root src]# tar zxvf redis-2.8.3.tar.gz

[root@root src]# cd redis-2.8.3

[root@root redis-2.8.3]# make

执行完后，会在当前目录中的src目录中生成相应的执行文件，如：redis-server redis-cli等；  
我们在/usr/local/目录中创建redis位置目录和相应的数据存储目录、配置文件目录等：

|  |  |
| --- | --- |
|  | [root@root redis-2.8.3]# mkdir /usr/local/redis/{conf,run,db} -pv  [root@root redis-2.8.3]# cd /usr/local/src/redis-2.8.3/  [root@root redis-2.8.3]# cp redis.conf /usr/local/redis/conf  [root@root redis-2.8.3]# cd src  [root@root src]# cp redis-benchmark redis-check-aof redis-check-dump redis-cli redis-server mkreleasehdr.sh /usr/local/redis/ |

到此Redis安装完成了。

下面来试着启动一下，并查看相应的端口是否已经启动：

[root@root ~]# /usr/local/redis/redis-server /usr/local/redis/conf/redis.conf

此时启动所有的配置都是默认的，可以看到redis是前台运行，也可以让redis在后台运行，这里要修改一下redis的配置文件了，主要是redis.conf这个文件。

这个配置文件说明如下：  
redis.conf:

|  |  |
| --- | --- |
|  | daemonize yes  #---默认值no，该参数用于定制redis服务是否以守护模式运行。---  pidfile /usr/local/webserver/redis/run/redis.pid  #默认值/var/run/redis.pid，指定redis服务的进程号文件路径，以守护模式运行时需要配置本参数；  port 6379   #默认值6379，指定redis服务的端口  # bind 127.0.0.1  #绑定ip，默认是本机所有网络设备；  timeout 0   #客户端空闲n秒后断开连接；默认是 0 表示不断开。  loglevel notice  ###设置服务端的日志级别，有下列几种选择：      debug：记录详细信息，用于开发或调试；      verbose：提供很多有用的信息，但是又不像debug那么详尽，默认就是这一选项；      notice：适度提醒，多用于产品环境；      warning：仅显示重要的警告信息；  logfile stdout   ##指定日志的输出路径，默认值stdout，表示输出到屏幕，守护模式时则输出到/dev/null；  如果要输出日志到syslog中，可以启动syslog-enabled yes，默认该选项值为no。  # syslog-enabled no  databases 16   ###指定数据库的数量，默认为16个，默认使用的数据库是DB 0。  ----以下为快照相关的设置:------  #   save <seconds> <changes>  ##指定多长时间刷新快照至磁盘，这个选项有两个属性值，只有当两个属性值均满足时才会触发；可以设置多种级别，例如默认的参数文件中就设置了：  save 900 1：每900秒(15分钟)至少一次键值变更时被触发；  save 300 10：每300秒(5分钟)至少10次键值变更时被触发；  save 60 10000：每60秒至少10000次键值变更时被触发；  save 900 1  save 300 10  save 60 10000  rdbcompression yes  ##默认值yes，当dump数据库时使用LZF压缩字符串对象，如果CPU资源比较紧张，可以设置为no，选择不压缩；  rdbchecksum yes  # The filename where to dump the DB  数据库文件名  dbfilename dump.rdb  ##默认值dump.rdb，dump到文件系统中的文件名  dir /usr/local/webserver/redis/db  ##默认值./，即当前目录，dump出的数据文件的存储路径；  ----以下为复制相关的设置，复制默认是不启用的，因此在默认的参数文件下列表参数均被注释----  # slaveof <masterip> <masterport>  ##指定主端ip和端口，用于创建一个镜像服务  # masterauth <master-password>  ##如果master配置了密码的话，此处也需做设置；  slave-serve-stale-data yes  ##默认值yes。当slave丢失与master端的连接，或者复制仍在处理，那么slave会有下列两种表现：  当本参数值为yes时，slave为继续响应客户端请求，尽管数据已不同步甚至没有数据(出现在初次同步的情况下)；  当本参数值为no时，slave会返回"SYNC with master in progreee"的错误信息；  slave-read-only yes  ##默认从Redis是只读模式  # repl-ping-slave-period 10  ###默认值10，指定slave定期ping master的周期；  # repl-timeout 60  ##默认值60，指定超时时间。注意本参数包括批量传输数据和ping响应的时间。  ------以下为安全相关的设置------  # requirepass foobared  ###指定一个密码，客户端连接时也需要通过密码才能成功连接；  # rename-command CONFIG b840fc02d524045429941cc15f59e41cb7be6c52  ###重定义命令，例如将CONFIG命令更名为一个很复杂的名字：  # rename-command CONFIG ""  取消这个命令；  -----以下为资源限制方面的设置------  # maxclients 10000  ##指定客户端的最大并发连接数，默认是没有限制，直到redis无法创建新的进程为止，设置该参数值为0也表示不限制，如果该参数指定了值，当并发连接达到指定值时，redis会关闭所有新连接，并返回'max number of clients reached'的错误信息；  # maxmemory <bytes>  ###设置redis最大可使用内存。当达到最大内存后，redis会尝试按照设置的回收策略删除键值。如果无法删除键值，或者保留策略设置为不清除，那么redis就会向发出内存的请求返回错误信息。当把redis做为一级LRU的缓存时本参数较为有用。  # maxmemory-policy volatile-lru  ###默认值volatile-lru，指定清除策略，有下列几种方法：  volatile-lru -> remove the key with an expire set using an LRU algorithm  allkeys-lru -> remove any key accordingly to the LRU algorithm  volatile-random -> remove a random key with an expire set  allkeys->random -> remove a random key, any key  volatile-ttl -> remove the key with the nearest expire time (minor TTL)  noeviction -> don't expire at all, just return an error on write operations  # maxmemory-samples 3    ###默认值3，LRU和最小TTL策略并非严谨的策略，而是大约估算的方式，因此可以选择取样值以便检查。  -----以下为APPEND的配置----  ONLY模式的设置，默认情况下redis采用异步方式dump数据到磁盘上，极端情况下这可能会导致丢失部分数据(比如服务器突然宕机)，如果数据比较重要，不希望丢失，可以启用直写的模式，这种模式下redis会将所有接收到的写操作同步到appendonly.aof文件中，该文件会在redis服务启动时在内存中重建所有数据。注意这种模式对性能影响非常之大。  appendonly no  ##默认值no，指定是否启用直写模式；  # appendfilename appendonly.aof  ###直写模式的默认文件名appendonly.aof  appendfsync：调用fsync()方式让操作系统写数据到磁盘上，数据同步方式，有下列几种模式：      always：每次都调用，比如安全，但速度最慢；      everysec：每秒同步，这也是默认方式；      no：不调用fsync，由操作系统决定何时同步，比如快的模式；      no-appendfsync-on-rewrite：默认值no。当AOF fsync策略设置为always或everysec，后台保存进程会执行大量的I/O操作。某些linux配置下redis可能会阻塞过多的fsync()调用。      auto-aof-rewrite-percentage：默认值100      auto-aof-rewrite-min-size：默认值64mb  # appendfsync always  appendfsync everysec  # appendfsync no  -----以下为高级配置相关的设置----  hash-max-zipmap-entries：默认值512，当某个map的元素个数达到最大值，但是其中最大元素的长度没有达到设定阀值时，其HASH的编码采用一种特殊的方式(更有效利用内存)。本参数与下面的参数组合使用来设置这两项阀值。设置元素个数；  hash-max-zipmap-value：默认值64，设置map中元素的值的最大长度；这两个  list-max-ziplist-entries：默认值512，与hash类似，满足条件的list数组也会采用特殊的方式以节省空间。  list-max-ziplist-value：默认值64  set-max-intset-entries：默认值512，当set类型中的数据都是数值类型，并且set中整型元素的数量不超过指定值时，使用特殊的编码方式。  zset-max-ziplist-entries：默认值128，与hash和list类似。  zset-max-ziplist-value：默认值64  activerehashing：默认值yes，用来控制是否自动重建hash。Active rehashing每100微秒使用1微秒cpu时间排序，以重组Redis的hash表。重建是通过一种lazy方式，写入hash表的操作越多，需要执行rehashing的步骤也越多，如果服务器当前空闲，那么rehashing操作会一直执行。如果对实时性要求较高，难以接受redis时不时出现的2微秒的延迟，则可以设置activerehashing为no，否则建议设置为yes，以节省内存空间。 |

关于Redis的安装和配置到此结束。

**Redis监控**

1. 安装 (保证自己的python版本在2.7之后)

CENTOS 6.X 系列默认安装的 Python 2.6 ，目前开发中主要是使用 Python 2.7 ，这两个版本之间还是有不少差异的，程序在 Python 2.6 下经常会出问题。安装2.7版本。

wget https://www.python.org/ftp/python/2.7.8/Python-2.7.8.tgz

tar xf Python-2.7.8.tgz

cd Python-2.7.8

./configure --prefix=/usr/local

make && make install

yum install python-setuptools

yum install python-devel

easy\_install pip

pip install tornado // 不安装的话在运行redis-live.py的时候报错

pip install redis

pip install python-dateutil

pip install argparse

yum install git

cd /root

git clone https://github.com/ kumarnitin/RedisLive.git

备用 ： git clone https://github.com/nkrode/RedisLive.git

|  |
| --- |
| ###修改redis-live.conf文件  {      "RedisServers":      [          {              "server": "10.20.111.188",              "port" : 6379          }        ],        "DataStoreType" : "redis",        "RedisStatsServer":      {          "server" : "10.20.111.188",          "port" : 6380      },        "SqliteStatsStore" :      {          "path":  "to your sql lite file"      }  }  ###修改完毕  启动redis  /usr/local/redis/redis-server /usr/local/redis/conf/redis.conf  ###启动监控服务，每30秒监控一次  ./redis-monitor.py --duration=30  ###再次开启一个终端，进入/root/RedisLive/src目录，启动web服务  ./redis-live.py |

   在浏览器输入： <http://192.168.1.175:8888/index.html>   即可看到下图：