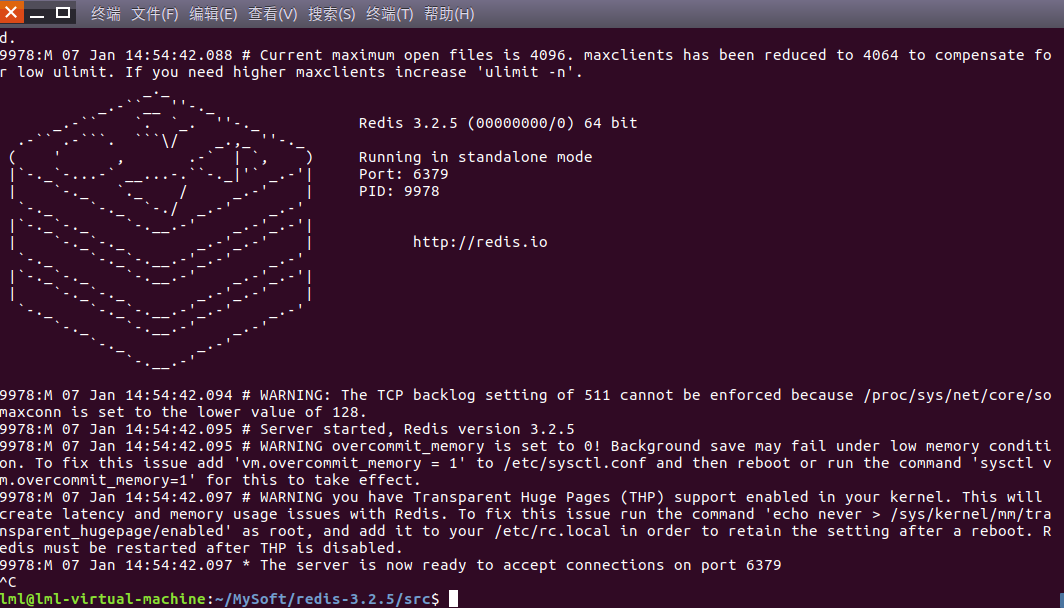
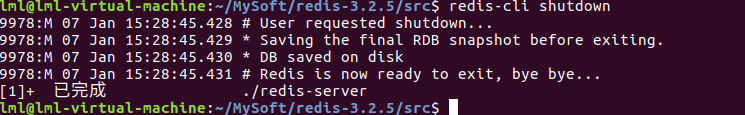
**Redis安装和操作步骤：**

1. 下载：<http://download.redis.io/releases/>
2. 安装：
3. cd到对应目录，解压：tar xvfz redis-2.8.9.tar.gz
4. 编译（sudo make）、安装（sudo make install）
5. 测试（sudo test）
6. 启动服务(./redis-server &),本次安装服务是在redis/src，是故要先cd到src目录下面，然后再执行命令,可以看到如下界面代表启动成功!



关闭服务：redis-cli shutdown，会提示命令不可用：然后可以根据提示执行：sudo apt install redis-tools，就会安装对应的命令支持工具

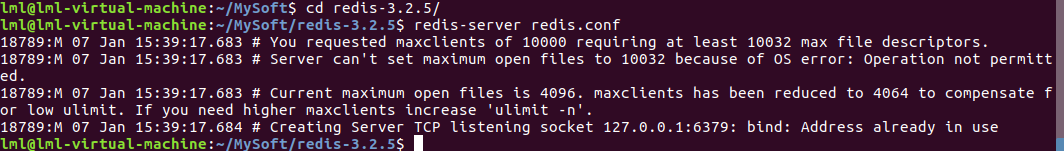
再次执行关闭服务，提示如下：



**另外，启动服务第二种方式：**

redis-server redis.conf #启动服务：

提示如下（注意：提示，如何解决这个问题？）：



**JDK安装和配置：**

1. 下载，地址如下：（本人JDK选择版本：jdk-8u111-linux-x64.tar.gz）<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

2、解压源码包  
 1）通过终端在/usr/local目录下新建java文件夹（sudo mkdir /usr/local/java）

2）将下载到压缩包拷贝到java文件夹中（cp jdk-8u25-linux-x64.tar.gz /usr/local/java）， 之后进入java目录（cd /usr/local/java）

3）解压压缩包（sudo tar xvf jdk-8u25-linux-x64.tar.gz），然后可以把压缩包删除（sudo rm jdk-8u25-linux-x64.tar.gz）

3、设置jdk环境变量，这里采用全局设置方法，它是是所有用户的共用的环境变量

1）打开对应配置文件（$sudo gedit ~/.bashrc）

2）打开之后在末尾添加如下内容：

export JAVA\_HOME=/usr/local/java/jdk1.8.0\_111

export JRE\_HOME=${JAVA\_HOME}/jre

export CLASSPATH=.:${JAVA\_HOME}/lib:${JRE\_HOME}/lib

export PATH=${JAVA\_HOME}/bin:$PATH

**注意：**在上述添加过程中，等号两侧不要加入空格，不然会出现“不是有效的标识符”，因为source /etc/profile 时不能识别多余到空格，会理解为是路径一部分。

1. 解决提示：

|  |
| --- |
| 程序 'javac' 已包含在下列软件包中：  \* default-jdk  \* ecj  \* gcj-5-jdk  \* openjdk-8-jdk-headless  \* gcj-4.8-jdk  \* gcj-4.9-jdk  \* gcj-6-jdk  \* openjdk-9-jdk-headless  请尝试：sudo apt install <选定的软件包> |

**解决方法是再终端执行以下命令：**

|  |
| --- |
| lml@lml-virtual-machine:~$ sudo update-alternatives --install /usr/bin/java java /usr/local/java/Java/jdk1.8.0\_111/bin/java 300  lml@lml-virtual-machine:~$ sudo update-alternatives --install /usr/bin/java java /usr/local/java/jdk1.8.0\_111/bin/java 300 |

**最后(java -version)，提示如下：**

|  |
| --- |
| lml@lml-virtual-machine:~$ java -version  java version "1.8.0\_111"  Java(TM) SE Runtime Environment (build 1.8.0\_111-b14)  Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode) |

**Myeclipse下载和安装：**

1. 下载：<http://www.myeclipsecn.com/bbs/?/question/25> （注册了Myeclipse中文官网：user：Hello\_lml pwd:you know it），然后本人下载版本为：MyEclipse 2015 Stable 3.0 for Linux离线安装版（文件大小：1.45G）
2. 移动到目录MySoft下，可以手动移动，也可以终端进行cp移动
3. 修改myeclipse权限（sudo chmod 777 myeclipse-2015-stable-3.0-offline-installer-linux.run）
4. 在当前路径安装（./myeclipse-2015-stable-3.0-offline-installer-linux.run ）
5. 下一步的时候指定安装目录：/home/lml/MySoft/Myeclipse2015，最后完成安装。

**Jedis下载和使用**

1. 下载（本人下载使用的是最近出的2.8.2版本的jar包文件，当然还可以下载其源码source.jar和javadoc.jar）：<http://search.maven.org/#search%7Cgav%7C1%7Cg%3A%22redis.clients%22%20AND%20a%3A%22jedis%22>

**补充：**javadoc.jar文件使用说明（转载自：http://blog.csdn.net/zljjava/article/details/7567632）：

|  |
| --- |
| 在使用**[Java](http://lib.csdn.net/base/javase" \o "Java SE知识库" \t "/home/lml/文档\\x/_blank)**语言开发的过程中，开发人员经常需要用到一些开源的工具包。在使用别人的jar包的时候，为了保证正常调用包内的方法需要把jar包添加到Referenced Libraries中。由于jar中通常只打包了类的二进制文件，其JavaDoc文件仍需要以浏览器或chm帮助文档的方式查看。Eclipse开发环境中提供了将JavaDoc文档附加到jar包中的功能，使开发者不再需要人工查找类或方法的JavaDoc注释，省时又省力。        Eclipse支持两种方式的JavaDoc导入：一种是在线的URL地址指向的JavaDoc页面，另一种是保存在本地的JavaDoc文档的jar包或zip包。给jar包附加JavaDoc的步骤如下：        1. 在Eclipse的Package Explorer视图下Referenced Libraries下找到想导入JavaDoc的jar文件，右键->选择properties进入属性配置菜单。        2. 在属性菜单中点击JavaDoc Location选项卡进入JavaDoc地址配置功能。        3. 根据能获得的JavaDoc文档的具体情况，选择JavaDoc URL或JavaDoc in Archive中的任一个分别设置相应的JavaDoc地址。如果选择以JavaDoc URL方式导入，则在javadoc location path中填入JavaDoc的URL地址，如http://hc.apache.org/httpcomponents-client-4.0.1/httpclient/apidocs/(注意，不要在路径中指定index.html）;如果选择以JavaDoc in Archive的方式导入，则在javadoc location path中填入本地JavaDoc文件的路径（注意，在打包javadoc文件里要把index.html文件放在文件夹的顶层目录下。        在导入了javadoc后，编写程序的过程中使用jar包中的方法时Eclipse会自动显示其注释说明，可以大大提高开发人员的效率。 |

**Ubuntu中Git安装和配置**

1. 安装

|  |
| --- |
| lml@lml-virtual-machine:~$ sudo apt-get update  lml@lml-virtual-machine:~$ sudo apt-get install git |

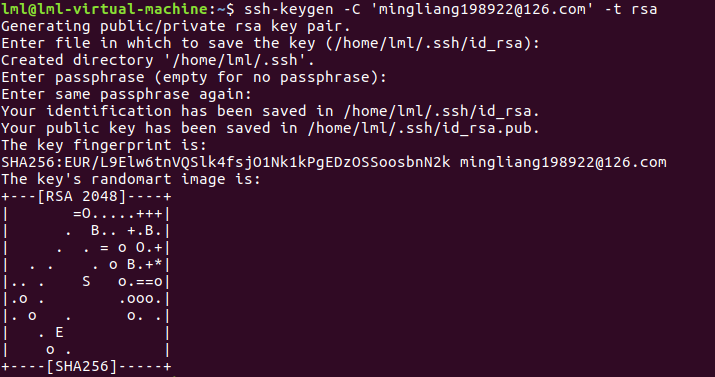
1. 配置

|  |
| --- |
| lml@lml-virtual-machine:~$ git config --global user.name "lml"  lml@lml-virtual-machine:~$ git config --global user.email ["mingXXXXXX@126.com"](mailto:\"mingliang198922@126.com\")  lml@lml-virtual-machine:~$ git config --list #查看配置 |

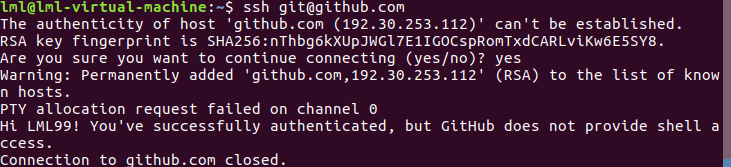
1. 链接github

|  |
| --- |
| **3.1生成 SSH Keys** 以Ubuntu 为例，需要用到 ssh-keygen 命令：  ssh-keygen -t rsa -C "mingXXXX@126.com" -f ~/.ssh/lml-github #lml-github为生成文件的别名 **简单介绍下参数含义：** -t 指定密钥类型，默认即 rsa ，可以省略 -C 设置注释文字，比如你的邮箱 -f 指定密钥文件存储文件名，会生成 lml-github 和 lml-github.pub 两个密钥文件  回车后，遇到提示输入 yes 即可，剩下一路回车，密钥文件就在指定路径下生成了。 |
| **3.2**将 **SSH 公钥添加到 Github**  登录 Github 帐号，找到帐号设置 -> SSH Keys 点击 Add New SSH Key 将本地生成的公钥文件（lml-github.pub）中的文字全选复制到 key 栏，点击 add key 保存。 |
| **3.3本地添加 SSH 别名** 如果本机有其它密钥，连接 github 时可能不会自动使用刚生成的密钥，需要设置别名：  $ sudo vi ~/.ssh/config  然后，加入以下的一段代码：  host lml-github  user git  hostname github.com  port 22  identityfile ~/.ssh/lml-github  保存退出。 |
| **3.4测试连接** $ ssh -T lml-github Hi csser! You've successfully authenticated, but GitHub does not provide shell access.  表示设置的 SSH Keys 认证通过，但 Github 不提供 shell 访问。 此时就可以正常使用 Github 了。 |

**备注：**本人使用的是默认生成的公钥匙文件名的方式，具体参考如下图：



本人测试链接还可以如下：



1. 开始使用github

|  |
| --- |
| 4.1获取源码  lml@lml-virtual-machine:~$ git clone [git@github.com:LML99/FirstRepository.git](mailto:git@github.com:LML99/FirstRepository.git) |
| 4.2初始化本地仓储  lml@lml-virtual-machine:~$ git init #之后会在本地当前目录下发现多了一个.git目录 |
| 4.3  增加远程仓库：  lml@lml-virtual-machine:~/MyGitRep$ git remote add remoteRep <https://github.com/LML99/FirstRepository>  查看远程仓库：  lml@lml-virtual-machine:~/MyGitRep$ git remote -v  本地当前分支和远程对应分支建立对应关系：  lml@lml-virtual-machine:~/MyGitRep$ git branch --set-upstream-to remoteRep/master  查看对应关系：  lml@lml-virtual-machine:~/MyGitRep$ git branch -vv  解决push的问题：  lml@lml-virtual-machine:~/MyGitRep$ git push remoteRep +master |
|  |