Centos6编译jdk8

- 环境说明:
  - centos6.8
  - java version "1.7.0_99" 安装源码需要
  - https://gitee.com/gorden5566/jdk8u 国内jdk参考镜像源
  - https://www.linuxidc.com/Linux/2017-06/144713.htm 安装参考文献
- 编译环境依赖安装:

```
* yum groupinstall "Development Tools"
* yum install libXtst-devel libXt-devel libXrender-devel
* yum install cups-devel
* yum install freetype-devel
* yum install alsa-lib-devel
```

安装过程:

- git clone https://gitee.com/gorden5566/jdk8u.git

```
[root@test01 jdk8u]# ls -lrt /usr/bin/java
lrwxrwxrwx. 1 root root 22 Sep 30 16:05 /usr/bin/java -> /etc/alternatives/java
[root@test01 jdk8u]# ls -lrt /etc/alternatives/java
lrwxrwxrwx. 1 root root 46 Sep 30 16:05 /etc/alternatives/java -> /usr/lib/jvm/jre-1.7.0-openjdk.x86_64/bin/java
[root@test01 jdk8u]# ls /usr/lib/jvm/jre-1.7.0-openjdk.x86_64/
bin  lib
```

下面进入到jdk8u的目录下，然后就可以编译了，这里后面可以带参数，下面参数是编译为64位
这里可以使用命令 sh ./configure --help 查看其他参数的作用

```
 [root@test01 jdk8u]# sh ./configure --with-target-bits=64
```

这里编译的时候容易遇到这个问题，提示该参数的报错

```
configure: error: The path given by --with-boot-jdk does not contain a valid Boot JDK
```

如果看到下面的提示就证明成功了，下面提示可以考虑安装一下ccache，另外安装后的都在build目录下

```
configure: creating /root/jdk8u/build/linux-x86_64-normal-server-release/config.status
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/spec.gmk
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/hotspot-spec.gmk
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/bootcycle-spec.gmk
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/compare.sh
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/spec.sh
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/Makefile
config.status: creating /root/jdk8u/build/linux-x86_64-normal-server-release/config.h

====================================================
A new configuration has been successfully created in
/root/jdk8u/build/linux-x86_64-normal-server-release
using configure arguments '--with-target-bits=64'.

Configuration summary:
* Debug level:    release
* JDK variant:    normal
* JVM variants:   server
* OpenJDK target: OS: linux, CPU architecture: x86, address length: 64

Tools summary:
* Boot JDK:       java version "1.8.0_121" Java(TM) SE Runtime Environment (build 1.8.0_121-b13) Java HotSpot(TM) 64-B
* C Compiler:     gcc (GCC) 4.4.7 20120313 (Red Hat-23) version 4.4.7 (at /usr/bin/gcc)
* C++ Compiler:   g++ (GCC) 4.4.7 20120313 (Red Hat-23) version 4.4.7 (at /usr/bin/g++)

Build performance summary:
* Cores to use:   1
* Memory limit:   9891 MB
* ccache status:  not installed (consider installing)

Build performance tip: ccache gives a tremendous speedup for C++ recompilations.
You do not have ccache installed. Try installing it.
You might be able to fix this by running 'sudo yum install ccache'.

WARNING: The result of this configuration has overridden an older
configuration. You *should* run 'make clean' to make sure you get a
proper build. Failure to do so might result in strange build problems.
```

然后下面就开始make，这里不带参数的话就默认为make all，但是下面提示错误:

```
[root@test01 jdk8u]# make
Building OpenJDK for target 'default' in configuration 'linux-x86_64-normal-server-release'

## Starting langtools
/bin/sh: line 0: cd: /root/jdk8u/langtools/make: No such file or directory
make: *** [langtools-only] Error 1
```

这里出现这个原因，查看了一下上面的提示，发现上面有警告，这里说找不到langtools这个目录，然后查看了一下发现，这里还需要先执行一下getModules.sh,这里就有该目录，但是因为这里github执行会很慢，另外需要先登录，并且把ssh-key的公钥复制到github.com的账号上

```
ssh-keygen -t rsa -b 4096 -C "xxx@xx.com"
```

注意这里-C后面跟的是登陆github的邮箱，然后这里复制公钥到账号上， 注意这里要打开id_rsa.pub复制，也就是vim 打开，也不能用cat查看复制，这里关于复制的过程就不提醒了，如果不会的可以百度

```
 ssh -T git@github.com
 sh ./getModules.sh
```

这里发现执行执行github.com的时候，因为服务器在国外，因此这里会很慢，这里的解决方法是修改一下hosts文件，刷新一下CDN,具体方法看下面的参考文献

- https://blog.csdn.net/github_34965845/article/details/80610060 git clone太慢的解决文献

  或者这里可以直接修改hosts文件，添加以下两行内容

  ```
  192.30.253.113 github.com
  151.101.185.194 github.global.ssl.fastly.net
  ```

  然后这里重启网络进行刷新一下

- 这里发现下载很慢，而且正确的下载地址应该是下面这个

- https://github.com/JetBrains/jdk8u

  这里重新从github上下载包，然后到本地安装重复上面的过程，除了ssh部分不用重复,另外这里为了方便，直接到github上下载getModules.sh里面所需的包，这里的地址和上面的连接地址类似

```
bash ./configure --with-target-bits=64 --with-boot-jdk=/usr/local/jdk1.7.0_80/
 --with-debug-level=slowdebug --enable-debug-symbols ZIP_DEBUGINFO_FILES=0
```

重新执行命令，注意这里--with-boot-jdk的写法，最后能看到下面的内容就可以了

```
----- Build times -------
Start 2018-10-04 13:21:14
End   2018-10-04 13:29:17
00:00:32 corba
00:03:23 hotspot
00:00:19 jaxp
00:00:25 jaxws
00:02:49 jdk
00:00:35 langtools
00:08:03 TOTAL
-------------------------
Finished building OpenJDK for target 'default'
```

接着用命令查看一下新生成的版本，注意这里编译后的文件放在./build下

```
 [root@localhost jdk8u]# ./build/linux-x86_64-normal-server-slowdebug/jdk/bin/java -version
openjdk version "1.8.0-internal-debug"
OpenJDK Runtime Environment (build 1.8.0-internal-debug-root_2018_10_04_13_20-b00)
OpenJDK 64-Bit Server VM (build 25.71-b00-debug, mixed mode)
```

接着下面就是新建一个java文件来测试一下

```
[root@localhost ~]# /root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/bin/javac test.java
[root@localhost ~]# /root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/bin/java test
hello world !
[root@localhost ~]# cat test.java
public class test{
    public static void main(String[] args){
        System.out.println("hello world !");
    }
```

```
}
```

下面就尝试使用gdb调试一下，具体执行过程见如下所示:

```
 [root@localhost ~]# gdb --args /root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/bin/java test
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-110.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/bin/java...done.
(gdb) break init.cpp:95
No source file named init.cpp.
Make breakpoint pending on future shared library load? (y or [n]) y

Breakpoint 1 (init.cpp:95) pending.
(gdb) run
Starting program: /root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/bin/java test
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Missing separate debuginfo for /root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/lib/amd64/server/libjvm.so
Try: yum --enablerepo='*debug*' install /usr/lib/debug/.build-id/18/a7103ea83b10e069e591f4cb76d896aba25b26.debug
[New Thread 0x7ffff7fdf700 (LWP 1846)]

Program received signal SIGSEGV, Segmentation fault.
[Switching to Thread 0x7ffff7fdf700 (LWP 1846)]
0x00007fffe10002b4 in ?? ()
Missing separate debuginfos, use: debuginfo-install glibc-2.17-222.el7.x86_64 libgcc-4.8.5-28.el7.x86_64 libstdc++-4.8.5-28.el7.x86_64
(gdb) l
88
89          __initenv = _environ;
90
91      #else /* JAVAW */
92      int
93      main(int argc, char **argv)
94      {
95          int margc;
96          char** margv;
97          const jboolean const_javaw = JNI_FALSE;
(gdb) quit
A debugging session is active.

        Inferior 1 [process 1842] will be killed.

Quit anyway? (y or n) y
```

Eclipse调试HotSpot虚拟机源码

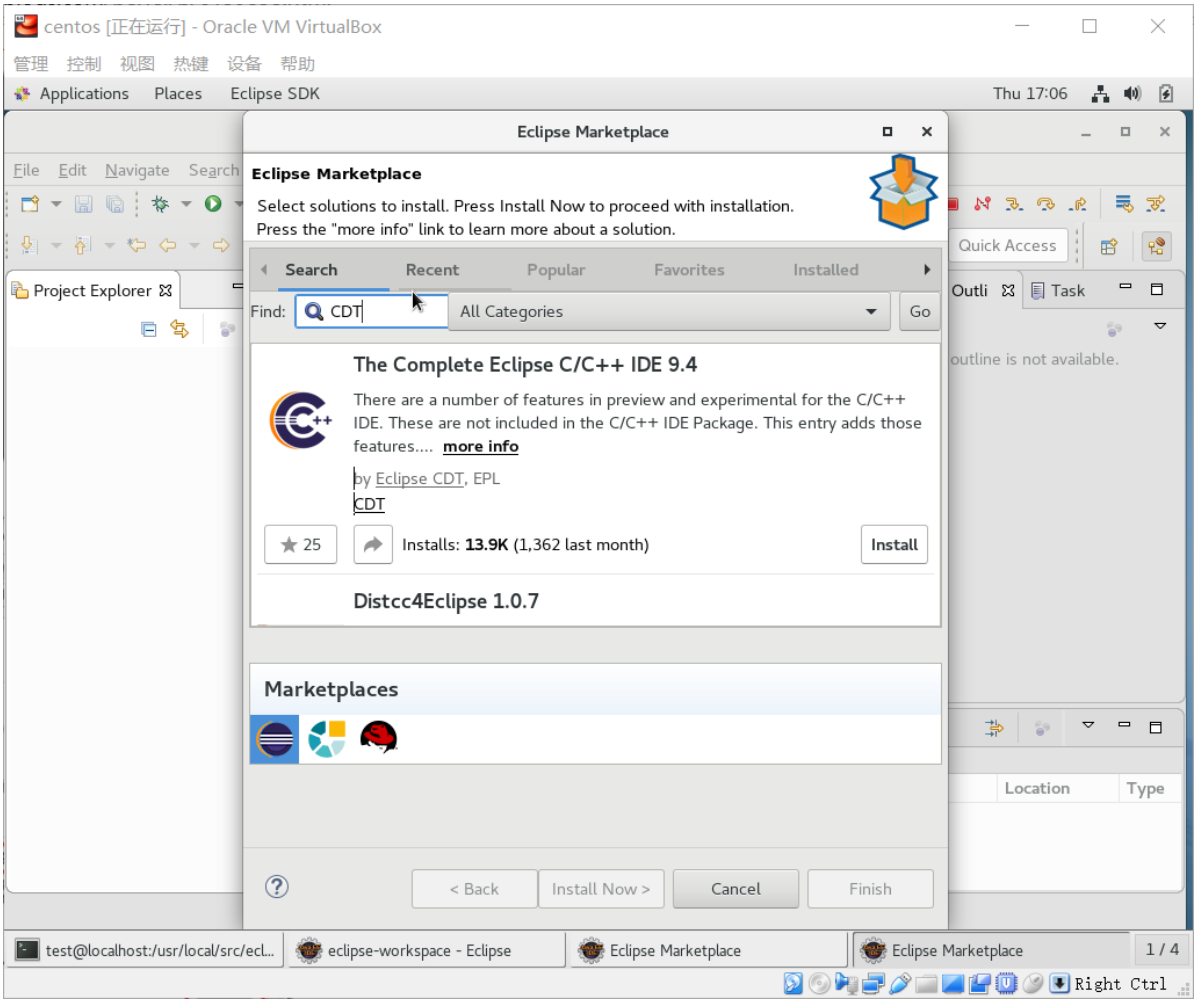
注意这里使用的eclipse版本是eclipse-jee-oxygen-R-linux-gtk-x86_64，该版本需要的jdk是1.8，前面使用编译安装的是1.7，因此这里需要切换一下，使用命令:

```
[root@localhost ~]# alternatives --config java

There are 2 programs which provide 'java'.

  Selection    Command
-----------------------------------------------
 + 1           java-1.7.0-openjdk.x86_64 (/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.                                    171-2.
*  2           java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.                                    161-2.

Enter to keep the current selection[+], or type selection number: 2
[root@localhost ~]# java -version
openjdk version "1.8.0_161"
OpenJDK Runtime Environment (build 1.8.0_161-b14)
OpenJDK 64-Bit Server VM (build 25.161-b14, mixed mode)
```

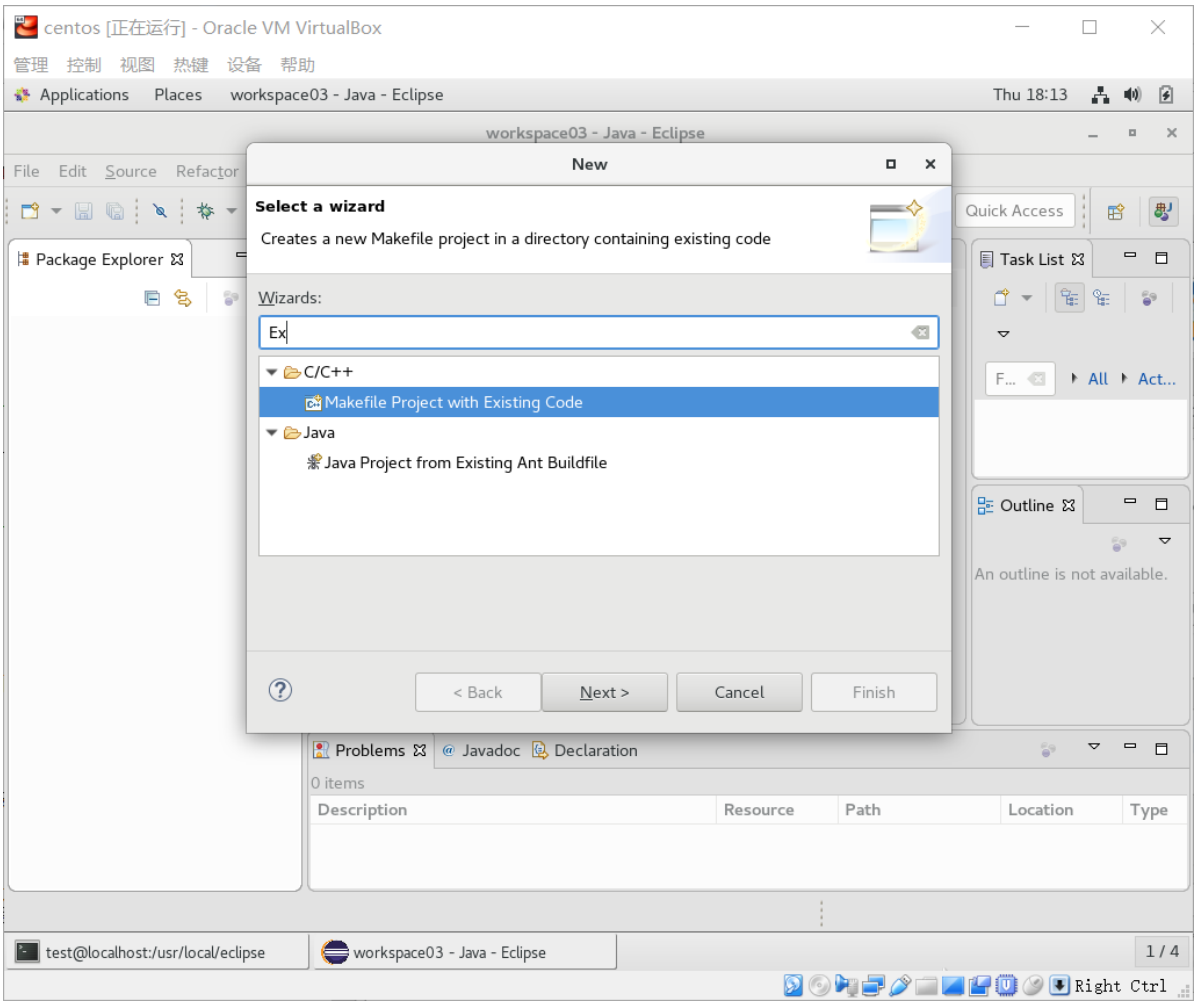
- https://www.linuxidc.com/Linux/2017-06/144713p2.htm 参考文献
- https://netbeans.org/downloads/ NetBeans工具下载地址

执行过程

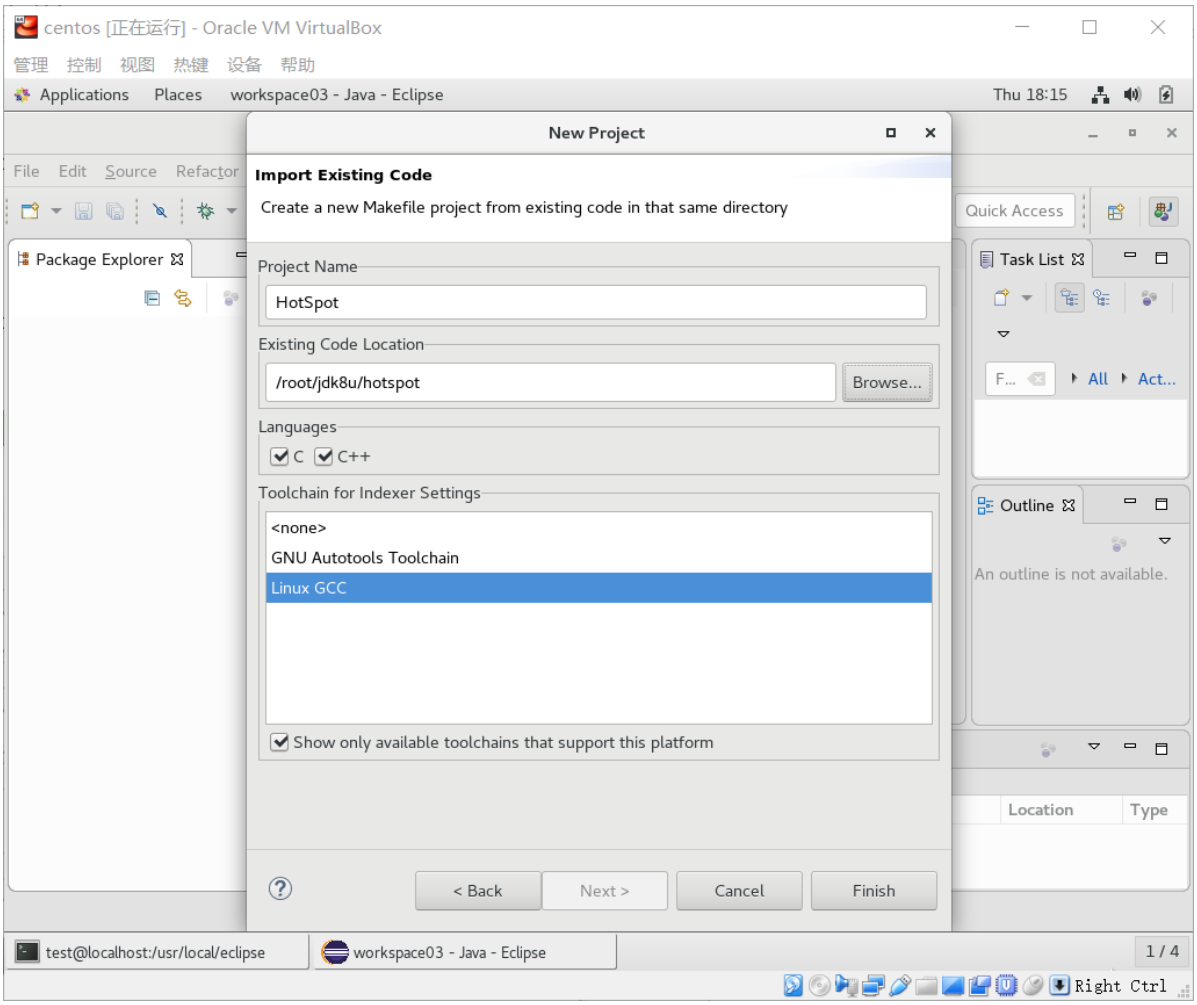
首先安装一下c++插件，这里help->Eclipse Maketplace，搜索CDT,安装第一个



这里发现该版本的eclipse安装这个之后是不行的，因此改换其他版本，这里使用参考文献上的eclipse-java-neon-1a-linux-gtk-x86_64版本，然后这里还是搜索CDT，接着安装上图的内容，这样就不会报错提示版本不适合了



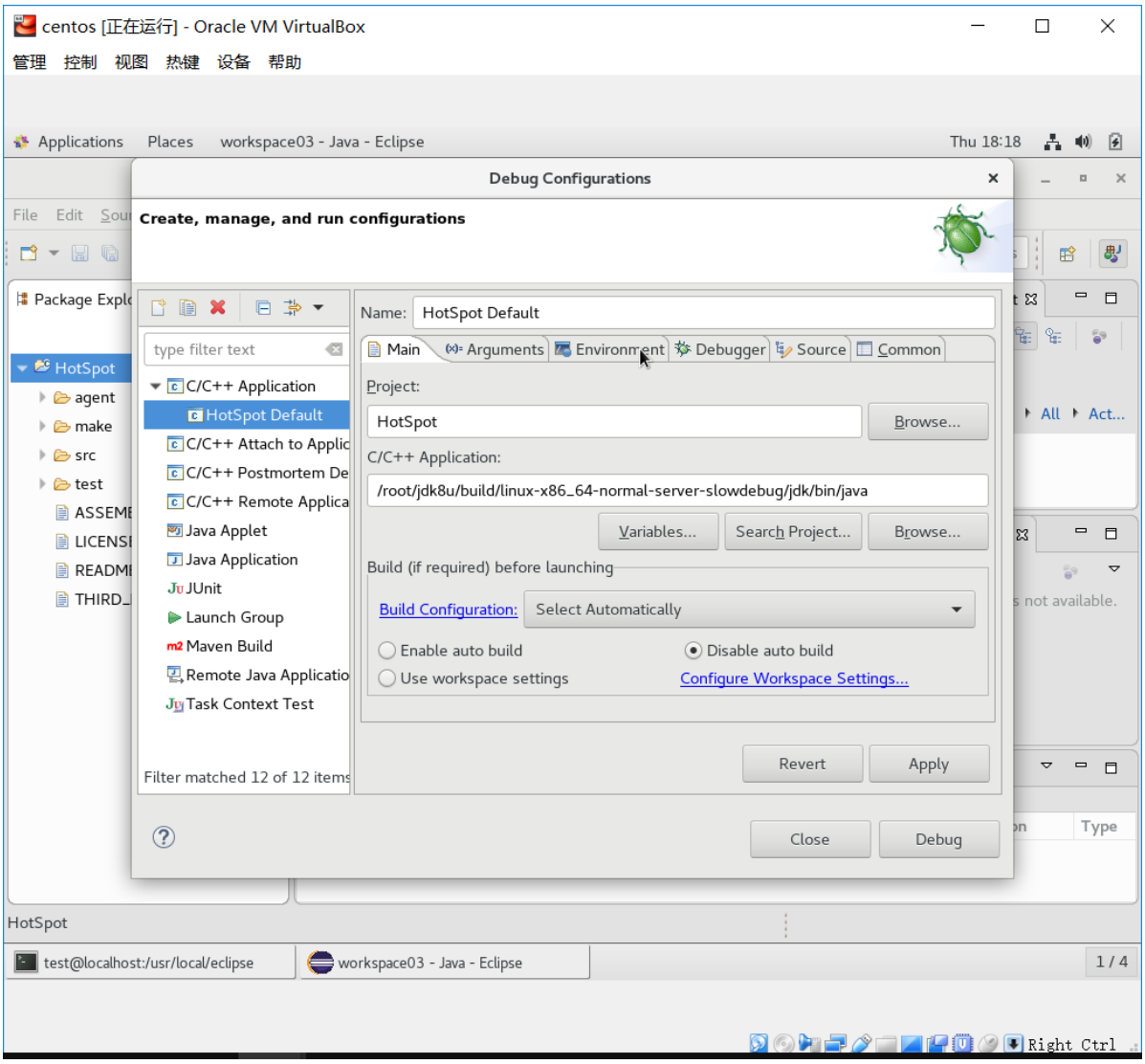
然后这里选择上面的工程进行导入，路径是File->new->other，接着下面就是配置一下项目，注意这里下面选择Linux GCC



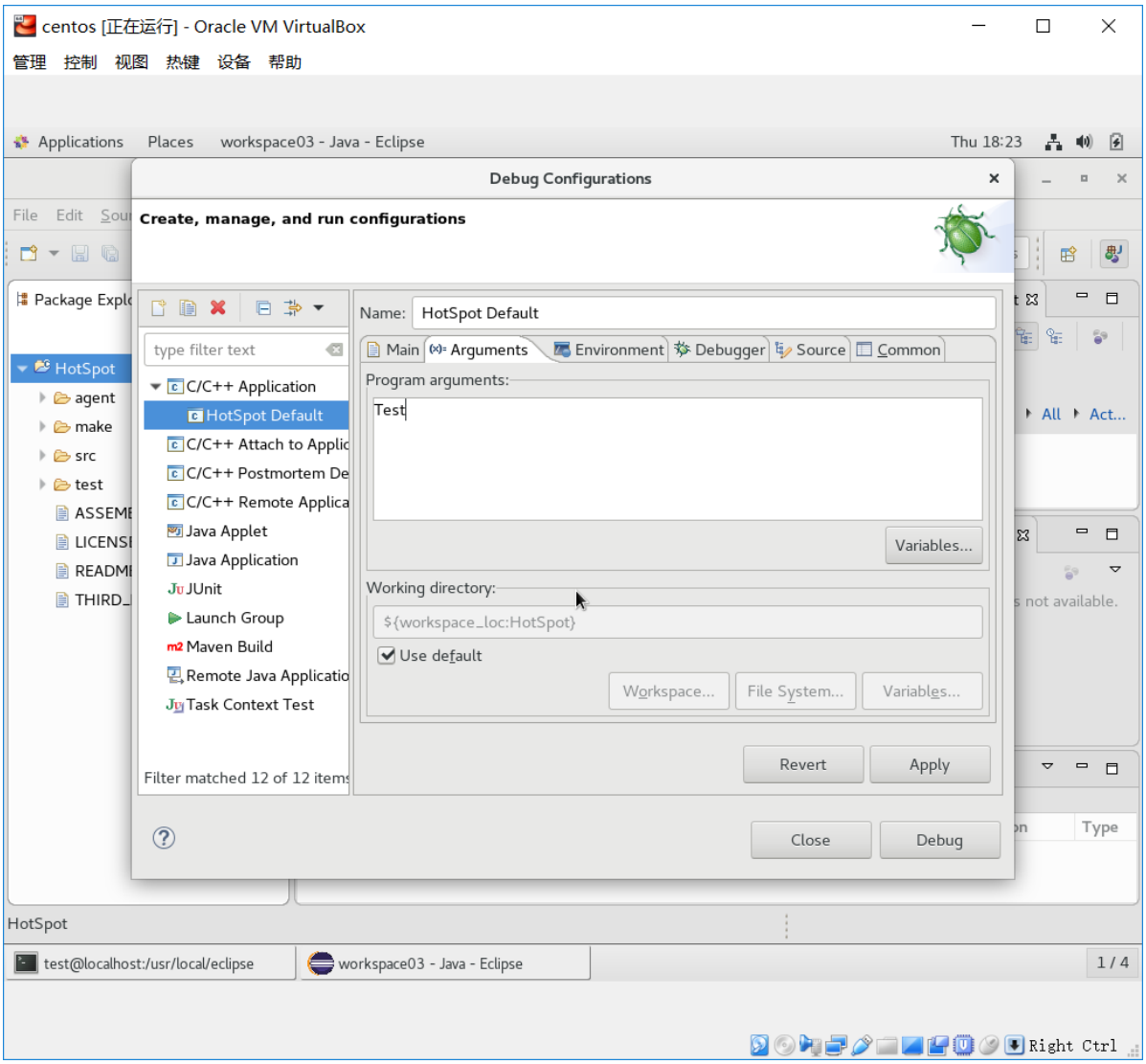
右键工程 -> Debug As -> Debug Configurations -> 右键左边的C/C++ Application -> New -> 进入Main选项卡；
在选项卡中:

```
Project：hotspot（选择导入的hotspot工程）
C/C++ Application：/root/openjdk/build/linux-x86_64-normal-server-slowdebug/jdk/bin/java（编译生成的openjdk虚拟机入口）
Disable auto build：
```

因为不再在eclipse里面编译hotspot源码,所以这里选上



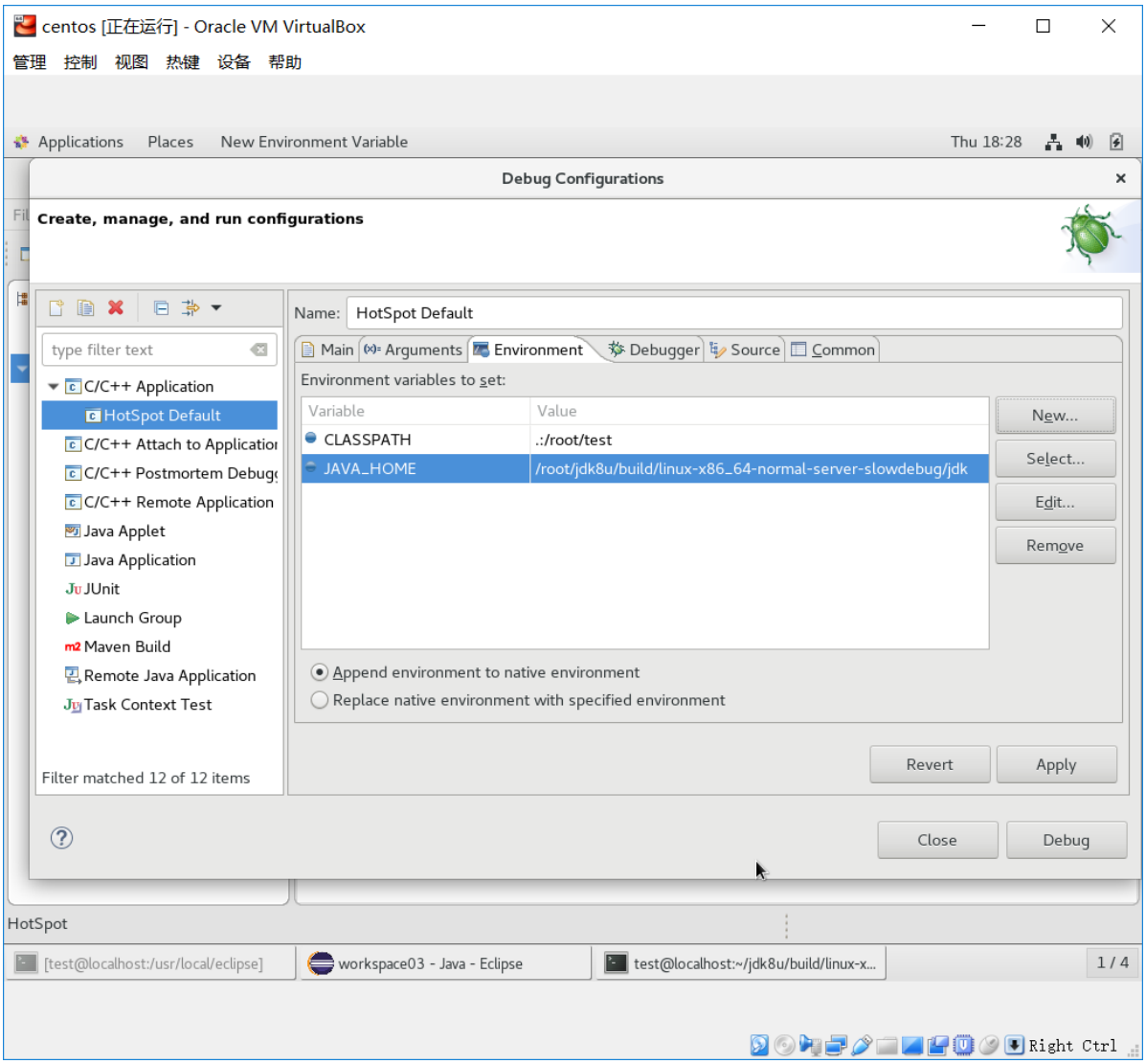
然后切换到Arguments选项卡, 输入Java的参数, 我们这里运行上面的Main类, 于是这里填上 "Test"也就是我们要执行的Java程序



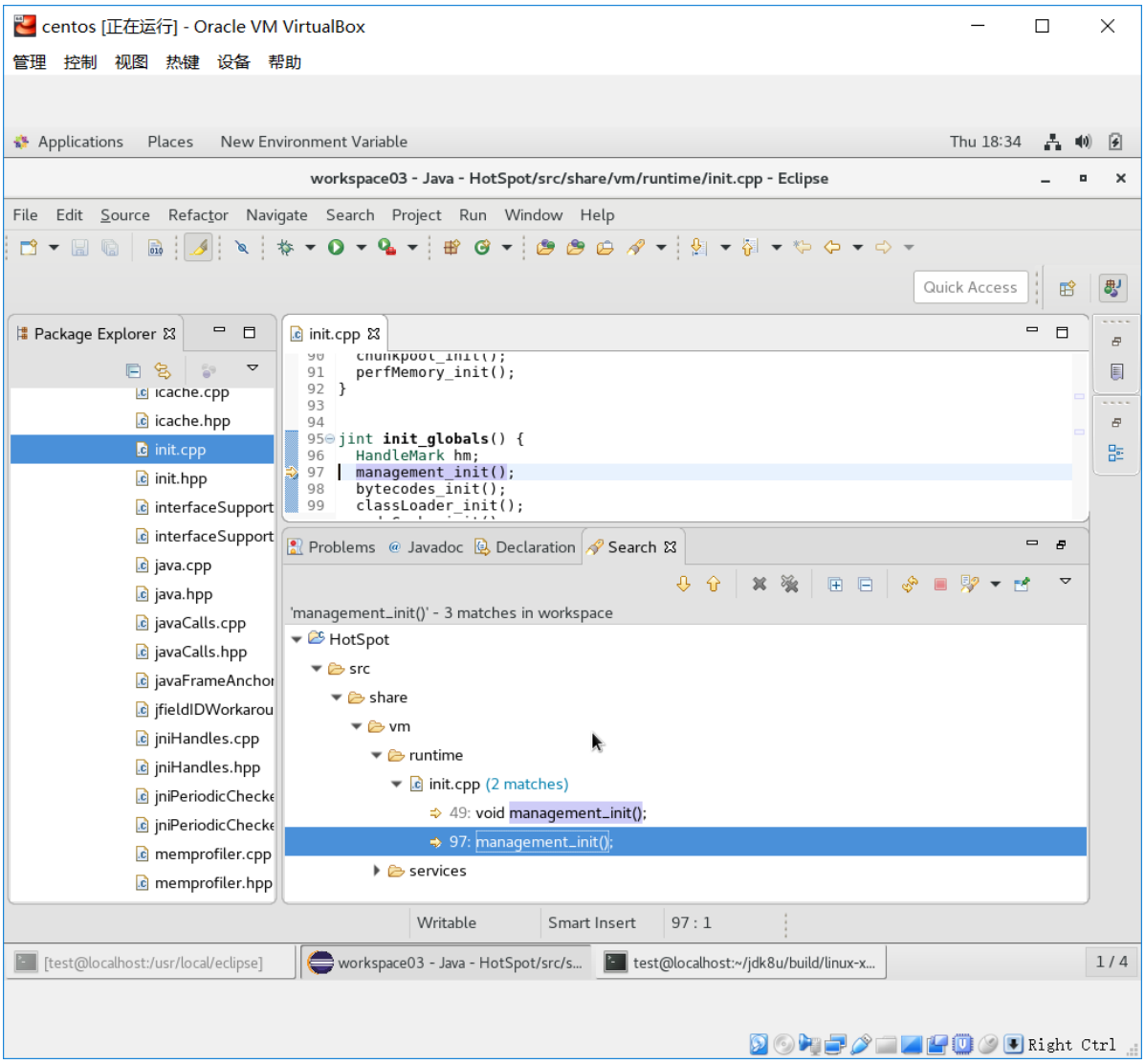
然后切换到Environment选项卡, 添加变量：

```
JAVA_HOME=/root/jdk8u/build/linux-x86_64-normal-server-slowdebug/jdk/
  （编译生成JDK所在目录）
CLASSPATH=.:/root/test （Test类文件所在目录）
```

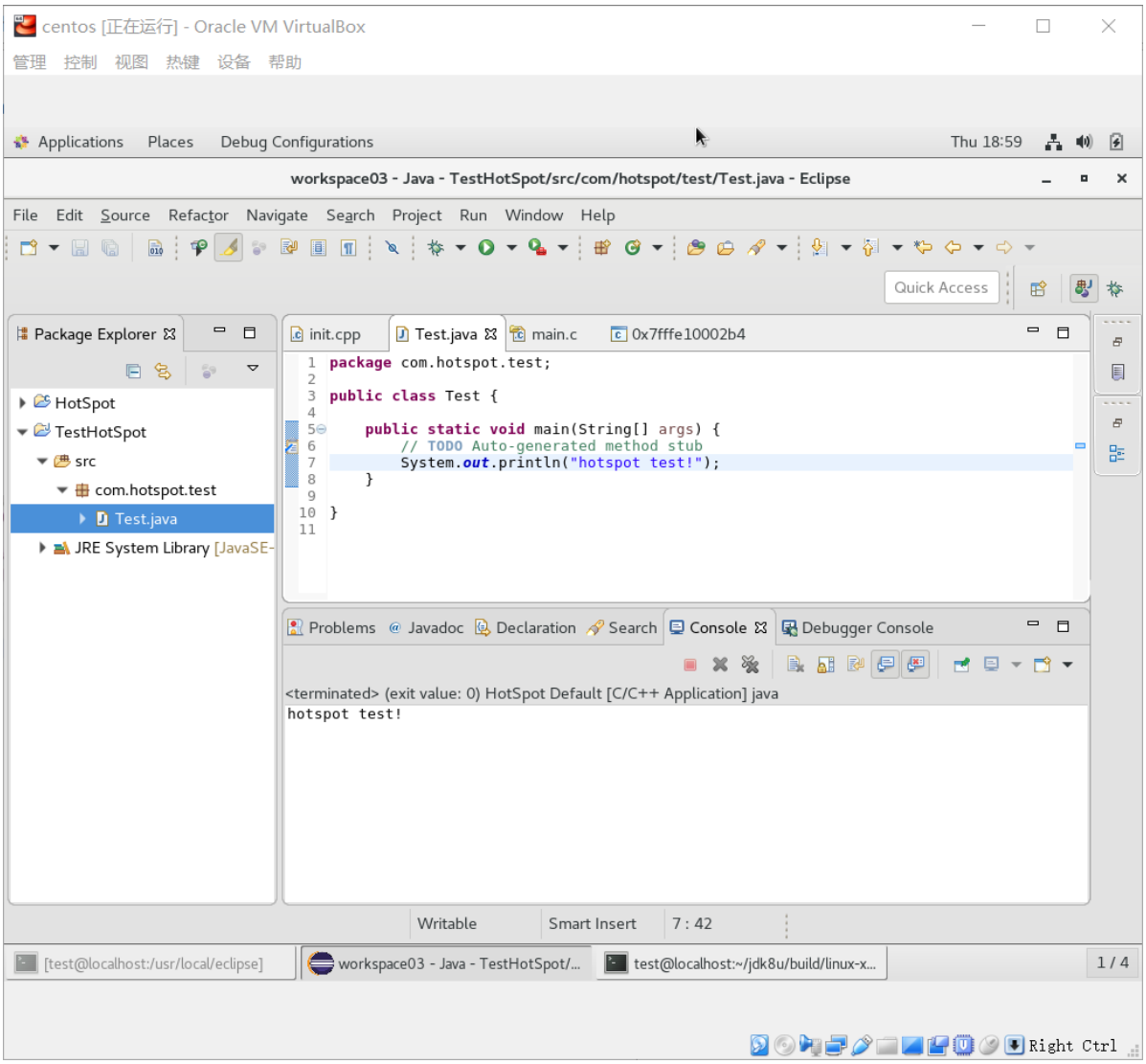
然后这里点击下面的apply保存



然后下面就是点击Search->Search->File Search，这里File Search中的Containing Text内容输入management_init()，然后搜索找到init.cpp，这里其实路径见下图所示: 加上断点

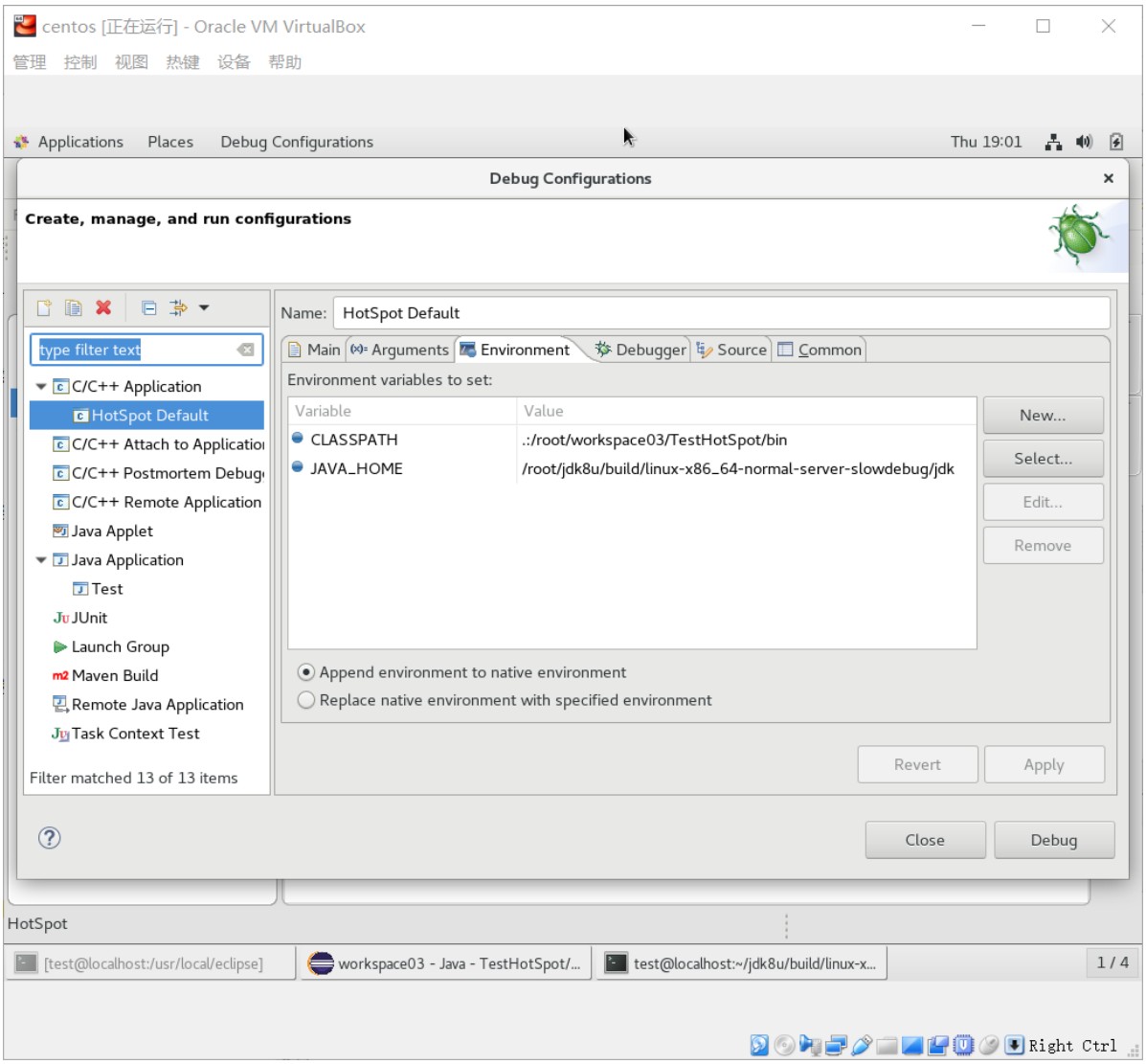


新建java工程，在工程新建包，在包中新建Test.java文件，测试程序如下：

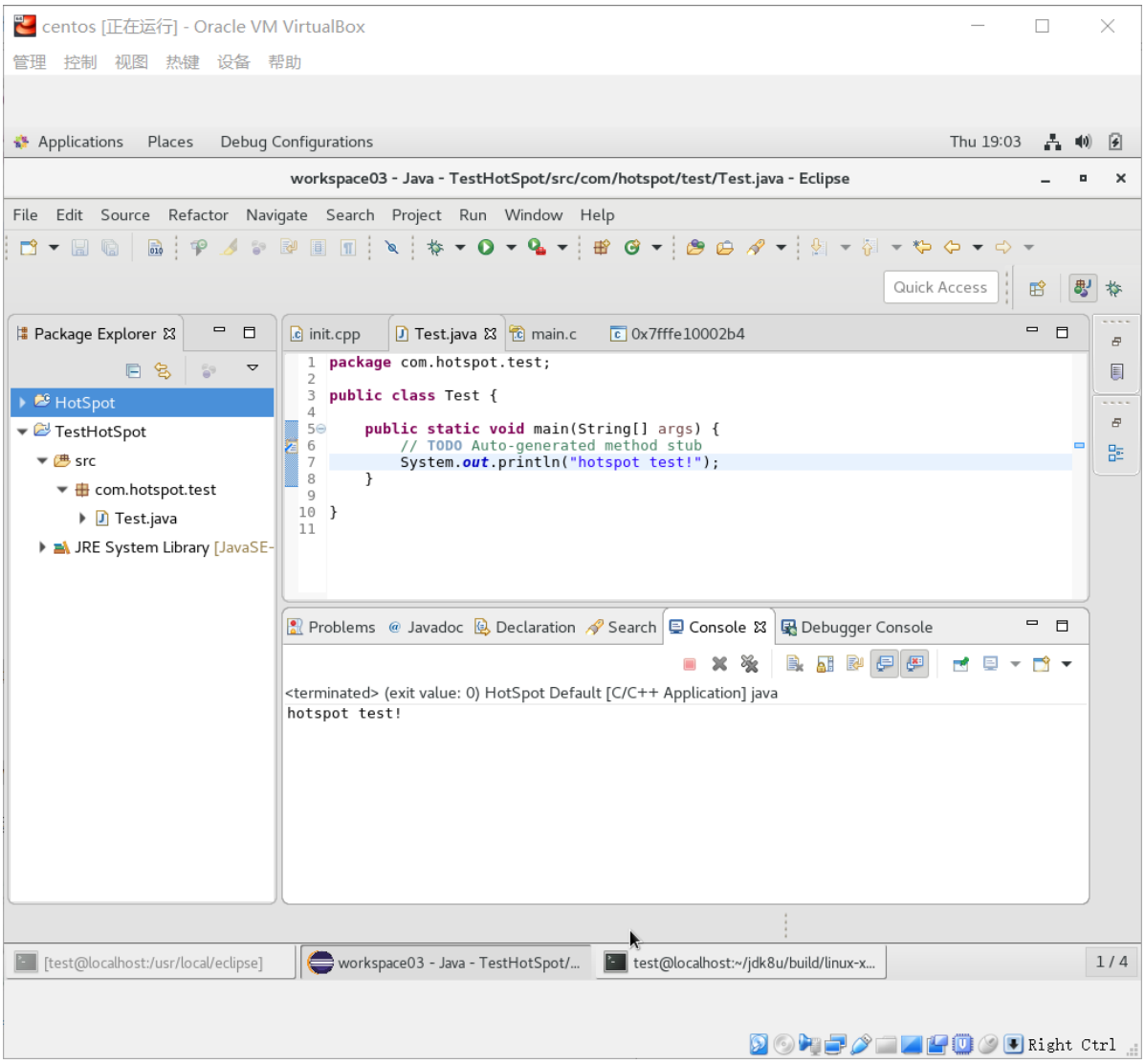


接着就是修改一下上面的HotSpot的配置参数，也就是

```
Arguments选项卡：
com.hotspot.test/Test（注意在前面加上包名，然后是运行类）
Environment选项卡，修改变量：
CLASSPATH=.:/root/workspace03/TestHotspot/bin（java工程生成的bin目录，不包括包名目录）
```



最后就是debug运行HotSpot项目了，这里可以看到会有提示异常，但是最后能够显示输出结果


centos [正在运行] - Oracle VM VirtualBox
管理 控制 视图 热键 设备 帮助
Applications Places Debug Configurations
Thu 19:03
workspace03 - Java - TestHotSpot/src/com/hotspot/test/Test.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
HotSpot
TestHotSpot
src
com.hotspot.test
Test.java
JRE System Library [JavaSE-
init.cpp
Test.java
main.c
0x7fffe10002b4
1 package com.hotspot.test;
2
3 public class Test {
4
5 public static void main(String[] args) {
6 // TODO Auto-generated method stub
7 System.out.println("hotspot test!");
8 }
9
10 }
11
Problems Javadoc Declaration Search Console Debugger Console
<terminated> (exit value: 0) HotSpot Default [C/C++ Application] java
hotspot test!
[test@localhost:/usr/local/eclipse]
workspace03 - Java - TestHotSpot/...
test@localhost:~/jdk8u/build/linux-x...
1 / 4
Right Ctrl