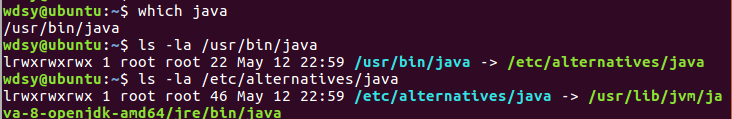
# 1. 单纯安装ODL，只需要配置好JAVA\_HOME即可

## 1 JAVA\_HOME的设定



## 2 也通过 which java命令来寻找JAVA\_HOME路径，示例如下：



则/usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java则是最后路径

## 3 示例：[Ubuntu16.04下安装OpenDaylight](https://blog.csdn.net/muyuxiaoxiang/article/details/79238959)



# 2. 除了odl外还需要安装其他需要java环境的软件

## 1 安装JDK

这里在Ubuntu中安装Java环境及配置相应的环境变量

下载安装JDK: [JDK下载](http://www.oracle.com/technetwork/java/javase/downloads/index.html)（一般用JDK8,现在Java已经到Java 10了，选择8及以上的，一般选择JDK8）

1. 下载安装



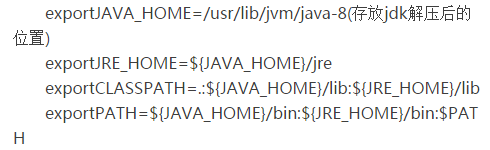
1. 源码安装



2里面的源码安装这个路径好像有问题，我访问不了，可以根据需要自行修改，其中JDK可以放在如上图目录下面，也可以放在/opt目录下

## 2 修改环境变量

（1）







（2）

后面编辑文件可以用vi，命令不熟的可以用gedit打开文件，记住**在编辑文件前要看其先看其权限看能不能写或执行，还有添加成功后可能会出现如下图错误**export命令大体都是一样的(最好选用后一种环境)



原因就是export和变量之前的空格要去掉

安装成功后可以查java版本

开启karaf后出现如下警告（20180709安装Lithium时遇到的问题）：

Java HotSpot(TM) 64-Bit Server VM warning: ignoring option MaxPermSize=512m; support was removed in 8.0

修改Linux 的环境变量配置文件 .bash\_profile，把环境变量值 ***export JAVA\_OPTS="-server -XX:MaxPermSize=500M -Xmx500M"***中的-XX:MaxPermSize去掉即可。

## 3 安装控制器 (参照SDN环境部署与opendaylight开发入门 程丽明编 P179)

## 3.1 Controller 项目的源码安装指南

3.1.1 配置好项目编译和运行的软件环境要求

3.1.2 下载controller源码

(1) git clone <https://git.opendaylight.org/gerrit/p/controller.git>

git branch //查看当前下载版本

git checkout –b origin/stable/lithium

(2) git clone <https://github.com/opendaylight/controller>

git checkout release/lithium

3.1.3 编译controller子项目源码

$ cd controller

$ mvn clean install –DskipTests

3.1.4 编外（项目更新）

$ git remote update

$ git rebase origin/master

$ mvn clean install –DskipTests

启动：

$ cd controller/karaf/opendaylight-karaf/target/assembly/bin

$ ./karaf

启动后可以通过feature:install命令来安装组件

## 3.2 Controller 项目的快速安装指南

在 <https://www.opendaylight.org/downloads>下载合适的发布版本，然后进行解压。

启动：

$ cd distribution-karaf/bin

$ ./karaf

[**opendaylight控制器源码编译**](http://yangxm.vchs.net.cn/2017/10/25/odl-controller-compiling)

**1. 源码下载**

  使用[git](https://git-scm.com/)或[Github Desktop](https://desktop.github.com/)从[Github](https://www.github.com/)克隆需要的源码。

1. [控制器controller](https://www.github.com/opendaylight/controller)，已包含md-sal、config、archetypes、commons、blueprint和model等features；
2. [网络配置组件netconf](https://www.github.com/opendaylight/netconf)，已包含protocol-framework和restconf等features；
3. [WEB服务核dlux](https://www.github.com/opendaylight/dlux)；
4. [WEB应用dluxspps](https://www.github.com/opendaylight/dluxapps)；
5. [交换机功能组件l2switch](https://www.github.com/opendaylight/l2switch)；
6. [OF协议openflowplugin](https://www.github.com/opendaylight/openflowplugin)，已包含openflowjava等features。

**2. 集成features**

  与其他早期的版本不同，当前版本的ODL控制器源码中，并没有包含相应的features，需要自己按需集成。虽然也可以编译安装完ODL控制器后，再执行其它features的编译和安装工作，但直接将必要的features源码集成到ODL控制器源码，有利于控制器的部署工作。

**2.1. 拷贝feature源码**

  实际操作时，请用待集成的feature的名称替换下文中的FeatureName，如netconf或dlux等。

**2.1.1 新建文件夹**

1. /pathto/controller/features/FeatureName
2. /pathto/contorller/opendaylight/FeatureName

**2.1.2 拷贝文件**

1. /pathto/FeatureName/features/\*.\* **TO** /pathto/controller/features/FeatureName/
2. /pathto/FeatureName/\*.\* **TO** /pathto/contorller/opendaylight/FeatureName/

FeatureName中与features和karaf等独立部署时需要使用的文件夹无需拷贝。   
例如：   
netconf的根文件夹中包含有docs、netconf、restconf、protocol-framework和karaf文件夹，其中的karaf就不需要拷贝，docs也可不拷贝。

**2.2. 修改feature的包管理文件pom.xml**

  修改文件/pathto/controller/opendaylight/FeatureName/pom.xml的modules节，将其中不需要的模块去掉。

例如： 对netconf（/pathto/controller/opendaylight/netconf/pom.xml），删除或注释(<!– 原有标记行 –>)掉features和karaf模块。

**2.3. 修改ODL控制器的包管理文件pom.xml**

**2.3.1. 添加模块**

  在文件/path/controller/pom.xml的moudles节中，逐个添加模块。

<module>opendaylight/FeatureName</module>

......

**2.3.2. 添加features**

  在/path/controller/features/pom.xm的moudles节中，逐个添加模块。

<module>FeatureName</module>

......

**注意：** 两个文件中的模块添加方式不一样，前者需要加上“opendaylight/FeatureName”，后者则直接使用FeatureName。

**2.4. 修改ODL控制器Karaf的包管理文件pom.xml**

  修改文件/pathto/controller/karaf/opendaylight-karaf/pom.xml，在其依赖集“dependencies”节中添加features。通常，在文件/path/controller/features/FeatureName/features-FeatureName/pom.xml中，会有添加feature所需要的信息，包括groupId、artifactId和version等。

  例如，添加netconf的features的代码如下

<dependency>

<groupId>org.opendaylight.netconf</groupId>

<artifactId>features-netconf</artifactId>

<version>1.4.0-SNAPSHOT</version>

<classifier>features</classifier>

<type>xml</type>

<scope>runtime</scope>

</dependency>

其中的classifier、type和scope基本上都是固定的，主要是修改groupId、artifactId和version这三个标记。

**3. 源码编译**

  由于maven-plugin-plugin、karaf-maven-plugin和maven-antrun-plugin这三个插件Eclipse已不集成，所以不能直接在Eclipse中编译OpenDaylight。而且，也不能在IDEA Intellij中编译。因此，只能在终端中使用mvn命令进行编译工作。初次编译会耗费比较长的时间。  整体编译控制器，打开命令窗口，使用mvn编译即可。

cd /pathto/controller

mvn clean install –Dcheckstyle.skip=true –DskipTest=true –Dmaven.javadoc.skip=true

  编译时可用的命令行参数：

| **参数** | **说明** |
| --- | --- |
| -D skipTests | 跳过测试 |
| -e | 显示详细的编译错误信息 |
| –settings=/pathto/settings.xml | 指定mvn的工作时使用的配置文件 |
| -rf :MoudleName | 从指定模块开始编译 |

**4. 运行**

  在命令终端中，运行karaf，即可启动控制器容器，但并没有全部安装需要的features。

**4.1. 启动karaf**

cd /pathto/controller/karaf/opendaylight-karaf/target/assembly/bin

./karaf

**4.2. 加载features**

  启动容器后，可以使用feature:list[ | grep xxx]查看所有的features或能与给定字符串匹配的features。

feature:install odl-restconf

feature:install odl-l2switch-switch

feature:install odl-mdsal-apidocs

feature:install odl-dlux-core

feature:install odl-dluxapps-applications

feature:install odl-dluxapps-yangutils

**5. 测试**

  启动浏览器，输入http://IP地址:8181/index.html，页面正常显示后，以用户名“admin”和密码“admin”登录，即可进入ODL控制器的管理界面。

  如果是其他情况，请注意karaf控制台的输出信息和日志。

**6. 调试OpenDaylight**

  由于不能在IDE中编译和运行OpenDaylight，所以，只能通过远程的方式调试。

**6.1. 启动控制器**

  带debug参数启动OpenDaylight，早期使用的-debug，但从Helium版开始，直接使用debug即可。

cd \pathto\karafdist\bin

[./]karaf debug

  注意ODL启动时给出的端口号。

**6.2. 调试工具设置**

  Eclipse和Interllij的设置都差不多，都是在运行（Run）的配置中将运行中的OpenDaylight和调试器进行关联，包括远程主机的IP地址和端口（上一步中的），以及配置源代码的位置。

**[控制器源码编译生成发行版控制器](https://www.sdnlab.com/17399.html)**

**SDN开发：**

**[SDN开发笔记](https://www.sdnlab.com/18316.html)**

**1、SDN开发环境的搭建（win7环境）  
2、SDN控制器的使用（ubuntu环境搭建、controller使用、mininet的使用）  
3、ODL源码编译生成发行版控制器  
4、md-sal应用程序开发  
5、应用程序集成到ODL控制器  
6、yang模型详解  
7、md-sal的l2switch源码分析**

[**应用程序集成到ODL控制器**](https://www.sdnlab.com/17863.html)

1. 项目拷贝到正常运行的ODL碳版本上

在ODL中，bundle位于ODL发行版的目录的system/子目录下，以maven的包的形式组织。

自定义项目的bundle存在于当前项目目录下的karaf/target/assembly/system下面，例如

项目：

<groupId>cn.com.weif.hello</groupId>

<artifactId>hello</aritifactId>

<package>cn.com.wei.hello::</package>

(1) copy

假设当前的hello项目位于/home/guest/bin/hello，那么拷贝命令为

cp -Ru /home/guest/bin/hello/karaf/target/assembly/system/cn   /home/guest/distribution-karaf-0.6.2-Carbon/system/

(2) repo-add

在/karaf/target/assembly/etc/org.apache.karaf.features.cfg

中查看featureRepositories这一行，在最后得到hello项目的maven路径：

mvn:cn.com.wei.odl/hello-features/1.0.0-SNAPSHOT/xml/features

并在运行的ODL中添加和安装hello的bundle：

#opendaylight*@root*> repo-add mvn:cn.com.wei.odl/hello-features/1.0.0-SNAPSHOT/xml/features

#opendaylight*@root*> feature：install odl-hello-ui

#opendaylight*@root*> feature:list –i | grep hello

可以看见运行结果正常

移除bundle命令：

#opendaylight*@root*> repo-remove mvn:cn.com.wei.odl/hello-features/1.0.0-SNAPSHOT/xml/features

2.编译hello项目到ODL发行版本中

（1） 编译ODL发行版

#git clone https://git.opendaylight.org/gerrit/integration/distribution.git

#git tag

#git checkout release/carbon-sr2  (切换需要的版本)

#mvn mvn clean install

编译成功后在当前目录下的distribution-karaf\target下面有压缩包形式的发行版:  
distribution-karaf-0.6.2-Carbon.tar.gz, distribution-karaf-0.6.2-Carbon.zip  
当前目录下的distribution-karaf\target\assembly为解压缩后的karaf执行版本

(2) 查看hello项目的信息

mvn:cn.com.wei.odl/hello-features/1.0.0-SNAPSHOT/xml/features

(3) 添加bundle信息

此步骤之前，确保hello编译运行和测试例运行通过, 并且用mvn clean install 安装其bundle到了.m2/repository下面。  
在当前的distribution目录下面, 修改如下两个pom.xml文件

vi features-index/pom.xml 添加：

<dependency>

<groupId> cn.com.wei.odl </groupId>

         <artifactId> hello-features</artifactId>

         <version> 1.0.0-SNAPSHOT</version>

          <classifier>features</classifier>

          <type>xml</type>

</dependency>

vi features-index/src/main/resources/features.xml 添加

<repository> mvn:cn.com.wei.odl/hello-features/1.0.0-SNAPSHOT/xml/features

</repository>

(4) 重新编译 mvn clean install

(5)-(7) 可选：

（5） 编译在distribution/distributions/karaf/target可得到

distribution-karaf-0.6.2-Carbon-SR2.tar.gz和

distribution-karaf-0.6.2-Carbon-SR2.zip

(6) 两个文件即官方上提供下载的发行版控制器。可以将distribution-karaf-0.6.2-Carbon-SR2.tar.gz复制到其他jvm环境解压运行。

#opendaylight*@root*> feature:install odl-hello-rest

#opendaylight*@root*> feature:list –I | grep hello