Instalando o “Java” e compilando com o “Javac” no Linux Kubuntu 13.04

Abra o terminal, e digite:

Primeiro vamos atualizar e limpar o sistema operacional.

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get purge

$ sudo apt-get autoremove

Agora vamos instalar o java via repositório

$ sudo add-apt-repository ppa:webupd8team/java

$ sudo apt-get update

$ sudo apt-get install -y oracle-java8-installer

De um reboot na máquina.

Eu sempre faço isso depois de cada nova instalação que eu considero de importância para o Sistema Operacional.

Dica:- Depois de atualizações em que o sistema operacional por si só pede para ser reiniciado, você deve navegar nele por alguns minutos e em seguida desliga o sistema; após desligado espere alguns segundos e ligue o computador de novo.

Fazendo isso percebi que alguns problemas de importância quase irrelevante simplesmente desaparecem.

Se você vai trabalhar com a linguagem de programação java pre4cisa (não necessariamente) de uma IDE e eu vou indicar o Eclipse, mas pode ser o NetBeans.

No terminal faça o seguinte:

$ sudo apt-get remove --purge eclipse

Atualize e limpe novamente o sistema operacional.

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get purge

$ sudo apt-get autoremove

Agora vamos instalar o Eclipse via repositório

$ sudo add-apt-repository ppa:eclipsescada/ppa

$ sudo apt-get update

Se tudo está certo continue:-

$ sudo apt-get install eclipse-scada-ping

Em seguida digite:

$ sudo apt-get install eclipse-platform

Pronto, o programa Eclipse está instalado e pronto para o uso.

Mas a programação em Java pode ser feita usando apenas um simples editor de texto como o Kate e a sua compilação pode ser feita uisando o “javac”, nativo da JDK (Java SE Development Kit).

Para saber se o seu “javac” esta funcionando vá para o terminal e digite o seguinte:

$ javac

Se o resultado for:-

$ javac

Usage: javac <options> <source files>

where possible options include:

-g	**Generate all debugging info**
-g:none	**Generate no debugging info**
-g:{lines,vars,source}	**Generate only some debugging info**
-nowarn	**Generate no warnings**
-verbose	**Output messages about what the compiler is doing**
-deprecation	**Output source locations where deprecated APIs are used**
-classpath <path>	**Specify where to find user class files and annotation processors**
-cp <path>	**Specify where to find user class files and annotation processors**
-sourcepath <path>	**Specify where to find input source files**
-bootclasspath <path>	**Override location of bootstrap class files**
-extdirs <dirs>	**Override location of installed extensions**
-endorseddirs <dirs>	**Override location of endorsed standards path**
-proc:{none,only}	**Control whether annotation processing and/or compilation is done.**
-processor <class1>[,<class2>,<class3>...]	**Names of**

the annotation processors to run; bypasses default discovery process

-processorpath <path> Specify where to find annotation processors

-parameters Generate metadata for reflection on method parameters

-d <directory> Specify where to place generated class files

-s <directory> Specify where to place generated source files

-h <directory> Specify where to place generated native header files

-implicit:{none,class} Specify whether or not to generate class files for implicitly referenced files

-encoding <encoding> Specify character encoding used by source files

-source <release> Provide source compatibility with specified release

-target <release> Generate class files for specific VM version

-profile <profile> Check that API used is available in the specified profile

-version Version information

-help Print a synopsis of standard options

-Akey[=value] Options to pass to annotation processors

-X Print a synopsis of nonstandard options

-J<flag> Pass <flag> directly to the runtime system

-Werror Terminate compilation if warnings occur

@<filename> Read options and filenames from file

$

O seu “Javac” está funcionando bem, caso contrário (porque obviamente algo não saiu bem, provavelmente ele não está lincado no path) use o link:- Solucionando o "javac: command not found" para solucionar o problema.

Com tudo testado e funcionando crie o seu código java no Kate e o salve com o nome que lhe convier, nesta caso vou salvá-lo como **YYY.java** no diretório **XXX** e só.

Em seguida vamos compilar o programa, no terminal vá para o diretório (pasta) **XXX** e nele digite:

$ javac YYY.java

Se ele não retornar nada, é porque a compilação foi perfeita e o código está impecável.

Agora note que foi criado o arquivo **YYY.class** no diretório onde o **YYY.java** foi compilado e neste local é que o iremos executar, com o comando, ainda no terminal:

$ java YYY

É só isso mesmo, e você verá no terminal a execução de suas instruções.

Se você, depois de tudo isso, ainda estiver com prolemas com o uso Java, vá para o link:-

http://www.webupd8.org/2012/09/install-oracle-java-8-in-ubuntu-via-ppa.html

Lá faça o download do java.deb que nneste meu caso tem o nome:-

oracle-java8-installer_8u0-1~webupd8~8_all.deb

Fontes:-

http://www.vivaolinux.com.br/dica/Compilando-com-o-javac

http://www.vivaolinux.com.br/dica/Solucionando-o-javac-command-not-found

http://www.vivaolinux.com.br/dica/Eclipse-42-Instalacao-no-Ubuntuederivados

http://wiki.eclipse.org/EclipseSCADA/Installation/ICMPCheck

São Paulo, SP, 10 de Abril de 2014

Mkmouse