I'm not a bot

Java Virtual Machine (JVM) is the essential component that allows any Java program to be executed. To operate it smoothly, you will need at least one Command Line Tool. However, if you are on the Ubuntu Operating System, your job has already been simplified significantly. If you are looking to complete the installation of Java in Ubuntu, you are just a few steps away. Unlike Python, Java doesn't come up with a preinstalled runtime environment on Ubuntu. So, the process of installing JDK in Linux, especially on Ubuntu will be a bit tricky. Downloading JDK files from the Internet and installing it on Ubuntu will not be the solution here. To Install Java in Ubuntu, a different set of commands we need to use. This article intends to discuss what are the 2 methods to install Java in Ubuntu operating system. How to Install Java in Ubuntu?To Install Java on Ubuntu, We're going to see the following 2 methods: Method 1: Install Java in Ubuntu Using Automatic MethodTo Install Java in Ubuntu via Automatic Method, Follow these 3 steps carefully - Step 1: First, we have to Install JDK in Linux. For that purpose, the following command will be executed. Command: sudo apt install default-jdk Step 2: At last, the JRE File of Java will be installed using the following command. sudo apt install default-jre Step 3: To verify the installation, the following command you can use. It will prompt the Java Version used there. java -version Hooray! Java is now present on the Ubuntu system using these simple executions of two commands. Method 2: Install Java in Ubuntu using the Manual MethodTo Install Java of any version in Ubuntu via Manual Method, You need to follow these 4 Steps - Step 1: On the Ubuntu Terminal, execute the following command. The command will make ready the Java Repository for installing JDK Files. Command: sudo add-apt-repository ppa:linuxuprising/java -y Step 2: Now, before performing the Java installation, the System Update should be done. For that reason, the following command will be used. Command: sudo apt-get update Step 3: Now, the installation of Java 13 will be performed. As per your choice, you can go for any other Java installation like Java 17. Command: sudo apt install oracle-java-installer Step 4: While installing the Java, one window will appear for your confirmation. Use the TAB Key to go to the Yes section. Press Enter. ConclusionSo, the above-mentioned steps are highly effective way to install Java in Ubuntu OS. In this case, we have to install JDK in Linux as well as the JRE separately to get the exact performance. If you want to download Java File on Ubuntu from the internet for the installation process, you will get an obvious error message while installing it. Installing software on Linux is usually easier, or at least that's what it seems like. But this is usually not the case as we've come to a realization that installing and configuring some specific tools on Linux might be more time-consuming than on Windows or MacOS. An example of this can be seen when you want to install the latest version of Java on Ubuntu and make it your default Java version. Yes, I know that you're wondering about other Linux distributions like Fedora or Arch, and so on, but each distribution has some specific advantages and disadvantages over others. Ubuntu is one of the most commonly used Linux distributions, and most people who would like to try out the Linux operating system for the first time usually start their Linux journey with Ubuntu. However, if you're a Java developer who is migrating from a Windows machine to an Ubuntu based Linux machine, you might find it tedious to configure the latest version of Java on Ubuntu as opposed to Windows where you simply have to download and install the latest version, and add the directory to the path variable. In this article, I will cover everything you need to know and do to configure your Ubuntu operating system for Java development. I will explain each of the processes with appropriate screenshots and test runs. I have also created a full-length video showing all of the procedures. You can find the video at the end of this article. By the way, if you're wondering, "Hey Fahim! Who told you that installing Java on Windows is easier?", then you should probably check out my article on how to install Java on Windows. Before proceeding further, you might want to check whether you already have Java installed on your Ubuntu. You can do this using the terminal. If it returns any version, then that means a version of Java is already installed on your machine. But if it returns something different, then we can assume that you do not have Java installed or it is not configured correctly. Simply open your terminal. You can also use the shortcut keys for opening the terminal on Ubuntu: Ctrl + Alt + T Open the terminal Then run the command: java --version java --version If it says "java not found" (like you see in the image above), then you can be sure that your system does not have Java installed. But if it returns any version of Java or JDK (Java Development Kit), then Java is already installed on your machine. Based on the installed version and your need for the other versions, you can remove the older one and install the newer version or you can keep both of them and make one version the default version. There are many ways to install Java on Ubuntu. You can use the CLI (Command Line Interface) where you will install via the terminal, or you can download the package and install it using GUI (Graphical User Interface). In this article, I am going to show you how you can download the latest Java from Oracle and install it on your Ubuntu machine. But that, you have to choose between installing the JRE (Java Runtime Environment) or the JDK (Java Development Kit) version for Java. If you don't understand the differences between them or which one you need for your tasks, then here is a comparison for you: Here are some differences between JDK and JRE in Java: JDKJRE It is used to develop Java applications and contains development tools like debuggers.It is used only to execute Java programs. As it is a complete package for Java development, it contains almost everything you might need as a Java developer.It is mainly used for end users, who do not develop Java applications but only run Java applications as software or tools in their systems. As it is responsible for Java development, you will get all of the development and debugging tools you need as a Java developer.If you only want something lightweight, only to run Java applications, then it is the right choice for you. But keep in mind that it doesn't contain any tools like compilers, debuggers, or any other necessary development or debugging features in it. The sole purpose of it is to support the files required for executing them on the end systems. If you are a developer, then you should install the JDK instead of the JRE. On the other hand, if you are just a normal user who will not program or write code at all, you can install the JRE. In this article, we'll be installing the JDK version because that covers everything. If you already have a JDK then you do not need to install JRE separately. Before we install Java, we need to ensure that we've installed all the necessary updates for the Ubuntu operating system. To update the your Ubuntu OS, simply use the sudo apt update command . Then provide your password and hit the enter key. Update the system After updating, if you get a message that says something needs to be upgraded then you can upgrade them using sudo apt upgrade. 23 packages can be upgraded on my system right now. In your case, it might be different. Make sure to press "y" or "Y" while upgrading. Upgrading the system The upgrade might take some time depending on the file sizes that it needs to download and your internet speed. Make sure that everything has been upgraded without creating any errors. You can download the official JDK from Oracle website. Once the landing page loads, click on Products on the top navigation bar: Oracle website navigation options Then click Java under the Hardware and Software section. Click Download Java. Here, you will get all the latest JDK files. For development purposes, it is recommended to use the LTS (Long Term Support) versions as they receive stable version updates for an extended period. I am going to install the JDK 20 version (which is the latest version as of the time of writing this article). It is not the LTS version of course, but if you follow through with this article, then you'll be able to install any version you want swiftly! In your case, I would recommend installing the latest LTS version of JDK. But if you want continuous access to latest JDK features (these features might not be stable), then you can download the latest JDK. To download, select Linux in the operating system section and download the file for x64 Debian Package. Clicking on the download link will start the download for the Debian package file. Depending on your internet speed, it can take a shorter or longer amount of time. As I am writing this article at night and my internet speed remains slow at night, it is taking longer to download the package file in my system. I have downloaded the package file using the Mozilla Firefox browser and by default, it downloads files in the Downloads directory. Simply go to the directory where you downloaded the file and open the terminal there. Usually, if you go into any directory and right-click, you will see a context menu that says Open in Terminal. By using that, you can open your terminal in that directory. Alternatively, if you open the terminal elsewhere, you can use the cd command to go into any specific directory. For example, I have opened my terminal elsewhere. I am using the cd command to go into my Downloads directory as can be seen in the image below: You can use the ls command to see all the files and folders available on a particular directory: After downloading the file, you will see that the file name also contains the version name which is necessary, but if you think that would be troublesome to type later on, you can shorten the filename as well. For this article, we'll use the default filename. Grab the full directory path where the JDK package file is downloaded. You can use the shortcut Ctrl + L to display the whole directory path. For me, the current directory path where my JDK Debian file is located is /home/fahim/Downloads/. Make sure to copy the address. Next, open the terminal. You can do that using the shortcut Ctrl + Alt + T. I like to install the JDK using the terminal, but if you can also install it using the GUI (Graphical User Interface). But I recommend using the terminal as that would also help you debug any issues you face during installation. Use the sudo apt install /home/fahim/Downloads/jdk_filename.deb command to start the installation. For me, the entire command is sudo apt install /home/fahim/Downloads/jdk-20_linux-x64_bin.deb. Hit the enter key: Input your password and type "y" when it asks for your permission to install the package. Make sure that you have successfully installed the package before proceeding to the next step. You might get N: Download is performed unsandboxed as root as file '/home/fahim/Downloads/jdk-20_linux-x64_bin.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied). But don't worry about that because we performed the installation "unsandboxed" intentionally. You will not face any problems if you use the Debian package downloaded from the right source. You can clear the terminal using the command clear. We need to make sure that if it updates, it does not download any downgraded version of Java. You can do that using the sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk-version/bin/java 1 command. Since I'm using the "JDK - 20" version, my command would be sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk-20/bin/java 1. Make sure to change the jdk-version to match your installed JDK version. We need to do the same thing for the javac (Java Compiler) version as well. The command would be sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/jdk-version/bin/javac 1. My command would look like this: sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/jdk-20/bin/javac 1. Remember to change the jdk-version to match your installed JDK version. We'll also do the same for jar. JAR is essential for running Java based applications directly in the system. The command would be sudo update-alternatives --install /usr/bin/jar jar /usr/lib/jvm/jdk-version/bin/jar 1. My command would look like this: sudo update-alternatives --install /usr/bin/jar jar /usr/lib/jvm/jdk-20/bin/jar 1. Then change the jdk-version to match your installed JDK version. This is all for most of the cases. You are good to go! But if you face any kind of problems, then check the complete video provided below. In that video, I talked about a lot of possible issues and how to solve them. If you have multiple Java versions installed on your system, then you need to make one of them the default. This is also covered in the video. Also, if you want to make more modifications, then the video is going to help you with that as well. But for most of the users, this article is everything that you need to install Java on your Ubuntu operating system. Check the complete video for troubleshooting any more issues or if you want to make more modifications. I hope you have enjoyed this article and are able to install Java on your Ubuntu operating system. If you have any questions then please let me know by reaching out on Twitter or LinkedIn. You can also follow me on:GitHub: FahimFBAYouTube: @FahimAmin If you are interested then you can also check my website: This simple tutorial shows how to install the latest Oracle Java (JDK 21 or JDK 24) in Ubuntu Linux via the official binary package. There used to be unofficial Ubuntu PPA to automate the process of installing Java JDK and setup the environment, however, discontinued. So, here's the tutorial to manually install the Java package and set as default. Step 1: Download the pre-build binary package Oracle provides the pre-build binary packages for Linux, Windows, macOS, available to download via the link below: First, select JDK version, then choose download: either "x64 Debian Package" for amd64 devices, e.g., modern Intel/AMD CPUs. or "ARM64 Compressed Archive" for arm64 (aarch64) devices, e.g., Raspberry Pi and Apple Silicon. If you don't even know your CPU architecture type, open terminal (Ctrl+Alt+T) and run uname -m or dpkg --print-architecture command to tell. Step 2: Install the JDK package After downloaded the pre-build package, install it via either option below depends on CPU architecture type. For x86_64 (amd64) AMD/Intel For modern AMD/Intel platform, press Ctrl+Alt+T to open terminal, and run command to install the downloaded .deb package: sudo apt install ~/Downloads/jdk-21_linux-x64_bin.deb Here replace 21 in command if you downloaded JDK 24. Or, just drag and drop the .deb package from your downloads folder into terminal to auto-insert path to that file. The .deb package include a post-install script that automatically set the JDK installation with a higher priority. So, it should be default if you have multiple java. Just in case, you may run the command below to manually set default Java: sudo update-alternatives --config java In the output, input the number for your desired Java version and hit Enter. Similarly, you may run the command below to configure default Javac, Jshell, Jar, etc. sudo update-alternatives --config javac For Intel/AMD platform, you're done! Install the Compressed Archive for ARM devices For the ARM computers (e.g., Raspberry Pi, Apple M1/2, etc), first open terminal (Ctrl+Alt+T) and run command to make sure the /usr/lib/jvm directory is exist: sudo mkdir -p /usr/lib/jvm Then, extract the downloaded archive to that directory by running command: sudo tar -zxf ~/Downloads/jdk-21_linux-aarch64_bin.tar.gz -C /usr/lib/jvm/ Also, replace 21 in command with 24 if you selected JDK 24. After that, you may run the command below to list that directory content which should include the new generated jdk-21 (or jdk-21.x.x) sub-folder. 3. Set JDK 21/24 as default (for ARM only): NOTE 1: For Intel/AMD, you're DONE after installed the .deb package and set default Java. This is only required for ARM platform. NOTE 2: The JDK root folder-name may vary depends on which package you installed. It may be 'jdk-21', 'jdk-21.0.1', 'jdk-24', 'jdk-24.0.1', and so forth. Run ls /usr/lib/jvm to tell and replace jdk-21 in commands below accordingly. a.) Create symbolic links for the executable files: sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk-21/bin/java 1 sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/jdk-21/bin/javac 1 sudo update-alternatives --install /usr/bin/jar jar /usr/lib/jvm/jdk-21/bin/jar 1 As mentioned, you need to replace "jdk-21" in command with e.g., jdk-21.0.1, jdk-21.0.6, jdk-24, jdk-24.0.1, etc. accordingly!!! Similarly, add links for other executable files (e.g., jarsigner, jlink, javadoc) as you need. b.) Next, run the commands below one by one, and type number to select Java JDK 21 as default. sudo update-alternatives --config java sudo update-alternatives --config javac sudo update-alternatives --config jar When done, verify by running command in terminal: java -version javac -version 4. Set JAVA_HOME (for ARM only): As well, you only need to do this for ARM package. If installed the .deb version, just verify JAVA HOME via the bottom command. Also, replace 'jdk-21' below according to ls /usr/lib/jvm command output (e.g., jdk-21.0.6, jdk-24, jdk-24.0.x). Option 1.) Set JAVA_HOME for current terminal console, that will work until you close it: export JAVA_HOME=/usr/lib/jvm/jdk-21 setenv JAVA_HOME=/usr/lib/jvm/jdk-21 Option 2.) To make it permanent, create and edit config file via command: sudo gnome-text-editor /etc/profile.d/jdk.sh Depends on your desktop environment, you may replace gedit with mousepad for XFCE, xed for Linux Mint, pluma for MATE, kate for KDE Plasma, gedit for Ubuntu 22.04 or older. then add following lines: export J2SDKDIR=/usr/lib/jvm/jdk-21 export J2REDIR=/usr/lib/jvm/jdk-21 export PATH=$PATH:/usr/lib/jvm/jdk-21/bin:/usr/lib/jvm/jdk-21/db/bin export JAVA_HOME=/usr/lib/jvm/jdk-21 export DERBY_HOME=/usr/lib/jvm/jdk-21/db And create anther one for C shell: sudo gedit /etc/profile.d/jdk.csh add following lines and save it: setenv J2SDKDIR /usr/lib/jvm/jdk-21 setenv J2REDIR /usr/lib/jvm/jdk-21 setenv PATH ${PATH}:/usr/lib/jvm/jdk-21/bin:/usr/lib/jvm/jdk-21/db/bin setenv JAVA_HOME /usr/lib/jvm/jdk-21 setenv DERBY_HOME /usr/lib/jvm/jdk-21/db Finally, change the permissions via command, and it should take place next time you log in. sudo chmod +x /etc/profile.d/jdk.csh /etc/profile.d/jdk.sh To verify JAVA HOME, run the command below in terminal at next login: java -XshowSettings:properties -version Uninstall Java JDK 21/24 To uninstall the Java package, open terminal (Ctrl+Alt+T) and run command: sudo apt remove --autoremove jdk-21 jdk-24 In the commands below, replace jdk-21, with jdk-21.0.5, jdk-24, jdk-24.0.1, etc, depends on which version you installed. For the ARM version, simply remove all the installed files, by running the commands below one by one: Remove the alternative links: sudo update-alternatives --remove java /usr/lib/jvm/jdk-21/bin/java sudo update-alternatives --remove javac /usr/lib/jvm/jdk-21/bin/javac sudo update-alternatives --remove jar /usr/lib/jvm/jdk-21/bin/jar Remove JDK installation files: sudo rm -R /usr/lib/jvm/jdk-21 And remove JAVA HOME config files: sudo rm /etc/profile.d/jdk.sh sudo rm /etc/profile.d/jdk.csh Enable this blog? Please spread the world :) Share — copy and redistribute the material in any medium or format for any purpose, even commercially. Adapt — remix, transform, and build upon the material for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the license terms. Attribution — You must give appropriate credit , provide a link to the license, and indicate if changes were made . You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original. No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits. You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation . No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material. Java and the JVM (Java's virtual machine) are required for many kinds of software, including Tomcat, Jetty, Glassfish, Cassandra and Jenkins. In this guide, you will install various versions of the Java Runtime Environment (JRE) and the Java Developer Kit (JDK) using apt. You'll install OpenJDK as well as the official JDK from Oracle. You'll then select the version you wish to use for your projects. When you're finished, you'll be able to use the JDK to develop software or use the Java Runtime to run software. Prerequisites To follow this tutorial, you will need: Installing the Default JRE/JDK The easiest option for installing Java is to use the version packaged with Ubuntu. By default, Ubuntu 20.04 includes Open JDK 11, which is an open-source variant of the JRE and JDK. To install this version, first update the package index: Next, check if Java is already installed: If Java is not currently installed, you'll see the following output: OutputCommand 'java' not found, but can be installed with: sudo apt install openjdk-11-jre-headless # version 11.0.11+9-0ubuntu2~20.04, or sudo apt install default-jre # version 2:1.11-72 sudo apt install openjdk-13-jre-headless # version 13.0.7+5-0ubuntu1~20.04 sudo apt install openjdk-16-jre-headless # version 16.0.1+9-1~20.04 sudo apt install openjdk-8-jre-headless # version 8u292-b10-0ubuntu1~20.04 Execute the following command to install the default Java Runtime Environment (JRE), which will install the JRE from OpenJDK 11: sudo apt install default-jre The JRE will allow you to run almost all Java software. Verify the installation with: You'll see output similar to the following: Outputopenjdk version "11.0.11" 2021-04-20 OpenJDK Runtime Environment (build 11.0.11+9-Ubuntu-0ubuntu2.20.04) OpenJDK 64-Bit Server VM (build 11.0.11+9-Ubuntu-0ubuntu2.20.04, mixed mode, sharing)) You may need the Java Development Kit (JDK) in addition to the JRE in order to compile and run some specific Java-based software. To install the JDK, execute the following command, which will also install the JRE: sudo apt install default-jdk Verify that the JDK is installed by checking the version of javac, the Java compiler: You'll see the following output: Outputjavac 11.0.11 Next, let's look at how to install Oracle's official JDK and JRE. Installing Oracle JDK 11 Oracle's licensing agreement for Java doesn't allow automatic installation through package managers. To install the Oracle JDK, which is the official version distributed by Oracle, you must create an Oracle account and manually download the JDK to add a new package repository for the version you'd like to use. Then you can use apt to install it with help from a third party installation script. The version of Oracle's JDK you'll need to download must match version of the installer script. To find out which version you need, visit the oracle-java11-installer page. Locate the package for Focal, as shown in the following figure: In this image, the version of the script is 11.0.7. In this case, you would need Oracle JDK 11.0.7. Your version number may vary depending on when you're installing the software. You don't need to download anything from this page; you'll download the installation script through apt shortly. Then visit the Downloads page and locate the version that matches the one you need. Click the JDK Download button and you'll be taken to a screen that shows the versions available. Click the .tar.gz package for Linux. You'll be presented with a screen asking you to accept the Oracle license agreement. Select the checkbox to accept the license agreement and press the Download button. Your download will begin. You may need to log in to your Oracle account one more time before the download starts. Once the file has downloaded, you'll need to transfer it to your server. On your local machine, upload the file to your server. On macOS, Linux, or Windows using the Windows Subsystem for Linux, use the scp command to transfer the file to the home directory of your sammy user. The following command assumes you've saved the Oracle JDK file to your local machine's Downloads folder: scp Downloads/jdk-11.0.7_linux-x64_bin.tar.gz sammy@your_server_ip:~ Once the file upload has completed, return to your server and add the third-party repository that will help you install Oracle's Java. Install the software-properties-common package, which adds the add-apt-repository command to your system: sudo apt install software-properties-common Next, import the signing key used to verify the software you're about to install: sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys EA8CACC073C3DB2A You'll see this output: Outputgpg: key EA8CACC073C3DB2A: public key "Launchpad PPA for Linux Uprising" imported gpg: Total number processed: 1 gpg: imported: 1 Then use the add-apt-repository command to add the repo to your list of package sources: sudo add-apt-repository ppa:linuxuprising/java You'll see this message: Output Oracle Java 11 (LTS) and 16 installer for Ubuntu (21.04, 20.10, 20.04, 18.04, 16.04 and 14.04), Pop!_OS, Linux Mint and Debian. Java binaries are not hosted in this PPA due to licensing. The packages in this PPA download and install Oracle Java, so a working Internet connection is required. The packages in this PPA are based on the WebUpd8 Oracle Java PPA packages: webupd8team/+archive/ubuntu/java Created for users of Installation instructions (with some tips), feedback, suggestions, bug reports etc.: . . . Press [ENTER] to continue or Ctrl-c to cancel adding it. Press ENTER to continue the installation. You may see a message about no valid OpenPGP data found, but you can safely ignore this. Update your package list to make the new software available for installation: The installer will look for the Oracle JDK you downloaded in /var/cache/oracle-jdk11-installer-local. Create this directory and move the Oracle JDK archive there: sudo mkdir -p /var/cache/oracle-jdk11-installer-local/ sudo cp jdk-11.0.7_linux-x64_bin.tar.gz /var/cache/oracle-jdk11-installer-local/ Finally, install the package: sudo apt install oracle-java11-installer-local The installer will first ask you to accept the Oracle license agreement. Accept the agreement, then the installer will extract the Java package and install it. Now let's look at how to select which version of Java you want to use. Managing Java You can have multiple Java installations on one server. You can configure which version is the default for use on the command line by using the update-alternatives command. sudo update-alternatives --config java This is what the output would look like if you've installed both versions of Java in this tutorial: OutputThere are 2 choices for the alternative java (providing /usr/bin/java). Selection Path Priority Status -- 0 /usr/lib/jvm/java-11-openjdk-amd64/bin/java 1111 auto mode 1 /usr/lib/jvm/java-11-openjdk-amd64/bin/java 1111 manual mode * 2 /usr/lib/jvm/java-11-oracle/bin/java 1091 manual mode Press to keep the current choice[*], or type selection number: Choose the number associated with the Java version to use it as the default, or press ENTER to leave the current settings in place. You can do this for other Java commands, such as the compiler (javac): sudo update-alternatives --config javac Other commands for which this command can be run include, but are not limited to: keytool, javadoc and jarsigner. Setting the JAVA_HOME Environment Variable Many programs written using Java use the JAVA_HOME environment variable to determine the Java installation location. To set this environment variable, first determine where Java is installed. Use the update-alternatives command: sudo update-alternatives --config java This command shows each installation of Java along with its installation path: OutputThere are 2 choices for the alternative java (providing /usr/bin/java). Selection Path Priority Status -- 0 /usr/lib/jvm/java-11-openjdk-amd64/bin/java 1111 auto mode 1 /usr/lib/jvm/java-11-openjdk-amd64/bin/java 1111 manual mode * 2 /usr/lib/jvm/java-11-oracle/bin/java 1091 manual mode Press to keep the current choice[*], or type selection number: In this case the installation paths are as follows: OpenJDK 11 is located at /usr/lib/jvm/java-11-openjdk-amd64/bin/java. Oracle Java is located at /usr/lib/jvm/java-11-oracle/jre/bin/java. Copy the path from your preferred installation. Then open /etc/environment using nano or your favorite text editor: sudo nano /etc/environment At the end of this file, add the following line, making sure to replace the highlighted path with your own copied path, but do not include the bin/ portion of the path: /etc/environment JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64" Modifying this file will set the JAVA_HOME path for all users on your system. Save the file and exit the editor. Now reload this file to apply the changes to your current session: Verify that the environment variable is set: You'll see the path you just set: Output/usr/lib/jvm/java-11-openjdk-amd64 Other users will need to execute the command source /etc/environment or log out and log back in to apply this setting. Conclusion In this tutorial you installed multiple versions of Java and learned how to manage them. You can now install software which runs on Java, such as Tomcat, Jetty, Glassfish, Cassandra or Jenkins. Share — copy and redistribute the material in any medium or format for any purpose, even commercially. Adapt — remix, transform, and build upon the material for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the license terms. Attribution — You must give appropriate credit , provide a link to the license, and indicate if changes were made . You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original. No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits. You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation . No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material. In this tutorial, we'll introduce different methods for installing a JDK on Ubuntu. Then, we'll briefly compare the methods. Finally, we'll show how to manage multiple Java installations on an Ubuntu system. As a prerequisite to each method, we need an Ubuntu system to be logged in as a non-root user with sudo privileges The instructions described below have been tested on Ubuntu 18.10, 18.04 LTS, 16.04 LTS, and 14.04 LTS. For Ubuntu 14.04 LTS, there're some differences, which are mentioned in the text. Please note that both the packages you can download from OpenJDK and Oracle and the packages available in repositories are updated regularly. The exact package names will probably change within some months, but the basic methods of installation will remain the same. 2. Installing JDK 11 If we want to use the latest and greatest version of JDK, often manual installation is the way to go. This means downloading a package from the OpenJDK or the Oracle site and setting it up so that it adheres to the conventions of how apt sets up the JDK packages. 2.1. Installing OpenJDK 11 Manually First of all, let's download the tar archive of the recently released OpenJDK 11: $ wget And we compare the sha256 sum of the downloaded package with the one provided on the OpenJDK site: $ sha256sum openjdk-11_linux-x64_bin.tar.gz Let's extract the tar archive: $ tar xzvf openjdk-11_linux-x64_bin.tar.gz Next, let's move the jdk-11 directory we've just extracted into a subdirectory of /usr/lib/jvm. The apt packages described in the next section also put their JDKs into this directory: $ sudo mkdir /usr/lib/jvm $ sudo mv jdk-11 /usr/lib/jvm/openjdk-11-manual-installation/ Now, we want to make the java and javac commands available. One possibility would be to create symbolic links for them, for example, in the /usr/bin directory. But instead, we'll install an alternative for both of them. This way, if we ever wish to install additional versions of JDK, they will play nicely together: $ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/openjdk-11-manual-installation/bin/java 1 $ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/openjdk-11-manual-installation/bin/javac 1 Let's verify the installation: $ java -version As we can see from the output, we've indeed installed the latest version of the OpenJDK JRE and JVM: openjdk version "11" 2018-09-25 OpenJDK Runtime Environment 18.9 (build 11+28) OpenJDK 64-Bit Server VM 18.9 (build 11+28, mixed mode) Let's have a look at the compiler version also: $ javac -version javac 11 2.2. Installing Oracle JDK 11 Manually If we want to make sure to use the newest version of Oracle JDK, we can follow a similar manual installation workflow, as for OpenJDK. In order to download the tar archive for JDK 11 from the Oracle website, we must accept a license agreement first. For this reason, downloading via wget is a bit more complicated than for OpenJDK: $ wget -c --header "Cookie: oraclelicense=accept-securebackup-cookie" \ 13/90cf5d8f270a4347a95050320eef3fb7/jdk-11.0.1_linux-x64_bin.tar.gz The example above downloads the package for 11.0.1 The exact download link changes for each minor version. The following steps are the same as for OpenJDK: $ sha256sum jdk-11.0.1_linux-x64_bin.tar.gz $ tar xzvf jdk-11.0.1_linux-x64_bin.tar.gz $ sudo mkdir /usr/lib/jvm $ sudo mv jdk-11.0.1 /usr/lib/jvm/oracle-jdk-11-manual-installation/ $ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/oracle-jdk-11-manual-installation/bin/java 1 $ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/oracle-jdk-11-manual-installation/bin/javac 1 The verification is also the same. But the output shows that this time, we've installed not OpenJDK but Java(TM): $ java -version java version "11.0.1" 2018-10-16 LTS Java(TM) SE Runtime Environment 18.9 (build 11.0.1+13-LTS) Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.1+13-LTS, mixed mode) And for the compiler: $ javac -version javac 11.0.1 2.3. Installing Oracle JDK 11 from a PPA Currently, Oracle JDK 11 is also available in a PPA (personal package archive). This installation involves 2 steps: adding the repository to our system and installing the package from the repository via apt: $ sudo add-apt-repository ppa:linuxuprising/java $ sudo apt update $ sudo apt install oracle-java11-installer The verifying steps should show the same result as after the manual installation in section 2.2.1.: $ java -version java version "11.0.1" 2018-10-16 LTS Java(TM) SE Runtime Environment 18.9 (build 11.0.1+13-LTS) Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.1+13-LTS, mixed mode) And for the compiler: $ javac -version javac 11.0.1 On Ubuntu 14.04 LTS the add-apt-repository command isn't available by default. In order to add a repository, first we need to install the software-properties-common package. $ sudo apt update $ sudo apt install software-properties-common Afterward, we can continue with add-apt-repository, apt update and apt install as shown above. 3. Installing JDK 8 JDK 8 is an LTS version that has been around for a while. For this reason, we can find an up-to-date version of OpenJDK 8 in the "Main" repository on most of the supported Ubuntu versions. Of course, we can also head to the OpenJDK website, grab a package there, and install it the same way we've seen in the previous section. But using the apt tooling and the "Main" repository provides some benefits. The "Main" repository is available by default on all Ubuntu systems. It's supported by Canonical — the same company that maintains Ubuntu itself. Let's install OpenJDK 8 from the "Main" repository with apt: $ sudo apt update $ sudo apt install openjdk-8-jdk Now, let's verify the installation: $ java -version The result should list a Runtime Environment and a JVM: openjdk version "1.8.0_181" OpenJDK Runtime Environment (build 1.8.0_181-8u181-b13-0ubuntu0.18.04.1-b13) OpenJDK 64-Bit Server VM (build 25.181-b13, mixed mode) Let's check that the javac executable is available as well: $ javac -version Now we should see the same version number as shown above: javac 1.8.0_181 3.2. Installing OpenJDK 8 on Ubuntu 14.04 LTS On Ubuntu 14.04 LTS, the OpenJDK packages aren't available in the "Main" repository, so we'll install them from the openjdk-r PPA. As we've seen in section 2.3 above, the add-apt-repository command isn't available by default. We need the software-properties-common package for it: $ sudo apt update $ sudo apt install software-properties-common $ sudo add-apt-repository ppa:openjdk-r/ppa $ sudo apt update $ sudo apt install openjdk-8-jdk 3.3. Installing Oracle JDK 8 from a PPA The "Main" repository does not contain any proprietary software. If we want to install Oracle Java with apt, we'll have to use a package from a PPA. We've already seen how to install Oracle JDK 11 from the linuxuprising PPA. For Java 8, we can find the packages in the webupd8team PPA. First, we need to add the PPA apt repository to our system: $ sudo add-apt-repository ppa:webupd8team/java Then we can install the package the usual way: $ sudo apt update $ sudo apt install oracle-java8-installer During the installation, we have to accept Oracle's license agreement. Let's verify the installation: $ java -version The output shows a Java(TM) JRE and JVM: java version "1.8.0_181" Java(TM) SE Runtime Environment (build 1.8.0_181-b13) Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode) We can also verify that the compiler has been installed: $ javac -version javac 1.8.0_181 4. Installing JDK 10 The versions Java 10 and Java 9 aren't supported anymore. You can install them manually, following similar steps as in section 2. You can grab the packages from: Both sites contain the same warning: These older versions of the JDK are provided to help developers debug issues in older systems. They are not updated with the latest security patches and are not recommended for use in production. 4.1. Installing OpenJDK 10 Manually Let's see how to install OpenJDK 10.0.1: $ wget $ sha256sum openjdk-10.0.1_linux-x64_bin.tar.gz $ tar xzvf openjdk-10.0.1_linux-x64_bin.tar.gz $ sudo mkdir /usr/lib/jvm $ sudo mv jdk-10.0.1 /usr/lib/jvm/openjdk-10-manual-installation/ $ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/openjdk-10-manual-installation/bin/java 1 $ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/openjdk-10-manual-installation/bin/javac 1 $ java -version $ javac -version 4.2. Installing Oracle JDK 10 Manually As we've seen in section 2.2., in order to download a package from the Oracle website, we must accept a license agreement first. Contrary to the supported versions, we can't download the older Oracle JDKs via wget and a cookie. We need to head to and download the tar.gz file. Afterward, we follow the familiar steps: $ sha256sum jdk-10.0.2_linux-x64_bin.tar.gz $ tar xzvf jdk-10.0.2_linux-x64_bin.tar.gz $ sudo mkdir /usr/lib/jvm $ sudo mv jdk-10.0.2 /usr/lib/jvm/oracle-jdk-10-manual-installation/ $ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/oracle-jdk-10-manual-installation/bin/java 1 $ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/oracle-jdk-10-manual-installation/bin/javac 1 $ java -version $ javac -version 5. Installing JDK 9 Just like we saw above with OpenJDK 10.0.1, we download the OpenJDK 9 package via wget and set it up according to the conventions: $ wget $ sha256sum openjdk-9.0.4_linux-x64_bin.tar.gz $ tar xzvf openjdk-9.0.4_linux-x64_bin.tar.gz $ sudo mkdir /usr/lib/jvm $ sudo mv jdk-9.0.4 /usr/lib/jvm/openjdk-9-manual-installation/ $ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/openjdk-9-manual-installation/bin/java 1 $ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/openjdk-9-manual-installation/bin/javac 1 $ java -version $ javac -version 5.2. Installing Oracle JDK 9 Manually Once again, we use the same method as for JDK 10. We need to head to and download the tar.gz file. Afterward, we follow the familiar steps: $ sha256sum jdk-9.0.4_linux-x64_bin.tar.gz $ tar xzvf jdk-9.0.4_linux-x64_bin.tar.gz $ sudo mkdir /usr/lib/jvm $ sudo mv jdk-9.0.4 /usr/lib/jvm/oracle-jdk-9-manual-installation/ $ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/oracle-jdk-9-manual-installation/bin/java 1 $ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/oracle-jdk-9-manual-installation/bin/javac 1 $ java -version $ javac -version 6. Comparison We've seen three different ways of installing a JDK on Ubuntu. Let's have a quick overview of each of them, pointing out the advantages and disadvantages. 6.1. "Main" Repository This is the "Ubuntu native" way of installation. A big advantage is that we update the packages via the "usual apt workflow" with apt update and apt upgrade. Furthermore, the "Main" repository is maintained by Canonical, which provides reasonably fast (if not immediate) updates. For example, OpenJDK versions 10.0.1 and 10.0.2 were both synced within a month of release. 6.2. PPA PPAs are small repositories maintained by an individual developer or a group. This also means that the update frequency depends on the maintainer. Packages from PPAs are considered riskier than the packages in the "Main" repository. First, we have to add the PPA explicitly to the system's repository list, indicating that we trust it. Afterward, we can manage the packages via the usual apt tooling (apt update and apt upgrade). 6.3. Manual Installation We download the package directly from the OpenJDK or Oracle site. Although this method offers a great deal of flexibility, updates are our responsibility. If we want to have the latest and greatest JDK, this is the way to go. 7. Exploring Other Versions of JDKs The examples in sections 2 and 3 reflect the current status on Ubuntu 18.04 LTS. Keep in mind that the JDKs and the corresponding packages are updated regularly. Thus it's useful to know how to explore our current possibilities. In this section, we'll focus on surveying the OpenJDK packages in the "Main" repository. If we've already added a PPA with add-apt-repository, we can explore it in a similar manner with apt list and apt show. To discover which PPAs are available, we can head to . If we don't find what we're looking for in the "Main" repository and in the PPAs, we'll have to fall back to manual installation. If we'd like to use an unsupported version, even that can be difficult. As of this writing, we didn't find any packages for Java 9 or Java 10 on the Op