jMonkey Specific Guide for Installing and Running on Android with (Root and Linux)

This guide is ripped from its original source from here:
((where comments can be made) you may notify the author though the commenting system on the other mirrors as well)
http://forum.xda-developers.com/showthread.php?p=40434313#post40434313
and here: ((where updates starts) you may watch me work here)
https://docs.google.com/document/d/17rboS2kaTNtcd7O7PCSIrlXVud79MHkqUTYohYNIbNI/edit?usp=sharing
and here: ((publicly Editable) you may make changes to main project here)
https://docs.google.com/document/d/1CIkOT7iX62z8fpDckoYCw48UtIImjhzG6IJN6E6Qe2E/edit?usp=sharing
and here: ((Maptools Specific Guide For Android) Currently Under construction)
https://docs.google.com/document/d/1W0Hb9B_MNs5CAFxoL_IaCVnmN1BWVvokBYViYlwBXOU/edit?usp=sharing
and here: ((jMonkey Specific Guide For Android) Currently Under construction)
https://docs.google.com/document/d/1_NnPRuqTElR_BDoSvrd6evIfxL7suwYmihhlA2EW7v4/edit?usp=sharing

jMonkey Specific Guide For Android Updates

05222013- Made this Guide today to make installing and running jMonkey as easy as possible on Android Tablets and Phones. The state of this guide is still very raw and the numbering system doesn’t make sense just yet but I’ll be fixing it very soon and making this streamlined. UPDATED this guide to include specific instruction on how to get and install openjdk and it’s family of packages so that you can now run jMonkey on Android. Starting the polishing experience now on the jMonkey clone, Gole: shrink it to under 4 posts on their forums. Section 1 on the whole is now much cleaner and more organized (about 14% more fiber). Cleaned up almost all of this guide as far as directions I'll write a mini-guide near the top to direct readers in how to quickly deploy this system so that mobile development is worry free, then I'll update the main guide, clean up and reformat this guide some more to get it to 'beta' level of usability and submit this to the jMonkey forums. Goodnight for now.

jMonkey Specific Guide For Android Introduction and User Warning

WARNING this guide is still UNDER CONSTRUCTION use at your own risk.
I confirm that for my system I was able to start and update the jMonkey SDK on my Epic 4g (an Android phone with slide out keyboard) running Ubuntu 12.04 with the Debian Kit installation instructions contained within the Debian Kit on board help and then this guide that I wrote to make my life and your life easier when on the go with the ich to develop and no computer handy.
WARNING:Root has the possibility of doing bad things to your device and those around you and running a full desktop Operating System (OS) on top of that is a further compounding of the danger to your device and others. And doing all that on a device that has access to the internet as a whole for long periods of time even furthers that risk. I take no responsibility for what you do with your property, time, or other’s property. This guide is for educational purposes and no harm is intended or wished by the author to anyone or community or the planet as a whole.

jMonkey Specific Guide For Android Contents

jMonkey Specific Guide For Android Downloads
You are here for jMonkey on Android or Raspberry Pi--
-- then follow “Section 1” through “Section 1:1:3.12”
or
-- follow “Section 1:2” through “Section 1:2.7”
then when you’ve got a device with Linux running on it
~~ Download and Run this script that I wrote to install and download everything requried to set up jMonkey on the ARM Linux OS
https://docs.google.com/document/d/19zyqqYSed4TDmXaY7YSeVOTA44QRfAIWhOlxS_djtI0/edit?usp=sharing
or
~~ use “Section 7:4-4.1.3“ to “manually install Java and jMonkey to your system “the hard way” its your choice but I’ve been making this as easy as possible for us all.

jMonkey Specific Guide For Android

Contents

1. Setting up base system:

Downloads for android device.

- Debian kit

https://play.google.com/store/apps/details?id=org.dyndns.sven_ola.debian_kit
- ConnectBot

https://play.google.com/store/apps/details?id=org.connectbot
- Remote RDP free

https://play.google.com/store/apps/details?id=org.toremote.rdpdemo
- Root browser (or use your prefered file browser)

https://play.google.com/store/apps/details?id=com.jrummy.root.browserfree
- Hacker's Keyboard

https://play.google.com/store/apps/details?id=org.pocketworkstation.pckeyboard
- debian-kit-1-5.shar

http://sven-ola.dyndns.org/repo/

1:1 Installing Linux system to Android Device

Currently I’ve documented a total of 4 ways (two apps each including an “in app” or “on device” installation instructions and “manual” or “computer assisted” install instructions) for you the reader to install a form of Linux onto your Android device, there are plenty more out there on the internet and marketplace and I will document as many here as I can for easy compatibility and fast deployment.
For Debian Kit on device install refer to section 1:3:1
For Debian Kit manual install refer to section 1:4
For Limbo in app install refer to section 1:5:1:2-1

For Limbo computer assisted install refer to section 1:5:1:2-2

1:1:2 Verify no major errors are present

1:1:2.1Open Debian Kit: check that there are no red X's; tap on anything in the list for more info and/or press menu and tap read me for the developers original doc.

1:1:2.2 Check that you've enough memory: from the home screen press menu and tap settings; Scroll down to application settings, tap it and view storage use; try to have as much available memory as possible by, moving/removing apps, ruffly 100MB+ worked for me. Note after the initial install you should be able to reinstall/restore the removed apps with little ill effect.

1:1:2.3 Remove data consuming tasks from the equation either by freezing them with an app like “ROM Tool Box” or turning off their sync options or uninstalling them. Mainly this step helps insure that while you're downloading the necessary data that your connection doesn't flip out and cause an error that could have been avoided. You can turn everything back on with little to no ill effect after the install is finished.

1:1:3 Start ConnectBot (rom toolbox terminal emulator also works well for this part with some modification)

1:1:3.1. Tap on the lower left corner where ssh button is and select local instead.

1:1:3.2. Tap in the text field next to the bottom now displaying local and type in a nickname like 'debInstall', hit enter and be presented with-

```
$
```

1:1:3.3. Type su, hit enter, and permit superuser permissions.

```
$ su
#
```

1:1:3.4. Unpack the installer with `sh /sdcard/download/debian-kit-*` note: if downloaded with dolphin browser or some other browser then you'll need to change the file path to reflect that.

```
# sh /sdcard/download/debian-kit-*
```

Or

```
# sh /sdcard/Dolphin_Browser_Mini/download/debian-kit-*
```

Or

```
# sh /mnt/sdcard/download/debian-kit-*
```

See 8:2:1 for example of what your screen should look like without errors.

1:1:3.5. Choose your path:

- Input `2s` will install a debian.img file 2gb in size to your sdcard. Note: this option is suggested as for the first try as this will allow for a stable and fairly sized environment to

test and play with, furthermore I will be writing this guide first for the debian squeeze option and later add the others as I've the time. If choosing this option scroll down to step 1:1:3.6. when the installer starts asking questions.
- If 2gb sounds to small and you're willing to brave the manual install, then you'll want to abort the auto installer at this point and scroll further to step 1:2 Manual Install.

1:1:3.6. Once the install has finished you'll be given three options yes, ovpn, q choose one and only one. When yes or ovpn is used things will be removed, when q is used the debian image is unmounted just as it will be automatically unmounted at the end of either yes or ovpn option. Note: when I install to debian.img file I choose ovpn as this allows for the most amount of space to be freed.

1:1:3.7. Now providing everything is error free the debian.img can be remounted with either of the two commands
```
# deb
```
or
```
# /data/local/deb/deb
```
Note: to see what it will look like the first time you run deb and not encounter errors see section 8:2:4

1:1:3.8. Now to first
- update the list of packages currently installed,
- second upgrade everything,
- third install andromize for compatibility,
- forth install andromize-lxde for a gui,
- fifth install ssh,
- sixth auto remove unnecessary packages,
- seventh clean up,
  and all that in one step that will take more than a few minutes to complete. Note: andromize at some point is going to request input from you;
  - tap on your screen and tap on the control button that briefly shows up in the lower corner, the hit the 'i' button on either you hard keyboard or onscreen keyboard. This will put you in insert mode so that with a d-pad and the spacebar or enter key you can select a highlighted option, I suggest: ctrl+i then space (or enter) as that has worked everytime for me.
```
# apt-get update; apt-get upgrade; apt-get install andromize; apt-get install andromize-lxde; apt-get install openssh-server; apt-get autoremove; apt-get clean
```

1:1:3.9. So you're set to rock and roll as root and have a lot of options if ya know what you're doing.
At some point with all this you may feel uncomfortable with running under sudo root user. Scroll down to '4:3:2 Command line commands for linux' and run those commands to secure yourself some. Or follow the abbreviated commands here to; create a new password for your root

account; and set up an account with normal user permissions; and add your new user to the sudo group:

- Make a new password for root: (this will prompt you for a new password, when you type it won't do anything until you press enter, you will then be prompted a second time to re-input your password and again it won't look like you are typing until you press enter)

```
# passwd
```

- Add a new user: (you will be prompted to input details about this user, feel free to be creative as these details are for you)

```
# adduser [new-user-name]
```

- Elevate that user to sudo level permissions so you can install and remove software with it

```
# adduser [user-name] sudo
```

- Log in to your new user so you can install things under that user

```
# login [user-name]
```

1:1:3.10. Start your new ssh server. Now that the basic requirements to run linux have been met and you've a clean OS we need a way to 'open and 'close' terminal windows that aren't going to remount the system a bunch or run us into out of memory or space errors.

- Start ssh with one of the following commands (in a local connection)

```
# deb s
```

or

```
# /data/local/deb/deb/deb s
```

- Stop ssh with one of the following commands (when you're done)

```
# deb S
```

or

```
# /data/local/deb/deb/deb S
```

1:1:3.11. Once ssh is on you can connect to your device from any other ssh compatible device. For now we are going to use identical directions, almost, to section 4:3:3-3

- Make or open a new or existing ssh connection with connectbot with your username that you want to log in as then the @ symbol followed by "localhost"
- Make sure that in the lower corner that the option is set to ssh and hit the enter key

1:1:3.12. Connectbot or whatever local running ssh app you're using (I'll be testing or hearing about a few I'm sure) should now prompt you for a password. Put in the one you (I hope setup before) set up and you should be greeted by a few lines of text and a # with your username before it. This is where many of the linux commands should be run.
Note: this method is prefered because you can connect and disconnect without sending multiple "deb" commands in the local connection that we only should use for installation and starting the system (first boot and after reboots). Sending multiple "deb" commands can cause all sorts of issues without sending "deb u; deb k" commands between. The other issue that arises with sending multiple "deb" commands followed by a "deb u; deb k" is that you can fill folder structures on the linux OS and your SD card that are designed to rescue unintentionally deleted

files, cache, and junk files, which in other circumstances is really good to know like when a picture disappears from your gallery, but in this case is a real pain. You'll get all sorts of errors so remember one "deb" command per boot and connect through ssh or remote RDP once your system has mounted linux and you'll have a happier experience. Okay moving on.

1:2 Manual install for Debian Kit

1:2.1. Unmount and eject the sd card from your device

1:2.2. Plug the sd card into enough adapters to plug it into your computer. I've used micro to sd adapters and then plugged that into yet another adapter sd to usb with success.

1:2.3. Backup all data. You're about to delete it all.

1:2.4. Using gParted on ubuntu 12.04

- Ensure that gParted has selected the sdcard and not your hard drive... don't select your hard drive that would be bad.
- Check out the current partition layout, perhaps even make note of it somewhere incase something goes wrong. My sandisk started out with a 4mb empty zone and a fat32 partition for the rest.
- Reformat original partition to fat32 by either write clicking the main partition or selecting the partition drop down menu at the top of the window.
- Resize first first partition (pay close attention to not accidently remove, overwrite, or move over the little bit reserved at the start of your sdcard) to the desired amount of space for the android side of your device such as space for; downloads, app cache, apps to sd, photos, and music. This space is all that my android recognizes by default so think about it.
- Second make a second partition filling up the trailing bit left over, you can try something fancy but I chose a fat 32 bootable.
- Apply the three operations with the green checkbox in the upper left quadrant of the window and let it run.
- When gParted finishes unmount and eject the sd card, wait a second or two and plug it all back into your computer.
- If you left gParted open then your sdcard and it's partitions should now show up as selectable in the drop down for devices. Close gParted.
- Open a file browser or two; one for your sd card's first partition and another (window or tab) for where you saved the backup of your sd card's contents.
- Copy your backups back over to the first partition of your sdcard (be sure its the first partition as that is all Android will see as availible and usable by default). I'd advise copying the important things first, app backups folders and system and/or nandroid backups. The things you can live without for a day or two you can come back for because you're here to get to the next step.
- Unmount and eject from computer. Plug back into phone and remount.

1:2.5. Run through the first part of the Debian Kit guide and about when given the option to make a debian.img file. (Sections 1:1 - 1:1.4)
Run the following commands in connectbot under superuser to figure out what the installer uses to identify the second partition on your external sdcard, format it and install linux to it.

1:2.6. This may show the uuid, drive letter, date modified. Make note of todays date and check for three dates in the list that match from when you were repartitioning. Take note of everything.

```
/data/local/deb/armel/busybox fdisk -l /dev/block/vold/*
```

Results may vary try navigation to /dev/ with a root browser on the android side of things, take a look around and adjust the bit after 'fdisk -l' to reflect your device's folder scheme and rerun till you get something that looks like this: (or see section 8:2:2 for full example)

```
            Device Boot      Start         End      Blocks  Id System
/dev/block/vold/179:0p1               1        1279    10264576   c Win95 FAT32 (LBA)
/dev/block/vold/179:0p2            1279        1925     5189632   b Win95 FAT32
```

1:2.7. Now to run the mk-debian -h script and change settings until satisfied. Note: I had one issue when I didn't tell the installer what 'drive' letter to install to;the letter preceding 'Win95 FAT32' in above example or in the below example command you can see the drive letter delineation specified at the end with '-L b' because on my device setup that's the letter in which I need to install to. (see section 8:2:3)

```
# /data/local/deb/mk-debian -i /dev/block/vold/179:2 -s 5310 -L b -h
```

- Which if configured correctly will look like the dump took and pasted below.

```
Script to format a loop disk file or disk partition with ext2/3/4 and install a Debian Linux there.

Usage:
 mk-debian [Options]

Options:
 -d <distro>   specify squeeze/lucid/precise
 -m <path>     specify mount point for disk/device
 -s <megabyte> specify size for new loop disk file
 -i <file|dev> specify loop disk file name or device
 -t <ext2/3/4> specify file system to be created
 -L <label>    specify disk label for formatting
 -D <ip>       specify DNS server to be used
 -M <url>      specify Mirror URL for downloading
 -u            update /system/bin/deb script and exit
 -U            uninstall Debian kit (not debian.img!)
 -C            clean files not required to run Debian
 -h            display usage text and settings

Settings:
 -d squeeze
 -m /data/local/mnt
 -s 5310 Mb
 -i /dev/block/vold/179:2
 -t ext4
```

-L b
-D 8.8.8.8
-M(squeeze) http://ftp.de.debian.org/debian
-M(lucid/precise) http://ports.ubuntu.com/ubuntu-ports
Action: install

Notes:
For creating a loop disk file, you may want to change the default size (in Mb, 512-2047). For formatting a device (-i /dev/xxx) or if overwriting an existing loop disk file, the size is ignored. The loop disk file name or the device name will be written to the bootdeb script's IMG= line if this script succeeds.

If you specify a device, the UUID of the formatted device is instead saved into bootdeb (IMG=uuid:<hexid>). With this, the correct device can be found even after some mixed USB drive/SD card eject-insert sequences.

Check your settings by adding '-h' as the last switch.

Remove the '-h' from the tail end of the command to run when you believe all is good and if all is good then scroll back up to step 7 of the main install guide where it talks about remounting debian.

________SPLIT FOR POSTINGS_______

7. Getting comfortable and customizing your new system

This section is dedicated to making your life easier if you're new to running Linux on Android. Many of these following guides will function no matter what method you use to run Linux on your device; for example I'll be testing ‘Limbo QEMU’ and ‘Linux on Android’ from the android marketplace, which use outright emulation or chroot “under the hood” so to speak to serve up a Linux environment, essentially giving you the same experience as ‘Debian Kit’... However, there will be some key differences in how the device’s hardware is accessed and I’ll have to split out the eccentricities in those methods so the main goal of the entire document can be achieved.

7:1 Connection Methods

Here will be a culmination of all the ways to connect to your linux system from another device or computer or make more than one linux device communicate with one or more devices.

7:1:1 Connecting locally

Type the 'deb' command or "/data/local/deb/deb" command in just about any terminal emulator on the device that Debian Kit is installed to and it should boot on up. Sometimes you'll be prompted during install to run the "bootdeb" script, you can, but all the bootdeb script was when I looked at it with a text editor was a shortcut back to the "deb" script itself.

7:1:2 Connecting wirelessly through wifi

There's a ton of ways to go about wifi networking with either Android or Linux, we just happen to have bothe... so the options for networking with this system are like a ton^2... I'll do my best to bring you all the best here.

7:1:2-1: find your android's IP address

At some point you'll need the IP address for your device. To get this information back out to your devices' home screen:

- Press menu and tap settings
- Tap 'Wireless & networks'
- Tap 'Wi-Fi'
- Press menu and tap 'Advanced'

There you should find your device's IP address and have the option to 'Use static IP' if you know how to use such things. For now make note of your device's IP address and continue on with some of the ways to use this set of numbers.

7:1:2-2.1 Remote RDP in from a second android device

To do this you'll want to have; already started linux on the Host machine (android device) with a 'deb x' command though Connectbot, connected to a wifi access point, and know your IP address.

- Open Remote RDP on the device that is not running Linux. This is usually called the client connection when reading other guides.
- Make a new connection by either tapping the plus (+) sign in the lower right of the screen or by pressing menu and tapping ‘add’
- Type in the IP address of the device that is running linux into the 'Host' line at the top.
- You may press back and then tap on your new connection to test it.

Note: you can always modify the other fields such as 'User' and 'Name' at a later time by disconnecting and then pressing and holding on the connection in Remote RDP main screen and tapping edit

- Input your credentials (sometimes you have to do it twice, don't know why) and be amazed that the interface is actually a little peppier.

Now that you have remote access to the device you can now leave it on a charger somewhere and connect into it from rooms away; even if someone else is using it for browsing, and some gaming... I have yet to try it while receiving a phone call though so you may have to reconnect after the call disconnects and that could loose you a session of whatever you were doing, other's comments encouraged.

7:1:2-2.2 Remote RDP in from a second computer

Internal update: tested while access point received call. Result, internet access dropped, connection between other device and computer stayed alive.
To perform these steps you’ll want a computer running a Remote Desktop Client.
I’m using a laptop running Ubuntu 12.04 and I’m connected to the same wifi network as the device (myTouch 3gs) that I wish to connect to. The Android device and computer are connected to the same wifi access point. The program that I’m using on my laptop is ‘Remmia’ which came stock installed with my operating system. To call up the help documentation:
7:1:2-2.2.1. Open a terminal window on your computer

7:1:2-2.2.2. input the following command

```
# remmina --help
```

7:1:2-2.2.3. This will either result in your system saying that you don't have that package installed and maybe a message too stating that it can be installed through an 'apt-get' command OR you'll receive a helpful message.

- if you get errors; then search on google for an equivalent application that comes on your operating system and see if you can't use that instead of installing yet another thing to your computer.
- if you get a helpful message; then continue to the next step of this mini-guide.

7:1:2-2.2.4. type the following command in your computer's terminal window to open a new window for creating a new connection.

```
# remmina --new
```

7:1:2-2.2.5. There will be a bunch of options but right now we're concerned with the 'Server' line. Input your device's IP address (see section 7:1:2-1 if you need help finding that)

7:1:2-2.2.6. Under 'Resolution' choose 'Custom' and pick something reasonable or leave it at '640x480'

7:1:2-2.2.7. Mouse down to the 'Color depth' option and choose 'High color (16 bpp)'

7:1:2-2.2.8. Click 'Connect to test without saving or click 'Save' to save it for later

7:1:2-2.2.9. Once you click connect a new window should appear, and, if you've already used 'Remote RDP' on the android side of things you'll likely know exactly what to do here... if not follow the next steps
Note: sometimes the screen will look off to one side within the window. To fix this mouse over to one of the corners of the window (like you would if you wanted to resize it) click and hold and move the mouse a little. The display within the window should snap to.

7:1:2-2.2.10. Input your username and password and simit. You may receive the usual error message stating that it couldn't connect; just try it a second time, input your username and password and try logging in.

7:1:2-2.2.11. Once logged in; jump to any other part of this guide and enjoy your linux box on a bigger screen.

7:1:2-2.2.12. To re-open and re-connect to the device after shutting down or being disconnected, run the following command (on the computer side of things) to open ubuntu's RDP

```
# remmina
```

7:1:3 Connecting wirelessly through bluetooth

Using the 'apt-cache search' command and adding bluetooth seems to return results for obtaining drivers and I'll be updating this section with source file install instructions when I get around to it. I worn you things of of this nature have the potential for abuse just like wireshark in end result, however, the other much more pressing matter is transmitting data such as audio to another device based on location data from the GPS so we can make a surround sound system that automatically transmits and plays the correct audio from the correct direction to the point selected as center or main listener compensating for the speed at which sound travels at your specific altitude. (this is another cookie recipe I'll be cooking up)

SPLIT FOR POSTINGS

7:2 Customization of GUI

Order and numbering of this subsection is subject to change, thus any comments should also reference it's subsection title. Here we'll have a compilation of ways to make your mobile viewing and interaction a more pleasant experience. These commands and directions will either be run on a terminal window or on the linux desktop within Remote RDP.

7:2:1 Saving Electrons

7:2:1.1. Turn off the screen saver.

- Mouse over to the lower left corner of the screen
- There you will see a little 'symbol' click it and in a second it'll pop up a dialog.
- Mouse over to options, then screen saver settings, and tap it
- In the upper right quadrant of the new window you'll find a drop down menu with disable screensaver being one of them.
- Select your choice, then exit out. (See note and source above 4:3:4-5)

7:2:1.2. Turn off the CPU monitor

- Open Remote RDP and login as your user with your password
- Once the GUI has loaded send the right click command to the window bar and select 'Add / Remove Panel Items'
- Select 'CPU Usage Monitor'
- Click on remove
- Close the window

7:2:1.3. Set auto lock

- Mouse over to the lower left corner of the screen
- There you will see a little 'symbol' click it and in a second it'll pop up a dialog.
- Mouse over to options, then screen saver settings, and tap it
- In the 'Display Modes' tab select the check box 'Lock Screen After'
- Set the time to the desired amount of inactivity in minutes
- Close the window and wait that long to test it.

7:2:2 Making it Pretty

7:2:2.1. Change your background

- Open Remote RDP and login as your user with your password
- Once the GUI has loaded send the right click command to the virtual desktop by either

    - Double tap+hold for a second or two, then release OR
    - Pressing menu on your device
    - Tapping ‘Mouse’
    - A circle will appear that you may tap and drag to control the mouse
    - 4 options will briefly pop up when you lift your finger away from that circle
    - 3 option from the left will send a right click command to the mouse, use it AND
- Mouse down to ‘Desktop Preferences’ and tap it
- A new window will appear after a moment
- Within the new window under the Appearance tab you’ll find your ‘Wallpaper’ settings and an option to set it.
- when you choose the option to change your wallpaper a new window/file browser will open. And there you may set a new one and discover where to put pictures from your android side so that they may be quickly accessed for wallpapering your desktop.

7:2:2.2. Change your theme

7:2:2.3. Change the location of your Task Bar

7:2:2.4. Task Bar customization
- Open Remote RDP and login as your user with your password
- Once the GUI has loaded send the right click command to the window bar and select 'Add / Remove Panel Items'

From here we can do a great many things to customize your experience but for now I'll step ya through duplicating my setup.
- Mouseover to 'Add'
- In the new pop up dialog choose 'Application Launch Bar' and click 'Add'
- In the main window now select 'Edit' and minimise the Panel Preferences window for now.
- In the Application Launch Bar window that popped behind select 'Accessories' and select 'Root Terminal'
- Click the 'Add' button
- Select 'LXTerminal' and click the 'Add' button
- Close out of the Application Launch Bar when you're done adding shortcuts and restore the Panel Preferences window
- Click on the 'Up' button until the selected application shortcuts are where they should be

7:3 Making things happen without excessive user input

7:3.1. Automount script that prompts then runs linux mount after device boot?

7:3.2. Disabling the prompt for the automount script.

________SPLIT FOR POSTINGS________

7:4 Methods of installing or unpacking

1. Moving files between android and linux when their dual-booted.

2. Un-zipping packages and the .rar's that sometimes get in the mix.

3. Installing or running .sh formated installers or scripts.

7:4-4. Installing from sources.

Apt-get is great for installing and uninstalling a plethora of fun and usefull software, however, it’s not going to be the magic bullet for installing everything you may want on your new linux system. What follows are the steps that I take when installing experimental and unsupported software to my devices. As always be careful when copying what I suggest here as your system may have differences to mine.

7:4-4.1 Unpacking tar.gz files

At some point you'll run into the need to extract or unpack something that you can't normally install. For this example I'll show you how to unpack java from sun built to run on ARM linux; you may find it at the link bellow and then download and copy to a new folder in your home directory named 'SourceFiles' or something that you'll remember.
http://www.oracle.com/technetwork/java/embedded/downloads/javase/index.html
7:4-4.1.1. Run the following commands to back out with change directory to the root comand line, then the next to extract the file

```
# cd /home/[username]/SourceFiles
# sudo tar -xzvf /home/[username]/SourceFiles/ejre-*
```

7:4-4.1.2. Input your password and let it sit for a bit

7:4-4.1.3. Next we'll move to the next section; installing Java for ARM

7:4-4.2 Installing Java for ARM from source

7:4-4.2.1. First to make sure my system is cleaned or purged of openjdk I ran the following command.

```
# sudo apt-get purge openjdk*
```

7:4-4.2.2. Make a new folder in the "usr/lib" called "jvm" by first "cd" over to it in the command window

```
# cd /usr/lib
```

7:4-4.2.3. Then make the directory or file with:

```
# sudo mkdir /jvm
```

7:4-4.2.4. Then we move the extracted source files over to jvm folder found under the [root directory of your device]/usr/lib/

```
# sudo cp -rv /home/[username]/SourceFiles/ejrel1.7.0_21/ /usr/lib/jvm/
```

Note: if you get "cp: cannot stat..." errors; then run

```
# cd ~
```

To get back to the root of your bash screen and try again.
Note: the letter 'v' in both the "cp -rv /home*" and the "tar -xzvf /home*" commands can be removed to keep the amount of text whizzing by to a minimum.

7:4-4.2.5. Then we create a system link in “sbin/java” that points back to where java is located (think of it like a shortcut that your system can use to run Java commands without asking you where it’s installed) with the following command
# sudo ln -fs /usr/lib/jvm/lib/java /sbin/java
Note: in the above command; the "-fs" part = 'f' forces the link, this makes it so that you can re-run this command if it messes up the first time and the 's' part = create a symbolic link, this is like a short cut... sort of, there are different kinds and you can find more by using
# ln --help
in a terminal window.
Note: the "/usr/lib/jvm/lib/java" part of the command = the full path to wherever the java executable really is and the "/sbin/java" part = where your system normally puts links in. These parts may need changing in odd cases, but, if you've followed along so far without issue then it should be 'safe' to continue.

7:4-4.2.6. Finally we check that the system can now find java with the following command
# java -version

7:4-4.2.7. Lone behold you'll likely get an error message stating how "java" isn't a bash command. All is not lost though; we just need to take this to the next guide... 4:2-4.3 Adding Commands to Bash... I know this because you can check your java version by running
# /usr/lib/jvm/lib/java -version

7:4-4.3 Adding Commands to Bash
There be two ways to go about this; one is temporary and uses the alias command in a terminal window, it lasts until your connection ends; the other is getting into your linux bash command file and adding the commands, this will last until you overwrite it again or restore it from a backup of the original or reinstall the entire linux distro, this carries small risk of breaking your install but it seems worth the trouble...
7:4-4.3.1. Open Remote RDP and log in as a normal user

7:4-4.3.2. Open the file browser (navigate to the home directory for your user if you're not already there)

7:4-4.3.3. Mouseover to 'view' at the top of the window (its between 'bookmarks' and 'tools') and select 'show hidden files'

7:4-4.3.4. Make a new folder for backups of linux system files and a new folder in that explaining where to put the back up. For example mine looks like this:
/home/[username]/BackItUP/found_in_home.[username]/[file-that-i-backed-up]

7:4-4.3.5. Find the file called ".bashrc"

7:4-4.3.6. Copy the file to the backups folder

7:4-4.3.7. Open the file ".bashrc" under the original file path eg: "/home/[username]/" with leafpad

7:4-4.3.8. Ensure the part about aliases has the 'if' statement not commented out. Meaning that if you scroll all the way down in leafpad there should be no '#' before the 'if [-f ~bash_aliasas*' where you to find one remove it and save the changes

7:4-4.3.9. Make a new file in your home directory titled '.bash_aliases' by sending the right click command to a blank spot in your home directory and selecting new and selecting create new blank file
Note. If you don't still have hidden files set as visible then the new file will not show up until you do.
10. Open the new file with leafpad and add aliases using the following format

```
# alias <desired alias>='<linux_command>'
```

So for this case we'll use this following command to make an alias for java so it can finely run free.

```
alias java='/usr/lib/jvm/lib/java'
```

Note: the single quotes around the file path need to be there.

7:4-4.3.11. Save and test.
Note: you may have to restart your connection ie: log out and back in for the changes to take effect. Totally worked for me though :-)
Note install process not yet complete; running '.jar' files still return errors. Now trying some new commands:
To let the system know there's a new program to play with we'll use the following command

```
# sudo update-alternatives --install "/usr/bin/java" "java" "/usr/lib/jvm/lib/java" 1
```

And to make it the default runtime for java commands we'll use the next command

```
# sudo update-alternatives --set java /usr/lib/jvm/lib/java
```

Navigate to the /ect folder on the root of your device and open the environment file with LeafPad; add the following lines and save it (of course make a backup before applying changes) logout and log back in for changes to take effect

```
JAVA_HOME="/usr/lib/jvm"
```

Edit your bashrc file with:
nano ~/.bashrc

And add the following two lines to the tail end and save and exit:
export JAVA_HOME="/usr/lib/jvm"
export PATH=$PATH:$JAVA_HOME/bin

7:4:4 Alternate way on installing Java

this method works for jMonkey with ubuntu on the epic 4g

7:4:4.1 Install openjdk
use the following command to find installable openjdk 6 java runtime and development tool kit packages
# apt-cache search openjdk

7:4:4.2 Use the following command to install most of what java programs will require to run on android

# sudo apt-get install openjdk-6-source openjdk-6-jre openjdk-6-jre-lib openjdk-6-jdk openjdk-6-dbg icedtea6-plugin

your system will need to download 160 MB of archives so sit back because after that it's going to unpack everything and that is going to take up a 360 MB chunk of you storage space.

Note:the other packages that it suggest to install other than "ttj-*" files are:
liblcms-utils pulseaudio openjdk-6-demo visualvm libnss-mdns sun-java6-fonts

Note the other things it suggest to install to that are "ttj-*" files are:
ttf-baekmuk ttf-unfonts ttf-unfonts-core ttf-sazanami-gothic ttf-kochi-gothic ttf-sazanami-mincho ttf-wqy-microhei ttf-wqy-zenhei ttf-indic-fonts

Note: you may use 'apt-cache search [package_name]' to quickly find a description of each of the packages in order to find out what they are before install.

Install jMonkey

7:4:4:2 Solved: Installing jMonkey on Android

7:4:4:2.1. Download jMonkey from their website. For Linux you'll want the one ending in '*.sh' and then transfer it to the linux os partition or file structure so you have full read/write permissions over it from the linux side of things http://jmonkeyengine.org/downloads/?did=2

7:4:4:2.2. Open a file browser in your prefered RDP GUI application and navigate to the 'jME3_SDK_3.ORC2-linux.sh' file; for me I put it in '/home/[user-name]/Downloads

7:4:4:2.3. Right click the '*.sh' file and click 'Properties'

7:4:4:2.4. In the 'File Properties' window that pops up, after a sec, click on the 'Permissions' tab and mouse over to the box that has the words 'Make the file executable' and click the box so that it shows a "check mark" instead of a "minus sign or dash"

7:4:4:2.5. Click the 'Ok' button on the 'File Properties' window and open a new terminal window (it's best not to use connectbot ssh or a terminal emulator for running this command as it will need a GUI in a second to load in) and put in the following commands, adjusting the file path to yours would be a good idea.

```
# sh /home/[user-name]/Downloads/jME3_SDK_3.ORC2-linux.sh
```

or

```
# sudo sh /home/[user-name]/Downloads/jME3_SDK_3.ORC2-linux.sh
```

it's going to hang out and do nothing for a minute right around line 'Configuring the installer...' and again at line 'Extracting installation data...' but will eventually start a GUI installation window with a red loading bar and stuff or it'll throw-up a message like...

...(authors reminder to self ?/home/[UN]/Downloads/jmonkeyUserDump)...

If so then:

7:4:4:2.6. Assuming all is well and slowly moving you'll be presented with the jMonkey installation wizard and you can run through the process of selecting where java runtime files can be found and where to extract the jMonkey collection of development tools. It's going to take a minute or sixty but, if you sit back and let it do it's thing without much else going on on the device, jMonkey will eventually install successfully.

7:4:4:2.7. It's not over just yet, once jMonkey is unpacked, installed, knows where "openjdk" or "Java JDK" is (I successfully tested this with openjdk-jdk and it's family of openjdk packages installed on the device; using an 'apt-get install' command' before running the jMonkey installer) and opens for the first time; you need to mouse over to 'help' at the top of the jMonkey window and mouse over to 'check for updates' and click it. This step is the one that everyone forgets in their excitement and one of the top reasons for people to search for error fixes right after installing jMonkey. The author on the main page of jMonkey's install help guide states these steps but there be a lot of words there and it's right near the top of the text block so I and many others have missed it.

7:4:4:2.8. After updating jMonkey though the help drop down menu you are ready to rock and roll. Try mousing over to 'File' click it and mouse down to 'new project' click it and click 'Next' and ‘Finish’ to keep default settings, wait a second or two, and start making stuff that can then be exported out to formats that are compatible on just about any device and any os you may want to write a game for.

7:4:4:2.9. For help with this specific program “jMonkey” you should direct your attention to:
http://jmonkeyengine.org/forum/
and
http://en.wikipedia.org/wiki/JMonkey_Engine
and
http://jmonkeyengine.org/wiki/doku.php/jme3
and for help with getting jMonkey to run on android with linux dual booted; then direct your attention here on this document and it’s mirrors.
You can comment on the program itself here too but, as I’m not the developer of the program, my knowledge of what make’s it work and not work is very trial and error at times. My primary concern is making it easy to install on your device so that developers on that site have another method/platform to do developing with.

7:4:4:2.10 So now you've got jMonkey on Android or Raspberry Pi; now what?

Well start building something of course.
The following link will take you to steps on outputting your project into something that runs on Android (an App)
http://hub.jmonkeyengine.org/wiki/doku.php/jme3:android
This next one is for downloading the Android SDK if you choose to use jMonkey for developing apps for Android (I'll be scripting installing it up shortly and the installer script linked in near the beguinning of the guide will be updated automaticly)
http://developer.android.com/sdk/index.html
http://incise.org/android-development-on-the-command-line.html

This link is for general deployment.
http://hub.jmonkeyengine.org/wiki/doku.php/sdk:application_deployment

~~~~~~~~~
Next up?... Syncing files between your android, dual booted Linux distro, And your home PC. Why? Because being able to sync between all your devices seamlessly will allow quicker development for these systems on the whole.
~~~~~~~~~