LineageOS Wiki

Building for Emulator/AVD

TABLE OF CONTENTS

Introduction

In case you don't have an officially supported device, don't want to test changes on your daily driver, or are just someone who wants to test apps with LineageOS-specific features, we've still got you covered.

These instructions will help you build an emulator-compatible version of LineageOS, ready to run on your computer. If you want to use Android Studio/AVD there are also instructions for packing up/installing your custom build instead of the default AOSP images that Google provides.

What you'll need

- A relatively recent 64-bit computer:
  - Linux, macOS, or Windows - these instructions are only tested using Ubuntu LTS, so we recommend going with that.
  - A reasonable amount of RAM (16 GB to build up to `lineage-17.1`, 32 GB or more for `lineage-18.1` and up). The less RAM you have, the longer the build will take. Enabling ZRAM can be helpful.
  - A reasonable amount of Storage (200 GB to build up to `lineage-17.1`, 300 GB for `lineage-18.1` and up). You might require more free space for enabling `ccache` or building for multiple devices. Using SSDs results in considerably faster build times than traditional hard drives.
- A decent internet connection and reliable electricity. :)
- Some familiarity with basic Android operation and terminology. It may be useful to know some basic command line concepts such as `cd`, which stands for "change directory", the concept of directory hierarchies, and that in Linux they are separated by `/`, etc.

✓ **TIP:** If you are not accustomed to using Linux, this is an excellent chance to learn. It's free – just download and run a virtual machine (VM) such as VirtualBox, then install a Linux distribution such as Ubuntu (AOSP vets Ubuntu as well). Any recent 64-bit version should work great, but the latest Long Term Support (LTS) version is recommended. There are plenty of instructions on setting up VirtualBox to run Ubuntu, so we'll leave that to you. Though it is worth noting, if you already use either a Linux distro or macOS, you can just proceed.

Let's begin!

LineageOS Wiki

NOTE: You only need to do these steps once. If you have already prepared your build environment and downloaded the source code, skip to Start the build

Install the platform-tools

If you haven't previously installed `adb` and `fastboot`, you can download them from Google. Extract it running:

```
unzip platform-tools-latest-linux.zip -d ~
```

TIP: The file may not be named identically to what stands in this command, so adjust accordingly.

Now you have to add `adb` and `fastboot` to your PATH. Open `~/.profile` and add the following:

```
# add Android SDK platform tools to path
if [ -d "$HOME/platform-tools" ] ; then
    PATH="$HOME/platform-tools:$PATH"
fi
```

Then, run `source ~/.profile` to update your environment.

Install the build packages

Several packages are needed to build LineageOS. You can install these using your distribution's package manager.

TIP: A package manager in Linux is a system used to install or remove software (usually originating from the Internet) on your computer. With Ubuntu, you can use the Ubuntu Software Center. Even better, you may also use the `apt-get install` command directly in the Terminal.

To build LineageOS, you'll need:

- `bc bison build-essential ccache curl flex g++-multilib gcc-multilib git gnupg gperf imagemagick lib32ncurses5-dev lib32readline-dev lib32z1-dev libelf-dev liblz4-tool libncurses5 libncurses5-dev libsdl1.2-dev libssl-dev libxml2 libxml2-utils lzop pngcrush rsync schedtool squashfs-tools xsltproc zip zlib1g-dev`

For Ubuntu versions older than 20.04 (focal), install also:

- `libwxgtk3.0-dev`

While for Ubuntu versions older than 16.04 (xenial), install:

- `libwxgtk2.8-dev`

Java

Different versions of LineageOS require different JDK (Java Development Kit) versions.

- LineageOS 18.1+: OpenJDK 11 (included in source download)
- LineageOS 16.0-17.1: OpenJDK 1.9 (included in source download)
- LineageOS 14.1-15.1: OpenJDK 1.8 (install `openjdk-8-jdk`)
- LineageOS 11.0-13.0: OpenJDK 1.7 (install `openjdk-7-jdk`)*

* Ubuntu 16.04 and newer do not have OpenJDK 1.7 in the standard package repositories. See the *Ask Ubuntu* question "How do I install openjdk 7 on Ubuntu 16.04 or higher?". Note that the suggestion to use PPA openjdk-r is outdated (the PPA has never updated their offering of openjdk-7-jdk, so it lacks security fixes); skip that answer even if it is the most upvoted.

Python

manually or creating a virtualenv for it. We recommend the latter:

Generate the virtualenv once using `virtualenv --python=python2 ~/.lineage_venv`. Afterwards, activate it in each terminal where you need `python2` as default by running `~/.lineage_venv/bin/activate`.

The path `~/.lineage_venv` can be chosen freely, this is just an example!

Create the directories

You'll need to set up some directories in your build environment.

To create them:

```
mkdir -p ~/bin
mkdir -p ~/android/lineage
```

The `~/bin` directory will contain the git-repo tool (commonly named "repo") and the `~/android/lineage` directory will contain the source code of LineageOS.

Install the `repo` command

Enter the following to download the `repo` binary and make it executable (runnable):

```
curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
chmod a+x ~/bin/repo
```

Put the `~/bin` directory in your path of execution

In recent versions of Ubuntu, `~/bin` should already be in your PATH. You can check this by opening `~/.profile` with a text editor and verifying the following code exists (add it if it is missing):

```
# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi
```

Then, run `source ~/.profile` to update your environment.

Configure git

Given that `repo` requires you to identify yourself to sync Android, run the following commands to configure your `git` identity:

```
git config --global user.email "you@example.com"
git config --global user.name "Your Name"
```

Turn on caching to speed up build

Make use of ccache if you want to speed up subsequent builds by running:

```
export USE_CCACHE=1
export CCACHE_EXEC=/usr/bin/ccache
```

and adding that line to your `~/.bashrc` file. Then, specify the maximum amount of disk space you want `ccache` to use by typing this:

```
ccache -M 50G
```

where `50G` corresponds to 50GB of cache. This needs to be run once. Anywhere from 25GB-100GB will result in very noticeably increased

occupied on your drive, so take this into consideration.

You can also enable the optional `ccache` compression. While this may involve a slight performance slowdown, it increases the number of files that fit in the cache. To enable it, run:

```
ccache -o compression=true
```

NOTE: If compression is enabled, the `ccache` size can be lower (aim for approximately 20GB for one device).

Initialize the LineageOS source repository

The following branches have been tested for building emulator images:

- lineage-16.0
- lineage-17.1
- lineage-18.1
- lineage-19.1

Enter the following to initialize the repository:

NOTE: Make sure the branch you enter here is the one you wish to build!

```
cd ~/android/lineage
repo init -u https://github.com/LineageOS/android.git -b lineage-20.0
```

Download the source code

To start the download of the source code to your computer, type the following:

```
repo sync
```

The LineageOS manifests include a sensible default configuration for repo, which we strongly suggest you use (i.e. don't add any options to sync). For reference, our default values are `-j 4` and `-c`. The `-j 4` part implies be four simultaneous threads/connections. If you experience problems syncing, you can lower this to `-j 3` or `-j 2`. On the other hand, `-c` makes repo to pull in only the current branch instead of all branches that are available on GitHub.

NOTE: This may take a while, depending on your internet speed. Go and have a beer/coffee/tea/nap in the meantime!

TIP: The `repo sync` command is used to update the latest source code from LineageOS and Google. Remember it, as you may want to do it every few days to keep your code base fresh and up-to-date. But note, if you make any changes, running `repo sync` may wipe them away!

Start the build

Time to start building!

Setup the environment:

```
source build/envsetup.sh
```

Select the build target by running the following command, where `<target>` is one of the entries in the table below:

```
lunch <target>
```

LineageOS Wiki

LineageOS 17 and below			
Phone	Emulator/GSI	`lineage_<arch>-eng`	`arm`, `arm64`, `x86` and `x86_64`
LineageOS 18.1			
Phone	Emulator/GSI	`lineage_<arch>-eng`	`arm`, `arm64`, `x86` and `x86_64`
TV	Emulator/GSI	`lineage_tv_<arch>-eng`	`arm`, `arm64`, `x86` and `x86_64`
Automotive	Emulator/GSI	`lineage_car_<arch>-eng`	`arm64` and `x86_64`
LineageOS 19 and above			
Phone	Emulator	`lineage_sdk_phone_<arch>-eng`	`x86` and `x86_64`
Phone	GSI	`lineage_gsi_<arch>-eng`	`arm`, `arm64`, `x86` and `x86_64`
TV	Emulator	`lineage_sdk_tv_<arch>-eng`	`arm` and `x86`
TV	GSI	`lineage_gsi_tv_<arch>-eng`	`arm`, `arm64`, `x86` and `x86_64`
Automotive	Emulator	`lineage_sdk_car_<arch>-eng`	`arm64` and `x86_64`
Automotive	GSI	`lineage_gsi_car_<arch>-eng`	`arm64` and `x86_64`

For starting, `x86` or `x86_64` is recommended, as your computer can run it natively using hardware acceleration.

Instead of `eng` one can also target `userdebug`, the latter is used by official AOSP emulator images, but ADB and communication with the emulator will need to be enabled first.

Now, build the image:

```
mka
```

Running the emulator

Assuming the build completed without errors, type the following in the terminal window the build ran in:

```
emulator
```

The emulator will fire up and you'll see the LineageOS boot animation. After some time, it will finish booting up and be ready to use.

Success! So… what's next?

You've done it! Welcome to the elite club of self-builders. You've built your operating system from scratch, from the ground up. You are the master/mistress of your domain… and hopefully you've learned a bit on the way and had some fun too.

Exporting for use in Android Studio/AVD

In case you want to run the emulator image independently from the system/terminal you built it in, you are able to export the built image into a format that can be used by Android Studio/AVD. To do that, run the following command in the same terminal that you originally started the build in:

```
mka sdk_addon
```

To deploy the build into your Android Studio installation, move the contained folder (which is named after the architecture that you built for) into a **subfolder** of `/path/to/android/sdk/system-images`. AOSP uses the following path name by default, but you are free to make up your own as well:

`system-images/android-<sdk version>/<tag>/<arch>` (where `<tag>` is one of `default`/`google_apis`/`google_apis_playstore`)

LineageOS emulator builds will use the tag `lineage` by default (visible as "LineageOS" in the images list).

As long as you **haven't** moved the folder directly into `system-images`, the emulator image should now show up in the of the lists of images when creating a new virtual Android device.

To get assistance

- #LineageOS-dev - A helpful, real-time chat room (or "channel"), on the Libera.Chat IRC network.