Source Control Tools

Working with Android code requires using both **Git** (an open-source version-control system) and **Repo** (a Google-built repository-management tool that runs on top of Git). See the Source Control Workflow page for a summary of regular actions you can take, such as uploading changes for review.

[Git]

Git handles large projects that are distributed over multiple repositories. Android uses Git for local operations such as local branching, commits, diffs, and edits. One of the challenges in setting up the Android project was determining how to best support the outside community---from the hobbyist community to the large OEMs building mass-market consumer devices. Google wanted components to be replaceable, and wanted interesting components to have a life of their own, outside of Android. Google first chose a distributed revision control system, then narrowed it down to Git.

For more details on Git, refer to this Git Documentation.

[Repo]

Repo unifies Git repositories when necessary, performs uploads to the Gerrit revision control system, and automates parts of the Android development workflow.

The Repo Launcher provides a Python script that initializes a checkout and downloads the second part, the full Repo tool. The full Repo tool is included in an Android source code checkout. It's located, by default, in `$SRCDIR/.repo/repo/...` and it receives forwarded commands from the downloaded Repo Launcher.

Repo doesn't replace Git, it only makes it easier to work with Git in the context of Android. Repo uses manifest files to aggregate Git projects into the Android superproject. You can put the `repo` command, which is an executable Python script, anywhere in your path. In working with the Android source files, you can use Repo for across-network operations such as with a single Repo working directory.

In most situations, you can use Git instead of Repo, or mix Repo and Git commands to form complex commands. However, using Repo for basic across-network operations makes your work much simpler. For more details on Repo, see the Repo Command Reference, Repo README, the Preupload Hooks (tests) that can be enabled in Repo, and general docs in AOSP.

To download and install the **Repo Launcher** from *git-repo- downloads*, see Installing Repo.

[Gerrit]

Gerrit is a web-based code review system for projects that use Git. Gerrit encourages a more centralized use of Git by allowing all authorized users to submit changes, which are automatically merged if they pass code review. In addition, Gerrit makes reviewing easy, displaying changes side by side in the browser and enabling inline comments.

Find the Android Gerrit review interface at android-review.googlesource.com and the code navigation interface at android.googlesource.com.

[Android Code Search]

Android Code Search allows you to search AOSP without downloading anything. You can use Code Search to view the AOSP source code, switch between open source branches, and navigate cross-references. For more information, see the Google Developers site for the Code Search documentation.

[Other tools]

Android Studio is the official integrated development environment (IDE) for Android app development.

Android Debug Bridge (ADB) lets you connect your development workstation directly to your Android device so you can install packages and evaluate your changes.

For Android 10 and higher, use the IntelliJ with AIDEgen IDE for Android platform development.

[Installing Repo]

Follow these steps to install Repo.

Run these commands to use the official package from your Linux distribution:

```
  sudo apt-get update
  sudo apt-get install repo
```

If those commands didn't work for your system--for example, you see that the package version is outdated, or there isn't an official package available from your Linux distribution, manually install Repo using the following commands:

```
export REPO=$(mktemp /tmp/repo.XXXXXXXXX)
curl -o $ https://storage.googleapis.com/git-repo-downloads/repo
gpg --recv-key 8BB9AD793E8E6153AF0F9A4416530D5E920F5C65
curl -s https://storage.googleapis.com/git-repo-downloads/repo.asc | gpg --verify - $
~/bin/repo
```

These commands set up a temp file, download repo to it, and verify that the key provided matches with the required key. If those are successful, the installation proceeds.

Note: Keys are registered in keys.openpgp.org and work by default for Debian-based distributions. If you can't download the key or if verification fails, consult your distribution's documentation for using Gnu Privacy Guard (GPG). For an additional resource, see the GnuPG website.

After installation, verify that `repo version` reports something similar to the following when it's running in a regular directory. (A regular directory isn't part of a repo client; for example, it's your home directory.)

Run this command:

```
repo version
```

Expect a report similar to this one:

```
<repo not installed>
repo launcher version 2.15
(from /usr/bin/repo)
```

- The `repo launcher version` number reporting as 2.15 or higher indicates a correct version number and proper installation.
- `(from /usr/bin/repo)` indicates installation from a package.
- `(from /home/<>/bin/repo)` indicates manual installation.

[Completing the installation]

Next: To complete your full Repo Tool installation, see Initializing a Repo client, on the Downloading the Source page.