

下载源代码

Android 源代码树位于由 Google 托管的 Git 代码库中。Git 代码库中包含 Android 源代码的元数据，其中包括对源代码进行的更改以及更改时间。本页介绍了如何下载适用于特定 Android 代码流水线的源代码树。

首先要了解特定设备的出厂映像，而非先了解如何下载源代码，请参阅[选择设备编译系统 \(/setup/build/running#selecting-device-build\)](/setup/build/running#selecting-device-build)。

安装 Repo

[Repo \(/setup/develop#repo\)](/setup/develop#repo) 是一款工具，可让您在 Android 环境中更轻松地使用 Git。如需详细了解 Repo，请参阅 [Repo 命令参考文档 \(/setup/develop/repo\)](/setup/develop/repo) 和 [Repo 自述文件 \(https://gerrit.googlesource.com/git-repo/+refs/heads/master/README.md\)](https://gerrit.googlesource.com/git-repo/+refs/heads/master/README.md)。

Repo 分为两部分：第一部分是您安装的启动器脚本，它可以与第二部分（即包含在源代码检出中的完整 Repo 工具）通信。要安装 Repo，请执行以下操作：

1. 确保您的主目录中有一个 `bin/` 目录，并且它包含在您的路径中：

```
mkdir ~/bin
PATH=~/bin:$PATH
```

2. 下载 Repo 启动器，并确保它可执行：

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



对于 1.25 版，Repo 的 SHA-256 校验和为

`d06f33115aea44e583c8669375b35aad397176a411de3461897444d247b6c220`。

对于 1.26 版，Repo 的 SHA-256 校验和为

`0cf5f52bcafb8e1d3ba0271b087312f6117b824af272bedd4ee969d52363a86b`。

初始化 Repo 客户端

安装 Repo 启动器后，设置您的客户端以访问 Android 源代码库：

1. 创建一个空目录来存放您的工作文件。如果您使用的是 MacOS，必须在区分大小写的文件系统中创建该目录。为其提供一个您喜欢的任意名称：

```
mkdir WORKING_DIRECTORY
cd WORKING_DIRECTORY
```

2. 使用您的真实姓名和电子邮件地址配置 Git。要使用 Gerrit 代码审核工具，您需要一个与已注册的 Google 帐号 (<https://www.google.com/accounts>) 相关联的电子邮件地址。确保这是您可以用来接收邮件的有效地址。您在此处提供的姓名将显示在您提交的代码的提供方信息中。

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

3. 运行 `repo init` 以获取最新版本的 Repo 及其最新的问题修复。您必须为清单指定一个网址，该清单用于指定 Android 源代码中包含的各个代码库将位于工作目录中的什么位置。

```
repo init -u https://android.googlesource.com/platform/manifest
```

要检出 `master` 之外的其他分支，请使用 `-b` 指定此分支。要查看分支列表，请参阅[源代码标记和细分版本 \(/setup/start/build-numbers#source-code-tags-and-builds\)](#)。

```
repo init -u https://android.googlesource.com/platform/manifest -b ar
```



初始化成功后，系统将显示一条消息，告诉您 Repo 已在工作目录中完成初始化。您的客户端目录现在应该包含一个 `.repo` 目录，这是清单等文件的存放位置。

下载 Android 源代码树

要将 Android 源代码树从默认清单中指定的代码库下载到工作目录，请运行以下命令：

```
repo sync
```

Android 源代码文件位于工作目录中对应的项目名称下。要加快同步速度，请使用 `-j threadcount` 标记。您也可以考虑添加 `-qc`，从而确保同步过程安静且仅在当前分支进行。如需了解更多详情，请参阅 [Repo 命令参考文档 \(/setup/develop/repo\)](/setup/develop/repo)。

使用身份验证

默认情况下，访问 Android 源代码为匿名操作。为了防止服务器被过度使用，每个 IP 地址都有一个相关联的配额。

当与其他用户共用一个 IP 地址时（例如，在越过 NAT 防火墙访问源代码库时），系统甚至会针对常规使用模式（例如，许多用户在短时间内从同一个 IP 地址同步新客户端）触发配额。

在这种情况下，您可以使用进行身份验证的访问方式，此类访问方式会对每位用户使用单独的配额，而不考虑 IP 地址。

第一步是使用密码生成器 (<https://android.googlesource.com/new-password>) 生成密码，然后按照密码生成器页面中的说明进行操作。

第二步是使用清单 URI `https://android.googlesource.com/a/platform/manifest` 强制进行身份验证访问。请注意 `/a/` 目录前缀如何触发强制性身份验证。您可以通过以下命令将现有客户端转换为使用强制性身份验证：

```
repo init -u https://android.googlesource.com/a/platform/manifest
```

排查网络问题

如果在使用代理的情况下下载内容（在一些企业环境中很常见），您可能需要明确指定 Repo 随后使用的代理：

```
export HTTP_PROXY=http://<proxy_user_id>:<proxy_password>@<proxy_server>:
export HTTPS_PROXY=http://<proxy_user_id>:<proxy_password>@<proxy_server>
```



一种比较少见的情况是，Linux 客户端遇到连接问题，在下载期间（通常是在“正在接收对象”期间）卡住。有人曾报告称，调整 TCP/IP 堆栈的设置并使用非并行命令可以改善这种情况。您需要拥有 root 权限才能修改 TCP 设置：

```
sudo sysctl -w net.ipv4.tcp_window_scaling=0
repo sync -j1
```

使用本地镜像

当您使用多个客户端时（尤其是在带宽不足的情况下），最好为所有服务器内容创建一个本地镜像，并从该镜像同步客户端（不需要访问网络）。一个完整镜像的下载文件比两个客户端的下载文件要小一些，而且包含更多信息。

以下说明假定在 `/usr/local/aosp/mirror` 中创建镜像。首先，创建并同步镜像本身。请注意 `--mirror` 标记，该标记只能在创建新客户端时指定：

```
mkdir -p /usr/local/aosp/mirror
cd /usr/local/aosp/mirror
repo init -u https://android.googlesource.com/mirror/manifest --mirror
repo sync
```

同步镜像后，您就可以从镜像创建新客户端了。请注意，务必要指定一个绝对路径：

```
mkdir -p /usr/local/aosp/master
cd /usr/local/aosp/master
repo init -u /usr/local/aosp/mirror/platform/manifest.git
repo sync
```

最后，要将客户端与服务器同步，请将镜像与服务器同步，然后再将客户端与镜像同步：

```
cd /usr/local/aosp/mirror
repo sync
cd /usr/local/aosp/master
repo sync
```

您可以将镜像存储在 LAN 服务器上，然后通过 NFS、SSH 或 Git 访问它。还可以将其存储在移动存储盘上，并在用户之间或计算机之间传递该存储盘。

验证 Git 标记

将以下公钥加载到您的 GnuPG 密钥数据库中。该密钥用于对代表各版本的带注释标记进行签名。

```
gpg --import
```

复制并粘贴以下密钥，然后输入 EOF (**Ctrl-D**) 以结束输入并处理密钥。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.2.2 (GNU/Linux)

mQGIBEnnWD4RBACt9/h4v9xnnGDou13y3dv0x6/t43LPPIxeJ8eX9WB+8LLuR0SV
lFhpHawsVAcFlmi7f7jdSRF+OvtZL9ShPKdLfwBJMNkU66/TZmPewS4m782ndtw7
8tR1cXb1970b8k0fQB3A9yk2XZ4ei4ZC3i6wVdqHLRxABdncwu5h0F9KXwCgkxMD
u4PVgChaAJzTYJ1EG+UYBIUEAJmfearb0qRAN7dEoff0FeXsEaUA6U90sEoVks0Z
wNj96SA8BL+a10oEUUfpMhiHyLuQSftxisJxTh+2QclzDviDyaTrkANjdYY7p2cq
/HMdOY7LJlHaqtXmZxXjjtw5Uc2QG8UY8aziU3IE9nTjSwCXeJnuyvoizl9/I1S5
jU5SA/9WwIps4SC84ielIXiGWEqq6i6/sk4I9q1YemZF2XVVKnml1F4iCMtNKsR4
MGSa1gA8s4iQbsKNWPgp7M3a51JCVCu6l/8zTpA+uUGapw4tWCp4o0dpIvDPBEa9
b/aF/ygcR8mh5hgUfpF9IpXdkn0sbKCvM9lSSfRciETykZc4wrRCVGhlIEFuZHJv
aWQgT3BlbiBTb3VyY2UgUHJvamVjdCA8aW5pdG1hbC1jb250cmliXDRpb25AYW5k
cm9pZC5jb20+igAEExECACAFaknnWD4CGwMGCwkIBwMCBBUCCAMEFgIDAQIEaQIX
gAAKCRDorT+BmrE0eNr+AJ42Xy6tEW7r3KzrJxnRX8mij9z8tgCdFfQYiHpYngkI
2t09Ed+9Bm4gmE05Ag0ESedYRBAIAKVW1JcMBWvV/0Bo9WiByJ9WJ5swMN36/vAl
QN4mWRhfzD0k/Rosdb0csA0/l8Kz0gKQP0f0btyYjvI8JMC3rmi+LIvSUT9806Up
hisyEmmHv6U8gUb/xHLIanXGxwhYzjgeuAXVCsv+EvoPIHbY4L/KvP5x+oCJIDbk
C2b1TvVk9PryzmE4BPIQL/NtgR1oLWm/uWR9zRUFtBnE411aMAN3qnAHBBMZzKMX
LWBGWE0znfRrnczI5p49i2YZJAjyX1P2WzmScK49CV82dzLo71MnrF6fj+Udtb5+
OgTg7Cow+8PRaTkJEW5Y2JIZpnRUq0CYxAmHYX79EMKHDSThf/8AAwUIAJPwsB/M
```

```
pK+KMs/s3r6nJrnYLTfdZhtmQXimpoDMJg1zxmL8UfNUKiQZ6esoAWtDgpqt7Y7s
KZ8laHRARonte394hidZzM5nb6hQvpPjt20lPRsyqVxw4c/KsjADtAuKW9/d8phb
N8bTy0Jo856qg4o0EzKG9eeF7oaZTYBy33BTL0408sEBxiMior6b8LrZrAhkqDjA
vUXRwm/fFKgps0ysxC6xi553CxBUCH2omNV6Ka1LNMwzSp9ILz8jEGqmUtkBszwo
G1S8fXgE0Lq3cdDM/GJ4QXP/p6LiWNF99faDMTV3+2SA0Gvyt0X6KjKVzK0SsfJQ
hN0DlsIw8hqJc0WISQQYEQIACQUCSedYRAIbDAAKCRDorT+BmrE0eCUOAJ9qmR0l
EXzeoxcdoafxqf6gZlJZlACgkWF7wi2YlW30a+jv2QSTlrx4KLM=
=Wi5D
-----END PGP PUBLIC KEY BLOCK-----
```

导入密钥后，您可以通过以下命令验证任何标记：

```
git tag -v TAG_NAME
```

获取专有二进制文件

您不能只通过纯源代码来使用 AOSP，还需要运行与硬件相关的其他专有库（例如用于硬件图形加速的专有库）。如需其他资源的下载链接和设备二进制文件 (</setup/build/requirements#binaries>)，请参阅以下各部分。

部分设备会将这些专有二进制文件打包到其 **/vendor** 分区。

下载专有二进制文件

对于运行带标记的 AOSP 版本分支的受支持设备，您可以从 Google 的驱动程序 (<https://developers.google.com/android/drivers>) 下载相关的官方二进制文件。有了这些二进制文件，您将有权使用采用非开源代码的其他硬件功能。要编译 AOSP 的 master 分支，请使用 二进制文件预览 (<https://developers.google.com/android/blobs-preview>)。在针对某种设备编译 master 分支时，请使用适用于最新编号版本 (</setup/start/build-numbers>) 的二进制文件或具有最新日期的二进制文件。

解压专有二进制文件

每组二进制文件都是压缩包中的一个自解压脚本。解压每个压缩包，从源代码树的根目录运行附带的自解压脚本，然后确认您同意附带的许可协议的条款。二进制文件及其对应的 `makefile` 将会安装在源代码树的 `vendor/` 层次结构中。

清理

为了确保新安装的二进制文件在解压后会被适当考虑在内，请使用以下命令删除所有以前编译版本的已有输出：

```
make clobber
```

Content and code samples on this page are subject to the licenses described in the [Content License \(/license\)](#).
Java is a registered trademark of Oracle and/or its affiliates.

Last updated 2020-02-05.

