Alpaquita Linux
Getting to know APK

Alpaquita Linux
Revision 1.0
November 2024

bellsoft

Copyright © BellSoft Corporation 2018-2024.

BellSoft software contains open source software. Additional information about third party code is available at https://bell-sw.com/third_party_licenses. You can also get more information on how to get a copy of source code by contacting info@bell-sw.com.

THIS INFORMATION MAY CHANGE WITHOUT NOTICE. TO THE EXTENT PERMITTED BY APPLICABLE LAW, BELLSOFT PROVIDES THIS DOCUMENTATION “AS IS” WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL BELLSOFT BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF BELLSOFT IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in this document is governed by the applicable license agreement, which is not modified in any way by the terms of this notice.

Alpaquita, Liberica and BellSoft are trademarks or registered trademarks of BellSoft Corporation. The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Java and OpenJDK are trademarks or registered trademarks of Oracle and/or its affiliates. Other trademarks are the property of their respective owners and are used only for identification purposes.

Contents

1. Overview

Alpaquita Linux uses the APK Package Manager tool to install, update, upgrade, search, list, and remove packages on a running Linux system. APK is a part of `apk-tools` package, which comes pre-installed on all Alpaquita Linux versions.

All repositories are described in the `/etc/apk/repositories` file that you can edit if necessary. To list available packages, run this command.

```
apk list
```

To list installed packages in your system, add the `-I` option as follows.

```
apk list -I
```

Each line in the list represents a package.

```
liberica11-lite-jdk-11.0.15.1_p2-r0 x86_64 {liberica11-lite} (GPL-2.0 WITH
Classpath-Exception-2.0) [installed]
```

- `liberica11-lite-jre-11.0.15.1_p2-r0 x86_64` - package name
- `{liberica11-lite}` - origin package
- `(GPL-2.0 WITH Classpath-Exception-2.0)` - license type
- `[installed]` - installed in the system

Alpaquita APK differences for Alpine users

Alpaquita Linux APK package manager is similar to the APK in Alpine Linux and contains the following improvements:

- Alpaquita provides optimized libraries that can substitute default ones. If you are installing Alpaquita from an iso image, you can choose to install the optimized library package or keep the default version. See Upgrading special packages for details.
- Alpaquita remote package repositories provide three (3) latest versions of the installed package. You can install the latest version and revert to one of the previous versions if necessary. See Packages and Repositories and Upgrading and downgrading a package for more information.
- APK search results in Alpaquita are sorted alphabetically, and you can search for reverse

dependencies in the package. See Searching for packages for details.

Quick guide on APK for RPM and DEB-based kit users

APK, RPM, and DEB-based tools are designed to maintain programs and components on computers running under different versions of Linux OS.

The following table lists some basic similarities and differences in manipulating packages between APK, RPM, and DEB-based tools.

Action	APK	RPM	DEB
Update package index/database	`apk update`	`yum update`	`apt update`
Search for packages	`apk search <string>`	`yum search <string>`	`apt search <string>`
Install packages	`apk add <package name>`	`yum install <package name>`	`apt install <package name>`
Remove packages	`apk del <package name>`	`yum remove <package name>`	`apt remove <package name>`

The next table lists commands for package version management in APK, RPM, and DEB-based tools.

Action	APK	RPM	DEB
Upgrade	`apk add -u <package name>`	`yum upgrade <package name>`	`apt install <package name>`
Downgrade	`apk add -d <package name>`	`yum downgrade <package name>`	-

Action	APK	RPM	DEB
Install specified version	`apk add <package name>=$version`	`yum install <package name>-$version`	`apt install <package name>=$version`

2. Command Summary

The following table lists some commands to get you started with APK. For more information, run `apk -h` in the command-line interpreter.

Command	Flags and options	Example	Description
`apk search`	`-v` `-r` : searches for reverse dependencies	`apk search -v 'liberica11-lite*'`	Searches through repository files and provides results in alphabetical order.
`apk info`	`-a` : lists all available information	`apk info -a liberica11-lite`	Provides package information.
`apk update`		`sudo apk update`	Updates the package index to the most recent package versions.
`apk add`	`-u` : installs or upgrades packages to the most recent version `-s` : simulates installation	`sudo apk add liberica11-lite`	Installs the specified package and it’s dependencies.
`apk del`		`sudo apk del liberica11-lite`	Removes a package.

3. Packages and Repositories

If you keep the default repository setup while installing Alpaquita Linux, the setup uses the following repositories that contain the most recent package versions.

core: This repository is the main place for different packages that you can install. The repository contains supported packages and tools, such as Liberica JDK and many others.

- https://packages.bell-sw.com/alpaquita/musl/<Alpaquita version>/core - for musl based implementation of the Linux packages
- https://packages.bell-sw.com/alpaquita/glibc/<Alpaquita version>/core - for glibc based implementation of the Linux packages

universe: This repository contains extra packages outside the core repository and hosts packages for various developer tools, services, and other useful Linux utilities.

- https://packages.bell-sw.com/alpaquita/musl/<Alpaquita version>/universe - for musl based implementation of the Linux packages
- https://packages.bell-sw.com/alpaquita/glibc/<Alpaquita version>/universe - for glibc based implementation of the Linux packages

4. Working with Packages

Before you make any changes in your packages, we recommend updating the package index by running the following command. This helps you install the latest package versions.

```
sudo apk update
```

You can add the short version of this command (-U) to another command to ensure you use the latest available version of packages. For example:

```
sudo apk -U add <package name>
```

Searching for packages

You can search for a package using wildcards as in the following example:

```
apk search -v 'liberica11*'
```

Run the following command to search for reverse dependencies of the package:

```
apk search -r <package name>
```

It is important to know the package details and its components to decide whether the package suits your needs. The following command can help you acquire package information including size, dependencies, and so on.

> **Note:**
>
> `-a` flag lists all available information.

```
apk info -a <package name>
```

Installing a package

To install a new package, use this command. It installs the latest version of the selected package.

```
sudo apk add <package name>
```

To downgrade to the previous version of the package, add the `-d` option as follows:

```
sudo apk add -d <package name>
```

To install the specified version of the package, run the following command:

```
sudo apk add <package name>=$version
```

Note:

The `add` command accepts multiple package names separated with spaces.

Installing Java

Java is a popular programming language that helps you run programs on many platforms, including Alpaquita Linux. If you want to create Java programs, you need to install a JDK (Java Development Kit). If you want to run a Java program, you can install a JRE (Java Runtime Environment). If in doubt, install the JDK because it includes all the required components.

Alpaquita installation contains different Liberica JDK versions that you can install either during the installation or later using the following commands.

Check the repository for available JDK versions.

```
apk info -a openjdk
```

Select a version of the JDK required for your environment and applications and install it. Let's assume you want to install a lightweight version of Liberica JDK 11, which is liberica11-lite. Run the following command:

```
sudo apk add liberica11-lite
```

Removing a package

To remove a package, execute the following command:

```
sudo apk del <package name>
```

Note:

The `del` command accepts multiple package names separated with spaces.

Upgrading and downgrading a package

Alpaquita APK package manager helps you maintain all installed packages up-to-date by upgrading to the latest package versions when you run the `apk add -u` command. Alpaquita remote package repositories contain the three latest versions of packages (where available) and you can upgrade and downgrade the packages and kernel.

To upgrade one or more packages to the latest version, run the following command. If you do not specify the package name, all packages are upgraded to the latest available version.

```
apk add -u <package name>
```

For example, let's assume the vim editor is installed on your system and you want to upgrade it by running `$ sudo apk add -u vim`. You get the output similar to the following:

```
(1/4) Installing xxd (9.0.0999-r0)
(2/4) Installing ncurses-terminfo-base (6.3_p20221119-r1)
(3/4) Installing ncurses-libs (6.3_p20221119-r1)
(4/4) Installing vim (9.0.0999-r0)
Executing busybox-1.35.0-r30.trigger
OK: 552 MiB in 186 packages
```

If you want to downgrade the vim editor to the previous version, run `$ sudo apk add -d vim`. You get the output similar to the following:

```
(1/1) Downgrading vim (9.0.0999-r0 -> 9.0.0820-r2)
Executing busybox-1.35.0-r30.trigger
OK: 552 MiB in 186 packages
```

Upgrading special packages

Optimized musl

Optimized musl implementation is already included in the installation image. It is a part of the core repository and you can choose it while installing Alpaquita or install it later by running the `apk add musl-perf` command. In case you need to use the stock version of musl, it can be installed to replace the optimized musl version.

Let's assume you have the default version of musl installed on your system and you want to upgrade to the BellSoft optimized version of musl-perf. In the following example we check the installed version of musl, install the optimized version, and check the installed version of musl again.

Note:

The following examples are based on the Alpaquita Linux Stream version.

```
$ apk info | grep musl
musl-utils
musl-default
musl-default-dev

$ sudo apk add musl-perf
fetch https://packages.bell-
sw.com/alpaquita/musl/stream/core/x86_64/APKINDEX.tar.gz
fetch
https://packages.bellsw.com/alpaquita/musl/stream/universe/x86_64/APKINDEX.tar.
gz
(1/4) Installing musl-perf (1.2.3-r6)
(2/4) Installing musl-perf-dev (1.2.3-r6)
(3/4) Purging musl-default-dev (1.2.3-r3)
(4/4) Purging musl-default (1.2.3-r3)
OK: 552 MiB in 186 packages

$ apk info | grep musl
musl-utils
musl-perf
musl-perf-dev
```

To install the default musl version, run the following command:

```
sudo apk add musl-default
```

Using the simulation

A handy option when installing, upgrading, or removing packages is the simulation `-s` flag. This option helps you simulate the action before actually performing it to ensure that only the necessary packages are altered. For example, if you need to exclude some packages from upgrading, run the `apk add -u` command with the `-s` option and notice the packages you do not want to upgrade. Run the `apk add -u -s` command specifying only the necessary packages again and check that the list does not contain the excluded packages.

Alpaquita Linux

Getting to know APK

bell/soft