Treble

The Android 8.0 release includes Project Treble, a major re-architect of the Android OS framework designed to make it easier, faster, and less costly for manufacturers to update devices to a new version of Android. Treble is for all new devices launching with Android 8.0 and beyond (the new architecture is already running on the Developer Preview for Pixel phones).

About Android updates

Project Treble separates the vendor implementation (device-specific, lower-level software written by silicon manufacturers) from the Android OS framework via a new vendor interface.

In Android 7.x and earlier, no formal vendor interface exists so device makers must update large portions of the Android code to move a device to a newer version of Android:

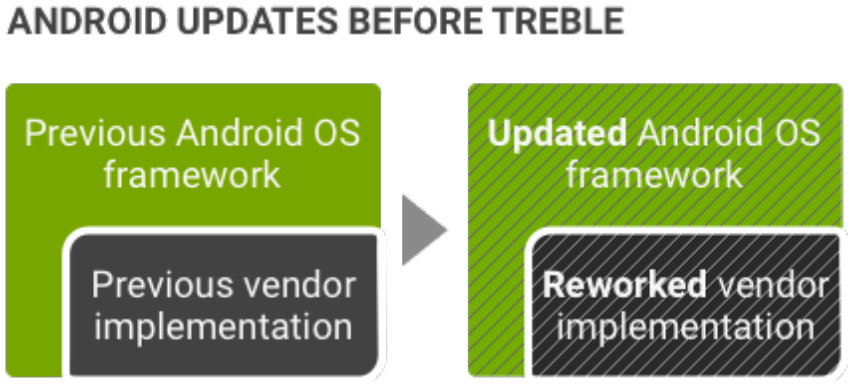

Figure 1. Pre-Treble Android update environment

With Treble, a new stable vendor interface provides access to the hardware-specific parts of Android, enabling device makers to deliver new Android releases simply by updating the Android OS framework—without any additional work required from the silicon manufacturers:

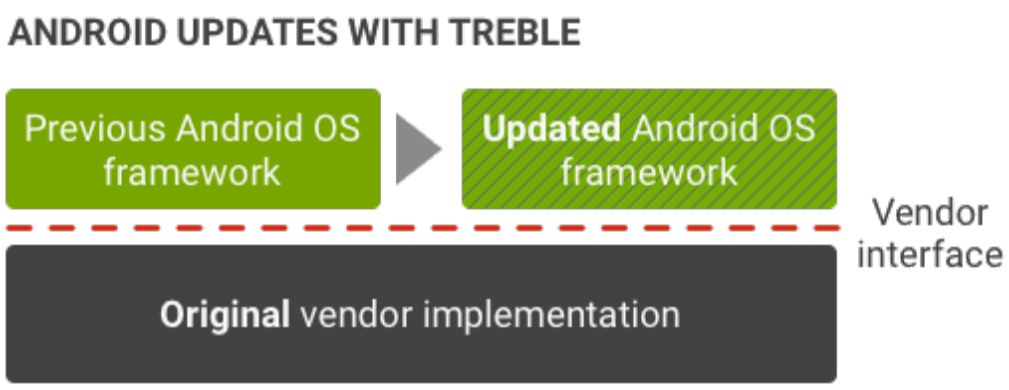

Figure 2. Treble Android update environment

Testing Treble

To ensure forward compatibility of vendor implementations, the new vendor interface is validated by the Vendor Test Suite (VTS) (https://source.android.com/devices/tech/vts/index.html), which is analogous to the Compatibility Test Suite (CTS) (https://source.android.com/compatibility/cts/). You can use VTS to automate HAL and OS kernel testing in both pre-Treble and Treble environments.

Treble resources

For details on the new Treble architecture, see the following sections:

- HAL Types (https://source.android.com/devices/architecture/hal-types.html). Describes binderized, passthrough, Same-Process (SP), and legacy HALs.
- HIDL (General) (https://source.android.com/devices/architecture/hidl/index.html). Contains general information about the HAL interface definition language (HIDL, pronounced "hide-l"), which is an interface description language (IDL) to specify the interface between a HAL and its users.
- HIDL (C++) (https://source.android.com/devices/architecture/hidl-cpp/index.html). Contains details for creating C++ implementations of HIDL interfaces.

- HIDL (Java) (https://source.android.com/devices/architecture/hidl-java/index.html). Contains details about the Java frontend for HIDL interfaces.
- ConfigStore HAL (https://source.android.com/devices/architecture/configstore/index.html). Describes the ConfigStore HAL, which provides a set of APIs for accessing read-only configuration items used to configure the Android framework.
- Device Tree Overlays (https://source.android.com/devices/architecture/dto/index.html). Provides details on using device tree overlays (DTOs) in Android.
- Vendor Native Development Kit (VNDK) (https://source.android.com/devices/architecture/vndk/index.html). Describes the VNDK, which is a set of libraries exclusively for vendors to implement their HALs.
- Vendor Interface Object (VINTF) (https://source.android.com/devices/architecture/vintf/index.html). VINTF objects aggregate relevant information about a device and make that information available through a queryable API.
- SELinux for Android 8.0 (https://source.android.com/security/selinux/images/SELinux_Treble.pdf). Details SELinux changes and customizations.

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 3.0 License (http://creativecommons.org/licenses/by/3.0/)*, and code samples are licensed under the Apache 2.0 License* (http://www.apache.org/licenses/LICENSE-2.0)*. For details, see our Site Policies* (https://developers.google.com/terms/site-policies)*. Java is a registered trademark of Oracle and/or its affiliates.*

Last updated August 21, 2017.