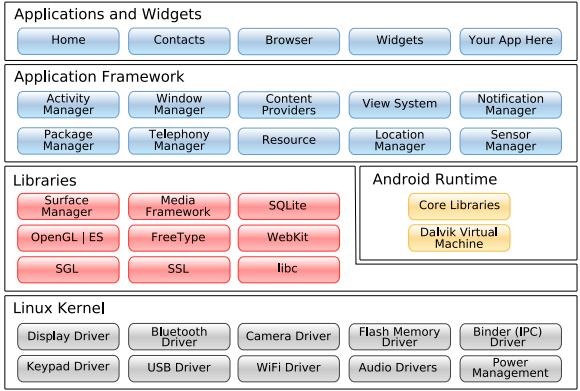
**Software Environment**

**Android** is a [Linux](http://en.wikipedia.org/wiki/Linux)-based [operating system for mobile devices](http://en.wikipedia.org/wiki/Mobile_operating_system) such as [smartphones](http://en.wikipedia.org/wiki/Smartphone) and [tablet computers](http://en.wikipedia.org/wiki/Tablet_computer). It is developed by the [Open Handset Alliance](http://en.wikipedia.org/wiki/Open_Handset_Alliance) led by [Google](http://en.wikipedia.org/wiki/Google) . Android has a large community of developers writing applications ("[apps](http://en.wikipedia.org/wiki/Mobile_apps)") that extend the functionality of the devices. Developers write primarily in a customized version of [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29).

## Android architecture

Android consists of a [kernel](http://en.wikipedia.org/wiki/Kernel_%28software%29) based on the [Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel), with [middleware](http://en.wikipedia.org/wiki/Middleware), [libraries](http://en.wikipedia.org/wiki/Software_library) and [APIs](http://en.wikipedia.org/wiki/Application_programming_interface) written in [C](http://en.wikipedia.org/wiki/C_%28programming_language%29) and [application software](http://en.wikipedia.org/wiki/Application_software) running on an [application framework](http://en.wikipedia.org/wiki/Application_framework) which includes Java-compatible libraries based on [Apache Harmony](http://en.wikipedia.org/wiki/Apache_Harmony). Android uses the [Dalvik virtual machine](http://en.wikipedia.org/wiki/Dalvik_virtual_machine) with [just-in-time compilation](http://en.wikipedia.org/wiki/Just-in-time_compilation) to run Dalvik dex-code (Dalvik Executable), which is usually translated from [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29) bytecode.

The main hardware platform for Android is the [ARM architecture](http://en.wikipedia.org/wiki/ARM_architecture). There is support for [x86](http://en.wikipedia.org/wiki/X86) from the [Android x86](http://en.wikipedia.org/wiki/Android_x86) project, and Google TV uses a special x86 version of Android.



Android's kernel is based on the [Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel) and has further architecture changes by Google outside the typical Linux kernel development cycle.Android does not have a native [X Window System](http://en.wikipedia.org/wiki/X_Window_System) nor does it support the full set of standard [GNU](http://en.wikipedia.org/wiki/GNU) libraries, and this makes it difficult to port existing Linux applications or libraries to Android.

Certain features that Google contributed back to the Linux kernel, notably a power management feature called wakelocks, were rejected by mainline kernel developers, partly because kernel maintainers felt that Google did not show any intent to maintain their own code.Even though Google announced in April 2010 that they would hire two employees to work with the Linux kernel community,[Greg Kroah-Hartman](http://en.wikipedia.org/wiki/Greg_Kroah-Hartman), the current Linux kernel maintainer for the -stable branch, said in December 2010 that he was concerned that Google was no longer trying to get their code changes included in mainstream Linux.Some Google Android developers hinted that "the Android team was getting fed up with the process", because they were a small team and had more urgent work to do on Android.

However, in September 2010, Linux kernel developer Rafael J. Wysocki added a patch that improved the mainline Linux wakeup events framework. He said that Android device drivers that use wakelocks can now be easily merged into mainline Linux, but that Android's opportunistic suspend features should not be included in the mainline kernel. In 2011 [Linus Torvalds](http://en.wikipedia.org/wiki/Linus_Torvalds) said that "eventually Android and Linux would come back to a common kernel, but it will probably not be for four to five years".

In December 2011, [Greg Kroah-Hartman](http://en.wikipedia.org/wiki/Greg_Kroah-Hartman) announced the start of the Android Mainlining Project, which aims to put some Android [drivers](http://en.wikipedia.org/wiki/Device_driver), patches and features back into the Linux kernel, starting in Linux 3.3. further integration being expected for Linux Kernel 3.4.

**Application Framework:**

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including:

* A rich and extensible set of [Views](http://developer.android.com/resources/tutorials/views/index.html) that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser
* [Content Providers](http://developer.android.com/guide/topics/providers/content-providers.html) that enable applications to access data from other applications (such as Contacts), or to share their own data
* A [Resource Manager](http://developer.android.com/guide/topics/resources/resources-i18n.html), providing access to non-code resources such as localized strings, graphics, and layout files
* A [Notification Manager](http://developer.android.com/reference/android/app/NotificationManager.html) that enables all applications to display custom alerts in the status bar
* An [Activity Manager](http://developer.android.com/reference/android/app/Activity.html) that manages the lifecycle of applications and provides a common navigation backstack

**Libraries:**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

* **System C library** - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
* **Media Libraries** - based on PacketVideo's OpenCORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
* **Surface Manager** - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
* **LibWebCore** - a modern web browser engine which powers both the Android browser and an embeddable web view
* **SGL** - the underlying 2D graphics engine
* **3D libraries** - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
* **FreeType** - bitmap and vector font rendering
* **SQLite** - a powerful and lightweight relational database engine available to all applications

## **Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language.

Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool.

The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

## **Linux Kernel**

Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

# Android SDK

### **You should update to the latest tools or platform using the Android SDK and AVD Manager, rather than downloading a new SDK starter package.**

http://dl.google.com/android/android-sdk\_r16-macosx.zip

Here's an overview of the steps you must follow to set up the Android SDK:

1. Prepare your development computer and ensure it meets the system requirements.
2. Install the SDK starter package from the table above. (If you're on Windows, download the installer for help with the initial setup.)
3. Install the ADT Plugin for Eclipse (if you'll be developing in Eclipse).
4. Add Android platforms and other components to your SDK.
5. Explore the contents of the Android SDK (optional).

# System Requirements

Supported Operating Systems

* Windows XP (32-bit), Vista (32- or 64-bit), or Windows 7 (32- or 64-bit)
* Mac OS X 10.5.8 or later (x86 only)
* Linux (tested on Ubuntu Linux, Lucid Lynx)
  + GNU C Library (glibc) 2.7 or later is required.
  + On Ubuntu Linux, version 8.04 or later is required.
  + 64-bit distributions must be capable of running 32-bit applications. For information about how to add support for 32-bit applications

### **Supported Development Environments for Android SDK**

#### **Eclipse IDE**

* Eclipse 3.6 (Helios) or greater

**Note:** Eclipse 3.5 (Galileo) is no longer supported with the latest version of ADT.

* Eclipse [JDT](http://www.eclipse.org/jdt) plugin (included in most Eclipse IDE packages)
* If you need to install or update Eclipse, you can download it from <http://www.eclipse.org/downloads/>.

Several types of Eclipse packages are available for each platform. For developing Android applications, we recommend that you install one of these packages:

* + Eclipse IDE for Java Developers
  + Eclipse Classic
  + Eclipse IDE for Java EE Developers
* [JDK 5 or JDK 6](http://www.oracle.com/technetwork/java/javase/downloads/index.html) (JRE alone is not sufficient)
* [Android Development Tools plugin](http://developer.android.com/sdk/eclipse-adt.html) (recommended)
* **Not** compatible with Gnu Compiler for Java (gcj)

#### **Other development environments or IDEs**

* [JDK 5 or JDK 6](http://www.oracle.com/technetwork/java/javase/downloads/index.html) (JRE alone is not sufficient)
* [Apache Ant](http://ant.apache.org/) 1.8 or later
* **Not** compatible with Gnu Compiler for Java (gcj)

### **Hardware requirements for Android SDK**

The Android SDK requires disk storage for all of the components that you choose to install. The table below provides a rough idea of the disk-space requirements to expect, based on the components that you plan to use.

|  |  |  |
| --- | --- | --- |
| **Component type** | **Approximate size** | **Comments** |
| SDK Tools | 35 MB | Required. |
| SDK Platform-tools | 6 MB | Required. |
| Android platform (each) | 150 MB | At least one platform is required. |
| SDK Add-on (each) | 100 MB | Optional. |
| USB Driver for Windows | 10 MB | Optional. For Windows only. |
| Samples (per platform) | 10M | Optional. |
| Offline documentation | 250 MB | Optional. |

Note that the disk-space requirements above are *in addition to* those of the Eclipse IDE, JDK, or other prerequisite tools that you may need to install on your development computer.

# Installing the SDK

## Step 1. Preparing Your Development Computer

Before getting started with the Android SDK, take a moment to confirm that your development computer meets the [System Requirements](http://developer.android.com/sdk/requirements.html). In particular, you might need to install the [JDK](http://java.sun.com/javase/downloads/index.jsp), if you don't have it already.

If you will be developing in Eclipse with the Android Development Tools (ADT) Plugin—the recommended path if you are new to Android—make sure that you have a suitable version of Eclipse installed on your computer as described in the [System Requirements](http://developer.android.com/sdk/requirements.html) document.

If you need to install Eclipse, you can download it from this location:

<http://www.eclipse.org/downloads/>

The "Eclipse Classic" version is recommended. Otherwise, a Java or RCP version of Eclipse is recommended.

## Step 2. Downloading the SDK Starter Package

The SDK starter package is not a full development environment—it includes only the core SDK Tools, which you can use to download the rest of the SDK components (such as the latest Android platform).

If you haven't already, get the latest version of the SDK starter package from the [SDK download page](http://developer.android.com/sdk/index.html).

If you downloaded a .zip or .tgz package (instead of the SDK installer), unpack it to a safe location on your machine. By default, the SDK files are unpacked into a directory named android-sdk-<machine-platform>.

If you downloaded the Windows installer (.exe file), run it now and it will check whether the proper Java SE Development Kit (JDK) is installed (installing it, if necessary), then install the SDK Tools into a default location (which you can modify).

Make a note of the name and location of the SDK directory on your system—you will need to refer to the SDK directory later, when setting up the ADT plugin and when using the SDK tools from the command line.

## Step 3. Installing the ADT Plugin for Eclipse

Android offers a custom plugin for the Eclipse IDE, called Android Development Tools (ADT), that is designed to give you a powerful, integrated environment in which to build Android applications. It extends the capabilites of Eclipse to let you quickly set up new Android projects, create an application UI, debug your applications using the Android SDK tools, and even export signed (or unsigned) APKs in order to distribute your application. In general, developing in Eclipse with ADT is a highly recommended approach and is the fastest way to get started with Android.

If you'd like to use ADT for developing Android applications, install it now. Read [Installing the ADT Plugin](http://developer.android.com/sdk/eclipse-adt.html#installing) for step-by-step installation instructions, then return here to continue the last step in setting up your Android SDK.

If you prefer to work in a different IDE, you do not need to install Eclipse or ADT. Instead, you can directly use the SDK tools to build and debug your application. The [Introduction](http://developer.android.com/guide/developing/index.html) to Android application development outlines the major steps that you need to complete when developing in Eclipse or other IDEs.

**Step 4. Adding Platforms and Other Components**

The last step in setting up your SDK is using the Android SDK and AVD Manager (a tool included in the SDK starter package) to download essential SDK components into your development environment.

The SDK uses a modular structure that separates the major parts of the SDK—Android platform versions, add-ons, tools, samples, and documentation—into a set of separately installable components. The SDK starter package, which you've already downloaded, includes only a single component: the latest version of the SDK Tools. To develop an Android application, you also need to download at least one Android platform and the associated platform tools. You can add other components and platforms as well, which is highly recommended.

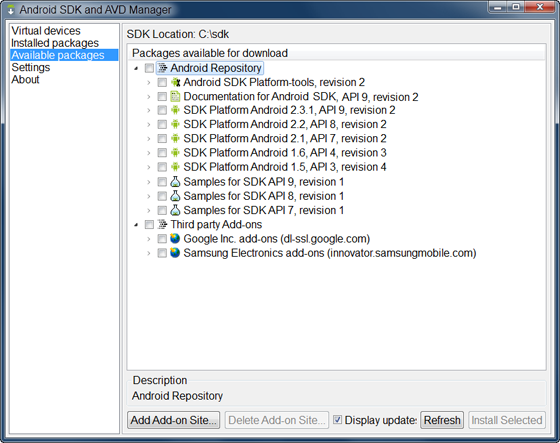
If you used the Windows installer, when you complete the installation wizard, it will launch the Android SDK and AVD Manager with a default set of platforms and other components selected for you to install. Simply click **Install** to accept the recommended set of components and install them. You can then skip to [Step 5](http://developer.android.com/sdk/installing.html#sdkContents), but we recommend you first read the section about the [Available Components](http://developer.android.com/sdk/installing.html#components) to better understand the components available from the Android SDK and AVD Manager.

You can launch the Android SDK and AVD Manager in one of the following ways:

* From within Eclipse, select **Window > Android SDK and AVD Manager**.
* On Windows, double-click the SDK Manager.exe file at the root of the Android SDK directory.
* On Mac or Linux, open a terminal and navigate to the tools/ directory in the Android SDK, then execute:

android

To download components, use the graphical UI of the Android SDK and AVD Manager to browse the SDK repository and select new or updated components (see figure 1). The Android SDK and AVD Manager installs the selected components in your SDK environment. For information about which components you should download, see [Recommended Components](http://developer.android.com/sdk/installing.html#which).



**Figure 1.** The Android SDK and AVD Manager's **Available Packages** panel, which shows the SDK components that are available for you to download into your environment.

**Available Components**

By default, there are two repositories of components for your SDK: *Android Repository* and *Third party Add-ons*.

The *Android Repository* offers these types of components:

* **SDK Tools** — Contains tools for debugging and testing your application and other utility tools. These tools are installed with the Android SDK starter package and receive periodic updates. You can access these tools in the <sdk>/tools/ directory of your SDK. To learn more about them, see [SDK Tools](http://developer.android.com/guide/developing/tools/index.html#tools-sdk) in the developer guide.
* **SDK Platform-tools** — Contains platform-dependent tools for developing and debugging your application. These tools support the latest features of the Android platform and are typically updated only when a new platform becomes available. You can access these tools in the <sdk>/platform-tools/ directory. To learn more about them, see [Platform Tools](http://developer.android.com/guide/developing/tools/index.html#tools-platform) in the developer guide.
* **Android platforms** — An SDK platform is available for every production Android platform deployable to Android-powered devices. Each SDK platform component includes a fully compliant Android library, system image, sample code, and emulator skins. To learn more about a specific platform, see the list of platforms that appears under the section "Downloadable SDK Components" on the left part of this page.
* **USB Driver for Windows** (Windows only) — Contains driver files that you can install on your Windows computer, so that you can run and debug your applications on an actual device. You *do not* need the USB driver unless you plan to debug your application on an actual Android-powered device. If you develop on Mac OS X or Linux, you do not need a special driver to debug your application on an Android-powered device. See [Using Hardware Devices](http://developer.android.com/guide/developing/device.html) for more information about developing on a real device.
* **Samples** — Contains the sample code and apps available for each Android development platform. If you are just getting started with Android development, make sure to download the samples to your SDK.
* **Documentation** — Contains a local copy of the latest multiversion documentation for the Android framework API.

The *Third party Add-ons* provide components that allow you to create a development environment using a specific Android external library (such as the Google Maps library) or a customized (but fully compliant) Android system image. You can add additional Add-on repositories by clicking **Add Add-on Site**.

**Recommended Components**

The SDK repository contains a range of components that you can download. Use the table below to determine which components you need, based on whether you want to set up a basic, recommended, or full development environment:

|  |  |  |
| --- | --- | --- |
| **Environment** | **SDK Component** | **Comments** |
| Basic | SDK Tools | If you've just installed the SDK starter package, then you already have the latest version of this component. The SDK Tools component is required to develop an Android application. Make sure you keep this up to date. |
| SDK Platform-tools | This includes more tools that are required for application development. These tools are platform-dependent and typically update only when a new SDK platform is made available, in order to support new features in the platform. These tools are always backward compatible with older platforms, but you must be sure that you have the latest version of these tools when you install a new SDK platform. |
| SDK platform | You need to download **at least one platform** into your environment, so that you will be able to compile your application and set up an Android Virtual Device (AVD) to run it on (in the emulator). To start with, just download the latest version of the platform. Later, if you plan to publish your application, you will want to download other platforms as well, so that you can test your application on the full range of Android platform versions that your application supports. |
| **+** | |  |
| Recommended (plus Basic) | Documentation | The Documentation component is useful because it lets you work offline and also look up API reference information from inside Eclipse. |
| Samples | The Samples components give you source code that you can use to learn about Android, load as a project and run, or reuse in your own app. Note that multiple samples components are available — one for each Android platform version. When you are choosing a samples component to download, select the one whose API Level matches the API Level of the Android platform that you plan to use. |
| Usb Driver | The Usb Driver component is needed only if you are developing on Windows and have an Android-powered device on which you want to install your application for debugging and testing. For Mac OS X and Linux platforms, no special driver is needed. |
| **+** | |  |
| Full (plus Recommended) | Google APIs | The Google APIs add-on gives your application access to the Maps external library, which makes it easy to display and manipulate Maps data in your application. |
| Additional SDK Platforms | If you plan to publish your application, you will want to download additional platforms corresponding to the Android platform versions on which you want the application to run. The recommended approach is to compile your application against the lowest version you want to support, but test it against higher versions that you intend the application to run on. You can test your applications on different platforms by running in an Android Virtual Device (AVD) on the Android emulator. |

Once you've installed at least the basic configuration of SDK components, you're ready to start developing Android apps. The next section describes the contents of the Android SDK to familiarize you with the components you've just installed.

For more information about using the Android SDK and AVD Manager, see the [Adding SDK Components](http://developer.android.com/sdk/adding-components.html) document.

**Step 5. Exploring the SDK (Optional)**

Once you've installed the SDK and downloaded the platforms, documentation, and add-ons that you need, we suggest that you open the SDK directory and take a look at what's inside.

The table below describes the full SDK directory contents, with components installed.

|  |  |  |
| --- | --- | --- |
| **Name** | | **Description** |
| add-ons/ | | Contains add-ons to the Android SDK development environment, which let you develop against external libraries that are available on some devices. |
| docs/ | | A full set of documentation in HTML format, including the Developer's Guide, API Reference, and other information. To read the documentation, load the file offline.html in a web browser. |
| platform-tools/ | | Contains platform-dependent development tools that may be updated with each platform release. The platform tools include the Android Debug Bridge (adb) as well as other tools that you don't typically use directly. These tools are separate from the development tools in the tools/ directory because these tools may be updated in order to support new features in the latest Android platform. |
| platforms/ | | Contains a set of Android platform versions that you can develop applications against, each in a separate directory. |
|  | *<platform>*/ | Platform version directory, for example "android-11". All platform version directories contain a similar set of files and subdirectory structure. Each platform directory also includes the Android library (android.jar) that is used to compile applications against the platform version. |
| samples/ | | Sample code and apps that are specific to platform version. |
| tools/ | | Contains the set of development and profiling tools that are platform-independent, such as the emulator, the Android SDK and AVD Manager, ddms, hierarchyviewer and more. The tools in this directory may be updated at any time using the Android SDK and AVD Manager and are independent of platform releases. |
| SDK Readme.txt | | A file that explains how to perform the initial setup of your SDK, including how to launch the Android SDK and AVD Manager tool on all platforms. |
| SDK Manager.exe | | Windows SDK only. A shortcut that launches the Android SDK and AVD Manager tool, which you use to add components to your SDK. |

Optionally, you might want to add the location of the SDK's tools/ and platform-tools to your PATH environment variable, to provide easy access to the tools.

### Android's Features

Handset layouts

The platform is adaptable to larger, [VGA](http://en.wikipedia.org/wiki/Video_Graphics_Array), [2D graphics](http://en.wikipedia.org/wiki/2D_computer_graphics) library, [3D graphics](http://en.wikipedia.org/wiki/3D_computer_graphics) library based on [OpenGL ES](http://en.wikipedia.org/wiki/OpenGL_ES) 2.0 specifications, and traditional smartphone layouts.

Storage

[SQLite](http://en.wikipedia.org/wiki/SQLite), a lightweight [relational database](http://en.wikipedia.org/wiki/Relational_database), is used for [data](http://en.wikipedia.org/wiki/Data) storage purposes.

Connectivity

Android supports connectivity technologies including [GSM](http://en.wikipedia.org/wiki/GSM)/[EDGE](http://en.wikipedia.org/wiki/Enhanced_Data_Rates_for_GSM_Evolution), [IDEN](http://en.wikipedia.org/wiki/Integrated_Digital_Enhanced_Network), [CDMA](http://en.wikipedia.org/wiki/Code_division_multiple_access), [EV-DO](http://en.wikipedia.org/wiki/Evolution-Data_Optimized), [UMTS](http://en.wikipedia.org/wiki/Universal_Mobile_Telecommunications_System), [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth), [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi), [LTE](http://en.wikipedia.org/wiki/LTE_Advanced), [NFC](http://en.wikipedia.org/wiki/Near_field_communication) and [WiMAX](http://en.wikipedia.org/wiki/WiMAX).

Messaging

[SMS](http://en.wikipedia.org/wiki/SMS) and [MMS](http://en.wikipedia.org/wiki/Multimedia_Messaging_Service) are available forms of messaging, including threaded [text messaging](http://en.wikipedia.org/wiki/Text_messaging) and now [Android Cloud To Device Messaging](http://en.wikipedia.org/wiki/Android_Cloud_To_Device_Messaging) (C2DM) is also a part of Android Push Messaging service.

Multiple language support

Android supports multiple languages.

Web browser

The web browser available in Android is based on the open-source [WebKit](http://en.wikipedia.org/wiki/WebKit) layout engine, coupled with [Chrome's](http://en.wikipedia.org/wiki/Google_Chrome) [V8 JavaScript engine](http://en.wikipedia.org/wiki/V8_JavaScript_engine). The browser scores 100/100 on the [Acid3](http://en.wikipedia.org/wiki/Acid3#Mobile_browsers) test on Android 4.0.

Java support

While most Android applications are written in [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29), there is no [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) in the platform and Java byte code is not executed. Java classes are compiled into Dalvik executables and run on [Dalvik](http://en.wikipedia.org/wiki/Dalvik_virtual_machine), a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU. [J2ME](http://en.wikipedia.org/wiki/J2ME) support can be provided via third-party applications.

Media support

Android supports the following audio/video/still media formats: [WebM](http://en.wikipedia.org/wiki/WebM), [H.263](http://en.wikipedia.org/wiki/H.263), [H.264](http://en.wikipedia.org/wiki/H.264) (in [3GP](http://en.wikipedia.org/wiki/3GP) or [MP4](http://en.wikipedia.org/wiki/MP4) [container](http://en.wikipedia.org/wiki/Container_format_%28digital%29)), [MPEG-4 SP](http://en.wikipedia.org/wiki/MPEG-4_Part_2), [AMR](http://en.wikipedia.org/wiki/Adaptive_multi-rate_compression), [AMR-WB](http://en.wikipedia.org/wiki/AMR-WB) (in 3GP container), [AAC](http://en.wikipedia.org/wiki/Advanced_Audio_Coding), [HE-AAC](http://en.wikipedia.org/wiki/HE-AAC) (in MP4 or 3GP container), [MP3](http://en.wikipedia.org/wiki/MP3), [MIDI](http://en.wikipedia.org/wiki/Musical_Instrument_Digital_Interface), [Ogg Vorbis](http://en.wikipedia.org/wiki/Vorbis), [FLAC](http://en.wikipedia.org/wiki/Free_Lossless_Audio_Codec), [WAV](http://en.wikipedia.org/wiki/WAV), [JPEG](http://en.wikipedia.org/wiki/JPEG), [PNG](http://en.wikipedia.org/wiki/Portable_Network_Graphics), [GIF](http://en.wikipedia.org/wiki/Graphics_Interchange_Format), [BMP](http://en.wikipedia.org/wiki/BMP_file_format).[[71]](http://en.wikipedia.org/wiki/Android_%28operating_system%29#cite_note-mediaformats-70)

Streaming media support

RTP/RTSP streaming ([3GPP PSS](http://en.wikipedia.org/w/index.php?title=3GPP_PSS&action=edit&redlink=1), [ISMA](http://en.wikipedia.org/wiki/Internet_Streaming_Media_Alliance)), HTML progressive download ([HTML5 <video> tag](http://en.wikipedia.org/wiki/HTML5_video)). Adobe Flash Streaming (RTMP) and HTTP Dynamic Streaming are supported by the [Flash plugin](http://en.wikipedia.org/wiki/Adobe_Flash_Player#Mobile_platforms). Apple HTTP Live Streaming is supported by [RealPlayer for Android](http://en.wikipedia.org/wiki/RealPlayer_for_Android), and by the operating system in Android 3.0 (Honeycomb).

Additional hardware support

Android can use video/still cameras, [touchscreens](http://en.wikipedia.org/wiki/Touchscreen), [GPS](http://en.wikipedia.org/wiki/Global_Positioning_System), [accelerometers](http://en.wikipedia.org/wiki/Accelerometer), [gyroscopes](http://en.wikipedia.org/wiki/Gyroscope), [barometers](http://en.wikipedia.org/wiki/Barometer), [magnetometers](http://en.wikipedia.org/wiki/Magnetometer), dedicated gaming controls, [proximity](http://en.wikipedia.org/wiki/Proximity_sensor) and [pressure sensors](http://en.wikipedia.org/wiki/Pressure_sensor), [thermometers](http://en.wikipedia.org/wiki/Thermometer), accelerated 2D [bit blits](http://en.wikipedia.org/wiki/Bit_blit) (with hardware orientation, scaling, pixel format conversion) and accelerated 3D graphics.

Multi-touch

Android has native support for [multi-touch](http://en.wikipedia.org/wiki/Multi-touch) which was initially made available in handsets such as the [HTC Hero](http://en.wikipedia.org/wiki/HTC_Hero). The feature was originally disabled at the kernel level (possibly to avoid infringing Apple's patents on touch-screen technology at the time). Google has since released an update for the [Nexus One](http://en.wikipedia.org/wiki/Nexus_One) and the [Motorola Droid](http://en.wikipedia.org/wiki/Motorola_Droid) which enables multi-touch natively.

Bluetooth

Supports [A2DP](http://en.wikipedia.org/wiki/A2DP), [AVRCP](http://en.wikipedia.org/wiki/AVRCP), sending files ([OPP](http://en.wikipedia.org/wiki/Object_Push_Profile)), accessing the phone book ([PBAP](http://en.wikipedia.org/wiki/Bluetooth_profile#Phone_Book_Access_Profile_.28PBAP.2C_PBA.29)), voice dialing and sending contacts between phones. Keyboard, mouse and joystick ([HID](http://en.wikipedia.org/wiki/Bluetooth_profile#Human_Interface_Device_Profile_.28HID.29)) support is available in Android 3.1+, and in earlier versions through manufacturer customizations and third-party applications.

Video calling

Android does not support native video calling, but some handsets have a customized version of the operating system that supports it, either via the [UMTS](http://en.wikipedia.org/wiki/UMTS) network (like the [Samsung Galaxy S](http://en.wikipedia.org/wiki/Samsung_Galaxy_S)) or over IP. Video calling through Google Talk is available in Android 2.3.4 and later. Gingerbread allows [Nexus S](http://en.wikipedia.org/wiki/Nexus_S) to place Internet calls with a SIP account. This allows for enhanced VoIP dialing to other SIP accounts and even phone numbers. Skype 2.1 offers video calling in Android 2.3, including front camera support.

Multitasking

Multitasking of applications is available.

Voice based features

Google search through voice has been available since initial release. Voice actions for calling, texting, navigation, etc. are supported on Android 2.2 onwards.

Tethering

Android supports [tethering](http://en.wikipedia.org/wiki/Tethering), which allows a phone to be used as a wireless/wired [Wi-Fi hotspot](http://en.wikipedia.org/wiki/Wi-Fi_hotspot). Before Android 2.2 this was supported by third-party applications or manufacturer customizations.

Screen capture

Android supports capturing a [screenshot](http://en.wikipedia.org/wiki/Screenshot) by pressing the power and volume-down buttons at the same time. Prior to Android 4.0, the only methods of capturing a screenshot were through manufacturer and third-party customizations or otherwise by using a PC connection (DDMS developer's tool). These alternative methods are still available with the latest Android.

External storage

Most Android devices include microSD slot and can read microSD cards formatted with [FAT32](http://en.wikipedia.org/wiki/FAT32), [Ext3fs](http://en.wikipedia.org/wiki/Ext3fs) or [Ext4fs](http://en.wikipedia.org/wiki/Ext4fs) file system. To allow use of high-capacity storage media such as [USB flash drives](http://en.wikipedia.org/wiki/USB_flash_drive) and [USB HDDs](http://en.wikipedia.org/wiki/USB_HDD), many Android tablets also include [USB](http://en.wikipedia.org/wiki/USB) 'A' receptacle. Storage formatted with [FAT32](http://en.wikipedia.org/wiki/FAT32) is handled by [Linux Kernel](http://en.wikipedia.org/wiki/Linux_Kernel) VFAT driver, while 3rd party solutions are required to handle other popular file systems such as [NTFS](http://en.wikipedia.org/wiki/NTFS), [HFS Plus](http://en.wikipedia.org/wiki/HFS%2B) and [exFAT](http://en.wikipedia.org/wiki/ExFAT).

### Version history

Android has seen a number of updates since its original release, each fixing [bugs](http://en.wikipedia.org/wiki/Software_bug) and adding new features. Each version is named, in alphabetical order, after a dessert.

* **2.3 Gingerbread** refined the user interface, improved the soft keyboard and copy/paste features, improved gaming performance, added [SIP](http://en.wikipedia.org/wiki/Session_Initiation_Protocol) support ([VoIP](http://en.wikipedia.org/wiki/VoIP) calls), and added support for [Near Field Communication](http://en.wikipedia.org/wiki/Near_Field_Communication).
* **3.0 Honeycomb** was a tablet-oriented release which supports larger screen devices and introduces many new user interface features, support for [multi-core processors](http://en.wikipedia.org/wiki/Multi-core_processor), hardware acceleration for graphics and full system encryption. The first device featuring this version, the [Motorola Xoom](http://en.wikipedia.org/wiki/Motorola_Xoom) tablet, went on sale in February 2011.
* **3.1 Honeycomb**, released in May 2011, added support for extra input devices, USB host mode for transferring information directly from cameras and other devices, and the Google Movies and Books apps.
* **3.2 Honeycomb**, released in July 2011, added optimization for a broader range of screen sizes, new "zoom-to-fill" screen compatibility mode, loading media files directly from SD card, and an extended screen support API. [Huawei](http://en.wikipedia.org/wiki/Huawei) MediaPad is the first 7 inch tablet to use this version.
* **4.0 Ice Cream Sandwich**, announced on October 19, 2011, brought Honeycomb features to smartphones and added new features including facial recognition unlock, network data usage monitoring and control, unified social networking contacts, photography enhancements, offline email searching, app folders, and information sharing using [NFC](http://en.wikipedia.org/wiki/Near_Field_Communication). Android 4.0.3 Ice Cream Sandwich is the latest Android version that is available to phones. The source code of Android 4.0.1 was released on November 14, 2011.