Введение в Android

* [1](https://vk.com/share.php?url=https%3A%2F%2Fskillberg.com%2Fcourses%2Fandroid%2Flessons%2Fvvedenie-v-android%2F&title=%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B2%20Android%20%E2%80%94%20%D0%BA%D1%83%D1%80%D1%81%20%22%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0%20%D0%BF%D0%BE%D0%B4%20Android%22.%20%D0%91%D0%B5%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D0%BD%D1%8B%D0%B5%20%D1%83%D1%80%D0%BE%D0%BA%D0%B8%20%7C%20Skillberg&description=&image=&utm_source=share2)

**Android** — самая популярная в мире мобильная операционная система, разрабатываемая компанией Google. На конец 2017 года её доля составляет около 85%, а всего активировано более 2 миллиардов устройств.

Изначально разработкой Android занималась компания **Android Inc**. Её история началась в далеком 2003 году когда Энди Рубин с ещё тремя товарищами решили создать мобильную операционную систему. В 2005 году на них обратила внимание компания Google, и 17 августа 2005 года приобрела Android Inc.

Судя по ранним версиям, Android должен был быть похожим скорее на Blackberry, нежели на то, чем он является сейчас — в те времена никто не считал нужным выпускать сенсорные телефоны.

[](https://skillberg.com/media/uploads/2017/11/17/2e2f676c033447cdae68f66b4e5e3c5d.png)Один из ранних прототипов Android

Однако, в **2007** году Apple совершает революцию на рынке мобильных устройств, выпустив iPhone, и это вынуждает Google пересмотреть свои взгляды на принципы взаимодействия пользователя с Android.

**22 октября 2008** года оператор T-Mobile начинает продажи HTC Dream (он же — T-Mobile G1) — первого смартфона на базе Android, имеющего сенсорный экран.

[](https://skillberg.com/media/uploads/2017/11/17/16f9fefe665040378b0afbfe2c8ba362.png)HTC Dream

HTC Dream стал достаточно успешным телефоном — всего за полгода T-Mobile смог продать в США более миллиона устройств.

В **2009** году Google выпустила целых четыре обновления Android — **1.1** (Banana Bread), **1.5** (Cupcake), **1.6** (Donut) и **2.0** (Eclair), а так же незначительное обновление **2.1**.

В середине **2010** года была представлена версия **2.2** (Froyo), а в конце — **2.3** (Gingerbread). Релиз Froyo был очень важен — именно в нем появилась поддержка JIT (Just-In-Time Compiler), и за счёт этого приложения стали работать существенно быстрее.

**22 февраля 2011** года была выпущена версия **3.0** (Honeycomb), в которой впервые появилась поддержка планшетов. Несмотря на то, что Android — открытая ОС, исходный код этой версии так и не был выпущен официально, хотя в репозиториях Android все же можно найти по тегам немалую часть кода этой версии. Honeycomb был промежуточной версией, работающей только на планшетах — Google готовился к выпуску универсальной версии.

И в конце **2011** года она выходит — **4.0** (Ice Cream Sandwich) стала первой версией, работающей и на смартфонах, и на планшетах. В этом обновлении также появился новый интерфейс Holo, ставший первой попыткой унификации интерфейсов приложений и просуществовавший целых три года, вплоть до выхода 5.0.

В период с **2011** по **2013** годы Android получает обновления до версий **4.1**, **4.2** и **4.3**, не принесшие ничего особо интересного (о том, что это минорные обновления, говорит даже тот факт, что название не менялось).

В **2013** году выходит Android **4.4** KitKat, выпущенный в сотрудничестве с Nestlé. Это был первый случай использования коммерческого названия в ОС Android. На этой же версии была основана первая версия Android для носимых устройств (например, умных часов) — **4.4W**.

В конце **2014** года выходит Android **5.0** Lollipop, одно из самых значимых обновлений Android. Именно в нем Google наконец-то выпускает Material Design. В отличие от Holo, это не просто стиль, а целая философия дизайна, при том прекрасно документированная и имеющая подробнейшние гайдлайны. Теперь простые разработчики могут выпускать красивые приложения своими силами, без помощи дизайнеров!

Чуть позже в этом же году Google выпускает релизную версию **Android Studio** и отказывается от Eclipse — **2014**год стал чуть ли не лучшим годом для Android-разработчиков :)

В августе **2015** года выходит Android **6.0** (Marshmallow), а в августе **2016** — Android **7.0** (Nougat).

**21 августа 2017** года Google выпустил версию **8.0** Oreo — второй случай использования коммерческого бренда в качестве названия ОС.

Технические подробности

Android работает на ядре **Linux**, разработка приложений ведется на языке Java, также возможно использование C/C++. Большая часть исходного кода как самой ОС, так и библиотек (выпущенных Google или сторонними разработчиками) распространяется под лицензией Apache 2.0.

Несмотря на использование Linux и Java, Android — это не просто JVM работающая поверх ядра Linux.

Например, вместо libc здесь используется разработанная специально для мобильных устройств стандартная библиотека языка C — bionic. Зачем Google так поступил? Причин несколько:

* bionic оптимизирован для работы на мобильных устройствах, где CPU существенно слабее "старших братьев".
* bionic существенно компактнее, чем libc
* bionic выпускается под лицензией BSD, более предпочтительной для Google, нежели GPL.

Есть, однако, и недостатки: жесткая оптимизация не позволила реализовать некоторые функции, имеющиеся в libc.

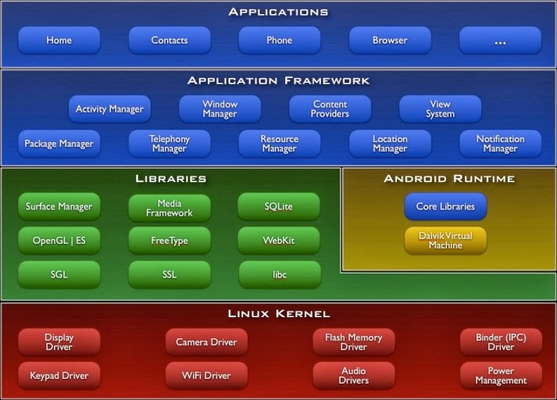
То же самое и с Java: в Android используется виртуальная машина **Dalvik** (а начиная с 5.0 — **ART**) вместо JVM от Oracle. Dalvik, так же как и bionic, был оптимизирован для работы на мобильных устройствах, использует свой собственный байт-код и кардинально отличается от JVM.

Начиная с Android 5.0 (официально и полностью; неофициально и опционально — с 4.4) Dalvik был заменен виртуальной машиной ART (Android RunTime). Он использует тот же байт-код, что и Dalvik, однако работает иначе: Dalvik использует JIT-компиляцию, т.е. компилирует байт-код во время работы приложения, а ART — во время установки. Таким образом, работа по оптимизации производительности производится заранее, что положительно сказывается на скорости работы приложений (и ресурсе аккумулятора).

В Android 7.0 был представлен JIT-компилятор для ART, который запускается во время работы приложения, и позволяет постоянно его оптимизировать.

Архитектура Android

Давайте рассмотрим компоненты операционной системы, с которой мы будем работать.

[](https://skillberg.com/media/uploads/2017/11/17/cba4369217634a5d939fc531bbc7d59b.png)Архитектура Android

Ядро Linux

Собственно, "сердце" системы. В ядре реализована работа с CPU, памятью, процессами; драйвера для взаимодействия с "железом".

В процессе переноса на Android ядро Linux получило огромное количество патчей. Изменениям подверглись многие компоненты, например:

* Стандартный IPC (InterProcess Communication), механизм связи между приложениями, был заменен на оптимизированный для мобильных устройств Binder.
* Подверглись изменениям механизмы работы с памятью.
* Добавлена поддержка файловой системы YAFFS2 (ФС для флеш-накопителей).
* И многое другое

Библиотеки

На этом уровне находятся ключевые библиотеки. Все они реализованы на C/C++, так как требуют максимальной производительности.

* **libc** (bionic, если точнее) — наш старый знакомый из прошлого раздела. Стандартная библиотека языка C.
* **SSL** — реализация криптографических функций.
* **SGL** (Skia Graphics Library) — графический движок.
* **OpenGL ES** — облегченная реализация OpenGL, мощной библиотеки для работы с 2D/3D графикой.
* **FreeType** — библиотека для работы со шрифтами.
* **WebKit** — браузерный движок. Все, что отображает HTML в Android (и, кстати, iOS), работает на WebKit.
* **SQLite** — встраиваемая база данных. С ней мы еще встретимся :)
* **Media Framework** — набор библиотек для работы с мультимедиа.
* **Surface Manager** — ответственен за отрисовку того, что мы видим на экране. Совмещает команды рисования от различных окон в одну "картинку".

Android Runtime

На этом же уровне находится среда выполнения приложений. В нее входят Core Libraries и, собственно, виртуальная машина — Dalvik (ART в Android 5.0+).

Ключевой особенностью Android является то, что каждое приложение работает в так называемой "песочнице", то есть изолировано от остальных. Когда происходит установка приложения, создается новый пользователь и директория для хранения данных приложения (/data/data/<package\_name>). Владельцем этой директории является созданный пользователь, и ни одно другое приложение не имеет доступа к файлам внутри нее. Соответственно, приложение тоже запускается из-под этого пользователя и в своей копии виртуальной машины.

Application Framework

С уровня фреймворка приложений начинается код, написанный на Java. Здесь расположены ключевые компоненты системы, с которыми постоянно взаимодействует разработчик:

* **Activity Manager** — менеджер Activity. Их мы будем рассматривать несколько позже.
* **Package Manager** — менеджер пакетов. Именно он отвечает за установку, обновление и удаление приложений (и за некоторые другие вещи).
* **Window Manager** — управляет окнами. По сути, более высокий уровень абстракции над **Surface Manager**.
* **Telephony Manager** — отвечает за все, что связано с мобильной сетью, например, за звонки и смс.
* **Content Providers** — помните, чуть выше мы узнали, что в Android каждое приложение жестко изолировано друг от друга? Content Provider — это то, что позволяет приложениям с легкостью обмениваться данными.
* **Resource Manager** — предоставляет доступ к ресурсам приложений (строкам, графике и т.д.).
* **View System** — набор графических компонентов, таких, как кнопки и поля для ввода текста.
* **Location Manager** — предоставляет API для получения данных о местоположении.
* **Notification Manager** — позволяет приложениям показывать уведомления.

Applications

Наконец, уровень приложений. Именно на нем работают, собственно, все приложения в Android. Больше об этом уровне сказать, собственно, нечего :)

На этом, пожалуй, все. Жду вас в следующем уроке, в котором мы создадим наше первое приложение!