электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

 # 05, май 2016

УДК 004.4'2

Анализ уровней API платформы ANDROID с точки зрения разработки мобильных приложений

***Галахова Н.Р.**, студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Компьютерные системы и сети»

***Гаврилова М.А.**, ассистент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Компьютерные системы и сети»
magavrilova@bmstu.ru

Введение Платформа Android – одна из наиболее динамично развивающихся разработок на современном IT-рынке. Android позиционируется как универсальная операционная система для широкого класса устройств, включая смартфоны и планшетные компьютеры, носимые устройства, телевизоры и автомобили [1]. На данный момент выпущено 22 версии платформы [2], каждая из которых характеризуется уровнем API – целочисленным значением, однозначно идентифицирующим ревизию фреймворка, предоставляемую версией платформы Android [3].

В каждой версии в платформу вносятся значительные изменения, касающиеся как исправления обнаруженных ошибок, так и введения новых функций и возможностей. В связи с этим часть средств инструментария предыдущих версий фреймворка помечается как устаревшая и нежелательная для использования (deprecated), а также появляются более совершенные либо принципиально новые инструменты, доступные для разработчика. Таким образом, с каждой версией платформа Android усложняется и становится более требовательной к аппаратным ресурсам устройства, вместе с тем получая всё более широкий функционал.

При проектировании приложений для операционной системы Android немаловажным является вопрос, какой минимальный уровень API следует поддерживать при разработке. Его решение зависит как от предполагаемой функциональности проектируемого приложения, так и от возможностей, предоставляемых различными версиями платформы. Данная статья посвящена рассмотрению ключевых различий между уровнями API.

Эволюция уровней API

Первая стабильная версия системы Android была выпущена 23 сентября 2008 г. [4] для процессорной архитектуры ARM и требовала для запуска 128 МБайт RAM и 256 МБайт ROM. На этом этапе в API были заложены основные возможности работы с системой: виртуальная машина Dalvik, оптимизированная для низкого потребления памяти, поддержка 2D- и 3D-графики, встраиваемая БД SQLite, фото- и видеосъёмка, GPS-навигация, поддержка мультимедиа, беспроводных стандартов Wi-Fi, Bluetooth, а также сотовой связи 2G и 3G.

В версии 1.5 (API level 3) появилась поддержка виджетов – миниатюрных приложений, встраиваемых в другие приложения. В версии 1.6 (API level 4) возник синтез речи и расширенная поддержка жестов. Основным изменением в версии 2.0 (API level 5) стали многочисленные функции камеры, включая вспышку, цифровой зум и макросъёмку.

Благодаря реализации JIT-компилятора в версии 2.2 (API level 8) производительность системы возросла в 3-5 раз [5]. Кроме того, появилась возможность установки приложений на SD-карту и использовать смартфон в качестве модема и точки доступа. В последующей версии 2.3 (API level 9) были обновлены средства разработки графического пользовательского интерфейса, введена поддержка IP-телефонии и технологии NFC, возникла возможность работы с несколькими камерами (включая фронтальную); также была расширена поддержка датчиков (например, гироскопов и барометров).

Версия 3.0 (API level 11) стала первой системой Android, предназначенной для работы на планшетных ПК и оптимизированной для устройств с большими экранами. Появились панели System Bar и Action Bar, виртуальные рабочие столы, поддержка периферийных устройств посредством USB и Bluetooth. Последующие модификации (3.1, API level 12; 3.2, API level 13) расширяли спектр поддерживаемых устройств и возможности масштабирования приложений в соответствии с разрешением устройства.

В версии 4.0 (API level 14) появился экран блокировки, допускающий немедленный запуск приложений, а также возможность разблокировки смартфона при помощи распознавания лиц. Возник Android Beam для быстрого обмена данными, технология Wi-Fi Direct, а также фреймворк VPN. Последующая версия 4.1 (API level 16) имеет улучшенный графический пользовательский интерфейс с частотой смены кадров до 60 в секунду [6]. При этом центральный и графический процессоры работают параллельно, благодаря чему достигается увеличение производительности. В то же время система стала более требовательной к аппаратным ресурсам, поэтому модификация 4.4 (API level 19) была оптимизирована для смартфонов с оперативной памятью 512 МБайт. Также в этой

версии появилась поддержка облачных сервисов (принтеров, хранилищ) и был впервые представлен как альтернатива Dalvik предкомпилятор Android Runtime (ART). В рамках этой же версии платформы возникла платформа для умных часов.

В версии 5.0 (API level 21) ART окончательно вытеснил Dalvik. Платформа получила поддержку 64-битных процессоров, появился Project Volta, регулирующий обращения к процессору и тем самым экономящий заряд, заменено средство работы с камерами. Возникла визуальная концепция Material Design, согласно которой окна должны переключаться плавно и незаметно, с использованием эффектов теней. Следующая версия, 5.1 (API level 22), получила официальную поддержку нескольких сим-карт и новую систему защиты.

Актуальная на данный момент версия 6.0 (API level 23) имеет нативную поддержку сканирования отпечатков пальцев, мобильную платёжную систему Android Pay, встраиваемую в любые мобильные приложения, экспериментальный мультиоконный режим и автоматическое полное резервное копирование и восстановление для приложений.

На сегодняшний день платформа Android представляет собой обширный комплекс средств, призванный сделать Android-устройство максимально востребованным и удобным для пользователя. Глубокая интеграция с сервисами Google, голосовой ввод, платёжная система и другие особенности делают устройство на платформе Android «персональным помощником» пользователя, берущим на себя широкий спектр разнообразных функций. Однако усложнение операционной системы влечёт за собой рост требований к аппаратным ресурсам и увеличение расходуемой энергии.

Факторы выбора уровня API

При разработке мобильного приложения на платформе Android программист выбирает минимальную и целевую API. Минимальной называется API с наименьшим номером, способная поддерживать работу приложения. Для обеспечения обратной совместимости желательно, чтобы номер минимальной API был как можно меньше. Целевая API показывает, для какого уровня API разработано и протестировано приложение. Существует также атрибут для указания максимальной API, однако его использование не рекомендуется к использованию, поскольку при обновлении, если номер нового поддерживаемого уровня API больше, чем значение максимальной API, приложение может быть удалено с устройства [7].

От выбранного уровня API зависит его успех на рынке, а также быстрота устаревания. Ориентированность на наиболее новую и ещё не распространившуюся

широко версию платформы лишает значительную часть пользователей возможности его установки и эксплуатации и тем самым снижает популярность разработки. Слишком низкая целевая API влечёт быстрое устаревание приложения и, как следствие, более раннюю необходимость обновлять код для новых версий.

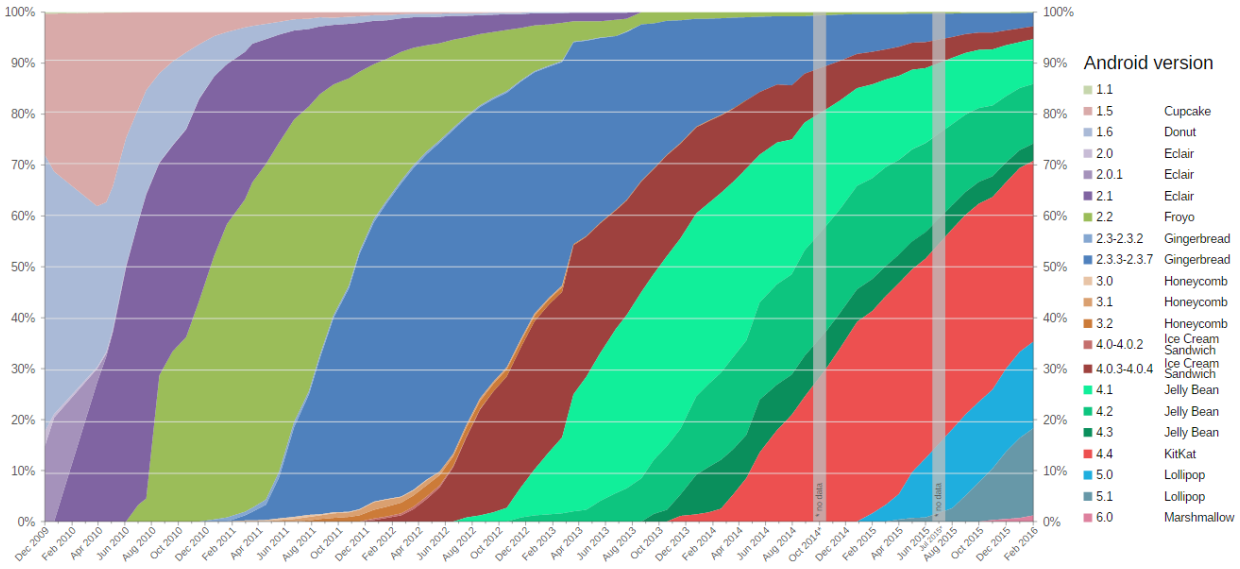
На выбор целевой и минимальной API влияет, в первую очередь, требование к классу устройств, поддерживающих разрабатываемое мобильное приложение. В случае, если приложение предназначено к использованию на конкретном устройстве с заранее известной установленной операционной системой (например, для «умных» часов, автомобиля либо специализированного устройства), целевая API определяется однозначно и соответствует системе, для которой разрабатывается приложение. Если приложение рассчитано на массовый рынок, необходимо учитывать разнообразие актуальных на данный момент версий платформы. В этом случае важным показателем является статистика распределения версий Android, предоставляемая Google и получаемая при помощи магазина приложений Google Play.

Кроме того, история развития платформы Android показывает, что каждая новая версия системы претерпевала в той или иной мере изменения в сравнении с предыдущей. Это не могло не отражаться на API, который дополнялся новыми возможностями и очищался от элементов, переставших быть актуальными. В результате при выборе уровня API необходимо знать, обеспечивает ли он поддержку необходимых аппаратных и программных средств, используемых приложением.

Рассмотрению вышеуказанных факторов выбора API посвящены следующие разделы.

Распределение версий Android

На рисунке ниже представлена диаграмма распределения версий платформы Android, используемых на устройствах, осуществляющих доступ к Google Play.



Из диаграммы видно, что на момент подготовки материала (февраль 2016 г.) наиболее распространённой версией платформы является 4.4 KitKat, соответствующая уровню API 19 (35,5% устройств). Следующая по популярности версия – более новый Android 5.0 и 5.1 Lollipop (уровни API 21 и 22), их доля составляет 17.0% и 17.1% соответственно. Также остаётся актуальным Android 4.1 – 4.3 Jelly Bean (уровни API 16 — 18), в сумме имеющий 23,9% от общего числа устройств. Среди этих версий наиболее популярна 4.2 (уровень API 17), процент использования которой составляет 11.7%. Более старые версии (4.0 Ice Cream Sandwich, 2.3 Gingerbread) продолжают использоваться на 5% устройств в сумме и постепенно вытесняются более совершенными модификациями. Доля новейшей версии 6.0 Marshmallow (уровень API 23) составляет 1,2% и будет расти по мере выпуска обновлений производителями и увеличения объёмов производства устройств с предустановленной последней версией платформы. Статистические данные взяты из [8].

На основании приведённой статистики можно сделать вывод, что на данный момент при разработке мобильных приложений для платформы Android следует ориентироваться на версии 4.4, 5.0 и 5,1 как на наиболее популярные среди пользователей Android-устройств. Предшествующие версии постепенно уходят с рынка как устаревающие и не соответствующие возможностям современных портативных устройств. Новейшая версия платформы 6.0 в настоящее время ещё не получила массового распространения, однако следует учитывать перспективный рост её доли в ближайшем будущем.

Инструментарий версий Android

При выборе целевого уровня API, кроме популярности среди пользователей, следует учитывать особенности набора средств, предоставляемых фреймворком данной версии. С

развитием аппаратного обеспечения расширяется программный инструментарий, доступный разработчику. В то же время совершенствуются уже существующие средства: в них устраняются ошибки и уязвимости и добавляется дополнительный функционал. В результате этих изменений множество инструментов более старых версий API помечается разработчиками платформы как нежелательные для использования (deprecated), а средства, доступные в более новых версиях, не работают в более старых.

Данные об изменениях наиболее важных элементов инструментария Android, произошедших в актуальных версиях (4.1 – 6.0), приведены в таблице ниже. Информация взята из [9] – [16].

Следует отметить, что поддержка API OpenGL ES имеет свои особенности. Для работы Open GL ES 3.0 API необходима аппаратная поддержка, обеспечиваемая производителем, и устройства с установленной системой Android 4.3 и выше могут не моддерживать эту версию API [17]. Согласно [8], наиболее распространена версия OpenGL ES 2.0.

Инструмент	**Введён в API level**	**Последнее изменение в API level**
Intent (абстрактное описание операции для исполнения)	1	23 (добавлен функционал)
Activity (средство для взаимодействия с пользователем)	1	23 (добавлен функционал, часть элементов помечена как «нежелательные»)
Fragment (поведение или часть интерфейса в Activity)	11	23 (добавлен функционал, часть элементов помечена как «нежелательные»)
Service (служба, выполняемая в фоновом режиме)	1	23 (часть элементов помечена как «нежелательные»)
Виджеты	3	21 (добавлен функционал)
Доступ к xml-файлам ресурсов	1	23 (добавлен функционал, часть элементов помечена как «нежелательные»)
Animator (базовая поддержка анимации)	11	19 (добавлен функционал)
Уведомления	1	23 (добавлен функционал, часть элементов помечена как «нежелательные»)
OpenGL ES	3	20 (часть элементов помечена как «нежелательные»)
Аппаратное ускорение	11	14 (по умолчанию включено)
MediaPlayer	1	23 (добавлен функционал)

Camera	1	21 (полностью помечен как «нежелательный»)
android.hardware.camera2 (новое средство управления камерой)	21	23 (добавлен функционал)
android.location (геолокация)	1	21 (не рекомендуется к использованию, заменён на сервис Google)
Sensor (управление датчиками)	3	21 (расширен спектр поддерживаемых датчиков)
Bluetooth	1	18 (добавлен функционал)
NFC	9	17 (добавлен функционал)
Резервное копирование	8	21 (добавлен функционал)
ConnectivityManager (управление сетевыми соединениями)	1	21 (добавлен функционал, часть элементов помечена как «нежелательные»)

Из приведённых данных видно, что последняя версия API (уровень 23) содержит большое количество изменений в наиболее востребованных инструментах. Эта версия поддерживает наибольшее число аппаратных и программных средств и, следовательно, имеет больше всего возможностей. Для того, чтобы максимально полезно задействовать инструментарий платформы, желательно разрабатывать и отлаживать приложения именно для версии 6.0 (т. е. указывать её целевой).

До выпуска Android 6.0 последней версией, претерпевшей значительные изменения, стала 5.0 (уровень API 21). Имено в этой ревизии появился Project Volta, а Dalvik был заменён на ART. Кроме того, полностью изменилось средство управления камерой, была введена поддержка API OpenGL ES 3.1 и коренным образом изменилось управление уведомлениями. Таким образом, если в мобильном приложении не планируется глубокая обратная совместимость (до версий 4.x и ниже), именно версию 5.0 целесообразно указать минимальной при разработке.

Заключение

В соответствии с анализом приведённых данных можно сделать вывод о выборе API для разработки мобильного приложения на платформе Android.

Наиболее популярными версиями системы Android на данный момент являются 4.4 KitKat и 5.x Lollipop. При этом в версии 5.0 произошли значительные изменения как пользовательских характеристик, так и доступных разработчику инструментов, и множество часто используемых объектов (таких, как уведомления, камера, геолокация)

получили обновлённые либо кардинально новые средства управления, впоследствии улучшенные в версии Android 6.0 Marshmallow. Кроме того, изменилась среда выполнения приложений: Dalvik уступил более быстрому Android Runtime. В связи с этим для приложений, разрабатываемых в условиях современного рынка программного обеспечения для платформы Android, целесообразно ориентироваться на эту версию платформы как на минимальную, если не планируется поддержка более старых версий. Уровень API, соответствующий Android 5.0, имеет номер 21.

Целевая версия зависит от предпочтений разработчика и требований к приложению, однако рекомендуется отлаживать приложение в новейшей версии платформы 6.0 Marshmallow, обладающей на данный момент наиболее широким спектром возможностей. Уровень API, соответствующий Android 6.0, имеет номер 23.

Список литературы

[1] Android. Available at: http://www.android.com/intl/ru_ru/, accessed 16.02.2016.

[2] Codenames, Tags, and Build Numbers. Available at: https://source.android.com/source/build-numbers.html, accessed 16.02.2016.

[3] What is API Level? Available at: http://developer.android.com/intl/ru/guide/topics/manifest/uses-sdk-element.html#ApiLevels, accessed 16.02.2016.

[4] Announcing the Android 1.0 SDK, release 1. Available at: http://android-developers.blogspot.ru/2008/09/announcing-android-10-sdk-release-1.html, accessed 18.02.2016.

[5] Android 2.2 Froyo. Available at: http://android-phones.ru/android/2-2/, accessed 18.02.2016.

[6] Google представила Android 4.1. Available at: http://www.webcitation.org/69nDifUIw, accessed 18.02.2016.

[7] <uses-sdk>. Available at: http://developer.android.com/intl/ru/guide/topics/manifest/uses-sdk-element.html, accessed 20.02.2016.

[8] Dashboards. Available at: http://developer.android.com/intl/ru/about/dashboards/index.html, accessed 20.02.2016.

[9] Android 4.1 APIs. Available at: http://developer.android.com/intl/ru/about/versions/android-4.1.html, accessed 20.02.2016.

[10] Android 4.2 APIs. Available at: http://developer.android.com/intl/ru/about/versions/android-4.2.html, accessed 20.02.2016.

[11] Android 4.3 APIs. Available at:

http://developer.android.com/intl/ru/about/versions/android-4.3.html, accessed 20.02.2016.

[12] Android 4.4 APIs. Available at: http://developer.android.com/intl/ru/about/versions/android-4.4.html, accessed 20.02.2016.

[13] Android 5.0 APIs. Available at: http://developer.android.com/intl/ru/about/versions/android-5.0.html, accessed 20.02.2016.

[14] Android 5.1 APIs. Available at: http://developer.android.com/intl/ru/about/versions/android-5.1.html, accessed 20.02.2016.

[15] Android 6.0 APIs. Available at: http://developer.android.com/intl/ru/about/versions/android-6.0.html, accessed 20.02.2016.

[16] Общие сведения о платформе Android. Available at: http://developer.android.com/intl/ru/guide/index.html, accessed 20.02.2016.

[17] OpenGL ES. Available at: http://developer.android.com/intl/ru/guide/topics/graphics/opengl.html, accessed 20.02.2016.

[18] Обзор и анализ методов обеспечения совместимости приложений для операционной системы Android. Available at: http://sntbul.bmstu.ru/file/711822.html?__s=1, accessed 20.02.2016.