УДК 004.428

ВЫБОР ЯЗЫКА ПРОГРАММИРОВАНИЯ ДЛЯ РАЗРАБОТКИ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

Мищенко И.И, Девятков В.В., Матейчук Р.А.
ст. преподаватель каф. КИСС Кузнецов Н.А.
Одесский Национальный Политехнический Университет, УКРАИНА

АННОТАЦИЯ. В рамках исследования рассмотрены преимущества и недостатки двух основных библиотек для разработки мобильных приложений Native Development Kit и Software Development Kit.

Введение. При разработке приложений для мобильных устройств часто приходится решать задачи связанные с оптимальным выбором средства разработки. Среди множества вариантов выделяют разработку приложений для мобильных устройств используя Native Development Kit (NDK) [1] и Software Development Kit (SDK). Каждая из перечисленных библиотек позволяют эффективно использовать аппаратные возможности одной из самых популярных мобильных платформ Android.

Цель работы. Целью данной работы является краткий обзор основных принципов Andoid NDK, так же критериев выбора между SDK и NDK.

Основная часть работы. Android-устройства набирают популярность быстрее, чем любая другая мобильная платформа, что делает их отличным выбором для первого знакомства с разработкой мобильных приложений, особенно для Java-программистов. Для разработки мобильных приложений Google предоставляет два пакета разработки: SDK и NDK[2]. Android NDK (native development kit) - это набор инструментов, который позволяет реализовать части вашего приложения с использованием языков с нативным кодом, таких как C и C ++. Для некоторых типов приложений это может помочь повторно использовать библиотеки кода, написанные на этих языках. Google предложил несколько рекомендаций к применению NDK среди которых :

- нужно увеличить производительность (например, сортировка большого объема данных);
- использовать стороннюю библиотеку. Например, много уже чего написано на C/C++ языках и нужно просто заиспользовать существующий материал. Пример таких библиотек, как, «Ffmpeg», «OpenCV»;
- программирование на низком уровне.

Используя Android Studio 2.2 и выше можно использовать NDK для компиляции кода C и C ++ в собственной библиотеке и упаковки его в APK с помощью Gradle, интегрированной системы сборки IDE. Затем созданный Java-код может вызывать функции в родной библиотеке через инфраструктуру Java Native Interface (JNI).

Отличным примером разработки с использованием NDK являются приложения в которых вычисления проводятся очень интенсивно, и их количество очень велико (например обработка изображения). В таких случаях достигается отличная производительность, значительно превышающая производительность SDK[3].

Для мобильной разработки у C ++ можно выделить следующие основные преимущества:

1) Время исполнения и объем памяти

За счет того что C++ не использует при работе программы ни каких виртуальных машин или прочих прослоек, программа использует значительно меньше памяти и использует процессор только для непосредственных вычислений (в отличии от JAVA где еще необходимо виртуализировать).

2) Более «быстрый» код

Ряд известных программных продуктов используют C++ для кроссплатформенной разработки, включая «Facebook Moments», «Dropbox», «Office», «Skype» и популярные игры, такие как «Clash of Clans». Поскольку C ++ обычно не имеет стандартного пользовательского

интерфейса, код пользовательского интерфейса написан на родном языке, а C ++ используется для логики.

На C++ также существуют кроссплатформенные библиотеки для написания пользовательского интерфейса, одним из ярчайших примеров которых является «Qt».

3) Среда разработки «Visual Studio»

В 2005 году, Microsoft выпустила бесплатную экспресс-версию «Visual Studio», а после и «Community Edition» в 2013 году, которая позволяет устанавливать плагины и управлять ими через «Nuget Package Manager». Благодаря этому, C++ можно скомпилировать, чтобы ориентироваться на Android и создавать Android-приложения с помощью «Native-Activity». При компиляции для Android платформа использует «CLANG toolchain». Среда «Visual Studio» включает быстрый эмулятор Android вместе с наборами для разработки Android (SDK, NDK), а также «Apache Ant» и «Oracle Java JDK», поэтому не нужно переключаться на другую платформу, чтобы использовать внешние инструменты[4]. Высококачественные аудиоприложения обычно требуют больше функциональности, чем просто воспроизведение или запись звука, и тут на помощь приходит ещё одна состовляющая NDK – OpenSL ES, кроссплатформенная библиотека, позволяющая высокопроизводительно обрабатывать звук[6]. Безусловно, одним из главных преимуществ использования NDK является не так давно добавленная поддержка Vulkan, графического интерфейса нового поколения[7].

К значительным преимуществам использования NDK является возможность использовать код практически без изменений. Например, если взять написанный на OpenGL игровой движок, то можно без особых изменений использовать код движка в приложениях под телефон, таким образом NDK позволяет разрабатывать кроссплатформенный код. Более того, можно разрабатывать и отлаживать код в одной среде разработки, а затем собирать проект для Android. Так часто поступают когда разрабатывают приложения под несколько платформ, к примеру при разработке игр под Windows, Linux и Andoid, сперва пишут код для Windows (выбор именно такой последовательности обусловлен удобствоми отладки программ в Windows), затем собирают под Linux и после этого под Andoid. Для обучения программирования с использованием NDK Google предлагает использовать ряд примеров[5].

Уже сейчас многие приложения в Google Play используют C++, к таким приложениям относятся игры, различные фото- и видео- редакторы.

Выводы. Обычно разработчики решают использовать нативный код в двух случаях: они хотят увеличить производительность своего приложения, или у них есть готовый C/C++ проект, который требуется с минимальными затратами портировать на Android. Одним из основных преимуществ использования NDK является скорость работы нативных приложений по сравнению с использованием SDK.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Хабрахарбр. Введение в Android NDK из песочницы [Електронний ресурс]. – Режим доступу: URL: https://habrahabr.ru/post/203014/

2. Хабрахарбр. Android SDK vs NDK — сравнение производительности однотипных участков кода из песочницы [Електронний ресурс]. – Режим доступу: URL: https://habrahabr.ru/post/215647/

3. Википедия. Android SDK [Електронний ресурс]. – Режим доступу: URL: https://ru.wikipedia.org/wiki/Android_SDK

4. Android Native Development Kit (NDK) [Електронний ресурс]. – Режим доступу: URL: http://www3.ntu.edu.sg/home/ehchua/programming/android/android_ndk.html

5. Google Android NDK samples [Електронний ресурс]. - Режим доступу: URL: https://github.com/googlesamples/android-ndk

6. Описание OpenSL ES [Електронний ресурс]. - Режим доступу: URL: https://developer.android.com/ndk/guides/audio/index.html?hl=ru

7. Описание Vulkan [Електронний ресурс]. - Режим доступу: URL: https://developer.android.com/ndk/guides/graphics/index.html?hl=ru