**Аннотация.** В целях освоения новых инструментов разработки под операционную систему Android на мультиплатформенном движке Unity и дальнейшего их правильного применения в оной среде, автором были проанализированы и протестированы встроенные возможности интеграции языка программирования Java в среду разработки проекта. При написании Android приложений, довольно часто приходиться взаимодействовать с Android API. Большинство наиболее часто используемых методов Android API уже встроено в инструментарий движка Unity, но в редких случаях его недостаточно, поэтому разработчики встроили возможность подключения “ JAR” и “AAR” плагинов с помощью классов AndroidJavaObject и AndroidJavaClass, которые дают возможность вызова методов из скомпилированного файла. Для написания правильно работающего плагина, нужно придерживаться определенного паттерна проектирования. Изменить и настроить Android Manifest при компиляции JAR или AAR файла.

**Ключевые слова:** Разработка, Unity, Android Application on Unity, Android Plugin for Unity, Java in Unity, JAR & AAR, Android API in Unity.

Каждый разработчик сталкивался с задачей взаимодействия разрабатываемого приложения и операционной системы, под которое оно разрабатывается.

Unity - кроссплатформенная среда разработки с помощью которой можно создавать игры и приложения под разные платформы, но большая часть разработанных приложений на этом движке реализованы именно под операционную систему Android. Сам же Unity разработан на языке C#, поэтому многие не опытные разработчики могут задаваться вопросом – «Как же взаимодействовать с системными вызовами». Большинство основных системных вызовов уже реализованы «под капотом» игрового движка, например: создание/удаление/запись файлов, запрос на разрешения приложения, геолокация, интернет запросы и так далее. Для написания простого приложения, которое должно будет сохранять что-либо или отправлять запросы на сервер встроенного инструментария хватит, но если приложение должно работать с Bluetooth устройствами, сервисами, ИК-портом (если присутствует в телефоне), то встроенных методов Unity недостаточно. Разработчики движка спрогнозировали это и дали возможность интеграции Java в приложение. Код написанный на Java можно импортировать только в уже скомпилированном файле с расширением AAR или JAR, а взаимодействие с этими «плагинами» будет осуществляться через классы AndroidJavaClass и AndroidJavaObject. AndroidJavaClass — это представление Unity универсального экземпляра “java.lang.Class”, а AndroidJAvaObject – “java.lang.Object”. Его можно использовать как бестиповый интерфейс для экземпляра любого класса Java внутри C# скрипта.

При написании кода на языке Java у вас открывается максимальный спектр возможностей взаимодействия с операционной системой Android. Но нужно придерживаться определенных правил, если код, написанный вами, используется лишь, как библиотека, к которой идёт обращение из основного приложения.

Первым делом нужно разобраться, как компилировать код на Java. При компиляции библиотеки под OC Windows, и дальнейшего взаимодействия с ней, на выходе получается файл с расширением DLL. При компиляции под ОС Android возможны два вида файлов:

* JAR - Java Archive. Содержит файлы классов, JAVA-ресурсы, зависимые библиотеки, и другие необходимые для приложения файлы. JAR может использоваться как в Android-приложении, так и в приложении на чистой Java.
* AAR - Android Archive. Так же, как и JAR является zip-архивом, но может использоваться только в Android-приложении. AAR содержит скомпилированный исходный код в файле classes.jar, Android -ресурсы, AndroidManifest, и другие файлы, входящие в состав APK.

При компиляции библиотеки для интеграции в Unity приложение подходят оба архива, но нужно учитывать особенности. В документации Unity разработчики советуют использовать AAR плагины из-за большей совместимости