LAZARUS: сборка приложения под андройд

Дата: 14.12.2013
Автор: Логинов Дмитрий Сергеевич
www.loginovprojects.ru
loginov_d@inbox.ru

Об ответственности

Автор данного документа не несет ответственности за возможный ущерб (моральный или материальный), полученный в результате чтения / использования изложенной информации.

На данный момент (2013 год) автор не является программистом под андройд, он не создал для него ни единой мало-мальски полезной программы. Знаком с андройдом весьма поверхностно.

Любой вывод, сделанный автором, может оказаться субъективным / ложным / неактуальным. Автор будет рад, если ему об этом сообщат.

Вступление

Впервые я услышал о Лазарусе в 2006 г. На том момент программировать в Лазарусе под Windows было еще нереально (в первую очередь его затачивали под Linux). Прошло несколько лет и Лазарус потихоньку превратился в средство для профессиональной разработки для Windows, Linux, MacOS. По сути, разработчики Лазаруса достигли уровня Delphi 7, но и во многом его превзошли (поддержка x64, Unicode). Лазарус за последние годы очень сильно продвинулся. Успехи Лазаруса в первую очередь зависят от успехов компилятора FreePascal. Данный компилятор умеет компилировать программы под множество платформ, как аппаратных (x86, x64, ARM и мн.др.) так и программных (Windows, Linux, MacOS, iOS, Android (?) и т.д.). А Лазарус выступает как удобный инструмент разработчика. Несколько лет назад FreePascal научился компилировать под процессоры ARM. Также несколько лет назад появились мобильные устройства (смартфоны/планшеты/навигаторы и мн.др), использующие Android в качестве операционной системы. На сегодняшний момент мобильные устройства на базе андройд лидируют на мировом рынке.

Современный успешный программист должен уметь программировать под Android, или хотябы представлять, что такое Android и с чем его едят.

Android – это проект, разрабатываемый по открытой лицензии компанией Google и другими разработчиками. Android основан на ядре Linux, однако, с точки зрения программиста, это совершенно разные системы. Если

вы умеете программировать под Linux, это совершенно не означает, что ваши программы будут работать под Android. Не будут! В Windows есть Windows-API, в Линуксе (по аналогии) есть Linux-API. Конечно, и в Андройде есть Android-API, однако его разработчики постарались сделать все, для того, чтобы мы, программисты, к нему не добрались. У нас нет доступа к Android-API. Вместо этого Google предлагает нам использовать JAVA-API. Т.е. если мы хотим что-то сделать (например, прочитать информацию из файла), то мы должны обратиться к JAVA-машине и вызвать у неё необходимую функцию (вернее необходимый метод определенного класса). Такой подход гарантирует высокий уровень безопасности, однако делает процесс программирования более сложным.

Но некоторые вещи мы все-же можем делать напрямую, минуя Java. Например, нарисовать на экране картинку. В первых версиях Android и рисование нужно было производить через Java-API, однако, изходя из соображений производительности, разработчики Android пошли на уступки и теперь предоставляют нам ряд SO-библиотек, в том числе для 2D и 3D графики.

Сразу отмечу, что в Лазарусе поддержка Android на данный момент очень слабая. Зато сам Лазарус бесплатный! И если у вас много сил и свободного времени, то используйте Лазарус, дерзайте!

Оченью этого года компания Embarcadero выпустила Delphi XE5 с полноценной поддержкой андройд. Возможностей в тысячу раз больше, чем в Лазарусе. Если Вы цените своё время, то разумнее приобрести Дельфи. В настоящее время разработчики Delphi проводят огромную работу над улучшением качества своего продукта, думаю, в ближайшее время разработка под Андройд будет не сложнее, чем разработка под Windows.

Можно также для программирования под Android использовать Eclipse. Однако, по моему мнению, такой выбор окажется недальновидным. В Delphi вы пишете программу, которую можно будет запускать и под Android и под iOS и под Windows и под MacOS. А программу, написанную в Eclipce, можно будет запускать только под Android.

Вообще, каждый производитель платформ стремится подготовить программиста и навсегда привязать его именно к своей платформе. И программисты на это охотно ведутся, т.к. производитель платформы предоставляет все необходимое для разработки абсолютно бесплатно.

Такой подход хорош для производителей платформ, но не программистов. В результате успешному программисту нужно знать и C# (для Windows) и C++ (для Linux) и Objective-C (для MacOS и iOS) и Java Android Edition (для Android) и многое-многое другое.

Разработчики Delphi поступают, нам мой взгляд, более гуманно по отношению к пользователю: предоставляют единый инструмент с единым языком программирования под все популярные платформы. Но резонно просят за это денюшку. При этом всем желающим использовать Delphi в образовательных целях предоставляют его совершенно бесплатно (а иначе почему Delphi так легко можно скачать с торрентов? :)

Разработчики Delphi выполняют поистине грандиозную работу. Еще ни кому, на мой взгляд, не удавалось обеспечить кроссплатформенности. Были лишь громкие лозунги (как в случае с Java). А Delphi к этой цели все ближе и ближе.

Компоненты, необходимые для запуска программы в ОС Android

Немного отвлеклись от темы! Итак, в Лазарусе можно разработать простейшее приложение, которое будет работать в ОС Android. Для сборки и запуска приложения необходимо подготовить следующие компоненты:

- файл «AndroidManifest.xml», который содержит необходимые параметры для правильной загрузки приложения Java-машиной. Также в этом файле перечислены разрешения (работа с файловой системой, сетью, СМС, камерами, сенсорами и т.д.), которые Java-машина должна предоставить приложению.

- основной файл «classes.dex», который будет загружаться Java-машиной. На самом деле файл classes.dex является результатом компиляции файла LCLActivity.java в стандартный байт-код (файл LCLActivity.class) и последующим преобразованием его в байт-код Java-машины Dalvik.

- файл «liblclapp.so», который является результатом компиляции под аппаратную платформу «ARM» программы, разработанной нами в Лазарусе. В файле LCLActivity.java есть код System.loadLibrary("lclapp"), благодаря которому Java-машина загружает библиотеку «liblclapp.so». После загрузки liblclapp.so происходит её инициализация, в рамках которой библиотека получает ссылку на запущенный экземпляр Java-машины, после чего она может вызывать любые методы любых классов Java-машины.

- файл «androidlcltest.apk», который является инсталляционным архивом, включающий в себя «AndroidManifest.xml», «classes.dex», «liblclapp.so» и другие файлы, необходимые для работы приложения в ОС Android. Этот файл можно без особых сложностей скопировать на телефон или планшет и запустить установку.

Необходимое ПО для компьютера

Всю работу будем выполнять в ОС Windows. Если у Вас нет Windows, но есть Linux, то ничем помочь не могу, делайте все по аналогии.

Я в качестве первоисточника брал документ «Lazarus and Android» по ссылке:
https://dl.dropboxusercontent.com/u/3753548/Lazarus%20and%20Android.pdf

(однако там много лишнего, и не взлетает. А мне хочется упростить)

Нам потребуется:

1) Скачать и установить комплект JDK 1.6 (32-разрядный).

Его можно скачать с сайта http://www.oracle.com

ссылка:
http://www.oracle.com/technetwork/java/javasebusiness/downloads/java-archive-downloads-javase6-419409.html#jdk-6u45-oth-JPR

файл: jdk-6u45-windows-i586.exe

Я не в курсе, почему нельзя использовать JDK 1.7. Говорят, что с ним какие-то ошибки лезут, но я не проверял, и желания такого не было.

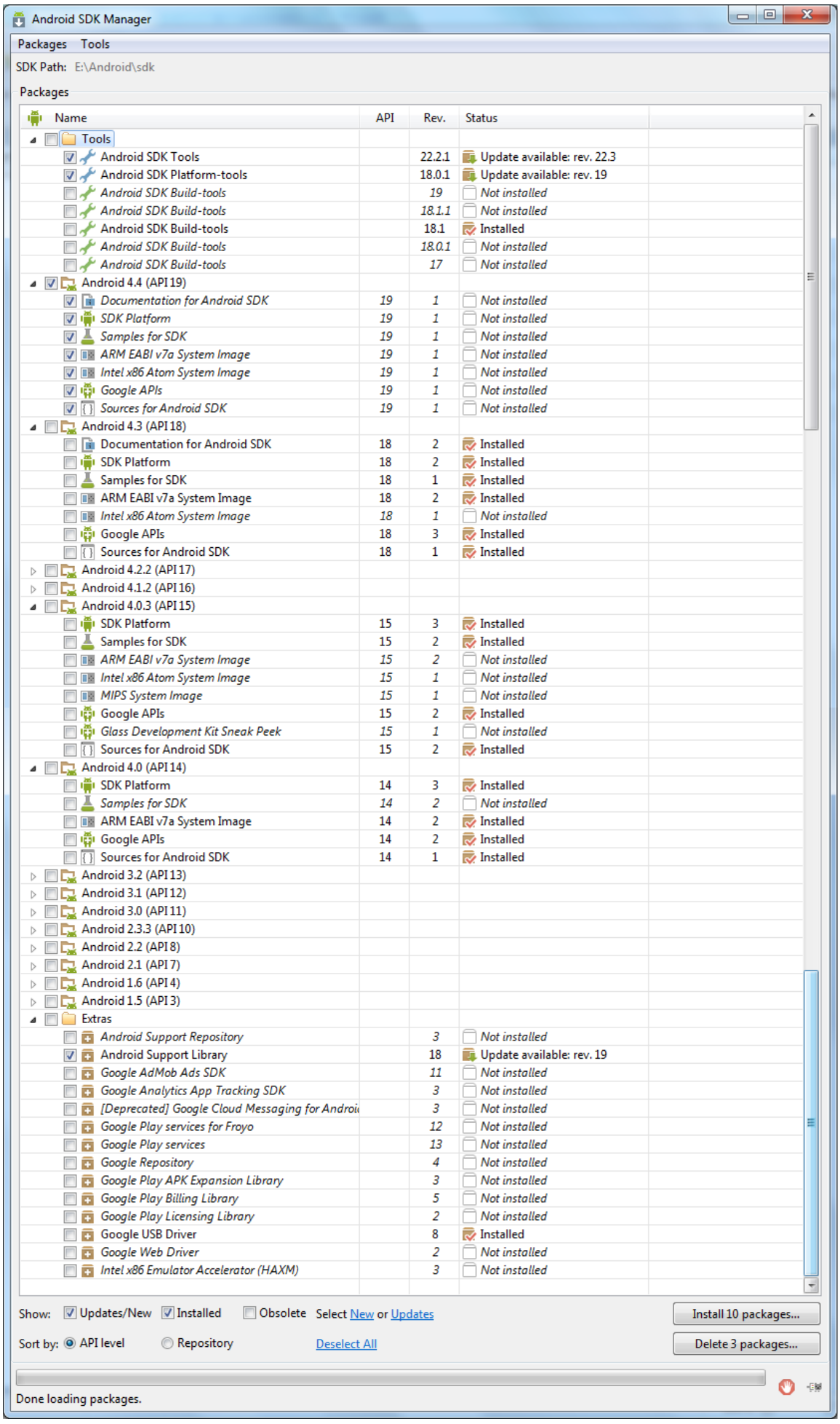
Этот JDK нужен для компиляции «LCLActivity.java» в файл «LCLActivity.class» (используется компилятор «javac» из JDK).

2) Скачать и установить комплект Android SDK.

Ссылка: http://dl.google.com/android/installer_r22.0.5-windows.exe

Рекомендую установить его в папку «E:\Android\sdk\» (после установки SDK в этом каталоге должно появиться несколько подкаталогов, а также файл «SDK Manager.exe»). И это только начало! Запустите файл «SDK Manager.exe» и скачайте дополнительные компоненты, как показано на рисунке на следующей странице. Я не в курсе, что означают те или иные галочки, однако с ними у меня все работает!

У меня папка «E:\Android\sdk\» заняла 2.2 ГБайт.


Android SDK Manager
Packages Tools
SDK Path: E:\Android\sdk
Packages
Name API Rev. Status
Tools
Android SDK Tools 22.2.1 Update available: rev. 22.3
Android SDK Platform-tools 18.0.1 Update available: rev. 19
Android SDK Build-tools 19 Not installed
Android SDK Build-tools 18.1.1 Not installed
Android SDK Build-tools 18.1 Installed
Android SDK Build-tools 18.0.1 Not installed
Android SDK Build-tools 17 Not installed
Android 4.4 (API 19)
Documentation for Android SDK 19 1 Not installed
SDK Platform 19 1 Not installed
Samples for SDK 19 1 Not installed
ARM EABI v7a System Image 19 1 Not installed
Intel x86 Atom System Image 19 1 Not installed
Google APIs 19 1 Not installed
Sources for Android SDK 19 1 Not installed
Android 4.3 (API 18)
Documentation for Android SDK 18 2 Installed
SDK Platform 18 2 Installed
Samples for SDK 18 1 Installed
ARM EABI v7a System Image 18 2 Installed
Intel x86 Atom System Image 18 1 Not installed
Google APIs 18 3 Installed
Sources for Android SDK 18 1 Installed
Android 4.2.2 (API 17)
Android 4.1.2 (API 16)
Android 4.0.3 (API 15)
SDK Platform 15 3 Installed
Samples for SDK 15 2 Installed
ARM EABI v7a System Image 15 2 Not installed
Intel x86 Atom System Image 15 1 Not installed
MIPS System Image 15 1 Not installed
Google APIs 15 2 Installed
Glass Development Kit Sneak Peek 15 1 Not installed
Sources for Android SDK 15 2 Installed
Android 4.0 (API 14)
SDK Platform 14 3 Installed
Samples for SDK 14 2 Not installed
ARM EABI v7a System Image 14 2 Installed
Google APIs 14 2 Installed
Sources for Android SDK 14 1 Installed
Android 3.2 (API 13)
Android 3.1 (API 12)
Android 3.0 (API 11)
Android 2.3.3 (API 10)
Android 2.2 (API 8)
Android 2.1 (API 7)
Android 1.6 (API 4)
Android 1.5 (API 3)
Extras
Android Support Repository 3 Not installed
Android Support Library 18 Update available: rev. 19
Google AdMob Ads SDK 11 Not installed
Google Analytics App Tracking SDK 3 Not installed
[Deprecated] Google Cloud Messaging for Androi 3 Not installed
Google Play services for Froyo 12 Not installed
Google Play services 13 Not installed
Google Repository 4 Not installed
Google Play APK Expansion Library 3 Not installed
Google Play Billing Library 5 Not installed
Google Play Licensing Library 2 Not installed
Google USB Driver 8 Installed
Google Web Driver 2 Not installed
Intel x86 Emulator Accelerator (HAXM) 3 Not installed
Show: Updates/New Installed Obsolete Select New or Updates
Install 10 packages...
Sort by: API level Repository Deselect All
Delete 3 packages...
Done loading packages.

3) Скачать и распаковать Android NDK

Ссылка: http://dl.google.com/android/ndk/android-ndk-r7c-windows.zip

У меня в результате имеется папка «e:\Android\ndk\android-ndk-r7c\», а уже в ней подкаталоги «build», «platforms» и др.

4) Скачать и распаковать Лазарус

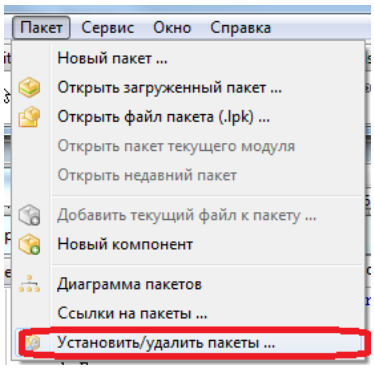
Ссылка: http://sdrv.ms/12cHbIZ

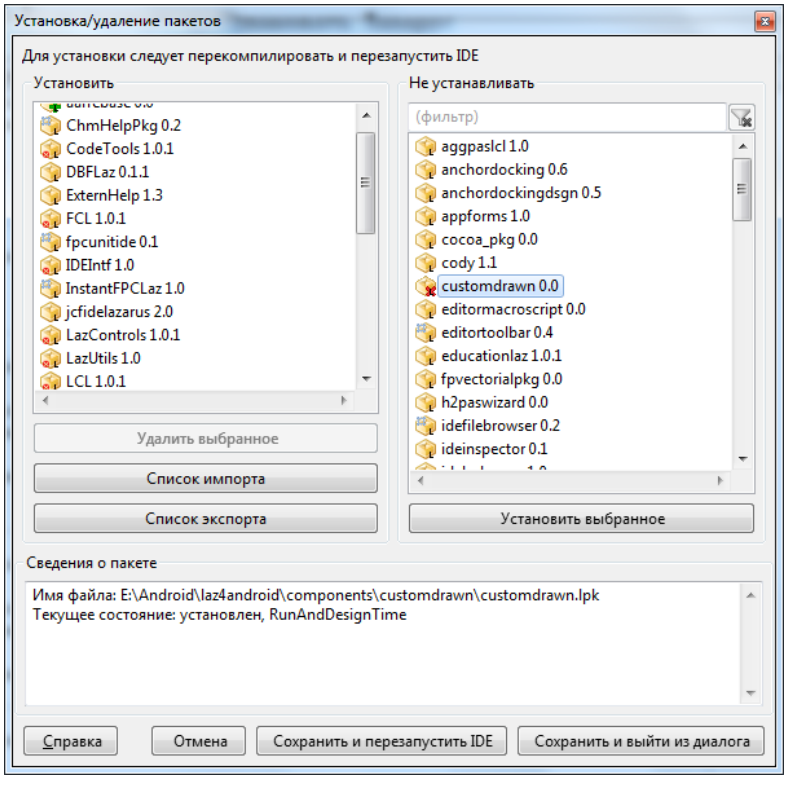
У меня в результате имеется папка «e:\Android\laz4android\», а уже в ней «components», «examples», «fpc» и др. Можно считать, что лазарус и FreePascal здесь уже установлены, осталось лишь немного поднастроить.

Для этого откройте в блокноте файл «laz4android\build.bat» и исправьте первую строку. У меня она выглядит следующим образом:

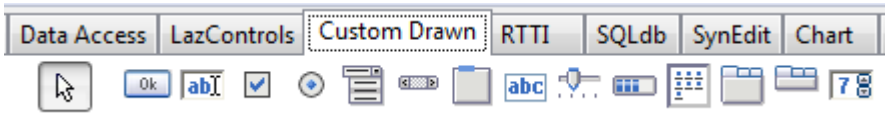
```
SET FPC_BIN_PATH=e:\Android\laz4android
```

Далее следует установить набор компонентов «customdrawn». См картинки:





В правой колонке выбрать «customdrawn 0.0», дважды щелкнуть по нему (он попадет в левый список и нажать «Сохранить и перезапустить IDE»). В результате набор компонентов «customdrawn» появится на вкладке «Custom Drawn»



Я не очень понимаю, на самом ли деле нужно ставить этот набор компонентов. Но не стал сопротивляться. Что плохого в еще одной вкладке?

Тем более сразу видно, какие компоненты гарантированно будут работатьв Android-приложении.

Сразу скажу, что в Android-приложении мы можем использовать компоненты «TEdit», «TButton», «TCheckBox», «TComboBox», «TProgressBar», «TTrackBar», «TTimer», «TMemo», «TBitBtn», «TImage», «TLabel» и некоторые другие. Список всех поддерживаемых визуальных компонентов, судя по всему, описан в модуле customdrawndrawers.pas:

```
TCDControlID = (
  cidControl,
  // Standard
  cidMenu, cidPopUp, cidButton, cidEdit, cidCheckBox, cidRadioButton,
  cidListBox, cidComboBox, cidScrollBar, cidGroupBox, cidPanel,
  // Additional
  cidStaticText,
  // Common Controls
  cidTrackBar, cidProgressBar, cidListView, cidCTabControl
  );
```

Если мы захотим использовать какие-то другие компоненты, но вряд ли они будут работать.

Однако в этом списке нет Timage, а он все-равно работает. Вероятно, что работают и все остальные компоненты-наследники «TGraphicControl».

Настройка и сборка проекта в Лазарусе

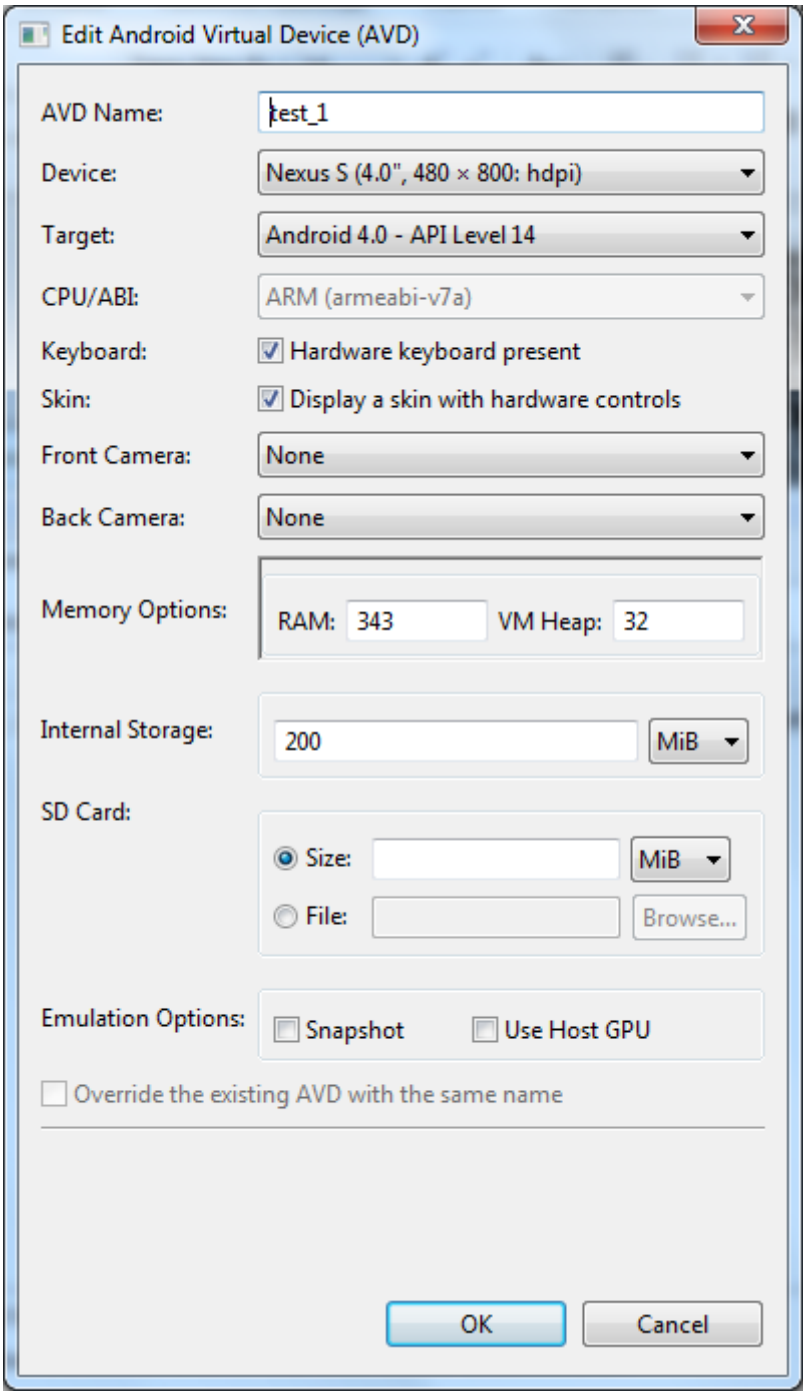
В составе Лазаруса в папке **examples** находится тестовый проект **androidlcl**. Однако, практически никому из программистов не удается его скомпилировать / запустить в ОС Android (по многим причинам). Мы с данным проектом разбираться не будем, оставим этот вопрос на совести разработчика, который его клепал.

Вместо этого скачиваем файл:

http://loginovprojects.ru/androidlcl.zip

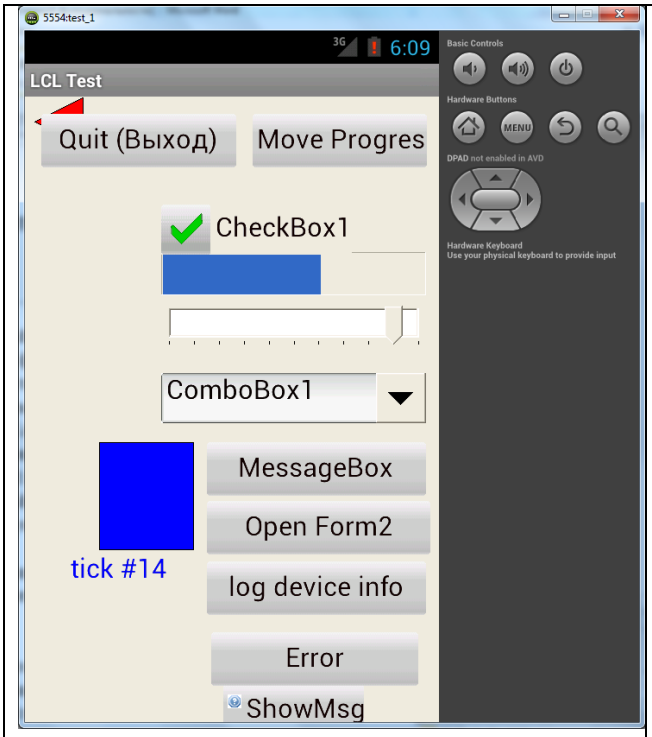
и распаковываем его в папку «e:\Android\projects\». В результате должна быть папка «e:\Android\projects\androidlcl\», содержащая каталог «android» и файлы «androidlcltest.lpr», «mainform.pas» и др.

Там уже есть файл «e:\Android\projects\androidlcl\android\bin\androidlcltest.apk». Попробуйте в первую очередь установить этот файл на своё Android-устройство (телефон или планшет) и убедиться что установка проходит и програма запускается. Если у вас нет Android-устройства, то запустите хотя бы виртуальную машину Android (с помощью «e:\Android\sdk\AVD Manager.exe»). С настройками виртуальной машины разбирайтесь самостоятельно. У меня настройки такие:

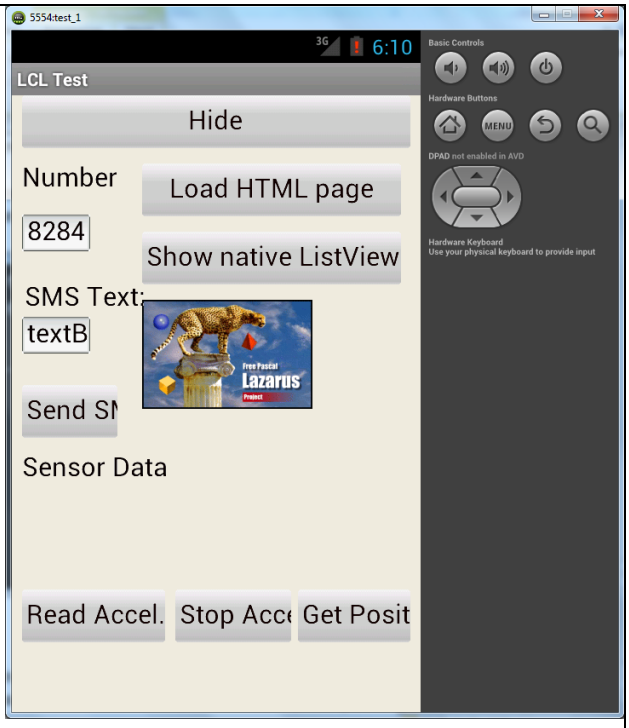


После того, как виртуальная машина Android запустится (около 5 минут), запустите файл: «e:\Android\projects\androidlcl\android\adb_install.bat». Он установит файл «androidlcltest.apk» на запущенную виртуальную машину Android (менее 1 минуты).

Найдите на телефоне/планшете (на реальном или виртуальном) приложение «LCLTest» и запустите его (менее 3 секунд). Картинка должна быть следующей:

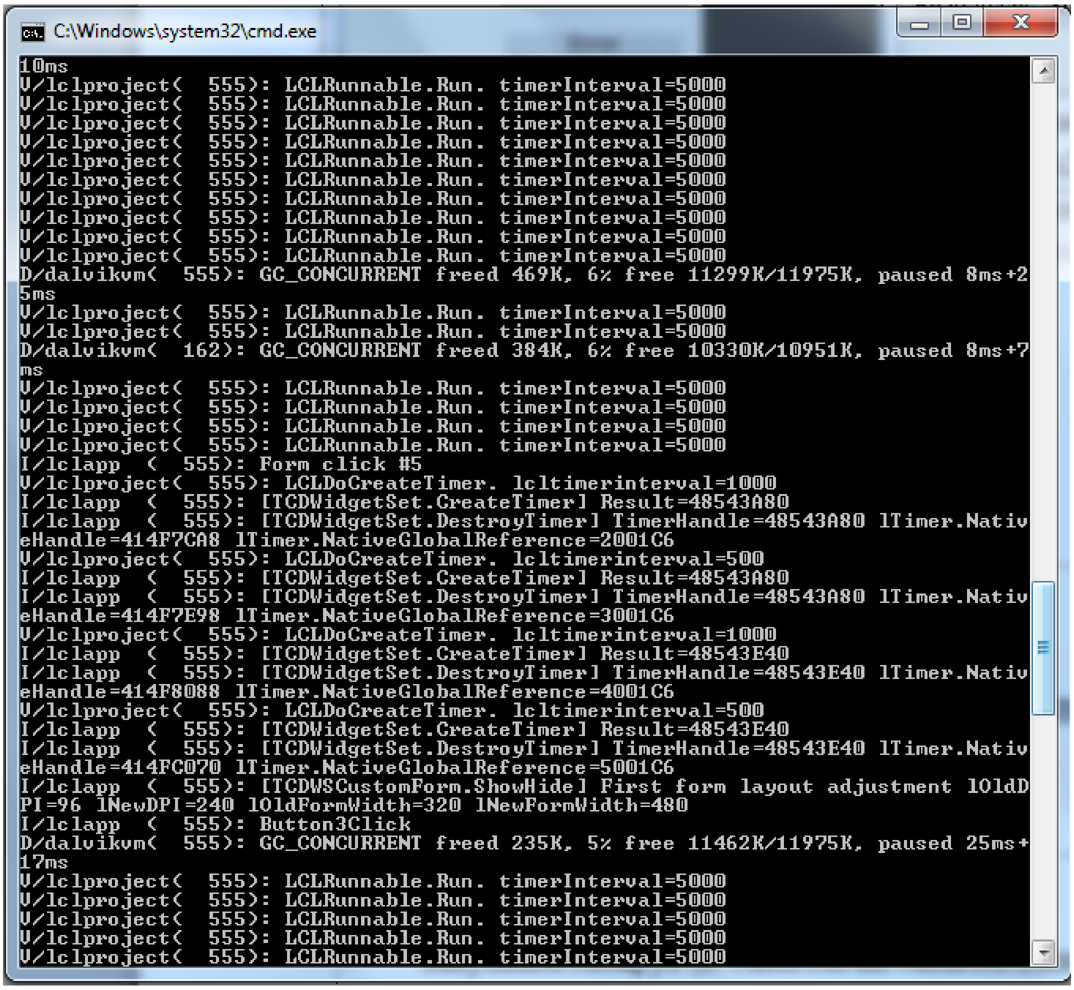


Главное окно программы



Окно, открывающиеся по кнопке «Open Form2»

Также будет интересно, если вы запустите программу «e:\Android\projects\androidlcl\android\adb_logcat.bat». Вы увидите лог работы ОС Android и запущенных приложений. Можно наблюдать результаты работы сборщика мусора ☺



Все заработало? Отлично!

Теперь запускаем Лазарус (e:\Android\laz4android\Lazarus.exe), открываем в нем проект «e:\Android\projects\androidlcl\androidlcltest.lpr» и компилируем (Ctrl+F9). Должен появиться файл «e:\Android\projects\androidlcl\android\libs\armeabi\liblclapp.so».

Далее запускаем файл «e:\Android\projects\androidlcl\android\build_debug_apk.bat»

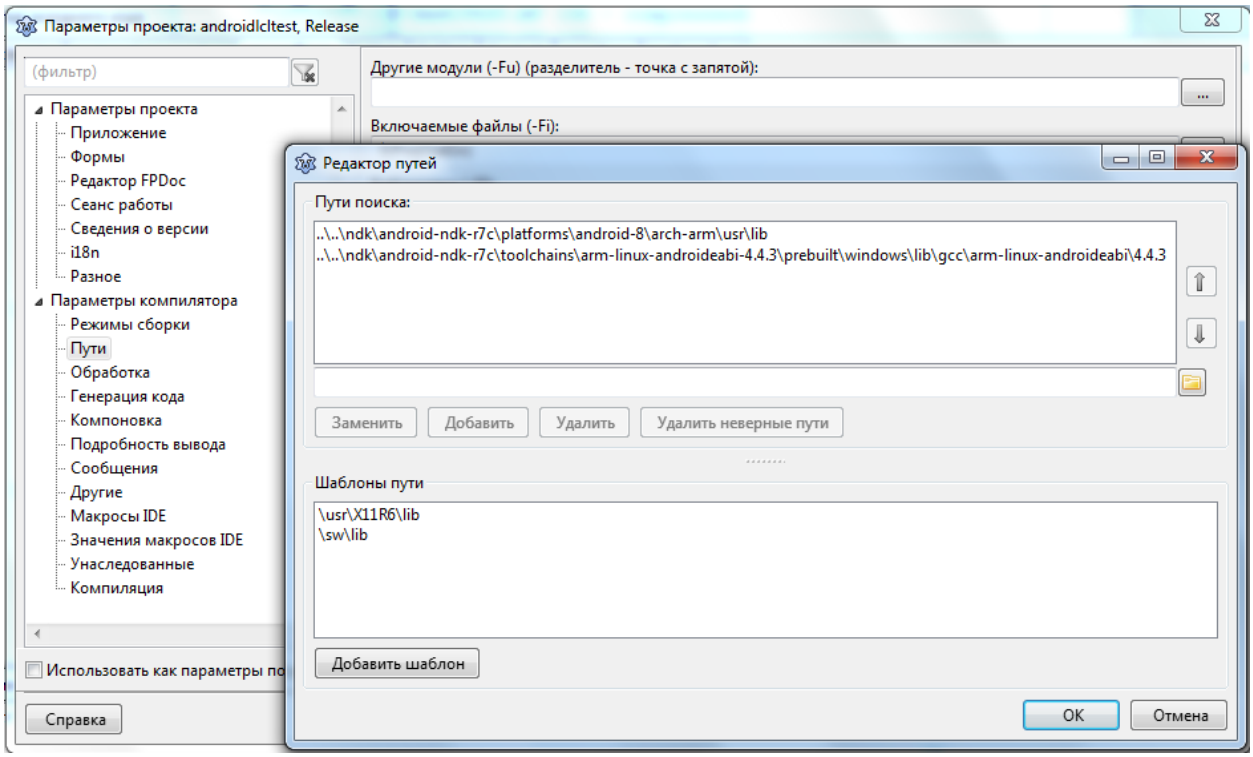
Он должен поработать секунд 15, после чего должен появиться файл «e:\Android\projects\androidlcl\android\bin\androidlcltest.apk».

Вывод файла «build_debug_apk.bat» должен быть примерно таким же, как в файле «build_result_example.log». Очень важно, чтобы был следующий текстовый вывод:

```
e:\Android\projects\androidlcl\android>zipalign -v 4 bin\androidlcltest-unaligned.apk bin\androidlcltest.apk
Verifying alignment of bin\androidlcltest.apk (4)...
      50 META-INF/MANIFEST.MF (OK - compressed)
     473 META-INF/LCLDEBUG.SF (OK - compressed)
     960 META-INF/LCLDEBUG.RSA (OK - compressed)
    1643 lib/armeabi/liblclapp.so (OK - compressed)
 1567694 AndroidManifest.xml (OK - compressed)
 1568684 resources.arsc (OK)
```

```
 1569664 res/drawable-hdpi/icon.png (OK)
 1572624 res/drawable-ldpi/icon.png (OK)
 1574632 res/drawable-mdpi/icon.png (OK)
 1577509 classes.dex (OK - compressed)
Verification succesful
```

Если такой вывод не появляется, значит что-то не так настроена. Проверьте настройки путей в проекте Лазаруса:



Пути, выставленный в файле «build_debug_apk.bat»:

```
SET PATH=e:\Android\sdk\tools;e:\Android\sdk\platform-tools\;C:\Program
Files (x86)\Java\jdk1.6.0_45\bin;E:\Android\sdk\build-tools\18.1.0\
SET APP_NAME=androidlcltest
SET ANDROID_HOME=e:\Android\sdk
SET APK_SDK_PLATFORM=e:\Android\sdk\platforms\android-14
SET APK_PROJECT_PATH=e:\Android\projects\androidlcl\android
```

Обычно именно здесь могут быть несовпадения!

Добейтесь, чтобы файл «build_debug_apk.bat» отрабатывал корректно!

Вот собственно и все!

Удачи!

Продолжение следует! ☺