MINISTRY OF DEVELOPMENT OF INFORMATION TECHNOLOGIES AND COMMUNICATIONS OF THE REPUBLIC OF UZBEKISTAN

TASHKENT UNIVERSITY OF INFORMATION TECHNOLOGIES NAMED AFTER MUHAMMAD AL-KHWARIZMI NUKUS BRANCH

Department of Information technologies

Computer engineering (Computer engineering) direction

Permitted to defence

Head of the department

Aytmuratov B.

_____

«___»_____ 2017 y.

DEVELOPED AN APPLICATION FOR SMARTPHONES RUNNING THE ANDROID OPERATING SYSTEM

GRADUATE QUALIFICATION WORK

Graduate: _____ Balatabev J.

Supervisor: _____ Mnajev B.

Nukus 2017

Content

Contents

Introduction

At this graduate diplomas research work they have meant applications for the smartphones running on the basis of operation system. This work consists of the introduction, 3 chapters, the conclusion and list of used literacies. In the introduction part there given short information about the all done works in this research. In the first chapter there is given information about possibilities of smartphones running on the basis of android system. For instance, for the programming language of Java, the example for making the basis of android system at the Eclipse compiler. So Delphi and C++ programming language Embarcadero Rad studio running android systems first applications making process is also planned. Embarcadero Rad studio makes possible work with IDE environment Delphi and C++ language operators.

At the third chapter it is planned to make application for smartphones running on the basis of android system with the help of android book maker program. Nowadays the process of making for android system stays one of the widespread issues.

The culmination of the theme: taking into consideration that the XXI century is the century of information technology, nowadays taking into consideration the widespread of phones and tablets, it is becoming very important to make programs in the Uzbek and Karakalpak language for them.

The aim of work is to make statistic information for phones running android system, and to make android applications in the Karakalpak language for the citizens.

The scientific innovation of the work, taking into consideration the difficulties of using Embarcadero Rad studio and Eclipse compilers, is given wide information about the possibilities of making statistic application for common users of the program Android Bookmaker.

I. Chapter. Advantages of Android system and phone

1.1. Advantages of Android operating system

From a commercial point of view Android:

• Does not require any certification of developers

• Provides access to the Google Play service (Android Market), where you can

Place and sell their programs

• allows you to develop your own version of the platform under your own brand, that is, it gives you complete control over the interface User.

From the developers' point of view:

• Android is focused primarily on developers

• Android is a powerful and intuitive platform for development, and thanks to this, programmers who have never dealt with the development of software for mobile devices can easily and quickly start creating their own full-fledged applications for Android Features and features inherent only in Android:

Google Map: The Google Map for Mobile service is very popular, and Android offers the ability to manage Google Map from your applications, allowing developers to create applications that make extensive use of cartographic capabilities.

- Background services allow you to create applications that implement events, but the oriented model, working seamlessly for the user, while he uses other programs or does not use the phone at all.

- Android allows applications to exchange messages, together

Process data and share it, and use these mechanisms to manage the data and functionality of applications built into Android. To reduce the risk of using such an "openness strategy", all processes, data stores and files are closed from external access, unless it is authorized by the resource management system and authorities.

- All applications in the system have the same status; Android does not highlight embedded applications among applications developed by third-party developers. This gives programmers (and users) unprecedented opportunities to change the appearance and functionality of devices, since they can replace any embedded applications with alternate ones that eventually have the same access to system data and hardware.

- Using widgets, "live directories", "live wallpaper" you can create "windows" in your applications directly on the "Desktop", and the Quick Search Bar allows you to include the results of your applications in the phone’s search engine.

Android Architecture

In terms of architecture, Android is a software stack for mobile devices, which includes the operating system, middleware and key applications.

A set of development tools for Android (Android SDK) contains the tools and application programming interfaces (APIs) required to develop applications on the Android platform using the Java programming language.

Android platform features

The composition of Android includes a variety of components that are interesting for developers:

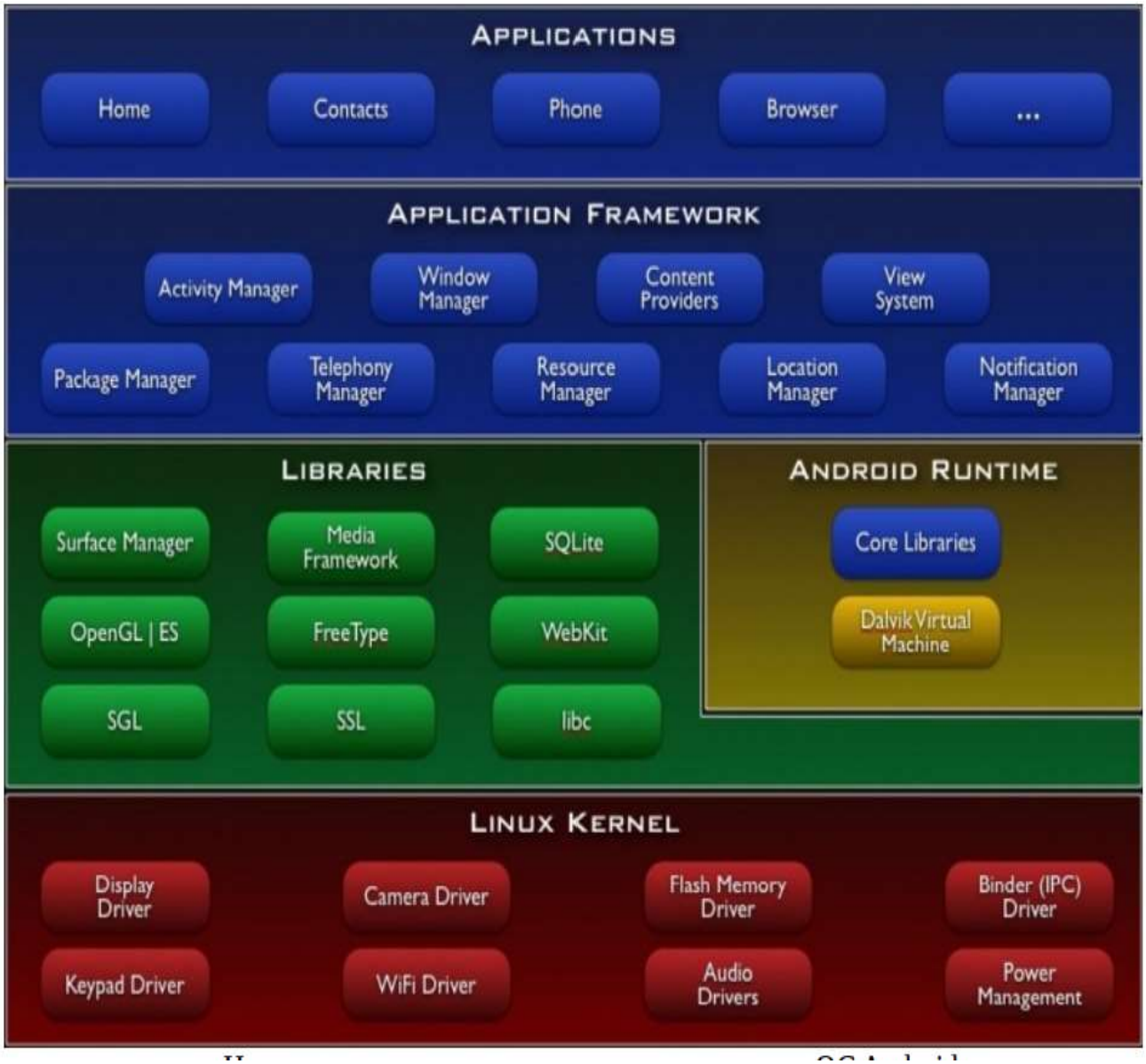
- An application framework that allows you to reuse and mutually replace software components.
- Virtual machine Dalvik, optimized for work on mobile Devices, which is an intermediate layer. Through the use of Virtual machine for executing program code, developers get at their disposal an abstraction level that allows not to worry about the design features of various devices.
- Built-in web-browser, based on the open-source engine WebKit, provides the ability to use the rendering of html-resources in the developed applications.
- Optimized graphics subsystem based on its own library for 6 works with two-dimensional graphics, 3D graphics is implemented on the basis of the OpenGL ES library (with possible support for hardware acceleration).
- Relational fault-tolerant embedded SQLite DBMS for storing and processing structured data.
- Built-in support for various audio and video formats, as well as static images (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- Depending on the equipment, GSM and CDMA telephony, as well as wireless data transmission technologies (Bluetooth, EDGE, 3G and Wi-Fi) can be supported.

- Support for one or more cameras with the ability to take photos / videos, GPS, compass and accelerometer.

- A powerful development environment, which includes, among other things, an emulator (virtual)

Devices with different (configurable) characteristics, tools for Debugging and profiling applications, as well as the ADT plug-in for IDE Eclipse.

Android Core Components



The diagram shows the main components of the Android OS. Simplified, the Android software stack can be thought of as a combination of a modified Linux kernel for mobile application and a set of C / C ++ libraries that are available in the application

framework. The framework provides management and operation of the runtime environment and applications.

Description of the main components of Android

- Applications: Android comes with a set of basic applications, including an email client, a calendar, maps, a browser, a program for managing contacts, and others. All these applications are written using Java programming.

- Framework: Providing an open platform for development, Android gives
Developers the ability to create powerful and innovative applications.
Developers can use the latest hardware, get information about the current location, run background services, set alarms, add notifications in the status bar, and much, much more. Developers have full access to the same API, which are used when creating embedded applications. The Android architecture is designed to simplify the reuse of software components, any application can "publish" its functionality, and any other applications can then use these features (given the security constraints imposed by the framework). The same mechanism allows users to replace software components, if necessary. The framework includes a set of services and systems that underlie all applications:

- A rich and extensible set of views (View), which can be
Used to create applications, including lists (ListView), text Fields (TextView), buttons (Button) and even embedded web browser (WebView) and Maps (MapView).

- Content Providers, which allow applications
Access data from other applications (for example, contacts), or Share their data.

- A resource manager (Resource Manager) that provides access to resources that do not A program code, such as localized strings,
Graphic resources (drawable) and markup files (Layouts).

- Notification Manager, which allows all
Applications to display custom notifications in the status bar Mobile device.

- Manager "Activities" (Activity Manager), which manages the life cycle of applications and provides the ability to switch between different "Activities".

- Libraries: Android includes a set of C / C ++ libraries used

Different components of the system. The framework provides developers Capabilities of all these libraries. Some of the major libraries are described below:

- The C system library (libc) is based on the libc code, borrowed from

BSD, the implementation of a standard system library, optimized for Mobile devices based on the Linux kernel.

- Media library - based on the OpenCore framework

PacketVideo, the library provides work with many popular Audio, video and image formats.

- SurfaceManager - controls access to the display and

Provides a "seamless" overlay of 2D and 3D graphics from several Applications.

- LibWebCore - a modern engine, used both as built-in Android

Web browser, and components within applications (WebView).

- SGL - the main mechanism for displaying two-dimensional graphics.

- 3D libraries - implement the API based on the OpenGL ES API; these

Libraries use either hardware acceleration of 3D graphics (if available Hardware accelerator) or built-in optimized software Rasterizer of three-dimensional graphics.

- FreeType - provides raster and vector rendering of fonts

- SQLite is a powerful and easy to use relational database engine, available to all

- Operating environment (runtime environment, RTE):

- Android includes a set of libraries that provides the majority

Runtime functions available in the main libraries of the programming language Java.

Every Android application runs in its own process, with its own copy of the virtual machine Dalvik. VM Dalvik was Designed and written in such a way that inside the mobile Devices could effectively run multiple virtual machines. The Dalvik virtual machine can execute programs in the executable Format DEX (Dalvik Executable). This format is optimized for Use the minimum amount of memory. Executable file with The *.dex extension is created by compiling Java classes using Tool dx, which is part of the Android SDK. When using the IDEEclipse and the ADT plugin (Android Development Tools) compile Java classes into The *.dex format occurs automatically.

The Dalvik virtual machine relies on the Linux kernel to perform basic system functions, such as many threads, I / O, and low-level memory management.

- Linux kernel: Android uses Linux version 2.6 to provide basic system services such as security, memory management, process management, network stack and driver handling. The kernel also acts as an additional layer of abstraction between the hardware of the mobile device and the rest of the software stack**.**

Security and Authorization (Permissions)

As mentioned above, the dx tool from the Android SDK compiles applications written in Java into the executable format of the Dalvik virtual machine. In addition to directly executable files, the Android application includes other auxiliary components (such as, for example, data files and resource files). The SDK packs everything you need to install the application into a file with the *.apk extension (Android package). All code in one *.apk file is considered to be one application, and this file is used to install this application on devices with Android OS.

Once installed on the device, every Android application lives in its own sandbox (security environment):

- The Android system is a multi-user Linux system, in which each

The application operates with unique user rights. In Android, a system of basic security and access control is typical for Unix-like OSes, in which a) everything in the system is a file that necessarily belongs to some user (has a corresponding User ID, UID) and b) any process in the system necessarily works With the rights of some user (also has its own UID); By associating the UIDs of the process and the file, Unix security allows or denies the operation requested by the process with a particular file.

- By default, the system assigns to each application a unique UID (this ID is only used by the operating system and is unknown Application). The system sets the access rights for all application files in such a way that only processes with UID installed for this application can access them.

- Each process runs inside its own Dalvik virtual machine, so that application code runs in isolation from other applications.

- By default, each application runs in its own Linux process. Android creates a process when you want to execute any of the application components, and destroys the process when it is no longer needed or when the system must provide memory to other applications. With the help of the described mechanism, Android implements the principle of least privileges: each application by default has access only to those components that are required to perform its tasks, and no more. This creates a very secure environment in which the application can not access parts of the system for which it does not have permission.

However, for applications, there are mechanisms for exchanging data with other applications and for accessing system services:

- Two (or more, if necessary) applications can be assigned the same UID, in which case they can access each other's files. To save system resources, applications with the same UID can also be configured to run in the same Linux process in the same virtual machine (while applications must be signed with the same developer certificate).
- The application can request permission (Permissions) to access the data in the Device, such as user contacts, SMS messages, removable storage media (SD cards), camera, Bluetooth, and much more. All such requests for access to resources are verified and resolved by the user in the Android OS during the installation of the application. If the application is installed, the user will no longer be asked to review these permissions.

Required credentials (as well as many other things) are specified in the Application Manifest File (AndroidManifest.xml). This file must be present for any Android application. Android SDK provides the developer with a rich arsenal of tools to manage the contents of this file. In addition, since the manifest file is a regular XML file, the necessary information in it can be specified using a regular editor (SDK, in which case it allows editing this file with great comfort). To use system resources and other applications, you need to add <uses-permission> tags to your application's manifest, specifying the required credentials. Standard names for describing the permissions used by System Activity and services in Android can be found in the android.Manifest.permission class in the form of static string constants and at

http://developer.android.com/reference/android/Manifest.permission.html

Below is an example of an application requested by the application to access the Internet:

<Manifest xmlns: android = http://schemas.android.com/apk/res/android package = "com.android.app.myapp">

<Uses-permission android: name = "android.permission.INTERNET" />

...

</ Manifest>

1.2. Installation and configuration of the components of the development environment

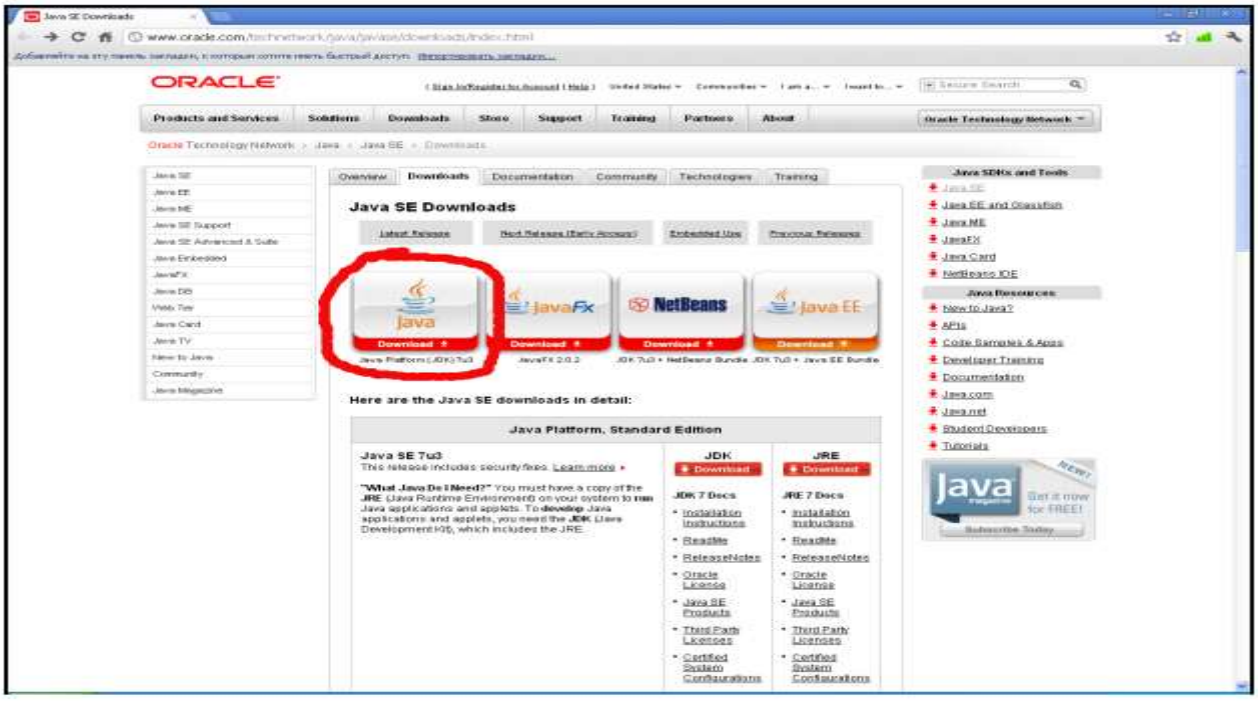
Applications for Android, like most applications for communicators, are developed on a standard PC where resources are unlimited (as compared to a mobile device) and downloaded to the target communicator for debugging, testing and subsequent use. Applications can be debugged and tested on a real device running Android or on an emulator. For initial development and debugging, it is more convenient to use the emulator, and then perform the final testing on real devices.

The developer is given the opportunity to use the application development tools for Android on a PC running any of the most common operating systems: Windows, Linux and Mac OS X. The process of installing the components of the development environment under Windows XP will be described in detail below; for other operating systems, the differences are very small.

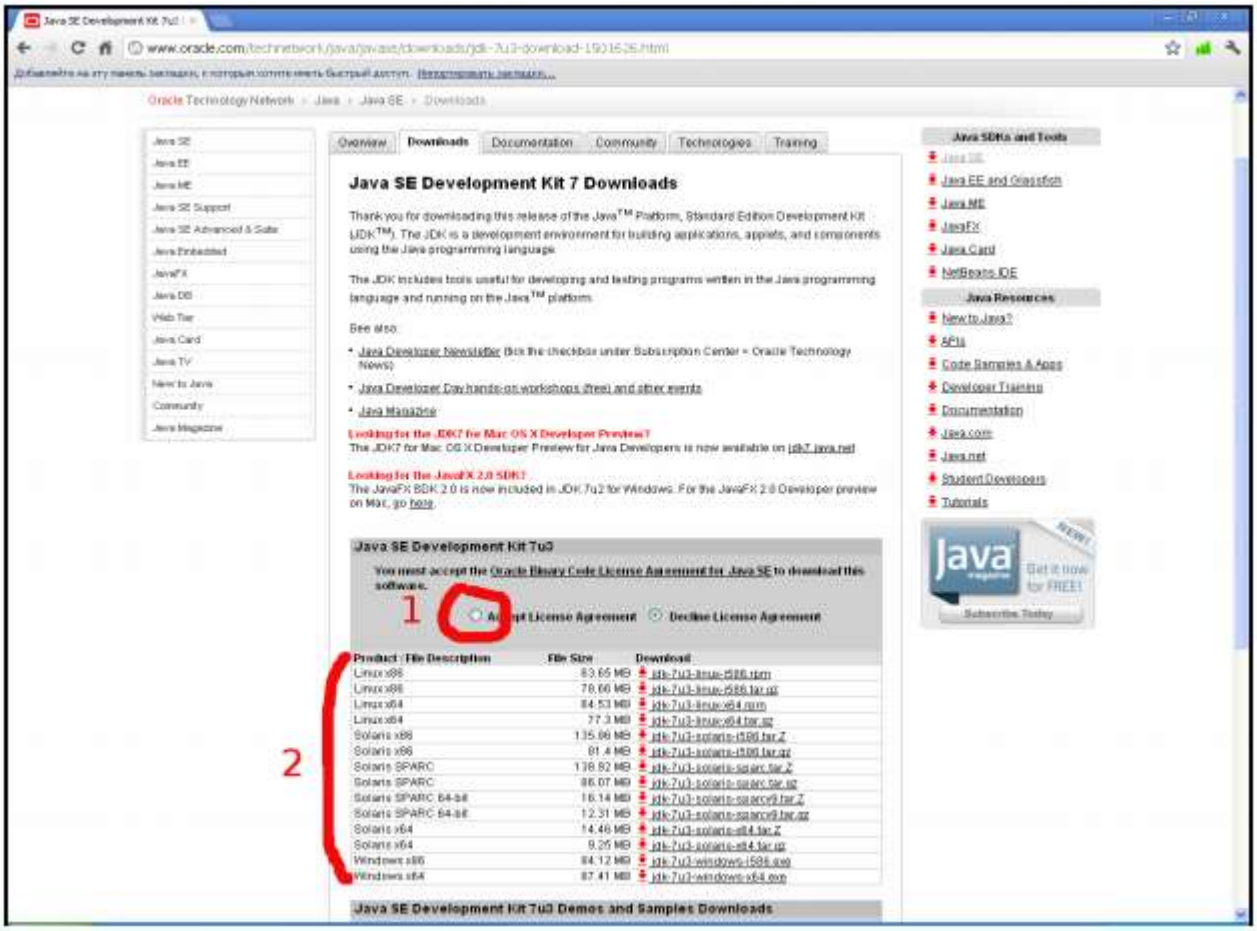
Installing JDK

The Android SDK requires a JDK version of at least 5 (at the time of writing this tutorial was relevant to the 7th version of Sun JDK and the 6th version of the Open-JDK). For Windows, Sun JDK is traditionally used, which can be downloaded for free on the developer's site: Download Page Sun JDK

http://www.oracle.com/technetwork/java/javase/downloads/index.html

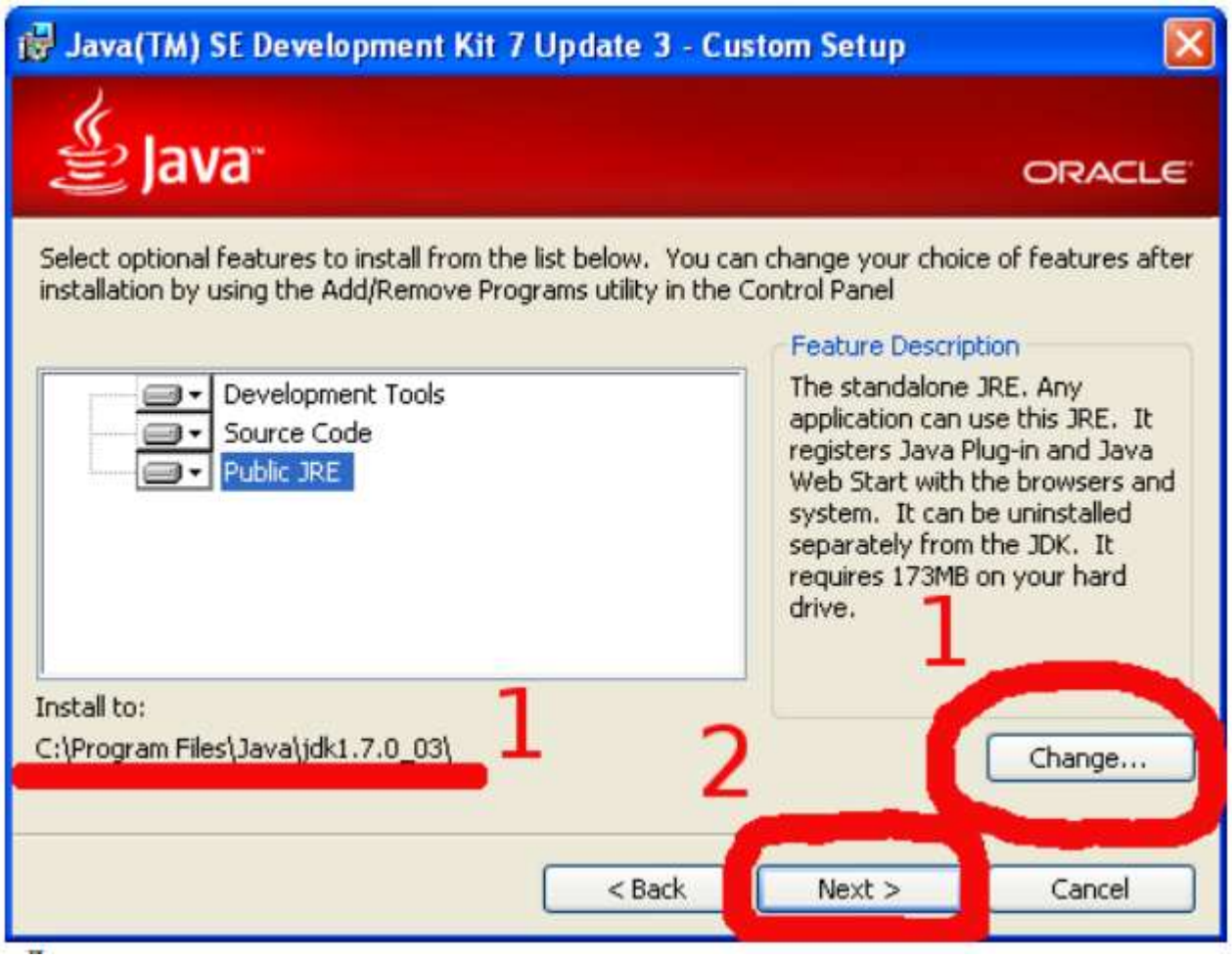


After accepting the terms of the Oracle Binary Code License Agreement for Java SE (1) license agreement, you can select the appropriate link for your platform (2):

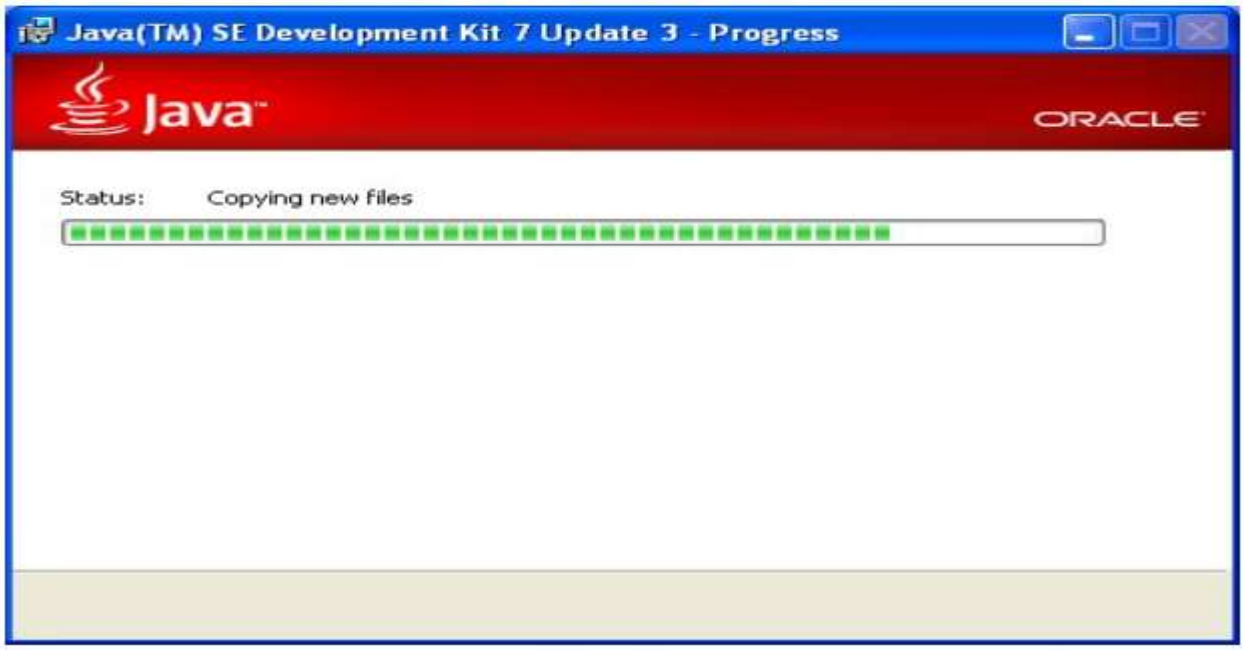


Installing the downloaded JDK is fairly simple:

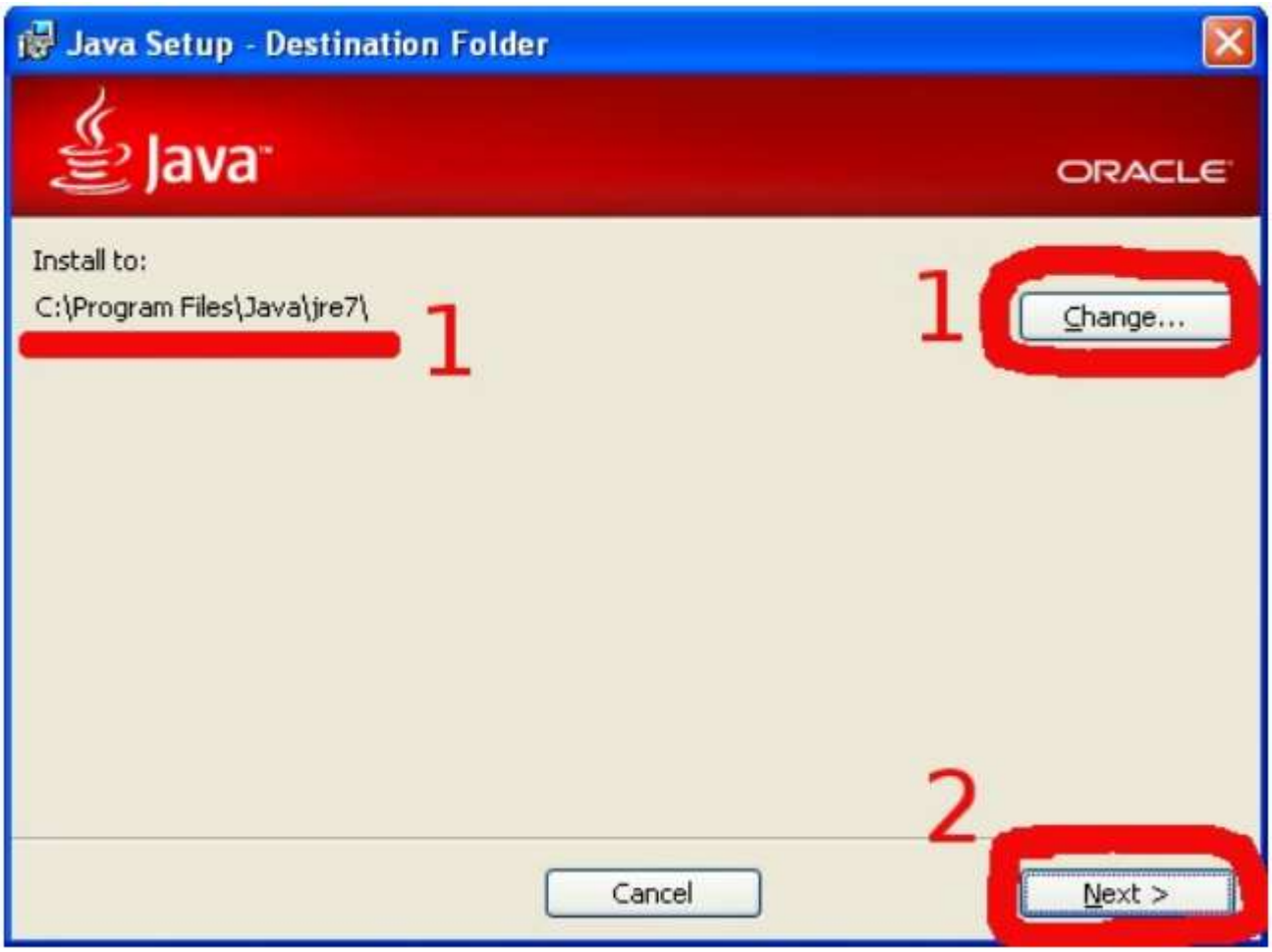
If the default installation directory (1) is not suitable, you can change it (2), and then continue with the installation:



Further everything is quite expected:

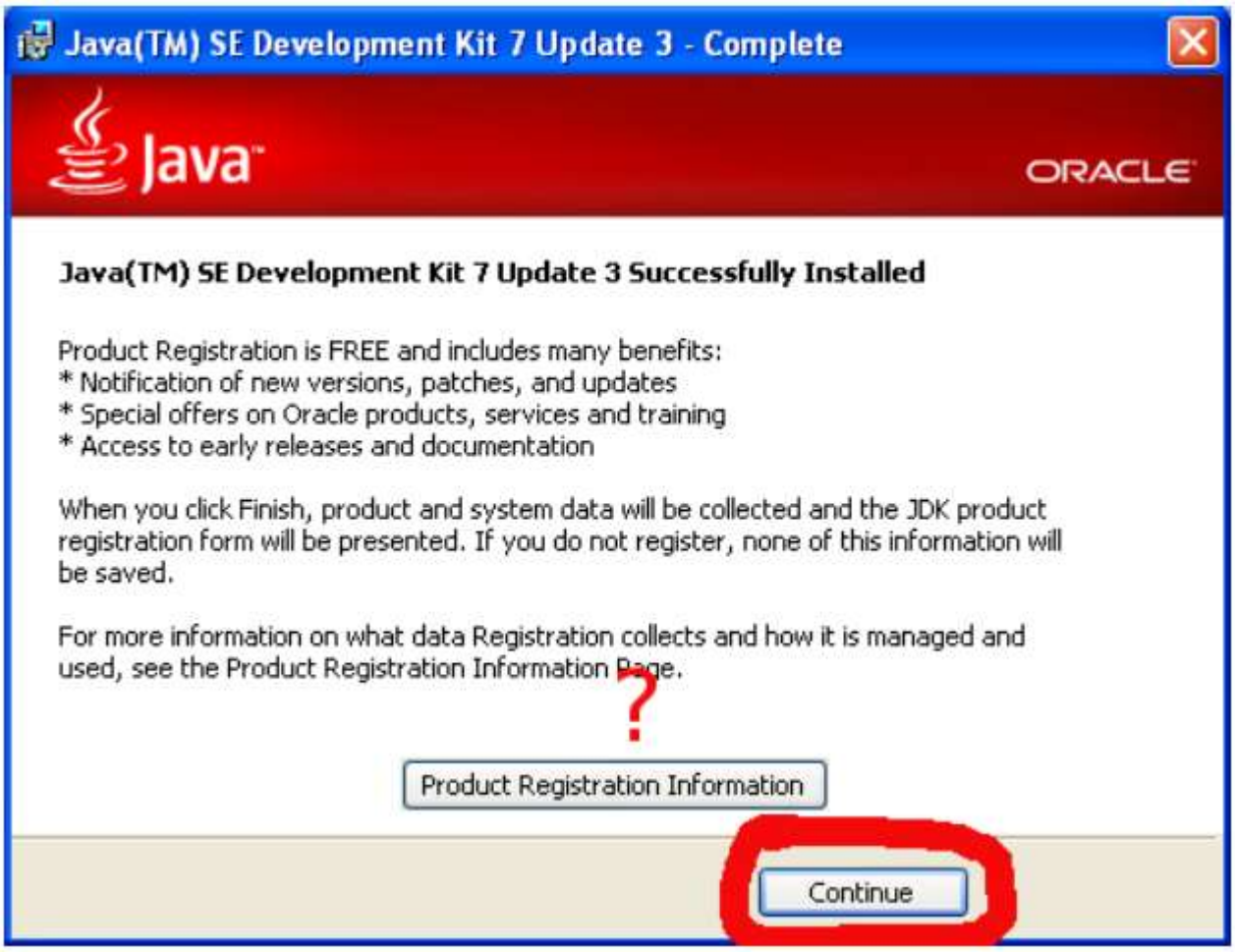


When installing Java Runtime, you can also change the installation directory:



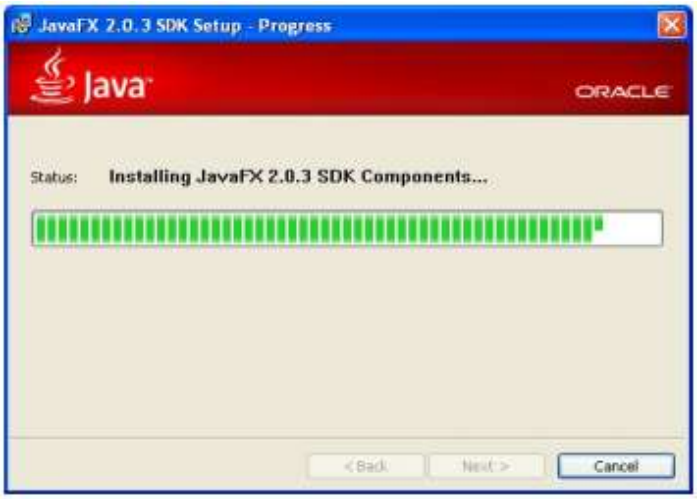
And continue the installation:

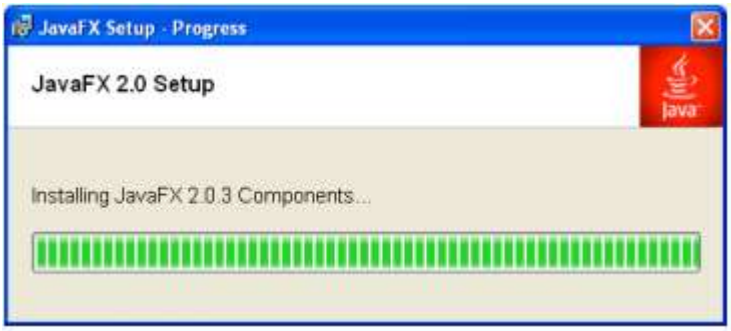
Further, if desired, you can register the installed product:



And install the Java FX SDK, if not a pity 50 MB of disk space:


JavaFX SDK Setup
Java
ORACLE
Setup Options
To install the SDK in this folder, click Next.
The SDK will need approximately 50MB of disk space.
Folder:
C:\Program Files\Oracle\JavaFX 2.0 SDK\
Browse...
1
2
< Back
Next >

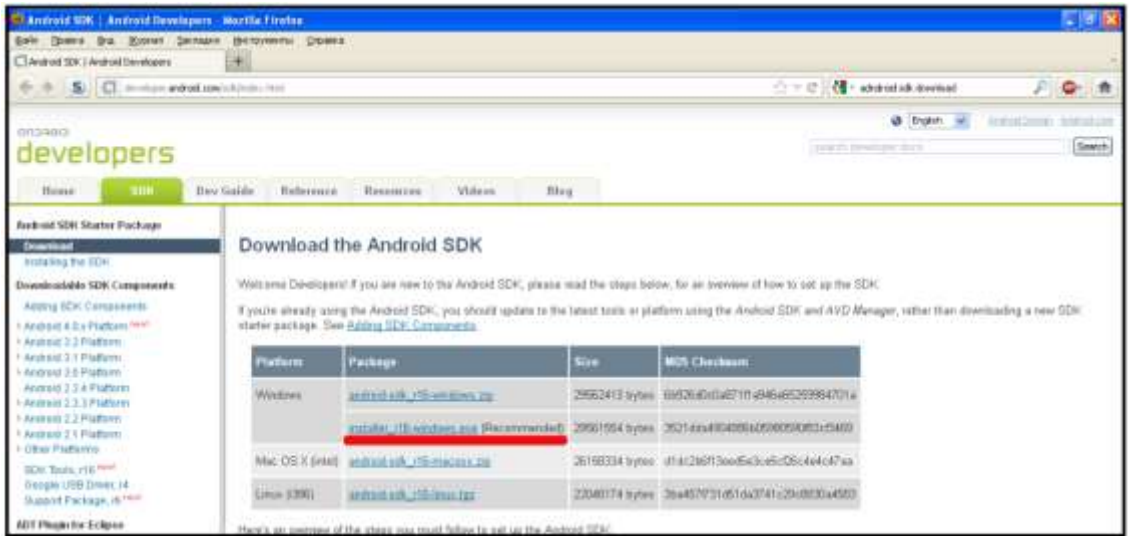

JavaFX 2.0.3 SDK Setup - Progress
Java
ORACLE
Status:
Installing JavaFX 2.0.3 SDK Components...
< Back
Next >
Cancel

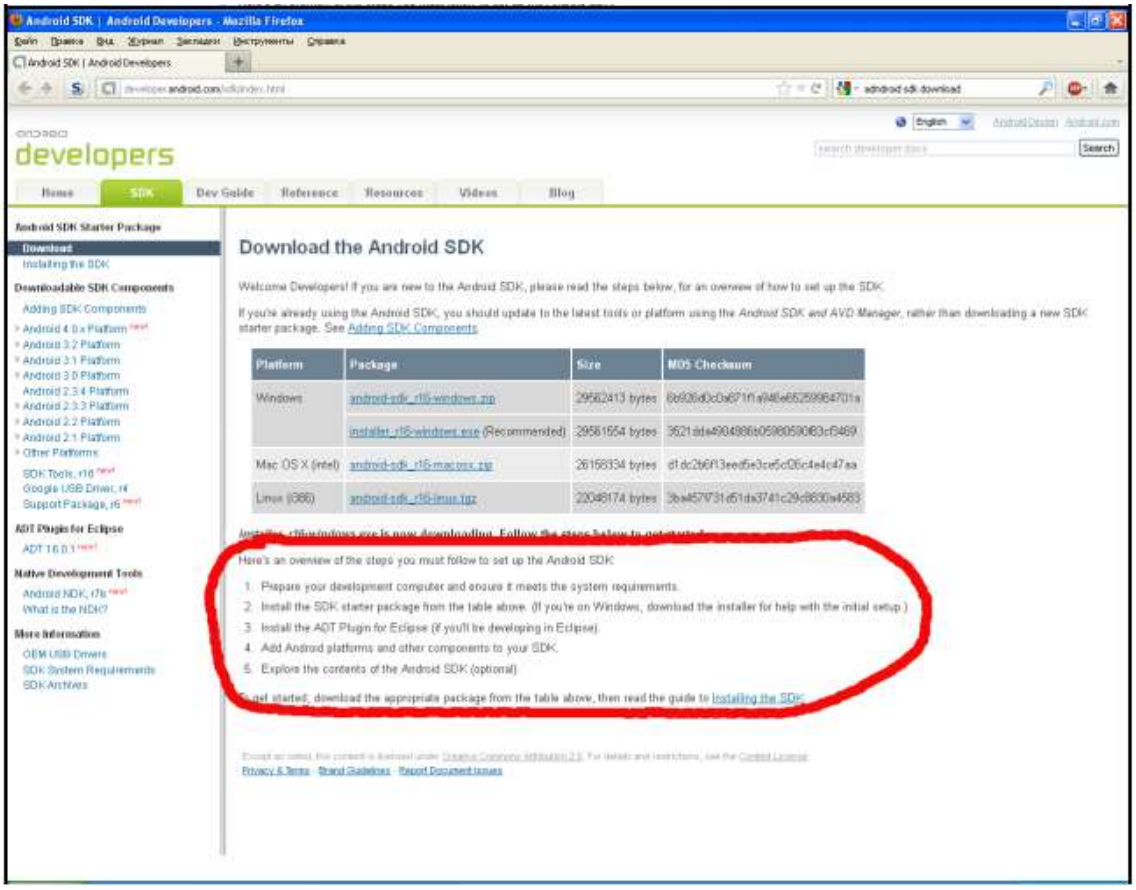

JavaFX Setup - Progress
JavaFX 2.0 Setup
Java
Installing JavaFX 2.0.3 Components...


JavaFX SDK Setup - Complete
JavaFX SDK Setup Complete
Java
ORACLE
Click close to exit the Setup Wizard
Close

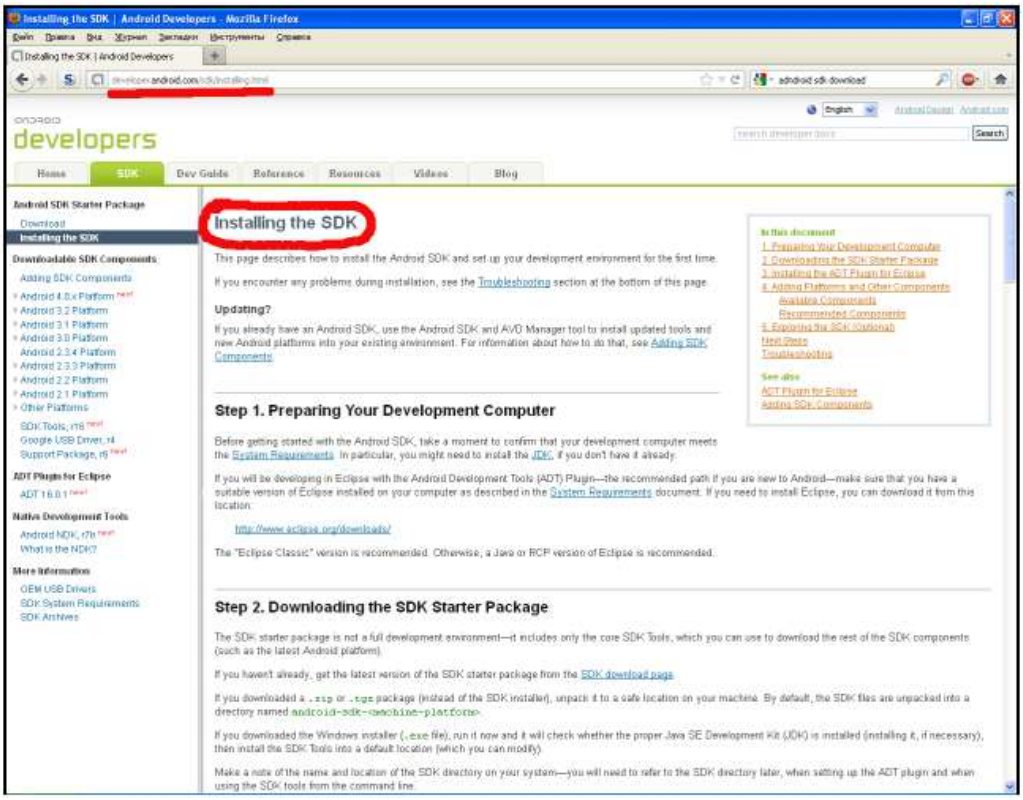
Install the Android SDK

On the page http://developer.android.com/sdk выбираем installer_r16-windows.exe





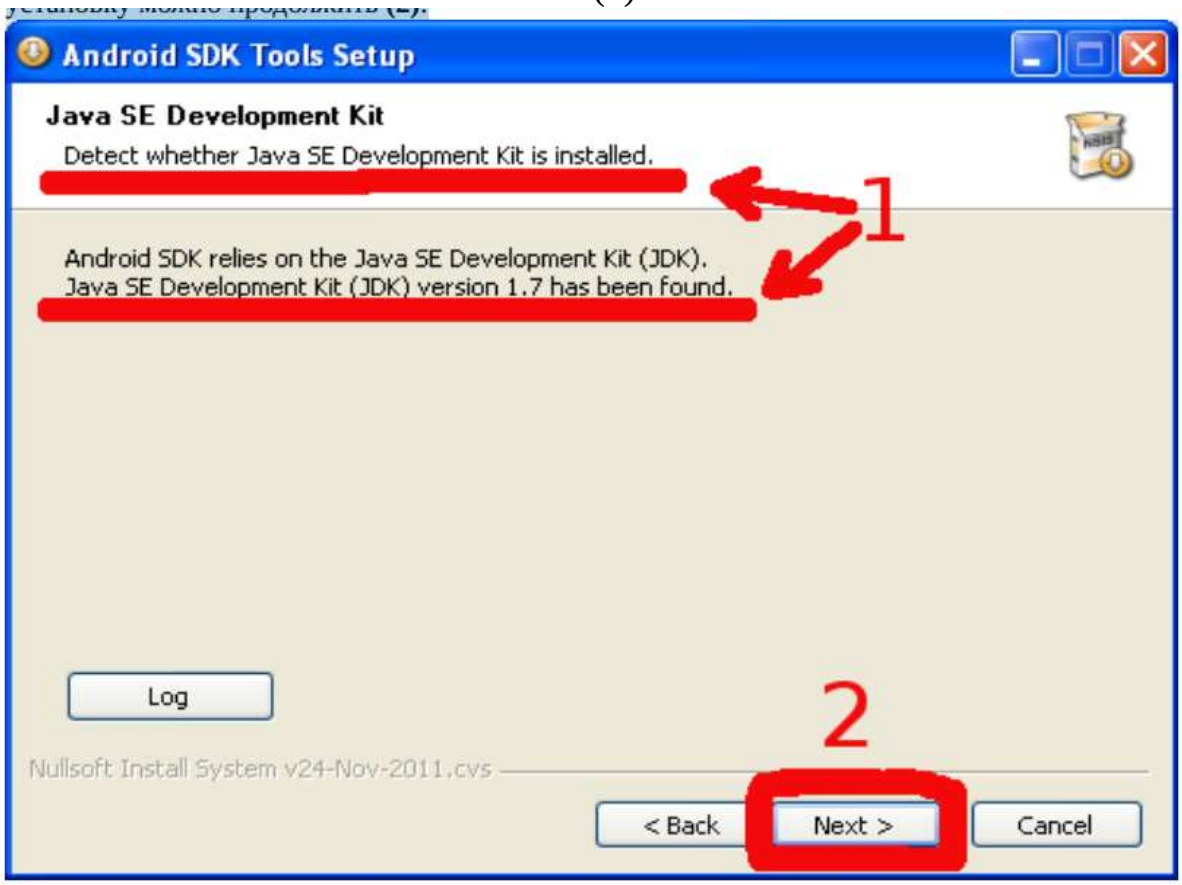
There are also additional installation instructions, as well as a link to the step-by-step instructions for installing the SDK on all supported platforms: http://developer.android.com/sdk/installing.html



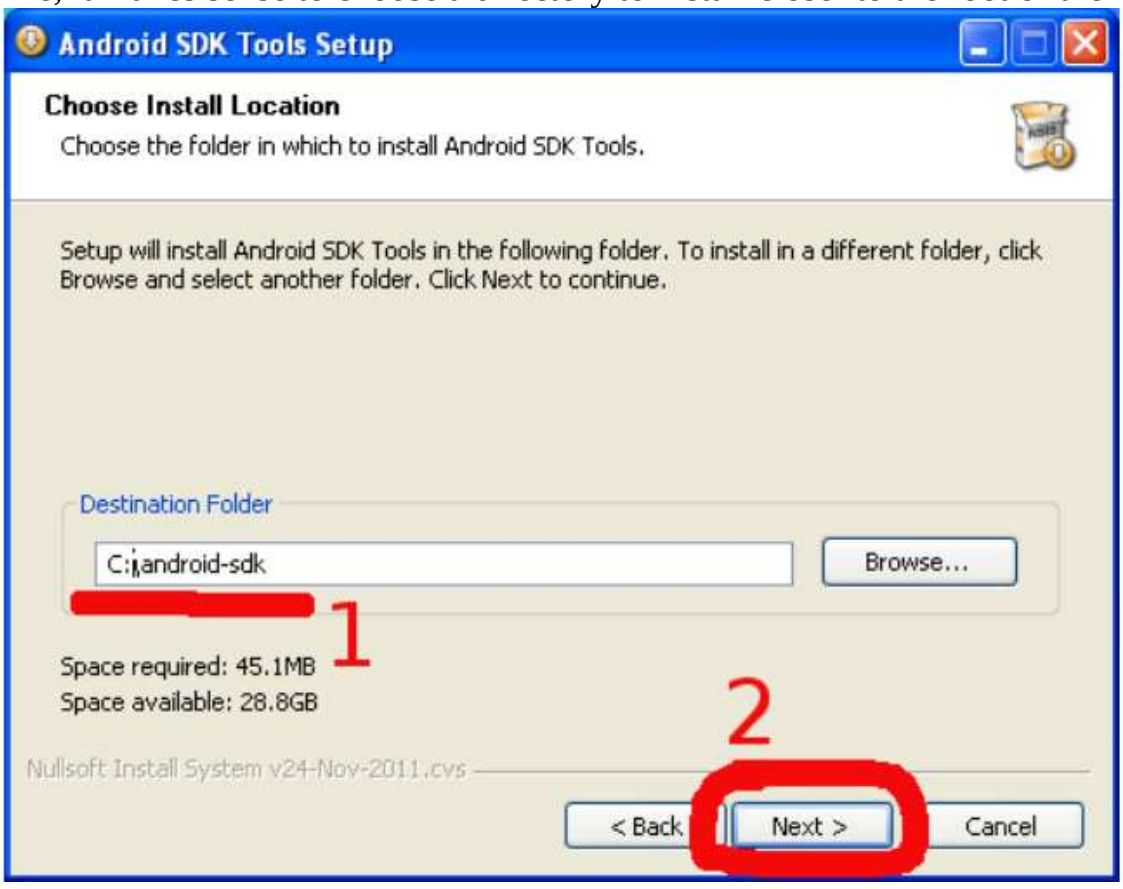
The http://developer.android.com resource is the main source of information about the Android platform "first-hand". It contains a lot of the most diverse information the developer needs, from the description of the API to things such as recommendations for application design and application performance. In this methodical manual in the future, there will be links to specific pages devoted to the topics under study.

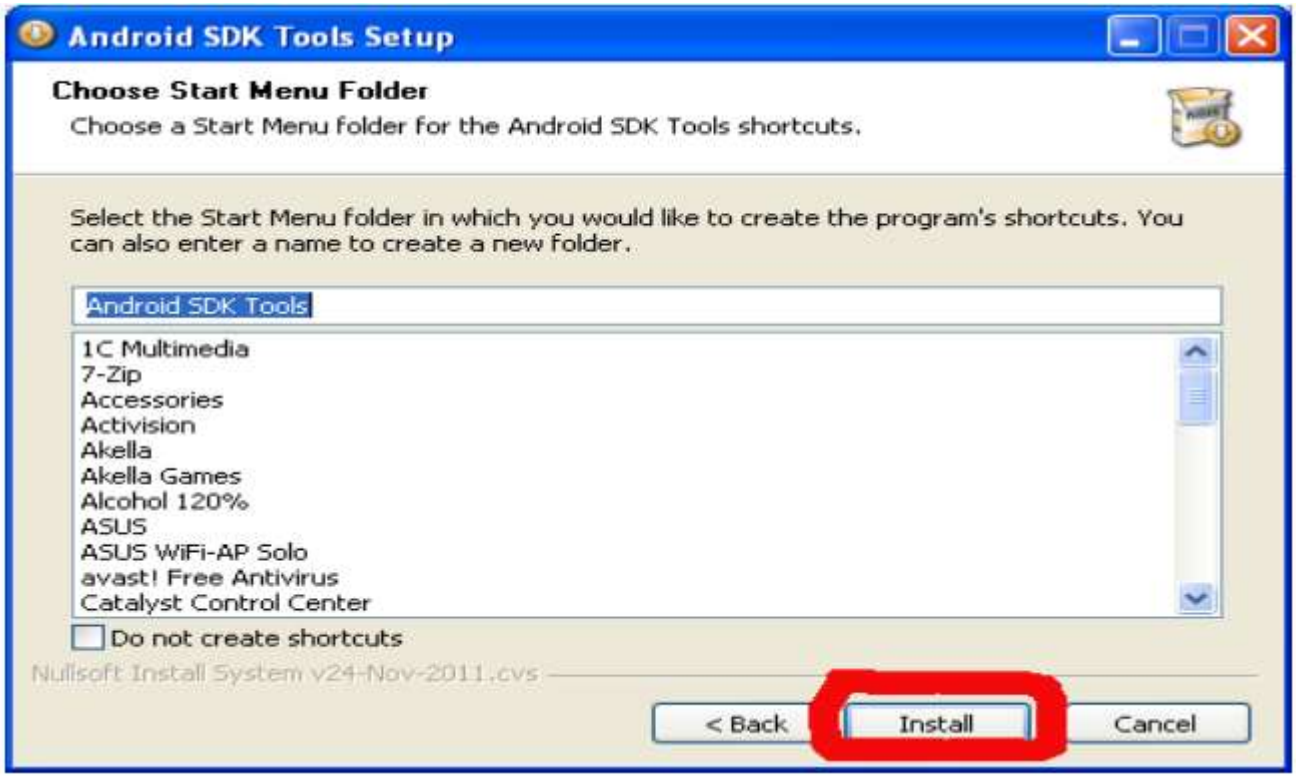
Installing the downloaded Android SDK is also not particularly difficult:

The installation wizard will detect (if it can) the installed version of JDK (1), after which the installation can be continued (2):

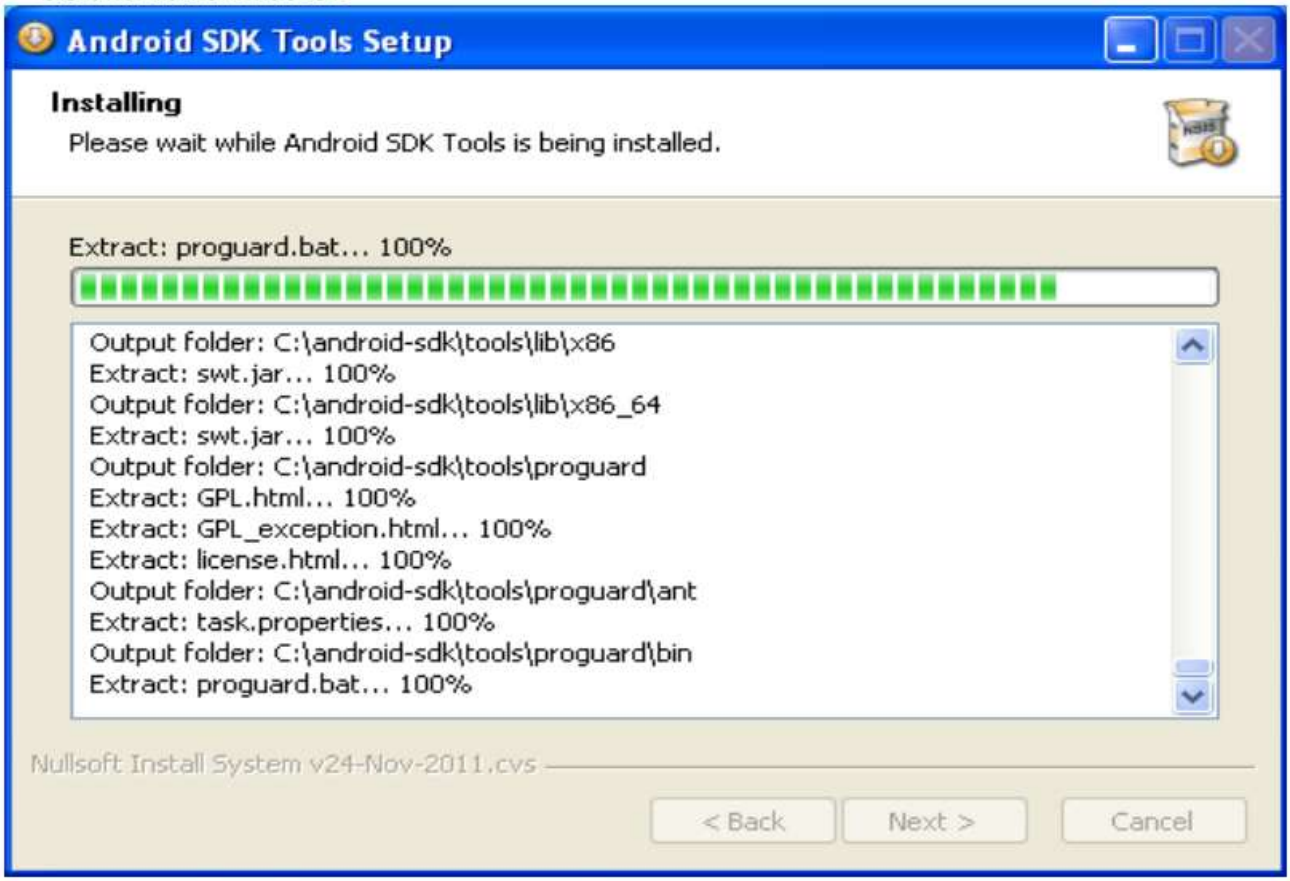


If you plan to run some of the tools included in the Android SDK from the command line, it makes sense to choose a directory to install closer to the root of the file system:

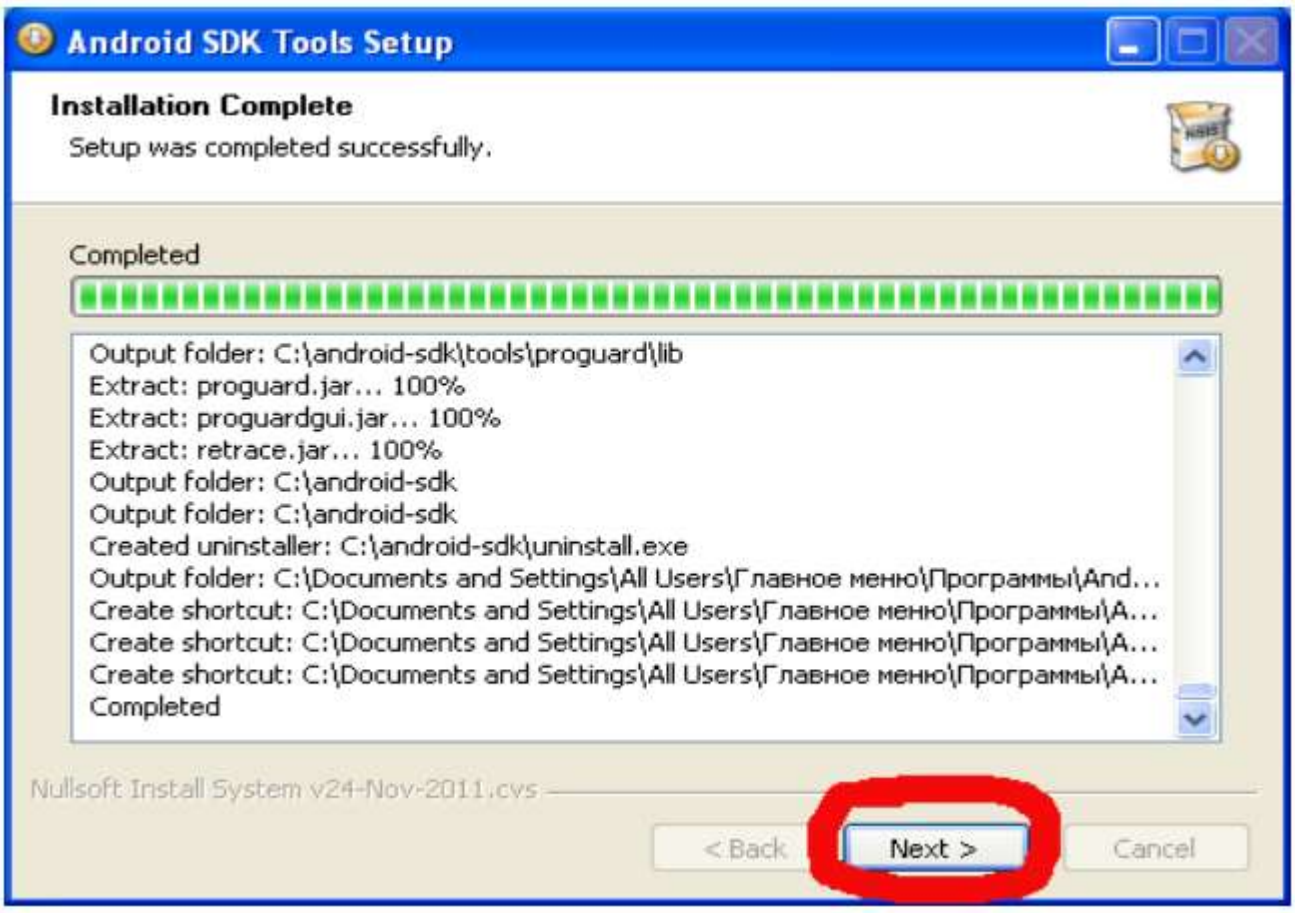




Installation starts

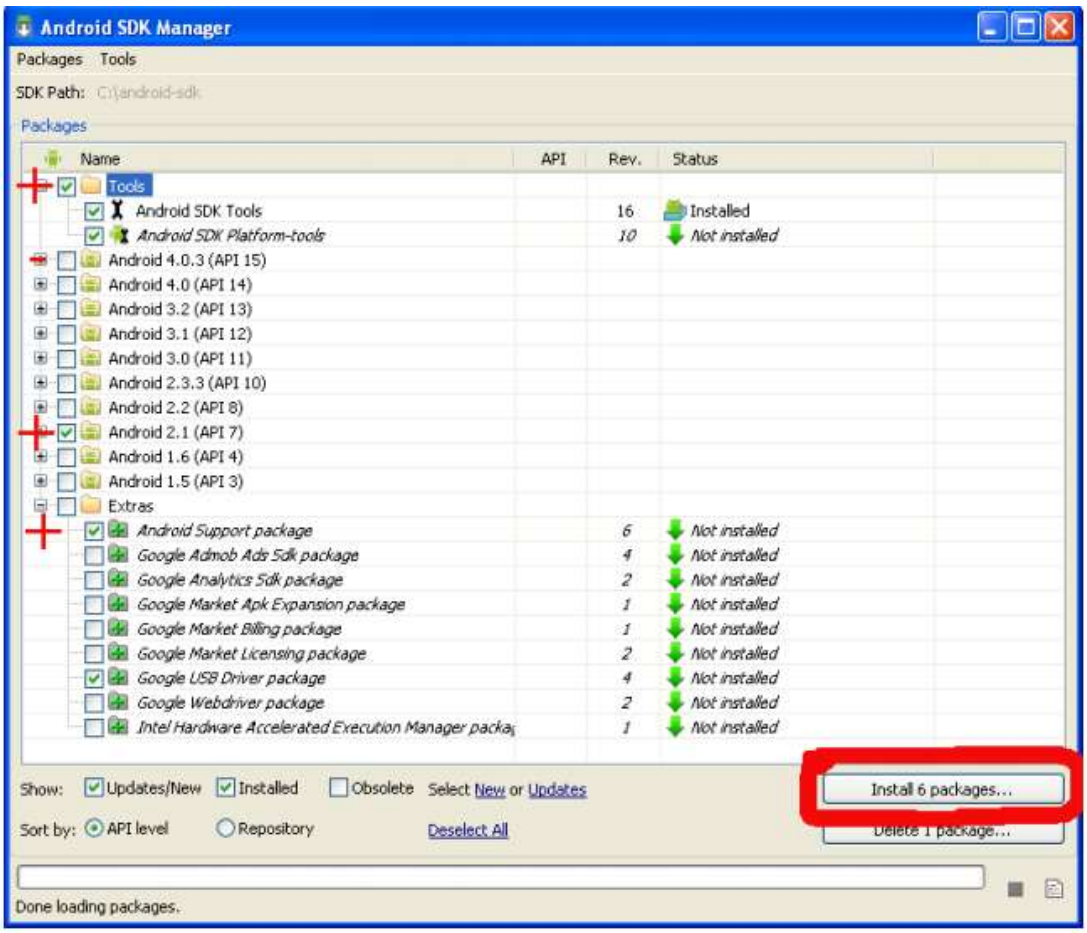


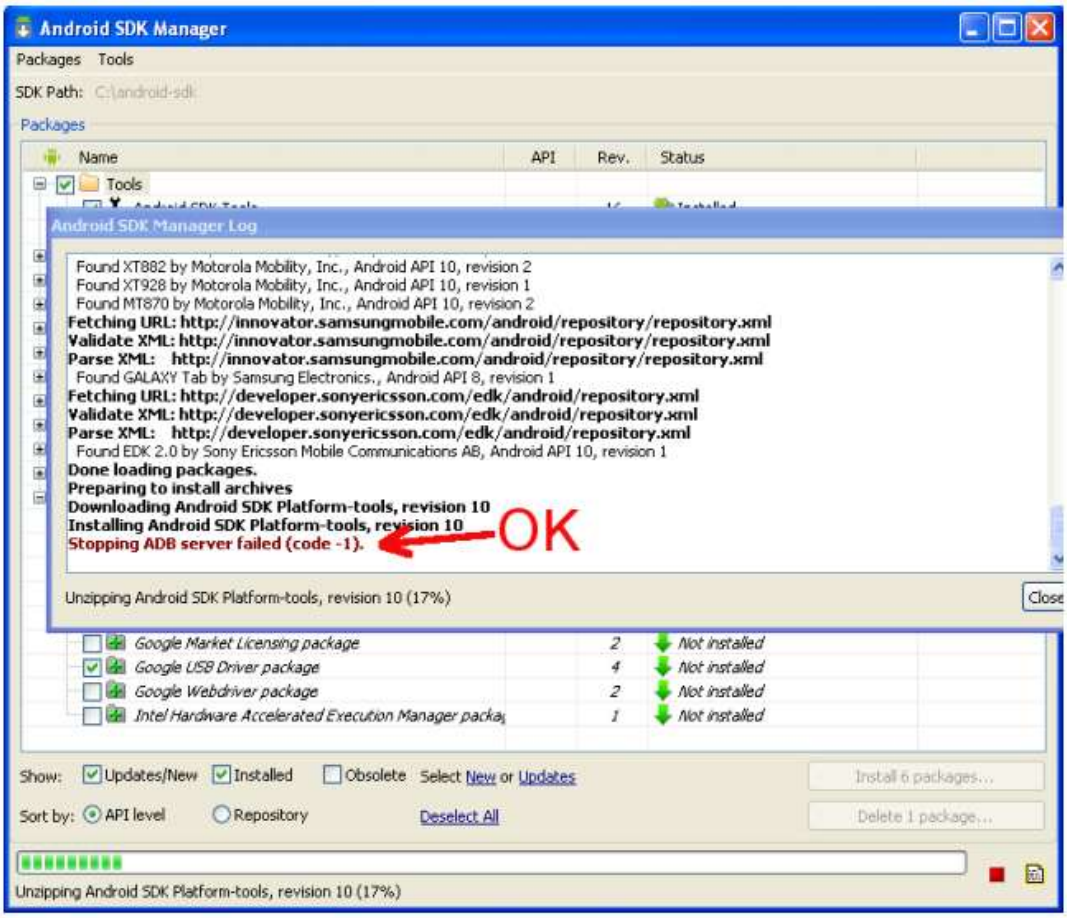
And ends:

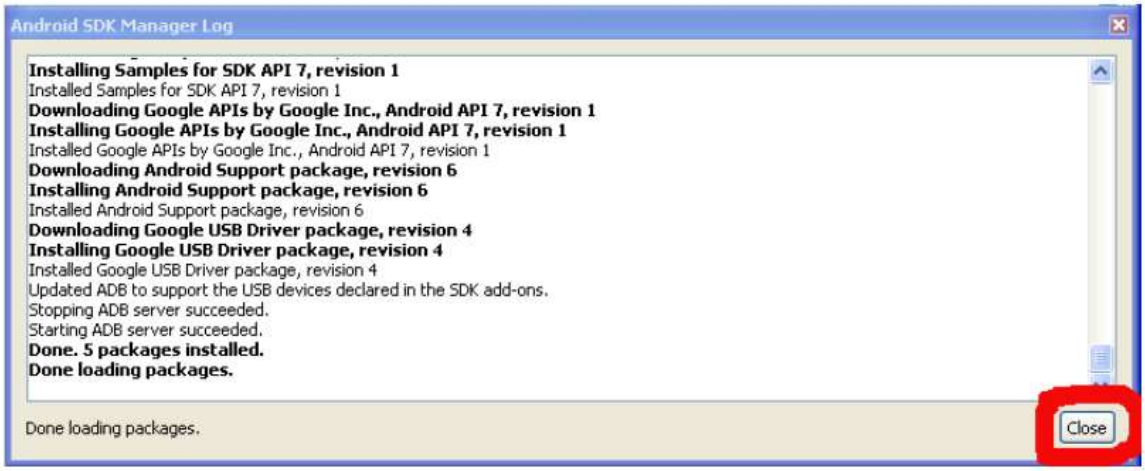


You can immediately put a daw (1) and run the SDK Manager for the final configuration (2):

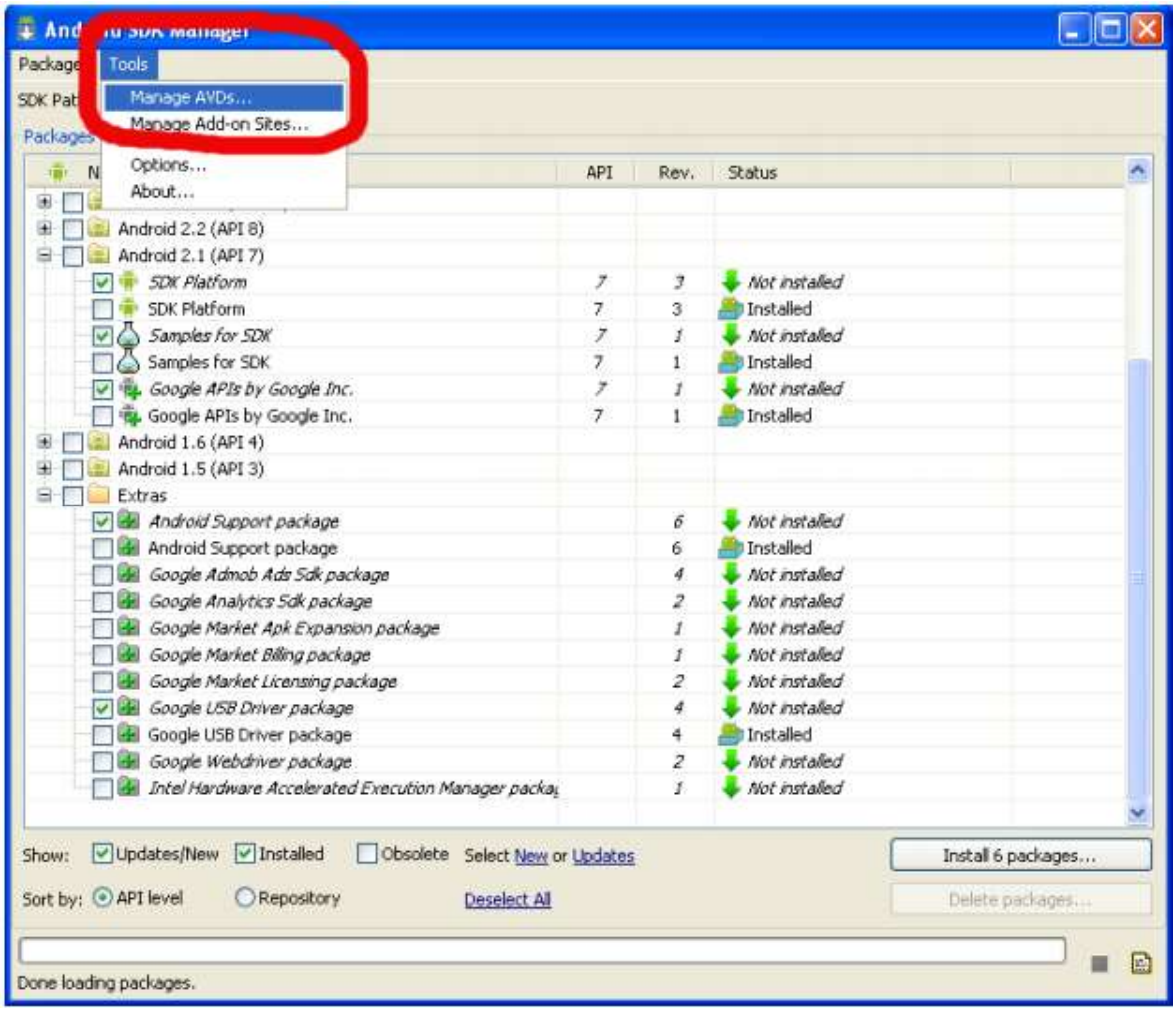
You need to download the required API versions and other useful tools:


Android SDK Manager
Packages Tools
SDK Path: C:\android-sdk
Packages
Name API Rev. Status
Tools
Android SDK Tools 16 Installed
Android SDK Platform-tools 10 Not installed
Android 4.0.3 (API 15)
Android 4.0 (API 14)
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 package 6 Not installed
Google Admob Ads Sdk package 4 Not installed
Google Analytics Sdk package 2 Not installed
Google Market Apk Expansion package 1 Not installed
Google Market Billing package 1 Not installed
Google Market Licensing package 2 Not installed
Google USB Driver package 4 Not installed
Google Webdriver package 2 Not installed
Intel Hardware Accelerated Execution Manager packa 1 Not installed
Show: Updates/New Installed Obsolete Select New or Updates
Sort by: API level Repository Deselect All
Install 6 packages...
Delete 1 package...
Done loading packages.

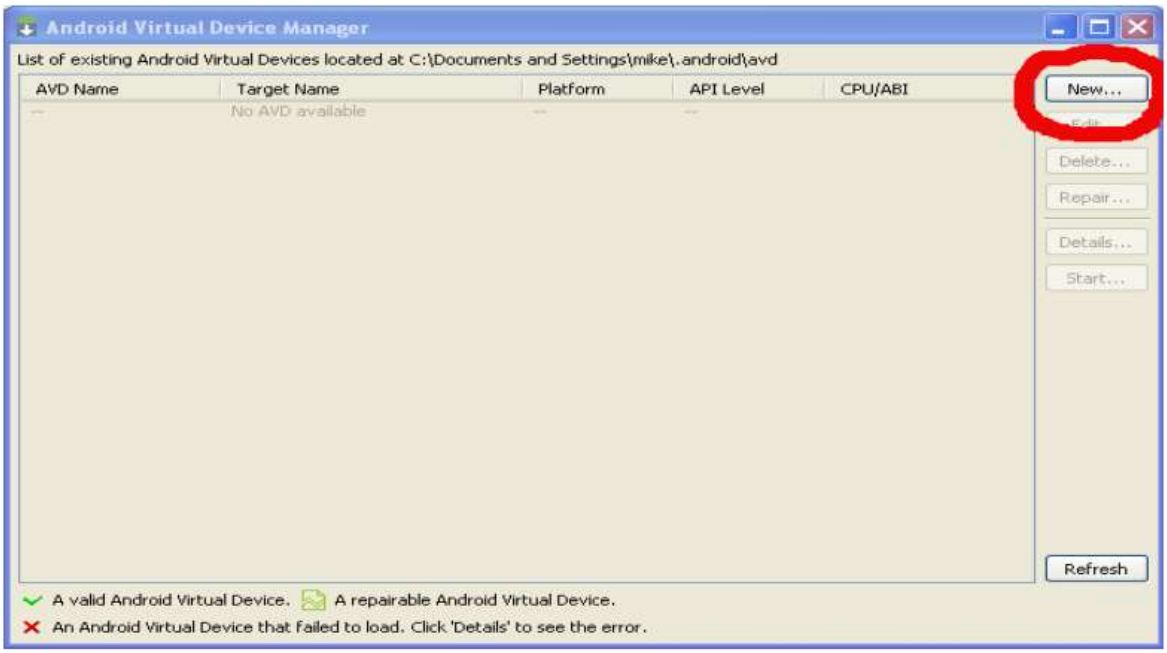

Android SDK Manager
Packages Tools
SDK Path: C:\android-sdk
Packages
Name API Rev. Status
Tools
Android SDK Manager Log
Found XT882 by Motorola Mobility, Inc., Android API 10, revision 2
Found XT928 by Motorola Mobility, Inc., Android API 10, revision 1
Found MT870 by Motorola Mobility, Inc., Android API 10, revision 2
Fetching URL: http://innovator.samsungmobile.com/android/repository/repository.xml
Validate XML: http://innovator.samsungmobile.com/android/repository/repository.xml
Parse XML: http://innovator.samsungmobile.com/android/repository/repository.xml
Found GALAXY Tab by Samsung Electronics., Android API 8, revision 1
Fetching URL: http://developer.sonyericsson.com/edk/android/repository.xml
Validate XML: http://developer.sonyericsson.com/edk/android/repository.xml
Parse XML: http://developer.sonyericsson.com/edk/android/repository.xml
Found EDK 2.0 by Sony Ericsson Mobile Communications AB, Android API 10, revision 1
Done loading packages.
Preparing to install archives
Downloading Android SDK Platform-tools, revision 10
Installing Android SDK Platform-tools, revision 10
Stopping ADB server failed (code -1).
OK
Unzipping Android SDK Platform-tools, revision 10 (17%)
Close
Google Market Licensing package 2 Not installed
Google USB Driver package 4 Not installed
Google Webdriver package 2 Not installed
Intel Hardware Accelerated Execution Manager packa 1 Not installed
Show: Updates/New Installed Obsolete Select New or Updates
Sort by: API level Repository Deselect All
Install 6 packages...
Delete 1 package...
Unzipping Android SDK Platform-tools, revision 10 (17%)



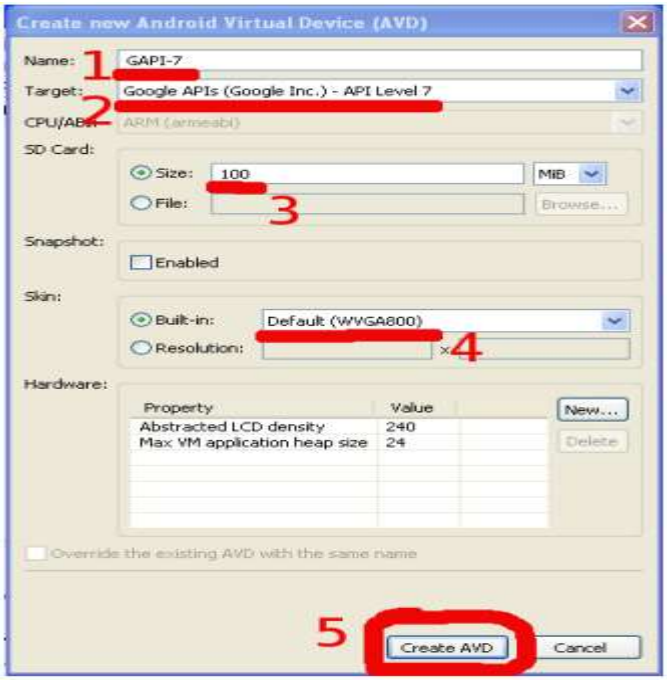
Next, we launch the AVD (Android Virtual Devices) manager and configure the emulator to use Android version 2.1 with the Google API extension (GAPI). GAPI is needed, as a rule, for applications that use the cartographic capabilities of Android.



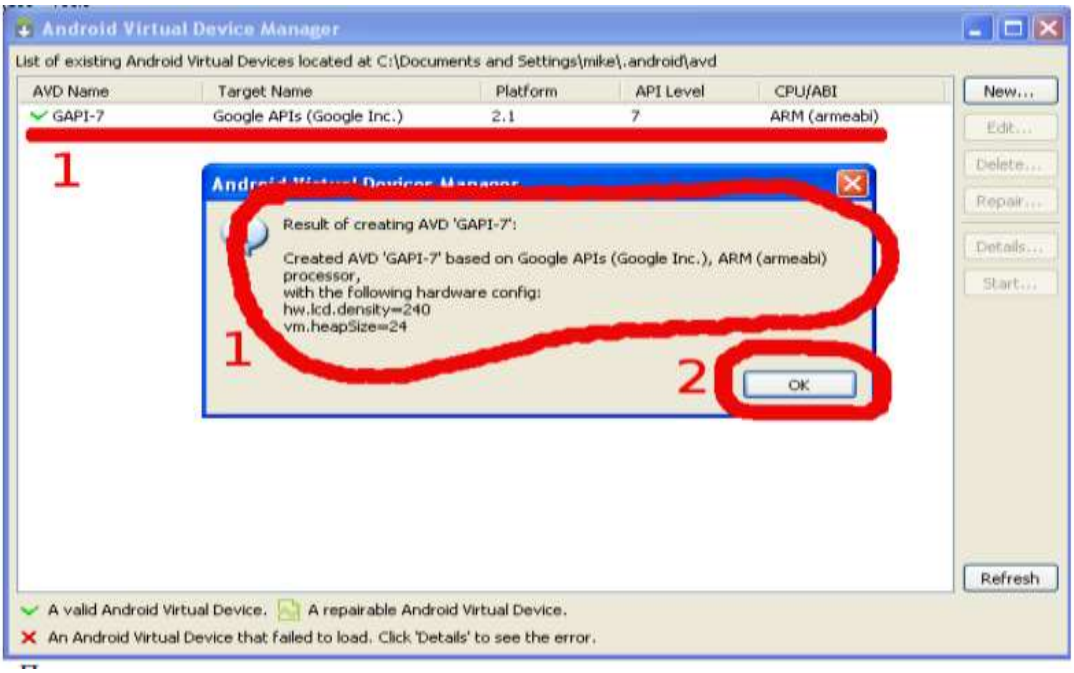
Create a new virtual device:



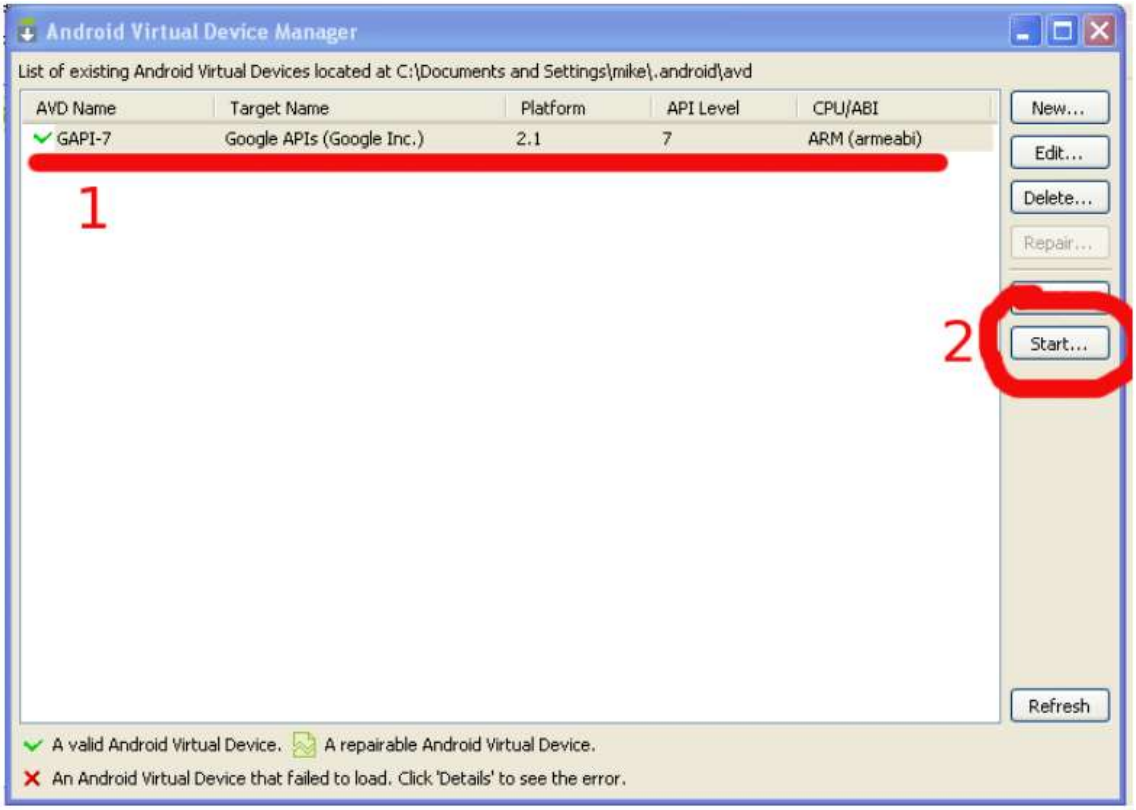
When creating a new virtual device, select its name (1), the target API (2), the size (or image file) of the virtual SD card (3), one of the standard or native resolution of the screen (4), and finally create the device 5). If necessary, you can specify the pixel density on the screen, the maximum heap size for applications within the virtual machine, and other parameters:



Make sure that everything turned out as desired (1) and continue to work (2):



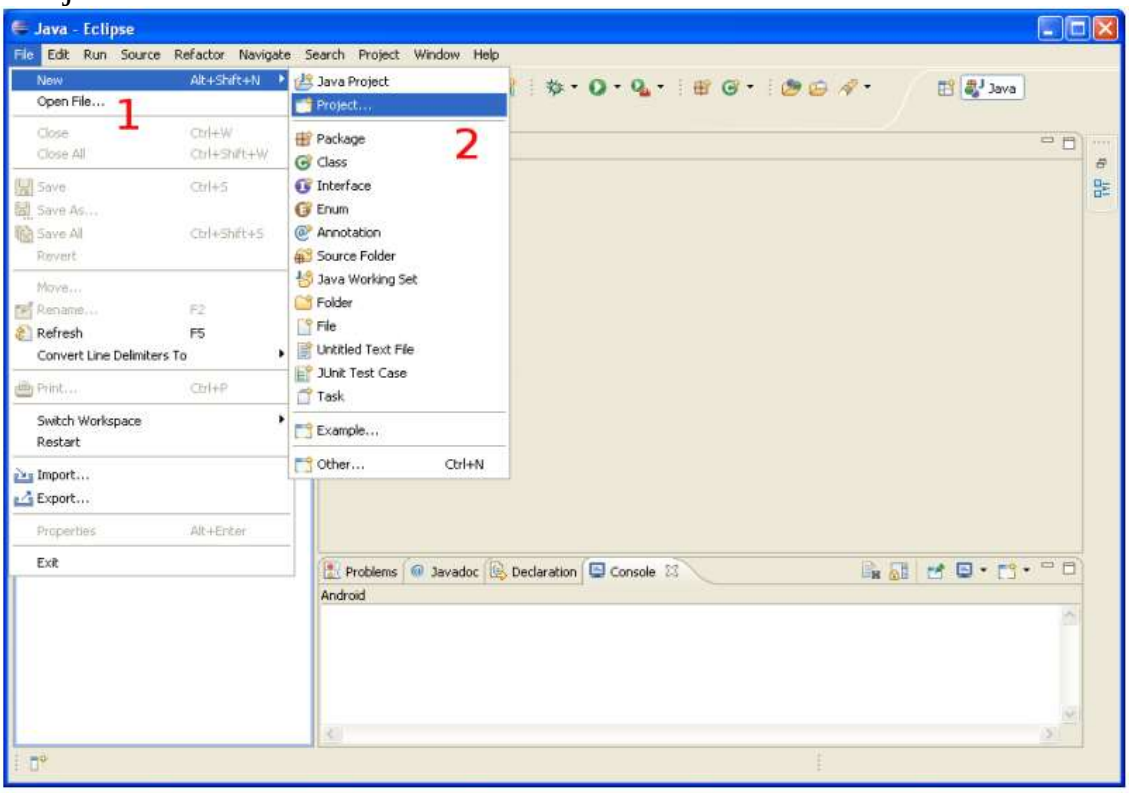
It's time to start the emulator:



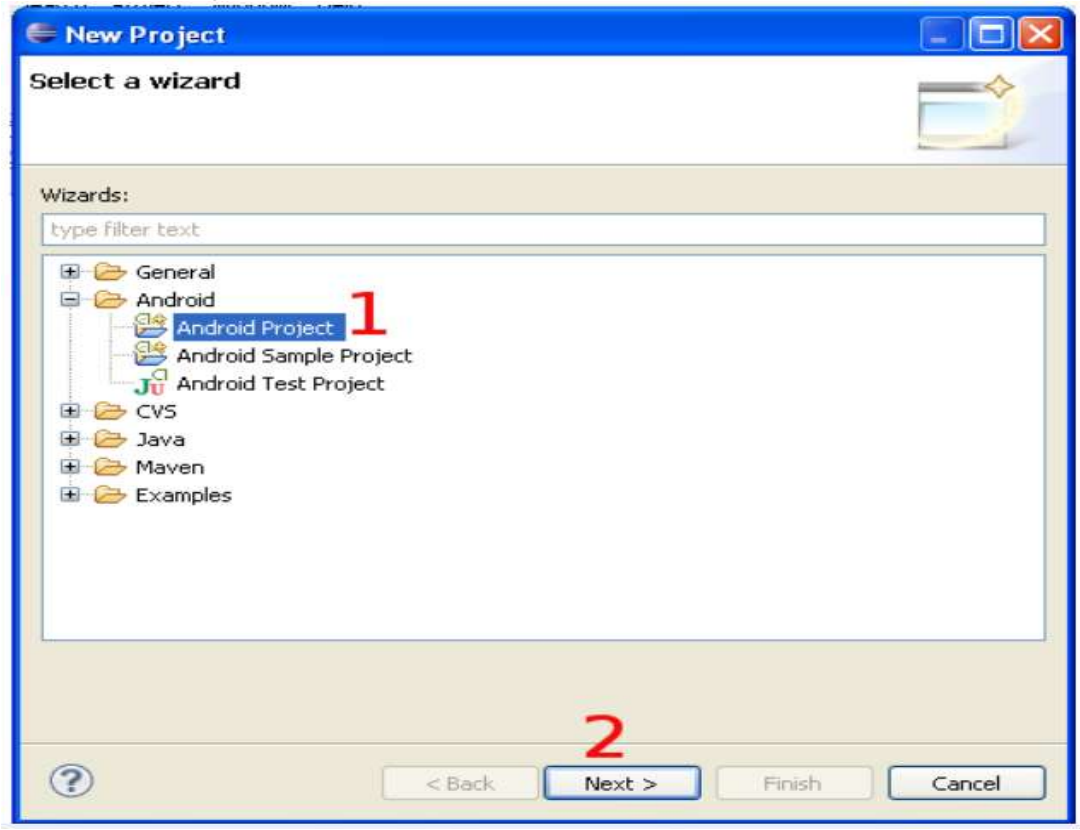
II.Chapter. Application creation under Android

2.1. Create the first Android app using Eclipse

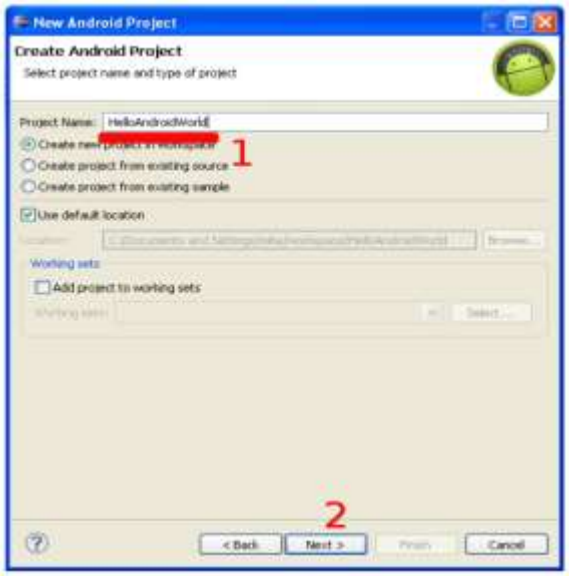
To create applications in our development environment, it is convenient to use the "New Project Wizard":



Choose "Android Project"



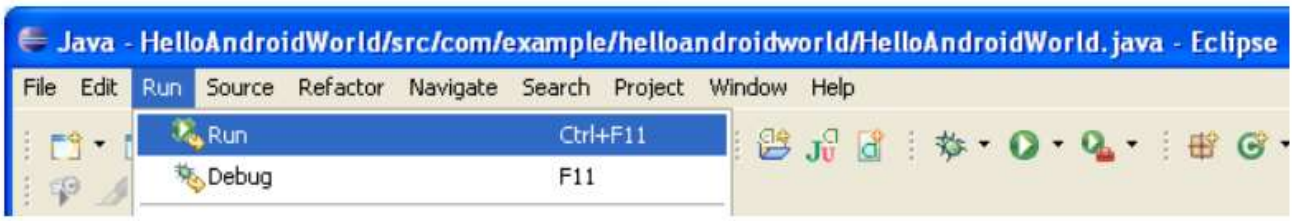
Enter the name of the project



Choose the target platform

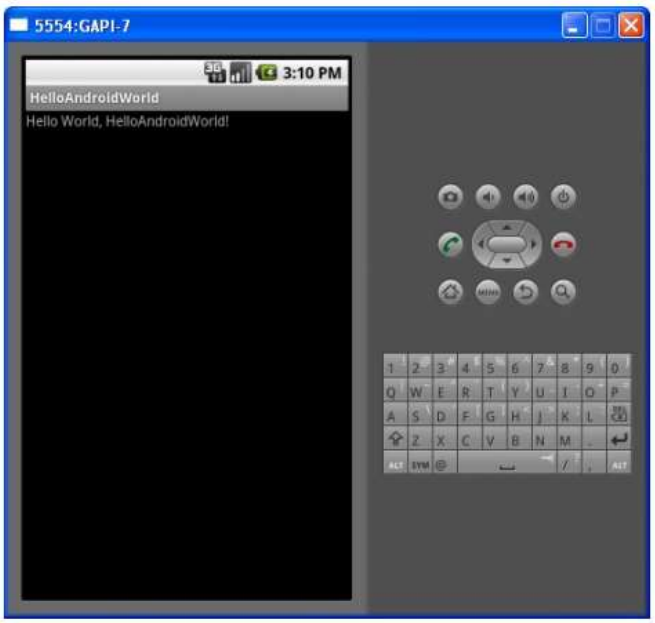
Enter the name of the package (1) and the name of the Activity class (2), which will be automatically created for us by the master:

The created project immediately while we did not make it incompatible with the life of the change is a workable application, so we can run it:



Let's understand Eclipse, exactly how we want to run our project:

And we enjoy the result:



2.2. Create the first Android app using Embarcadero RAD Studio

To create the first android application using Embarcadero RAD Studio start working further

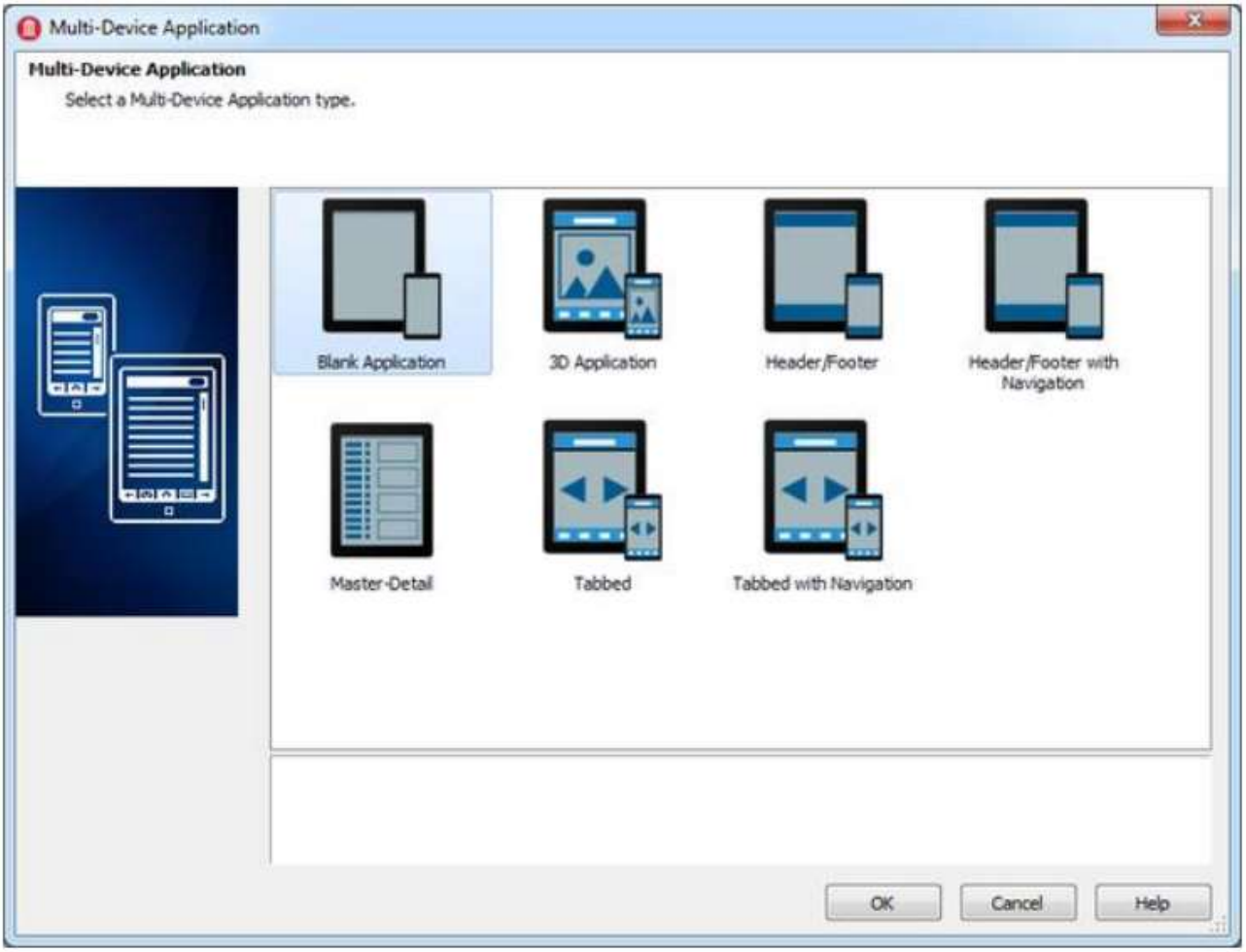
To develop mobile (iOS and Android) applications using RAD Studio, you need to complete some important configuration steps. This qualification graduate work assumes that you have completed all the necessary setup steps.

Step 1: Create a New FireMonkey Application for Android

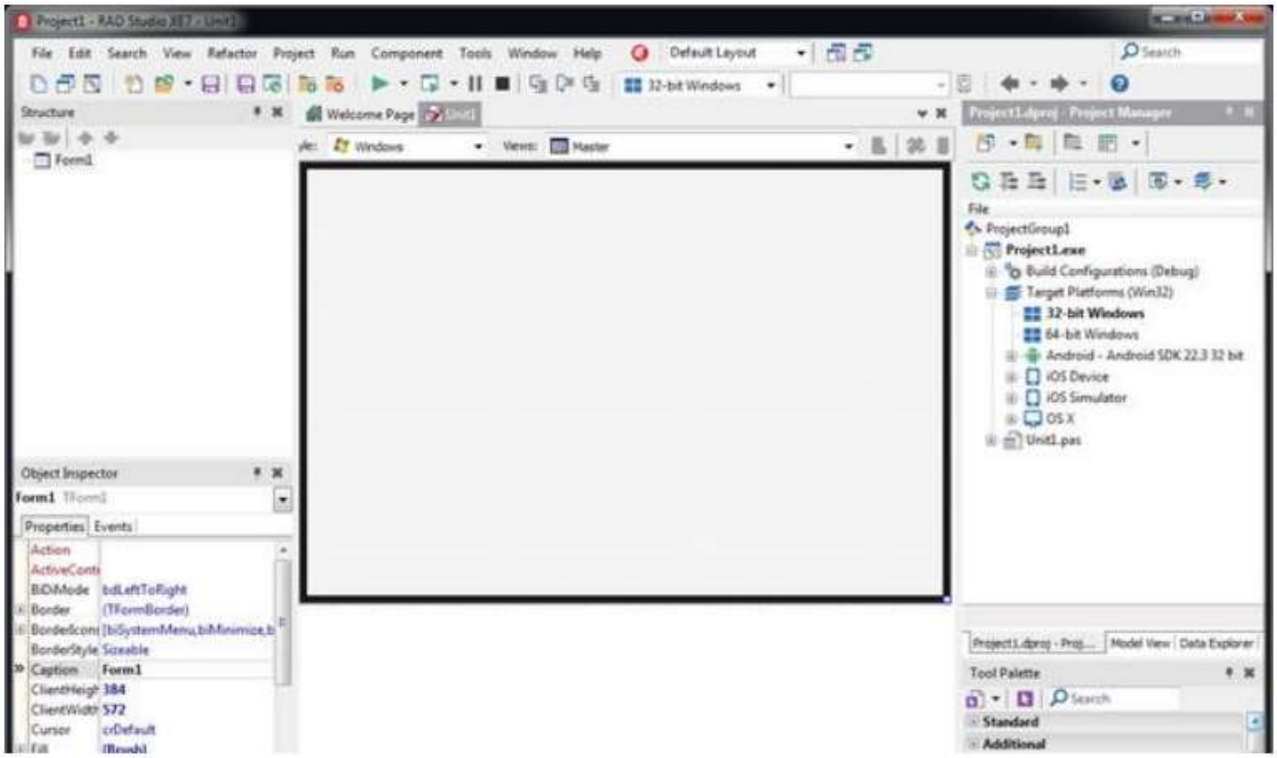
1. Select either:

- File > New > Multi-Device Application – Delphi
- File > New > Multi-Device Application - C++Builder

The Multi-Device Application wizard appears:



1. Select Blank Application. The Form Designer shows a new form:



2. Select the target platform from the Project Manager.

1. Android: See Configuring Your System to Detect Your Android Device to use an Android device.

2. iOS: If you want to create an iOS app, open the Target Platform node in the Project Manager and double-click iOS Simulator (only for Delphi) or a connected iOS device (for either Delphi or C++):

Step 2: Select a Style

1. Select either iOS or Android from the Style drop-down menu in order to define the Master view to show all the properties related with this style.

Step 3: Place Components on the Multi-Device Form

The first step in creating a multi-device application is designing the user interface. There are many reusable components available in the IDE for creating user interfaces.

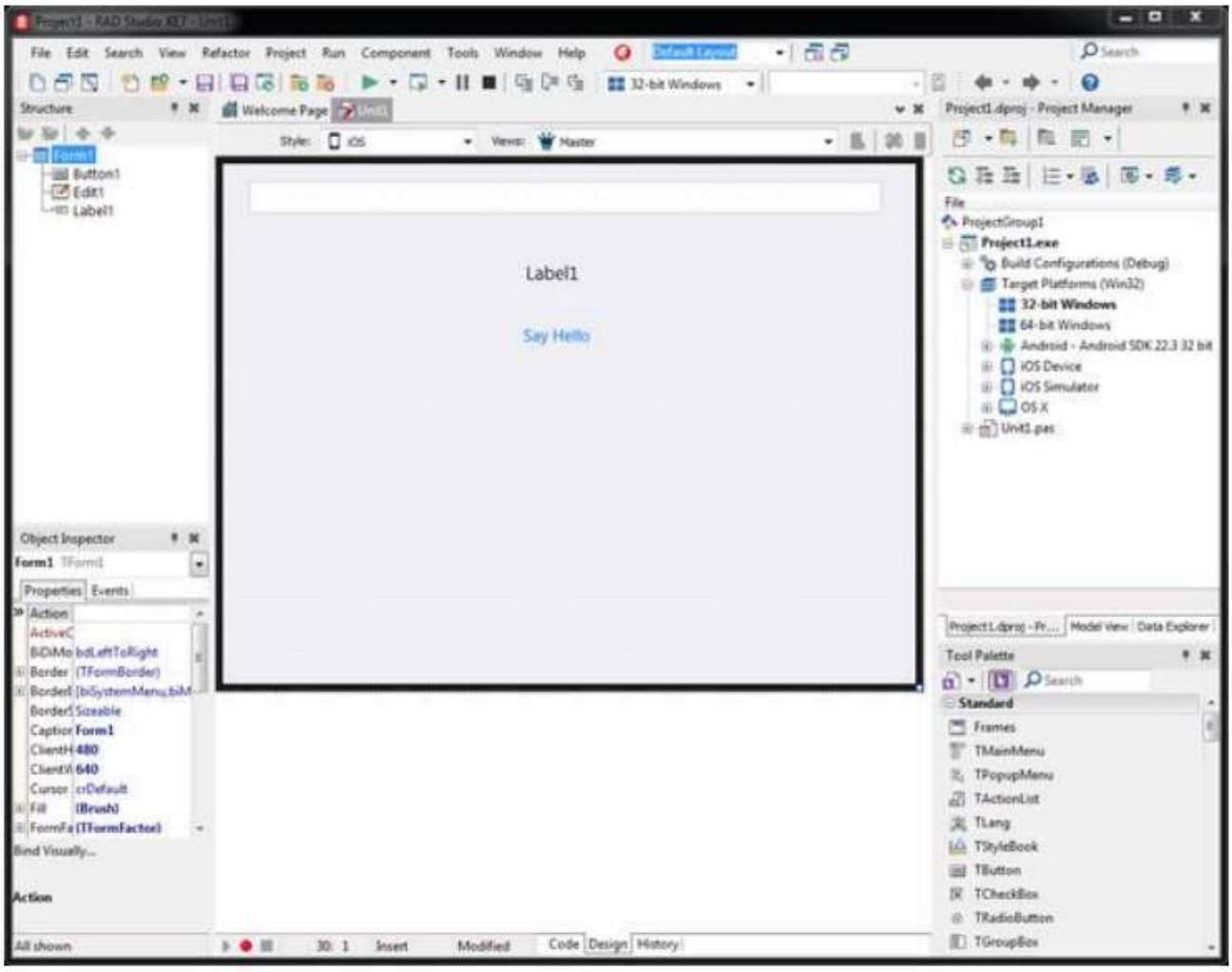
1. Move the mouse pointer over the Tool Palette, and expand the Standard category by clicking the plus (+) icon next to the category name.
2. Select the TEdit component and either double-click TEdit or drop it onto the Form Designer.

3. Repeat these steps, but now add a TLabel and a TButton component to the form.

4. Select the edit box and set the KillFocusByReturn property in the Object Inspector to True.

5. Select the button and change the Text property in the Object Inspector to "Say Hello".

6. Now you should see three components on the Form Designer. Here is an iOS app:

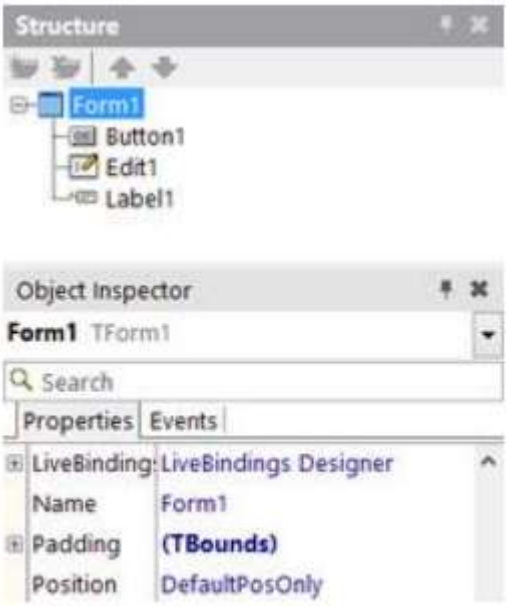


7. After you place these components on the Form Designer, the IDE automatically sets names for the components.

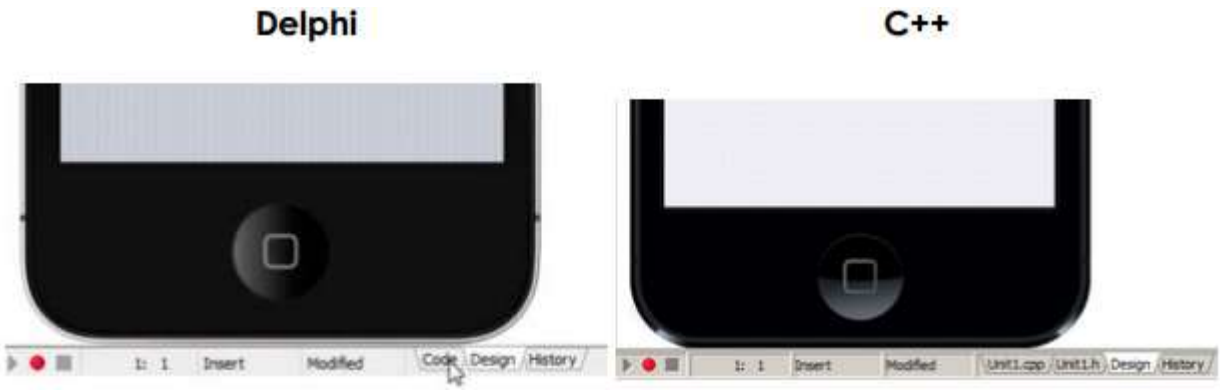
To see or to change the name of a component, click the component on the Form Designer, and then find its Name property in the Object Inspector and the Structure View:

For a TButton component, the component name is set by default to Button1 (or Button2, Button3, depending on how many TButtons you have created in this application).

8. The form on which these components are located also has a name. Select the background of the Form Designer, and select the Name property in the Object Inspector. The name of the form Form1 (or Form2, Form3,...) is displayed. You can also locate the name of the form in the Structure View:

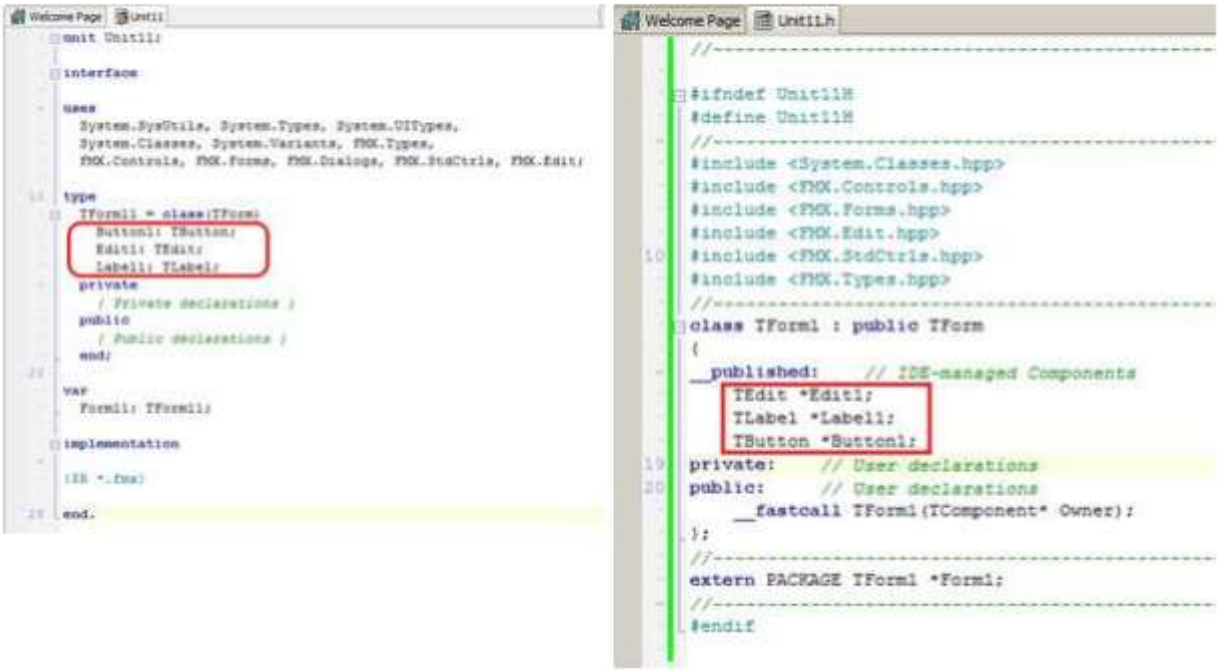


9. You can easily switch to source code by selecting the Code (for Delphi) or <unit name>.cpp/<unit name>.h (for C++) tab at the bottom of the Form Designer. You can also press the F12 key to switch between the Form Designer and the Code Editor:



The Code Editor displays the source code that the IDE has generated. You should find three components defined (Edit1, Label1, and Button1):

Delphi C++



Step 4: Adding Views to Your Project

If you want to customize you application for a particular type of device, you can do it using Views.

1. Go to the Views selector.
2. Select the available views you want to add just by clicking on them.
3. Go to the view to do the changes you want to include.

To add a customized view, see Adding a Customized View to the View Selector.

Step 5: Write an Event Handler for a Button Click by the User

The next step is defining an event handler for the TButton component. You can define event handlers for your application in the same way you define event handlers for desktop platforms. For the TButton component, the most typical event is a button click.

Double-click the button on the Form Designer, and RAD Studio creates skeleton code that you can use to implement an event handler for the button click event:

Now you can implement responses within the Button1Click method. The following code snippets (Delphi and C++) implement a response that displays a small dialog box, which reads "Hello + <name entered into the edit box>":

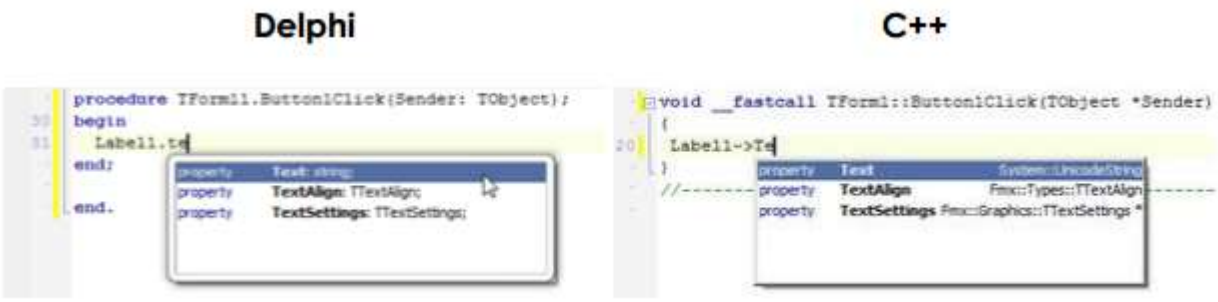
Delphi code:

```
Label1.Text := 'Hello ' + Edit1.Text + ' !';
```

C++ code:

```
Label1->Text = "Hello " + Edit1->Text + " !";
```

In Delphi, the quotation marks that surround string literals must be straight single quotation marks (that is, 'string'). You can use the plus (+) sign to concatenate strings. If you need a single quote inside a string, you can use two consecutive single quotes inside a string, which yields a single quote. While you are typing code, some tooltip hints appear, indicating the kind of parameter you need to specify. The tooltip hints also display the kinds of members that are supported in a given class:



Step 6: Test Our Mobile Application

The implementation of this application is finished, so now you can run the application.

We can click the Run button ( ) in the IDE, press F9, or select Run > Run from the RAD Studio main menu:

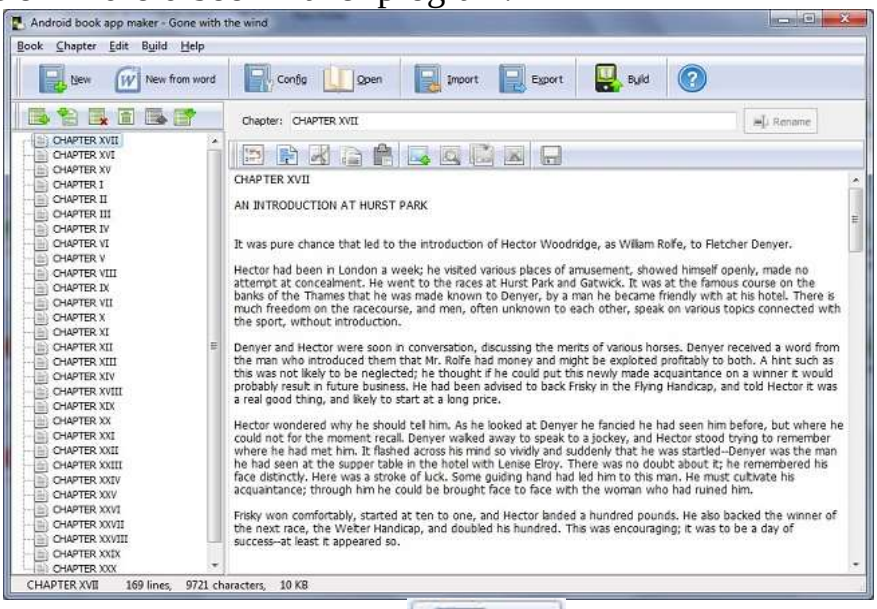
Test Our Android Application on the Android Device

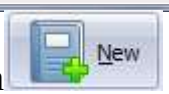
If we complete the steps described in qualification graduate work: Set Up Our Development Environment on Windows PC (Android) before creating your new project, we can now run your Android app on an Android device connected to our PC by USB cable.

III. Chapter. Create the first Android application for smartphones

3.1. The workflow to develop an app for Android

Main Interface of Android book maker program.



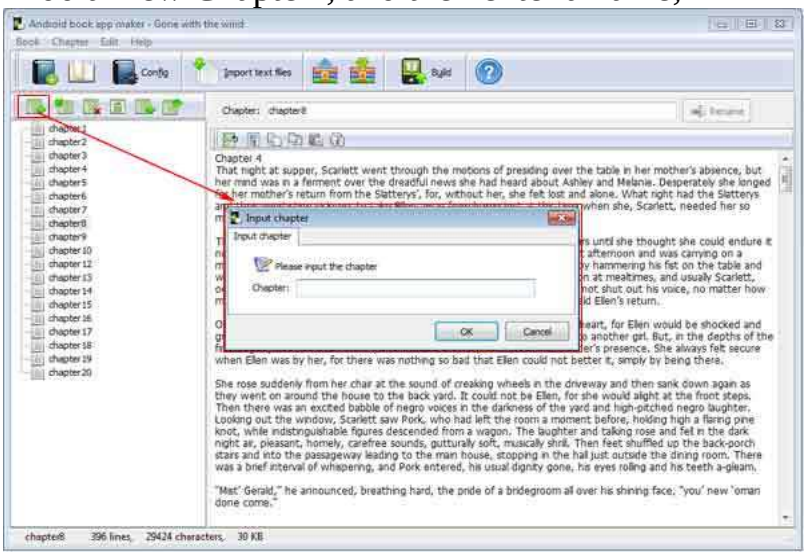
Create a new book via clicking button 

and set the basic information of current book.

Click  icon to import text files.
Optional, edit chapters and contents,

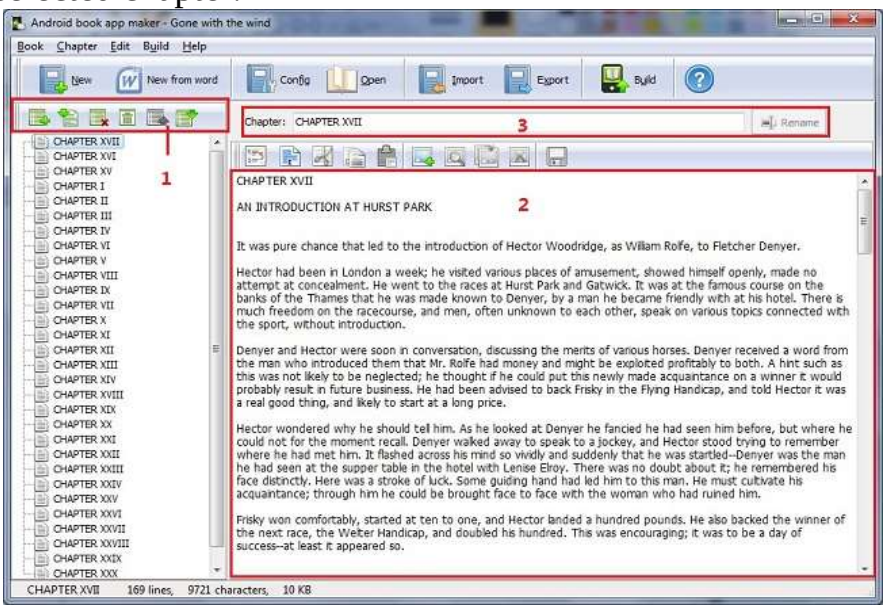
Click 

icon to output eBook in the form of APK.
After creating the APK file, you may:
Test it following the topic.
Publish it following the topic.
Create chapters :
I. Create new chapters manually:
1 Click the icon "Add a New Chapter", and then enter a name;

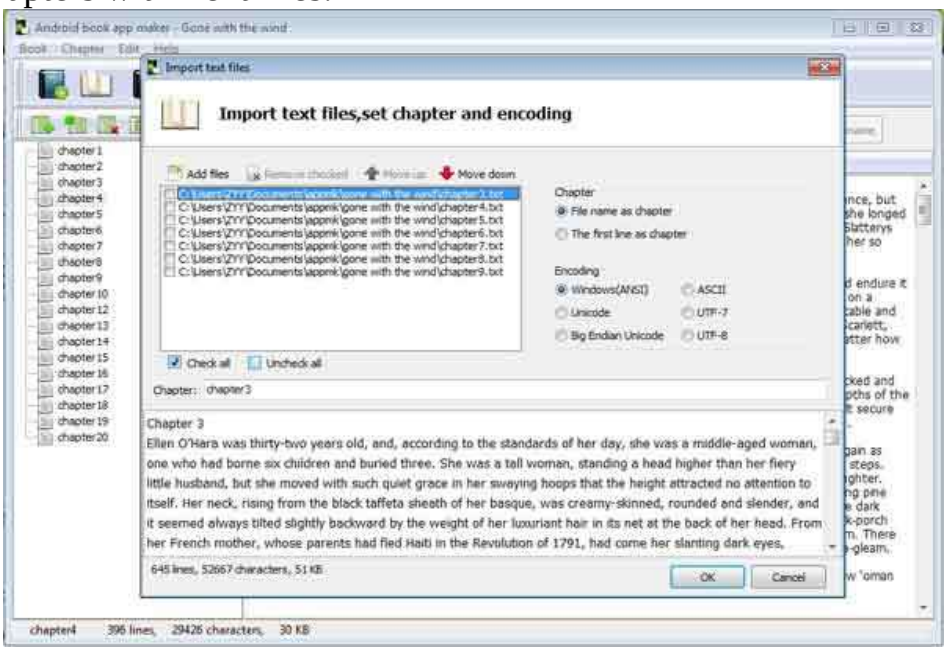


2) Edit chapter:

<1> Append, insert, remove, clear all, move up, move down.
<2> Remove the non-paragraph line breaks, select all, cut, copy, paste and save current content.
<3> Rename selected chapter.



II. Create chapters with Text files:

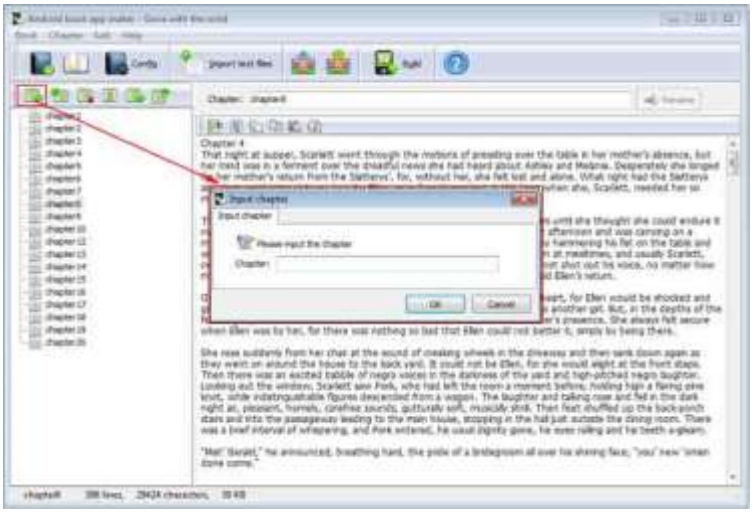


1) Add text files from local PC via clicking button "Add files".
2) Remove the checked text files from the list, and sort them.
3) Name chapter: "File name as chapter" and the "The first line as chapter".
4) Choose encoding: Windows(ANSI), ASCII, Unicode, UTF-7, UTF-8, Big Endian Unicode.

Create chapters:

I. Create new chapters manually:

1) Click the icon "Add a New Chapter", and then enter a name;

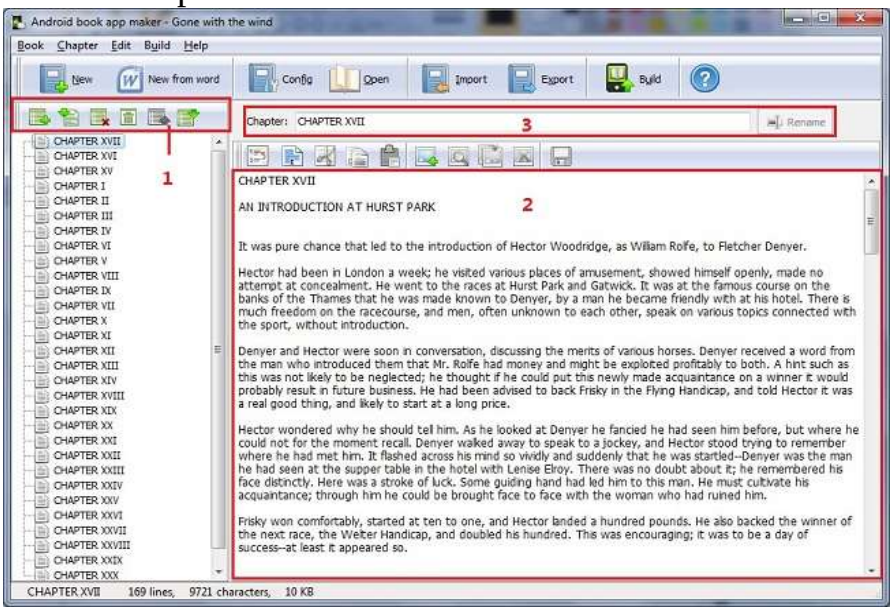


2) Edit chapter:

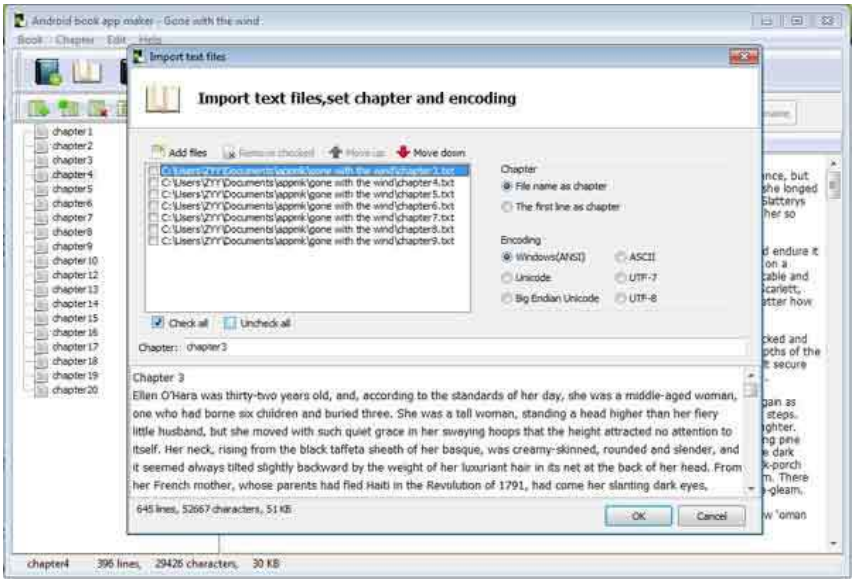
<1> Append, insert, remove, clear all, move up, move down.

<2> Remove the non-paragraph line breaks, select all, cut, copy, paste and save current content.

<3> Rename selected chapter.



II. Create chapters with Text files:

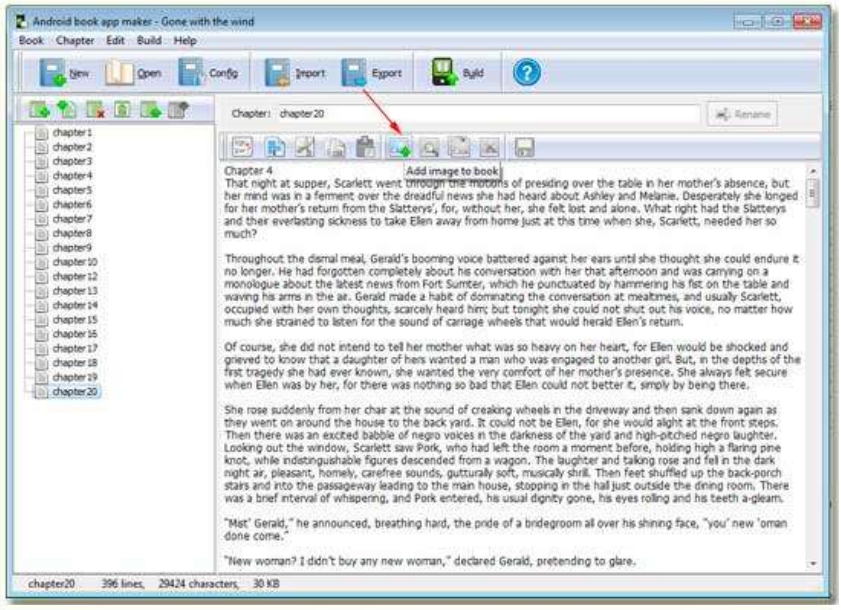


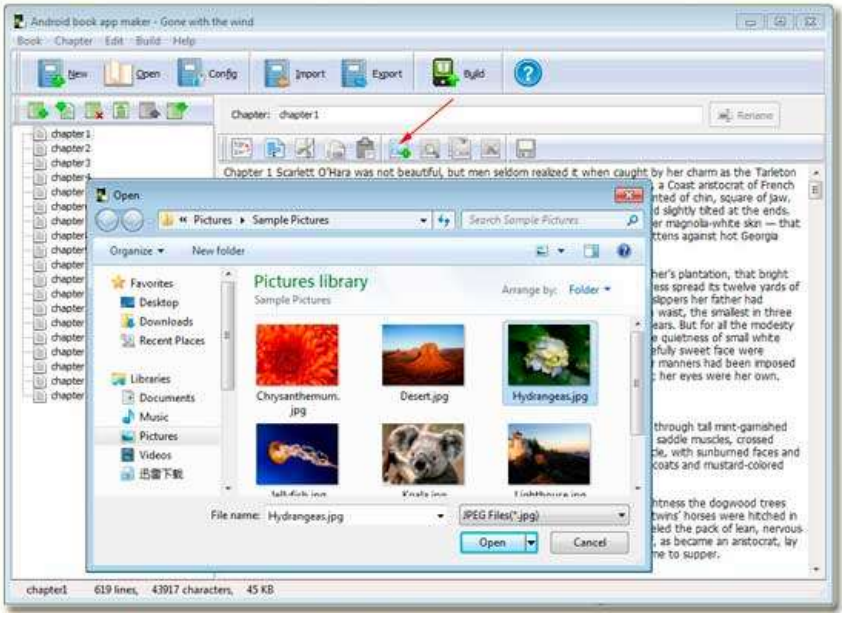
1) Add text files from local PC via clicking button "Add files".
2) Remove the checked text files from the list, and sort them.
3) Name chapter: "File name as chapter" and the "The first line as chapter".
4) Choose encoding: Windows(ANSI), ASCII, Unicode, UTF-7, UTF-8, Big Endian Unicode.

Insert Images:

I. You would love to add some content-relevant illustration pictures. Scroll mode doesn't support illustration inserting.

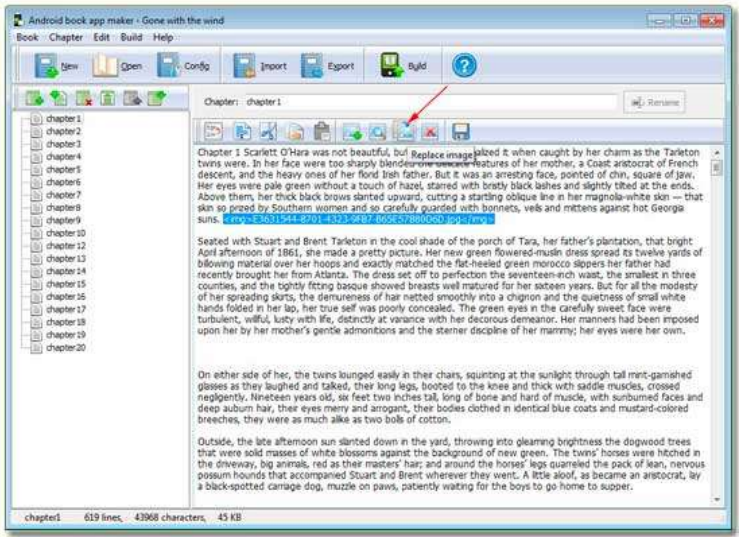
1) Add illustrations in content. Click the icon of Add image to book and then select an image from the pop-up image folder**.**



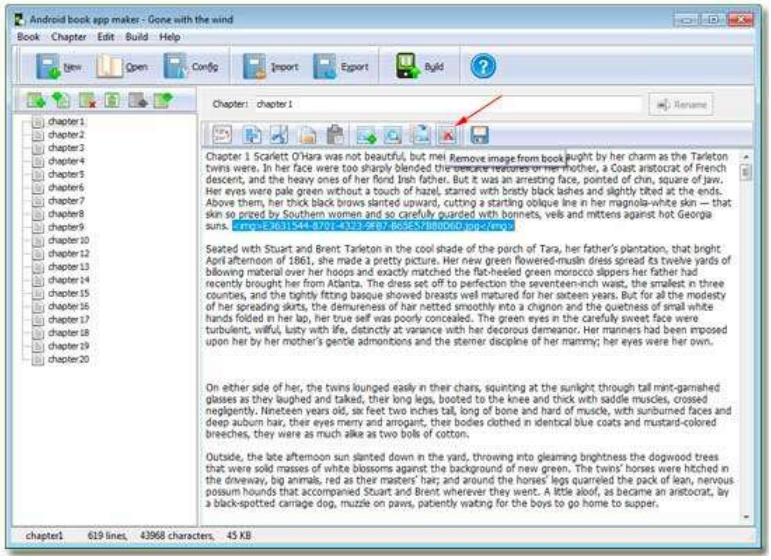


2) Preview the image just now inserted. Click the icon of View image.

3) Replace the selected image with other picture. Click the icon of Replace image.

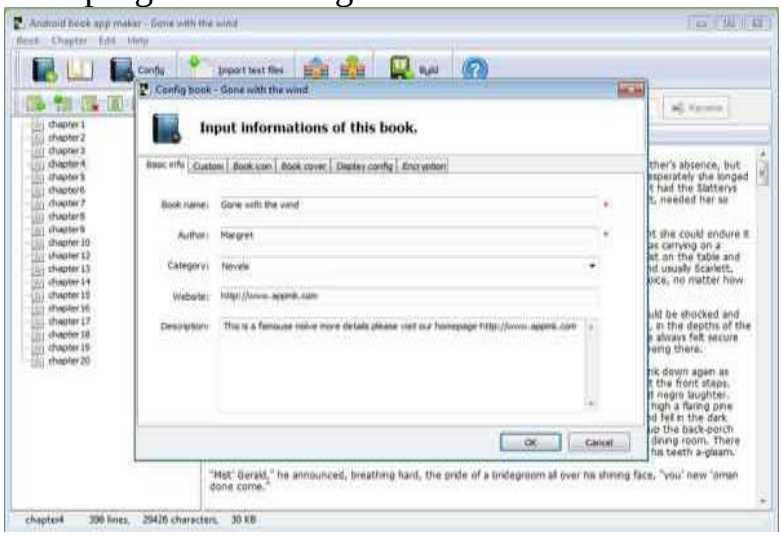


4) Remove the selected image.



II. Please view the output effect on Android screen.

Andro0id Book maker programm Config



Modify the basic information, book icon, book cover and display config for the current editing book .

Export and import project

If you want to edit the book on other computers, could package the editing book and export. When it's needed, could beimported to continue editing.

Click icon 

to export current editing book as package file for transmitting among friends and future edit.

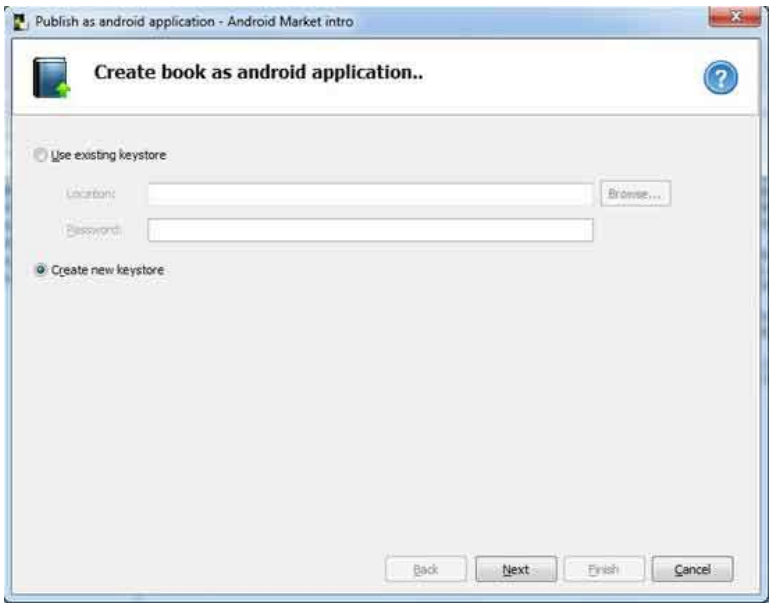
Click icon 

to import package file exported last time to continue editing.

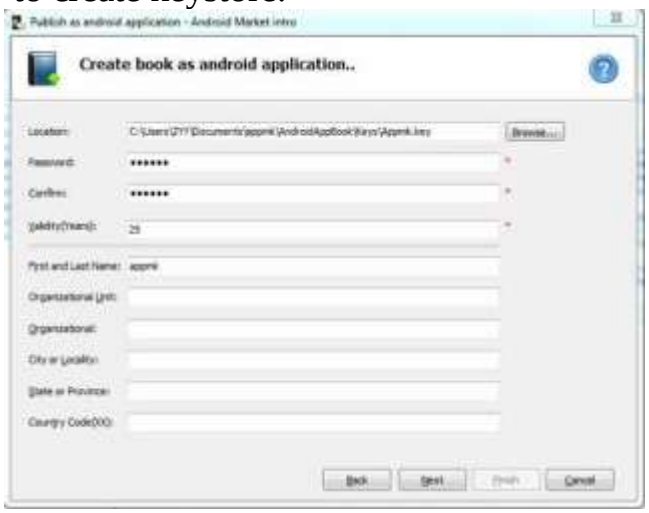
Build eBook APP:

Before building an eBook APP, you should first create a private key saved in the keystore file (Android book app maker provides keystore to save the digital certificate to self sign when the app run in Android).

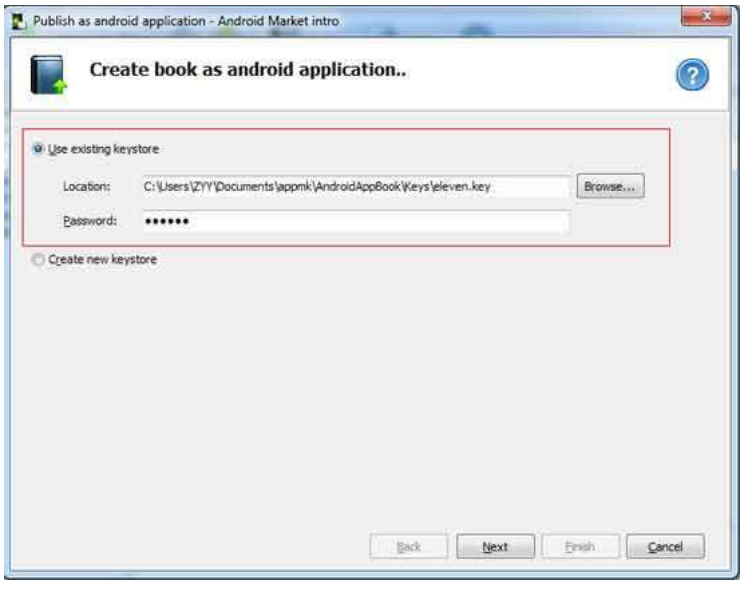
1) Check the checkbox "Create new keystore", and click button "Next" (If you have created one, should skip to step 3).



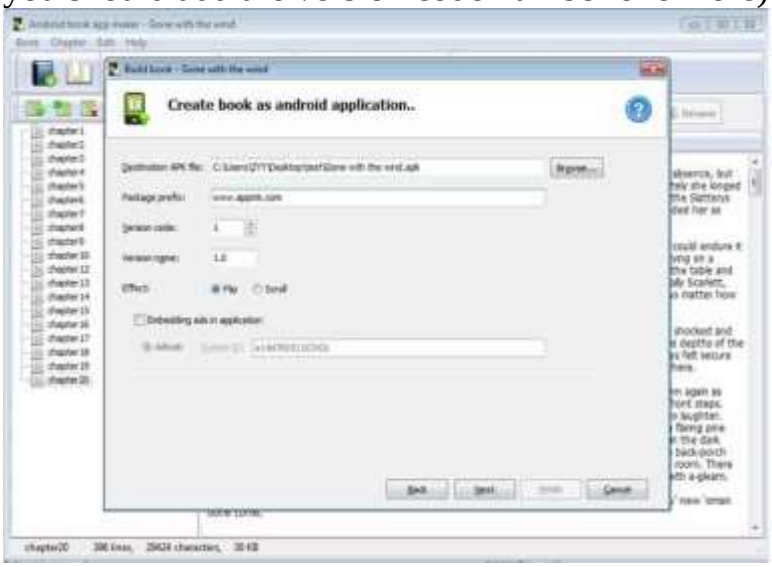
2) Browse a saving path for the key and name it (should not the existed). The forms whose tail noted by red asterisk is required and the others at least you have to enter one. Need to emphasize that the required form "Validity (Years)" is at least 25 years. And then click "Next" to create keystore.



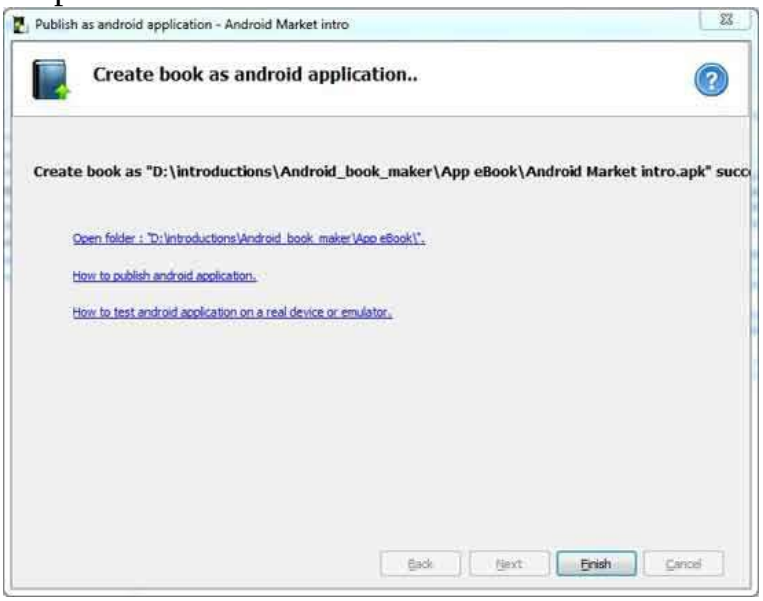
3) If you have created a private key, browse the key from its saving directory and then enter its password. Click "Next".



4) Give your package file (need to continue to edit) a frefix provides convenience for remembering. Select book browsing effect: Scroll and Flip. If want to embed Admob ads, check "Embedding ads in application" and then enter the "Publish ID". As you see , you can enter the app version code and name (Once upload the same eBook app into the same Market, you should add the version code number one more).

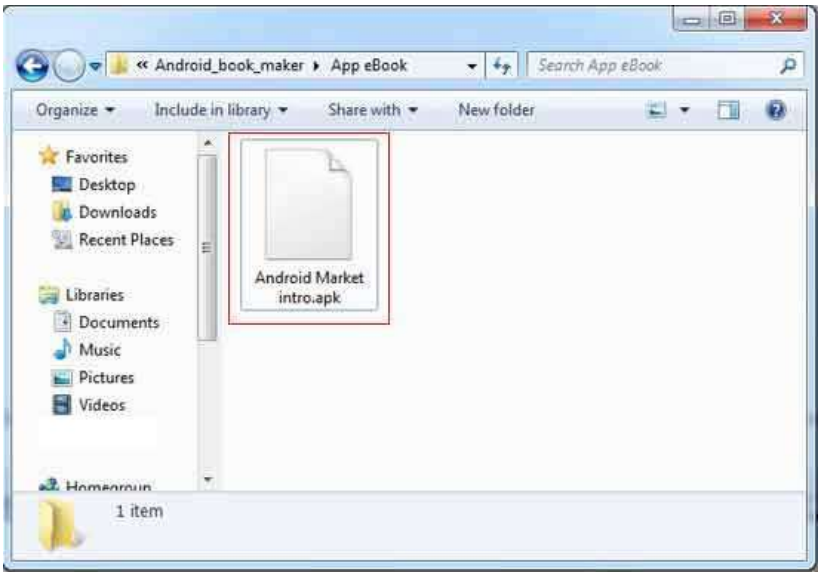


5) Click "Next" to skip to the below "Finish" window.



Testing eBook APP:

We have built an APK file as below illustration with Android book app maker. But now we couldn't use it untill install it on Android-powered Phone or Android Emulator. So the below content will introduce how to test it on both devices.



How to transmit the APK file to SD card?

You may use one of the below methods:

<1> Data line or Bluetooth: copy the APK file into SD card and install it by connecting your computer and Android phone (so does Andriod Emulator, only does it use some software to connect) with them.

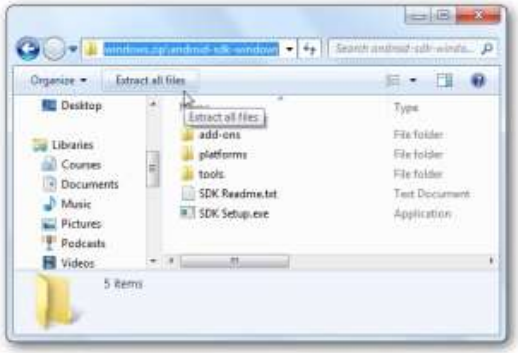
<2> Web server: create a web server on PC and put the APK file in one of its directorys, next visit the web and open the downloading webpage to download the APK into SD card via WiFi or other Network on the android-powered phone (so does Android Emulator).

Note: The emulator is as same as the visiable Android-powered Phone. Merely emulator provides much convenience for APP developpers to test in local computer. Next, I will tech you how to install an Android Emulator in your computer.

How to install Android Emulator?

1) The Android Emulator requires JDK (Java Development Kit), download them from the link: http://www.oracle.com/technetwork/java/javase/downloads/index.html.

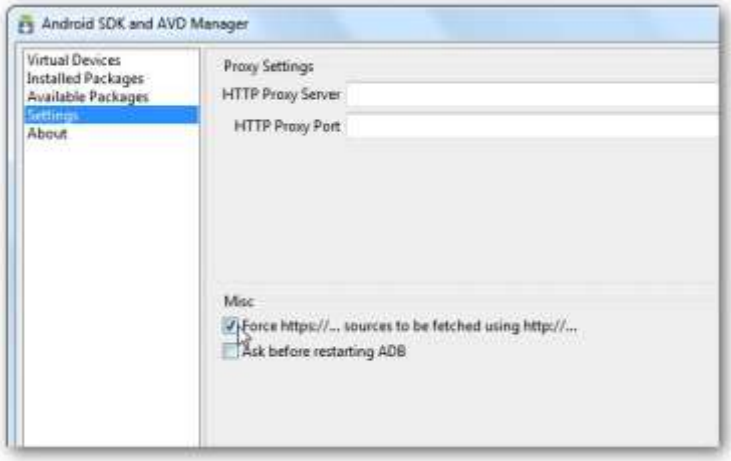
2) Then, download the Android SDK from this link: http://developer.android.com/sdk/index.html, and make sure to select the correct version for computer. Once it's downloaded, unzip the files as normal.



3) In below Window, run the SDK Setup.exe to get started running Android on PC.

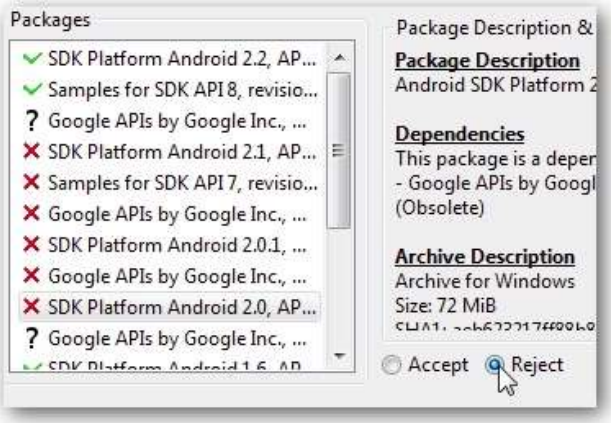
4) Wait a while, the SDK are checking Google's servers for available packages.

5) In below window click the tab "Settings", if face to an SSL error message. We uncheck the checkbox: "Force https box...", click "Ok", and then reopen the setup.

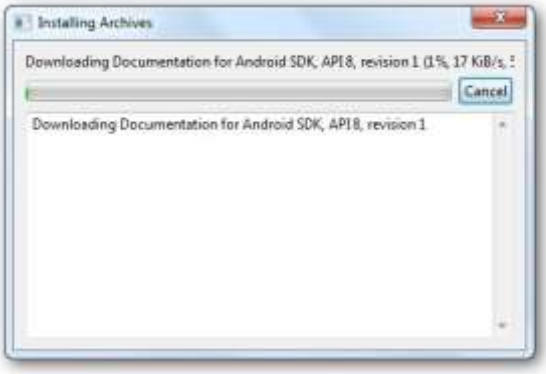


6) The Package Installer pop up. You should choose what you need to install, and then click button "Install" to set up the Android emulator on computer.

7) Acquiescently, all of the SDK platforms, samples, and APIs will be selected for installing. It would cost quite a while to download, as several versions of Android are currently available. Then click the "Reject" checkbox, and then click button "Install". The below illustration show the version 2.2. You could download the latest Android 2.3 as well as the older 1.6.

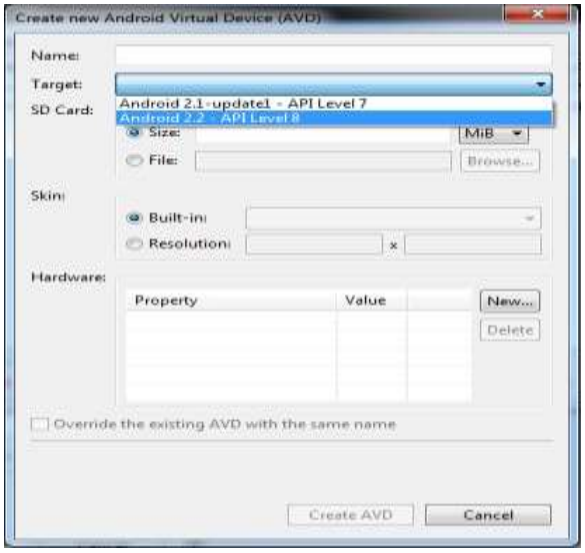


8) Once you've started installing, you'll wait a while for the download and installation progress.

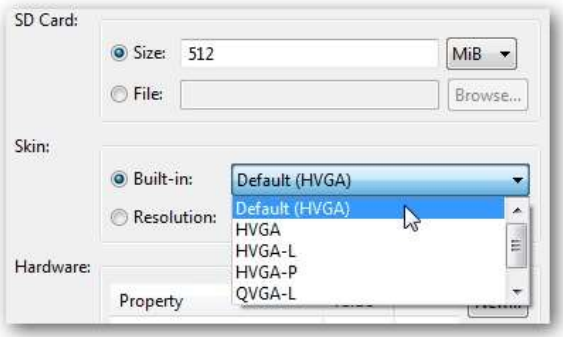


9) Setup an emulator to test the Android on computer. Select the "Virtual Devices", and then click the button "New" on the right edge.

10) Enter a name for the virtualized Android, and select the version of Android from the drop-down menu.



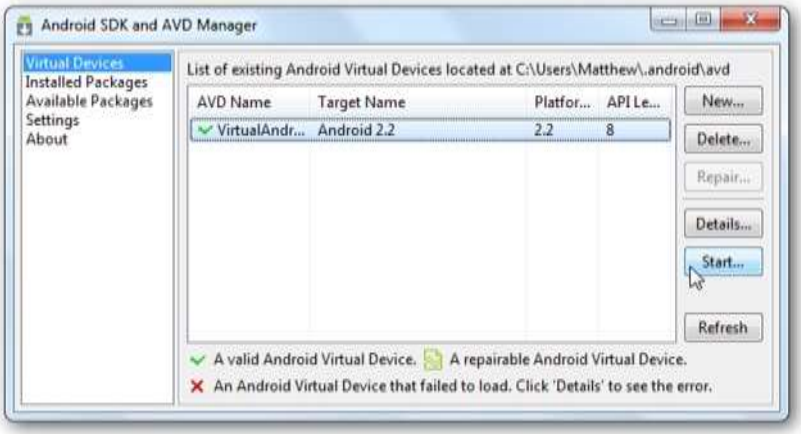
11) Enter number for the size of SD Card which is a virtual SD card that's actually an IMG file that Android will use to store users' settings and files. Then, select a screen size from the "skin" select options. The default is a standard, Nexus One-type display, while the others are different sizes for diverse device.



12) Next, click button "Create AVD".

13) Wait a moment for the freezing of the program while it is creating the AVD, so just wait until the confirmation window comes up.

14) Now ready to run Android, select a new virtual Android, and click Start on the right edge.



15) Scale the display window, and then click button "Launch".

16) On emulator Android begins to load.

Android may take a moment to load, especially on the first run. After a while, the boot screen will switch to an Android boot animation.
Android home screen comes up. Click any APP to launch but double-click**.**

We can open pre-installed apps from the launcher menu, though unfortunately these emulator images do not include the Android Marketplace.

Now we have installed the emulator, nevertheless how to copy the APK generated by Android book app maker into it?

I will tell you an easy method about how to install the output APK into the Android emulator.

ADB: Android Debug Bridge (adb) is a versatile tool lets you manage the state of an emulator instance or Android-powered device.

We can find the adb tool in <sdk>/platform-tools/.

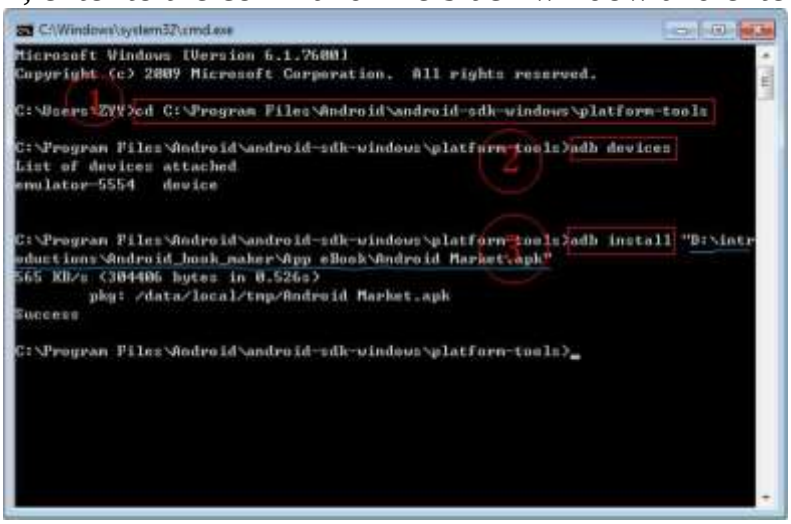
Precondition:

<1> Firstly, you must confirm that you have launched the Amulator.

<2> secondly, find out the directory of the adb.exe.

<3> thirdly you should know where the output APK files.

Now click "Start" on the down-left cornor of your screen, and enter letters "cmd" in the searching form, enter to the command line black window and enter command line**.**



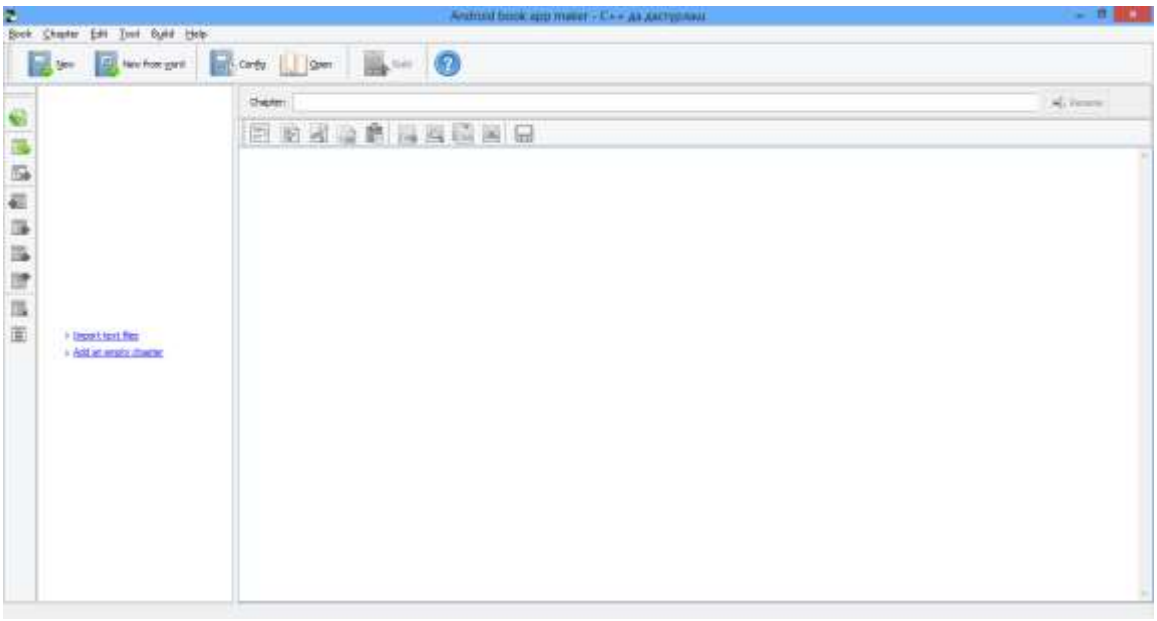
<1>Enter command line: cd C:\Program Files\Android\android-sdk-windows\platform-tools (adb.exe root directory, we should enter our correct directory).

<2> Enter command linse: adb devices

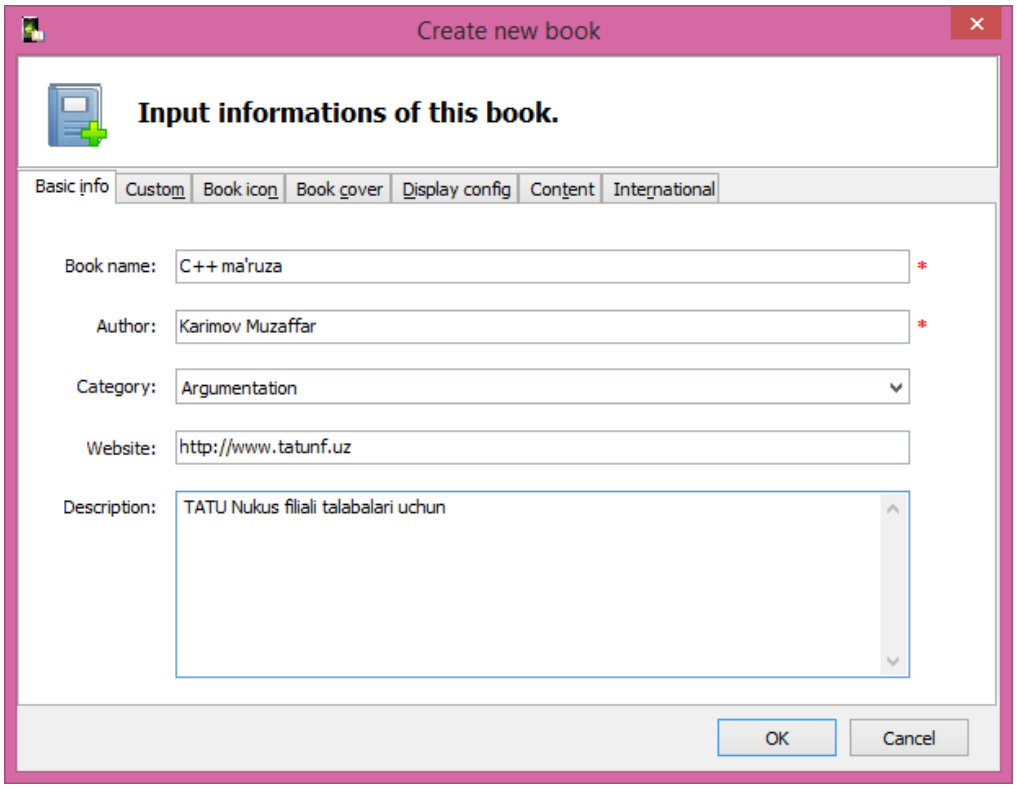
<3> Enter command lines: adb "the directory where our APK files saved".

3.2. Create *.apk files with Android book maker

At first we process Android Book App Maker. Interface of the program will be in the form like this:



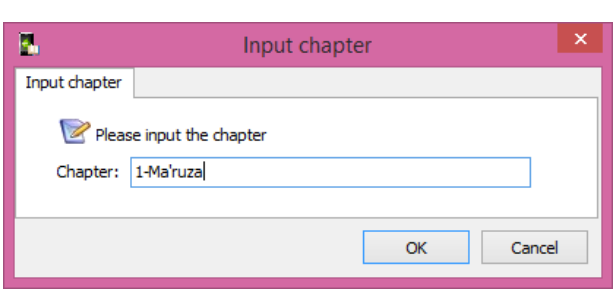
Like this we start to creat new program. As a sample we use lectures of the subject C++ for the first year students of Nukus branch of TUIT and fulfil its creation step by step. We will choose *Create new book* part and created interface will be filled in like this:

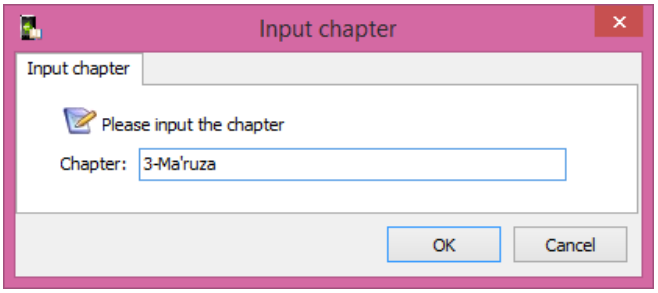


Having filled in we press the botton “OK”.

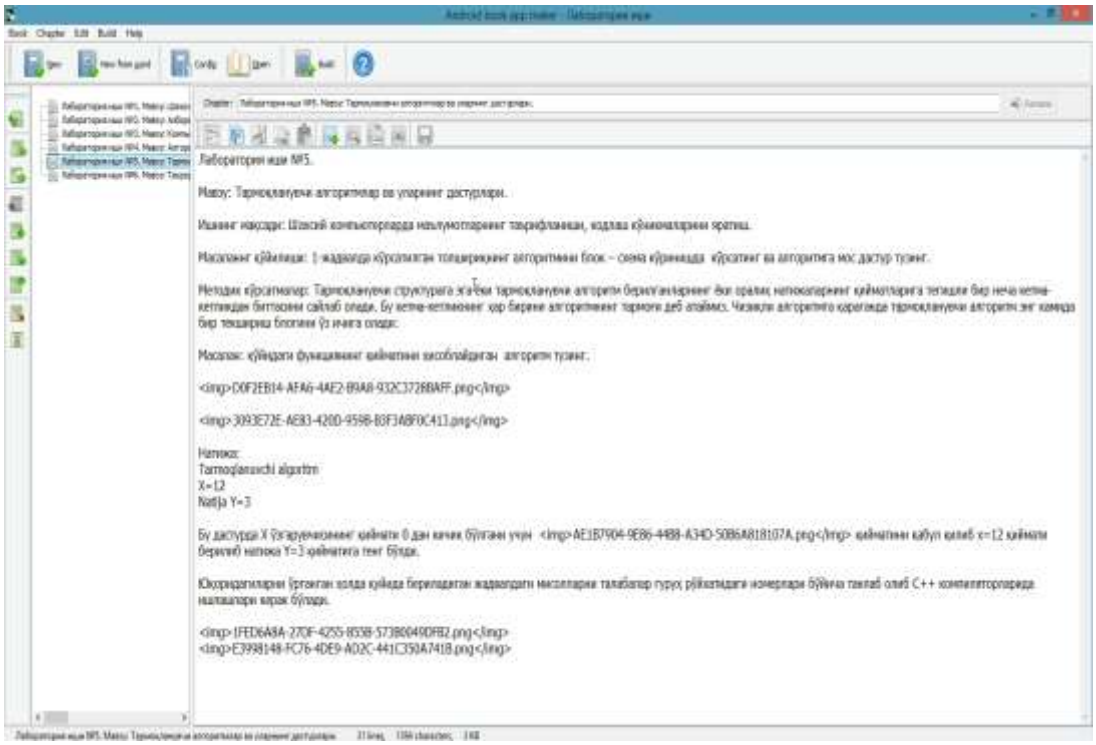
Now the data will be fed into the creating program. Choosing "*Add an empty chapter*" from the part “Chapter” we fill in the opened interface like this:

Repeating this process several times we fill in Chapter spaces.



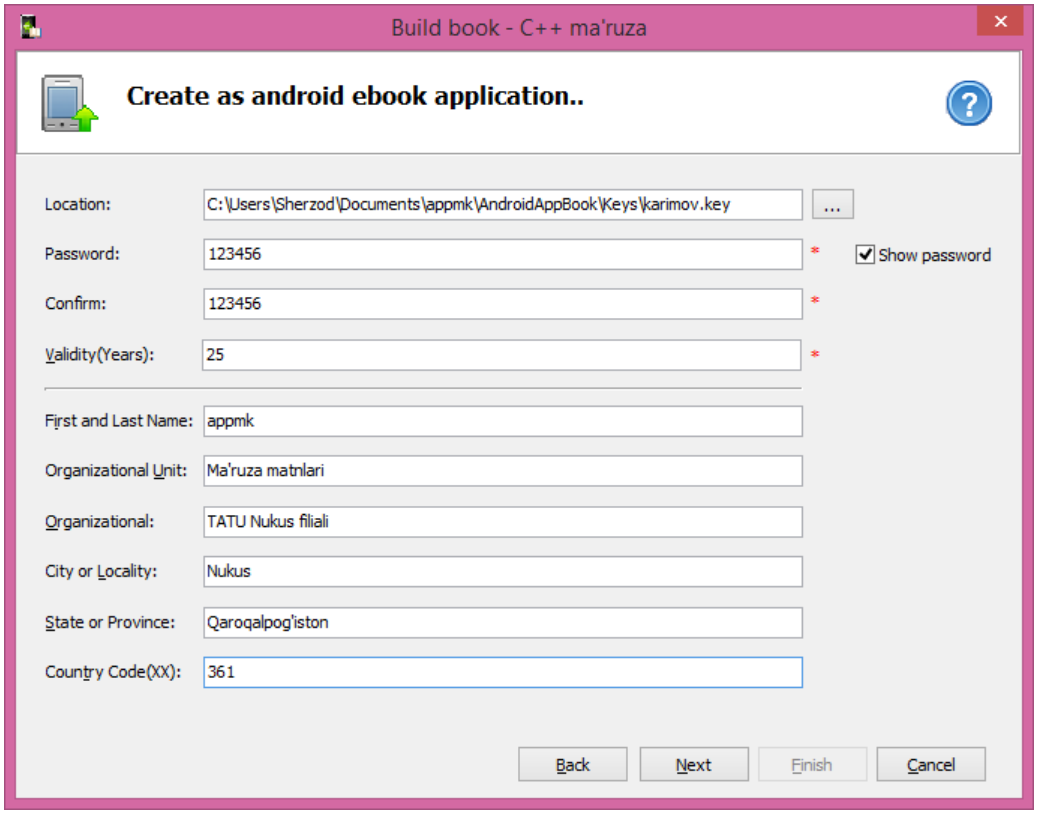


Then we feed the data into each created part.

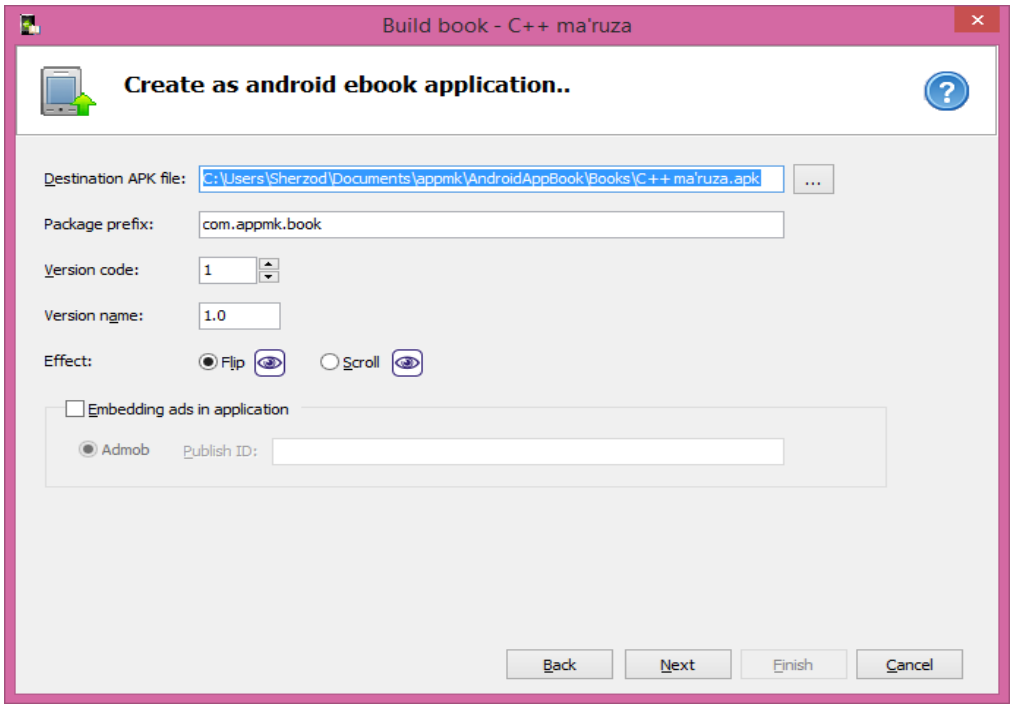


Having filled in the data, we convert this program into *.apk.* To do it, we choose "*Build*" in the menu "*Build*" and then interface will be appeared.

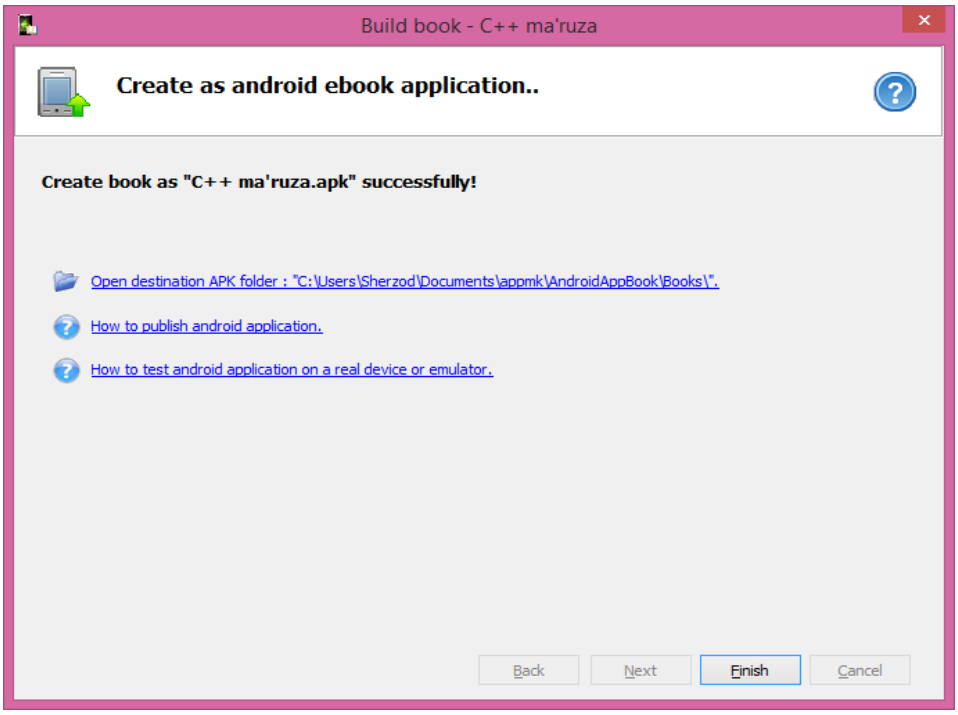
We press "Next". Then we will fill in the created interface like this:



We click "Next", and then we fill in the open interface:



After that, this interface will be appeared and press "Finish" botton:



Now, the created manual is ready to use.

The Conclusion

At this diplomas research work it was studied applications for running Android system programs. The Android system is nowadays, both the most developed and widespread system. The Android system running phones is the most popular phones among the Information Communication Technologies working phones. At the first chapter of the work we have given the information about the Android system and it is installation. And we studied to install the Java language working jdk and jre files. At the second chapter we learned now to make applications by using Eclipse and Embarcadero RAD studio compliers. While Eclipse compilers provide to work with Java programming language, whereas Eclipse and Embarcadero RAD studio compliers provides to work with Delphi and C++ programming language. It was possible to make applications for the Android system by using Java programming language before, but now we can make applications by using Delphi, C++ programming language. At the third given as a practical chapter of this work. At this chapter it is given applications making process by using Android Bookmaker program and by using current data we managed to make the *.apk applications. As a conclusion it is important to note we learned while making this work the android system and android system based programs. We can suggest this work for future investigation for dependent studying of professor-teachers and students.