ISEL / ADEETC

Master in Communication and Multimedia Network Engineering

Interactive Multimedia Applications

Tutorial 1

Interactive Multimedia Applications

Hello Mobile World

Rui Jesus

Introduction

This work aims to introduce the integrated development environment (IDE), Android Studio, used for the developing of Android Applications. The tutorial begins with the download and installation of the necessary tools. The following is to build a simple application typically called "Hello World". Below is displayed a link that should be consulted during the development of this work.

Android Developers

http://developer.android.com/training/index.html

Note: this tutorial should be done in class and the code of the resulting Android projects must be delivered through the Moodle platform until the 4th of March.

Laboratory Work

Download and installation of the Android Studio (Android official IDE)

1. Download the Android Studio on link http://developer.android.com/sdk/index.html.

The Android Studio package includes all the tools necessary for the development of Android applications:

"Android Studio IDE"

"Android SDK tools"

"Android 8.0 (Oreo) Platform"

"Android Emulator system image" (allows to create AVD's - Android Virtual Devices to simulate some Android devices)

Because the programming language used is Java you must have also installed, the latest release of the Java Development Kit (JDK) and the Java Runtime Environment (JRE).

2. Run the “android-Studio.exe” file obtained by the previous download to install the Android Studio package.

3. Run the Android Studio IDE. The first time you run Android Studio you can choose whether you want to import applications or settings from a previous version (Figure 1).

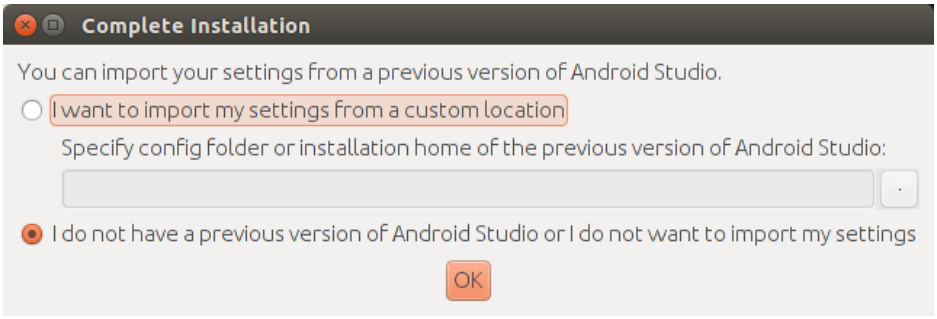

Figure 1. Android Studio after being run for the first time.

4. After clicking the "OK" button you will go through several configuration steps in order to install Android Studio (Figures 2 to 5).

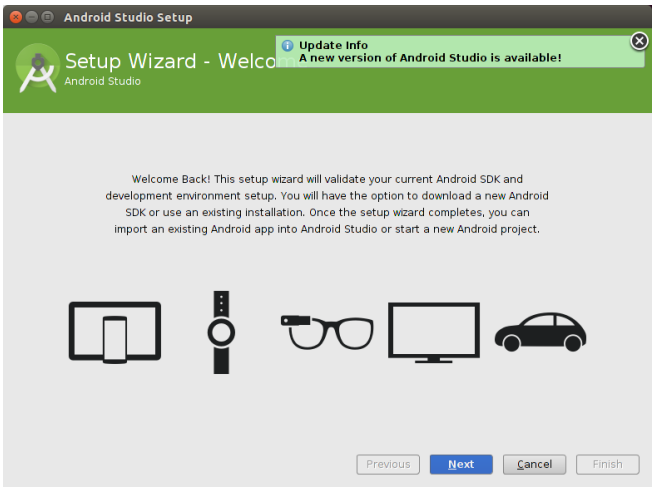

Figure 2. SETUP – Start Setup Wizard.

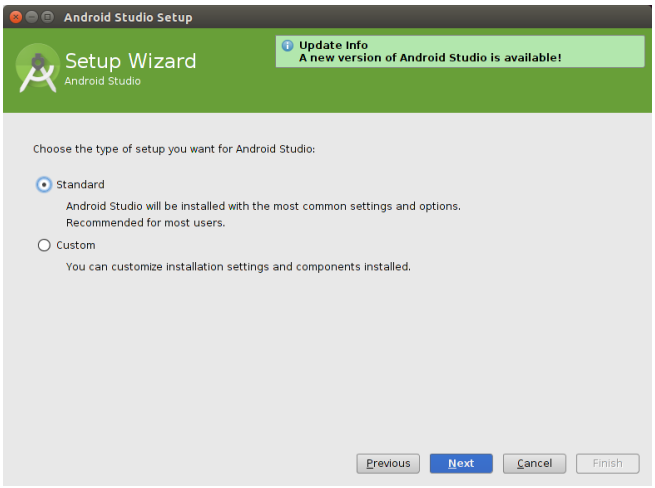

Figura 3. SETUP – Option "Standard".

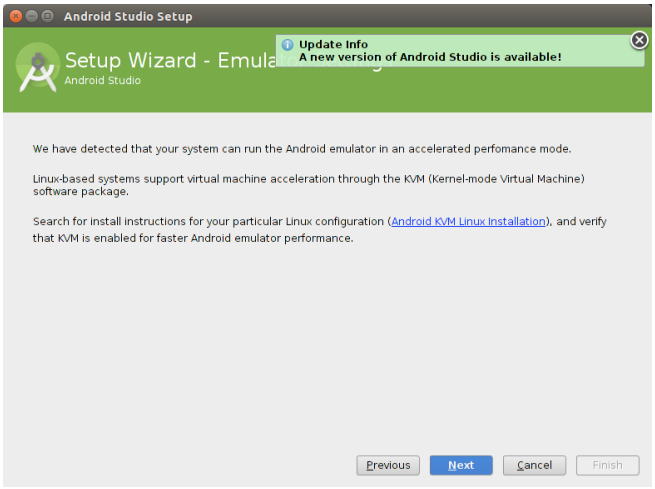

Figure 4. SETUP – Some PC's may have this option to speed up the emulator.

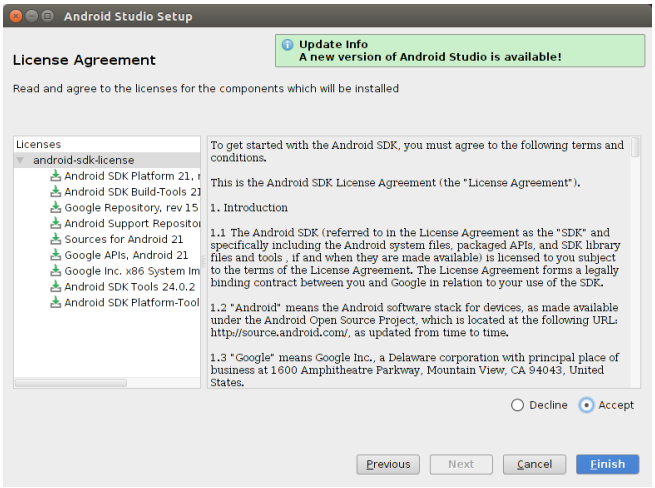

Figure 5. SETUP - Finishing the installation.

Updates

5. Run the Android Studio app. In the first panel choose the option "Start a new Android Studio project". In the following steps click on the "Next" button and the last click on the "Finish" button. Leave the various fields with the default values. You just created your first Android project. More ahead we will create a project, for now, let's go to the configuration issue.

6. All installed components of the Android Studio package should be updated. If not, proceed by opening the menu "Android Studio" (see Figure 6) and selecting the **Check for Updates** option (**Do not install Beta versions**).

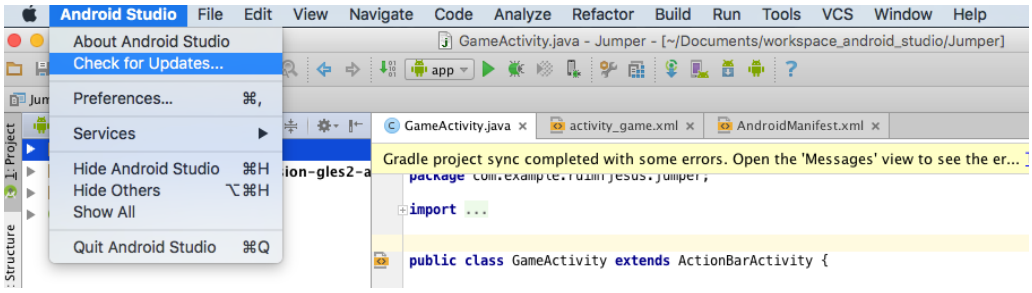

Figure 6. Android Studio – **"Android Studio"** menu and option **"Check for Updates"**.

Android SDK Manager Tool

7. The Android SDK Manager allows you to install/update a specific version of the Android API or a specific library. To install or update, proceed by opening the menu Tools -> Android -> SDK Manager or by selecting the toolbar icon (see Figure 7) of the SDK Manager to open the window.

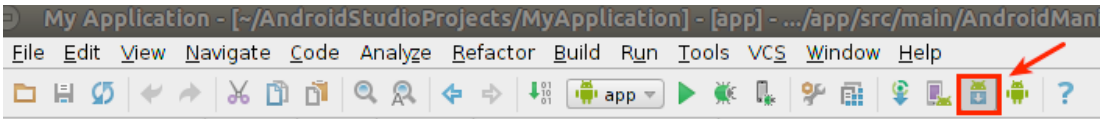

Figure 7. SDK manager icon (red element in the toolbar).

8. In the SDK Manager panel (see Figure 8) select the Android version you wish to develop and press the "OK" button. In the figure, select to install version 19 of the Android API or the latest version that is compatible with your device. Select "Show Package Details" to see in detail the components that are part of each version of the Android API.

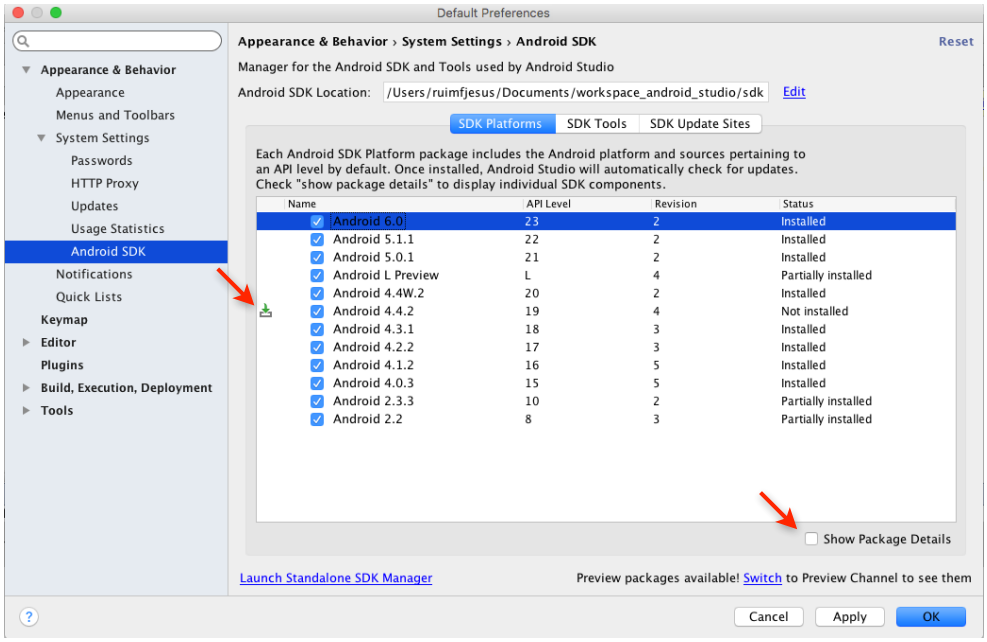

Figure 8. SDK Manager – The "SDK Platforms" tab is used to install Android API versions and the "SDK Tools" tab is used to install development tools..

Android Virtual Devices (AVD)

9. An AVD is an emulator that emulates a smartphone, tablet, or any other Android device on the PC. An AVD allows you to test Android applications in different Android versions and settings without using an actual device. To create/configure an AVD, proceed by selecting the **Tools** menu and choosing the **Android Virtual Device Manager** option (or click the **AVD Manager** icon in the toolbar). Then, click the "Create Virtual Device" button to get the panel to create and configure an AVD (see figure 9).

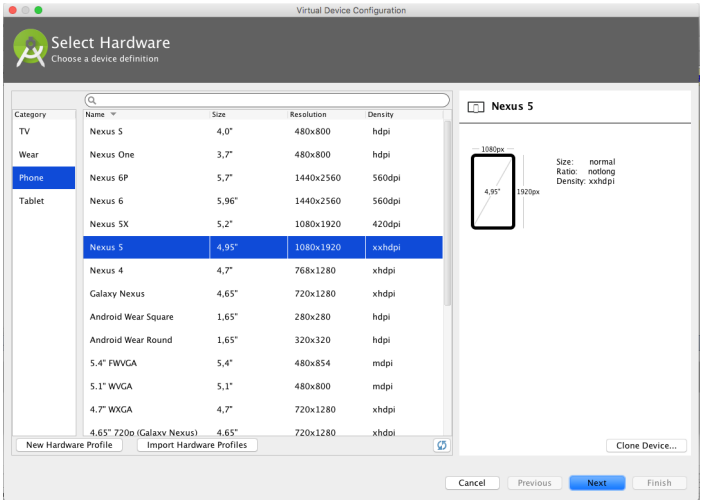

Figure 9. Virtual Device Configuration panel.

10. Create an emulator for one **Nexus 5** smartphone and another one for a **Nexus 10** tablet. In the panel of Figure 9, click the "Next" button and in the following panel proceed by selecting the KitKat, API Level 19, x86, Android 4.4 and click the "Next". The remaining parameters, for now, are set to "default".

    Note 1: You should select the latest Android API version that is compatible with your device so that you can run applications on the virtual device and on your device without having to make changes to the code.

11. After creating the two emulators in the **Android Virtual Device Manager** panel, a list of all emulators created is displayed (see Figure 10). Click the "Play" button to launch the **Nexus 5** smartphone emulator.

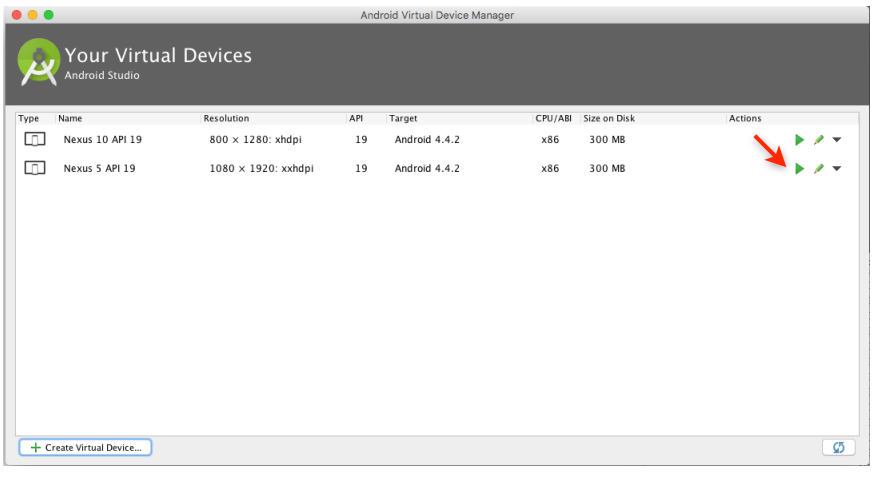

Figure 10. Two emulators: Nexus 10 tablet and Nexus 5 smartphone.

Optimization

Note 2: If you stop an AVD during the process you are booting (after pressing the "Play" button), the AVD may become corrupted. The first time an AVD is released can take up to 10 minutes (depending on the computing power of the PC).

Note 3: The AVD after being started should not be stopped / closed during development. If you make changes to the code, just do the "re-deploy" of the application in the AVD.

Note 4: During the creation of an emulator you can choose between the options: "Snapshot" or "Host GPU". If you choose the first option, in case of stopping the AVD, the launch / start of the emulator is fast because the AVD saves the state. If you choose the second option, rendering is faster because the PC graphics card is used.

Note 5: When choosing the Android API version, when creating the emulator, you can select a version where the AVD image is based on an ARM CPU architecture or an Intel CPI architecture. An AVD that is based on an Intel system is much faster if run on Intel hardware. It is not necessary to convert the ARM CPU instructions to the Intel architecture of the PC.

Hello Mobile World

Android OS was originally created by Andy Rubin in the early 21st century as a mobile operating system. In 2005, Google acquired Android Inc. and made Andy Rubin the director of Google's mobile platforms.

Android OS is an operating system based on the **Linux Kernel**. That's why Android devices are essentially Linux computers. The project responsible for developing the Android system is called "Android Open Source Project (AOSP)" and is led by Google.

When an Android program is compiled, Android uses a virtual machine called the **Dalvik Virtual Machine** (DVM) to convert the code into an optimized binary format that works as an executable but optimized for running on a small, portable device. The executable file generated by the Dalvik virtual machine, a **.DEX** file, is usually placed in the Android project folder. Following this, it is included in a single file with the **.apk** extension on the device. The **.apk** file is the application that runs on the device, includes the **.dex** files, the resources, the "assets" and the "manifest" file.

The following components may exist in an Android application:

Android **Activities** (presentation layer);

Android **Services** (background processing layer);

Android **Broadcast Receiver**s (communication layer);

Android **Content Providers** (storage layer).

Another relevant element is Android **Intent** (Communication between the various components of an application).

These elements will be introduced in more detail when they are used. For now, it's important to talk about Android **Activities**.

An **activity** is a collection of design elements that together make up each screen (layout) of the application. These elements can be user interface elements, text, graphics, 3D content, digital video, pop-up menus, animated elements and any other visual elements that make up the interface between the application and the user.

In Android terminology, an **Activity** is composed of a **Layout Container** that organizes a series of user interface elements called **Widgets**. Android **Layouts** are built based on the **ViewGroup** class and **Widgets** are used through the **View** class.

For graphics such as images or animations on Android the term **Drawables** is used. There are also **Events** and **Event Handling** functions that among other things allow you to create more robust graphical interfaces.

12. To create a new Android project, proceed by selecting **File-> New-> New Project** to open a dialog box with several options (see Figure 11).

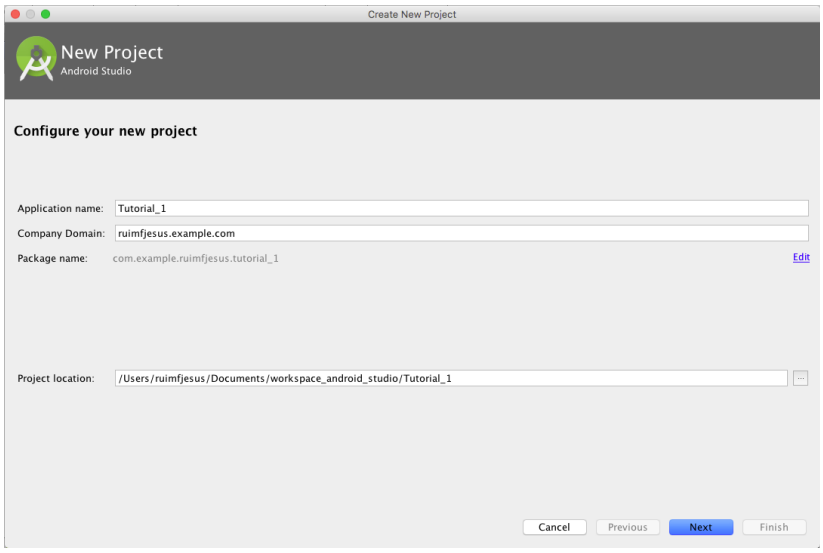

Figure 11. Window to create a new project.

13. Type the name of the application **"Tutorial_1"**. Then write the name of the package, for example, **"user_name.example.com"**. Also indicate the path where the project should be saved. Click **"Next"** and advance to the next window (Figure 12).

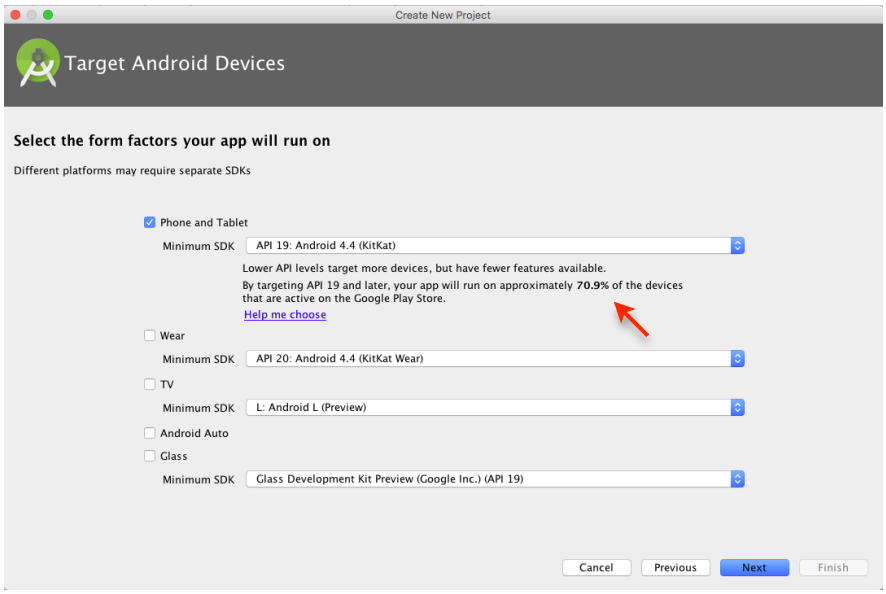

Figure 12. Project settings window - Version of Android API.

14. In the **Target Android Devices** window (see Figure 12) choose the **"Phone and Tablet"** option and the **KitKat** version of the **Android API (Android 4.4, API 19)**. The arrow in red (Figure 12) indicates an estimate of the percentage of devices where the application can run. Try changing to other versions of the Android API to see the estimate.

15. The following is the "**Add an Activity to Mobile**" window (see Figure 13). Choose the "**Blank Activity**" option and click **"Next"**. This option allows the amount of code automatically generated be minimal.

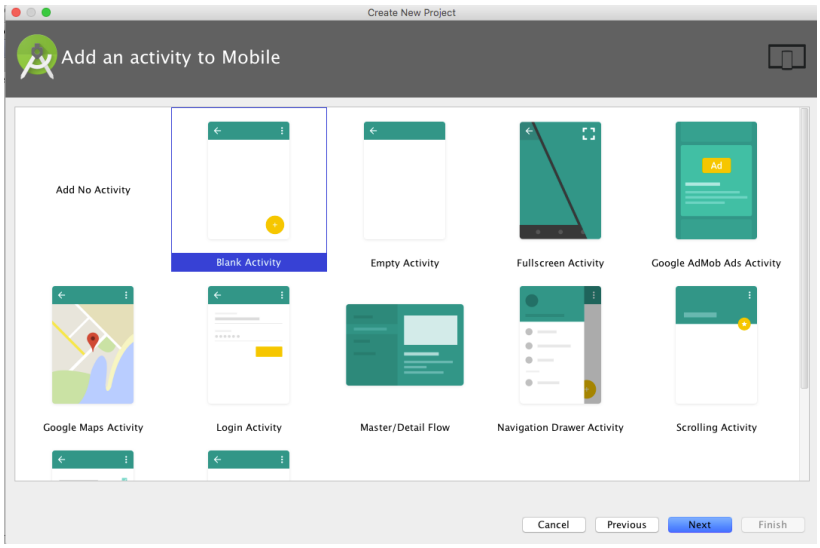

Figure 13. Activity settings window.

16. In the last window to create the project, accept the suggestions and click "**Finish**". Here we define the activity name and an XML file, "*activity_main.xml*" where the **activity Layout** is defined. The project is created.

17. To run the application we can do **Run-> Run**. The **NEXUS 5** smartphone emulator has been released previously. In the window choose your **NEXUS 5** smartphone and click the "**OK**" button.

18. The result of the application is shown in Figure 14. Try to notice the various elements of the project you have created. Change the phrase "Hello world!" to "Olá Mundo!". Repeat the previous point but now choose your device to run the application.

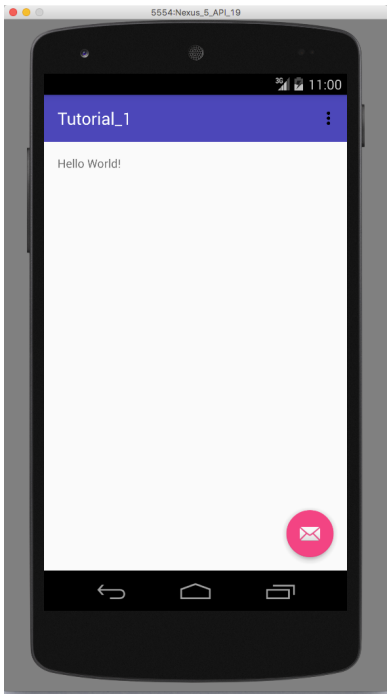

Figure 14. Final result.

19. Change the previous program to show the following phrases in separate lines: "Hello World!" and "This is my first Android application !!" (see Figure 15). Download from the web an image. Use the image and a watch in the application as in Figure 15.

Figure 15. Result obtained after the previous (19) changes.

Note 6: variables for the strings should be created in the "*string.xml*" file. To use the strings in the "content_main.xml", you should do: android:text="@string/msg1" (msg1 - variable name).

Note 7: to include more strings in the application, a <TextView …/> element should be used and to include an image, an <ImageView…/> element should be added in "content_main.xml" file.

Note 8: images used in the application should be placed in the "drawable" folder of the project. To place the image in the application, in "content_main.xml" file and in the <ImageView…/> element you should include: android:src="@drawable/smile" (smile - name of the image file).

Note 9: AnalogClock is deprecated since API 23 and for that reason it is not available in the components list of the Layouts Editor Interface. However, it can be included in the XML file related, using for instance:

```
<AnalogClock
    android:id="@+id/analogClock"
    android:layout_width="122dp"
    android:layout_height="124dp"
    android:layout_centerHorizontal="true"
    android:background="@color/ClockColor"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/imageView"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.741" />
```