Lab 3 – IDE Introduction – Installation and usability

Installing and getting familiar with the IDE. The purpose of this lab is to get familiar and set up the Android Studio environment. A brief introduction to Android Programming is provided at the end of the lab with some basic exercises.

1.1 Brief introduction to Android Studio

Android Studio became the official IDE for Android application development. It is based on ItenlliJ IDEA. Before Android Studio appeared,it was *Eclipse for Android Programming* that was mostly used for developing the mobile applications.
The following capabilities of Android Studio, makes it being even more popular for using.

- Flexible Gradle-based build system
- Build variants and multiple apk file generation
- Code templates to help you build common app features
- Rich layout editor with support for drag and drop theme editing
- Lint tools to catch performance, usability, version compatibility, and other problems
- ProGuard and app-signing capabilities
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

1.2 Set up your environment

With Android Studio it is much easier since all you need to do it to download the Studio without additional downloading.

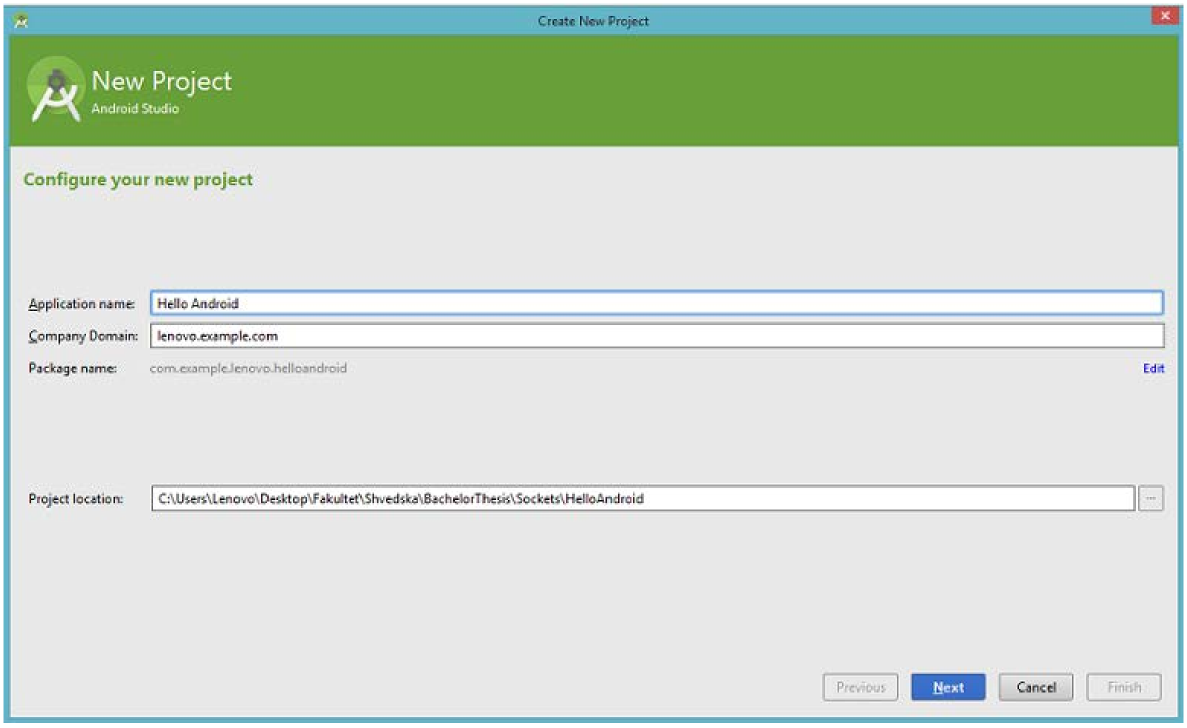
- The official download link for Android Studio: http://developer.android.com/sdk/index.html
- Download the latest SDK tools and platforms using the SDK Manager: https://developer.android.com/tools/help/sdk-manager.html

1.3 Getting started with Android

In order to start you have to create a new project. The following steps explain you how to do that:

- Open Android Studio
- Click ***File > New Project***
- Under **Configure your new Project**, name your application **HelloAndroid** and choose the Project Location
    - **Application Name** is the app name that the users can see
    - **Company domain** provides a qualifier that will be appended to the package name
    - **Package name** is fully qualified name for the Project

- **Project location** is the directory on your system that has the Project files



Application name: Hello Android
Company Domain: lenovo.example.com
Package name: com.example.lenovo.helloandroid
Project location: C:\Users\Lenovo\Desktop\Fakultet\Shvedska\BachelorThesis\Sockets\HelloAndroid

- Click ***Next***.
- If no specific requirements are needed about the mínimum API level, click ***Next***.
  - The Minimum Required SDK is the earliest versión of Android that your app supports. In order to support as many devices as posible, you should set this to the lowest versión/
- Choose ***Blank Activity***
- Click ***Next***. Give your own Activity name and layout name or leave it like this.
- Click ***Finish***

1.4 Activity

Activities are characteristic feature for Android framework. There can be many activities and all they provide the user with access to your app.
Usually there is one main activity that launches the application, another activity for when a content for viewing is selected.
As it explains on the link below “An Activity is an application component that provides a screen with which the users can interact in order to do something such as dial the phone, take a photo, send an email or view a map”.

Check the following link for more infromaiton :
http://developer.android.com/guide/components/activities.html

1.5 Android device vs emulator

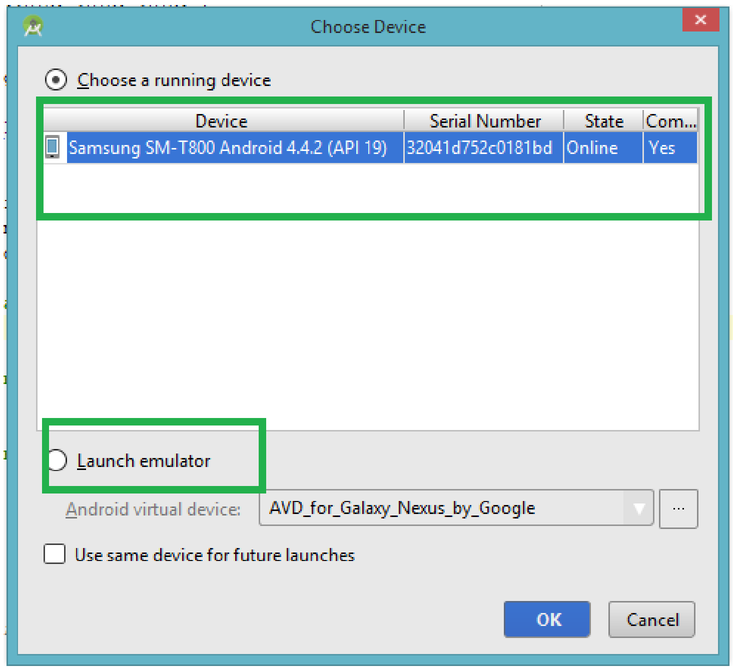
There are variety of ways to test the program that you are writing. The easiest and fastest way to do it is by having a real Android device i.e. mobile or tablet that have compatible features, considering their SDK.

- **Set up your device**

1. Plug in your device to your development machine with a USB cable.
2. Enable **USB debugging** on your device
   - on most devices running Android 3.2 or older you can find it under **Settings -> Applications -> Development**
   - on Android 4.0 and newer **Settings -> Developer options**

Note: On Android 4.2 and newer **Developer options** is hidden. Go to **Settings -> About phone** and tab **Build number** 7 times. Return back to find **Developer Options** now.

- **Emulator**


Choose Device

Choose a running device

Device	Serial Number	State	Com...
Samsung SM-T800 Android 4.4.2 (API 19)	32041d752c0181bd	Online	Yes

Launch emulator

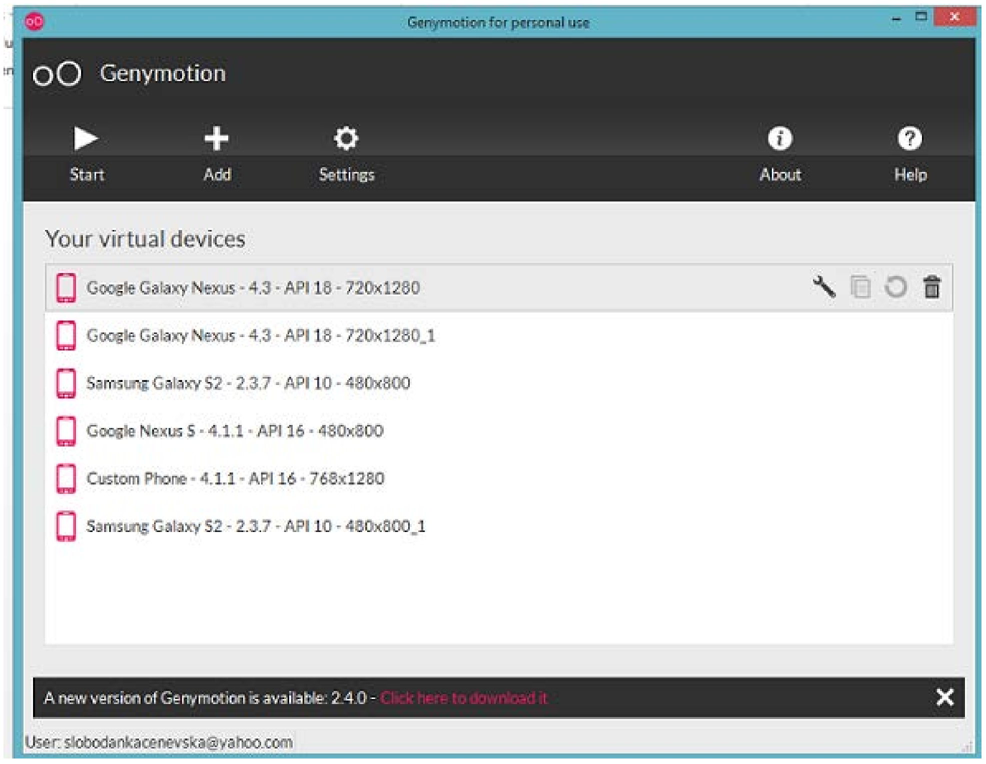
Android virtual device: AVD_for_Galaxy_Nexus_by_Google

Use same device for future launches

Since devices come with different dimensions and different features you can always test your program depending on the properties you need using an emulator.
Emulator is a practical imitation of an Android device. You can create your own Android device with the features you want and need.
Creating the device can be done from Android Studio itself. As soon you press the **Run** button in Android Studio, a window will appear asking you about how to run the program. If you don't have devices installed, you can always do that by clicking in the appropriately marked place.

You can consider **Genymotion** when working with emulators. You can download the program online. It contains variety of different emulators and working with them is fast and easy.

Note: it takes more time for the program to execute on an emulator than on a real device.

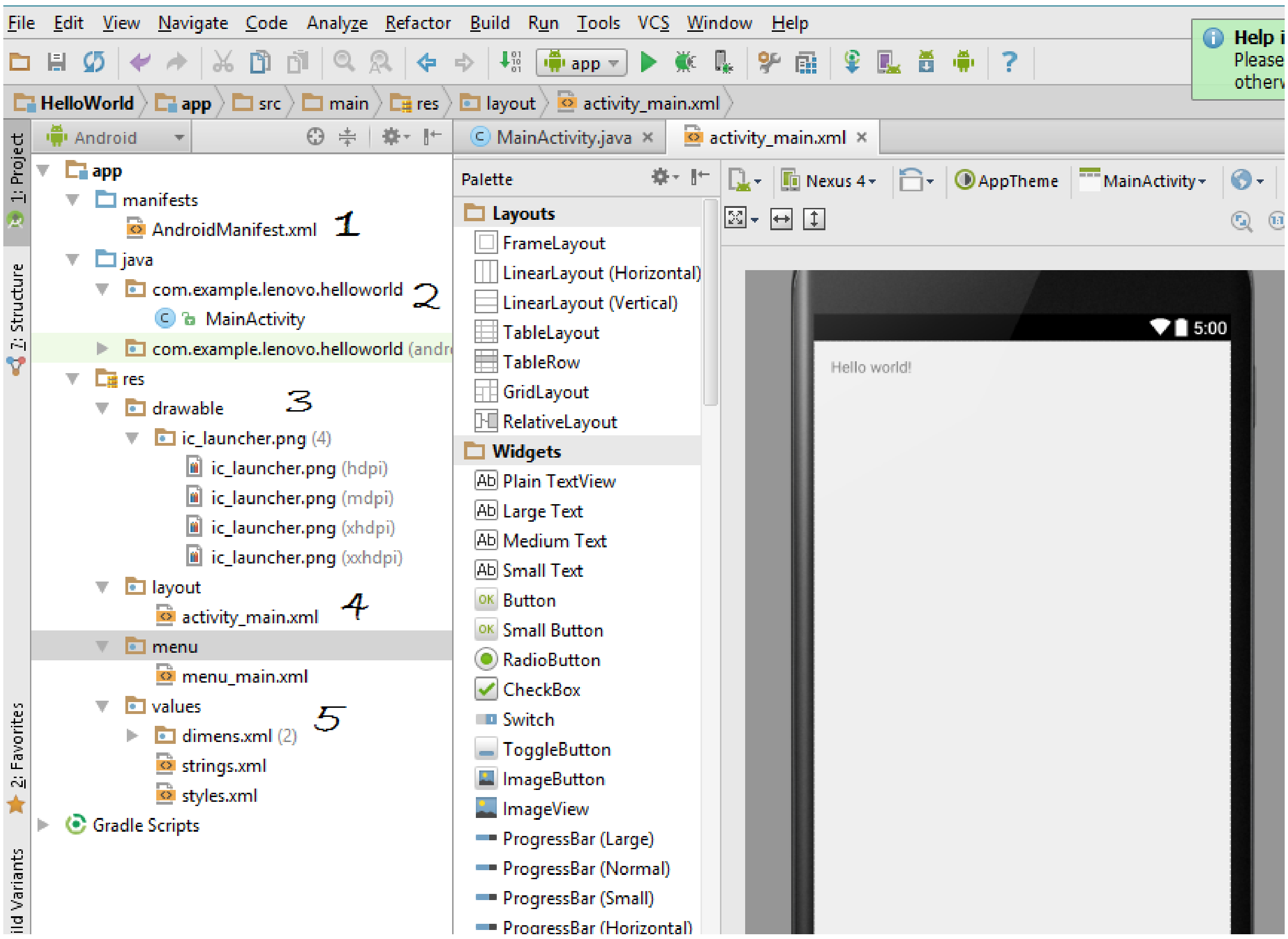


Your virtual devices

Google Galaxy Nexus - 4.3 - API 18 - 720x1280
Google Galaxy Nexus - 4.3 - API 18 - 720x1280_1
Samsung Galaxy S2 - 2.3.7 - API 10 - 480x800
Google Nexus S - 4.1.1 - API 16 - 480x800
Custom Phone - 4.1.1 - API 16 - 768x1280
Samsung Galaxy S2 - 2.3.7 - API 10 - 480x800_1

A new version of Genymotion is available: 2.4.0 - Click here to download it

User: slobodankacenevska@yahoo.com



1. **AndroidManifest.xml** – This is the manifest file which describes the fundamental characteristis of the app and defines each of its components
2. The main activity code is a Java file i.e. **MainActivity.java**. This is the actual application file.
3. **Res/drawable**-hdpi – This is a directory for drawable objects that are designed for high-density screens.
4. **Res/layout** - this is a directory for files that define your app's user interface.
5. **Res/values** – This is a directory for other various XML files that contain a collection of resources, such as strings and colors definitions.

- **Gradle**

Android Studio uses Gradle to compile and build your app. There is a build.grafle file for each module of your Project as well as a build.gradle file for the entire Project.
More information about Gradle: http://developer.android.com/sdk/installing/studio-build.html

- ## The R.java file

R.java is an auto generated file when you build your Android application. Every application uses resources such as strings, layouts, drawbles, styles etc.
What happens here is, all the resources in your program will be created as one class. They will be created as static members so their values are not supposed to be changed. So, instead of coding such resources into an application, one externalizes them and refers to them by ID. The **R.java** contains all those resource IDs.
These resources are accessed, for example, like **R.drawable.ic_launcher**.

- ## How to execute?

In the top menu click at **Run** > **Run 'app'**.
The window previously described will appear and will ask you for what device to use in order to execute the program.
Choose the suitable device for you and procceed further. This will install the new program on your device. After installing it, you can always start the program without using a cable, just like downloading an app from the **Play Store**.

Tips:

- Try to play around with the components, make some simple program on your own and investigate a little bit.
- Review the "Introduction to Android Programming" document provided.
- If you have any problems when executing the code always read the Log output as it will point out what your error is.

References:

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

R.java :
http://developer.android.com/guide/topics/resources/accessing-resources.html
http://www.yugandroid.in/android-tutorials/r-java-file.html