ACDS Lecture Series

Lecture - 20

Android Studio

G. N. Sastry and Team

ADVANCED COMPUTATION AND DATA SCIENCES (ACDS) DIVISION

CSIR-North East Institute of Science and Technology, Jorhat, Assam, India

20. Android Studio

20.1 Introduction

- Android is a mobile operating system that is based on a modified version of Linux.
- It was originally developed by a start-up company, Android, Inc. which was later purchased by Google.
- Google released Android Code under the open source Apache License.
- Users can simply download the full Android source code and add their own proprietary extensions to Android and customize Android to differentiate their products from others.
- Android offers a unified approach to application development.

20.1.1 History.

2003 – Android Inc., founded by Andy Rubin and backed by Google, was born.

2005 – Google bought Android Inc.

2007 – Android was officially open source. Google turned over its ownership to the Open Handset Alliance (OHA).

2008 – version 1.0 released.

2009 – versions 1.1, 1.5 (Cupcake), 1.6 (Donut), and 2.0 (Eclair) were released.

2010 – versions 2.2 (Froyo), and 2.3 (Gingerbread) were released.

2011 – 3.0 (Honeycomb) and 4.0 (Ice Cream Sandwich) were released.

2012 – version 4.1 (Jellybean)

2013 – version 4.4 (Kitkat)

2014 – versions 5.0-5.1 (Lollipop); Android became 64-bit.

2015 – version 6.0 (Marshmallow).

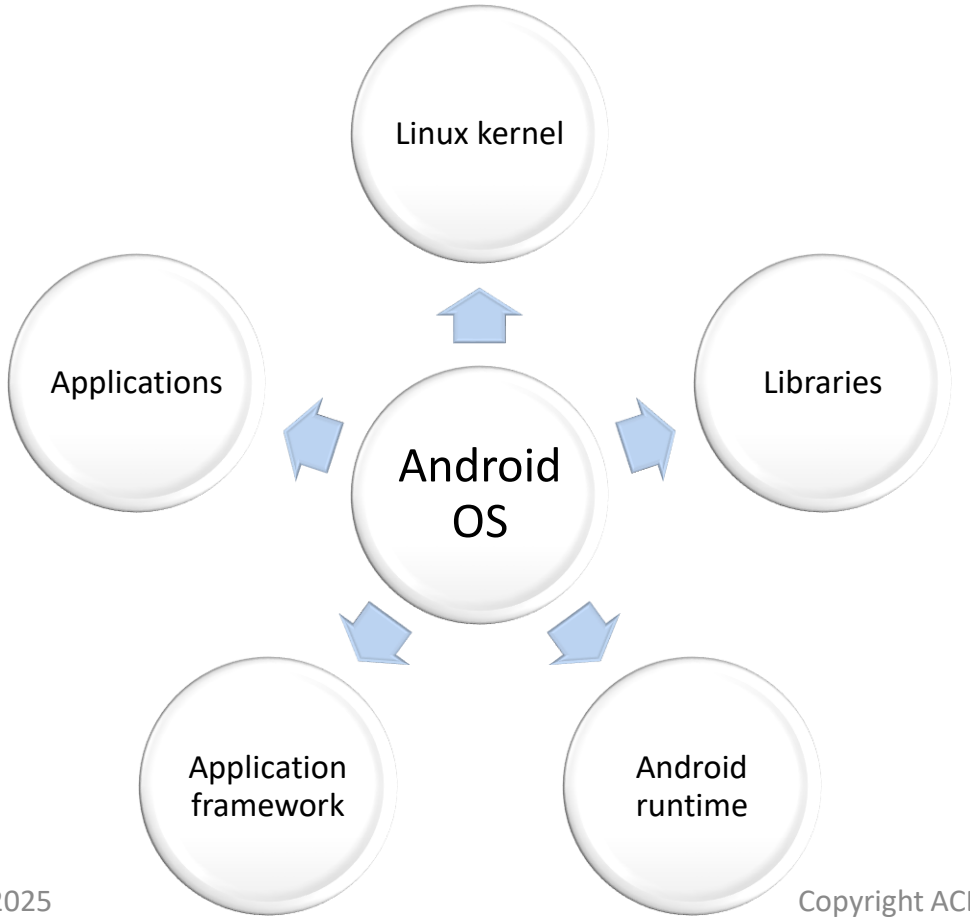
2016 – versions 7.0-7.1.2 (Nougat)

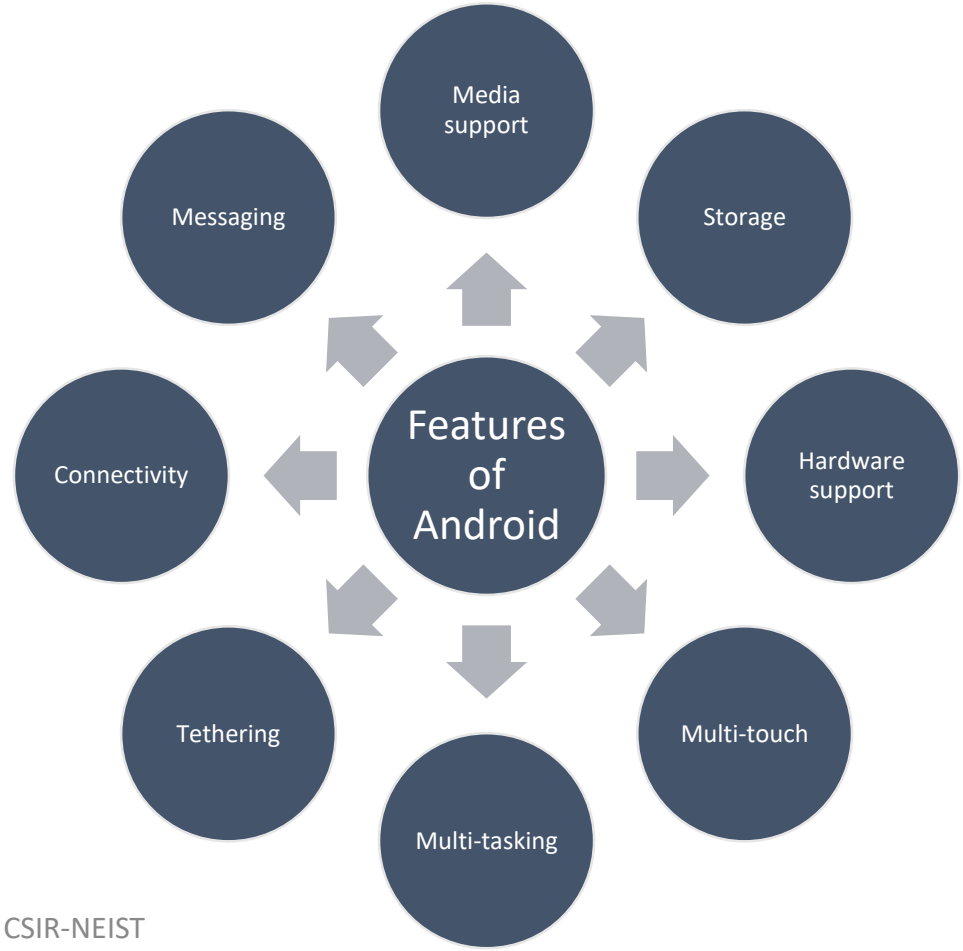
2017 – version 8 (Oreo)

20.1.2 Operating system (OS)

OS is the most visible part of android that stands between a user and the hardware. An OS does three things-

- Manages hardware on behalf of applications
- Provides services to applications like networking, security and memory management
- manages execution of applications





20.2.1 Android studio set up

- The first and the most important piece of software you need to download is Android Studio 2/3.
- After you have downloaded and installed Android Studio, you can use SDK Manager to download and install multiple versions of the Android SDK.
- Having multiple versions of the SDK available enables you to write programs that target different devices.
- The Android Studio 3 is available for MAC OS, Windows, and Linux. The download page detects the OS you are using, so you should be able to spot the download button fairly quickly.
- The Android SDK contains all of the packages and tools required to develop a functional Android application.

20.2.1.1 mac OS

The installation notes for AS3 are at https://developer.android.com/studio/preview/install-preview.html

1. Unpack the zipped file
2. Drag the application file into the applications folder
3. Launch AS3
4. AS3 will prompt you to import some settings

20.2.1.2 Windows

1. Unzip the installer file
2. Move it to a folder location of your choice (e.g., C:\AndroidStudio). Drill down to this folder
3. Inside, you will find a bin folder; inside it, you will find studio64.exe. This file is what you need to launch. If you are on a 32-bit Windows, the launcher file is named studio.exe

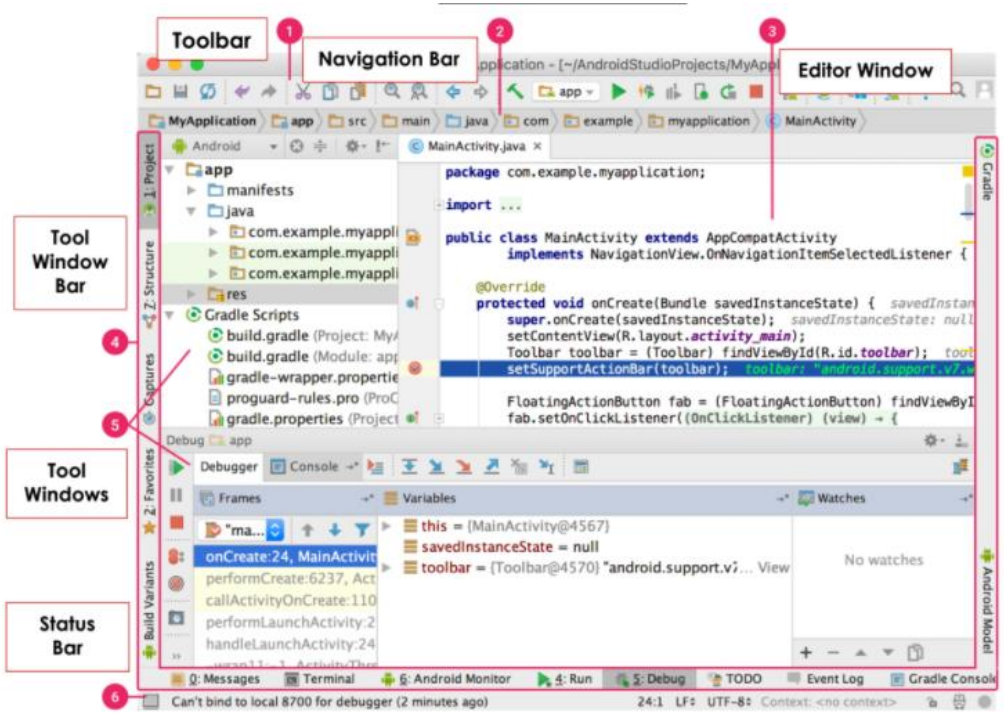
Windows	Mac
Microsoft Windows 7/8/10 (32- or 64- bit)	Mac®OS X® (Yosemite) or higher, up to 10.12 (macOS Sierra)
3GB RAM minimum, 8 GB RAM recommended, plus 1GB for Android Emulator	3GB RAM minimum, 8 GB RAM recommended: plus 1 GB for the Android Emulator
2 GB of available disk space minimum,	2 GB of available disk space minimum,
4GB recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)	4GB recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)
1280 x 800 minimum screen resolution	1280 x 800 minimum screen resolution
For accelerated emulator: Intel® processor with support for Intel® VT-x, Disable (XD) Bit functionality	

20.2.1.3 Linux

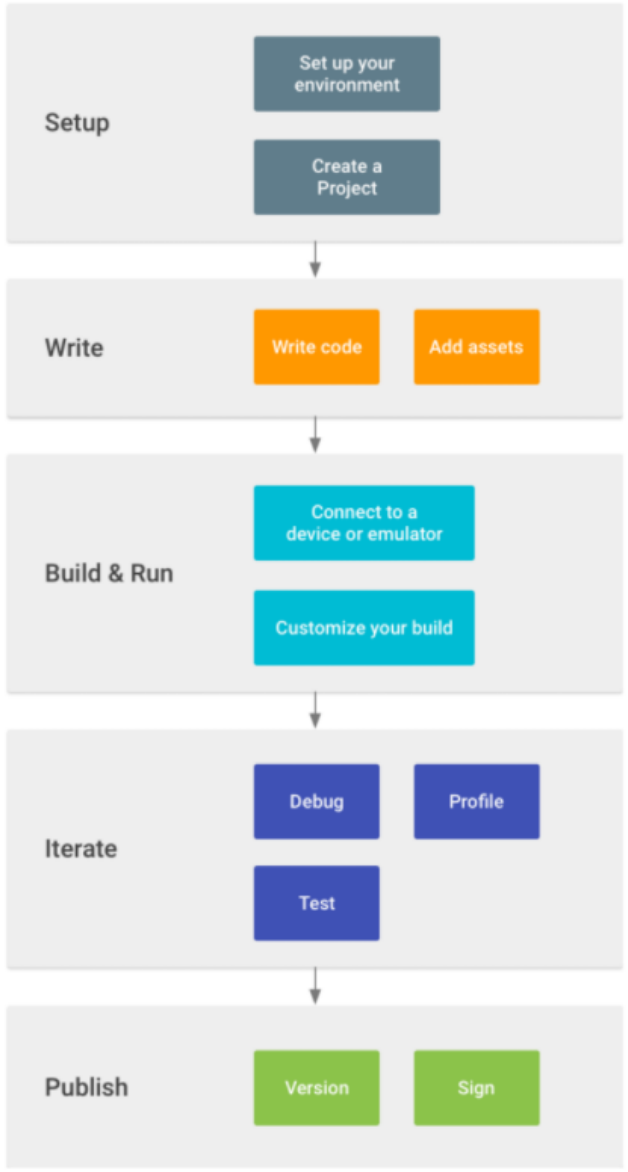
1. Unpack the downloaded installer file. You can unpack the file using command-line tools or using the GUI tools; you can, for example, right-click the file and select the “Unpack here” option, if your file manager has that option
2. After unzipping the file, rename the folder to AndroidStudio
3. Move the folder to a location where you have read, write, and execute privileges. Alternatively, you can also move it to /usr/local/ AndroidStudio
4. Open a terminal window and go to the AndroidStudio/bin and execute ./studio.sh
5. At first launch, AS3 will ask you if you want to import some settings; if you have installed a previous version of Android Studio, you may want to import those settings into AS3

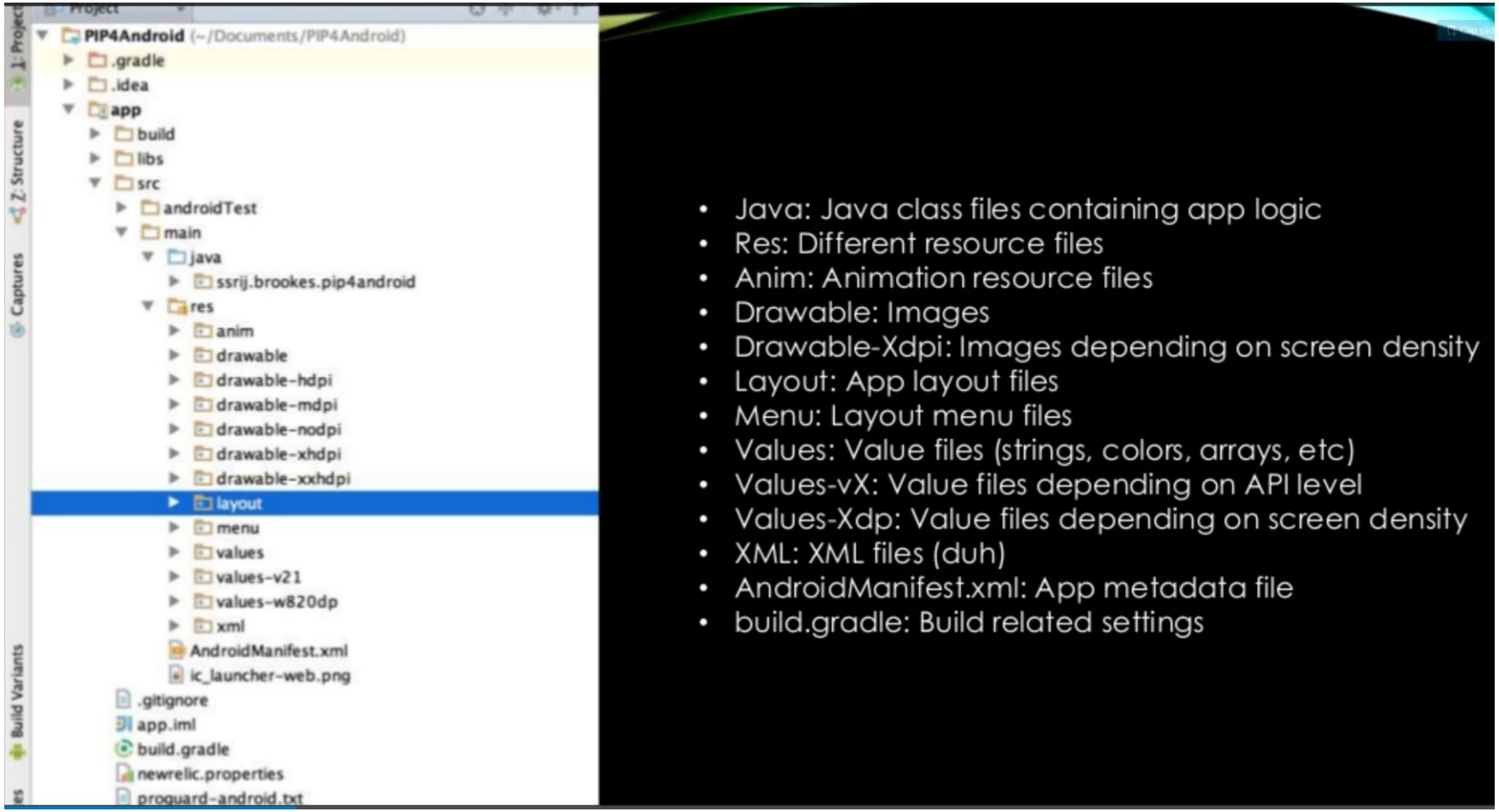
20.2.2 The User Interface

- The **toolbar** lets you carry out a wide range of actions including running your app and launching Android tools.
- The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the Project window.
- The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file.
- The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
- The **tool windows** give you access to specific tasks like project management, search, version control and more. You can expand them and collapse them.
- The **status bar** displays the status of your project and the ICE itself, as well as any warnings or messages.



20.2.3 Developer basic workflow





20.2.4 Configuring Android Studio

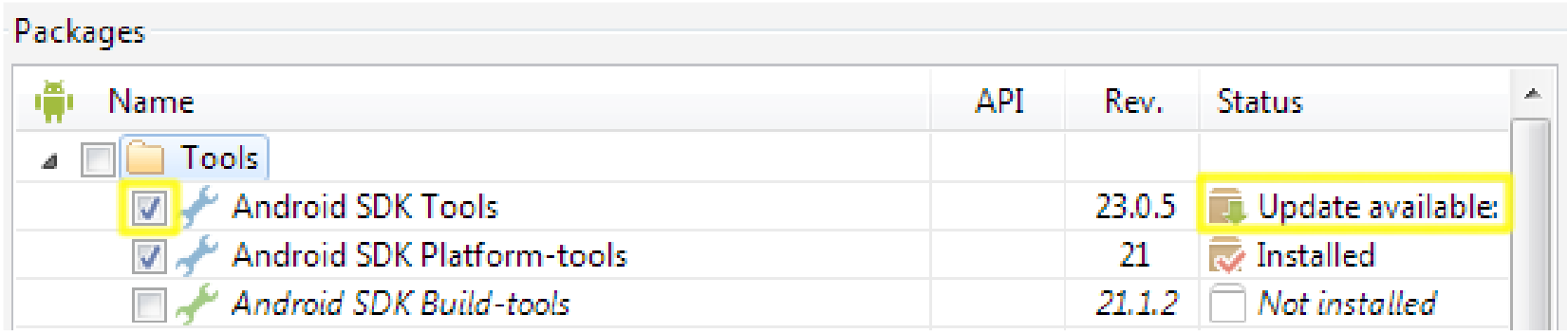
- From the opening dialog, click “Configure” and choose “SDK Manager” from the drop-down list. This should take you to a window where you can select the SDK platforms to download.
- When you get to the SDK window, enable the “Show Package Details” option so you can see a more detailed view of each API level.
- SDK levels or platform numbers are specific versions of Android. Android 8 or Android “O” is API level 26, and Nougat is API level 24 and 25.
- Once you have completed the selection, click the “OK” button to start the download process.

20.2.5 Hardware acceleration

- Running on an emulator can sometimes be slow;, this why Google and Intel came up with HAXM.
- It is an emulator acceleration tool that makes testing your app a bit more bearable.
- This is definitely a boon to developers, that is, if you are using a machine that has an Intel processor which supports virtualization and that you are not on Linux.
- macOS users probably have it the easiest because HAXM is automatically installed with AS3.

20.3 Tools

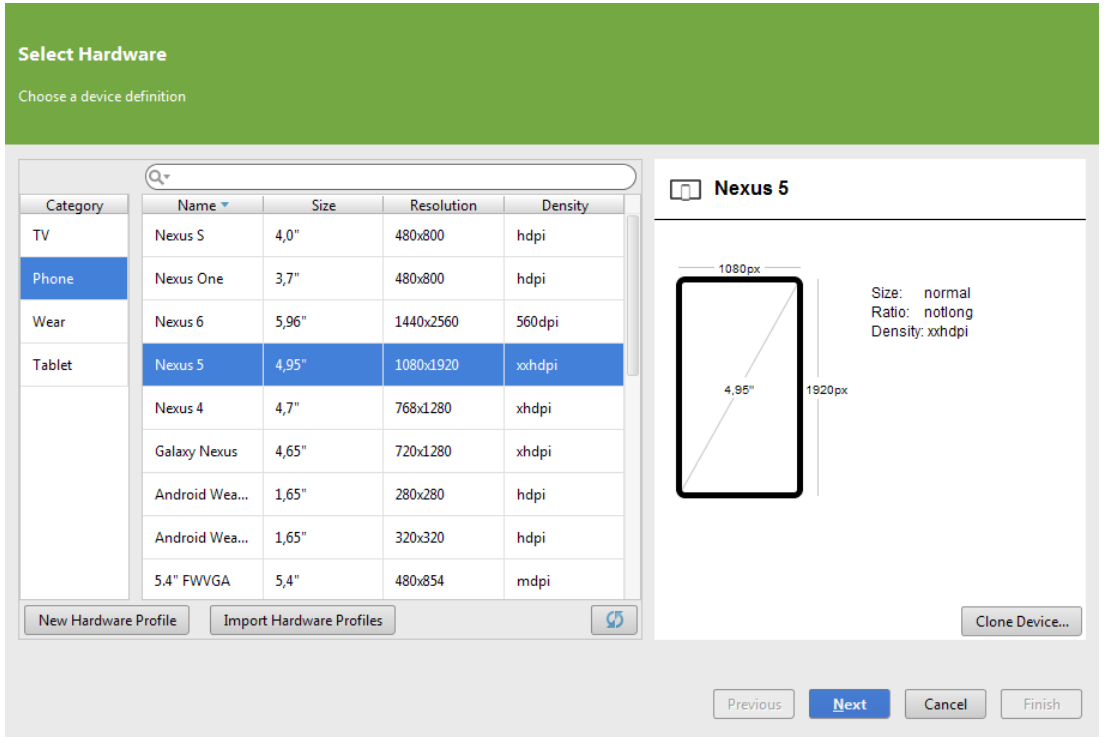
20.3.1 SDK Manager



- The SDK Manager is an Android tool accessible from Android Studio to control our Android SDK installation.
- From this tool, we can examine the Android platforms installed in our system, update them, install new platforms, or install some other components such as Google Play services or Android Support Library.
- The SDK Manager displays the list of available packages with the following properties:
- **Name**: This is the name of the package or the container that aggregates related packages.
- **API**: This is the API number in which the package was added.
- **Rev.**: This is the package revision or version.
- **Status**: This is the status of the package on your system. The status can be **Not installed**, **Installed**, **Update available**, **Not compatible**, or **Obsolete**.

20.3.2 AVD Manager

- The AVD Manager is an Android tool accessible from Android Studio to manage the Android virtual devices that will be executed in the Android emulator.
- The AVD Manager displays the list of existing virtual devices.
- To create our first virtual device, click on the **Create Virtual Device** button to open the configuration dialog.
  - The first step is to select the hardware configuration of the virtual device.
  - The hardware definitions are listed in the left-hand side of the window.
  - Select one of them, such as **Nexus 5**, to examine its details on the right-hand side, as shown in the following screenshot.
  - Hardware definitions can be classified into one of these categories: **Phone**, **Tablet**, **Wear**, or **TV**.



20.3.3 The navigation editor

- The Navigation Editor is a tool used to create and structure the layouts of the application using a graphical viewer.
- This tool opens a file in XML format, named main.nvg.xml. This file is stored in your project at /.navigation/app/raw/.
- The navigation editor only shows this main layout.
- If you select the layout, detailed information about it is displayed on the panel on the right-hand side of the editor. If you double-click on the layout, the XML layout file will be opened in a new tab.
- We can create a new activity by right-clicking on the editor and selecting the **New Activity** option. We can also add transitions from the controls of a layout by shift clicking on a control and then dragging to the target activity.

20.3.4 Generating a Javadoc

- A Javadoc is a utility to document Java code in HTML format.
- The Javadoc documentation is generated from comments and tags added to Java classes or methods.
- The use of a Javadoc is integrated in Android Studio.
- To generate a complete Javadoc, we have to write the Javadoc comments about our classes and methods.
- The Javadoc comments are automatically inserted containing the available information from the method declaration: parameters and return type.
- To generate the Javadoc documentation, navigate to **Tools | Generate Javadoc...**.
- We can choose the scope, output directory, and visibility of the included elements, and create a hierarchy tree, a navigation bar, and an index if needed.

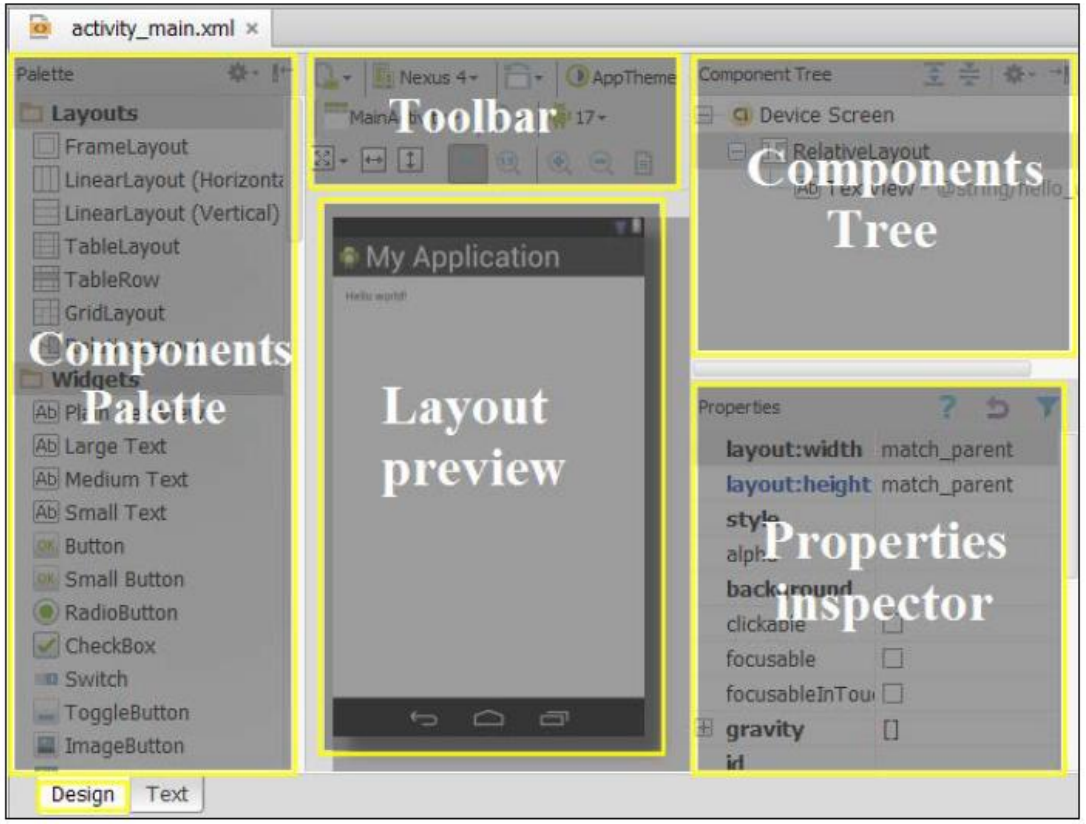
20.3.5 Version control systems

- Android Studio integrates some **version control systems** (**VCS**): GitHub, CVS, Git, Mercurial, and Subversion.
- To enable version control integration, navigate to **VCS | Enable Version Control Integration** and select the type of system.
- Options will be added to the **VCS** menu:
  - To update the entire project, navigate to **VCS | Update Project**
  - To commit all the changes of the project, navigate to **VCS | Commit Changes**
  - To clean up the project, navigate to **VCS | Cleanup Project**
- The first step is to do the checkout from the version control system.
- The version control actions can also be applied to individual files.
- A simpler way to control the file versions is by using the **Local History** option.

20.4 The Android Studio interface:

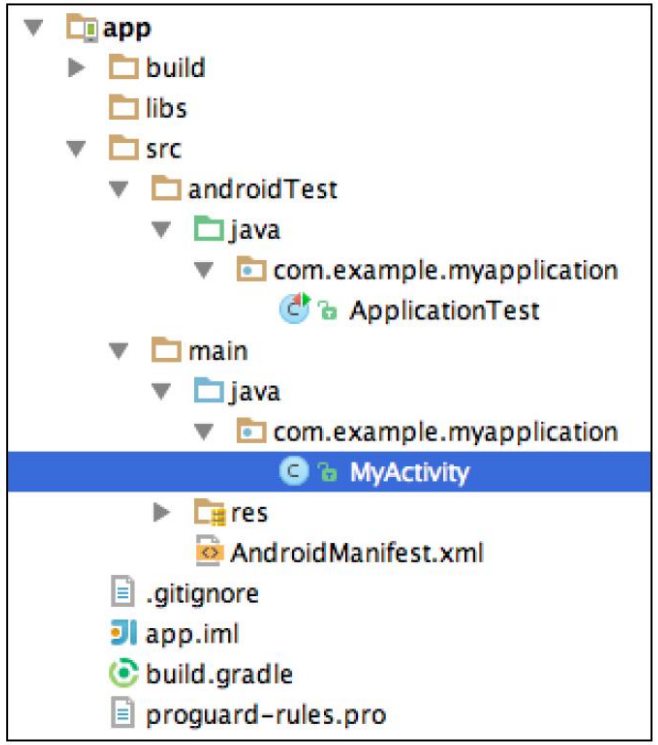
20.4.1 General layout

- Graphical editor
  - **Toolbar** contains some options that can be used to change the layout style and preview.
  - **Components Tree** displays the components placed in the layout as a hierarchy.
  - **Properties inspector** shows the properties of the selected component from the layout and allows us to change them.
  - **Components Palette** lists the existing **user interface** (**UI**) components to place in our layout.



20.4.2 Project structure

- We can examine the project structure in the project navigation pane.
- The project structure includes a folder with the name of our application.
- This folder contains the application structure and files.
- The most important elements of the application structure are in the app directory. These include:
  - build/: This is a folder that contains the resources compiled after building the application and the classes generated by the Android tools,
  - libs/: This is a folder that contains the libraries referenced from our code.
  - src/androidTest/: This is a folder that contains the test classes of the Java classes that need to be tested.
  - src/main/: This is a folder that contains the sources of our application.
  - java/: This is a folder that contains Java classes organized as packages
  - res/: This is a folder that contains project resources such as the XML files that specify layouts and menus, or image files.
  - AndroidManifest.xml: This is an essential file in an Android project, which is generated automatically when we create the project.



20.5 Contents providers

20.5.1 Sharing data in Android

- In Android, using a content provider is the recommended way to share data across packages.
- Android ships with many useful content providers, including the following:
  - Browser – Stores data such as browser bookmarks, browser history etc
  - CallLog – Stores data such as missed calls, call details, etc
  - Contacts – Stores contact details
  - MediaStore – Stores media files such as audio, video, and images
  - Settings – Stores the device's settings and preferences
- Besides the many built-in content providers, you can also create your own content providers.

20.5 Debugging

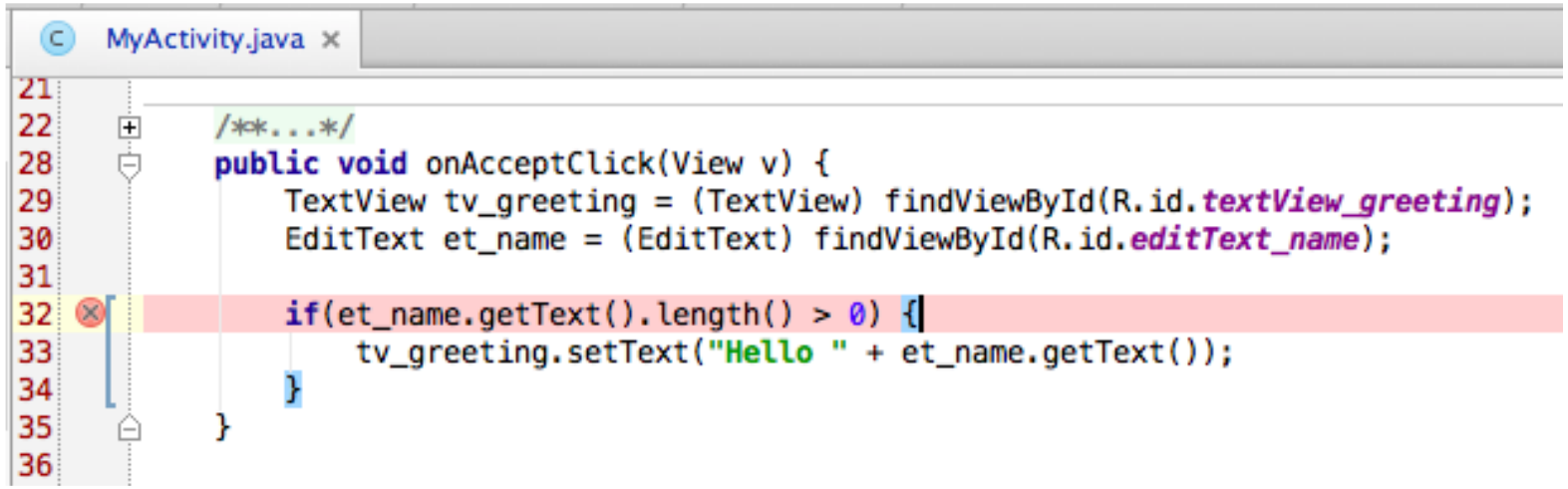
- The debugging environment is one of the most important features of an IDE.
- Using a debugging tool allows you to easily optimize your application and improve its performance.
- Android applications can be run from Android Studio in a real device using a USB connection or in a virtual device using the emulator.
- Virtual devices make it possible to test our applications on different types of hardware and software configurations.
- To debug an application, navigate to **Run | Debug 'app'** or click on the Bug icon from the toolbar. While debugging, you will notice that, at the bottom of Android Studio, there are three tabs contained in the **Debug** panel: **Debugger**, **Console**, and **LogCat**.

20.5.1 Console

- **Console** displays the events that are taking place while the emulator is being launched.
- Open it to examine the messages and check that the emulator and the application are being correctly executed.
- The actions that should appear are:
  - **Waiting for device**: This is the starting point when the emulator is being launched.
  - **Uploading file**: This event states that the application is packed and stored in the device.
  - **Installing**: This event states that the application is being installed in the device. After the installation, a success message should be printed.
  - **Launching application**: This event takes place when the application starts to execute.
  - **Waiting for process**: This event takes place when the application is running and the debug system is trying to connect to the application process in the device.

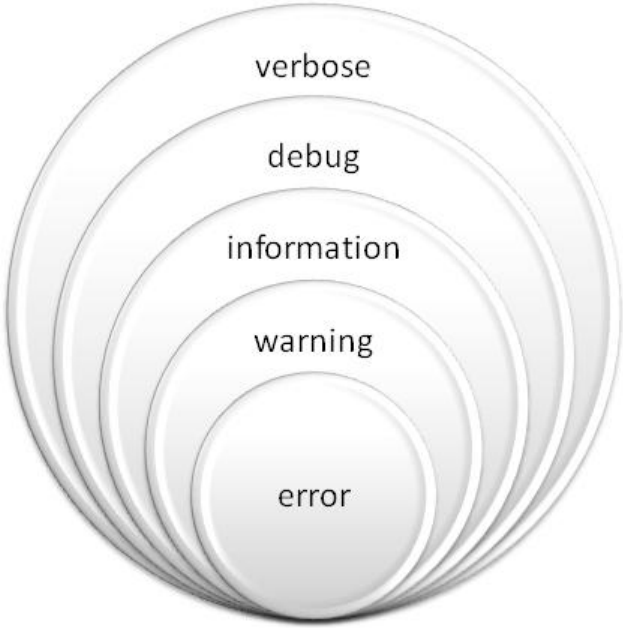
20.5.2 Debugger

- **Debugger** manages the breakpoints, controls the execution of the code, and shows information about the variables.
- From the **Debugger** tab, we can examine the method call hierarchy and the state of the variables at that point of execution.
- To add a breakpoint in our code, just click on the left edge of a line of code.
- A red point will appear next to the line of code to indicate the breakpoint.
- Add a breakpoint in the conditional statement of the onAcceptClick method of your main activity and debug the application again, as shown:



20.5.3 Logcat

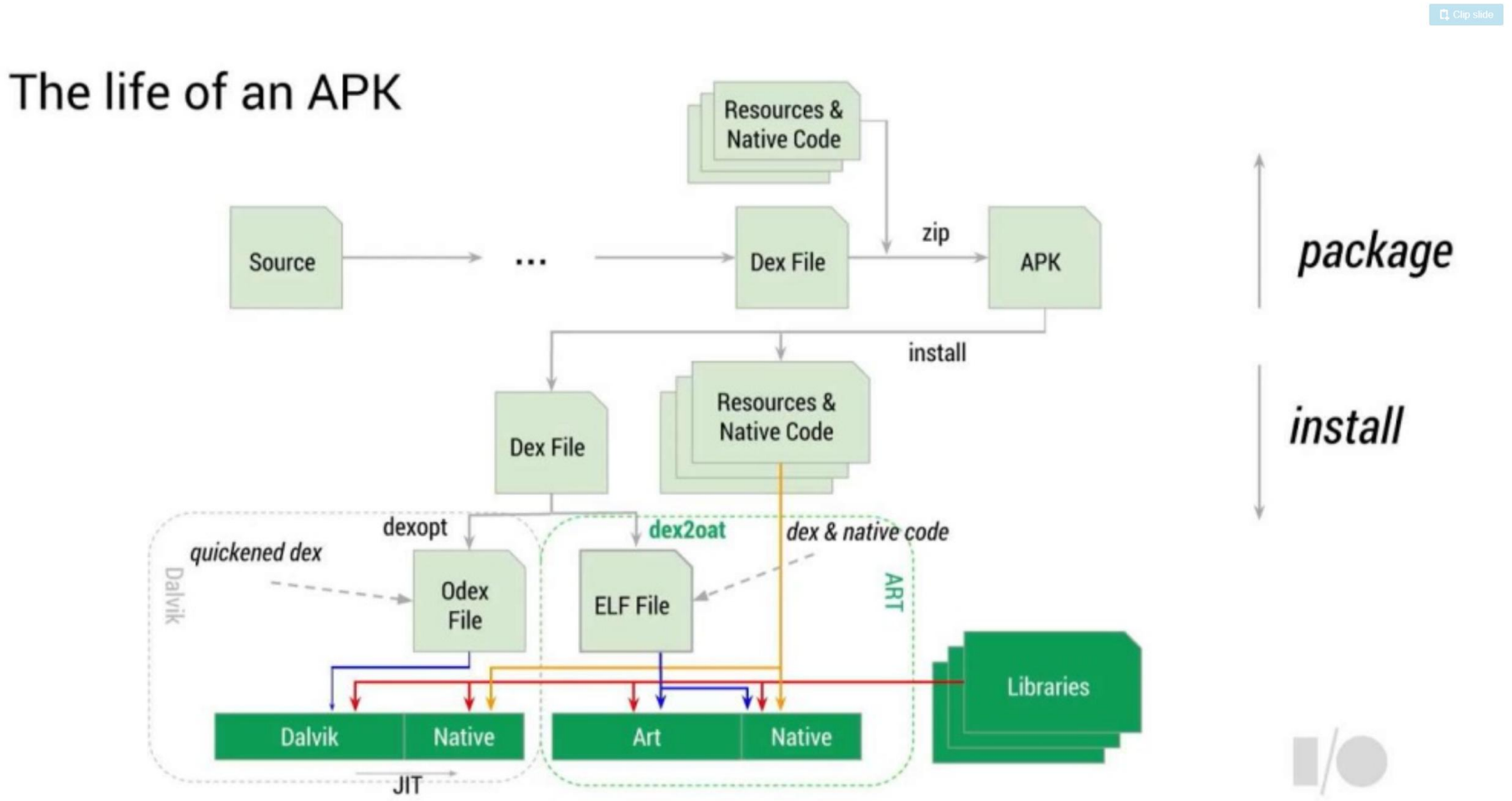
- **LogCat** is the Android logging system that displays all the log messages generated by the Android system in the running device.
- Log messages have several levels of significance.
- From the **LogCat** tab, we can filter log messages by these levels.
- For example, if we select the information level as the filter, the messages from information, warning, and error levels will be displayed.
- To print log messages from our code, we need to import the Log class.



20.6.1 Understanding an APK file

- Android applications are packed in a file with the .apk extension.
- These files are just compressed ZIP files, so their content can be easily explored.
- An APK file usually contains the following:
  - assets/: This is a folder that contains the asset files of the application. This is the same assets folder that exists in our project.
  - META-INF/: This is a folder that contains our certificates.
  - lib/: This is a folder that contains compiled code, in case it is necessary for a processor.
  - res/: This is a folder that contains the application resources such as images, strings, and so on.
  - AndroidManifest.xml: This is the application manifest file.
  - classes.dex: This is a file that contains the application's compiled code.
  - resources.arsc: This is a file that contains some precompiled resources such as binary XML files.

20.6.1 Understanding an APK file



20.6.2 Steps to take before releasing your app

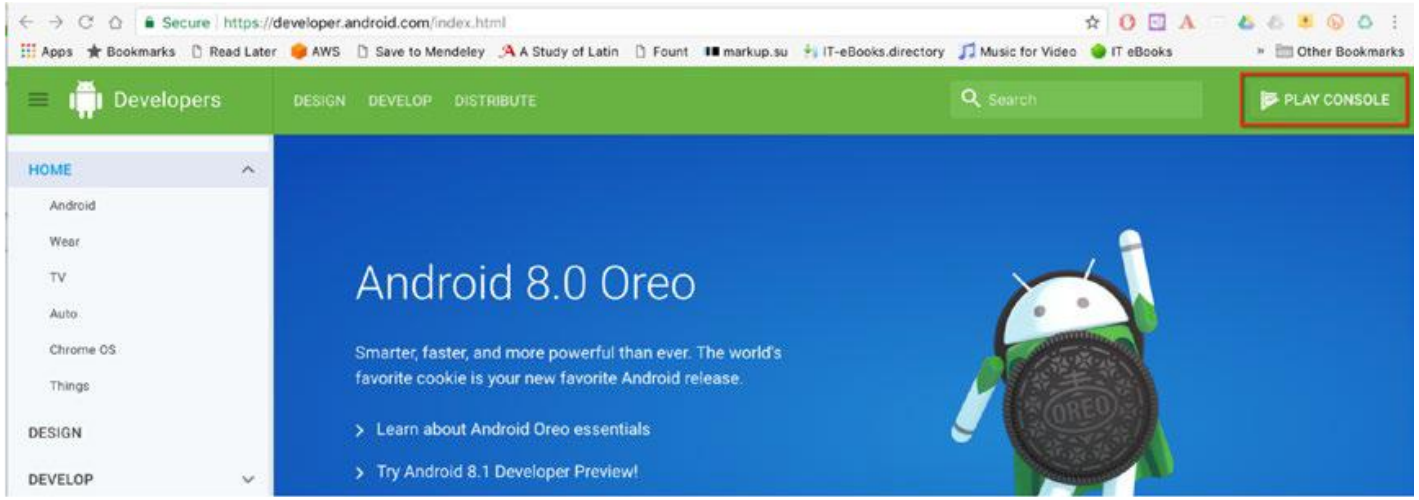
- Firstly, make sure you have completely tested your application
- Secondly, we have to check the log messages that are printed from our application.
- Thirdly, if your application communicates with a server, check whether the configured URL is the production URL. It is possible that during the debug phase, you referenced to a URL of a server in a prerelease environment.
- Finally, set the correct value for the android:versionCode and android:versionName properties from the application manifest file. The version code is a number (integer) that represents the application version. The version name is a string that represents the application version.
-

20.6.3 Generating a signed APK

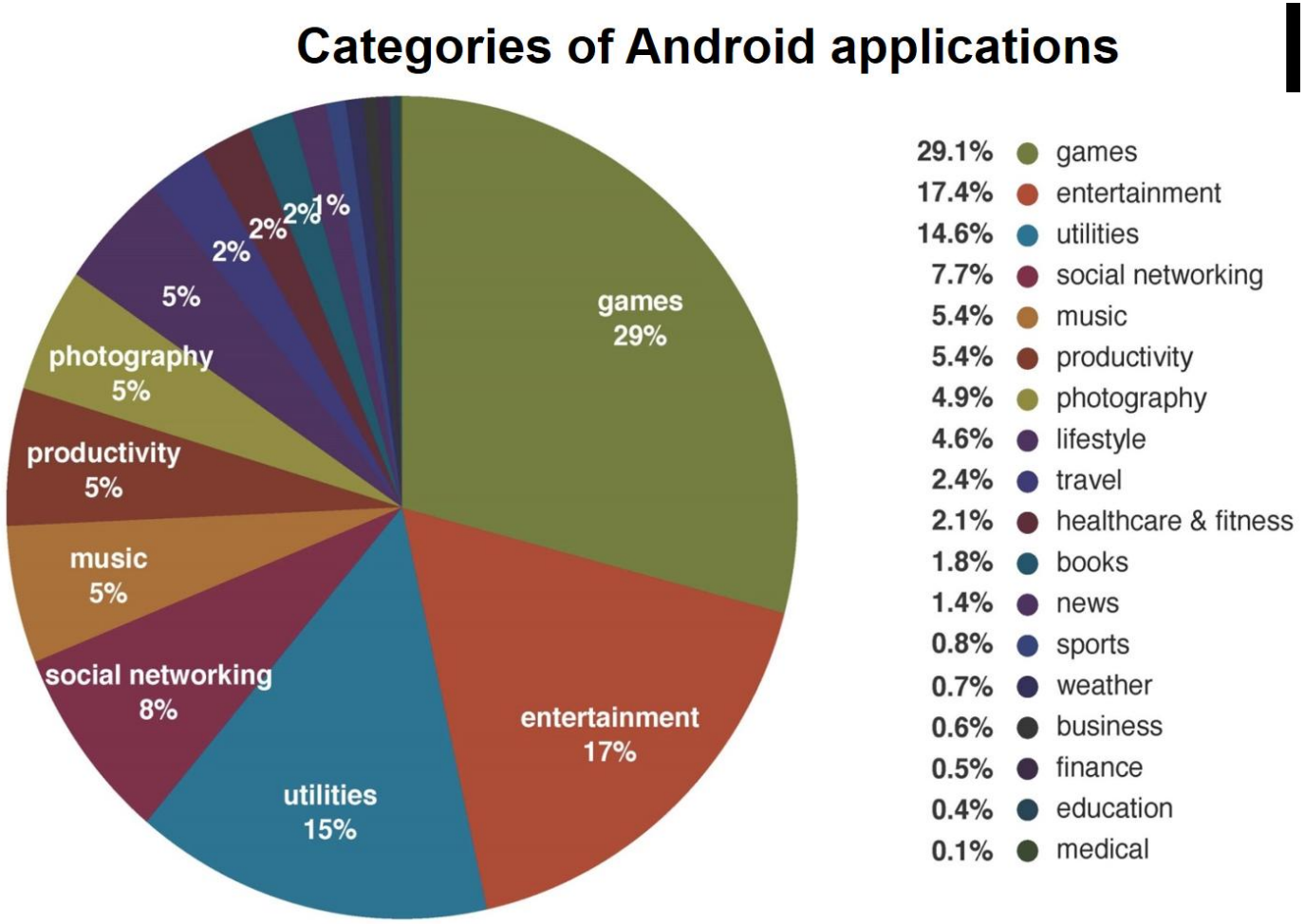
- To generate the signed APK, navigate to **Build | Generate Signed APK**. Select the app module and click on the **Next** button.
- Click on the **Create new** button to open the dialog to create a new key store. Now fill in the following information:
  - **Key store path**: This is the path in your system to create the key store. The key store is a file with the .jks extension, for example, release_ keystore.jks.
  - **Password**: This is the key store password. You have to confirm it.
  - **Alias**: This is the alias for your certificate and is a pair of public and private keys.
  - **Password**: This is the certificate password. You have to confirm it.
  - **Validity (years)**: This is the certificate that will be valid until the validity date. A value of 25 years or more is recommended.
  - **Certificate**: This is the personal information contained in the certificate.

Releasing the App

- Before you can submit your app to Google Play, you will need a developer account.
- If youdon’t have one yet, you can sign up at https://developer.android.com; then head over to the Play Console.
- Sign in with a Google account, read and agree to the developer agreement, and finally, proceed to payment.



20.7 Summary



20.8 Exercises

20.9 References: Books

1. A. Gerber.; C. Craige, *Learn Android Studio: Build Android Apps Quickly and Effectively*, Apress, **2015**. ★ ★ ★

A well characterized book covering necessary y topic of Android Studio, which may be beneficial for beginners. The language has been very simplified, which makes learning easier.

2. T. Lagos, *Learn Android Studio 3; Efficient Android App Development*, Apress, **2018.** ★ ★ ★ ★

Basic concept of the topic and the contents has been extracted from this book. Inspite of being a book about Android Studio, it isn't a book for beginners. Prior knowledge of certain language should be know in order to make the best use of this book.

3. J.F. DiMarzio, *Beginning Android Programming with Android Studio*, John Wiley & Sons, Inc., **2017.** ★ ★ ★

This book covers necessary topics in a linear manner so that one can build on their knowledge without being overwhelmed by the details. It also has numerous Try It Out sections in each chapter to assists the readers.

4. B.C.Zapata, *Android Studio Essentials*, Packt Publishing, **2015.** ★ ★ ★ ★

All topics has been very systematically and well arranged in this book, which makes its reading very pleasing. This book is highly recommended if one wants understand the basic knowledge of Android Studio.

Thank you