1. Introduction

What is Android?

Android is an open source and Linux-based **Operating System** for mobile devices such as smart phones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.

The **Android SDK** provides the tools and APIs necessary to begin applications on the Android platform using the **Java programming language**.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007, whereas the first commercial version, Android 1.0, was released in September 2008. On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 **Jelly Bean**.

The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

Features of Android

Android is a powerful operating system which supports great features. Few of them are listed below:

Features	Description
Beautiful UI	Android OS screen provides a beautiful and interactive user interface.
Connectivity	It also provides various connectivity techniques like GSM, CDMA, Bluetooth, Wi-Fi and WiMAX.
Storage	SQLite, a lightweight relational database, is used for data storage purposes.
Multi-touch	Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero.
Multi-tasking	User can jump from one task to another and same time various applications can run simultaneously.
Resizable widgets	Widgets are resizable, so users can expand them to show more content or shrink them to save space
Messaging	SMS and MMS

Android API

Name	Version Number	API Level
No name	1.0	1
Petit Four	1.1	2
Cupcake	1.5	3
Donut	1.6	3
Eclair	2.0 - 2.1	5-7
Froyo	2.2 - 2.2.3	8
Gingerbread	2.3 - 2.3.7	9-10
Honeycomb	3.0 - 3.2.6	11-13
Ice-cream Sandwich	4.0 - 4.0.4	14-15
Jelly Bean	4.1 - 4.3.1	16-18
KitKat	4.4 – 4.4.4	19
Lollipop	5.0 - 5.1.1	21-22
Marshmallow	6.0 – 6.0.1	23
Naugat	7.0-7.1.2	24-25
O	8.0	26

2. Application Components

Key Terms

1. SDK (Software Development Kit): A set of tools and libraries that allow the user to create an application based on a product.
2. IDE (Integrated Development Environment): A software application that consists of a source code, editor, a compiler, build automation tools and a debugger. It makes programming and running application easier.
3. ADT (Android Development Tools): A plugin for eclipse that extends the Eclipse IDE by providing more tools to develop Android Applications.
4. AVD (Android Virtual Device): An android emulator that allows to simulate how the application will run on an actual android device.
5. JDK (Java SE Development Kit): A popular Java SDK that is used to program Android Application.

Components

There are following four main components that can be used within an Android application:

Components	Description
Activities	They dictate the UI and handle the user interaction to the smart phone screen.
Services	They handle background processing associated with an application.
Broadcast Receivers	They handle communication between Android OS and applications

Content Providers	They handle data and database management issues.

- Activities

  An activity represents a single screen with a user interface. For example, an email application might have one activity that shows a list of new emails, another activity to compose an email, and one for reading emails. If an application has more than one activity, then one of them should be marked as the activity that is presented when the application is launched. An activity is implemented as a subclass of **Activity** class as follows:

```
public class MainActivity extends Activity
{


}
```

- Services

  A service is a component that runs in the background to perform long-running operations. For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity. A service is implemented as a subclass of **Service** class as follows:

```
public class MyService extends Service
{


}
```

- Broadcast Receivers

  Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action. A broadcast receiver is implemented as a subclass of **BroadcastReceiver** class and each message is broadcasted as an **Intent** object.

```
public class MyReceiver extends BroadcastReceiver
{


}
```

- Content Providers

  A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the *ContentResolver* class. The data may be stored in the file system, the database or somewhere else entirely. A content provider is implemented as a subclass of **ContentProvider** class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class MyContentProvider extends ContentProvider
{


}
```

Android Widgets

There are given a lot of **android widgets** such as Button, EditText, AutoCompleteTextView, ToggleButton, DatePicker, TimePicker, ProgressBar etc.

- Android Button: This one of the most used components. It can be pressed by a user and when pressed we can launch
- Android Toast: Displays information for the short duration of time.
- Custom Toast: Displays a customize the toast, such as we can display image on the toast.
- ToggleButton: It has two states ON/OFF.
- CheckBox: It displays different checkboxes from which one can select multiple options.
- AlertDialog: AlertDialog displays a alert dialog containing the message with OK and Cancel buttons.
- Spinner: Spinner displays the multiple options, but only one can be selected at a time.
- RatingBar: RatingBar displays the rating bar.
- DatePicker: Datepicker displays the datepicker dialog that can be used to pick the date.
- TimePicker: TimePicker displays the timepicker dialog that can be used to pick the time.
- ProgressBar: ProgressBar displays progress task.
- ImageView: This component is used to show an image on the screen. The image we want to show can be placed inside our apk or we can load it remotely.

ArrayAdapter

- ArrayAdapter (Context context, int resource)
- ArrayAdapter (Context context, int resource, int textViewResourceId)
- ArrayAdapter (Context context, int resource, T[] objects)
- ArrayAdapter (Context context, int resource, int textViewResourceId, T[] objects)
- ArrayAdapter (Context context, int resource, List<T> objects)
- ArrayAdapter (Context context, int resource, int textViewResourceId, List<T> objects)

Parameters	
Context	Context: The current context.
Resource	int: The resource ID for a layout file containing a TextView to use when instantiating views.
textViewResourceId	int: The id of the TextView within the layout resource to be populated
Objects	T: The objects to represent in the ListView.
Objects	List: The objects to represent in the ListView.

Layouts

When one create an app some special features are used which act as a container. These special views control how other components are placed on the smartphones screen. Android provides a collection of Layout Managers and each of them implements a different strategy to hold, manage and place its components. From the API point of view, all the Layout managers derive from the **ViewGroup class**. There are some layouts that place the components horizontally or vertically, and others that implement a different strategy.

Android provides several standard layout managers:

- Linear Layout: This is the simplest Layout manager. This layout disposes its components vertically or horizontally depending on the orientation parameter. The orientation attribute is the most important attribute because it determines how views are placed. It can assume two values that are horizontal or vertical.
- Table Layout: This is layout manager that disposes its components in a table, grouping them in rows and columns.
- Relative Layout: This is the most flexible layout in Android. This layout manager uses a policy where the container places its Views relative to other Views. We can implement, using this layout manager, very complex UI structures.
- Frame Layout: FrameLayout is a special layout that we will cover in more detail later. We saw different layout manager that implements some specific strategies to place views on the UI. FrameLayout is used when we want to display dynamically a view. It is very useful when we use Fragments.

- Grid Layout: It is very similar to TableLayout and it was introduced since Android 4.0. This layout manager disposed its views in a grid form, but respect to the TableLayout it is easier to use. GridLayout divides the screen area into cells. Its children occupy one or more cells.

3. Creating Android Application

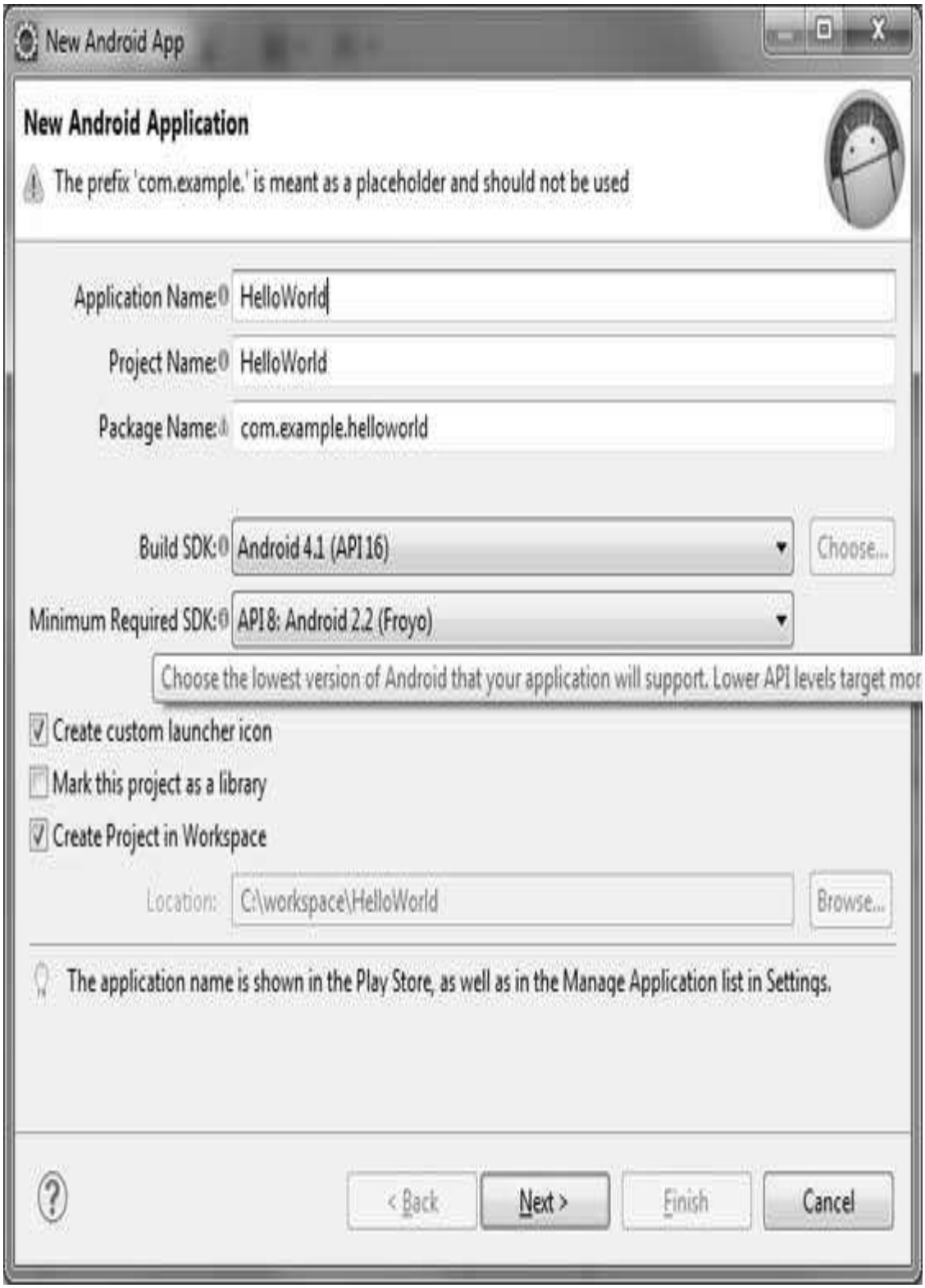
Steps to Create an Android App

a) Setup the development environment

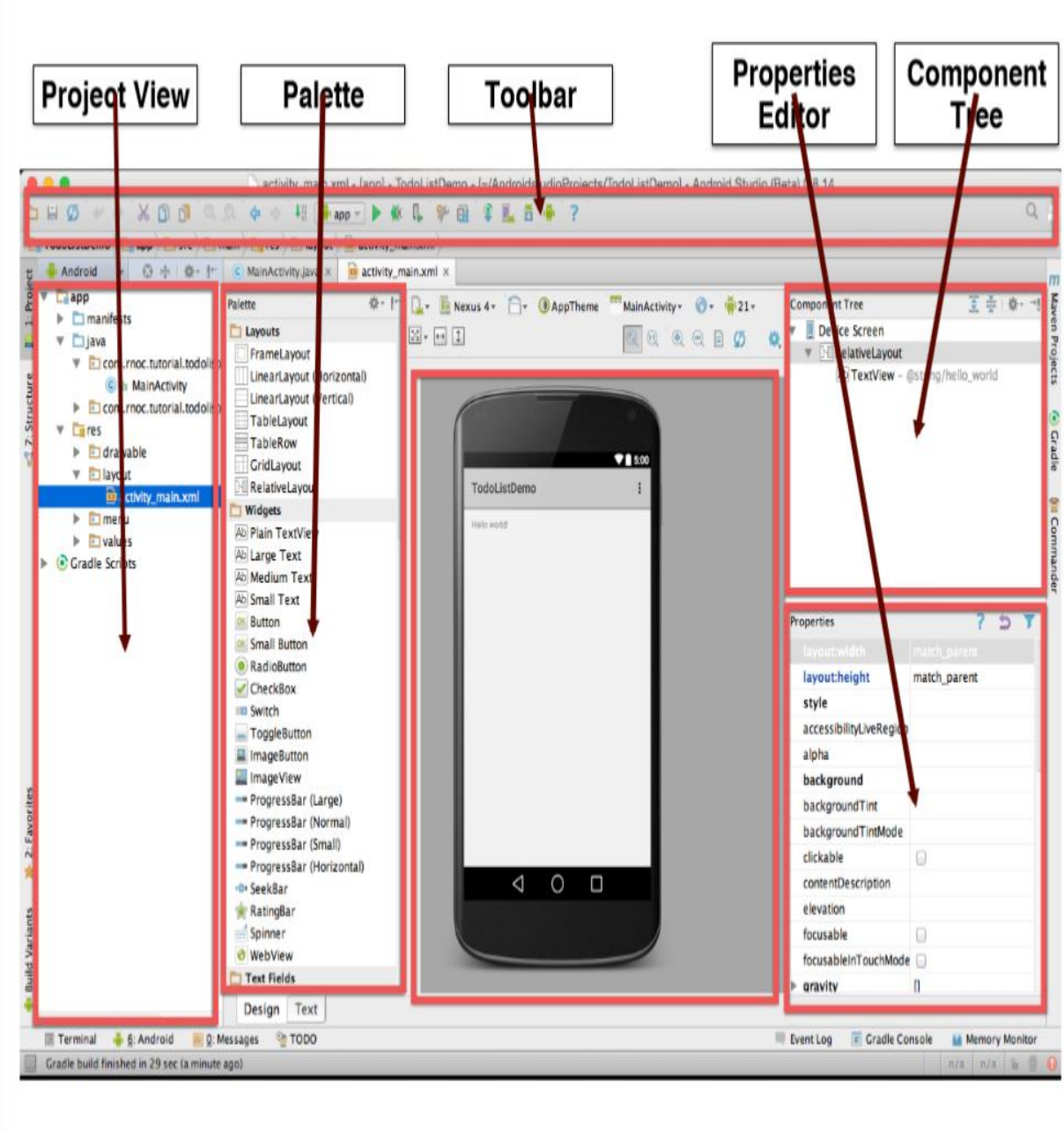
- Install Eclipse
- Install Android SDK (Android libraries)
- Install ADT plugin (Android Development Tools)
- Create AVD (Android Virtual Device)

b) Create a new Android project in Eclipse

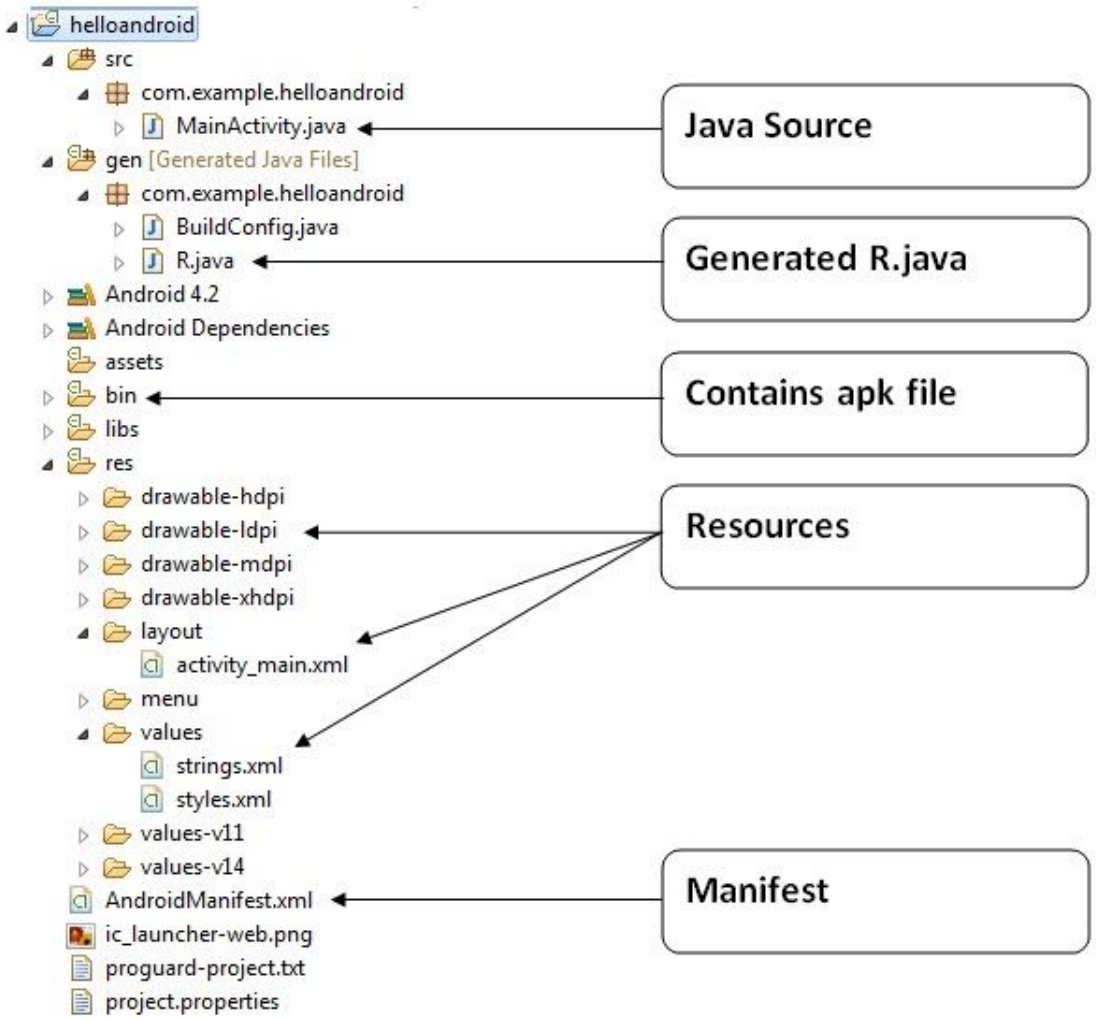
The first step is to create a simple Android Application using Eclipse IDE. Follow the option: **File -> New -> Project** and finally select **Android New Application** wizard from the wizard list. Now name the application as **HelloWorld** using the wizard window as follows:


New Android App
New Android Application
The prefix 'com.example.' is meant as a placeholder and should not be used
Application Name:
HelloWorld
Project Name:
HelloWorld
Package Name:
com.example.helloworld
Build SDK:
Android 4.1 (API 16)
Choose...
Minimum Required SDK:
API 8: Android 2.2 (Froyo)
Choose the lowest version of Android that your application will support. Lower API levels target mor
Create custom launcher icon
Mark this project as a library
Create Project in Workspace
Location:
C:\workspace\HelloWorld
Browse...
The application name is shown in the Play Store, as well as in the Manage Application list in Settings.
< Back
Next >
Finish
Cancel

Once the project is successfully created one will see the following window:



Android application contains different files such as java source code, string resources, images, manifest file, apk file etc which are shown in following figure:



- Java **Source:** This contains the **.java** source files for project. By default, it includes *MainActivity.java* source file having an activity class that runs when the app is launched using the app icon.
- **Generated R.java:** This contains the **.R** file, a compiler-generated file that references all the resources found in the project. One should not modify this file.
- **Bin:** This folder contains the Android package files **.apk** built by the ADT during the build process and everything else needed to run an Android application.

- **res/drawable-hdpi:** This is a directory for drawable objects that are designed for high density screens.
- **res/layout:** This is a directory for files that define app's user interface.
- **res/values:** This is a directory for other various XML files that contain a collection of resources, such as strings and colors definitions.
- **AndroidManifest.xml:** This is the manifest file which describes the fundamental characteristics of the app and defines each of its components.

c) Run the created project in the emulator

One has to created an AVD while doing environment setup. To run the app from Eclipse, open one of the project's activity files and click Run icon from the toolbar. Eclipse installs the app on AVD and starts it and if everything is fine with setup and application. It will show the following window: