# MOBILE APPLICATION DEVELOPMENT

Computer Engineering

6th Sem

# Unit 3

**ANDROID**

- Android is a mobile operating system that is based on a modified version of Linux.
- It was originally developed by a start-up of the same name, Android, Inc.
- In 2005, as part of its strategy to enter the mobile space, Google purchased Android and took over its development work.

# Features of Android

- Because Android is open source and freely available to manufacturers for customization, there are no fixed hardware or software configurations.

Android itself supports the following features:

- **Storage — Uses SQLite, a lightweight relational database, for data storage.**
- **Connectivity — Supports GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth (includes** A2DP and AVRCP), Wi-Fi, LTE, and WiMAX.
- **Messaging — Supports both SMS and MMS. Chapter 8 discusses messaging in more detail.**
- **Web browser — Based on the open source WebKit, together with Chrome's V8 JavaScript**
- engine
- **Media support — Includes support for the following media: H.263, H.264 (in 3GP or MP4**
- container), MPEG-4 SP, AMR, AMR-WB (in 3GP container), AAC, HE-AAC (in MP4 or
- 3GP container), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, and BMP
- **Hardware support — Accelerometer Sensor, Camera, Digital Compass, Proximity Sensor,**
- and GPS
- **Multi-touch — Supports multi-touch screens**
- **Multi-tasking — Supports multi-tasking applications**
- **Flash support — Android 2.3 supports Flash 10.1.**
- **Tethering — Supports sharing of Internet connections as a wired/wireless hotspot**

# Architecture of Android

- The Android OS is roughly divided into fi ve sections in four main layers:
- ➤ Linux kernel — This is the kernel on which Android is based. This layer contains all the low level device drivers for the various hardware components of an Android device.
- ➤ Libraries — These contain all the code that provides the main features of an Android OS. For example, the SQLite library provides database support so that an application

# Architecture of Android

**Android runtime** At the same layer as the libraries, the Android runtime provides a set of core libraries that enable developers to write Android apps using the Java programming language.

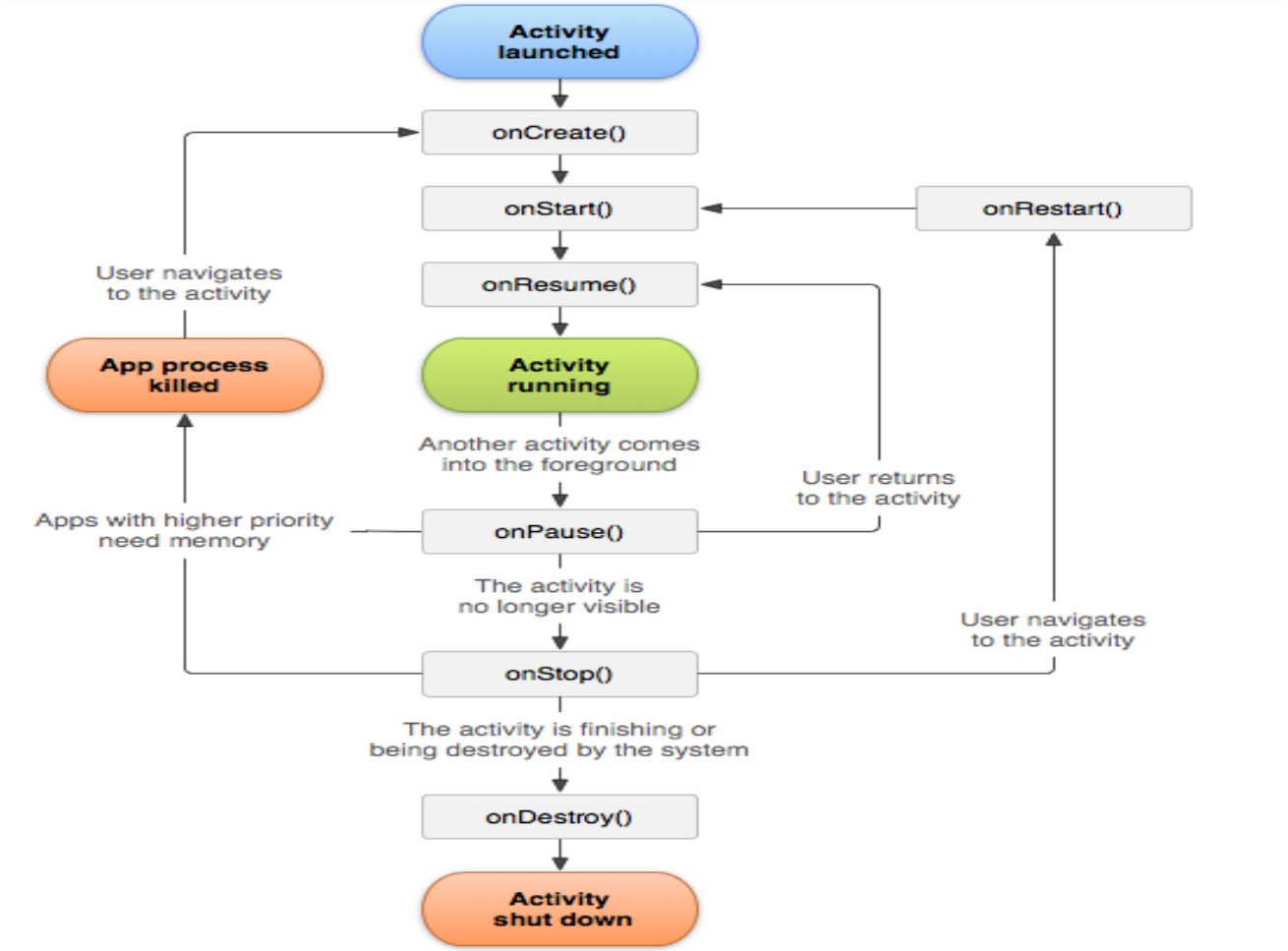
**Application framework** — Exposes the various capabilities of the Android OS to application developers so that they can make use of them in their applications.

**Applications** — At this top layer, you will find applications that ship with the Android device (such as Phone, Contacts, Browser, etc.), as well as applications that you download and install from the Android Market. Any applications that you write are located at this layer.

# Activity, Fragments

- An application can have zero or more activities. Typically, applications have one or more activities; and the main purpose of an activity is to interact with the user.
- From the moment an activity appears on the screen to the moment it is hidden, it goes through a number of stages, known as an activity's *life cycle.*
- *Understanding the life cycle of an activity is vital* to ensuring that your application works correctly.
- Fragments as "miniature" activities that can be grouped to form an activity.

# Activity Lifecycle



# COMPONENTS OF A SCREEN

- *An activity* displays the user interface of your application, which may contain widgets such as buttons, labels, textboxes, and so on.
- Typically, you define your UI using an XML fi le

**Views and ViewGroups**

- An activity contains *views and ViewGroups. A view is a widget that has an appearance on screen.*
- Examples of views are buttons, labels, and textboxes. A view derives from the base class android
- .view.View.

# Layouts

Android supports the following layouts:

- LinearLayout
- AbsoluteLayout
- TableLayout
- RelativeLayout
- FrameLayout
- ScrollView

# ADAPTING TO DISPLAY ORIENTATION

- One of the key features of modern smartphones is their ability to switch screen orientation, and
- Android is no exception. Android supports two screen orientations: *portrait and landscape.*
- *By* default, when you change the display orientation of your Android device, the current activity that is displayed automatically redraws its content in the new orientation.
- This is because the onCreate() method of the activity is fi red whenever there is a change in display orientation.

# UTILIZING THE ACTION BAR

- Besides fragments, another newer feature introduced in Android 3 and 4 is the Action Bar.
- In place of the traditional title bar located at the top of the device's screen, the Action Bar displays the application icon together with the activity title.
- Optionally, on the right side of the Action Bar are *action items.*

# LISTENING FOR UI NOTIFICATIONS

- Users interact with your UI at two levels: the activity level and the view level. At the activity level, the Activity class exposes methods that you can override.
- Some common methods that you can override in your activities include the following:
- onKeyDown — Called when a key was pressed and not handled by any of the views contained within the activity
- onKeyUp — Called when a key was released and not handled by any of the views contained within the activitY
- onMenuItemSelected — Called when a panel's menu item has been selected by the user
- onMenuOpened — Called when a panel's menu is opened by the user

# Unit 4

**USING BASIC VIEWS**

- some of the basic views that you can use to design the UI of your Android applications:
  - TextView
  - EditText
  - Button
  - ImageButton
  - CheckBox
  - ToggleButton
  - RadioButton
  - RadioGroup
- These basic views enable you to display text information, as well as perform some basic selection.

# USING PICKER VIEWS

- Selecting the date and time is one of the common tasks you need to perform in a mobile application.
- Android supports this functionality through the TimePicker and DatePicker views

**TimePicker View**

- The TimePicker view enables users to select a time of the day, in either 24-hour mode or AM/PM mode.

**DatePicker View**

- Using the DatePicker, you can enable users to select a particular date on the activity.

# LIST VIEWS

- List views are views that enable you to display a long list of items.
- there are two types of list views:
- ListView and SpinnerView.
- Both are useful for displaying long lists of items.

**ListView View**

- The ListView displays a list of items in a vertically scrolling list.

**Using the Spinner View**

- The ListView displays a long list of items in an activity, but sometimes you may want your user interface to display other views, and hence you do not have the additional space for a full-screen view like the ListView. In such cases, you should use the SpinnerView.
- The SpinnerView displays one item at a time from a list and enables users to choose among them.

# IMAGE VIEWS

## USING IMAGE VIEWS TO DISPLAY PICTURES

- So far, all the views you have seen until this point are used to display text information.
- For displaying images, you can use the ImageView, Gallery, ImageSwitcher, and GridView views.

# IMAGE VIEWS

**Gallery and ImageView Views**

- The Gallery is a view that shows items (such as images) in a center-locked, horizontal scrolling list.

**ImageSwitcher**

- The previous section demonstrated how to use the Gallery view together with an ImageView to display a series of thumbnail images so that when one is selected, it is displayed in the ImageView.
- However, sometimes you don't want an image to appear abruptly when the user selects it in the Gallery view — you might, for example, want to apply some animation to the image when it transitions from one image to another.
- In this case, you need to use the ImageSwitcher together with the Gallery view.

# GridView

- The GridView shows items in a two-dimensional scrolling grid.
- You can use the GridView together with an ImageView to display a series of images.

# Creating the Helper Methods

- Before creating options and context menus, you need to create two helper methods.
- One creates a list of items to show inside a menu
- The other handles the event that is fired when the user selects an item inside the menu.

# Context Menu

- The previous section showed how the options menu is displayed when the user presses the MENU button.
- Besides the options menu, you can also display a context menu.
- A context menu is usually associated with a view on an activity, and it is displayed when the user taps and holds an item.
- For
- example, if the user taps on a Button view and holds it for a few seconds, a context menu can be displayed.

# AnalogClock and DigitalClock Views

- AnalogClock view displays an analog clock with two hands — one for minutes and one for hours.
- DigitalClock view, displays the time digitally.
- Both views display the system time only, and do not allow you to display a particular time (such as the current time in another time zone).
- Hence, if you want to display the time for a particular region, you have to build your own custom views.

# USING MENUS WITH VIEWS

- Menus are useful for displaying additional options that are not directly visible on the main UI of an application.

There are two main types of menus in Android:

- **Options menu** — Displays information related to the current activity. In Android, you activate the options menu by pressing the MENU button.
- **Context menu —** Displays information related to a particular view on an activity. In Android, to activate a context menu you tap and hold on to it.

# SHARING DATA IN ANDROID

- In Android, using a content provider is the recommended way to share data across packages.
- Think of a content provider as a data store. How it stores its data is not relevant to the application using it; what is important is how packages can access the data stored in it using a consistent programming interface. A content provider behaves very much like a database — you can query it, edit its content, as well as add or delete content.
- However, unlike a database, a content provider can use different ways to store its data. The data can be stored in a database, in fi les, or even over a network.
- Android ships with many useful content providers, including the following:
- **➤ Browser — Stores data such as browser bookmarks, browser history,**
- **➤ CallLog — Stores data such as missed calls, call details, and so on**
  **➤ Contacts — Stores contact details**
- **➤ MediaStore — Stores media fi les such as audio, video, and images**
- **➤ Settings — Stores the device's settings and preference**

# SMS MESSAGING

- SMS messaging is one of the main A*pplications on a mobile phone today for some* users as ecessary as the phone itself.
- Any mobile phone you buy today should have at least SMS messaging capabilities, and nearly all users of any age know how to send and receive such messages.
- Android comes with a built-in SMS application that enables you to send and receive SMS messages.

# Unit 5

## Location-Based Services

- **DISPLAYING MAPS**
- Google Maps is one of the many applications bundled with the Android platform. In addition to simply using the Maps application, you can also embed it into your own applications and make it do some very cool things.
- This section describes how to use Google Maps in your Android applications and programmatically perform the following:
- Change the views of Google Maps.
- Obtain the latitude and longitude of locations in Google Maps.
- Perform geocoding and reverse geocoding (translating an address to latitude and longitude and vice versa).
- Add markers to Google Maps.

# Obtaining the Maps API Key

- Beginning with the Android SDK release v1.0, you need to apply for a free Google Maps API key before you can integrate Google Maps into your Android application. When you apply for the key
- you must also agree to Google's terms of use, so be sure to read them carefully.

# GETTING LOCATION DATA

- Nowadays, mobile devices are commonly equipped with GPS receivers. Because of the many satellites orbiting the earth, you can use a GPS receiver to find your location easily.
- However, GPS requires a clear sky to work and hence does not always work indoors or where satellites can’t penetrate (such as a tunnel through a mountain).

# MONITORING A LOCATION

- Main feature of the LocationManager class is its ability to monitor a specific location.
- This is achieved using the addProximityAlert() method.

# CONSUMING WEB SERVICES USING HTTP

- One common way to communicate with the outside world is through HTTP.
- HTTP is no stranger to most people; it is the protocol that drives much of the web's success.
- Using the HTTP protocol, you can perform a wide variety of tasks, such as downloading web pages from a web server, downloading binary data, and more.

# Downloading Binary Data

- A common task you need to perform is downloading binary data from the web.
- For example, you may want to download an image from a server so that you can display it in your application.

# Downloading Text Content

- Besides downloading binary data, you can also download plain-text content.
- For example, you might want to access a web service that returns a string of random quotes.

# Accessing Web Services

- The previous section showed how to download some plain text from a server. Very often, you need to download XML fi les and parse the contents (a good example of this is consuming web services).
- In this section you learn how to connect to a web service using the HTTP GET method.
- Once the web service returns a result in XML, you extract the relevant parts and display its content using the Toast class.