Android Structure

Himadri Parikh

Contents

- Manifest
- Resources
- Layout
- Drawable
- Activity
- Activity Lifecycle

Android Manifest

- Every app project must have an AndroidManifest.xml file (with precisely that name) at the root of the project source set. The manifest file describes essential information about your app to the Android build tools, the Android operating system, and Google Play.
- Among many other things, the **manifest file is required to declare the following**:
  - The **app's package name**, which usually matches your code's namespace.
  - The **components of the app**, which include all activities, services, broadcast receivers, and content providers.
  - The **permissions** that the app needs in order to access protected parts of the system or other apps.
  - The **hardware and software features** the app requires, which affects which devices can install the app from Google Play.
- If you're using Android Studio to build your app, the manifest file is created for you, and most of the essential manifest elements are added as you build your app.

Resources

- Resources are the additional files and static content that your code uses, such as bitmaps, layout definitions, user interface strings, animation instructions, and more.
- You should place each type of resource in a specific subdirectory of your project's res/ directory.
- The res/ directory contains all the resources (in subdirectories): an image resource, two layout resources, mipmap/ directories for launcher icons, and a string resource file.

Layouts

- A base class that manages text layout in visual elements on the screen.
- For text that will be edited, use a Dynamic Layout, which will be updated as the text changes. For text that will not change, use a Static Layout.
- Types of Layouts:
  - Linear Layout
  - Grid Layout
  - Relative Layout
  - Constraint Layout

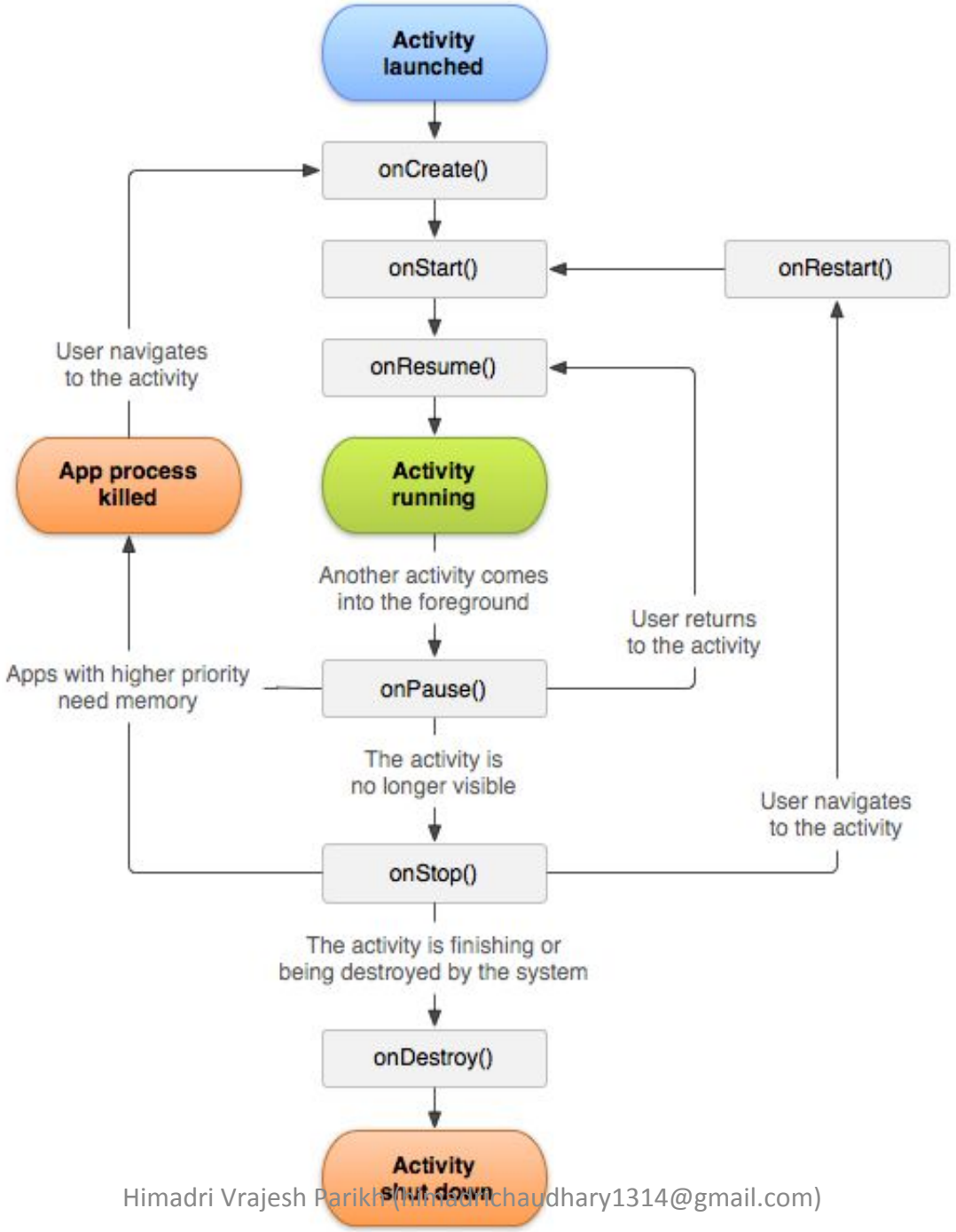
Drawable Overview

- When you need to display static images in your app, you can use the Drawable class and its subclasses to draw shapes and images.
- A Drawable is a general abstraction for *something that can be drawn*.
- There are two ways to define and instantiate a Drawable besides using the class constructors:
  - Inflate an image resource (a bitmap file) saved in your project.
  - Inflate an XML resource that defines the drawable properties.

Activities

- Activities are one of the fundamental building blocks of apps on the Android platform.
- They serve as the entry point for a user's interaction with an app,
  - and are also central to how a user navigates within an app (as with the Back button) or between apps (as with the Recents button).
- Skillfully managing activities allows you to ensure that, for example:
  - **Orientation changes** take place smoothly without disrupting the user experience.
  - **User data** is not lost during activity transitions.
  - The **system kills processes** when it's appropriate to do so.

Activity Lifecycle



Stages

- There are seven stages of activity lifecycle:
  - onCreate()
  - onRestart()
  - onResume()
  - onStart()
  - onPause()
  - onStop()
  - onDestroy()

Code


- ```
  @Override
  public void onCreate(Bundle savedInstanceState) {
      // call the super class onCreate to complete the creation of activity like
      // the view hierarchy
      super.onCreate(savedInstanceState);

      // recovering the instance state
      if (savedInstanceState != null) {
          mGameState = savedInstanceState.getString(GAME_STATE_KEY);
      }

      // set the user interface layout for this activity
      // the layout file is defined in the project res/layout/main_activity.xml file
      setContentView(R.layout.main_activity);

      Toast.makeText(getApplicationContext(), “in method onCreate”, Toast.LENGTH_LONG).show()

  }
  ```
- Same can be done on other methods of Activity
  ```


THANK YOU FOR LISTENING
ANY QUESTIONS??
makeameme.org