SERVICES

- Introduction to Services
- Threads and Process
  - Main Thread and Worker Thread
- Types of Services
  - Scheduled Service
  - Started Service, and Intent service
  - Bound Service
- Life cycle of a Service
- Application Security
- Inter-process communication
  - Messenger
  - AIDL- Android Interface Definition Language

Service Fundamentals

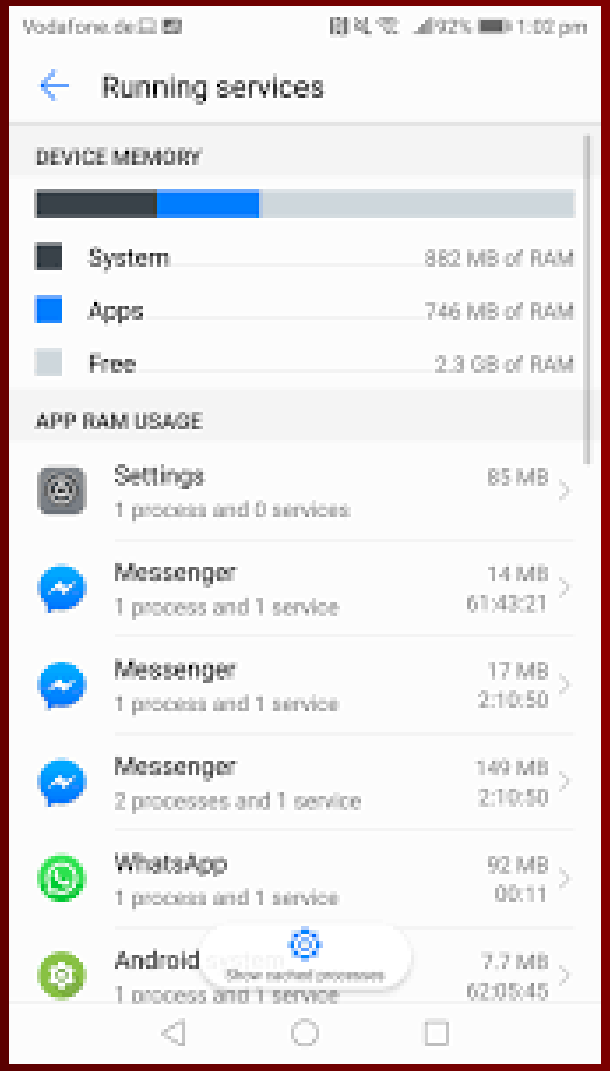
Building Blocks of Android

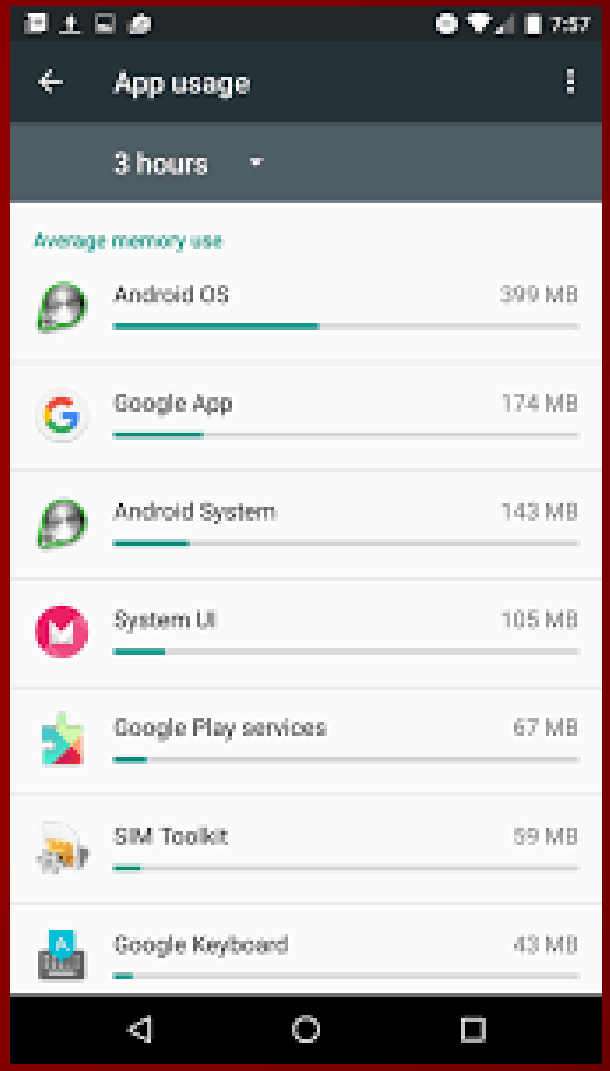
- Activity
- **Service**
- BroadcastReceiver
- ContentProvider

Services

- Visible items in the android screen is an Activity
- Background Tasks – carried by service
  - Playing Music
  - Downloading a File
  - Updating apps
  - Whatsapp, facebook apps – fetching data from servers in background
- Service is an android component that allows you to perform tasks in the background without disturbing your current interaction with the activity

Services - Example





Services - Introduction

- Application component that executes tasks in the background
- No user Interface
- By default runs in app's main thread
  - Does not run in Separate thread
  - Does not run in separate process

Types of Services

1. Scheduled Service

2. Started Service

3. Bound Service

*Bound Service provides Two way communication service

Scheduled Service

- Introduced in API 21 (Android 5.0)
  - Can’t be used for below API 21 devices
- To run or execute a task
  - Based on certain specific conditions
  - Not at specific time
- Replacement of alarm manager – as it was time specific and not consition specific
- Examples
  - Sync data only when connected with wifi
  - Start downloading the file only when the device is plugged in

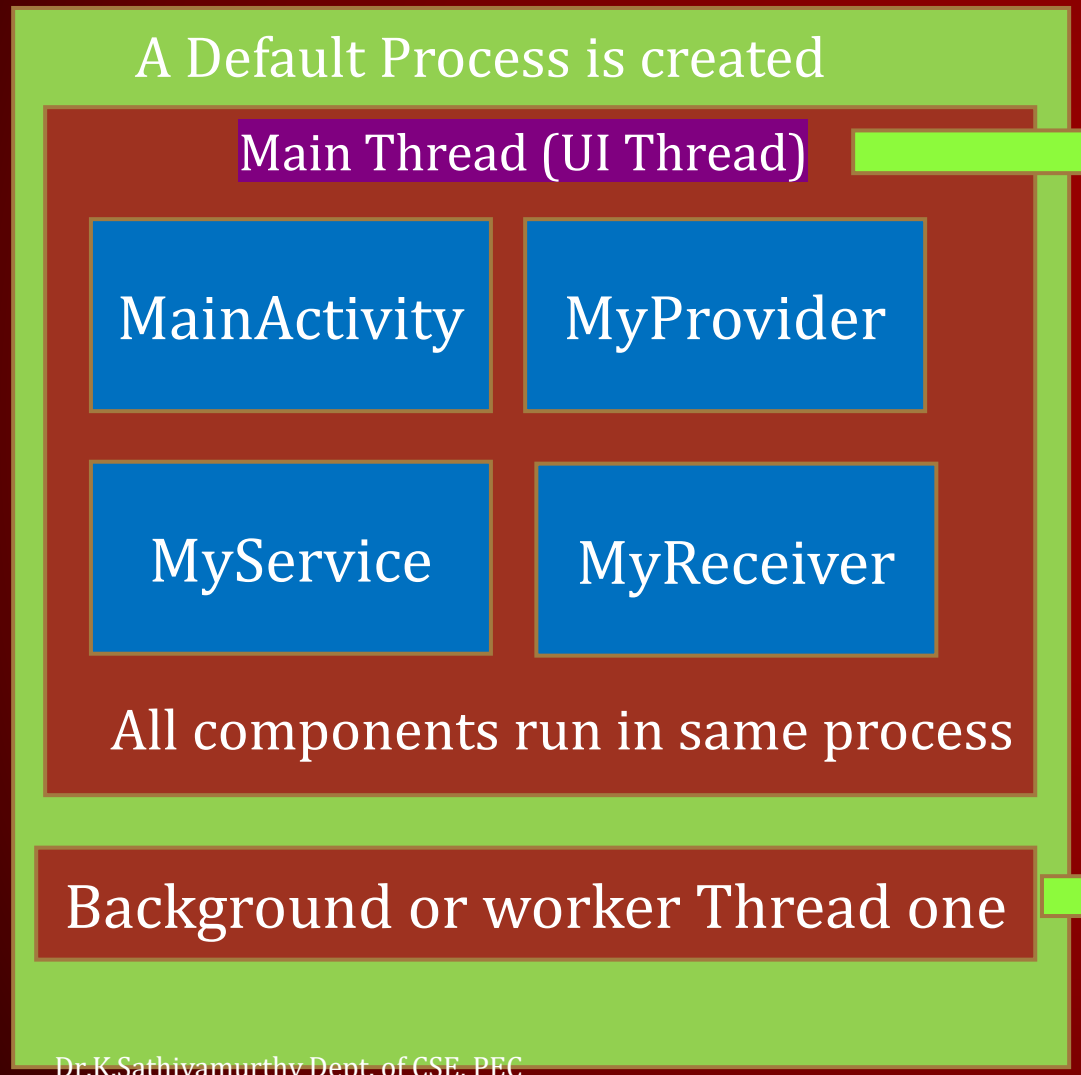
Started Service

- Started by android component – Activity, Service,BroadcastReceiver, ContentProvider
  - startService()
- When started can run in background indefinitely – even when the component that created the service is destroyed
- Performs single operation
- By default, does not return anything back to the caller
  - Use ResultReceiver, BroadcastReceiver or BoundService
- Call stopSelf() or stopService() after the task is finished

Bound Service

- Started by android component(Activity/Provider/Service)
- Android components that can bind to the service are (Activity/Provider/Service)
- Broadcast Receiver cannot bind to the service
- Method-bindService()
- Exists as long as there is atleast one component bound to it
  - When all the calling component are destroyed, the service is also destroyed
  - Service is totally dependent on the calling component
- Continuously interact with calling component
- By default has mechanism to return result back to the caller

MyAndroidApplication



Operations:
- Small Operations
- Light weight operations

Operations:
-Long Operations
-Time taking operations

MyAndroidApplication

A Default Process is created

Main Thread (UI Thread)

MainActivity

MyProvider

MyService

MyReceiver

All components run in same process

Background or worker Thread one

Process 2

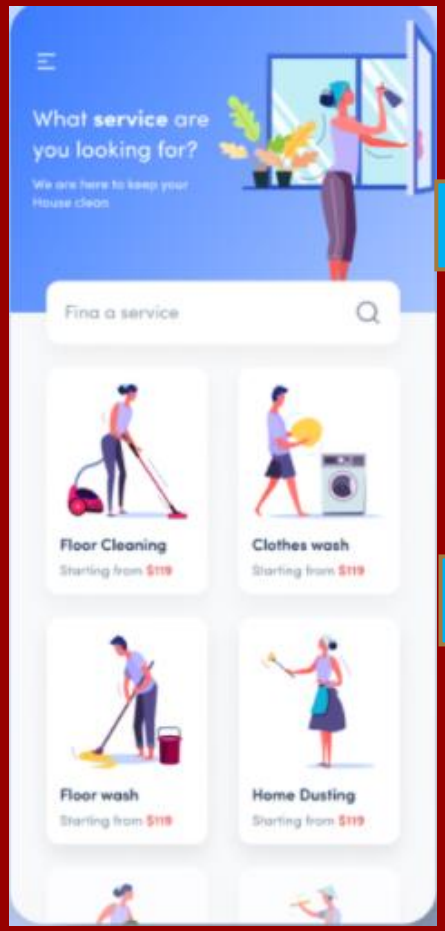
Default Main Thread

MySecondService

worker Thread

Default Process

Main UI Thread



UI Interaction

Mathematical Operations

Small Operations

Button Click

Small logical Operations

Main UI Thread

Worker or Background Thread

Long Operations

File Download

Network Operations

Image Loading

Database Queries

STARTED SERVICE

Started Service

- Runs on the main thread despite being independent of the calling component
  - May block UI if executed for long duration
- Used to perform single task that does not return anything to the calling component
- Started Service can receive data through an intent, but cannot return data back to the calling component
  - Can use Bound Service to receive
  - Implement a receiver to receive data back

Starting a Started Service

- Use start service(intent) to start service
- If the service is not running, it will be instantiated and started
  - OnCreate (to instantiate service)
  - OnStartCommand (to start service)
- StartService always runs in the main thread (in background)
- StartService is a method of context(super class of activity)
  - Use it inside the Activity class
  - Our responsibility to stop service

Stopping a Started Service

- stopService(intent)
  - A startedService will run until it receives a call to stopService(intent)
  - stopService stops a started service no matter how many times startService( ) was called earlier
  - If the service is not running calling stopService does nothing
- stopSelf()
  - Called from within the Service class itself

Steps to create a Started Service

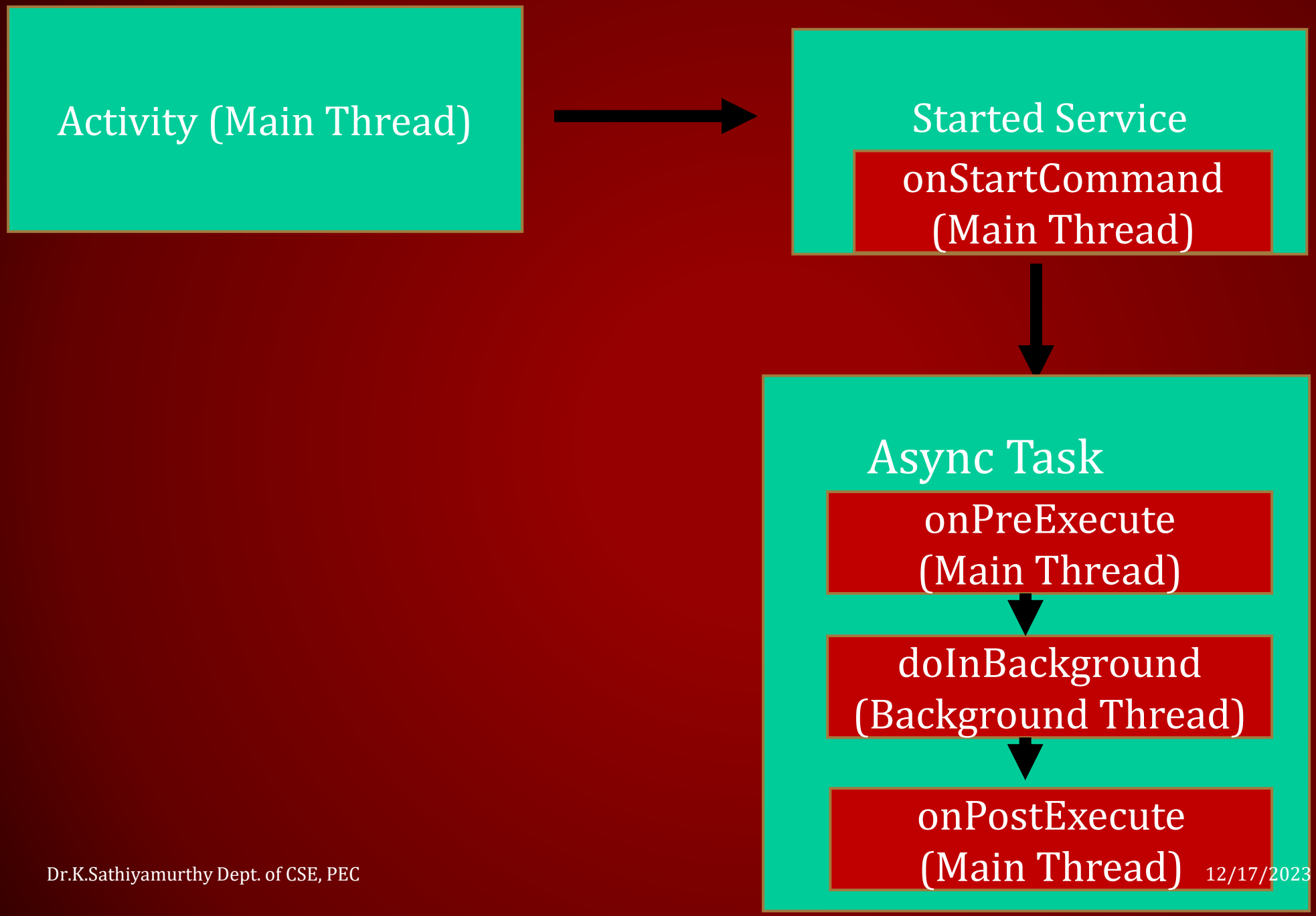
- Sub class Service – MyStartedService.java
- Override methods
  - onStartCommand
    - Should be overridden for started service
    - Once executed the service will run until either stopself(in the service) or stopService(in the calling component) is called
  - onBind
    - Always required
    - For a started service, return null
  - onCreate
    - Called if the service is not already running and needs to be created
  - onDestroy
    - Called when the service is no longer being used
- **Since service is an application component it has to be declared in AndroidManifest.xml**

Started Service- onStartCommand return Flags

- Sometimes Android system may terminate the service(may be due to lack of memory)
- START_STICKY
  - Service restarts automatically
  - Intent Lost(becomes NULL)
- START_REDELIVER_INTENT
  - Service restarts automatically
  - Intent redelivered
- START_NOT_STICKY
  - Service not restarted

Started Service- onStartCommand return Flags

- StartedService works in main thread
  - Never block the main thread as it will block the UI
  - System will generate ANR dialog
  - App will finally crash
- Solution
  - Perform the long running task in background thread


Activity (Main Thread)
Started Service
onStartCommand
(Main Thread)
Async Task
onPreExecute
(Main Thread)
doInBackground
(Background Thread)
onPostExecute
(Main Thread)