Background Processes

We will focus only on Asynchronous background tasks.

Whenever you are performing a task that takes some milliseconds, then perform it in the background. Otherwise, there will be an exception called ANR (Application Not Responding) and the application will crash. There can be four types of background tasks:

# Immediate

* Needs user interaction to proceed with the task
* API: Threads, Concurrency
* Threads cannot communicate with each other.



## Threading

### Run On UI Thread

* Provides a tunnel to pass the data between threads



### Views Post

* There is a method called post in every View which is similar to Message Passing in Operating System concepts.
* Post requires a runnable object same as thread.



### Handler

* It is normally used for communication among any number of threads.



# Exact

Time-bound task

Alarm, Meeting Records

API: Alarm Manager

# Expedited

Execute as soon as possible

Not bound to resources

Not used mostly

API: Work Manager

# Deferred

Late Execution

Will do it when resources are available

API: Work Manager

# Long Running Tasks

## Services

* Creates their own process and run in that process
* If the application gets killed, the service will not die.
* There were two types of services available: Started and Bound.
* Now we have only one type of service: **foreground service**. There’s a limitation on this that it should inform a user what task is being performed and how it is being performed. They have their own life cycles. **[Important]**

## Foreground

There are two main things to do for Foreground tasks.

* Service
* Notification

### [Generate Notification](https://developer.android.com/develop/ui/views/notifications/build-notification#kts)

There are the following steps to create a notification

* Create a Notification Manager
* Create a Notification with a small icon and its title
* Create a channel with channel id, channel name, and its importance
* Register the channel on notification manager
* Show notification via notification manager



There is a tiny problem with the above code. The notification does nothing. Let’s create a notification that when clicked opens the app. We can do this using get Activity from Pending Intent. It requires context, request code, intent, and intent flag.



# Services

Create a new Service

* Check enabled
* Uncheck Exported if you don’t want other applications to access it.







