

PendingIntent

Added in API level 1

[Kotlin](#) | [Java](#)

```
public final class PendingIntent
extends Object implements Parcelable
```

```
java.lang.Object
↳ android.app.PendingIntent
```

A description of an Intent and target action to perform with it. **Instances of this class are created with `getActivity(Context, int, Intent, int)`, `getActivities(Context, int, Intent[], int)`, `getBroadcast(Context, int, Intent, int)`, and `getService(Context, int, Intent, int)`; the returned object can be handed to other applications so that they can perform the action you described on your behalf at a later time.**

By giving a PendingIntent to another application, you are granting it the right to perform the operation you have specified as if the other application was yourself (with the same permissions and identity). As such, you should be careful about how you build the PendingIntent: almost always, for example, the base Intent you supply should have the component name explicitly set to one of your own components, to ensure it is ultimately sent there and nowhere else.

A PendingIntent itself is simply a reference to a token maintained by the system describing the original data used to retrieve it. This means that, even if its owning application's process is killed, the PendingIntent itself will remain usable from other processes that have been given it. If the creating application later re-retrieves the same kind of PendingIntent (same operation, same Intent action, data, categories, and components, and same flags), it will receive a PendingIntent representing the same token if that is still valid, and can thus call `cancel()` to remove it.

Because of this behavior, it is important to know when two Intents are considered to be the same for purposes of retrieving a PendingIntent. A common mistake people make is to create multiple PendingIntent objects with Intents that only vary in their "extra" contents, expecting to get a different PendingIntent each time. This does *not* happen. The parts of the Intent that are used for matching are the same ones defined by `Intent.filterEquals`. If you use two Intent objects that are equivalent as per `Intent.filterEquals`, then you will get the same PendingIntent for both of them.

There are two typical ways to deal with this.

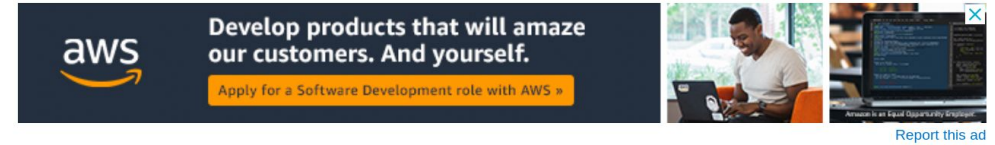
If you truly need multiple distinct PendingIntent objects active at the same time (such as to use as two notifications that are both shown at the same time), then you will need to ensure there is something that is different about them to associate them with different PendingIntents. This may be any of the Intent attributes considered by `Intent.filterEquals`, or different request code integers supplied to `getActivity(Context, int, Intent, int)`, `getActivities(Context, int, Intent[], int)`, `getBroadcast(Context, int, Intent, int)`, or `getService(Context, int, Intent, int)`.

If you only need one PendingIntent active at a time for any of the Intents you will use, then you can alternatively use the flags `FLAG_CANCEL_CURRENT` or `FLAG_UPDATE_CURRENT` to either cancel or modify whatever current PendingIntent is associated with the Intent you are supplying.

Also note that flags like `FLAG_ONE_SHOT` or `FLAG_IMMUTABLE` describe the PendingIntent instance and thus, are used to identify it. Any calls to retrieve or modify a PendingIntent created with these flags will also require these flags to be supplied in conjunction with others. E.g. To retrieve an existing PendingIntent created with `FLAG_ONE_SHOT`, **both** `FLAG_ONE_SHOT` and `FLAG_NO_CREATE` need to be supplied.

Show android notification action buttons expanded by default

Asked 4 years, 4 months ago Modified 3 years, 8 months ago Viewed 7k times

[Report this ad](#)

Is there a way to show action buttons in notification expanded by default? I use ongoing notification to control training process in my app. I want controlling buttons such as "Stop" and "Pause" to be visible right after notification appeared in the notification area.

8



3

[android](#) [notifications](#) [android-notifications](#) [notification-action](#)

Share Follow

asked Nov 16, 2017 at 8:11

**Oleksandr Albul**
1,446 ● 20 ● 30

Please go through to <https://stackoverflow.com/a/23331716/5308778> – [yashkal](#) Nov 16, 2017 at 8:20

@LakshayJuneja thanks for quick reply. But I do not use `BigTextStyle`, I just want my action buttons was visible. Do those rules apply for buttons as well? – [Oleksandr Albul](#) Nov 16, 2017 at 8:31


@ Oleksandr Albul... **yes because you can only set priority to your notifications as `setPriority(NotificationCompat.PRIORITY_HIGH)` or any other.** To show notification as you expected is OS dependent. – [yashkal](#) Nov 16, 2017 at 10:15

@LakshayJuneja I've already done `setPriority(NotificationCompat.PRIORITY_HIGH)`, but notification appears in the top and still has buttons collapsed. – [Oleksandr Albul](#) Nov 16, 2017 at 14:46

1 sorry for the such delay, as I have already mentioned above you couldn't force OS to show notification as expanded. – [yashkal](#) Nov 20, 2017 at 5:02

Add a comment

1 Answer

Sorted by: Highest score (default) 

You can't expand notification. The only solution is, set Priority Max, then the top of the notification list where it would be expanded. And it depends on the device as well.

8



```
mBuilder.setPriority(Notification.PRIORITY_HIGH)
```

Share Follow

answered Jul 25, 2018 at 8:37

**Anjal Saneen**
2,888 ● 21 ● 35