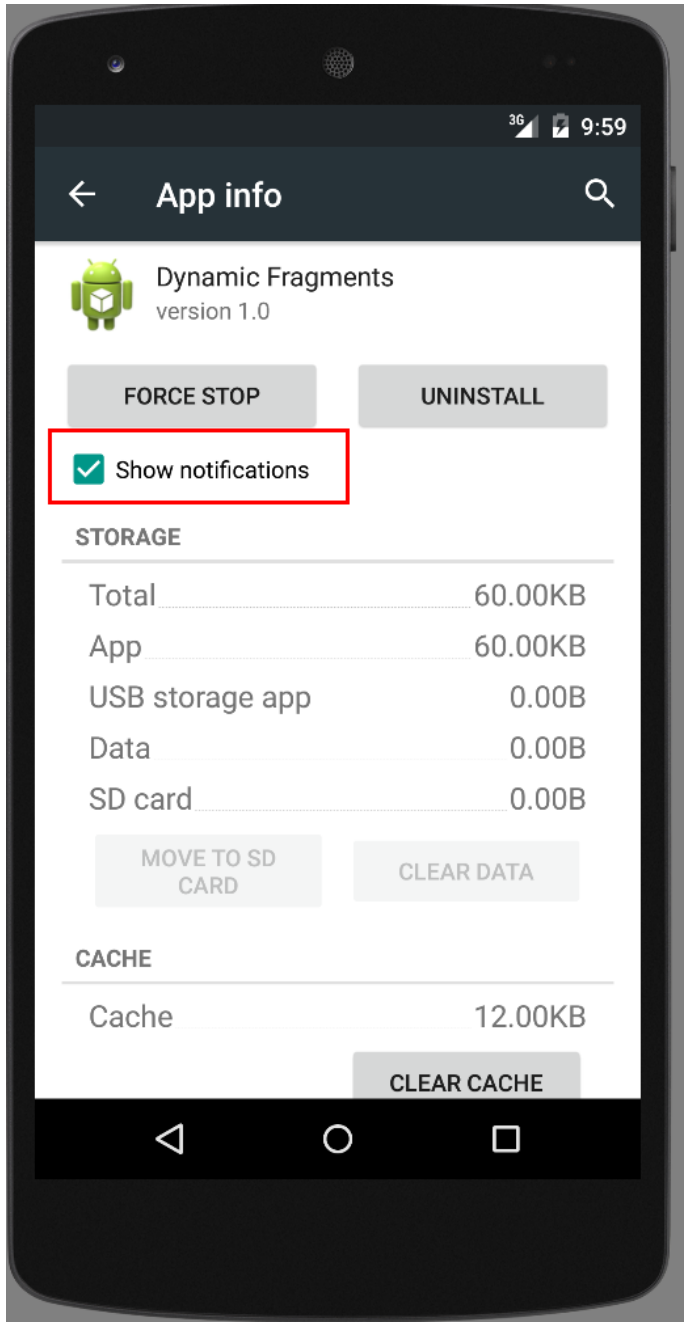
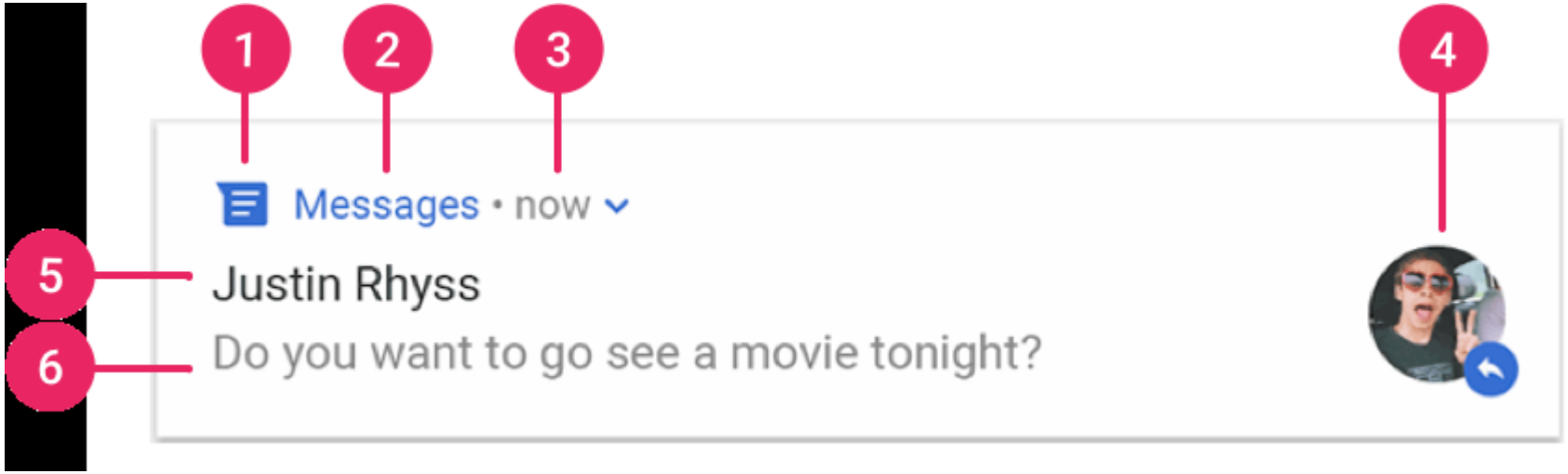
# Notifications

# Notifications

- A **notification** is a message you can display to the user outside of your application's normal UI. When you tell the system to issue a notification, it first appears as an icon in the notification area. To see the details of the notification, the user opens the notification drawer.
- Both the notification area and the notification drawer are system-controlled areas that the user can view at any time.
- Android Toast class provides a handy way to show users alerts but problem is that these alerts are not persistent which means alert flashes on the screen for a few seconds and then disappears.
- Because notifications can be very annoying, the user can disable notifications for each application. This can be done via the Settings application of the Android device


9:59
App info
Dynamic Fragments
version 1.0
FORCE STOP
UNINSTALL
Show notifications
STORAGE
Total 60.00KB
App 60.00KB
USB storage app 0.00B
Data 0.00B
SD card 0.00B
MOVE TO SD CARD
CLEAR DATA
CACHE
Cache 12.00KB
CLEAR CACHE




1. Small icon: This is required and set with setSmallIcon().
2. App name: This is provided by the system.
3. Time stamp: This is provided by the system but you can override with setWhen() or hide it with setShowWhen(false).
4. Large icon: This is optional (usually used only for contact photos; do not use it for your app icon) and set with setLargeIcon().
5. Title: This is optional and set with setContentTitle().
6. Text: This is optional and set with setContentText().

# Notification Properties

The properties of Android notification are set using **NotificationCompat.Builder** object

- setSmallIcon(): It sets the icon of notification.
- setContentTitle(): It is used to set the title of notification.
- setContentText(): It is used to set the text message.
- setAutoCancel(): It sets the cancelable property of notification.
- setPriority(): It sets the priority of notification.

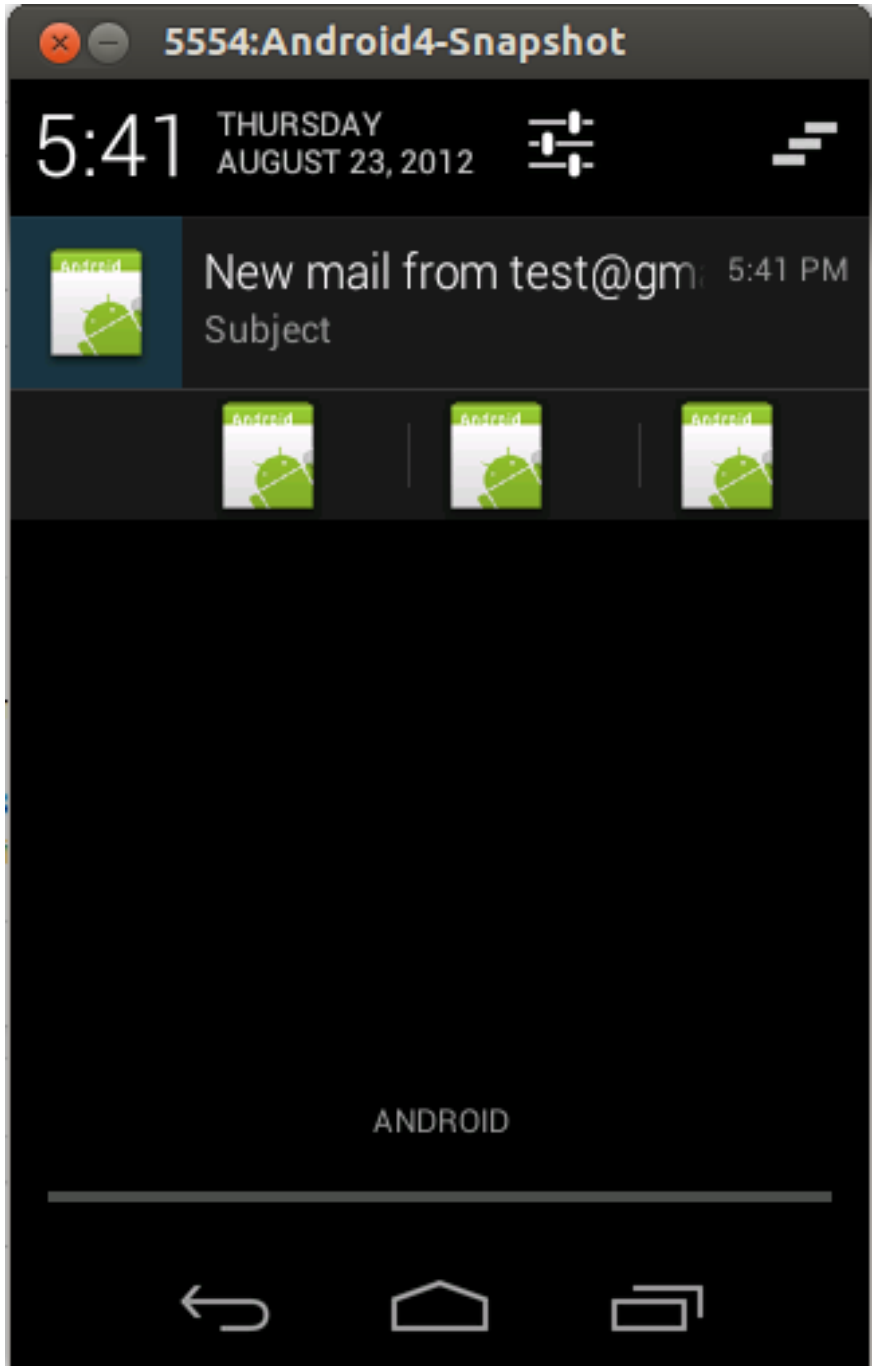
- NotificationManager notificationManager = (NotificationManager)getSystemService(NOTIFICATION_SERVICE);
- The **Notification.Builder** provides a builder interface to create an Notification object. You use a **PendingIntent** to specify the action which should be performed once the user select the notification.
- Finally, you pass the Notification object to the system by calling **NotificationManager.notify()** to send your notification. Make sure you call **NotificationCompat.Builder.build()** method on builder object before notifying it. This method combines all of the options that have been set and return a new Notification object.

```
Intent intent = new Intent(this, NotificationReceiver.class);
// use System.currentTimeMillis() to have a unique ID for the pending
intent
PendingIntent pIntent = PendingIntent.getActivity(this, (int)
System.currentTimeMillis(), intent, 0);

Notification n  = new Notification.Builder(this)
                .setContentTitle("New mail from " + "test@gmail.com")
                .setContentText("Subject")
                .setSmallIcon(R.drawable.icon)
                .setContentIntent(pIntent)
                .setAutoCancel(true)
                .addAction(R.drawable.icon, "Call", pIntent)
                .addAction(R.drawable.icon, "More", pIntent)
                .addAction(R.drawable.icon, "And more", pIntent).build();


NotificationManager notificationManager =
    (NotificationManager) getSystemService(NOTIFICATION_SERVICE);

notificationManager.notify(0, n);
```


5554:Android4-Snapshot
5:41
THURSDAY
AUGUST 23, 2012
New mail from test@gm
5:41 PM
Subject
ANDROID

Notice that the NotificationCompat.Builder constructor requires that you provide a channel ID. This is required for compatibility with Android 8.0 (API level 26) and higher, but is ignored by older versions.
By default, the notification's text content is truncated to fit one line. If you want your notification to be longer, you can enable an expandable notification by adding a style template with setStyle(). For example, the following code creates a larger text area:

```
NotificationCompat.Builder builder = new
NotificationCompat.Builder(this, CHANNEL_ID)
        .setSmallIcon(R.drawable.notification_icon)
        .setContentTitle("My notification")
        .setContentText("Much longer text that cannot fit one
line...")
        .setStyle(new NotificationCompat.BigTextStyle()
                .bigText("Much longer text that cannot fit one
line..."))
        .setPriority(NotificationCompat.PRIORITY_DEFAULT);
```

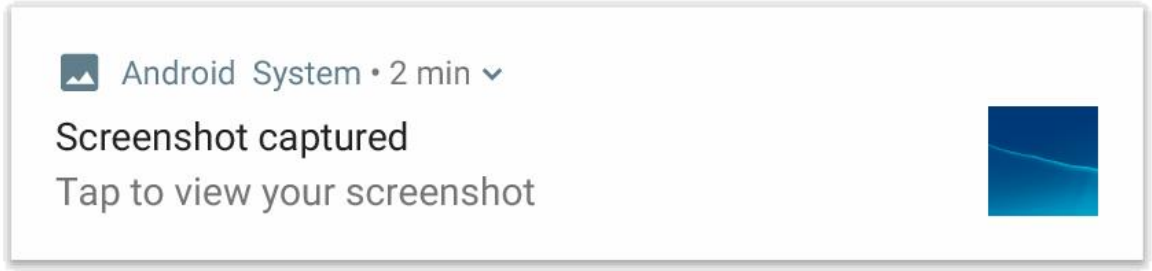
To add an image in your notification, pass an instance of NotificationCompat.BigPictureStyle to setStyle().

```
Notification notification = new
NotificationCompat.Builder(context, CHANNEL_ID)
        .setSmallIcon(R.drawable.new_post)
        .setContentTitle(imageTitle)
        .setContentText(imageDescription)
        .setStyle(new NotificationCompat.BigPictureStyle()
                .bigPicture(myBitmap))
        .build();
```

To make the image appear as a thumbnail only while the notification is collapsed, call setLargeIcon() and pass it the image, but also call BigPictureStyle.bigLargeIcon() and pass it null so the large icon goes away when the notification is expanded:

```
Notification notification = new
NotificationCompat.Builder(context, CHANNEL_ID)
        .setSmallIcon(R.drawable.new_post)
        .setContentTitle(imageTitle)
        .setContentText(imageDescription)
        .setLargeIcon(myBitmap)
        .setStyle(new NotificationCompat.BigPictureStyle()
                .bigPicture(myBitmap)
                .bigLargeIcon(null))
        .build();
```

Collapsed




Expanded