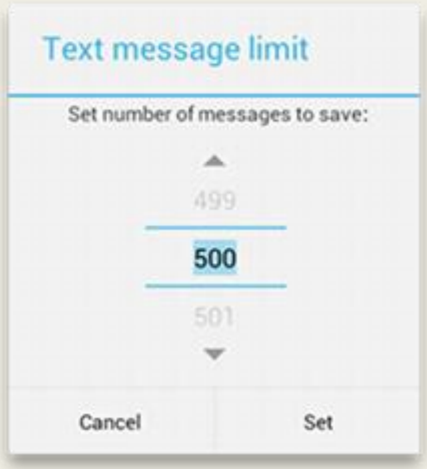
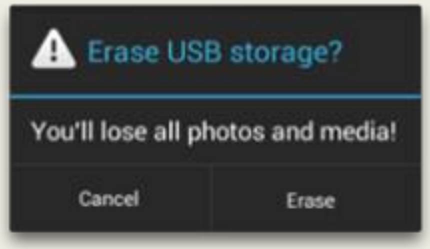
ANDROID
DIALOGS AND NOTIFICATIONS

Efthimios Alepis

Dialogs

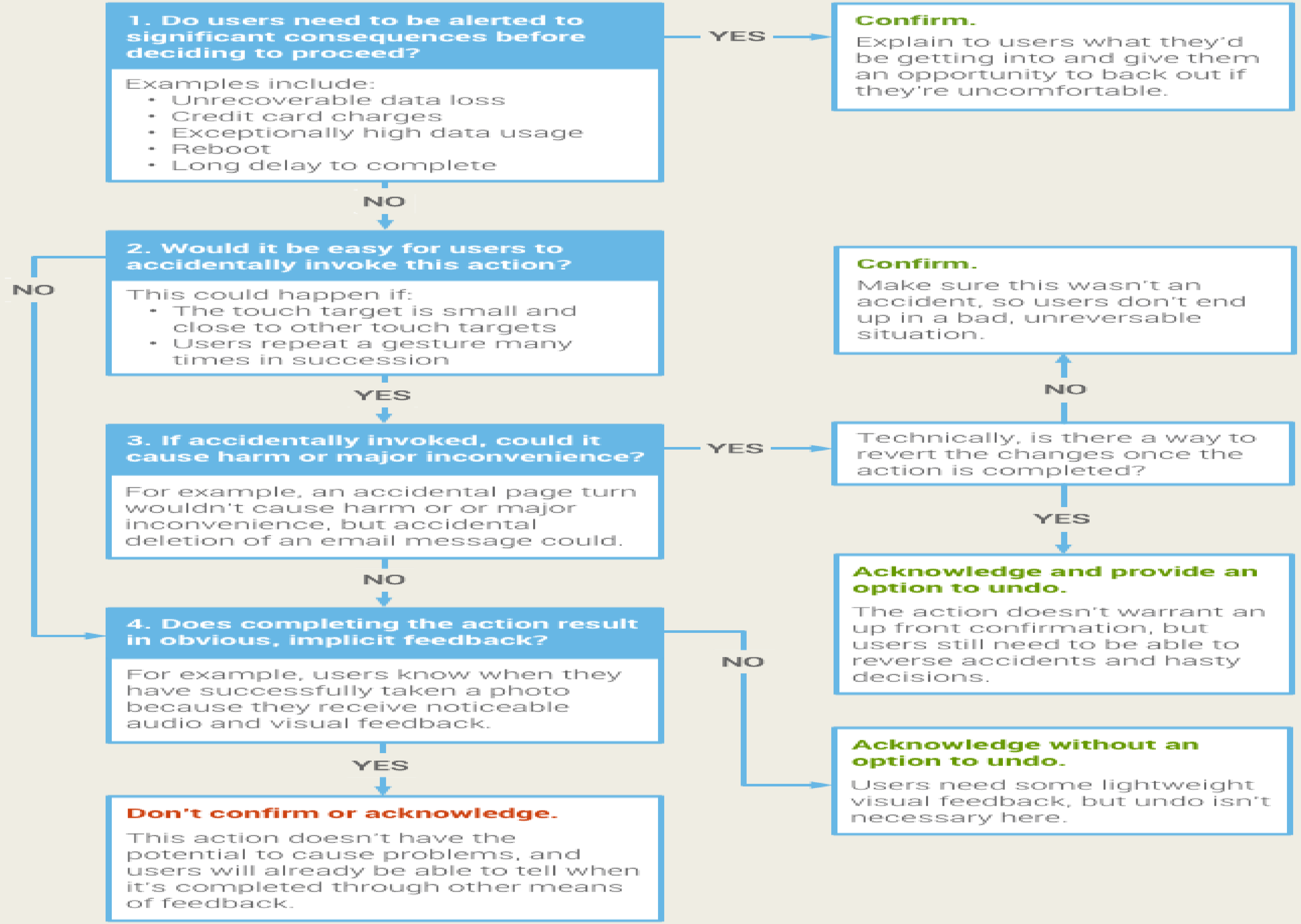
- A dialog is a small window that prompts the user to make a decision or enter additional information
- A dialog does not fill the screen and is normally used for modal events that require users to take an action before they can proceed
- Dialogs contain text and UI controls. They retain focus until dismissed or a required action has been taken. Use dialogs sparingly because they are interruptive.





Confirming vs Acknowledging

- In some situations, when a user invokes an action in your app, it's a good idea to *confirm* or *acknowledge* that action through text
- **Confirming** is asking the user to verify that they truly want to proceed with an action they just invoked
- **Acknowledging** is displaying text to let the user know that the action they just invoked has been completed
- Confirming -> Dialogs
- Acknowledging -> Toasts



1. Do users need to be alerted to significant consequences before deciding to proceed?

Examples include:
- Unrecoverable data loss
- Credit card charges
- Exceptionally high data usage
- Reboot
- Long delay to complete

Confirm.
Explain to users what they'd be getting into and give them an opportunity to back out if they're uncomfortable.

2. Would it be easy for users to accidentally invoke this action?

This could happen if:
- The touch target is small and close to other touch targets
- Users repeat a gesture many times in succession

Confirm.
Make sure this wasn't an accident, so users don't end up in a bad, unreversable situation.

3. If accidentally invoked, could it cause harm or major inconvenience?

For example, an accidental page turn wouldn't cause harm or or major inconvenience, but accidental deletion of an email message could.

Technically, is there a way to revert the changes once the action is completed?

Acknowledge and provide an option to undo.
The action doesn't warrant an up front confirmation, but users still need to be able to reverse accidents and hasty decisions.

4. Does completing the action result in obvious, implicit feedback?

For example, users know when they have successfully taken a photo because they receive noticeable audio and visual feedback.

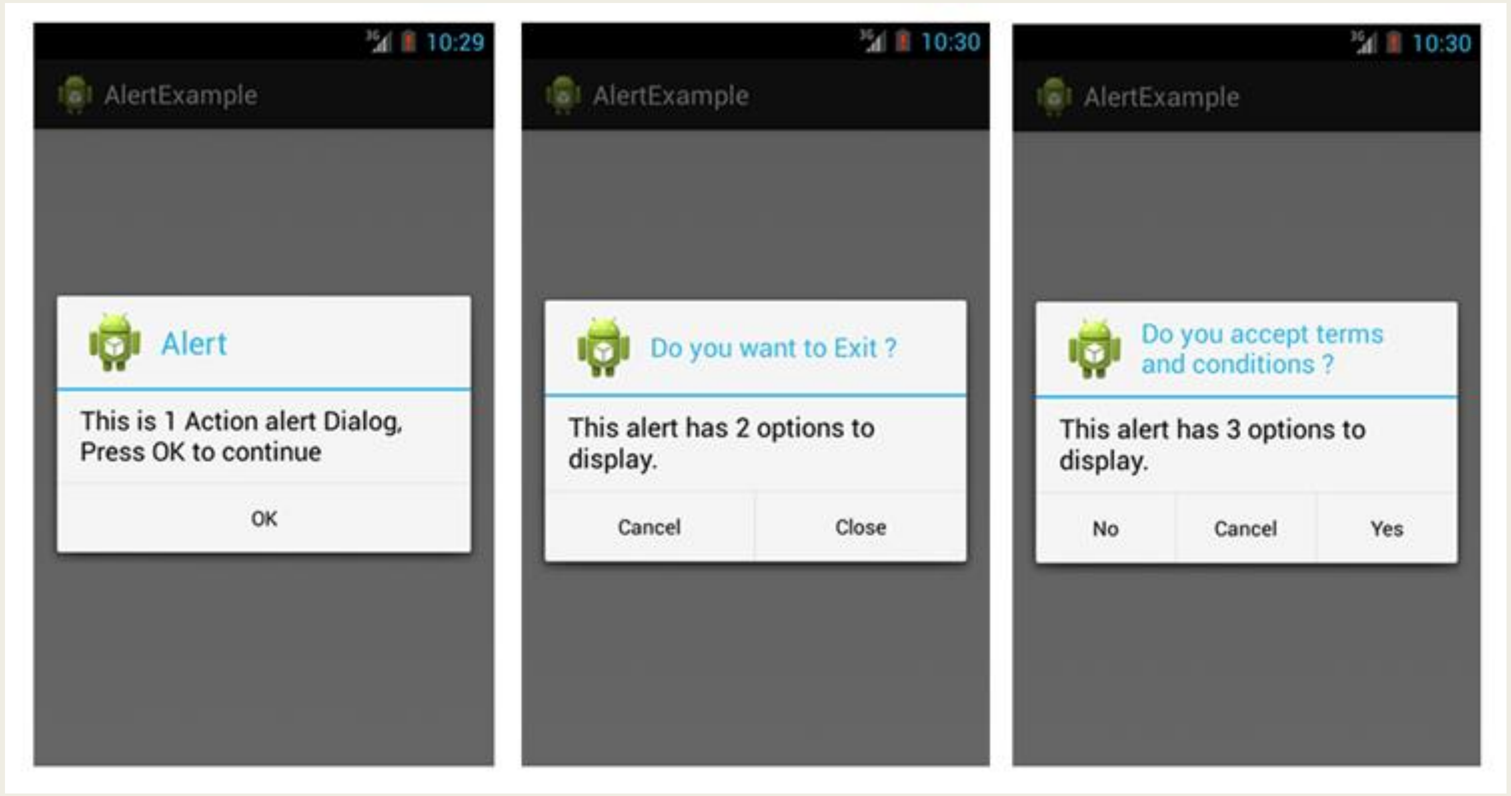
Acknowledge without an option to undo.
Users need some lightweight visual feedback, but undo isn't necessary here.

Don't confirm or acknowledge.
This action doesn't have the potential to cause problems, and users will already be able to tell when it's completed through other means of feedback.

Dialog Class

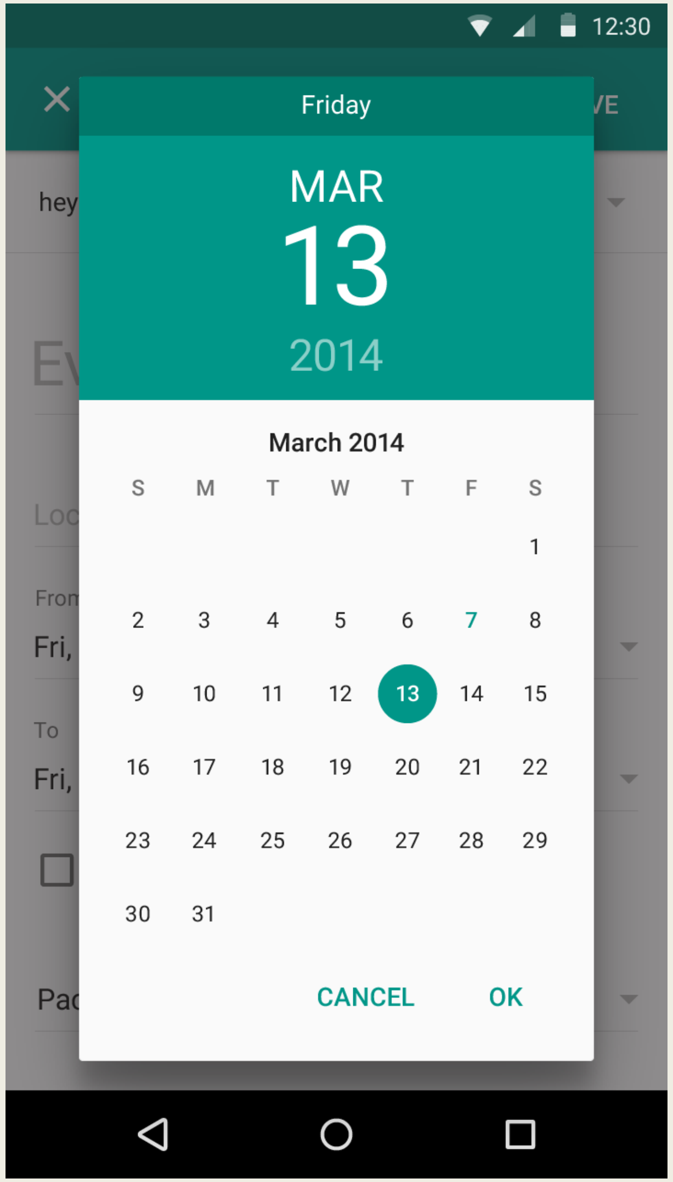
- The Dialog class is the base class for dialogs, but you should avoid instantiating Dialog directly. Instead, use one of the following subclasses:
- Direct Subclasses
  - *AlertDialog, AppCompatDialog, CharacterPickerDialog, Presentation*
- Indirect Subclasses
  - *AlertDialog, BottomSheetDialog, DatePickerDialog, MediaRouteChooserDialog, MediaRouteControllerDialog, ProgressDialog, TimePickerDialog*

AlertDialog

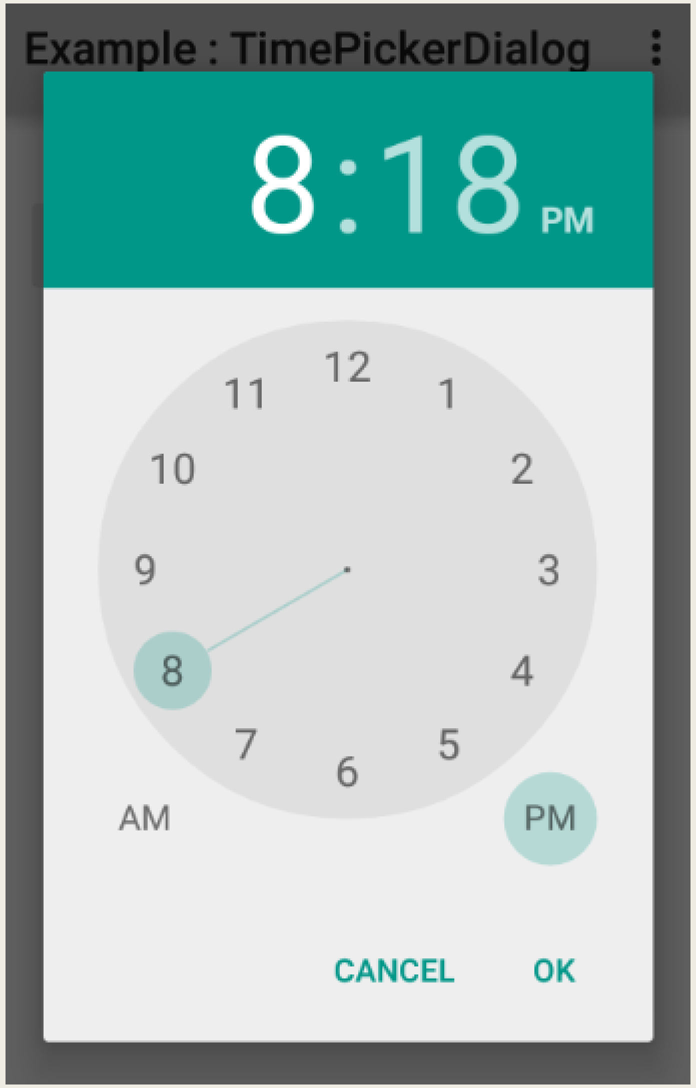

10:29
AlertExample
Alert
This is 1 Action alert Dialog, Press OK to continue
OK
10:30
AlertExample
Do you want to Exit ?
This alert has 2 options to display.
Cancel
Close
10:30
AlertExample
Do you accept terms and conditions ?
This alert has 3 options to display.
No
Cancel
Yes

Alerts are urgent interruptions, requiring acknowledgement, that inform the user about a situation.

DatePickerDialog



TimePickerDialog

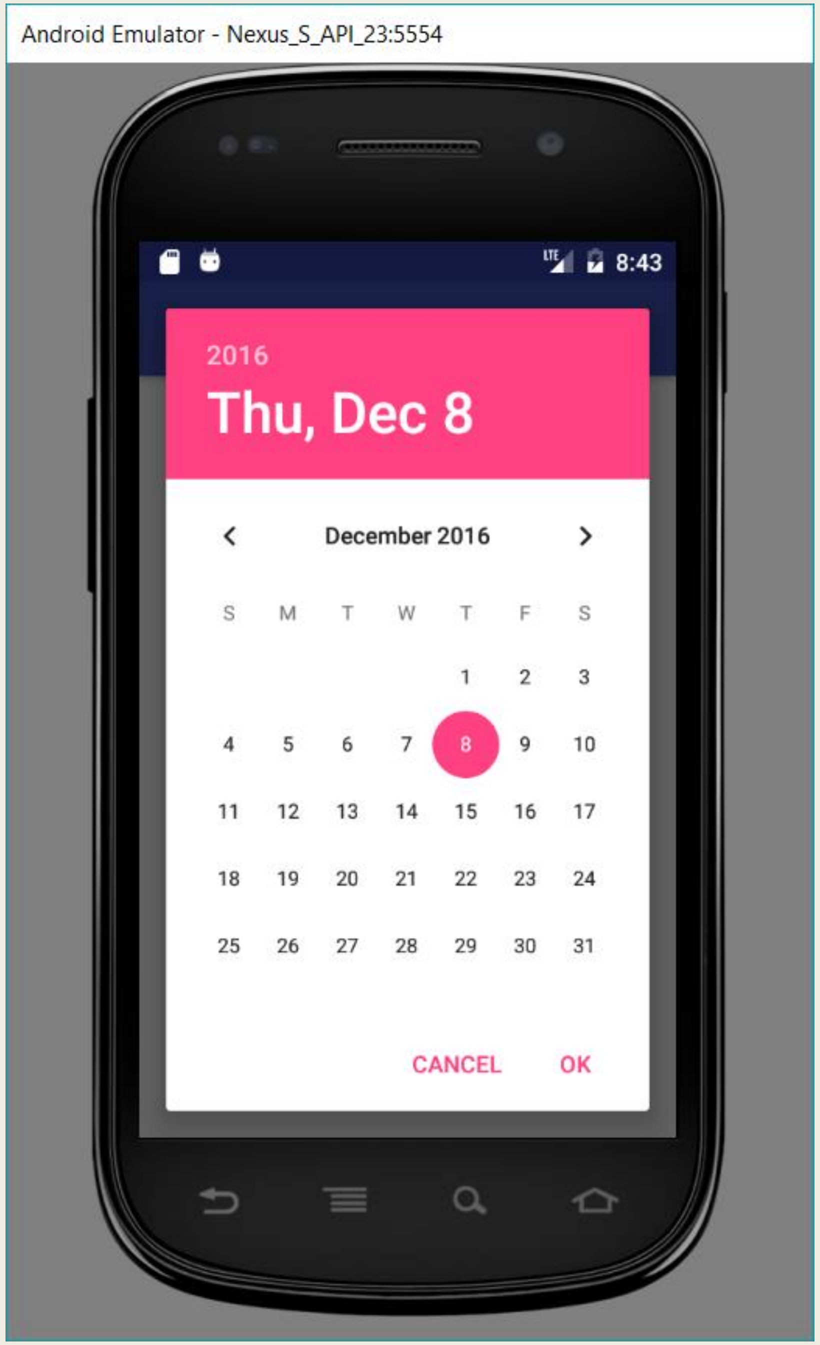


Creating a Date Picker

- Creating a DatePickerDialog is just like creating a TimePickerDialog. The only difference is the dialog you create for the fragment.

1. Extend DialogFragment for a date picker
2. Define the onCreateDialog() method to return an instance of DatePickerDialog
3. Implement the DatePickerDialog.OnDateSetListener interface to receive a callback when the user sets the date
4. Once you've defined a DialogFragment, you can display the date picker by creating an instance of the DialogFragment and calling show()

Example


Android Emulator - Nexus_S_API_23:5554
8:43
2016
Thu, Dec 8
December 2016
S M T W T F S
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
CANCEL
OK

The Alert Dialog class

- The AlertDialog class allows you to build a variety of dialog designs and is often the only dialog class you'll need
- There are three regions of an alert dialog:
  - *Title. This is optional and should be used only when the content area is occupied by a detailed message, a list, or custom layout.*
  - *Content area. This can display a message, a list, or other custom layout.*
  - *Action buttons. There should be no more than three action buttons in a dialog.*

Creating an Alert Dialog

1. Instantiate an AlertDialog.Builder with its constructor

   ```
   AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
   ```

2. Chain together various setter methods to set the dialog characteristics

   ```
   builder.setMessage(R.string.dialog_message).setTitle(R.string.dialog_title);
   ```

3. Get the AlertDialog from create()

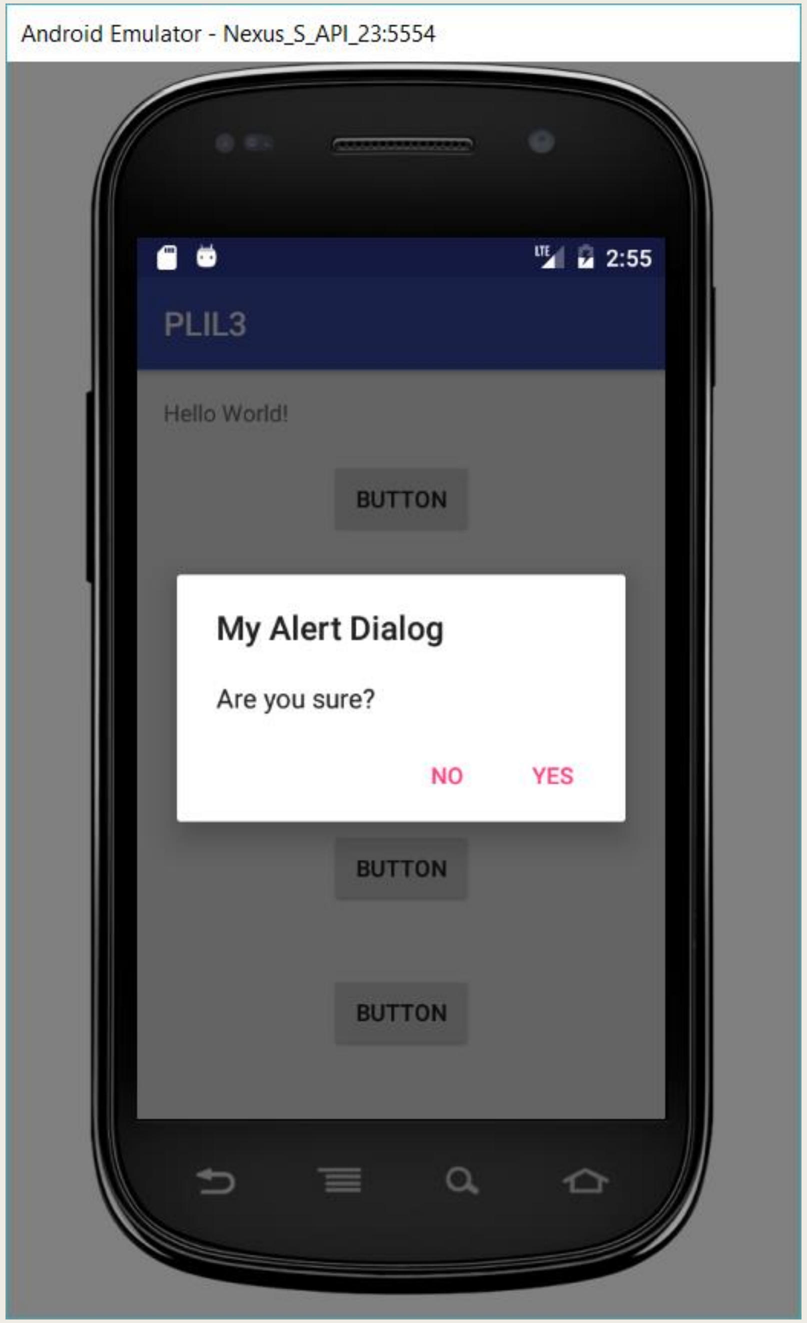
   ```
   AlertDialog dialog = builder.create();
   ```

4. Show the created dialog.

   ```
   dialog.show();
   ```

5. Add Buttons? -> Through builder object. Options: “Positive”, “Negative” and “Neutral” buttons

Example

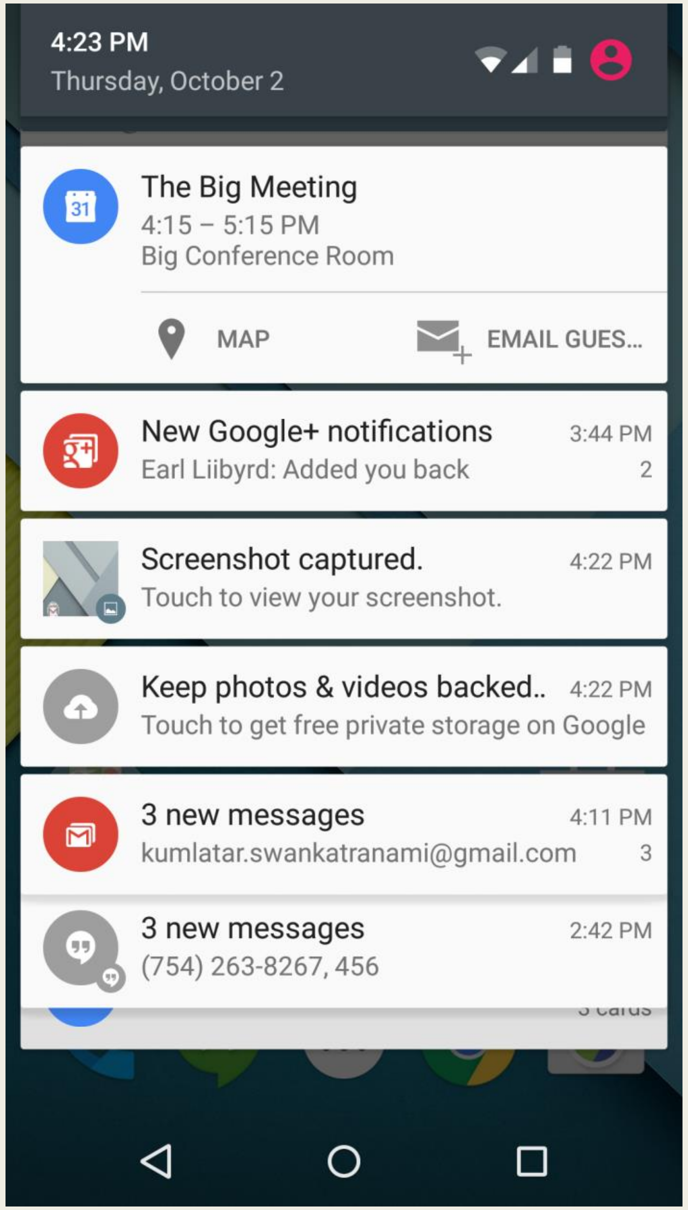

Android Emulator - Nexus_S_API_23:5554

My Alert Dialog

Are you sure?

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



4:23 PM
Thursday, October 2

The Big Meeting
4:15 – 5:15 PM
Big Conference Room
MAP
EMAIL GUES...

New Google+ notifications 3:44 PM
Earl Liibyrd: Added you back 2

Screenshot captured. 4:22 PM
Touch to view your screenshot.

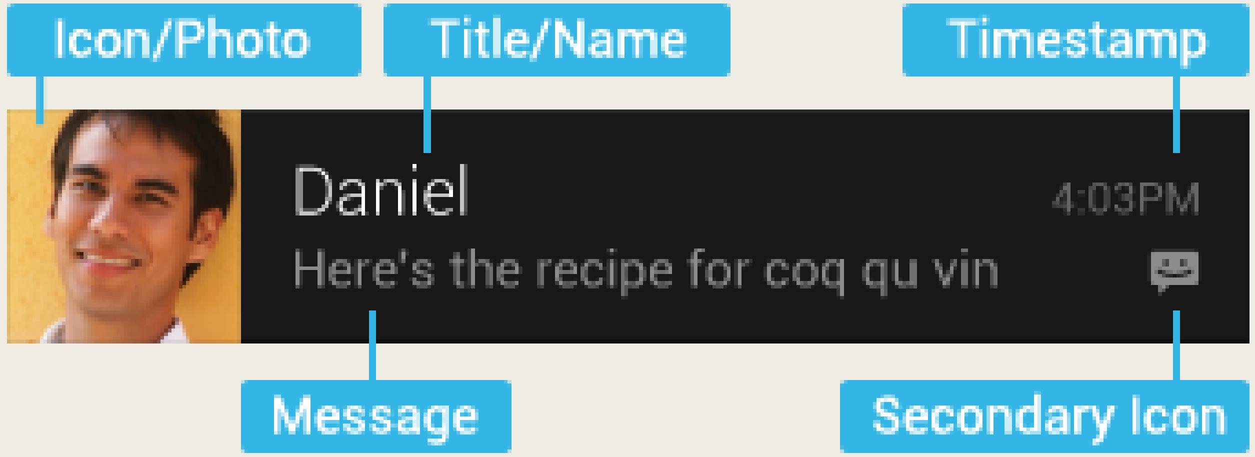
Keep photos & videos backed.. 4:22 PM
Touch to get free private storage on Google

3 new messages 4:11 PM
kumlatar.swankatranami@gmail.com 3

3 new messages 2:42 PM
(754) 263-8267, 456

Anatomy of a notification

- Sending application's notification icon or the sender's photo
- Notification title and message
- A timestamp
- A secondary icon to identify the sending application when the sender's image is shown for the main icon



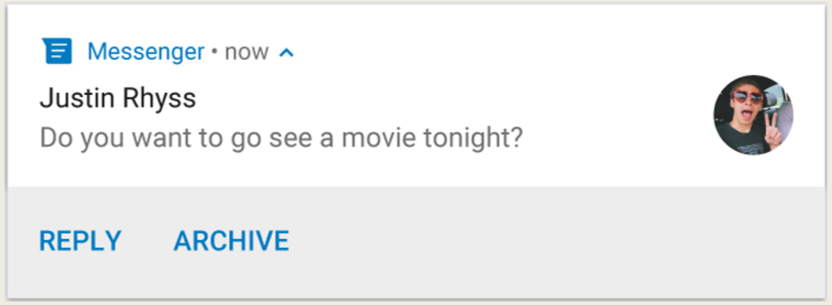
Daniel
4:03PM
Here's the recipe for coq qu vin

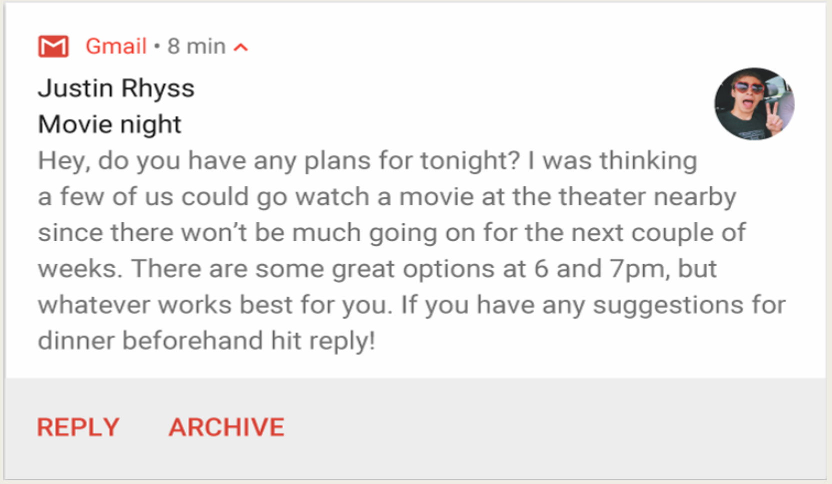
How notifications may be noticed

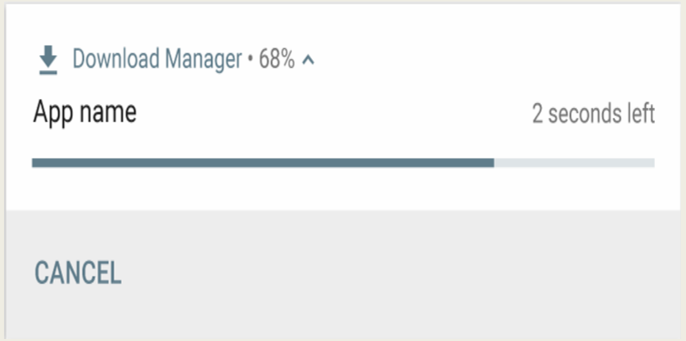
- Showing a status bar icon
- Appearing on the lock screen
- Pulsing the device's LED
- Playing a sound or vibrating
- Peeking onto the current screen

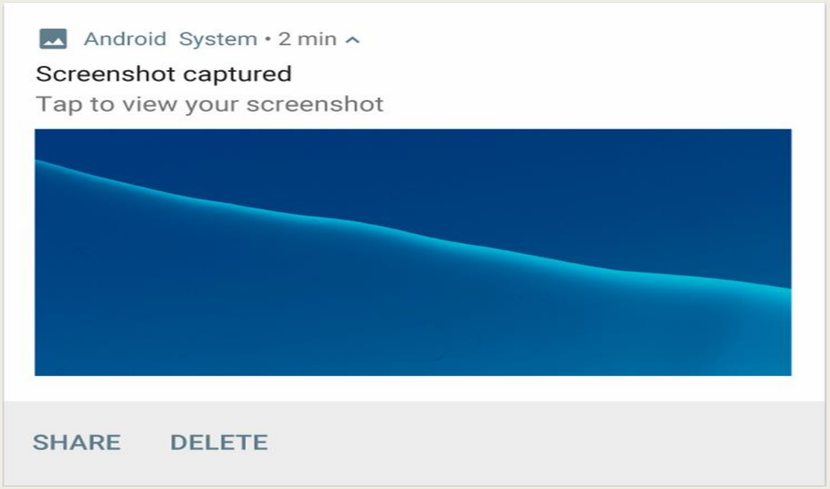
Notification Templates

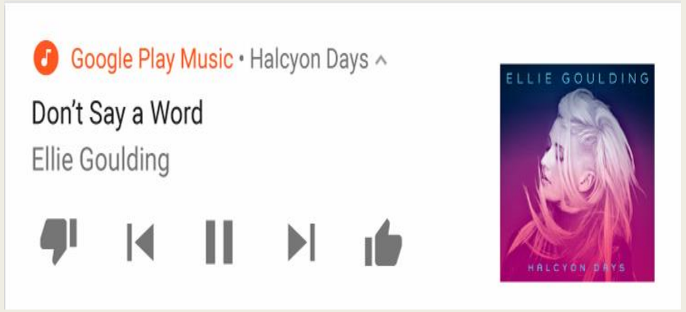
- Standard
- Big text
- Big picture
- Progress
- Media


Messenger • now
Justin Rhyss
Do you want to go see a movie tonight?
REPLY ARCHIVE


Gmail • 8 min
Justin Rhyss
Movie night
Hey, do you have any plans for tonight? I was thinking a few of us could go watch a movie at the theater nearby since there won't be much going on for the next couple of weeks. There are some great options at 6 and 7pm, but whatever works best for you. If you have any suggestions for dinner beforehand hit reply!
REPLY ARCHIVE


Download Manager • 68%
App name 2 seconds left
CANCEL

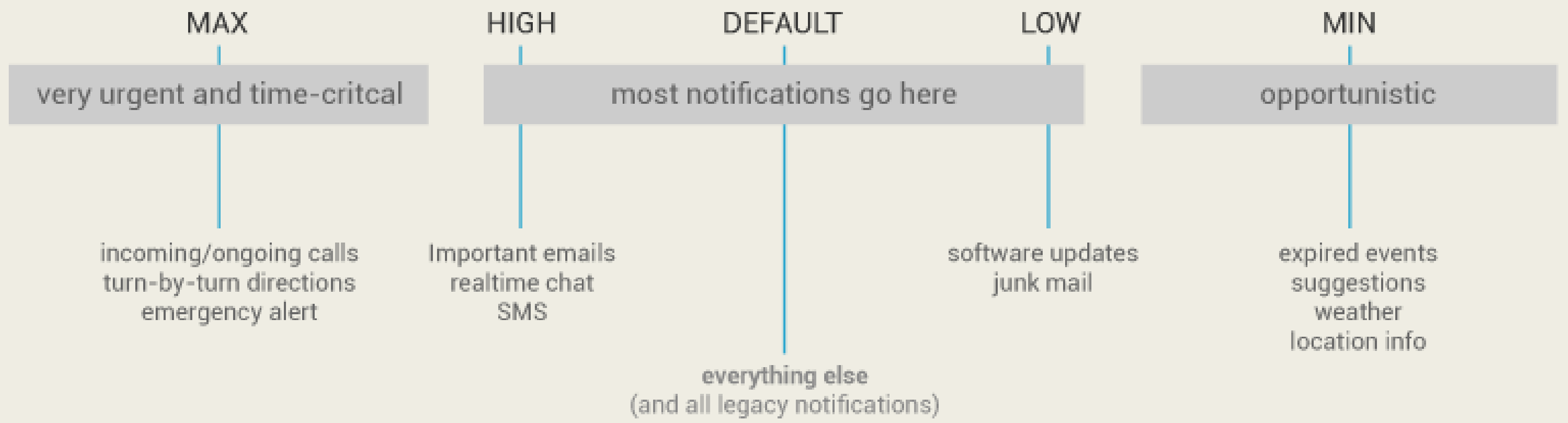

Android System • 2 min
Screenshot captured
Tap to view your screenshot
SHARE DELETE


Google Play Music • Halcyon Days
Don't Say a Word
Ellie Goulding

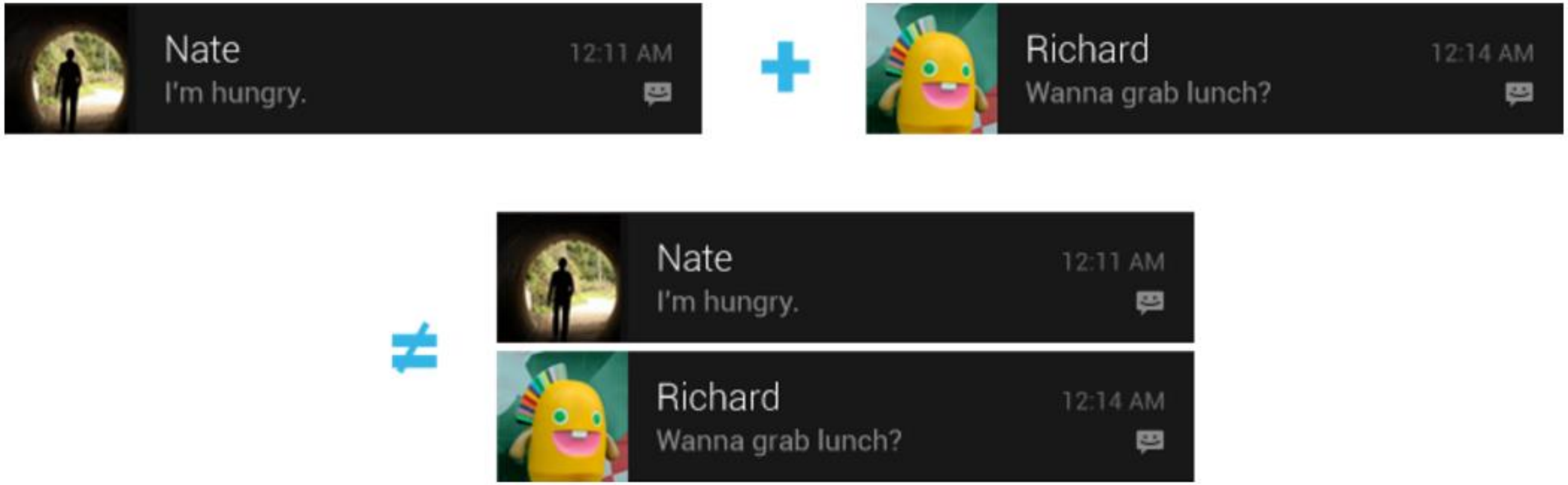
Notification Priority

- Optionally, you can set the priority of a notification
- The priority acts as a hint to the device UI about how the notification should be displayed
- To set a notification's priority, call NotificationCompat.Builder.setPriority() and pass in one of the NotificationCompat priority constants
- There are five priority levels, ranging from PRIORITY_MIN (-2) to PRIORITY_MAX (2); if not set, the priority defaults to PRIORITY_DEFAULT (0)

Priority	Use
MAX	Use for critical and urgent notifications that alert the user to a condition that is time-critical or needs to be resolved before they can continue with a particular task.
HIGH	Use high priority notifications primarily for important communication, such as message or chat events with content that is particularly interesting for the user.
DEFAULT	The default priority. Keep all notifications that don't fall into any of the other categories at this priority level.
LOW	Use for notifications that you still want the user to be informed about, but that rate low in urgency.
MIN	Contextual/background information (e.g. weather information, contextual location information). Minimum priority notifications will not show in the status bar. The user will only discover them when they expand the notification tray.



Don't:



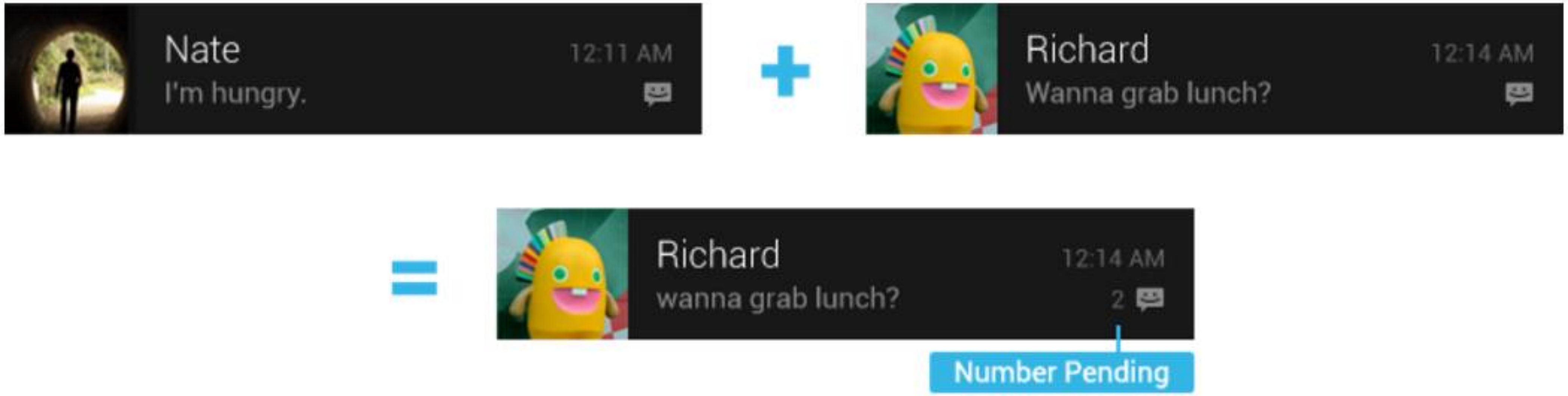
Nate 12:11 AM
I'm hungry.

Richard 12:14 AM
Wanna grab lunch?

Nate 12:11 AM
I'm hungry.

Richard 12:14 AM
Wanna grab lunch?

Do:



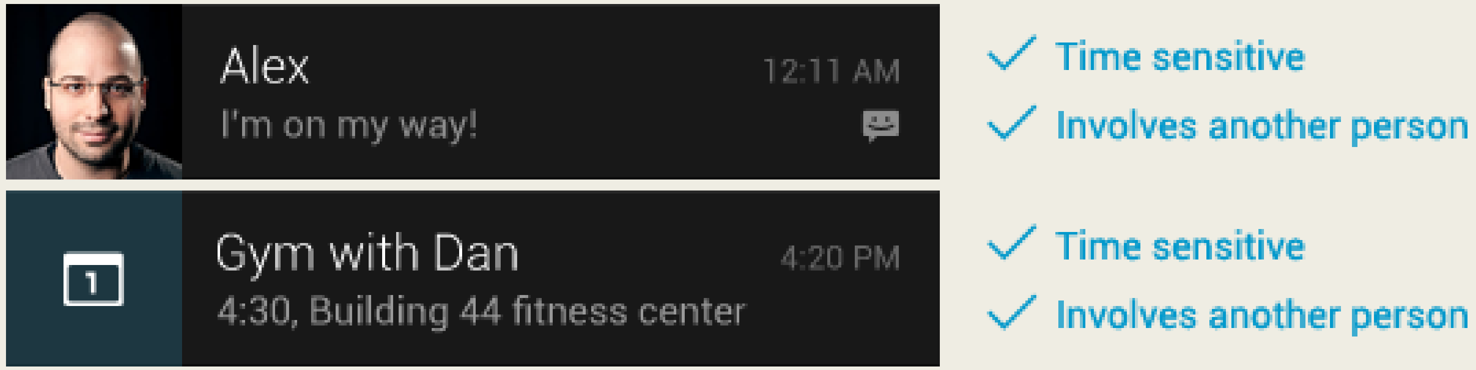
Nate 12:11 AM
I'm hungry.

Richard 12:14 AM
Wanna grab lunch?

Richard 12:14 AM
wanna grab lunch? 2

When to display notifications

- There are two basic rules:
  - **Time sensitive events**
  - **Events involve other people**



Alex 12:11 AM
I'm on my way!

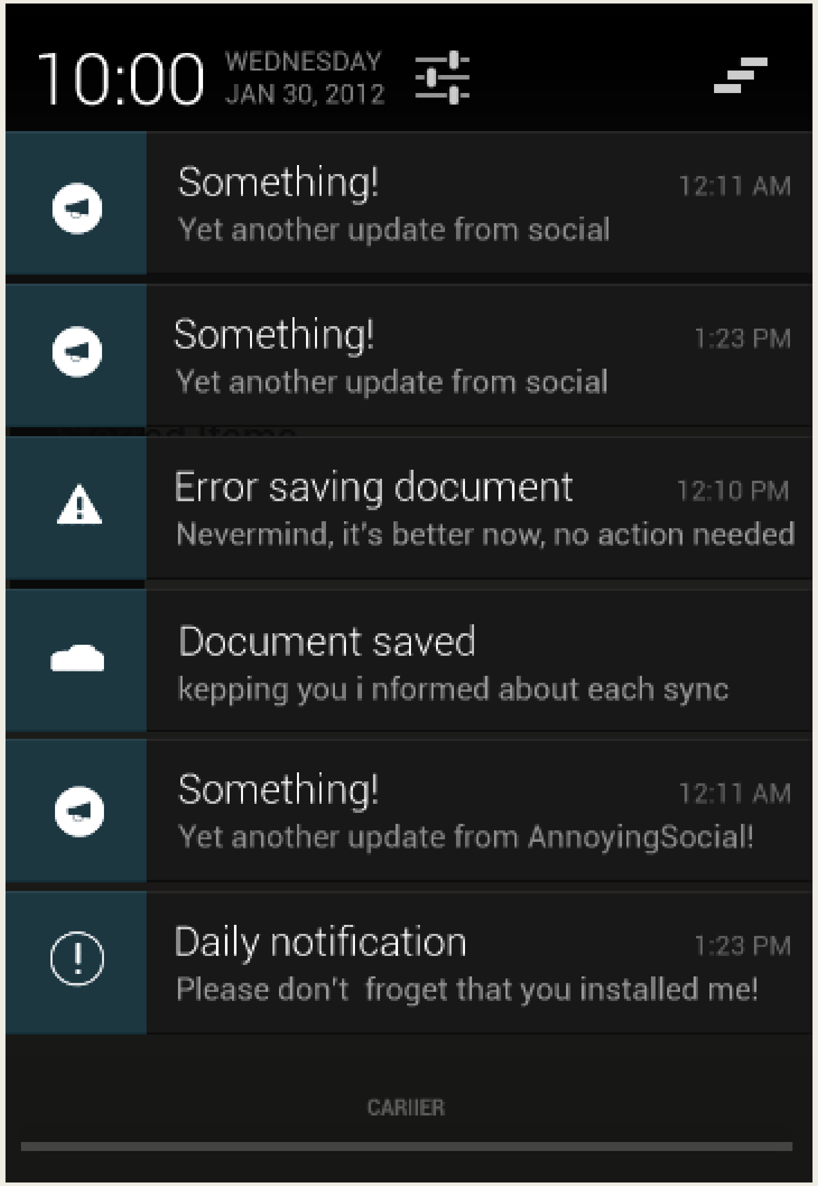
✓ Time sensitive
✓ Involves another person

Gym with Dan 4:20 PM
4:30, Building 44 fitness center

✓ Time sensitive
✓ Involves another person

When not to display notifications

- Avoid notifying the user of information that is not directed specifically at them
- Don't create a notification if the relevant new information is currently on screen
- Don't interrupt the user for low level technical operations, like saving or syncing information
- Don't interrupt the user to inform them of an error if it is possible for the application to quickly recover from the error
- Don't create notifications that have no true notification content and merely advertise your app
- Don't create superfluous notifications just to get your brand in front of users


10:00 WEDNESDAY JAN 30, 2012

Something! 12:11 AM
Yet another update from social

Something! 1:23 PM
Yet another update from social

Error saving document 12:10 PM
Nevermind, it's better now, no action needed

Document saved
kepping you i nformed about each sync

Something! 12:11 AM
Yet another update from AnnoyingSocial!

Daily notification 1:23 PM
Please don't froget that you installed me!

CARIIER

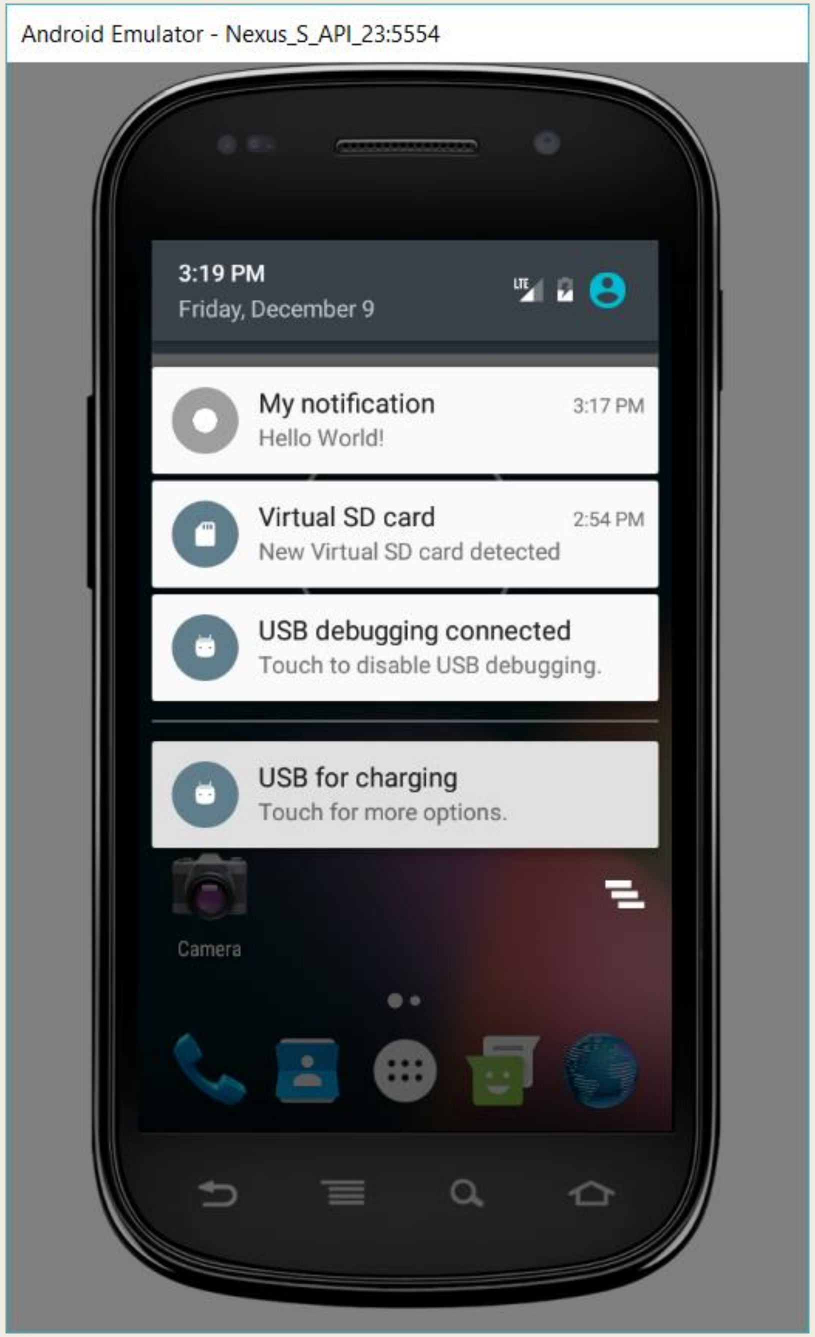
Creating a Notification

- You specify the UI information and actions for a notification in a NotificationCompat.Builder object
- To create the notification itself, you call NotificationCompat.Builder.build(), which returns a Notification object containing your specifications
- To issue the notification, you pass the Notification object to the system by calling NotificationManager.notify()

Required notification contents

- A Notification object must contain the following:
  - *A small icon, set by setSmallIcon()*
  - *A title, set by setContentTitle()*
  - *Detail text, set by setContentText()*
- Optional:
  - *Define the Notification's Action*

Example


Android Emulator - Nexus_S_API_23:5554

3:19 PM
Friday, December 9

My notification 3:17 PM
Hello World!

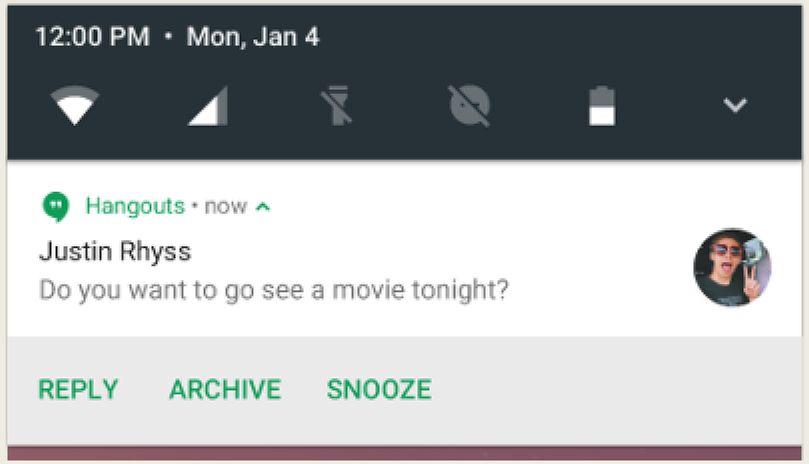
Virtual SD card 2:54 PM
New Virtual SD card detected

USB debugging connected
Touch to disable USB debugging.

USB for charging
Touch for more options.

Replying to notifications

- Starting in Android 7.0 (API level 24), users can respond directly to text messages or update task lists from within the notification dialog
- The inline reply action appears as an additional button displayed in the notification
- When a user replies via keyboard, the system attaches the text response to the intent you had specified for the notification action and sends the intent to your handheld app


12:00 PM • Mon, Jan 4

Hangouts • now

Justin Rhyss

Do you want to go see a movie tonight?