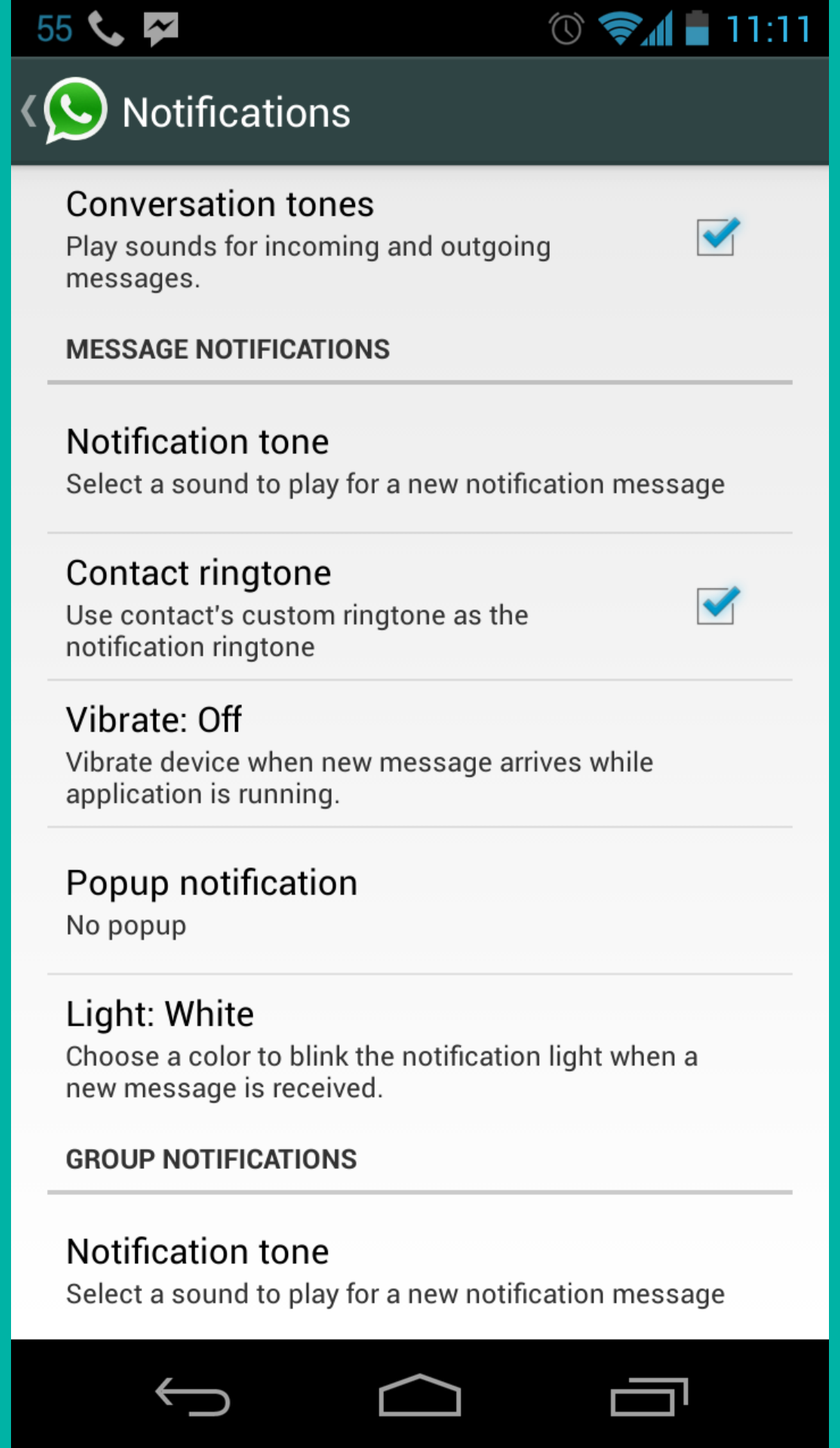
ListView


55
11:11
Notifications
Conversation tones
Play sounds for incoming and outgoing messages.
MESSAGE NOTIFICATIONS
Notification tone
Select a sound to play for a new notification message
Contact ringtone
Use contact's custom ringtone as the notification ringtone
Vibrate: Off
Vibrate device when new message arrives while application is running.
Popup notification
No popup
Light: White
Choose a color to blink the notification light when a new message is received.
GROUP NOTIFICATIONS
Notification tone
Select a sound to play for a new notification message

ListView is a subclass of ViewGroup

- LinearLayout and Relative Layout are other views that extend ViewGroup
- ViewGroup is just a view that can contain child views

Lists and Activities

- Activity has a special subclass `ListActivity` which hosts a ListView
- ListViews can also be used with regular Activities
- If using a ListActivity, your ListView object must have the id "@android:id/list"

Rows

- Each row in list view is its own view
- Rows can be composed with their own layout xml file
- Views can be loaded in listview using findViewById

So we have this problem..

ListViews are really really boring and frankly useless without.....

Enter The Adapter

Adapters

- Adapters serve as a bridge between your data model and view
- Adapter manages the data model and adapts it to individual rows of the ListView
- Each row in the ListView can be composed of as many widgets as your application sees fit
- Adapters can be subclassed for specific use cases.....

- Which is exactly why adapters are so powerful
- Decouple, decouple, decouple
- Swap out adapters when your implementation details change

ex: ArrayAdapter -> CursorAdapter

You have an app backed by an Array. You decide you want to use SQLite to persist your data.

ArrayAdapter

- Concrete subclass of BaseAdapter
- Constructor `public ArrayAdapter(Context context, int resource, T[] objects)`
- Uses each object of type T's toString method to provide text to a TextView resource

Steps for using ArrayAdapter

- Create List of objects ex. `List<Person>people = new ArrayList<people>(); people.add(...)
- Get a reference to ListView GridView etc..
- Instantiate Adapter `ArrayAdapter adapter = new ArrayAdapter(this, R.Layout.some_layout_with_text_view, people)`
- Set List View's Adapter `listView.setAdapter(adapter);`

Sometimes you need a little more control

Subclassing ArrayAdapter

- Override desired Constructor and pass in a context (Context should be stored in a member variable)
- Override `public View getView(int position, View convertView, ViewGroup parent)` method to configure view as desired