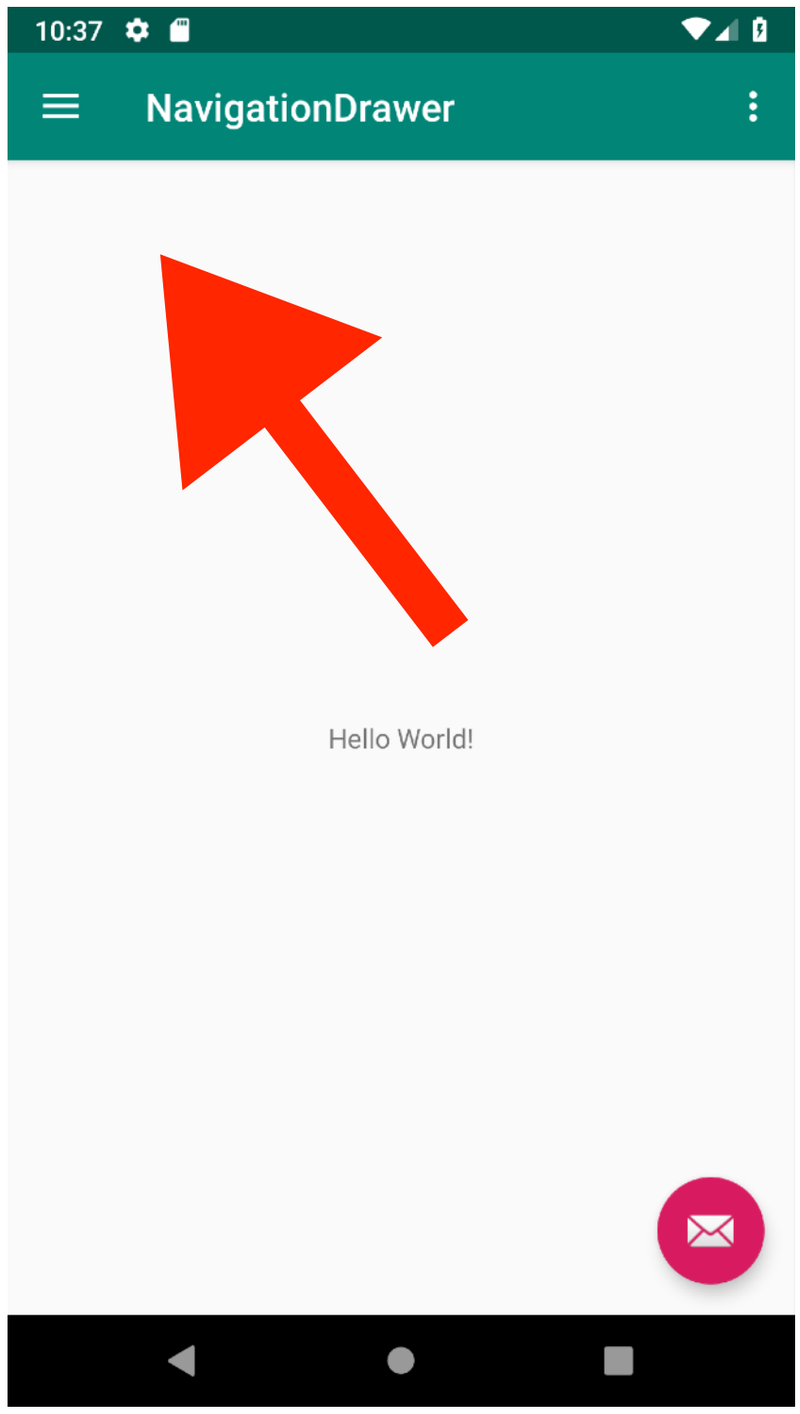
Lecture #3
Navigation and Rest Resources

Mobile Applications 2019-2020

Navigation Drawer

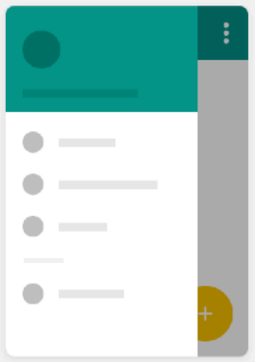
- App main navigation menu.
- Hidden when not in use.
- Appears:
  - with a left swipe from the screen edge
  - when the user touches the drawer icon in the app bar



https://developer.android.com/training/implementing-navigation/nav-drawer

Create New Project

Configure your project



Navigation Drawer Activity

Creates a new Activity with a Navigation Drawer.

Name

My Application

Package name

com.example.myapplication

Save location

/Users/dan/work/school/ma/lectures/3/NavigationDrawer

Language

Kotlin

Minimum API level API 15: Android 4.0.3 (IceCreamSandwich)

Your app will run on approximately **100%** of devices.

Help me choose

This project will support instant apps

Use androidx.* artifacts

Cancel Previous Next Finish

8:03

Android Studio
android.studio@android.com

Home

Gallery

Slideshow

Tools

Communicate

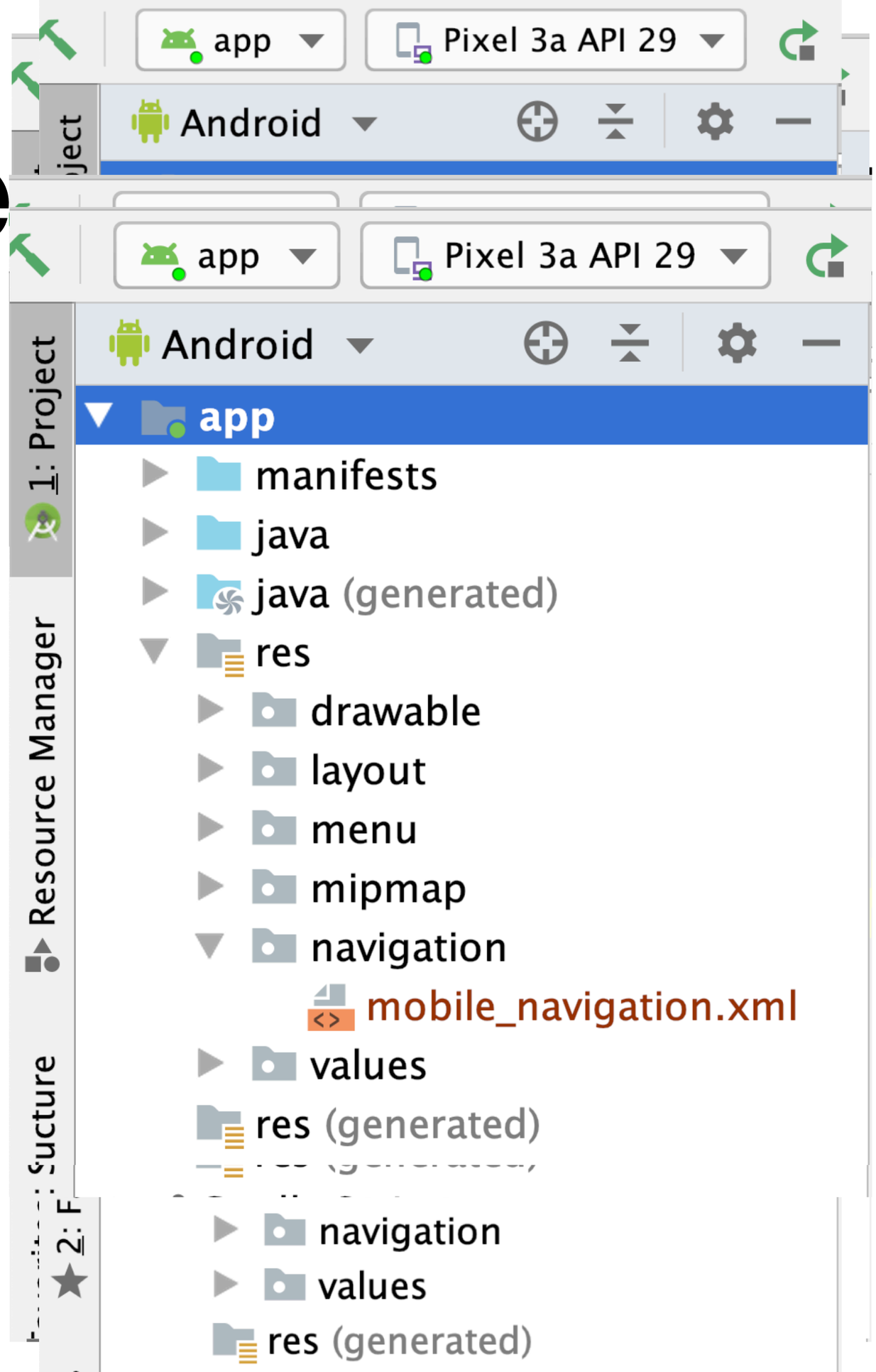
Share

Send

nt

Gene…s

- Sources
- Layouts
- Menus
- Navigation



Dependencies

```
buildscript {
    ext.nav_version = "2.1.0"
}
...
dependencies {
  implementation "androidx.navigation:navigation-fragment-ktx:$nav_version"
  implementation "androidx.navigation:navigation-ui-ktx:$nav_version"
}
```

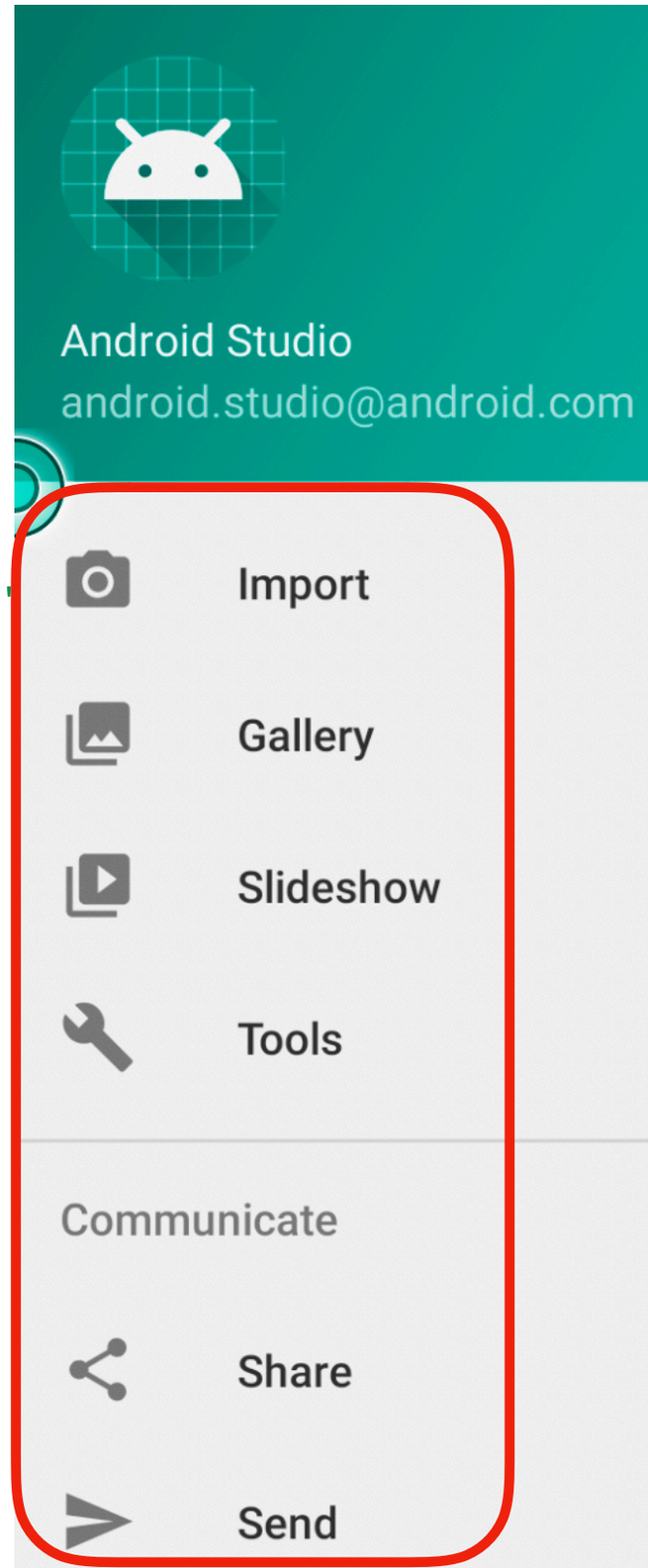
Add a drawer to a layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:id="@+id/drawer_layout"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:fitsSystemWindows="true"
  tools:openDrawer="start">
 <include
    layout="@layout/app_bar_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
 <com.google.android.material.navigation.NavigationView
    android:id="@+id/nav_view"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:fitsSystemWindows="true"
    app:headerLayout="@layout/nav_header_main"
    app:menu="@menu/activity_main_drawer" />
</androidx.drawerlayout.widget.DrawerLayout>
```

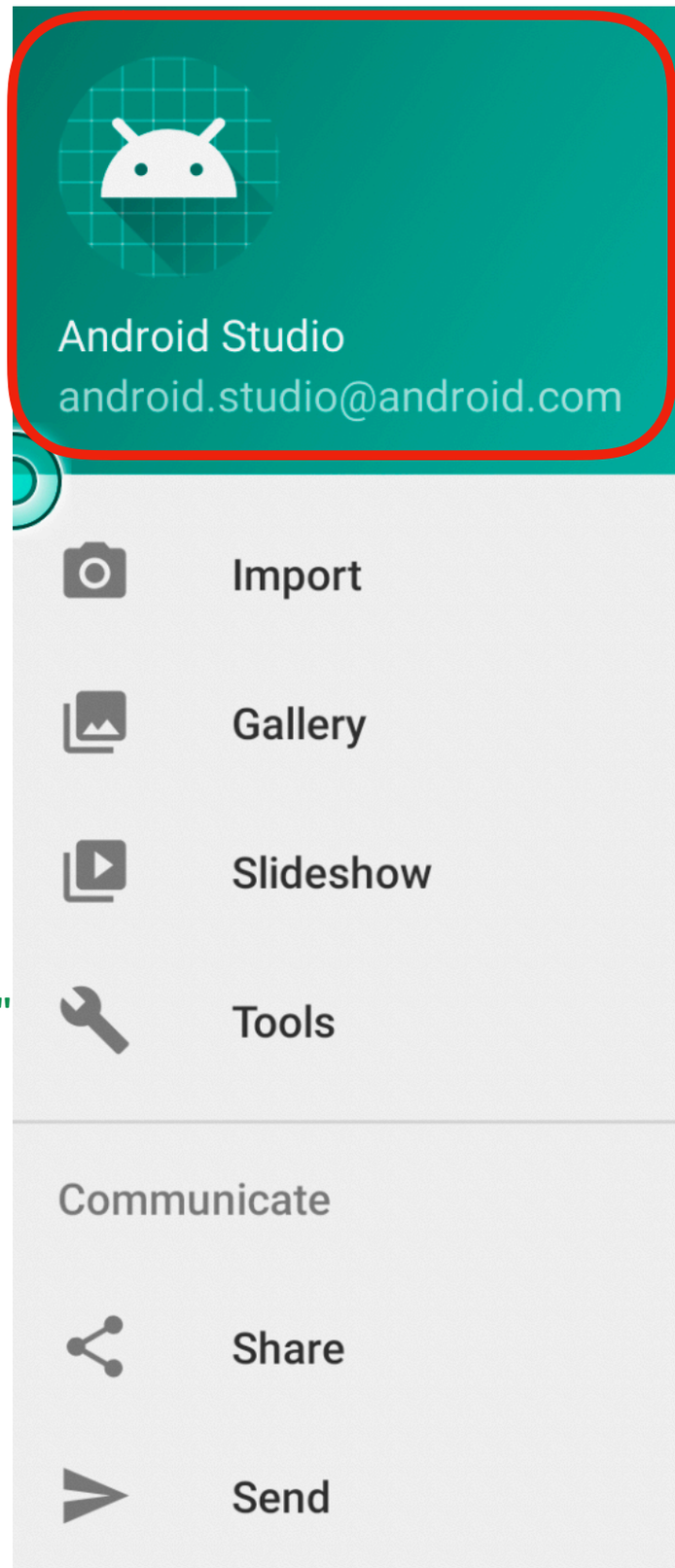
Declare the menu items

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_camera"
            android:icon="@drawable/ic_menu_camera"
            android:title="@string/import" />
        <item
            android:id="@+id/nav_gallery"
            android:icon="@drawable/ic_menu_gallery"
            android:title="@string/gallery" />
        <item
            android:id="@+id/nav_slideshow"
            android:icon="@drawable/ic_menu_slideshow"
            android:title="@string/slideshow" />
        ...
    </group>
</menu>
```

```
<android.support.design.widget.NavigationView
    android:id="@+id/nav_view"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:fitsSystemWindows="true"
    app:menu="@menu/activity_main_drawer" />
```



Add a header to the nav drawer




```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="192dp"
    android:background="?attr/colorPrimaryDark"
    android:padding="16dp"
    android:theme="@style/ThemeOverlay.AppCompat.Dark"
    android:orientation="vertical"
    android:gravity="bottom">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="My header title"
        android:textAppearance=
           "@style/TextAppearance.AppCompat.Body1"/>
</LinearLayout>
```

```
<android.support.design.widget.NavigationView
    android:id="@+id/nav_view"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:fitsSystemWindows="true"
    app:menu=”@menu/activity_main_drawer” />
    app:headerLayout="@layout/nav_header_main"
```

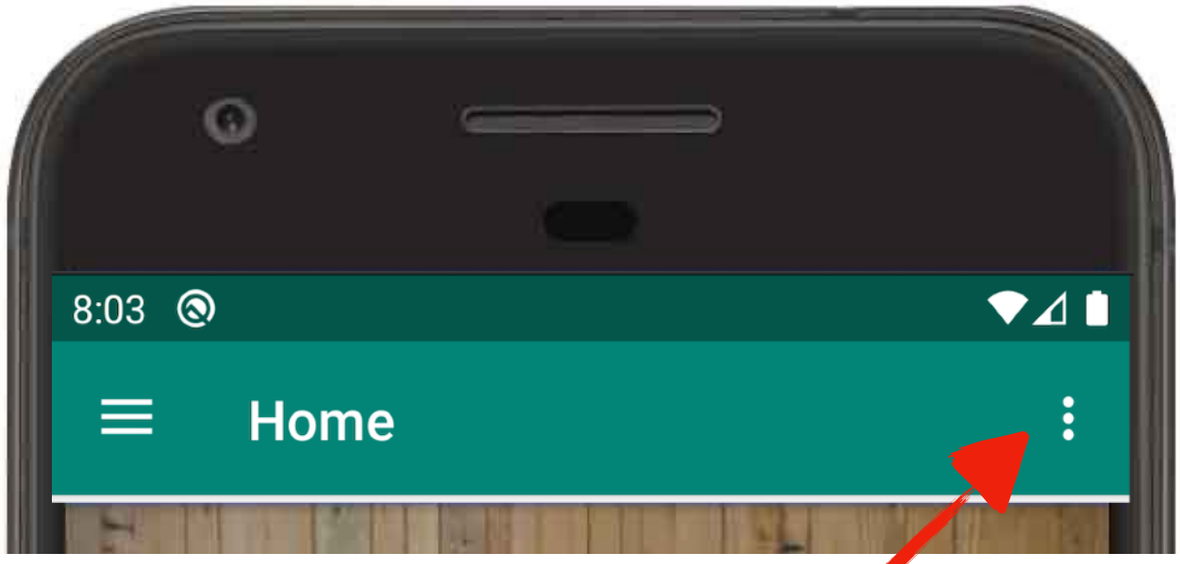
Handle navigation events

```
class MainActivity : AppCompatActivity() {

  private lateinit var appBarConfiguration: AppBarConfiguration

  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    setSupportActionBar(toolbar)

    val navController = findNavController(R.id.nav_host_fragment)

    appBarConfiguration = AppBarConfiguration(
        setOf(
            R.id.nav_home,
            R.id.nav_gallery
        ), drawer_layout
    )
    setupActionBarWithNavController(navController, appBarConfiguration)
    nav_view.setupWithNavController(navController)
  }
}
```

Add a toolbar

OLD

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
  <com.google.android.material.appbar.AppBarLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:theme="@style/AppTheme.AppBarOverlay">
    <androidx.appcompat.widget.Toolbar
      android:id="@+id/toolbar"
      android:layout_width="match_parent"
      android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
      app:popupTheme="@style/AppTheme.PopupOverlay" />
  </com.google.android.material.appbar.AppBarLayout>
```



NEW

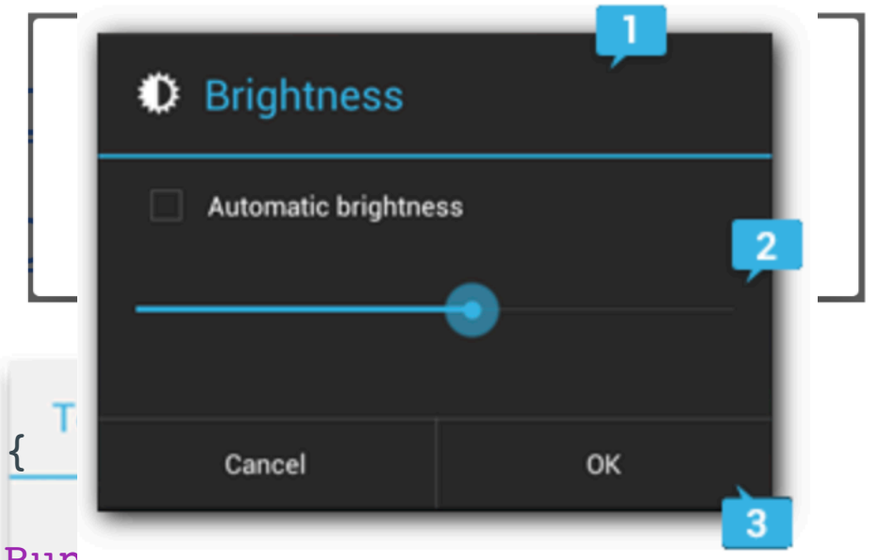
```
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    // Inflate the menu; this adds items to the action bar if it is preser
    menuInflater.inflate(R.menu.main, menu)
    return true
}
```

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(
        savedInstanceState: Bundle?) {
        ...
        setSupportActionBar(toolbar)
        ...
    }
```

Other State Changes

```
drawer_layout.addDrawerListener(
  object : DrawerLayout.DrawerListener {
    override fun onDrawerSlide(drawerView: View, slideOffset: Float) {
      // Respond when the drawer's position changes
    }

    override fun onDrawerOpened(drawerView: View) {
      // Respond when the drawer is opened
    }

    override fun onDrawerClosed(drawerView: View) {
      // Respond when the drawer is closed
    }

    override fun onDrawerStateChanged(newState: Int) {
      // Respond when the drawer motion state changes
    }
  }
)
```

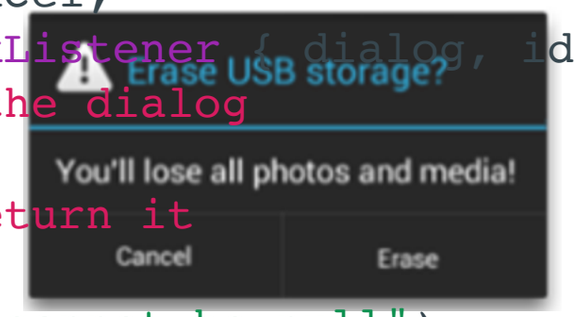
Dialogs




```
class FireMissilesDialogFragment : DialogFragment() {

    override fun onCreateDialog(savedInstanceState: Bundle): Dialog {
        return activity?.let {
            // Use the Builder class for convenient dialog construction
            val builder = AlertDialog.Builder(it)
            builder.setMessage(R.string.dialog_fire_missiles)
                    .setPositiveButton(R.string.fire,
                            DialogInterface.OnClickListener { dialog, id ->
                                // FIRE ZE MISSILES!
                            })
                    .setNegativeButton(R.string.cancel,
                            DialogInterface.OnClickListener { dialog, id ->
                                // User cancelled the dialog
                            })
            // Create the AlertDialog object and return it
            builder.create()
        } ?: throw IllegalStateException("Activity cannot be null")
    }
}
```

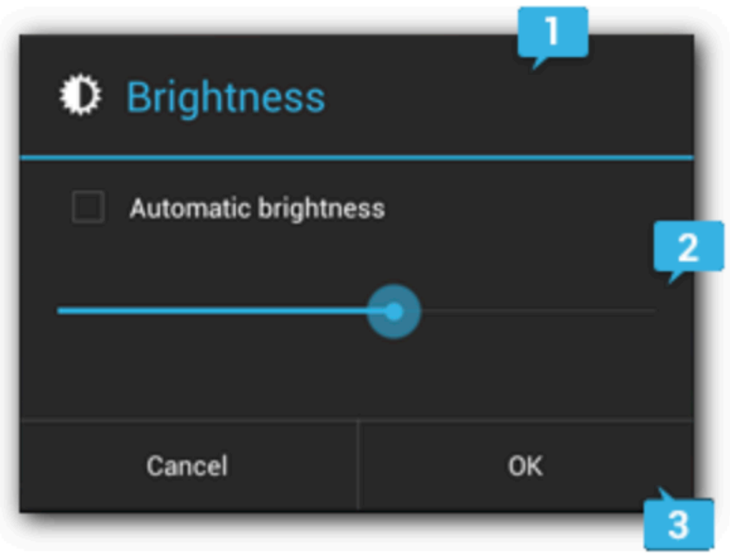
```
// 1. Instantiate an AlertDialog.Builder with its constructor
val builder: AlertDialog.Builder? = activity?.let {
    AlertDialog.Builder(it)
}

// 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()
val dialog: AlertDialog? = builder?.create()
```

- Small window.
- Prompts the user to take a decision.
- Modal, by default.



https://developer.android.com/guide/topics/ui/dialogs

Adding actions

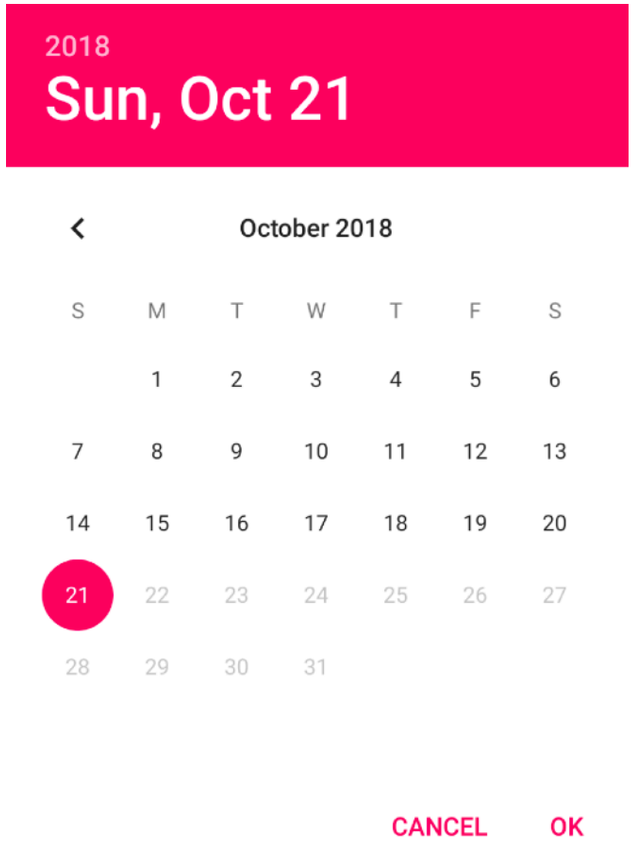



```
val alertDialog: AlertDialog? = activity?.let {
    val builder = AlertDialog.Builder(it)
    builder.apply {
        setPositiveButton(R.string.ok,
                DialogInterface.OnClickListener { dialog, id ->
                    // User clicked OK button
                })
        setNegativeButton(R.string.cancel,
                DialogInterface.OnClickListener { dialog, id ->
                    // User cancelled the dialog
                })
    }
    // Set other dialog properties
    ...

    // Create the AlertDialog
    builder.create()
}
```

https://developer.android.com/guide/topics/ui/dialogs

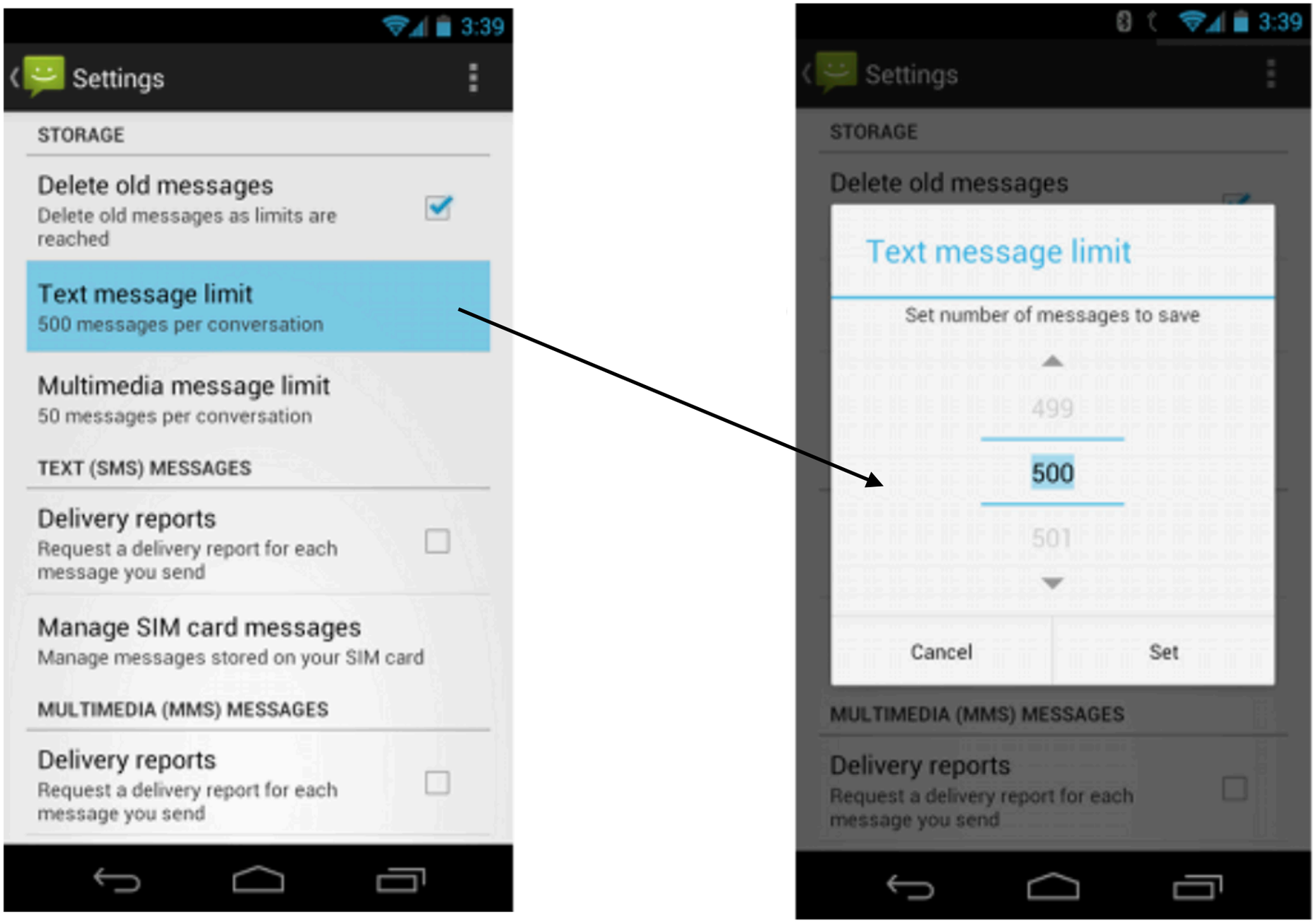
Using Anko

```
alert {
    isCancelable = false
    lateinit var datePicker: DatePicker
    customView {
        verticalLayout {
            datePicker = datePicker {
                maxDate = System.currentTimeMillis()
            }
        }
    }
    yesButton {
        val parsedDate =
          "${datePicker.dayOfMonth}/${datePicker.month + 1}/${datePicker.year}"
        toast("Selected date: $parsedDate")
    }
    noButton { }
}.show()
```

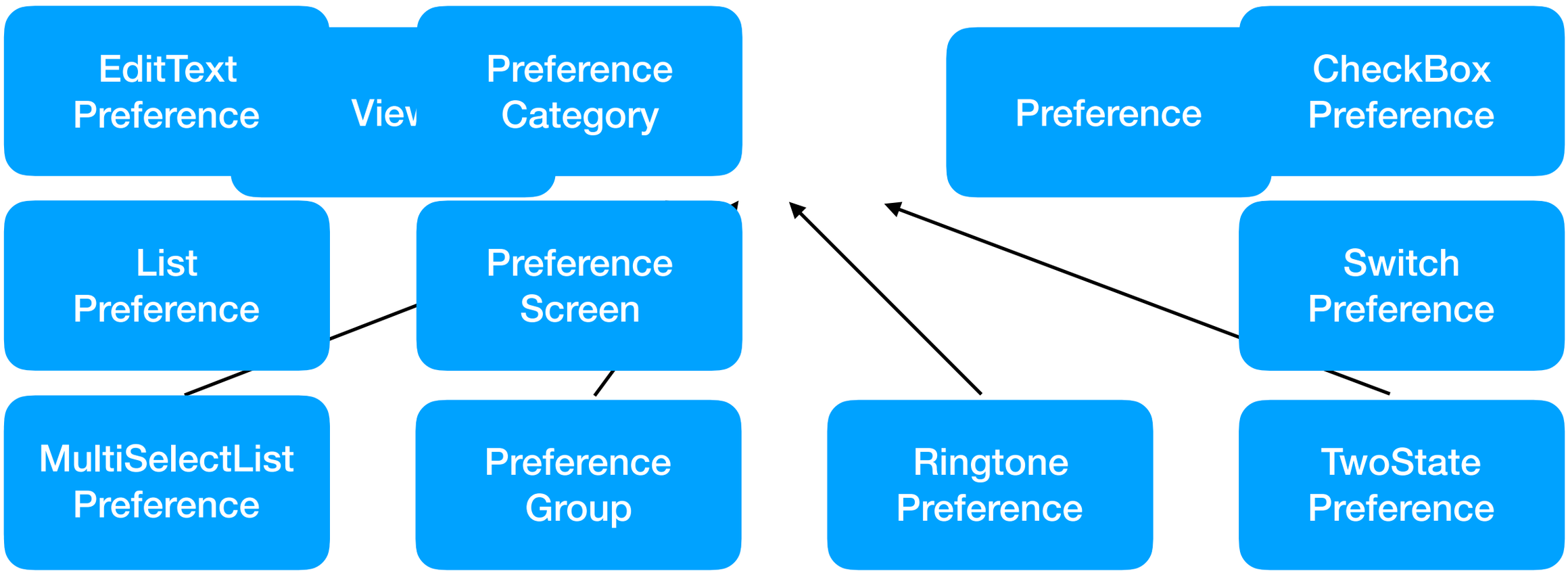


https://github.com/Kotlin/anko/wiki/Anko-Commons---Dialogs

Preferences



Preferences



https://developer.android.com/reference/android/preference/Preference

Preferences

Preference

String

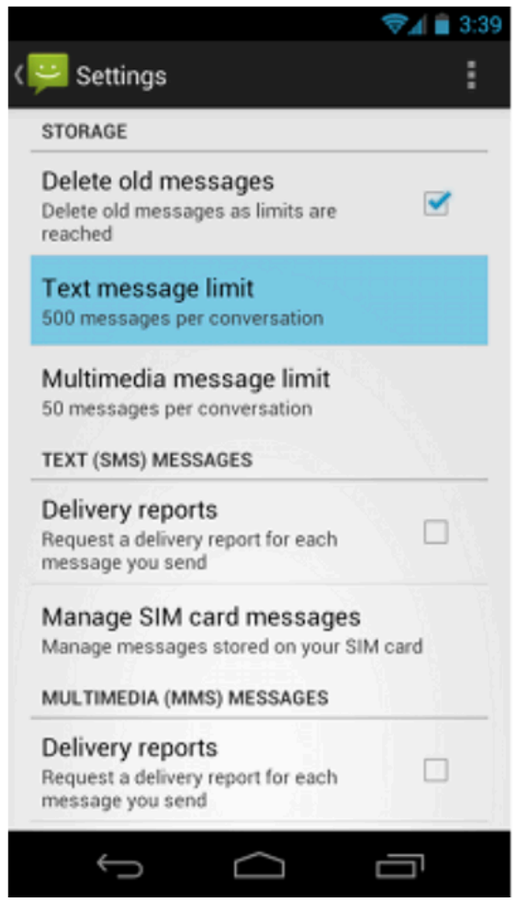
Set<String>

DEMO

PreferenceScreen

```xml
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
    xmlns:android="http://schemas.android.com/apk/res/android">
     <CheckBoxPreference
         android:key="pref_sync"
         android:title="@string/pref_sync"
         android:summary="@string/pref_sync_summ"
         android:defaultValue="true" />
          <ListPreference
         android:dependency="pref_sync"
         android:key="pref_syncConnectionType"
         android:title="@string/pref_syncConnectionType"
         android:dialogTitle="@string/pref_syncConnectionType"
         android:entries="@array/pref_syncConnectionTypes_entries"
         android:entryValues="@array/pref_syncConnectionTypes_values"
         android:defaultValue="@string/pref_syncConnectionTypes_default" />
```

```
class SettingsActivity : PreferenceActivity() {

    override fun onCreate(savedInstanceState: Bundle) {
        super.onCreate(savedInstanceState)
        addPreferencesFromResource(R.xml.preferences)
    }
}
```



Saving & Reading Local Files

- Internal storage
  - Internal cache files
- External storage
- Shared preferences
- Databases

https://developer.android.com/guide/topics/data/

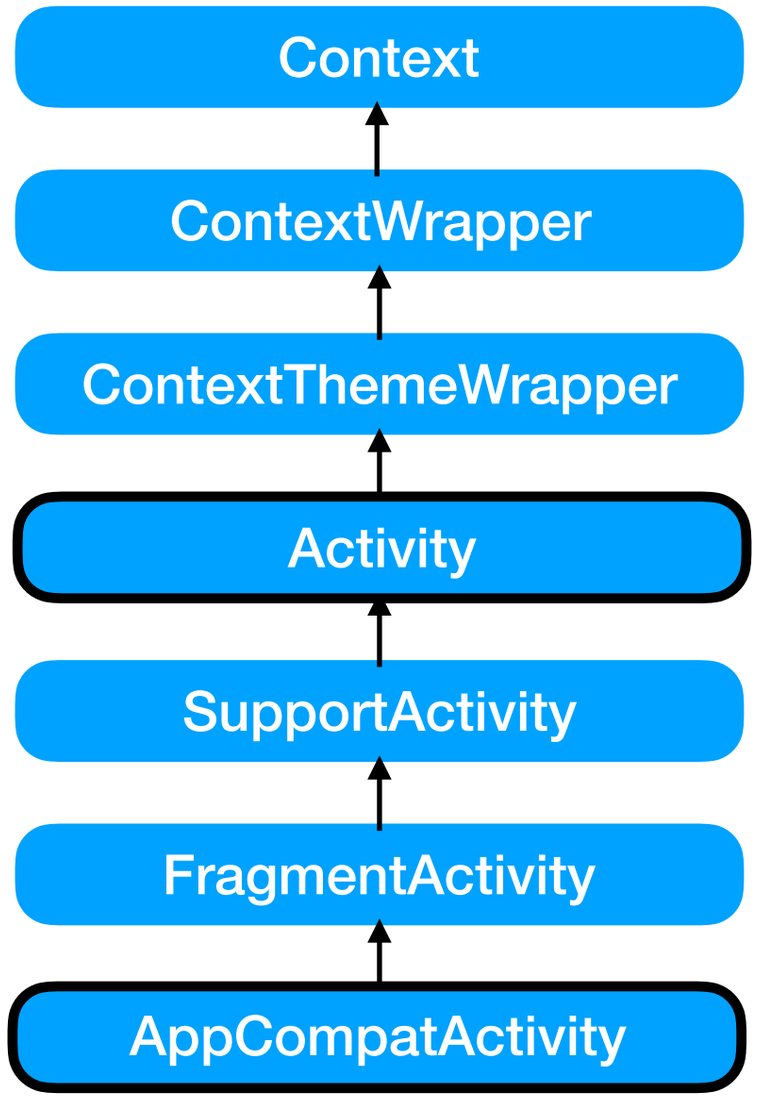
Internal Storage

- It's always available.
- Available only to your app.
- On uninstall everything is removed.

Neither the user nor other apps can access your files!

https://developer.android.com/guide/topics/data/data-storage#filesInternal

Internal Storage




```
val file = File(context.filesDir, filename)
```

```
public abstract class Context {
    ...
    public abstract File getFilesDir();
    ...
    public abstract File getCacheDir();
    ...
}
```

```
val filename = "myfile"
val fileContents = "Hello world!"
context.openFileOutput(filename,
    Context.MODE_PRIVATE).use {
      it.write(fileContents.toByteArray())
}
```

```
private fun getTempFile(
    context: Context, url: String): File? =
 Uri.parse(url)?.lastPathSegment?.let { filename ->
    File.createTempFile(filename, null, context.cacheDir)
 }
```

External Storage Permissions

```
<manifest ...>
    <uses-permission android:name=
       "android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name=
       "android.permission.WRITE_EXTERNAL_STORAGE" />
    ...
</manifest>
```

< Android 4.4 (API level 19)


```
/* Checks if external storage is available for read and write */
fun isExternalStorageWritable(): Boolean {
    return Environment.getExternalStorageState() == Environment.MEDIA_MOUNTED
}

/* Checks if external storage is available to at least read */
fun isExternalStorageReadable(): Boolean {
     return Environment.getExternalStorageState() in
        setOf(Environment.MEDIA_MOUNTED, Environment.MEDIA_MOUNTED_READ_ONLY)
}
```

https://developer.android.com/training/data-storage/files#WriteInternalStorage

FileProvider

content://**com.example.myapp.fileprovider/myimages**/default_image.jpg

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myapp">
    <application
        ...>
        <provider
            android:name="android.support.v4.content.FileProvider"
            android:authorities="com.example.myapp.fileprovider"
            android:grantUriPermissions="true"
            android:exported="false">
            <meta-data
                android:name="android.support.FILE_PROVIDER_PATHS"
                android:resource="@xml/filepaths" />
        </provider>
        ...
    </application>
</manifest>
```

```
<paths>
    <files-path path="images/" name="myimages" />
</paths>
```

https://developer.android.com/training/secure-file-sharing/setup-sharing

Lecture outcomes

- Navigate between screens/ views.
- Use dialogs and pickers.
- Manage files & preferences.