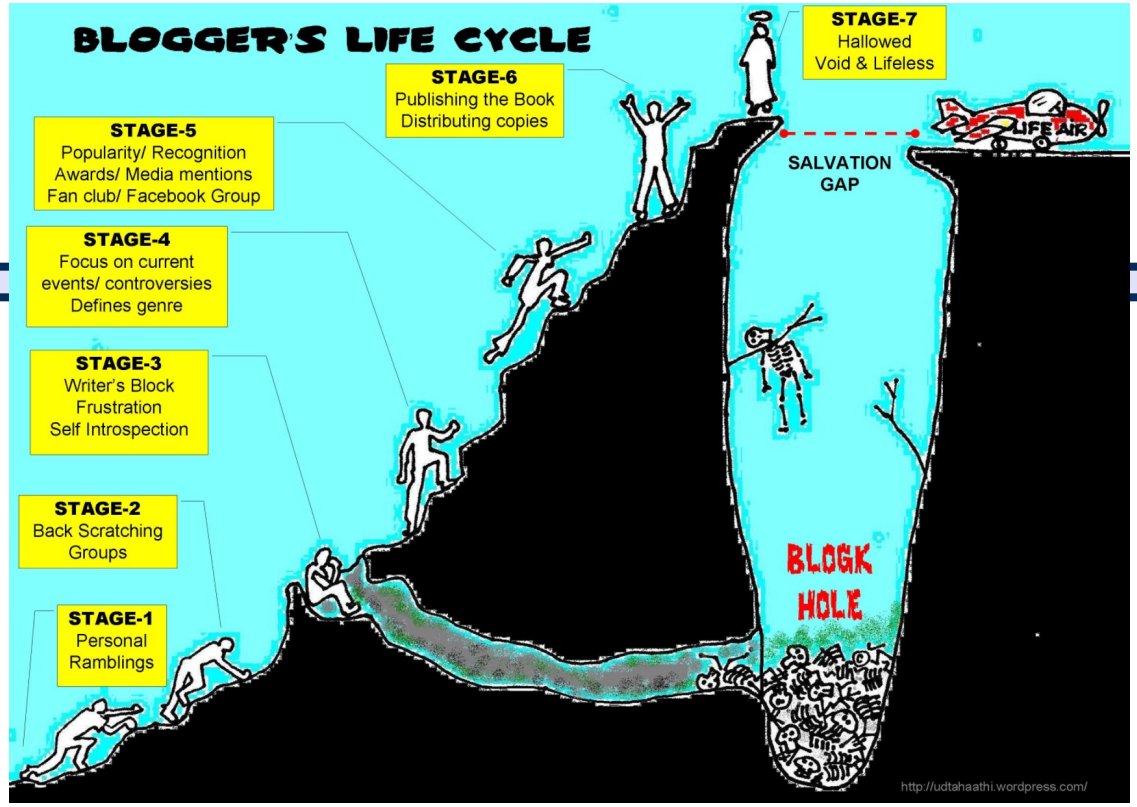
Mobile Applications
CSCI 448
Lecture 08



Activity Lifecycle

+ Resources

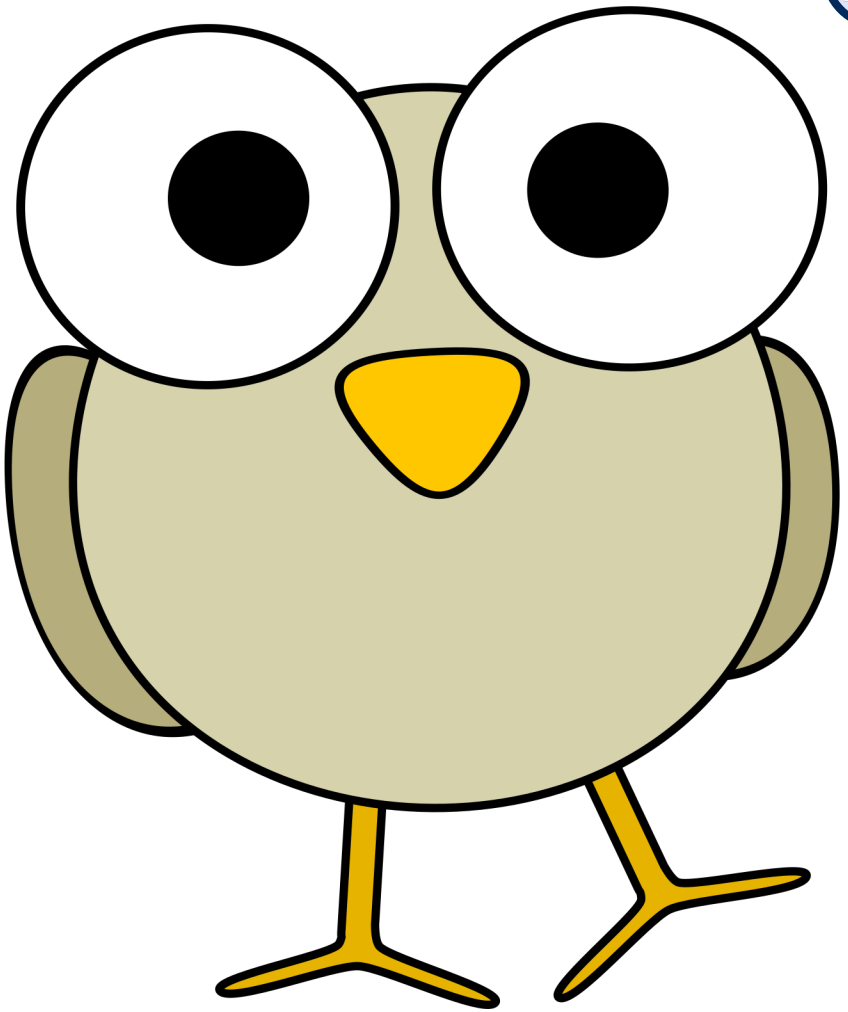
Download TempConverter v2

For Latest Snapshot

Previously in CSCI 448

- Connection between MVVM & Three-Tier Architecture
  - UI/Presentation/Model Logic

Questions?



Learning Outcomes For Today

- Discuss the process of creating and destroying an activity
- Explain the role of the ActivityManager
- Describe how to support multiple screen sizes
- Discuss when resources files get used

On Tap For Today

- Activity Life Cycle
- Logging
- Resources
- Practice

On Tap For Today

- Activity Life Cycle
- Logging
- Resources
- Practice

Starting Your App

- You tell the OS, you want to start a specific app

Starting Your App

- You tell the OS, you want to start a specific app
  1. Application object is created

Starting Your App

- You tell the OS, you want to start a specific app
  1. Application object is created
  2. Launcher Activity object is created

Starting Your App

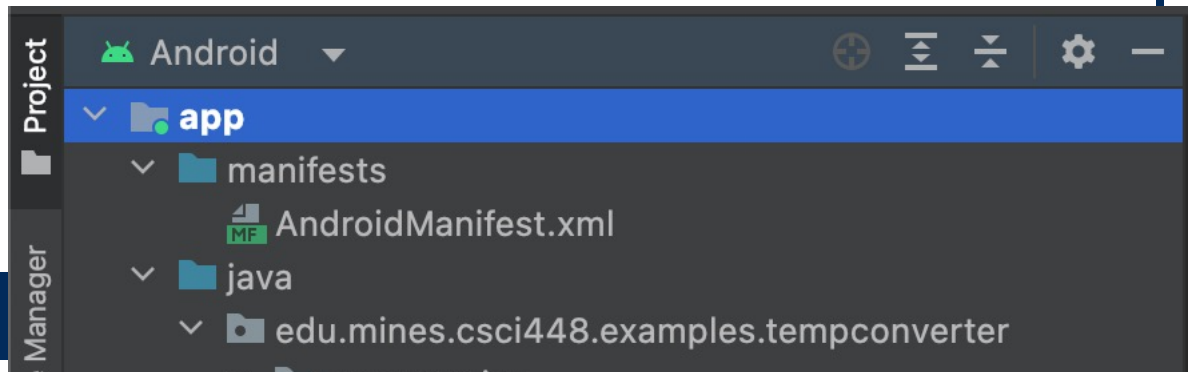
- You tell the OS, you want to start a specific app
  1. Application object is created
  2. Launcher Activity object is created

- How does OS know what to create?

Starting Your App

- You tell the OS, you want to start a specific app
  1. Application object is created
  2. Launcher Activity object is created

- How does OS know what to create?
  - Check the Manifest!



AndroidManifest.xml

```
<application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="TempConverter"
    android:supportsRtl="true"
    android:theme="@style/Theme.TempConverter"
    tools:targetApi="31">
```

AndroidManifest.xml

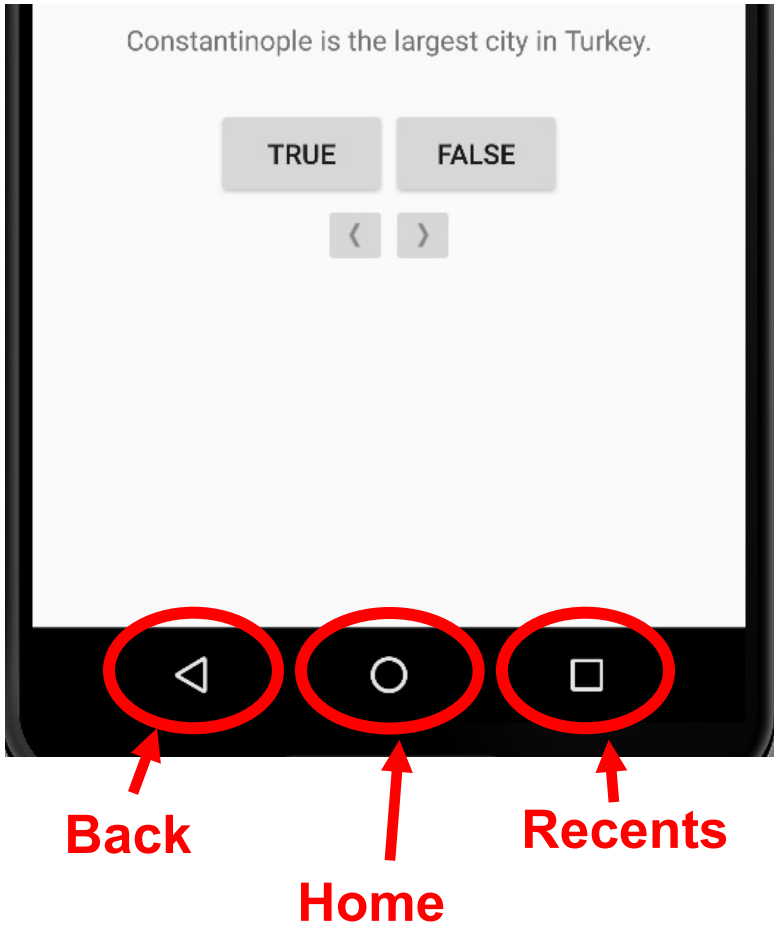
```
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:label="TempConverter"
    android:theme="@style/Theme.TempConverter">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Activity Lifecycle States

- Running
  - visible, user interacting
- Paused
  - visible, user not interacting
- Stopped
  - not visible, can be terminated
- Non-existent

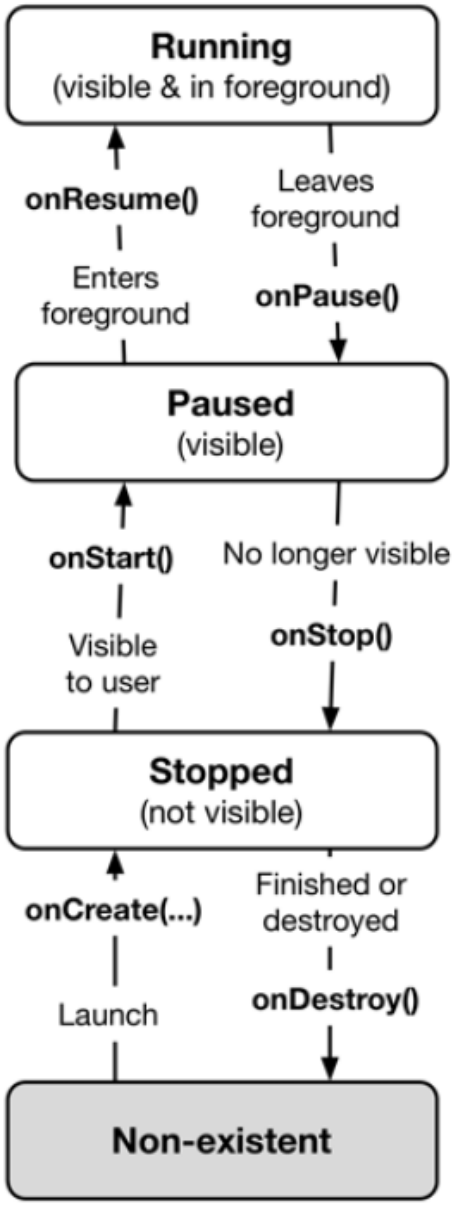
Example - User Actions

- Pressing the “Home” or the “Recents” buttons makes the activity go to the “stopped” state
- Pressing the “back” button terminates the activity

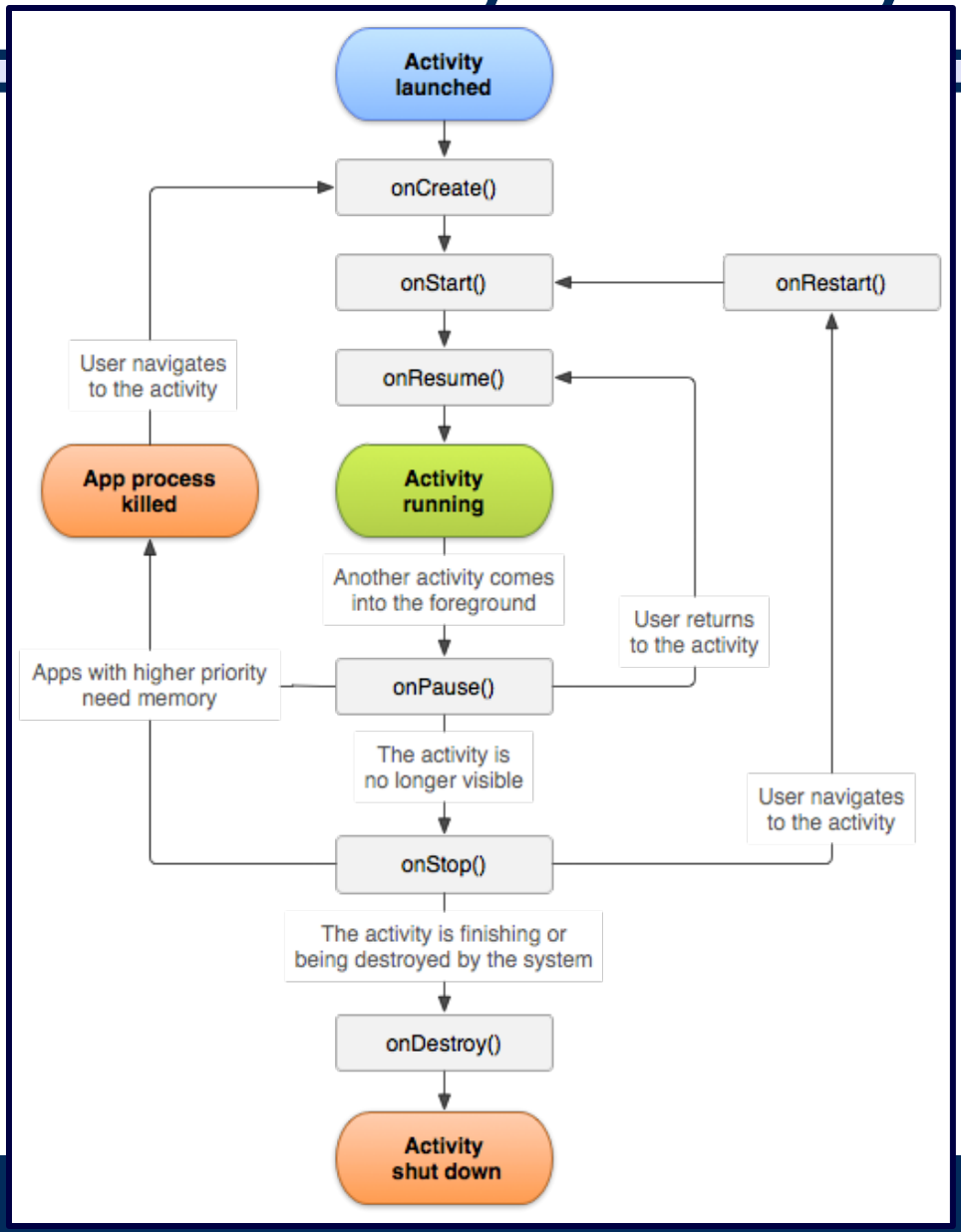


Activity Lifecycle

- Android AcivityManager calls specific methods when the state changes

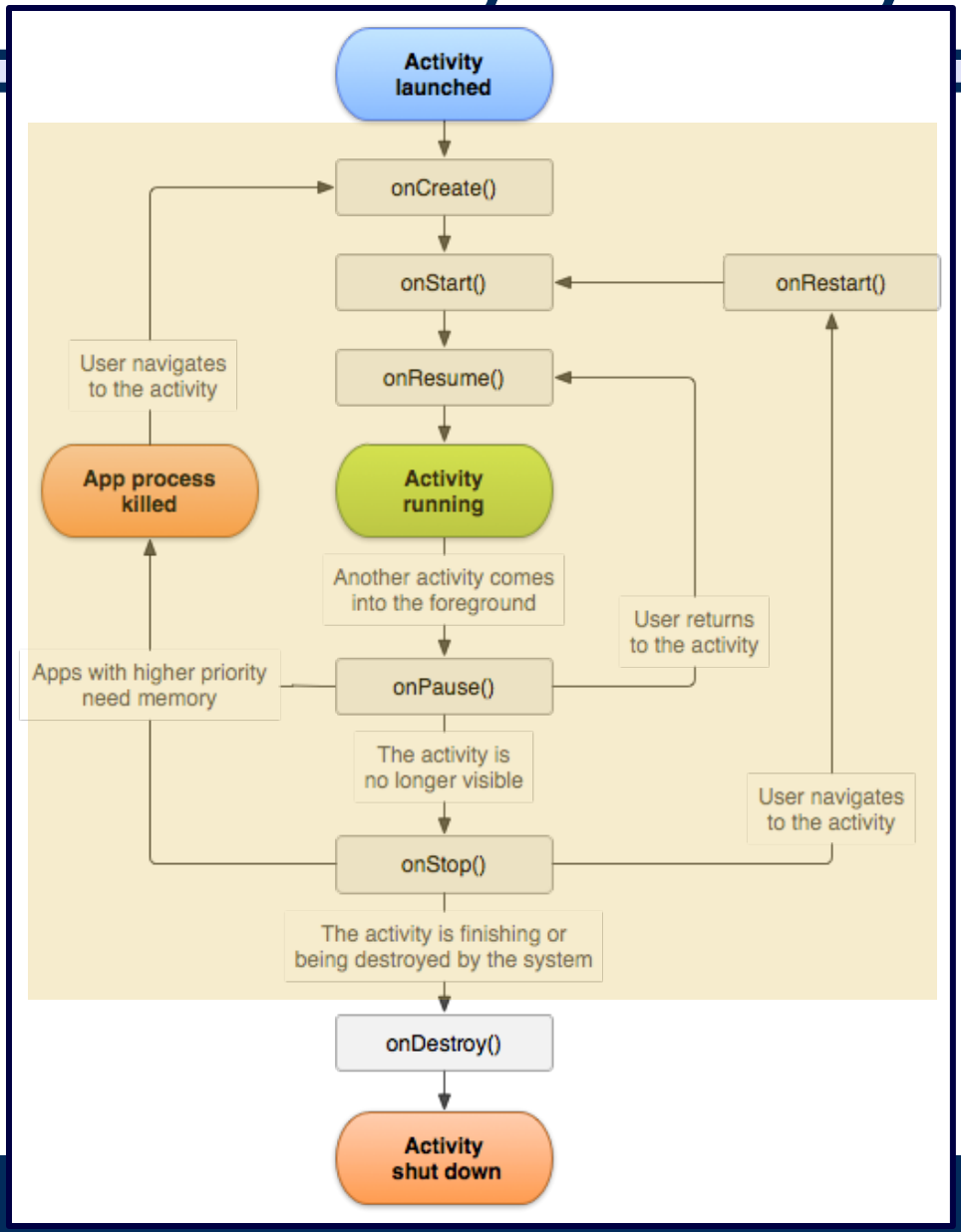


The Activity Life Cycle



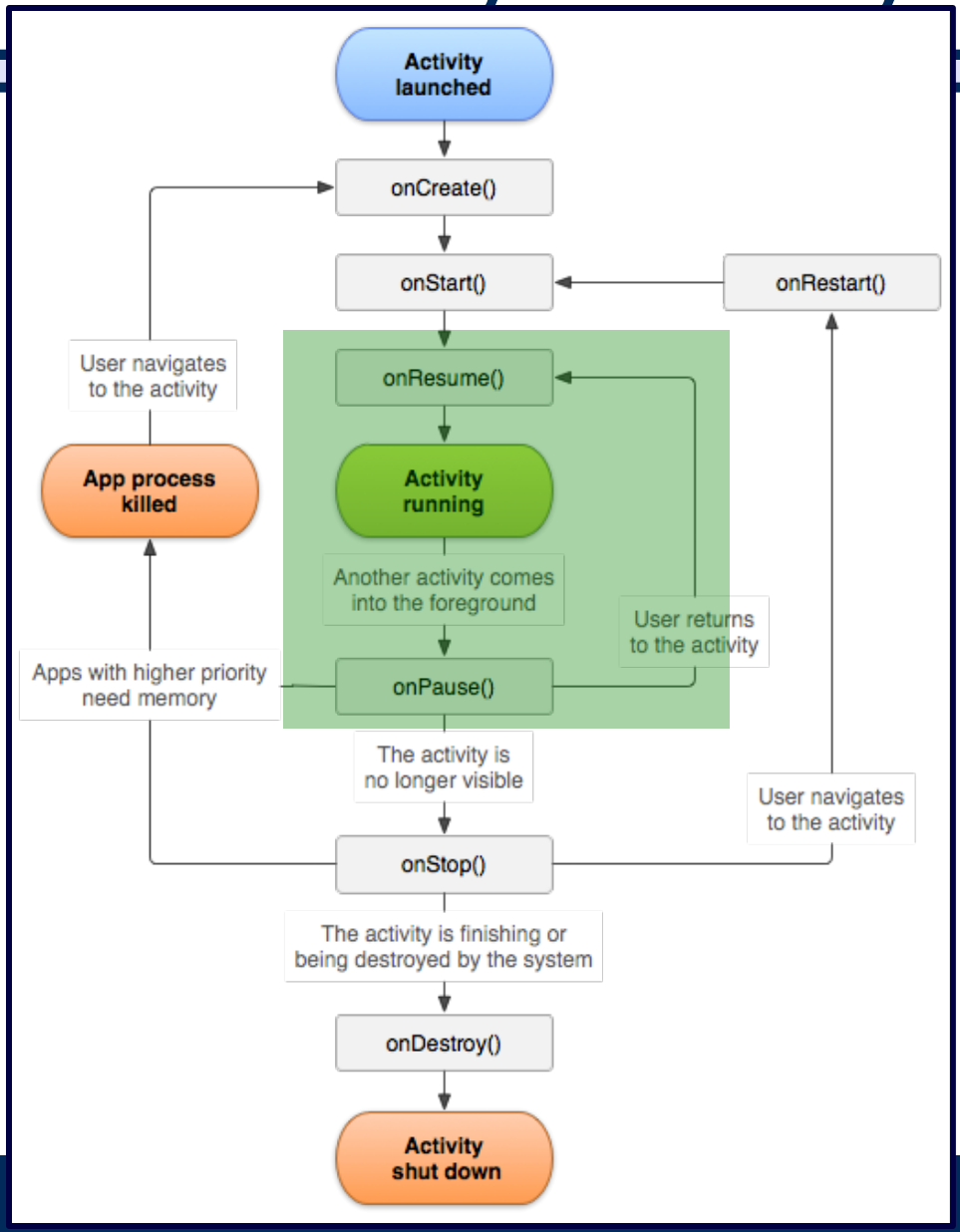
The Activity Life Cycle

The Entire Lifetime



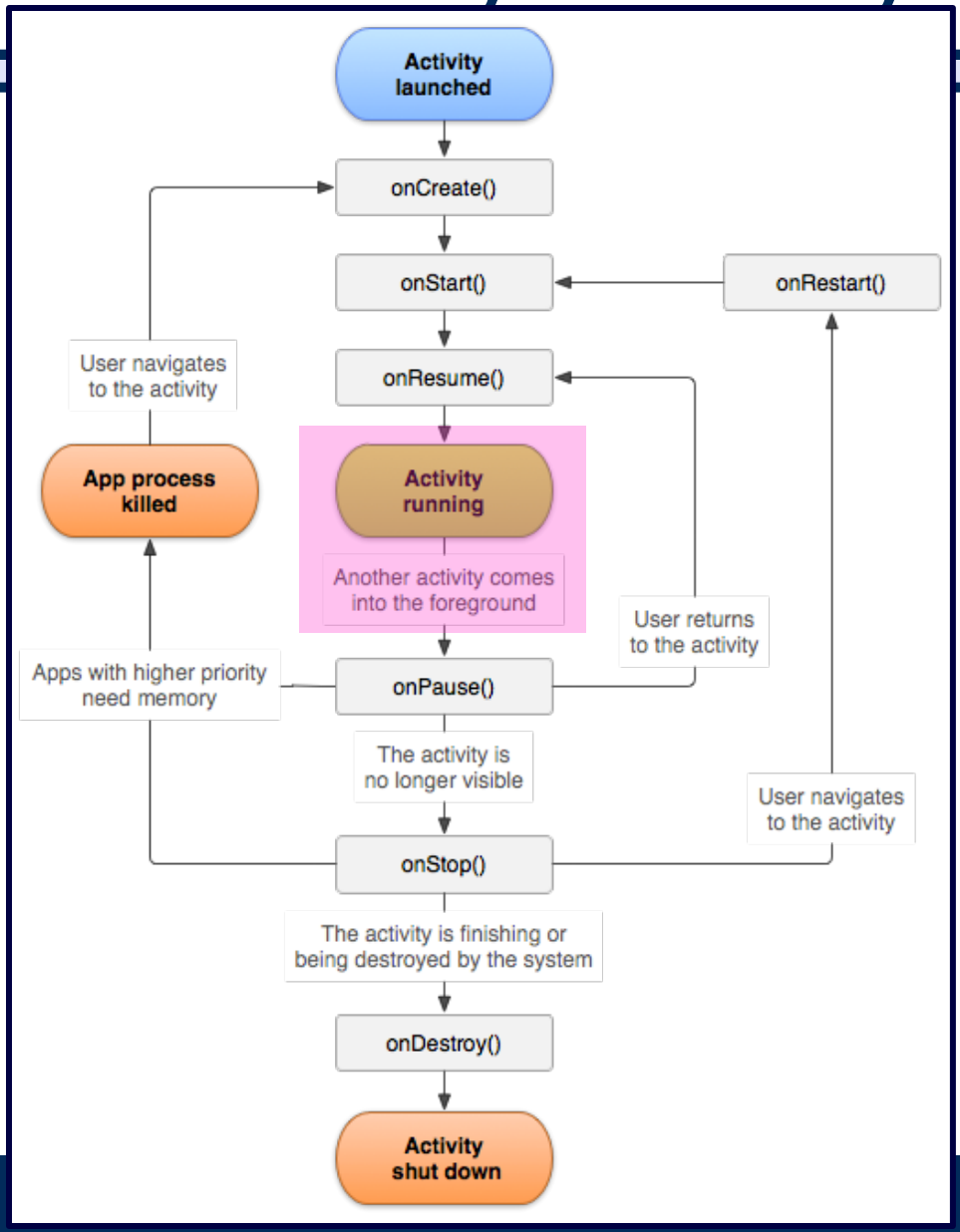
The Activity Life Cycle

The Visible Lifetime

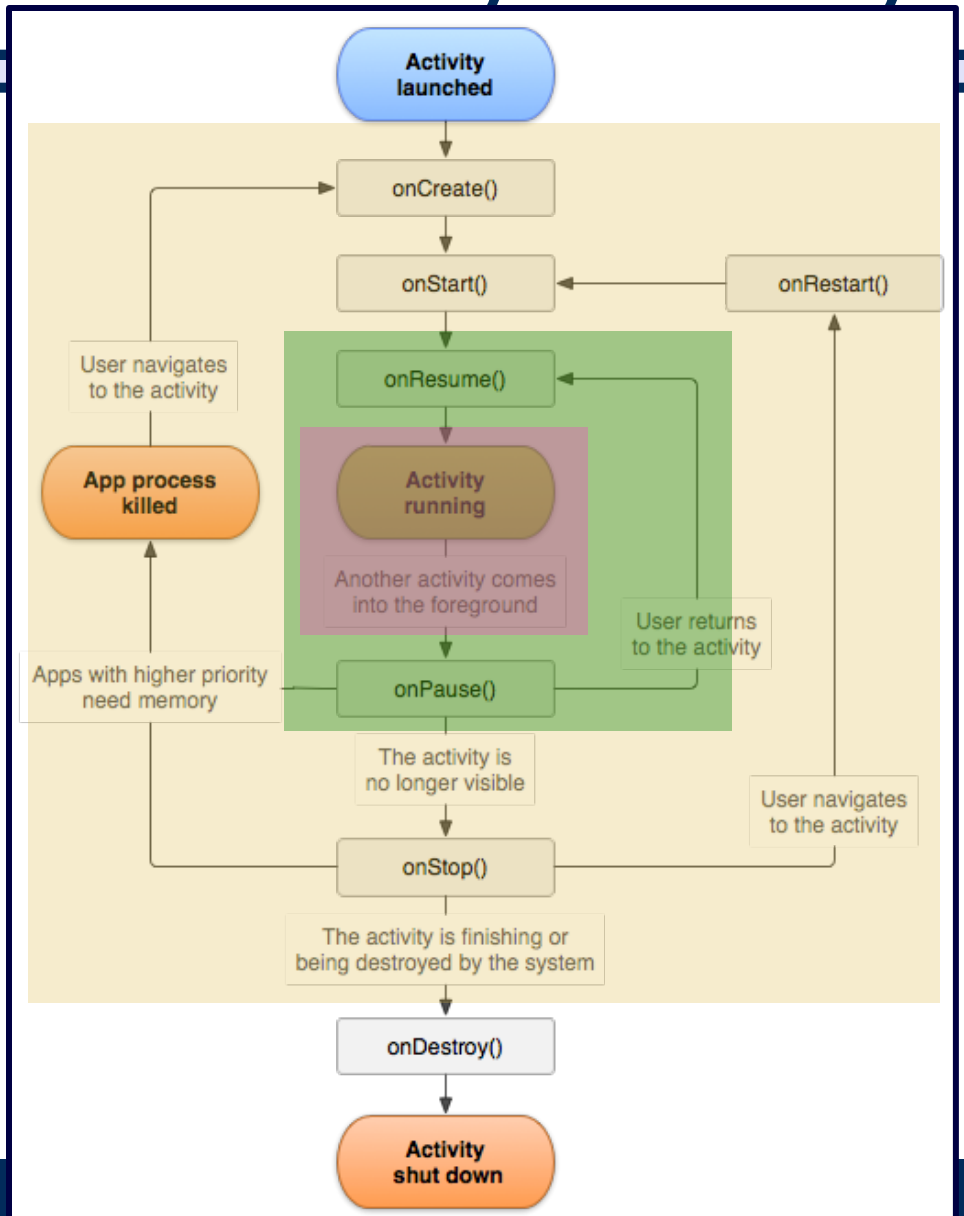


The Activity Life Cycle

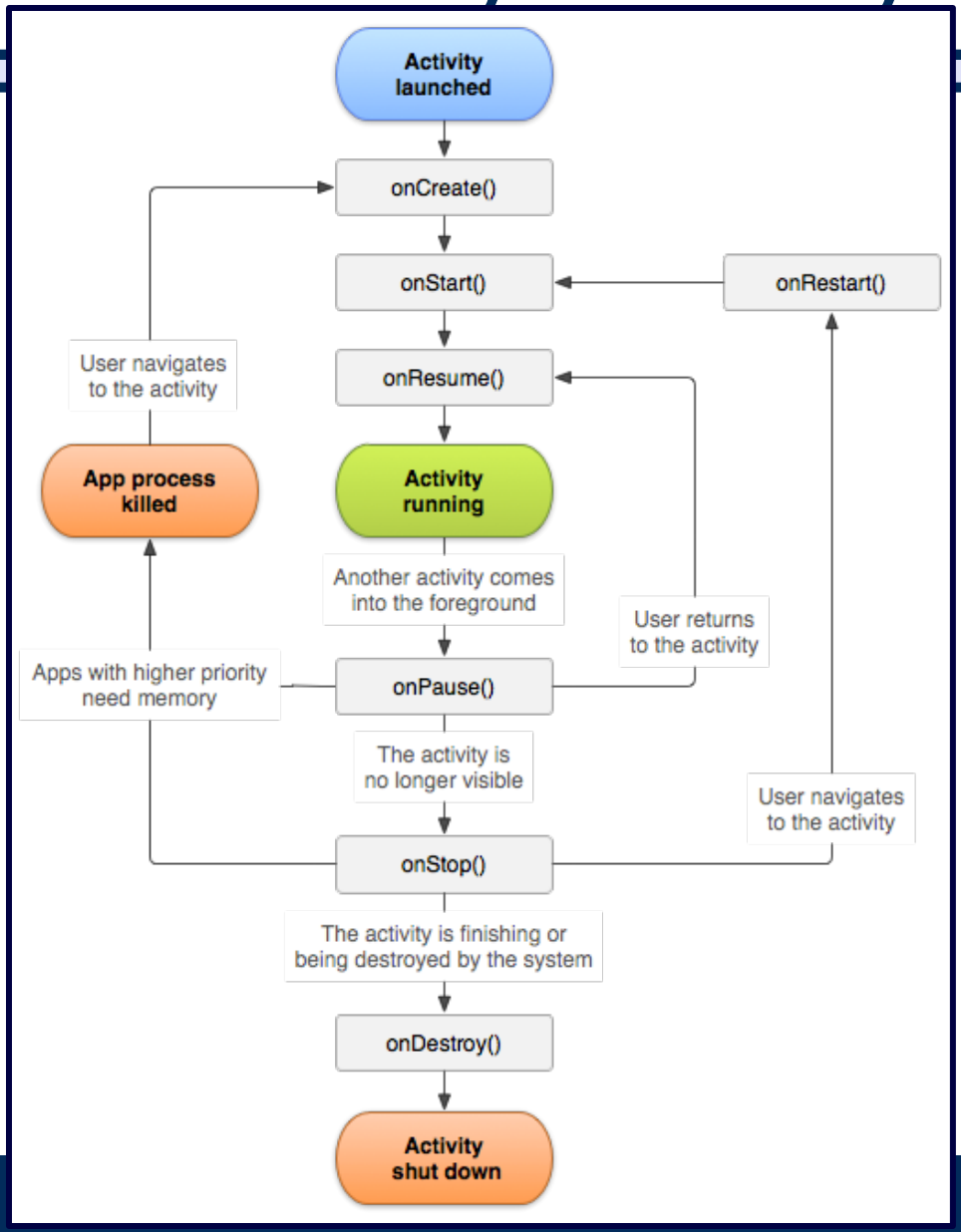
The Foreground Lifetime



The Activity Life Cycle



The Activity Life Cycle



Activity Lifecycle

- Android calls specific methods when the state changes
- You can override these to implement specific behavior

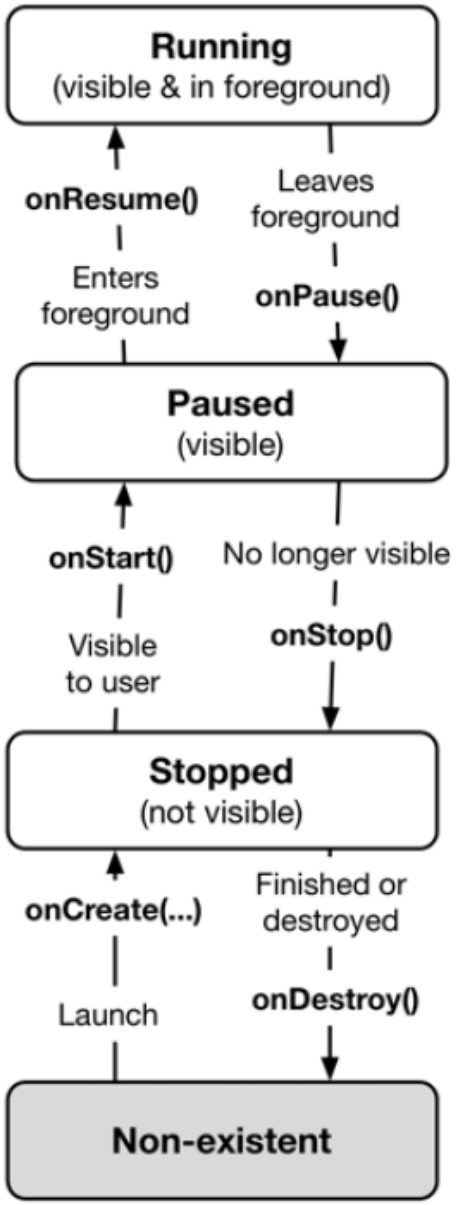


Figure 3.1 – Android Programming – The Big Nerd Ranch Guide 3rd Edition

Example – “onCreate”

```
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContent {
        ...
    }
  }
}
```

You must call the superclass implementation

This calls the composable to emit the corresponding screen

On Tap For Today

- Activity Life Cycle
- Logging
- Resources
- Practice

Log Messages

- Print messages during runtime

```
Log.type( String TAG, String msg );
```

Table 3.2 Log levels and functions

Log level	Function	Used for
ERROR	`Log.e(…)`	errors
WARNING	`Log.w(…)`	warnings
INFO	`Log.i(…)`	informational messages
DEBUG	`Log.d(…)`	debug output (may be filtered out)
VERBOSE	`Log.v(…)`	development only

TempConverter

- Let's log some lifecycles!

On Tap For Today

- Activity Life Cycle
- Logging
- Resources
- Practice

Resources

- What resources do we have?
  - drawables/
  - mipmap/
  - values/
    - colors
    - strings
    - themes

Resources

- Why do resource files exist?
  - Exist in one place → reference values
  - Values chosen at runtime based on current device configuration
- Able to support
  - Multiple languages
  - Multiple screen sizes / densities / resolutions

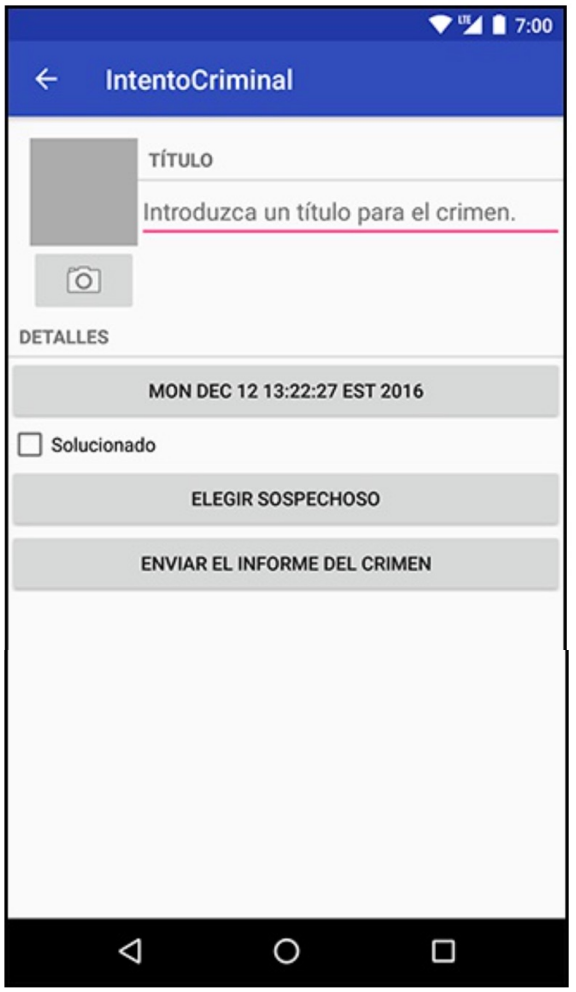
Resources

- Why do resource files exist?
  - Exist in one place → reference values
  - Values chosen at runtime based on current device configuration
- Able to support
  - Multiple languages
  - Multiple screen sizes / densities / resolutions

Localization

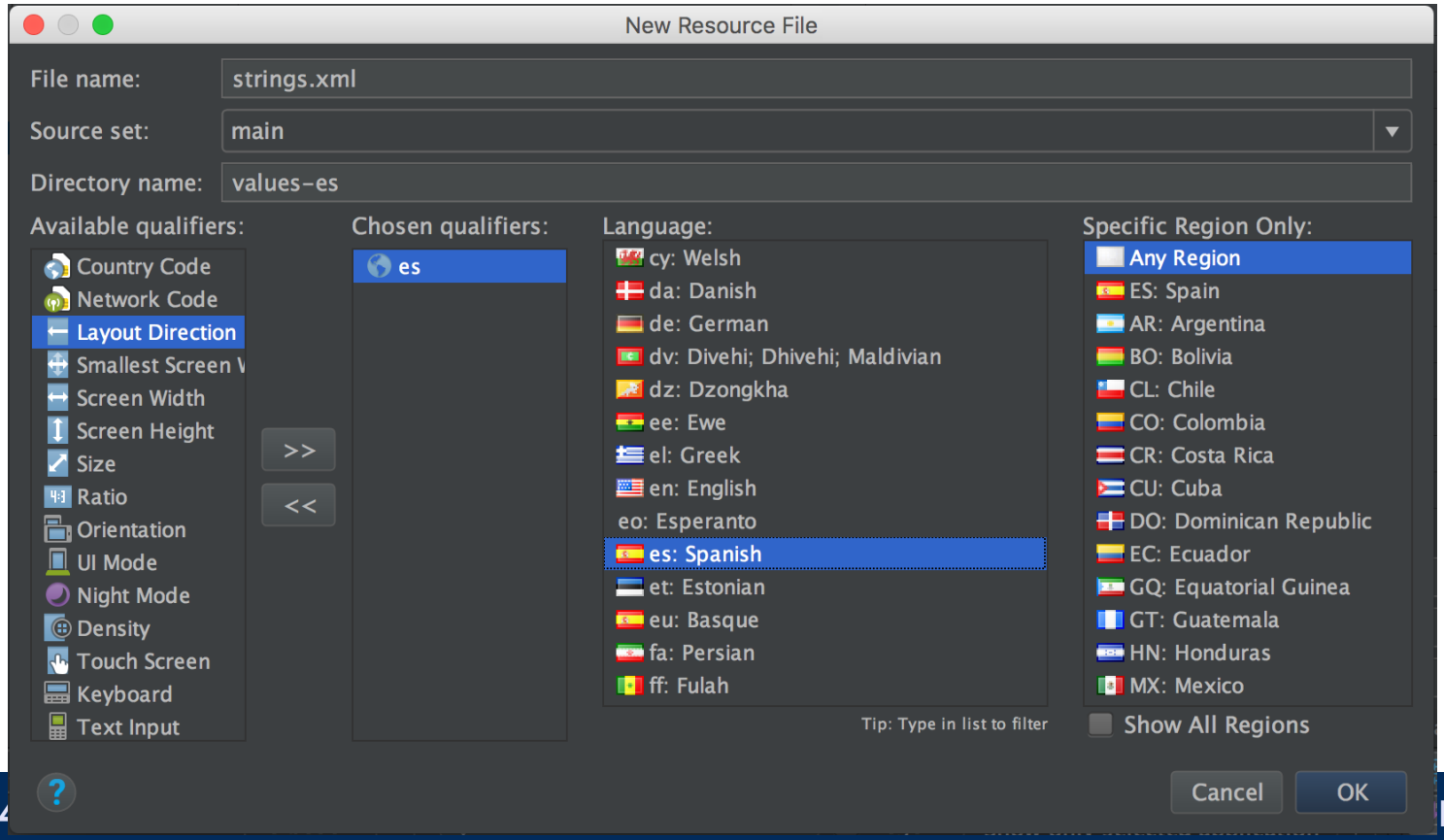
- Providing resources based on device's current language settings

Figure 18.1 IntentoCriminal



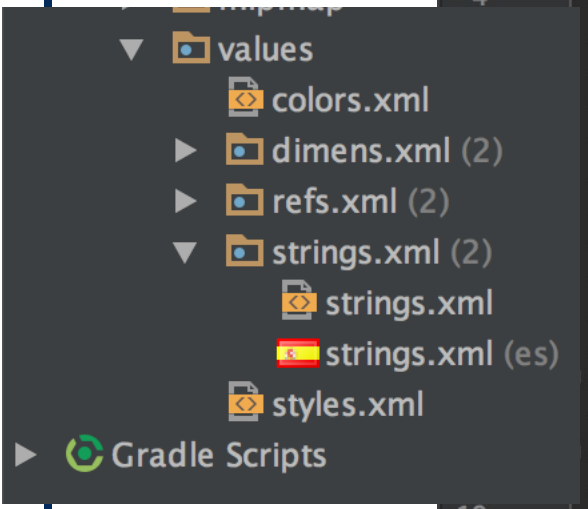
Creating the File

- Add a new resource
  - Choose “Locale” as qualifier



String Localization

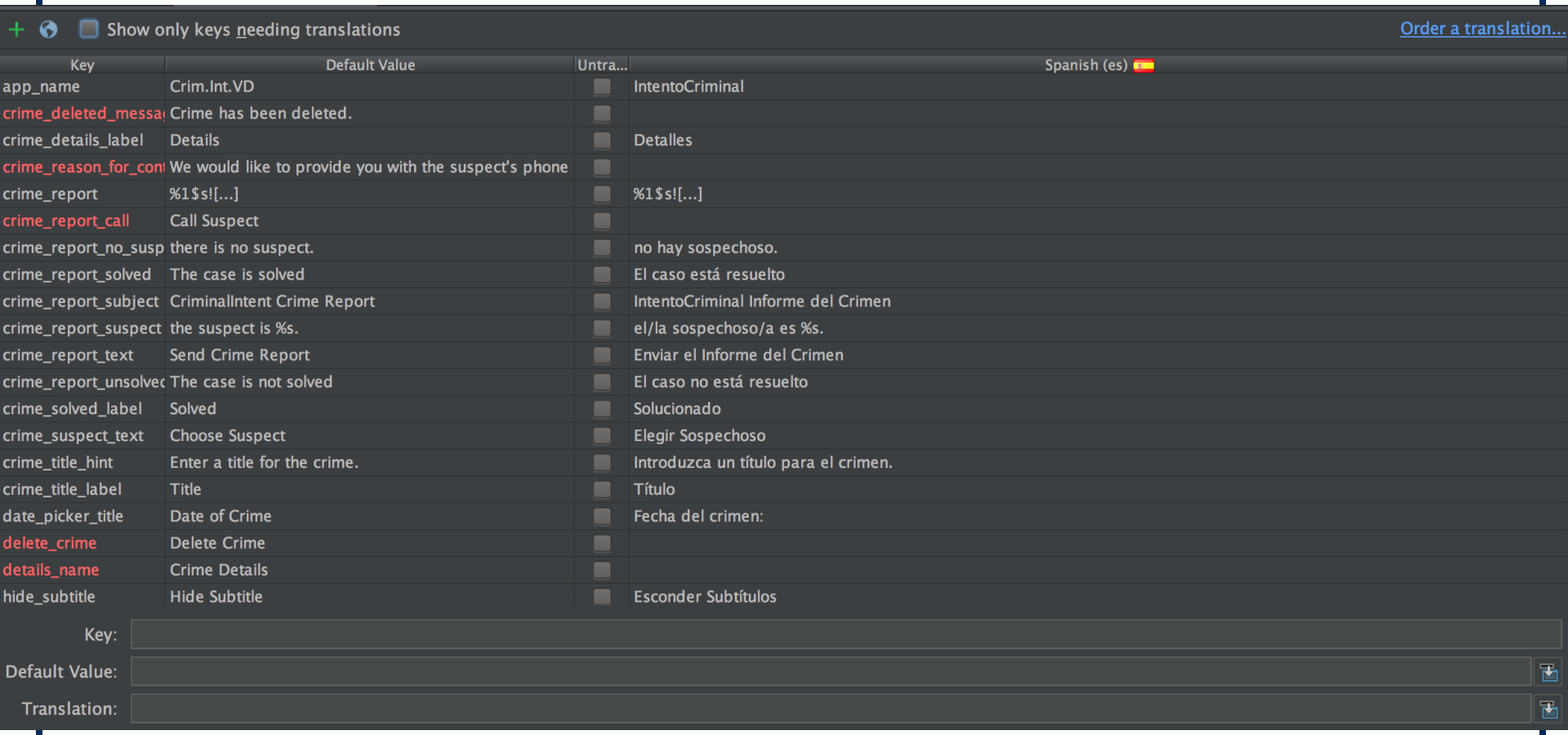
- Create a strings.xml resource file for each language



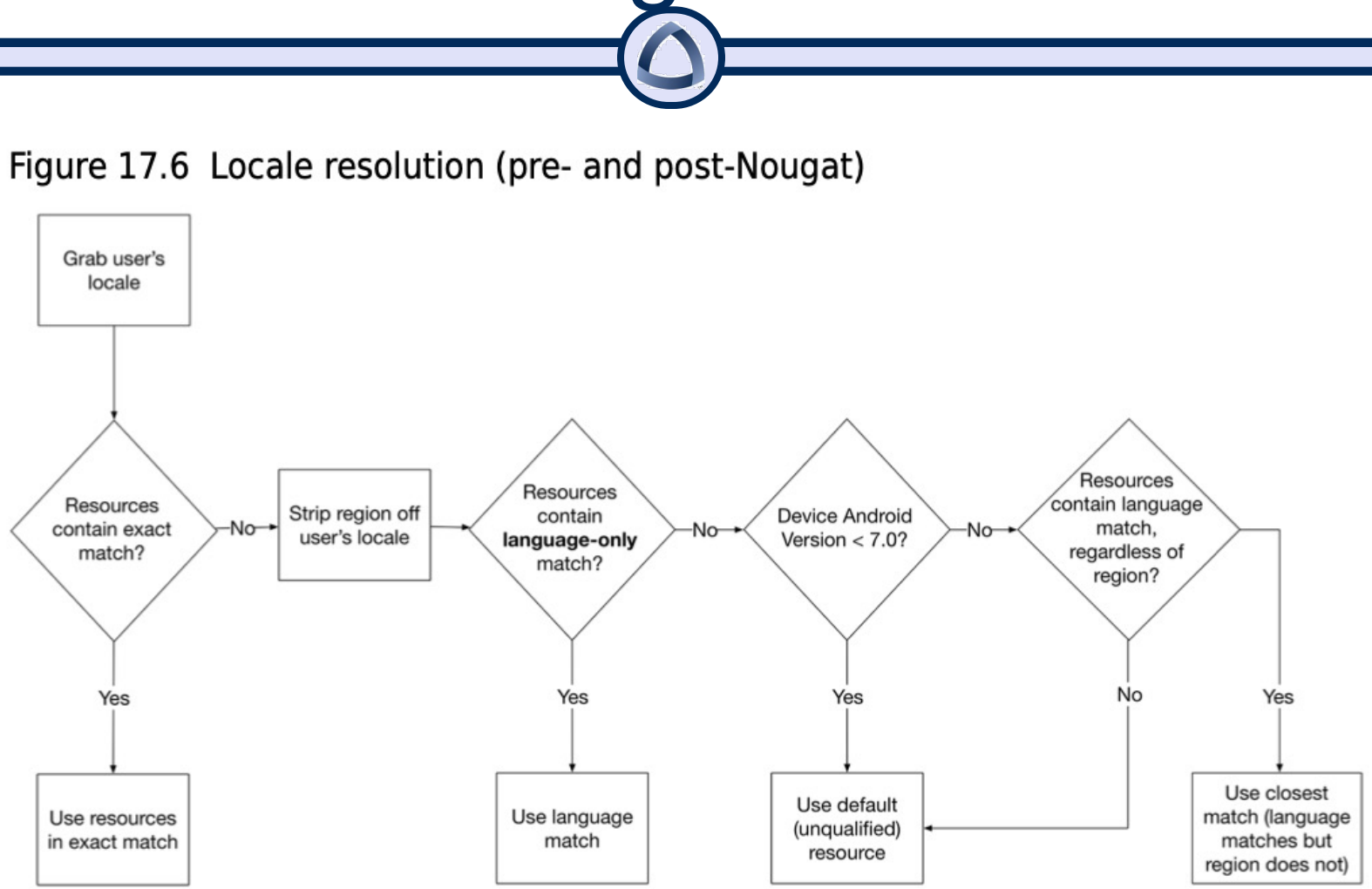
Edit translations for all locales in the translations editor.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">IntentoCriminal</string>
    <string name="crime_title_hint">Introduzca un título para el crimen.</string>
    <string name="crime_title_label">Título</string>
    <string name="crime_details_label">Detalles</string>
    <string name="crime_solved_label">Solucionado</string>
    <string name="date_picker_title">Fecha del crimen:</string>
    <string name="new_crime">Crimen Nuevo</string>
    <string name="show_subtitle">Mostrar Subtítulos</string>
    <string name="hide_subtitle">Esconder Subtítulos</string>
    <string name="subtitle_format">%1$s crímenes</string>
    <string name="crime_suspect_text">Elegir Sospechoso</string>
    <string name="crime_report_text">Enviar el Informe del Crimen</string>
    <string name="crime_report">%1$s!
        El crimen fue descubierto el %2$s. %3$s, y %4$s
    </string>
    <string name="crime_report_solved">El caso está resuelto</string>
    <string name="crime_report_unsolved">El caso no está resuelto</string>
    <string name="crime_report_no_suspect">no hay sospechoso.</string>
    <string name="crime_report_suspect">el/la sospechoso/a es %s.</string>
    <string name="crime_report_subject">IntentoCriminal Informe del Crimen</string>
    <string name="send_report">Enviar el informe del crimen a través de</string>
</resources>
```

Translation Editor



Choosing a Locale



Multiple Qualifiers

- Follows priority

1. mobile country code (MCC), optionally followed by mobile network code (MNC)
2. language code, optionally followed by region code
3. layout direction
4. smallest width
5. available width
6. available height
7. screen size
8. screen aspect
9. round screen (API level 23 and above)
10. screen orientation
11. UI mode
12. night mode
13. screen density (`dpi`)
14. touchscreen type
15. keyboard availability
16. primary text input method
17. navigation key availability
18. primary non-touch navigation method
19. API level

Resources

- Why do resource files exist?
  - Exist in one place → reference values
  - Values chosen at runtime based on current device configuration
- Able to support
  - Multiple languages
  - Multiple screen sizes / densities / resolutions

Supporting Multiple Screens

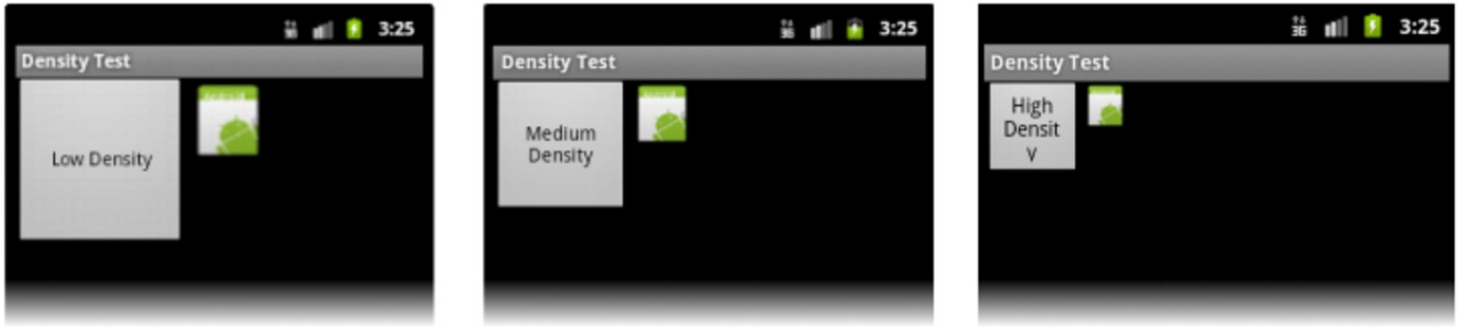


Figure 2. Example application without support for different densities, as shown on low, medium, and high-density screens.

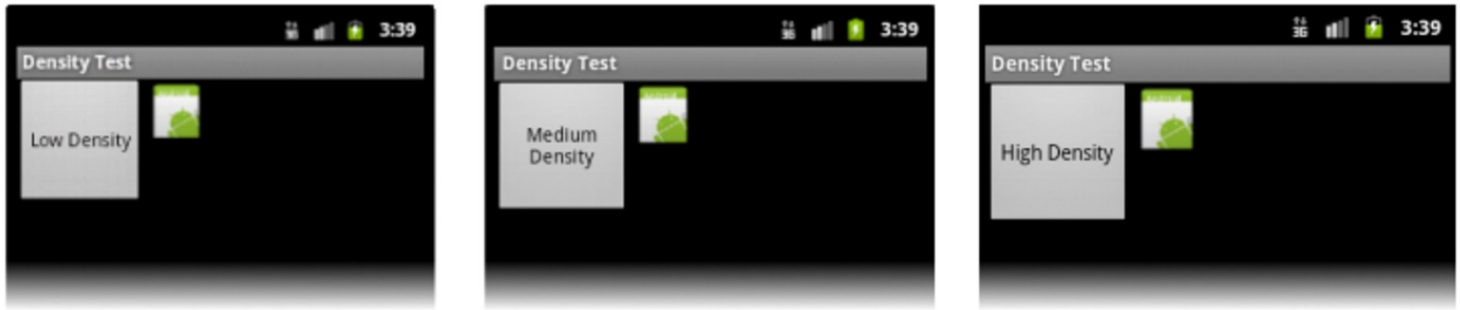


Figure 3. Example application with good support for different densities (it's density independent), as shown on low, medium, and high density screens.

http://developer.android.com/guide/practices/screens_support.html

Pixels / dp / sp / in / mm

- For height/width/padding, can set fixed size in different units
  - Pixels: one color unit on the screen
  - dp (dip): density-independent pixel
  - sp (sip): scale-independent pixel
  - in: inches
  - mm: millimeters

Screen Statistics

- **Screen Size**: Actual physical size along diagonal in inches or millimeters
- **Resolution**: total number of physical pixels on the screen
- **Screen Density**: quantity of pixels in a given area of the screen – referred to as dpi (dots per inch) or ppi (pixels per inch)
  - low, medium, high, extra-high, extra-extra-high, extra-extra-extra-high (ldpi, mdpi, hdpi, xhdpi, xxhdpi, xxxhdpi)

Screen Density

- ldpi = 120dpi (0.75X)
- mdpi = 160dpi (baseline)
- hdpi = 240dpi (1.5X)
- xhdpi = 320dpi (2X)
- xxhdpi = 480dpi (3X)
- xxxhpi = 640dpi (4X)

dp

- Density-independent pixel
  - A virtual pixel unit
  - Equivalent to one physical pixel on a 160dpi screen
  - At runtime, the system converts from dp to physical pixels using equation

```
px = dp * (dpi / 160)
```

  - Ex:
    - On a 240dpi screen, 1 dp = 1.5 pixels
    - On a 515dpi screen, 1 dp = 3.2 pixels

sp

- Scale-independent pixel
  - Same properties as dp
  - Scaled with user's font size preference

When to use dp or sp?

- Use sp for fonts/text
- Use dp for everything else

Configuration Changes

- When the device configuration changes (such as changing language) this causes the activity to be destroyed and re-created!

- See this by generating “log” messages in the activity callback methods

Configuration Changes

- When the device configuration changes (such as changing language) this causes the activity to be destroyed and re-created!
- See this by generating “log” messages in the activity callback methods
- Why?

On Tap For Today

- Activity Life Cycle
- Logging
- Resources
- Practice

To Do For Next Time

- Lab02 due Friday
- Final Project Proposal due Friday
- Kotlin Strings quiz due by Friday's class