NDK
native API crunch

Android SDK

Lecture 2

Android Native Development Kit

4 March 2014

NDK
native API crunch

Applications

Activities

Services

Intents

Broadcast Receivers

Content Providers

Keywords

Outline

Android Manifest

- *xml* file
- In the root of the app directory
- Describes application components and resources
  - Application name and Java package (unique)
  - Activities, Services, Broadcast Receivers, ContentProviders
  - Start activity
  - Permissions
  - Libraries
  - Minimum API level

Permissions

- Determine what resources and APIs an application can access
- Meant to provide security through sandboxing
- Declared in the Manifest
  - *<uses-permission android:name="android.permission.RECEIVE_SMS" />*
- Permission is asked at install time: all or nothing
- Can not be granted afterwards
- Control who can access your components and resources
  - Start activity, start or bind service, send broadcasts, access data in Content Provider
  - URI permissions

Resources

- *res/* directory
- Each resource type in a different subdirectory with specific name
  - *drawable/*, *layout/*, *values/*, *menu/*, *xml/*, etc.
  - The default resources
- For different configurations we may need different resources
  - Bigger screen -> different layout
  - Different language -> different strings
  - Subdirectory for each alternative set of resources
  - *<resources_name>-<config_qualifier>*
  - *drawable-hdpi/* for High Density Screens
  - Resource chosen at runtime based on the configuration
- An ID is generated for each resource name in *gen/*

Layouts

- Resources from *res/layouts/*
- Describe the UI of an activity or part of the UI
- *res/layout/filename.xml*
  - filename is used as resource ID
- UI elements
- Can be edited as an xml or using the tools provided
- Easy to learn, hard to master

Drawables

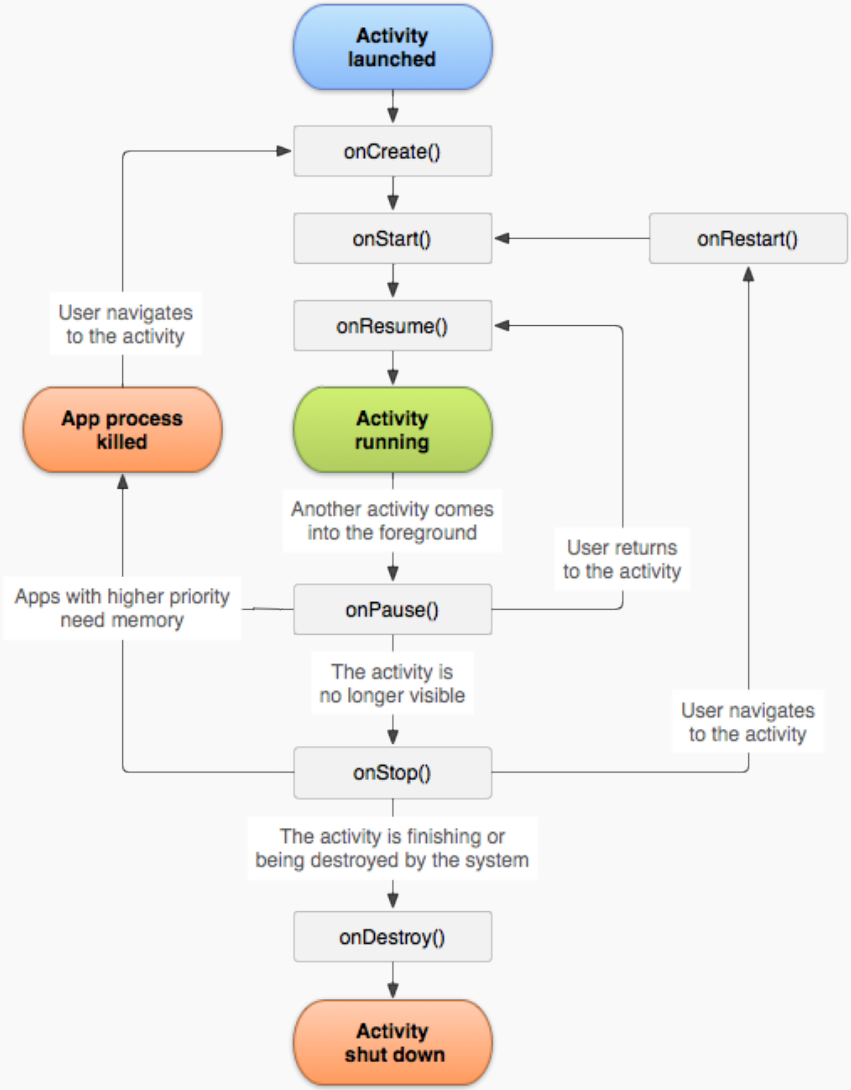
- Resources from *res/drawables/*
- Element that can be drawn on the screen
- Can be images (.png, .jpg, or .gif) or xmls
- xmls describe how an UI element reacts to input (pressed, focused)
- xmls point to images
- Visual feedback for interaction

Outline

Activity

- Application component
- User interface window, provide user interaction
- Require a layout
- Can only draw and change UI from the Looper thread
  - Computationally intensive or wait based tasks on separate threads
- An application may include multiple activities
  - Only one is the main activity
  - Activities can start each other -> the previous one is stopped
  - Activity stack ("*back stack*")
  - Back -> activity destroyed and previous one resumed

Activity Life cycle

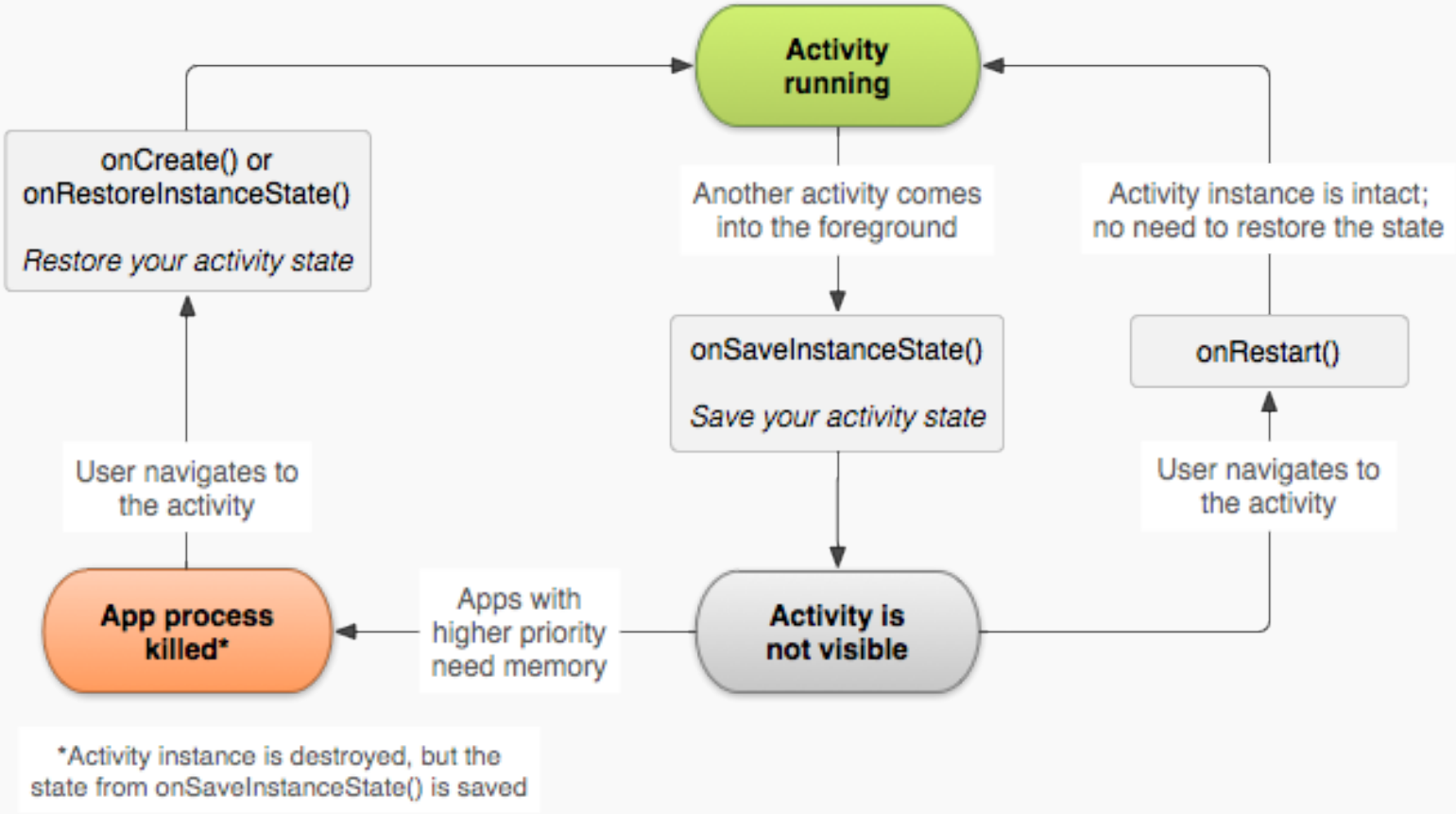


Source: http://developer.android.com

Activity Life cycle

- Activities can be killed after *onPause()*, *onStop()* in low memory situations
  - The activity state (objects) are lost
  - Can preserve state by saving objects
  - User interaction can be saved and restored
  - Callback *onSaveInstanceState()*
    - Save information in a Bundle
  - *onCreate()*, *onRestoreInstanceState()*
    - Restore the activity state
  - Threads can be stopped graciously
    - In *onPause()* threads should be signaled to stop

Save Activity State



Source: http://developer.android.com

UI

- UI is a hierarchy of views
- *View*: rectangular space, provides user interaction
- Buttons, Lists, Fragments, Images, Text Boxes
- Callbacks for actions
- A *ViewGroup* includes other View or ViewGroup objects
- Classes can be extended to provide more complex views
- Adapters allow for more complex data types to displayed

Outline

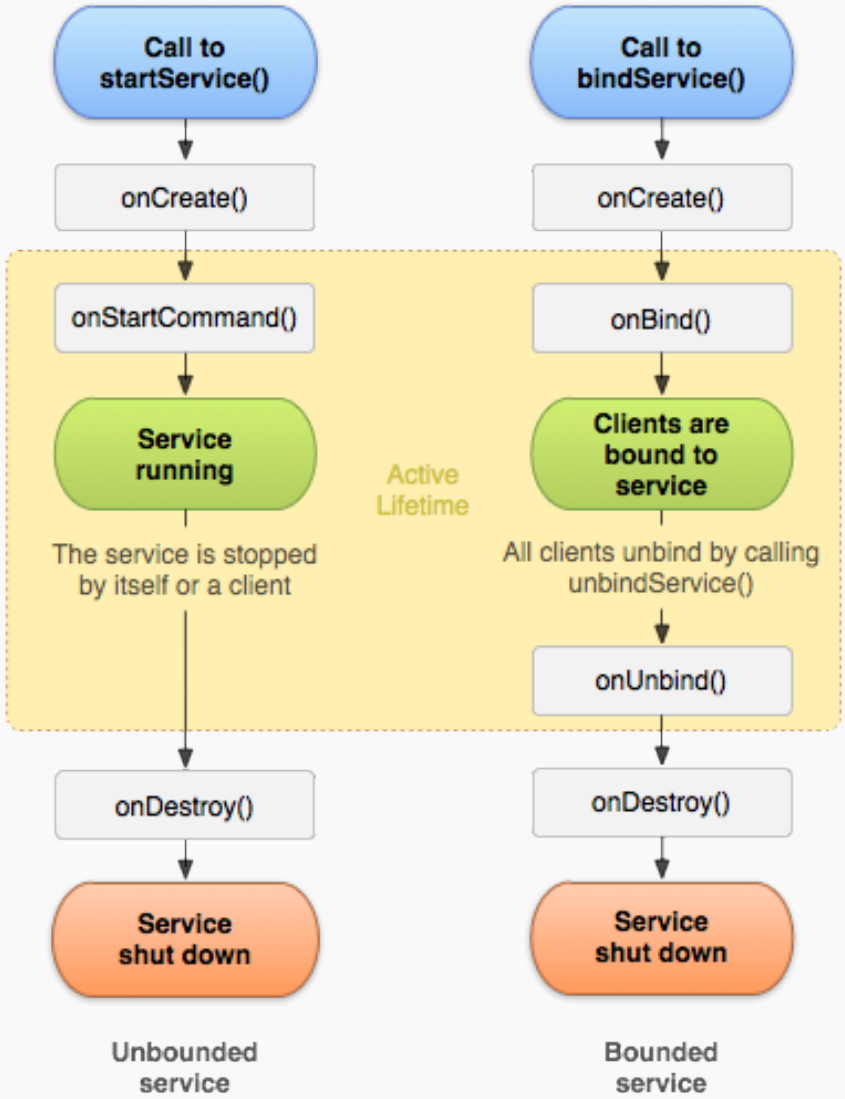
Services

- Performs operations in the background
- Does not provide a UI
- Continues to run even if another application is in foreground
- Is able to perform network transactions, file I/O operations, interact with content providers, etc.
- A service runs in the main thread of the hosting process
  - A separate thread should be created if the service performs CPU intensive or blocking operations
- Can be started by any application using an Intent
  - Block access from other apps by declaring the service as private in the manifest

Types of services

- Started
  - An application component calls *startService()*
  - Performs a single operation, then stops itself and does not return a result to the caller
  - Runs even if the caller component is destroyed
- Bound
  - An application component binds to it by calling *bindService()*
  - Provides a client-server interface - send requests, return results
  - Runs as long as the application component is bound to it
  - Multiple components can bind to a service at once
  - Service destroyed after all components unbind

Service Life Cycle



Source: http://developer.android.com

Outline

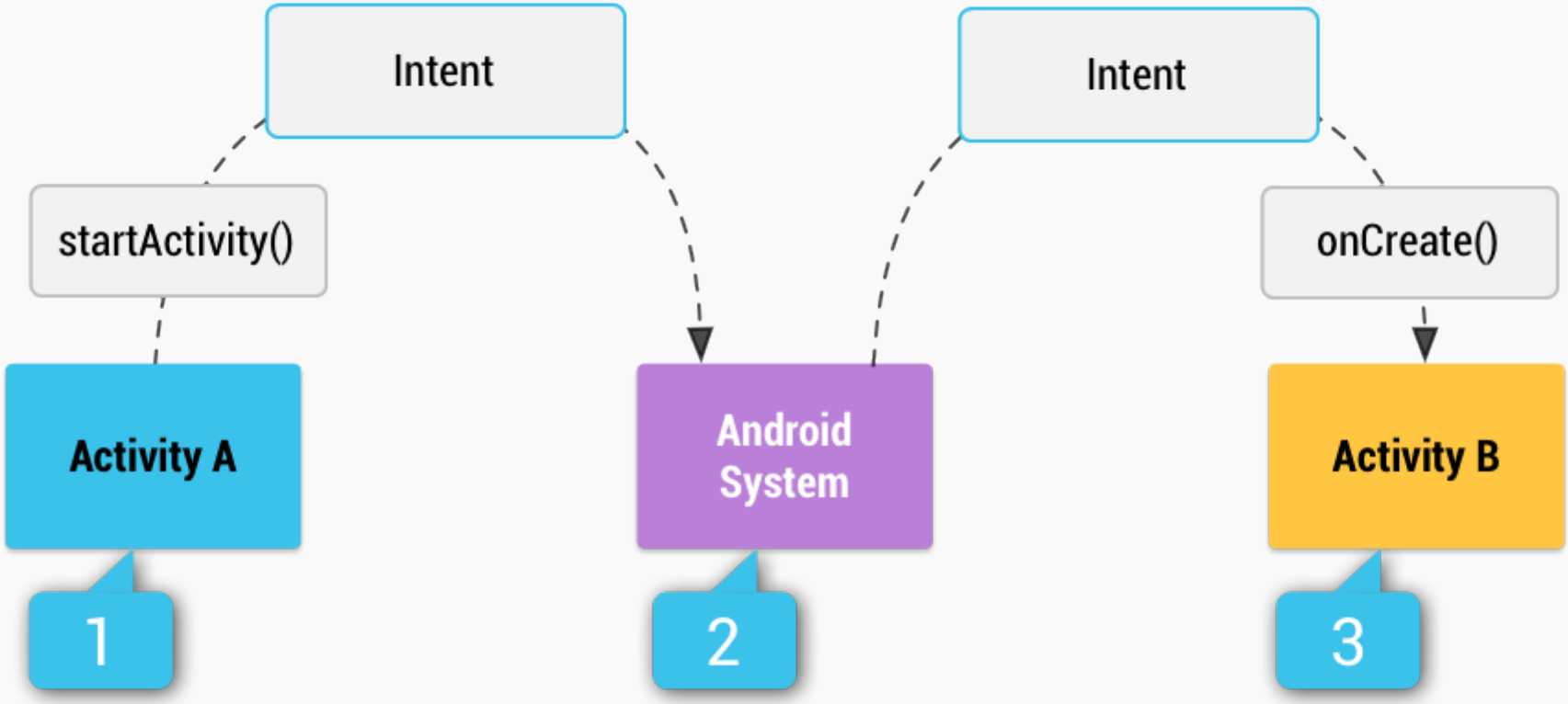
Intents

- An object used for delivering a message
- Used for
  - Starting an activity
  - Starting or binding a service
  - Delivering a broadcast message
- Includes component name, action and data
- Intent filters
  - Declare the types of intents that a component can receive
  - Specified in the manifest - *<intent-filter>*
  - *<action>*, *<data>*

Intents

- Two types of intents
  - Explicit intents
    - Specify exactly which component to start (the class name)
    - Typically used to start components in your own app
    - Will be delivered even if there is no intent filter declared
  - Implicit intents
    - Do not specify the exact component
    - Declare a general action to be performed
    - The Android system finds the appropriate component
    - Compares the intent to the intent filters in the manifest of the apps

Implicit Intents



Source: http://developer.android.com

Outline

Broadcast Receivers

- Responds to system-wide broadcast announcements
- The system generates many broadcasts
  - Example: battery is low, screen has turned off, etc.
- Apps can generate broadcasts - send an announcement for other apps
- No UI, may create a notification in the status bar to alert the user
- The receiver lets other components perform the work based on the event

Broadcast Receivers

- Each broadcast is delivered as an *Intent*
  - Intent passed to *startBroadcast()* or *startOrderedBroadcast()*
- Local broadcasts using *LocalBroadcastManager*
  - More efficient
  - Data does not leave the app
  - Other apps cannot send the broadcast - no security holes
- Register a receiver in two ways
  - Statically in the manifest using the *<receiver>* tag
  - Dynamically using *Context.registerReceiver()*

- Two types of broadcasts
  - Normal broadcasts
    - Completely Asynchronous
    - All receivers run in an undefined order
  - Ordered broadcasts
    - Delivered to one receiver at a time
    - Each receiver executes and may propagate the result to the next or abort the broadcast
    - The order is determined using the *android:priority* in the <*intent-filter*> of the receiver

Outline

Content Providers

- Provide access to a repository of data
- To access a provider you have to request specific permissions (in the manifest)
  - *<uses-permission android:name="android.permission.READ_USER_DICTIONARY">*
- Two ways of storing data
  - File data - audio, video, photos
  - Structured data - database, array, etc.
    - Form compatible with tables of rows and columns
    - Usually a SQLite database

Content Providers

- Interface for accessing data in one process from another process
  - Provider and provider client
  - The application that owns the data includes the provider
  - The client application owns the provider client
- Access data using a *ContentResolver* client object
  - Its methods provide CRUD (create, retrieve, update, delete) functions
  - Calls the methods with the same name in the *ContentProvider* object

Content URIs

- Identify data in the provider
- Include a symbolic name for the provider (*authority*) and a name for the table (*path*)
  - Example: *content://user_dictionary/words*
  - The *ContentResolver* uses the *authority* for identifying the provider
  - From a system table with all known providers
  - The *ContentResolver* sends a query to the provider
  - The *ContentProvider* uses the *path* to identify the table

Outline

NDK
native API crunch

- Manifest file
- Permissions
- Resources
- Layouts
- Drawables
- Activity

- Service
- Intent
- Broadcast Receiver
- Content Provider
- Content URI