# Android Activity Lifecycle

22 March 2011
CMPT166
Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

# Outline for today

- Component-based applications
  - Activity, Service, BroadcastReceiver, ContentProvider
- Android Activity: life cycle, states
  - Life cycle methods
  - Saving persistent / transient state
- Android Views (widgets):
  - Widgets and layouts
  - XML configuration, layout editor

# Getting started with Android

- Eclipse IDE for Java
- Android SDK starter package
- ADT plugin for Eclipse
- From plugin, add Android 1.6 platform
  - Could also develop for 1.5, 2.0, 2.1, etc.
  - Setup an emulator instance(virtual phone)
- Try the “Hello World!” tutorial
  - Run/debug on the emulator
  - Run/debug on actual phone via USB

# Component-based apps

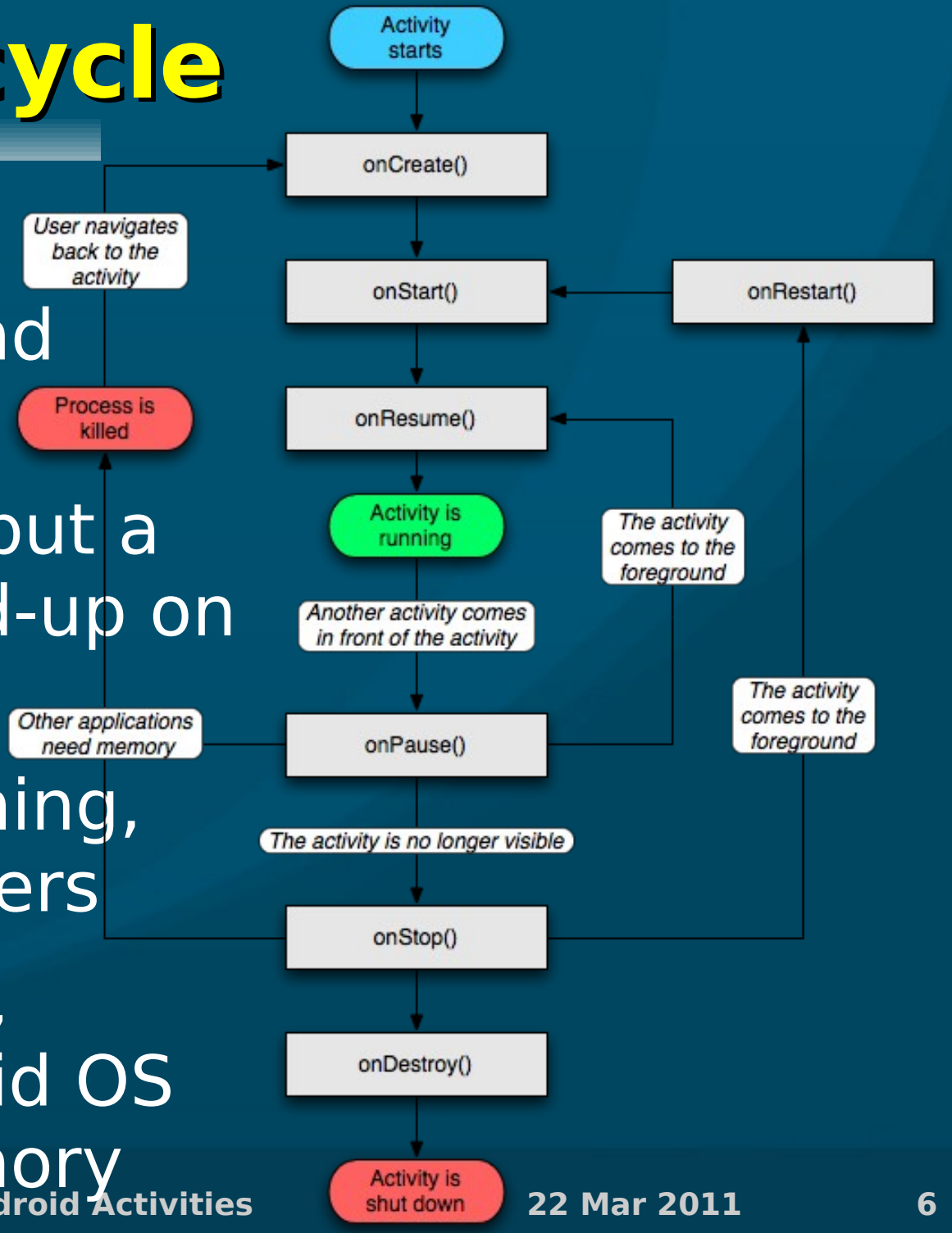
- No single entry point (i.e., main())
  - Instead, subclass an Android class and override certain methods ("hooks")
- Other apps can use parts of your app
  - e.g., use Browser to request a web page
  - e.g., search in Contacts for a phone number
- Android can resume your app if crashed
  - It can also kill your app if out of memory
  - So save/load state and be prepared to die at (nearly) any time

# Android app components

- Activity: present UI for one interactive task
  - e.g.: get username+password, display map
- Service: background task, often w/o UI
  - e.g.: play music, fetch file over network
- Broadcast Receiver: respond to announcements
  - e.g.: if timezone changes, battery low, etc.
- Content Provider: access/query a datastore
  - e.g.,: music library, student database, etc.
- We will focus on Activities (simplest)

# Activity life cycle

- Four states:
- Active: running and in foreground
- Paused: running, but a dialog has popped-up on top of it
- Stopped: still running, but hidden by others
- Dead: terminated, perhaps by Android OS when low on memory



# Life cycle methods

- Activity exists from onCreate to onDestroy():
  - Initial setup and final tear down of resources
- Activity is visible from onStart to onStop():
  - onRestart() also called when return to fore
- In foreground from onResume to onPause():
  - In foreground means accepting user input
  - onPause: commit unsaved changes, etc.
- A paused activity might be destroyed before it ever resumes!

# Saving state

- Persistent state can be saved in onPause()
  - e.g. draft of a message being composed
  - Write to storage: preferences, SQL database, app-specific file, or SD card
- Transient state: use onSaveInstanceState()
  - e.g. how user filled out form before “submit”
  - Save in a Bundle, which is passed to both onCreate() and onRestoreInstanceState()
  - Use this, e.g., to fill out the form again when user goes “Back” to this activity

# Views (widgets)

- View is Android's widget class (JComponent)
- Subclasses: Button, TextView (label), EditText (text area), Spinner (pull-down list),
  - Or make your own subclass to customize!
- ViewGroups (layout managers) LinearLayout Relative, GridView, TableLayout, TabHost, ...
- Within onCreate(), call setContentView() to declare the activity's main View (panel):
  - **TextView tv = new TextView( this );**
  - **setContentView( tv );**

# “Hello, Android!” tutorial

- Only one activity: HelloAndroid
  - Package: domain name, application name
- onCreate(): called when activity is run
- The parameter is the saved state Bundle:
  - Use this to restore transient state if desired
  - Also pass up to superclass' onCreate()
- Create a view (widget): TextView
  - Set the text to “Hello, Android!”
  - Set as the main view for the activity
  - Use a layout if want > 1 widget

# XML layout

- Laying out widgets can be complex in code
- You may use an XML config file for layout:
  - Create a file under res/layout/*.xml
  - XML is like HTML: <tag> ... </tag>
- Specify layouts, widgets, font/colour/text/...
  - Eclipse ADT has a WYSIWYG layout editor!
- XML gets compiled into an object (R class)
  - R is auto-generated; don't edit directly!
  - Refer to R.layout.myLayout
    (follows name of the XML file)