UIs and Intents

Steven R. Bagley

Previously…

- Looked at how to install and setup the Android SDK
- Created a very simple ‘app’

Today

- Look at the Android UI library
- See exactly how Intents work and how we can start other Activities

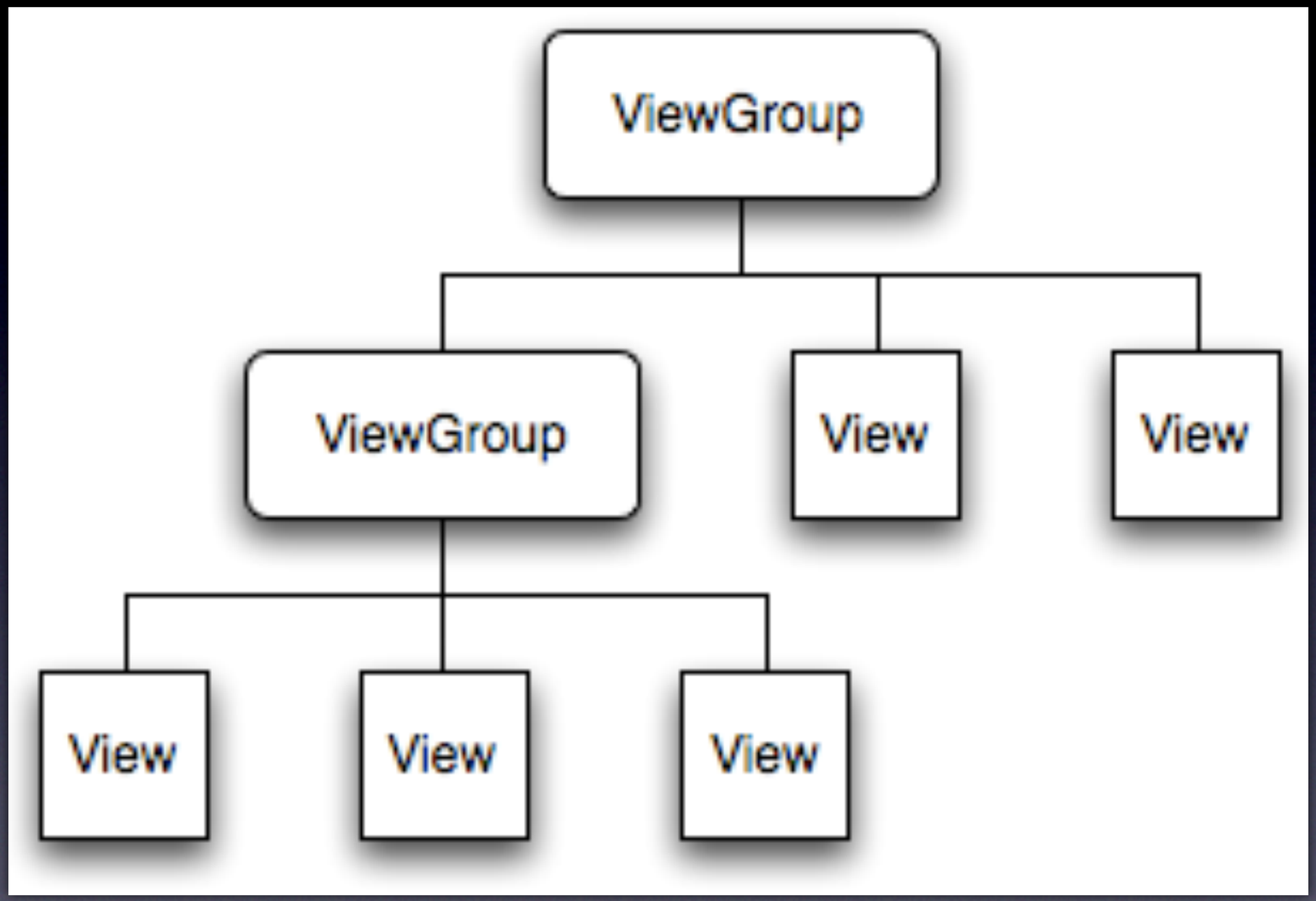
Android UI

- An Activity has a window associated with it
- Usually, full screen but can hover over another activity
- Within the window there is a hierarchy of `View` objects
- Set with `setContentView()`
- Usually specified in the XML

View Hierarchy

- Three types of `View` subclasses
- Ones that display something
- `ViewGroup`s which layout a collection of subviews
- Various layout types available — can specify in the XML
- Also `Widget`s — `View`s that allow interaction

Look at the widget SDK page


ViewGroup
ViewGroup
View
View
View
View
View

Layout Types

Layout Type	Description
`FrameLayout`	Simplest, contains a single object
`LinearLayout`	Aligns all children in a single direction, based on the `orientation` attribute
`TableLayout`	Positions children into rows and columns
`RelativeLayout`	Lets the child views specify their position relative to the parent view or to each other
`ScrollView`	A vertically scrolling view, like `FrameLayout` only contains a single element (e.g. a `LinearLayout`)

Non exhaustive

See http://developer.android.com/guide/topics/ui/layout-objects.html for more details

Manipulating Views

- We can alter the view hierarchy as the program runs
- `ViewGroup` provides methods to add (`addView()`) or remove (`removeView()`) children
- Need to either keep a reference to it or call `setId()` on the view so we can find it later

Multiple Activities

- One Activity is fine for some apps
- But often we'll need more than one
- Already noted that apps contain a stack of Activities
- How do we create and display another Activity?

Creating Activities

- Need another sub-class of `Activity`
- XML file for the layout…
- Don’t forget to add details of the application to the manifest
- But how to create and display the `Activity`?

Intents

- Don't just instantiate the Activity sub-class
- Already noted that Android works by passing `Intent` objects about
- Intent is used to describe an operation
  - Action and the data to operate on (as a URI)
- Allows for runtime binding

Starting an Activity

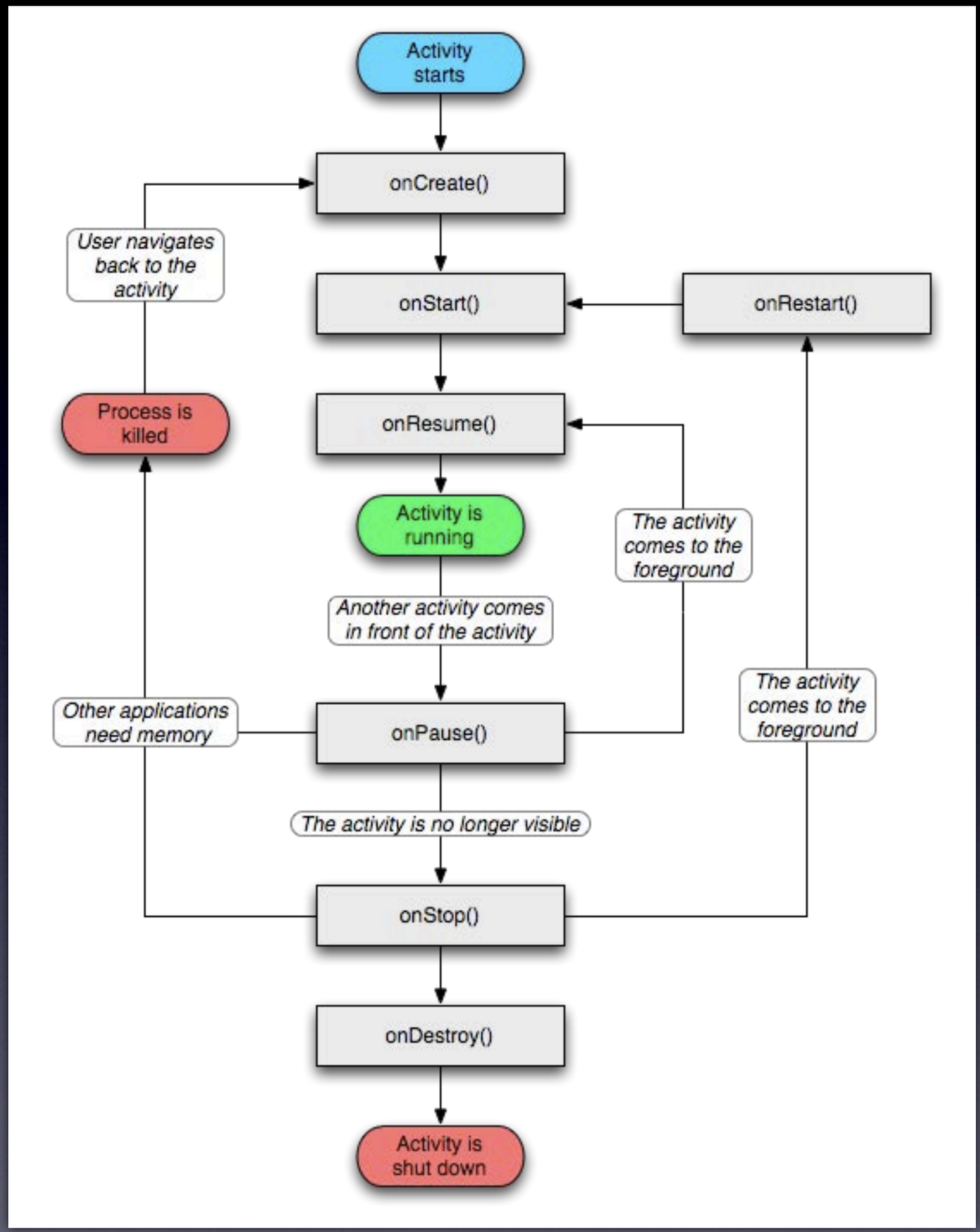
- Create a new `Intent` object
- Specify what you want to send it to
  - Either implicitly, or explicitly
- And the data you want to send it…
- Pass the `Intent` object to `Context.startActivity()`
- New Activity then started

Finishing Activity

- The called Activity can return to the original one
- By calling the method `finish()`

Lifecycle

- Remember the lifecycle…
- When a new Activity starts, the first is no longer running
- How does the start of the new Activity interleave with the finish of the previous?
- Let's test it…



Can do this by logging when the various methods are called from the two Activities are called then seeing the overlap in the logfile

Obtaining results

- `Context.startActivity()` doesn't allow the Activity to return a result
- If we want to return a result then we need to use `Activity.startActivityForResult()`
- Still takes an `Intent` object, but also a numerical request code

Results

- Returns an integer result code, set with `setResult()`
- And `finish()` to end the Activity
- `onActivityResult()` then called on the calling `Activity`

Returning results

- If you need to return more data you can package it up in an `Intent`
- Activity creates an `Intent` object containing the result
- Calls `setResult()` to return the Intent
- `Intent` object then passed to `onActivityResult()`