Accelerating Information Technology Innovation

http://aiti.mit.edu

Cali, Colombia
Summer 2012
Lesson 4 – Android Activities

Android Activities

- Apps are composed of Activities
- Just like a Website contains many web pages, an Android app contains many Activities
- Each Activity is a "page", or user interface view
- Activities are interconnected with one another, just like web pages contain navigation from one web page to another
- Each Activity contains a Layout View with specific view objects

MIT AITI

Multiple Activities

- An android application may consist of one or multiple Activity objects

- Only one activity can be the *main* activity (i.e. the entry point to the user)
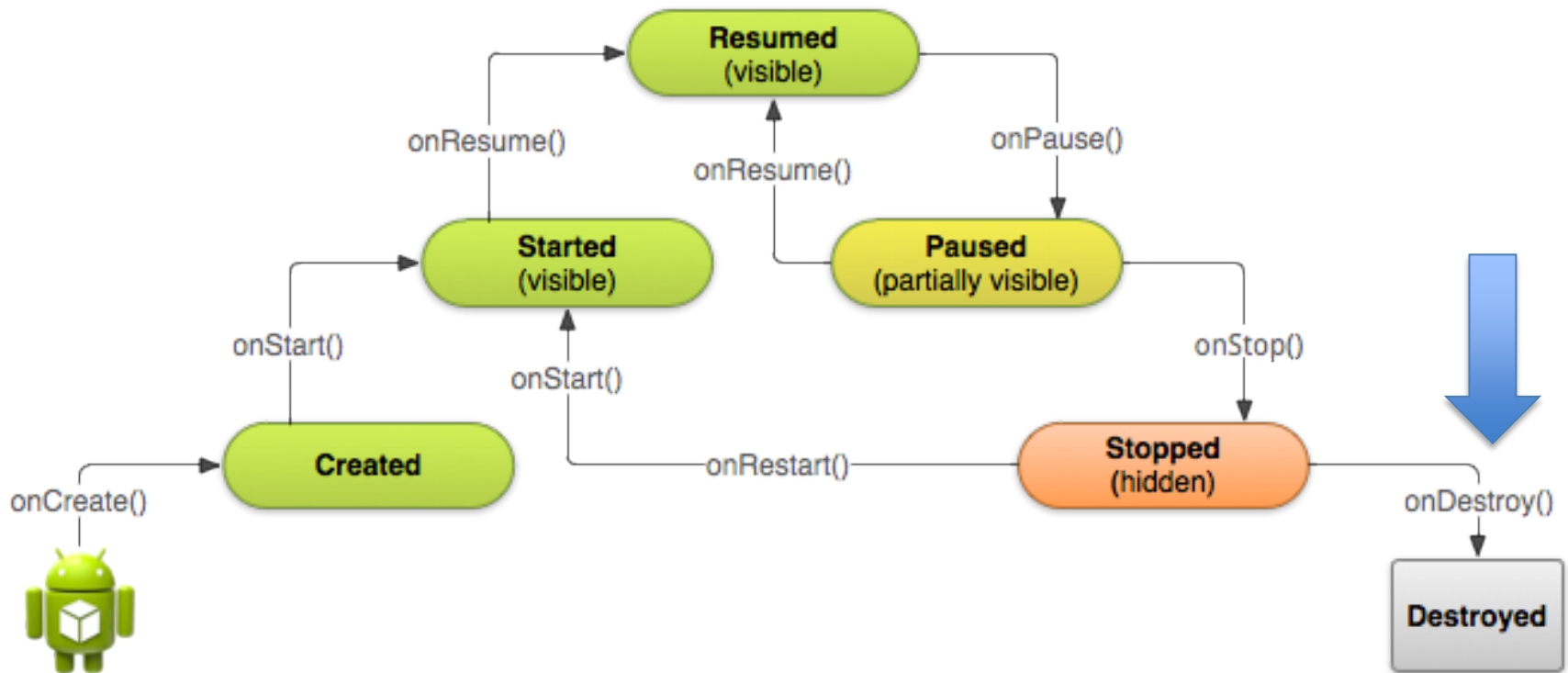
```
<application android:label="Snake on a Phone">
  <activity android:name="Snake"
    android:theme="@android:style/Theme.NoTitleBar"
    android:screenOrientation="portrait"
    android:configChanges="keyboardHidden|orientation">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    define other activities here...

</application>
```

Basic Activity Lifecycle
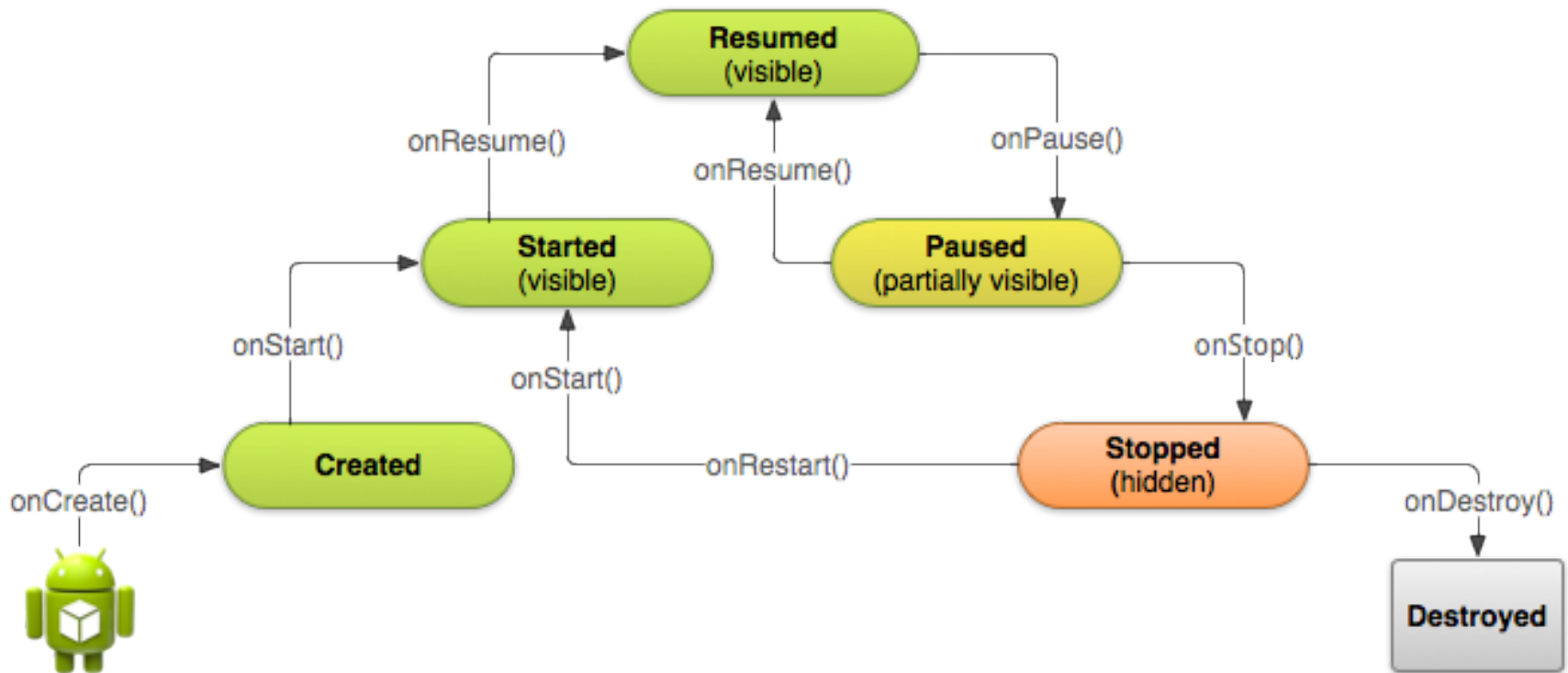
Basic Activity Lifecycle

Destroy Application

- Two ways of destroying an application

- onPause() -> onStop() -> onDestroy()
  - Most common. Most cleanup done on onPause() and onStop()
  - Kill background threads in onDestroy()

- onCreate() -> finish() -> onDestroy()
 - Activity operates as a temporary decision maker. Destroy immediately after being created.

MIT AITI

Basic Activity Lifecycle
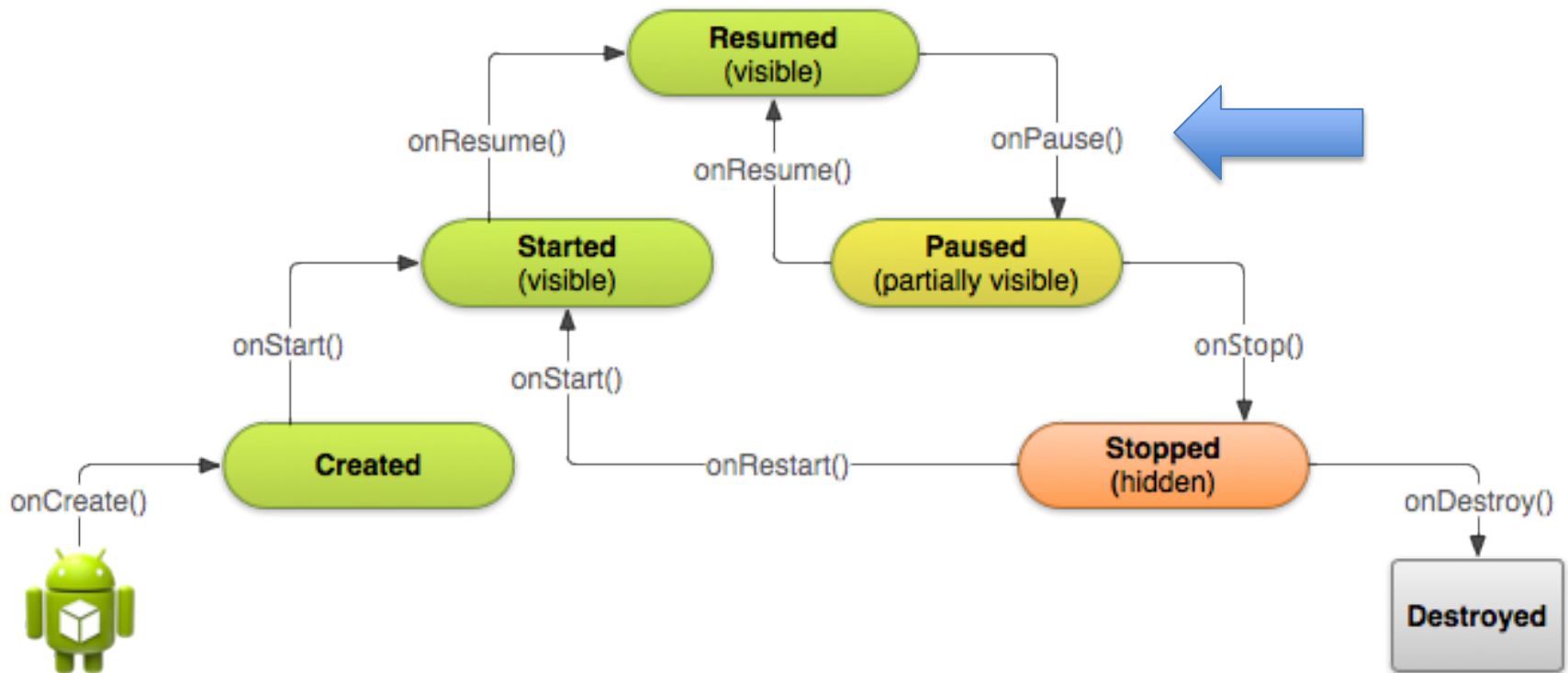
Actions of onPause()

- Stop animations or other ongoing actions that could consume CPU.

- Commit unsaved changes, but only if users expect such changes to be permanently saved when they leave (such as a draft email).

- Release system resources, such as broadcast receivers, handles to sensors (like GPS), or any resources that may affect battery life while activity is paused and the user does not need them.

- Keep it simple – avoid intensive computation. User may come back to it soon.

Basic Activity Lifecycle
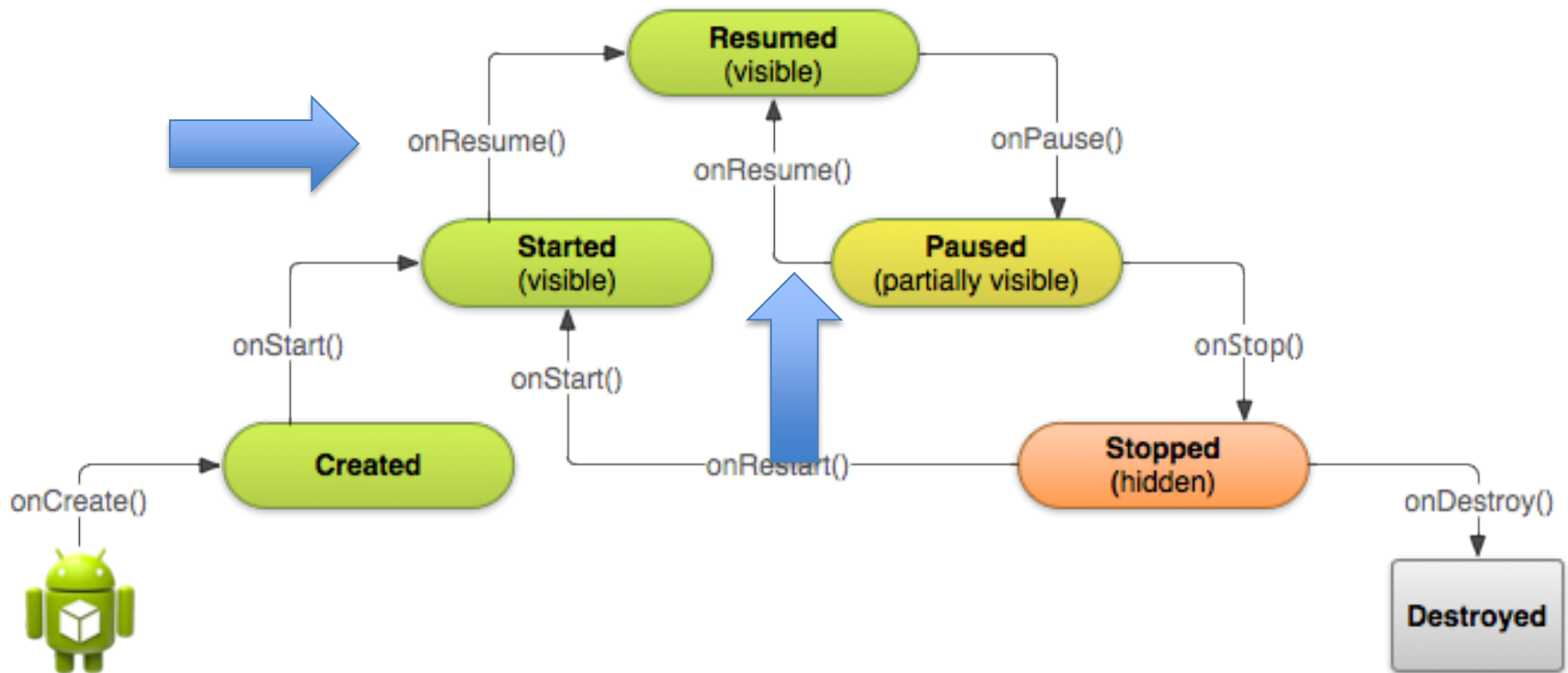
Actions of onResume()

- Called from Started and Paused states

- Initialize components that you release during onPause() and perform any other initializations that must occur each time the activity enters the Resumed state (such as begin animations and initialize components only used while the activity has user focus).

- Counterpart of onPaused()

Save your Activity State

• Additional methods. Allows you to specify additional state data you would like to save in case the Activity instance must be recreated
• System calls these methods as an Activity is being destroyed or restored (recreated).

MIT AITI
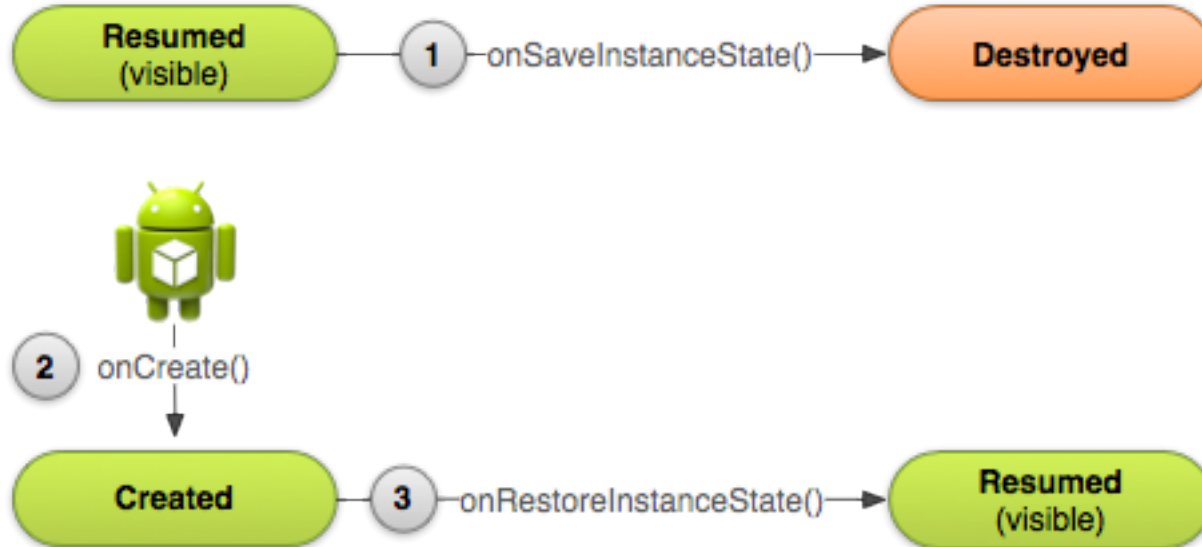
Example: onSaveInstanceState()

```java
static final String STATE_SCORE = "playerScore";
static final String STATE_LEVEL = "playerLevel";
...

@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
    // Save the user's current game state
    savedInstanceState.putInt(STATE_SCORE, mCurrentScore);
    savedInstanceState.putInt(STATE_LEVEL, mCurrentLevel);

    // Always call the superclass so it can save the view hierarchy state
    super.onSaveInstanceState(savedInstanceState);
}
```

Recreate an Activity

- Use onCreate() – check if bundle is not empty

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState); // Always call the superclass first

    // Check whether we're recreating a previously destroyed instance
    if (savedInstanceState != null) {
        // Restore value of members from saved state
        mCurrentScore = savedInstanceState.getInt(STATE_SCORE);
        mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);
    } else {
        // Probably initialize members with default values for a new instance
    }
    ...
}
```

Recreate an Activity

- Better way – use onRestoreInstanceState()
- No need to check if Bundle is empty
- Called by system after the onStart() method

```java
public void onRestoreInstanceState(Bundle savedInstanceState) {
    // Always call the superclass so it can restore the view hierarchy
    super.onRestoreInstanceState(savedInstanceState);

    // Restore state members from saved instance
    mCurrentScore = savedInstanceState.getInt(STATE_SCORE);
    mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);
}
```

Resources

- Managing the Activity Lifecycle

  - http://developer.android.com/training/basics/activity-lifecycle/index.html

- Images and source code examples taken from site above

MIT AITI