



Activity

# Programming the Android Platform

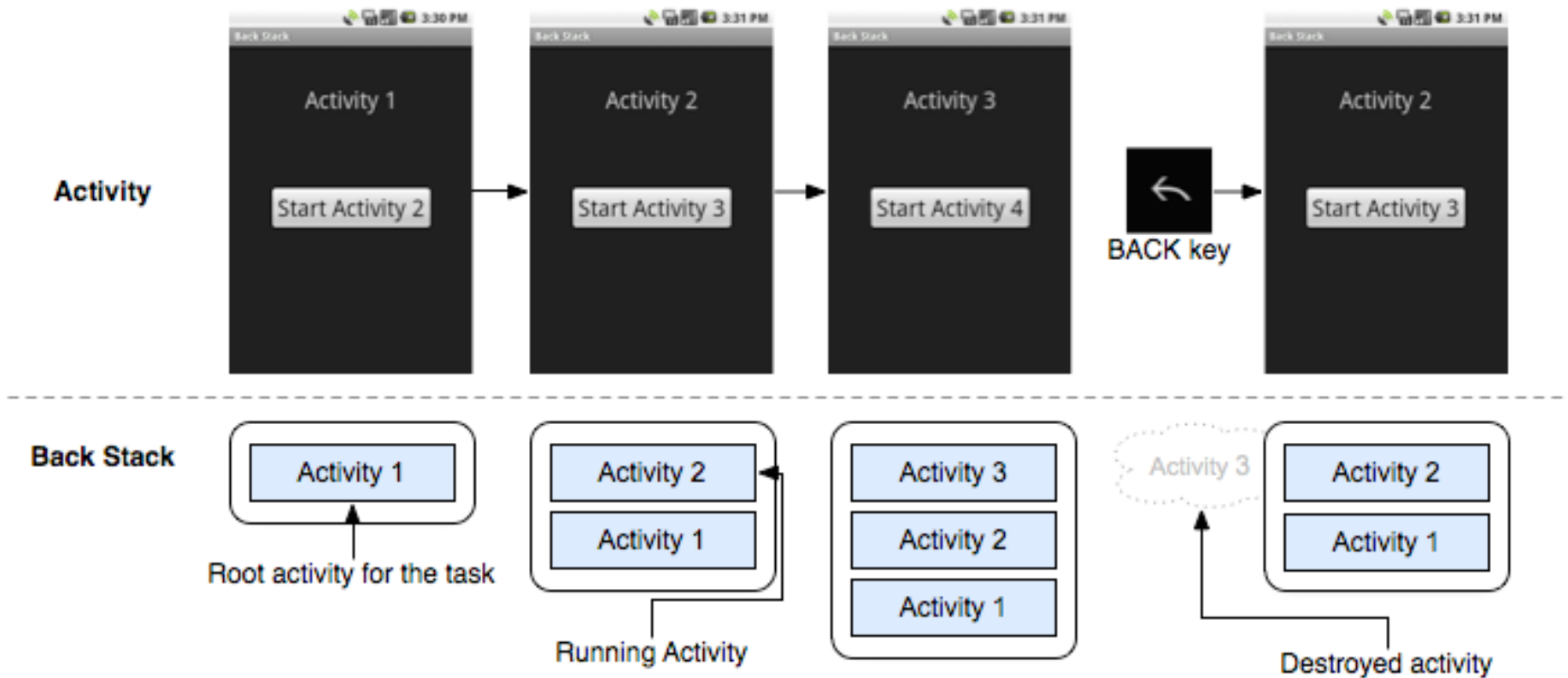
# Activity

- Provides a visual interface for user interaction
- Typically supports one thing a user can do
  - View an email message
  - Show a login screen
- Applications can include several activities

# Tasks

- A Task is a set of related Activities
  - Task not necessarily provided by a single application
- Android manages an Activity back stack
  - Launching an Activity places it on top of the stack
  - Hitting back button pops current Activity off the stack

# Task Stack



[developer.android.com/guide/topics/fundamentals/tasks-and-back-stack.html](https://developer.android.com/guide/topics/fundamentals/tasks-and-back-stack.html)

# Activity States

- Not started – not yet created
- Active
  - Resumed/Running - visible, has focus
  - Paused - visible, does not have focus, can be terminated
  - Stopped - not visible, does not have focus, can be terminated
- Finished – done

# The Activity Lifecycle

---

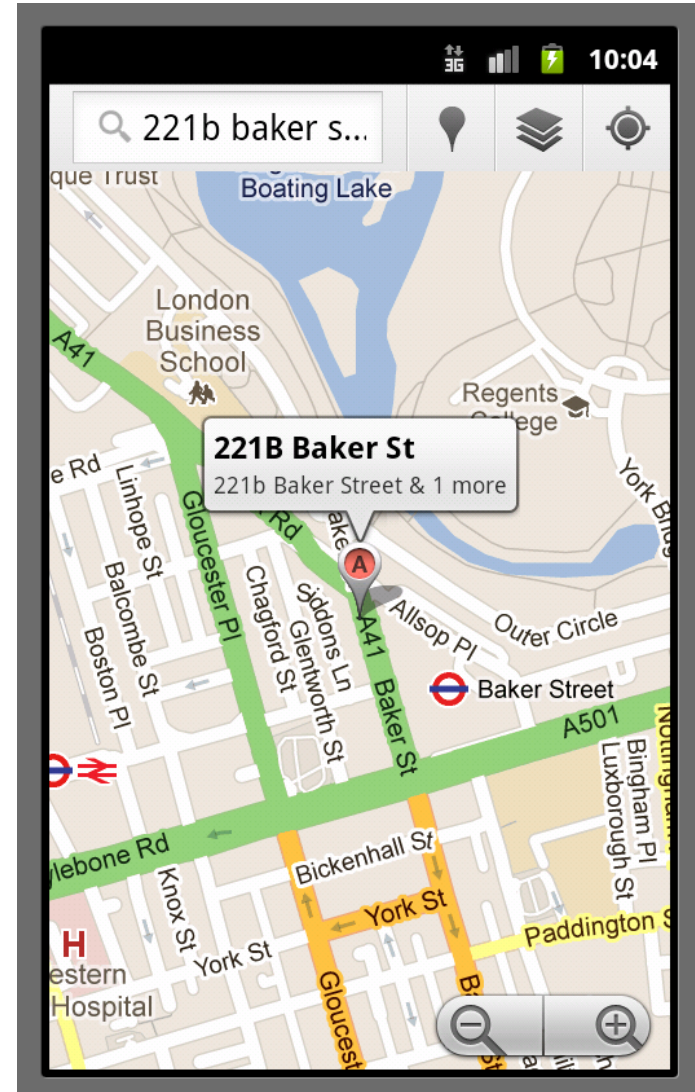
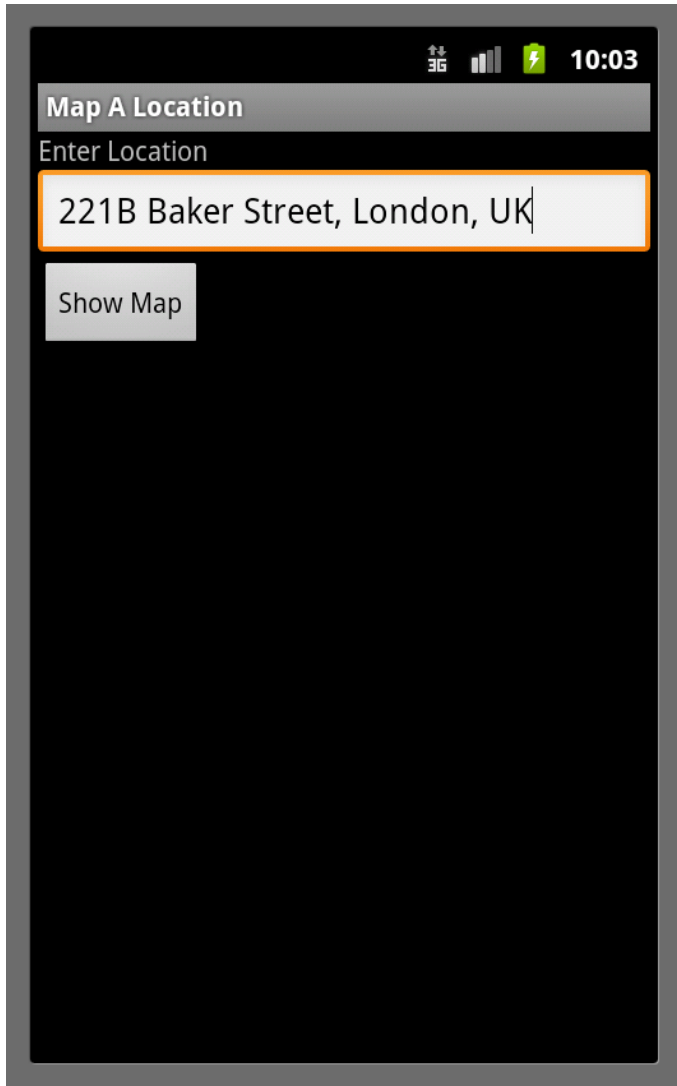
- Android communicates state changes to application by calling specific lifecycle methods

# Activity Lifecycle Methods

- protected void onCreate()
- protected void onStart()
- protected void onResume()
- protected void onPause()
- protected void onRestart()
- protected void onStop()
- protected void onDestroy()



# MapLocation



# onCreate()

- Called when Activity is first being created
- Setup global state
  - Call super.onCreate()
  - Inflate UI views
  - Configure views as necessary
  - Set the Activity's content view

# MapLocation.onCreate()

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    final EditText addressfield = (EditText) findViewById(R.id.location);
    final Button button = (Button) findViewById(R.id.mapButton);
    button.setOnClickListener(new Button.OnClickListener() {
        public void onClick(View v) {
            try {
                String address = addressfield.getText().toString();
                address = address.replace(' ', '+');
                Intent geoIntent = new Intent(android.content.Intent.ACTION_VIEW,
                                                Uri.parse("geo:0,0?q=" + address));

                startActivity(geoIntent);
            } catch (Exception e) {}
        }
    });
}
```

# onRestart()

- Called if the Activity has been stopped and is about to be started again
- Typical actions
  - Read cached state

# onStart()

- Activity is about to become visible
- Typical actions
  - Reset application

# onResume()

- About to start interacting with user
- Typical actions
  - Start foreground-only behaviors

# onPause()

- Focus about to switch to another Activity
- Typical actions
  - Shutdown foreground-only behaviors

# onStop()

- Activity is no longer visible to user
  - But may be restarted later
- Typical actions
  - Cache state

# onDestroy()

- Activity is about to be destroyed
- Typical actions
  - Save persistent state

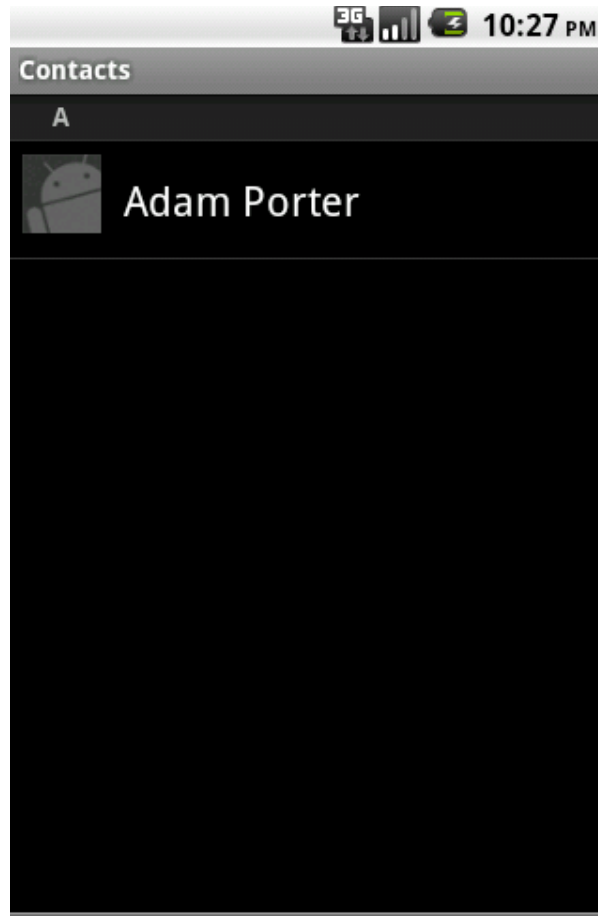
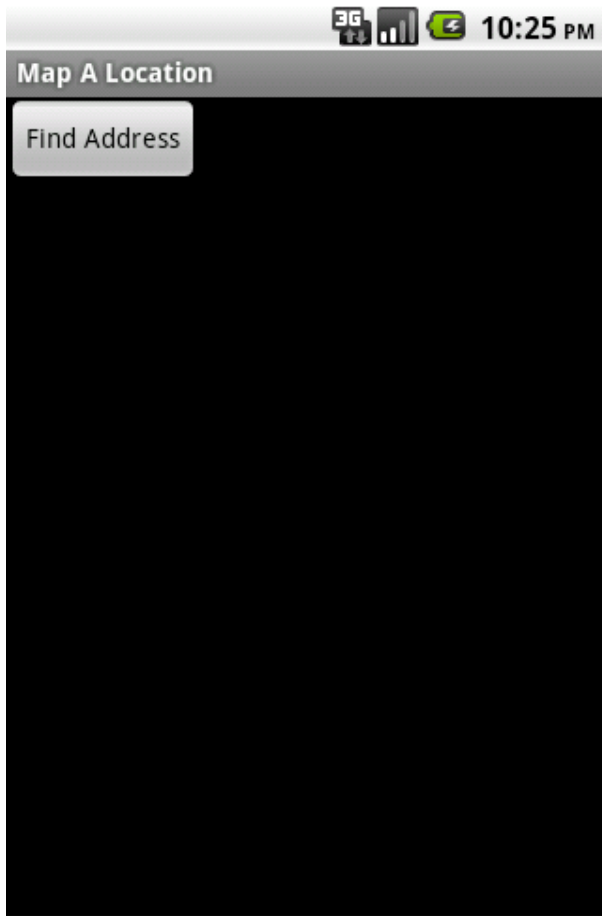
# Starting Activities

- Create an Intent object specifying the Activity to start
  - We'll discuss Intents in detail in later lectures
- Pass newly created Intent to one of the following methods
  - `startActivity()`
  - `StartActivityForResult()`
    - Callback to return result when called Activity finishes

# startActivity()

```
protected void onCreate(Bundle savedInstanceState) {  
    ...  
    public void onClick(View v) {  
        ...  
        Intent geoIntent = new Intent(android.content.Intent.ACTION_VIEW,  
                                     Uri.parse("geo:0,0?q=" + address));  
        startActivity(geoIntent);  
        ...  
    }  
    ...  
}
```

# MapLocationFromContacts



\*Not really my address 😊

# startActivityForResult()

```
private static final int PICK_CONTACT_REQUEST = 0;
...
protected void onCreate(Bundle savedInstanceState) {
...
    public void onClick(View v) {
        try {
            Intent intent = new Intent(Intent.ACTION_PICK,
                                      ContactsContract.Contacts.CONTENT_URI);
            startActivityForResult(intent, PICK_CONTACT_REQUEST);
        } catch (Exception e) {}
    }
};
}
```

# startActivityForResult() (cont.)

- Started Activity sets result by calling `Activity.setResult()`
  - `public final void setResult (int resultCode)`
  - `public final void setResult (int resultCode, Intent data)`
- `resultCode` (an int)
  - `RESULT_CANCELED`
  - `RESULT_OK`
  - `RESULT_FIRST_USER`
    - Custom resultCodes can be added after this

# startActivityForResult() (cont.)

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (resultCode == Activity.RESULT_OK &&  
        requestCode == PICK_CONTACT_REQUEST) {  
        ...  
        String address = /* extract address from data */  
        Intent geoIntent = new Intent(android.content.Intent.ACTION_VIEW,  
                                     Uri.parse("geo:0,0?q=" + address));  
        startActivity(geoIntent);  
    }  
}
```

# Configuration Changes

- Device configuration can change at runtime
  - Keyboard, orientation, locale, etc
- On configuration changes, Android usually kills & restarts the current Activity
- Activity restarting should be fast. If necessary you can:
  - Retain an Object during a configuration change
  - Manually handle the configuration change

# Retaining an Object

- Hard to recompute data can be cached to speed up handling of configuration changes
- Override `onRetainNonConfigurationInstance()` to build & return configuration Object
  - Will be called between `onStop()` and `onDestroy()`
- Call `getLastNonConfigurationInstance()` during `onCreate()` to recover retained Object
- **Note:** These methods have been deprecated in favor of methods in the `Fragment` class (will discuss at a later date).

# Manual Reconfiguration

- Can prevent system from restarting Activity
- Declare the configuration changes the Activity handles in AndroidManifest.xml file, e.g.,  

```
<activity android:name=".MyActivity  
    android:configChanges="orientation keyboardHidden"  
    ...>
```
- When configuration changes, Activity's `onConfigurationChanged()` method is called & passed a `Configuration` object specifying the new device configuration

# Source Code Examples

- MapLocationFromContacts