Master in Computer Engineering

Information Management Mobile Devices

Activities

Javier Abad (abad@decsai.ugr.es)

Dept. Of Computer Science and Artificial Intelligence

http://decsai.ugr.es

Let's remember...

- Key components of an Android application:
  - Activities (Activities)
  - Services (Services)
  - Ad receivers (Broadcast Receivers)
  - Content providers (Content Providers)
- **contents:**
  - The Activity class
  - Stack activities (Task Backstack)
  - The life cycle of activities
  - Initiation program activities
  - Change management device

Activities

- It provides a visual interface that allows user interaction.
- Modular: each activity must support a specific task you can perform user (read an email, displaying a login screen, etc.).
- Applications consist of several activities.
- Android navigation aid between activities:
  - Chores
  - Task stack
  - Suspension and resumption of activities

Chores

- A task is a set of related activities.

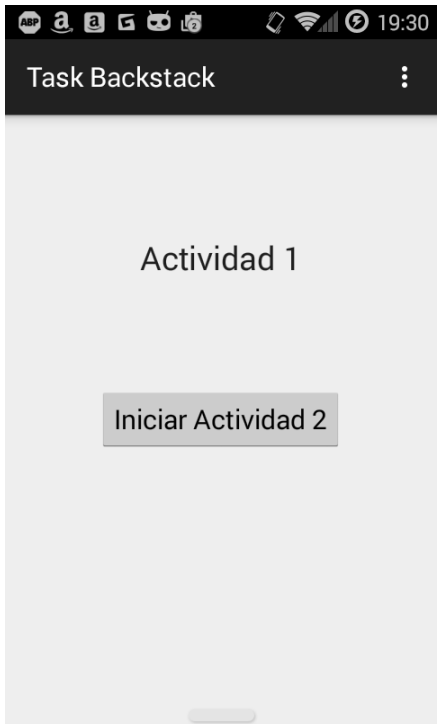
- These activities may belong to the same application but do not have to.
- Most tasks are launched from the home screen.

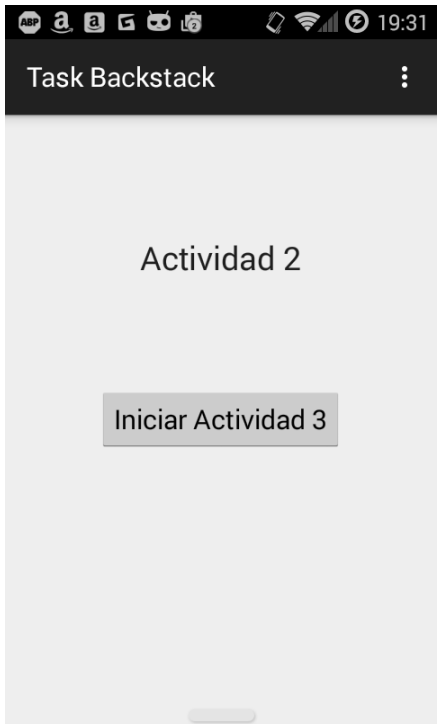
http://developer.android.com/guide/components/tasks-and-back-stack.html

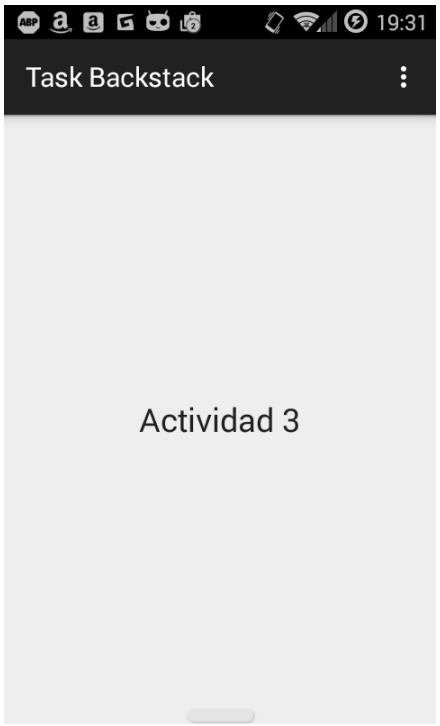
Task stack

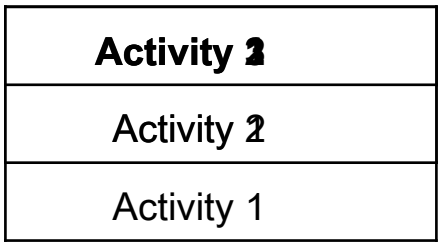
- When an activity is launched, it is inserted into the top of the stack task (is stacked - push).
- When the activity is destroyed, it is removed from the stack of tasks (it pops - pop).

Task stack


19:30
Task Backstack
Actividad 1
Iniciar Actividad 2


19:31
Task Backstack
Actividad 2
Iniciar Actividad 3


19:31
Task Backstack
Actividad 3


Activity 3
Activity 2
Activity 1

The life cycle of activities

- When you run an application, the activities are created, suspended, they resumed and terminated as is necessary.

- Some of these state changes depend on the user's actions.
  - For example, pressing the Back button or the Home button.
- Others rely on Android.
  - For example, you can kill when you need your resources activities.

States lifecycle activities

- Resumed / Running (Resumed, Running)
  - Visible, the user interacts with the activity
- Paused (Paused)
  - Visible, the user does not interact
- Stopped (stopped)
  - Can not visible, terminated

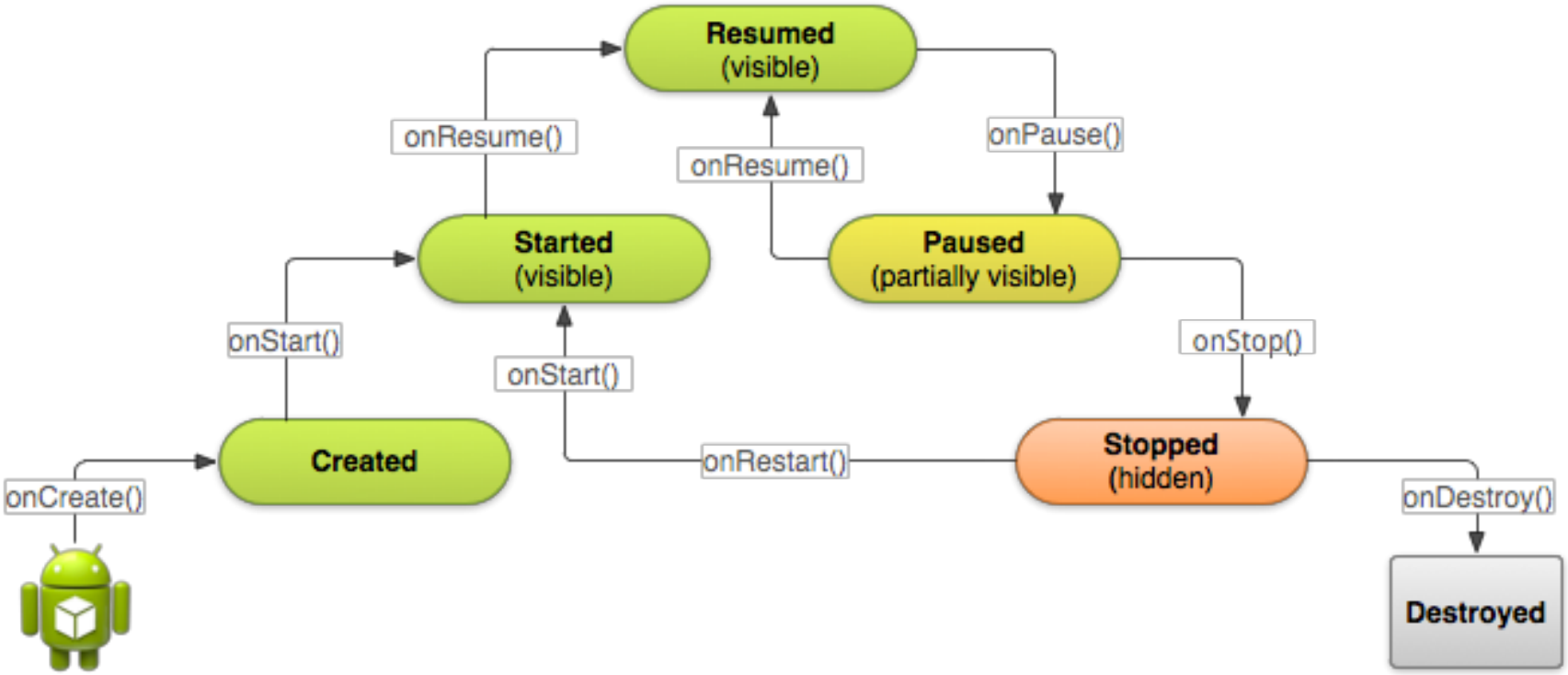
Methods of lifecycle activities

- Android announces changes in activity life cycle by calling specific methods.

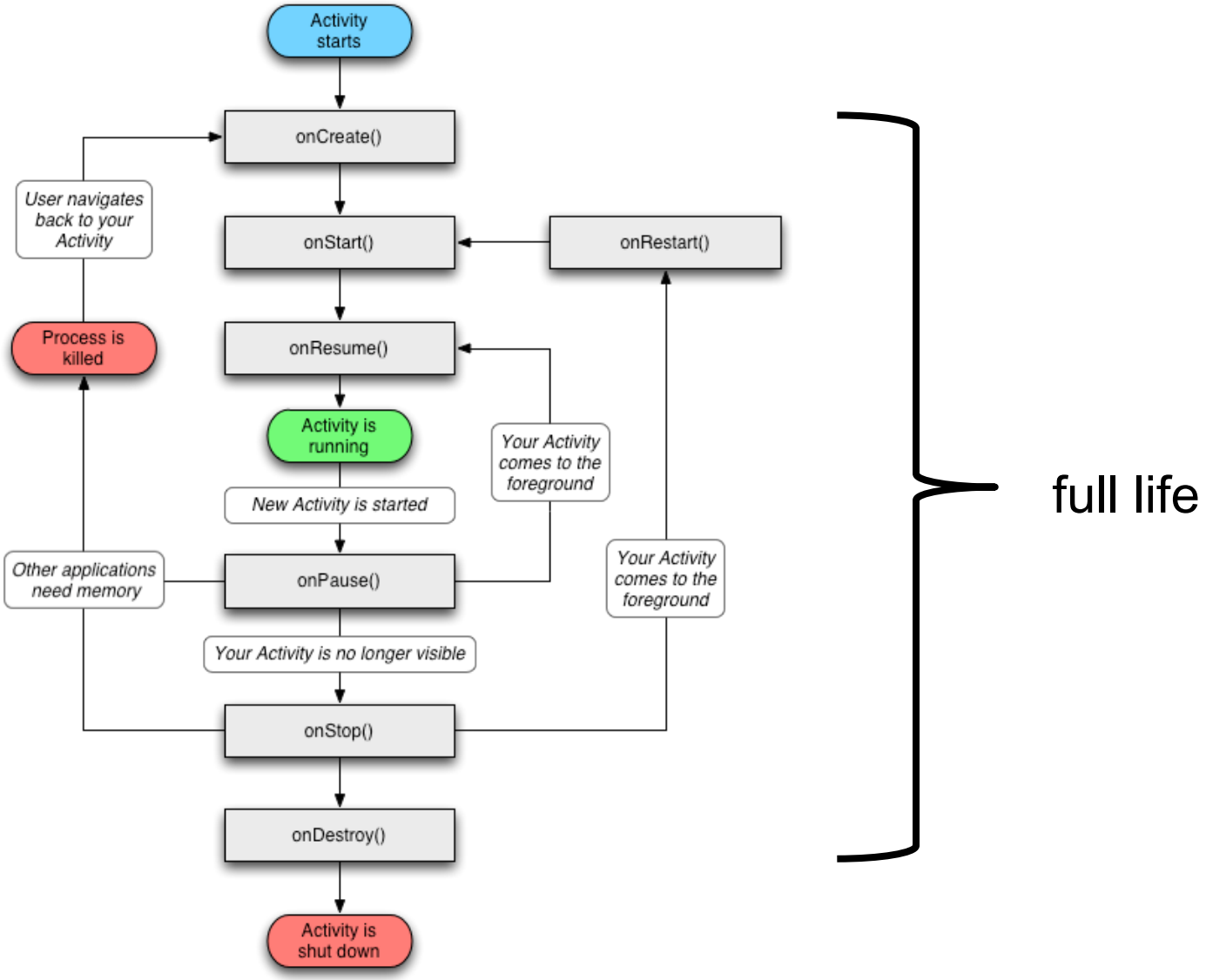
protected void onCreate (Bundle savedInstanceState) protected void onStart () protected void onResume () protected void onPause () protected void onRestart () protected void onStop () protected void OnDestroy ()

- We'll have to redefine (override) one or more of these methods in our business.

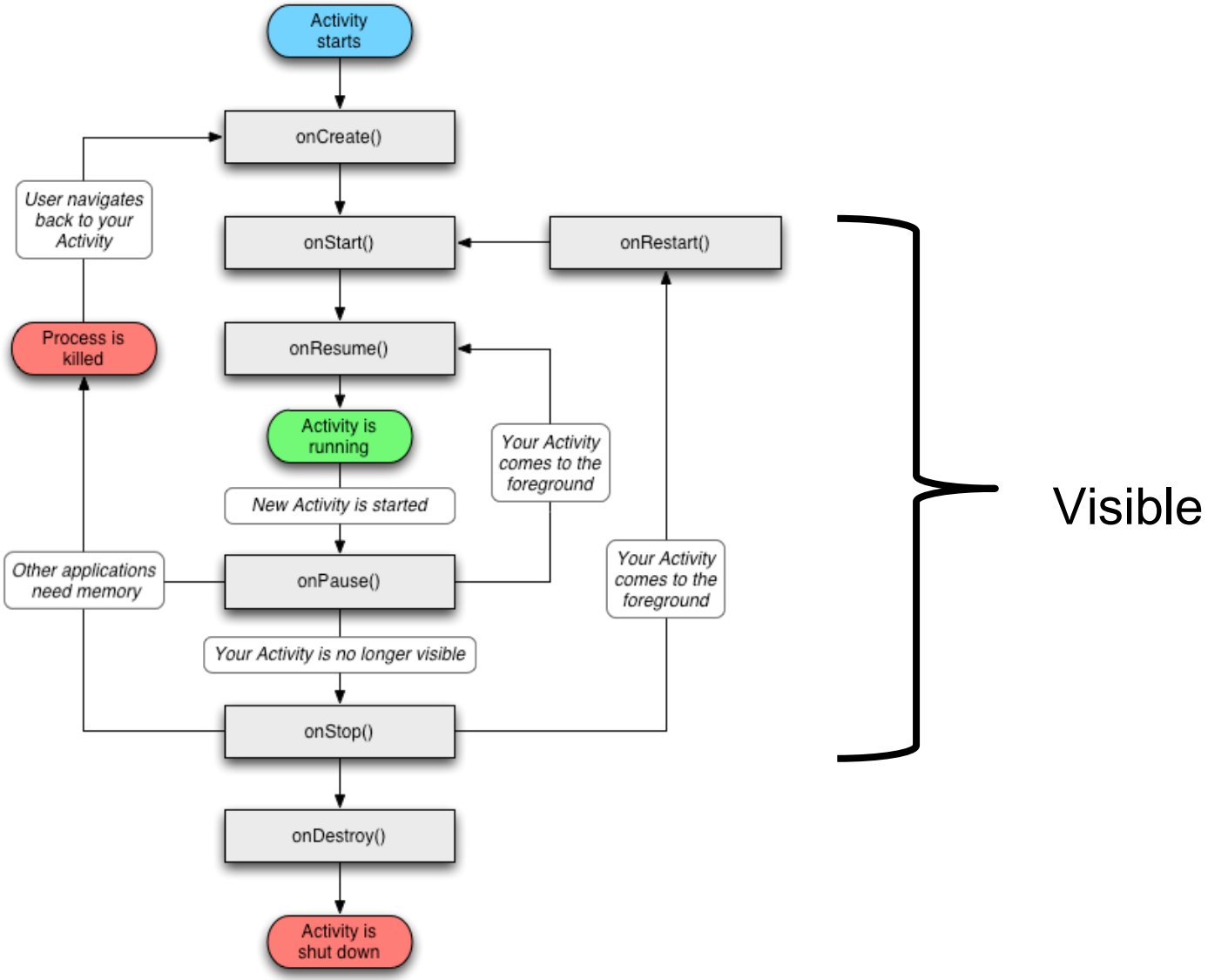
The life cycle of activities



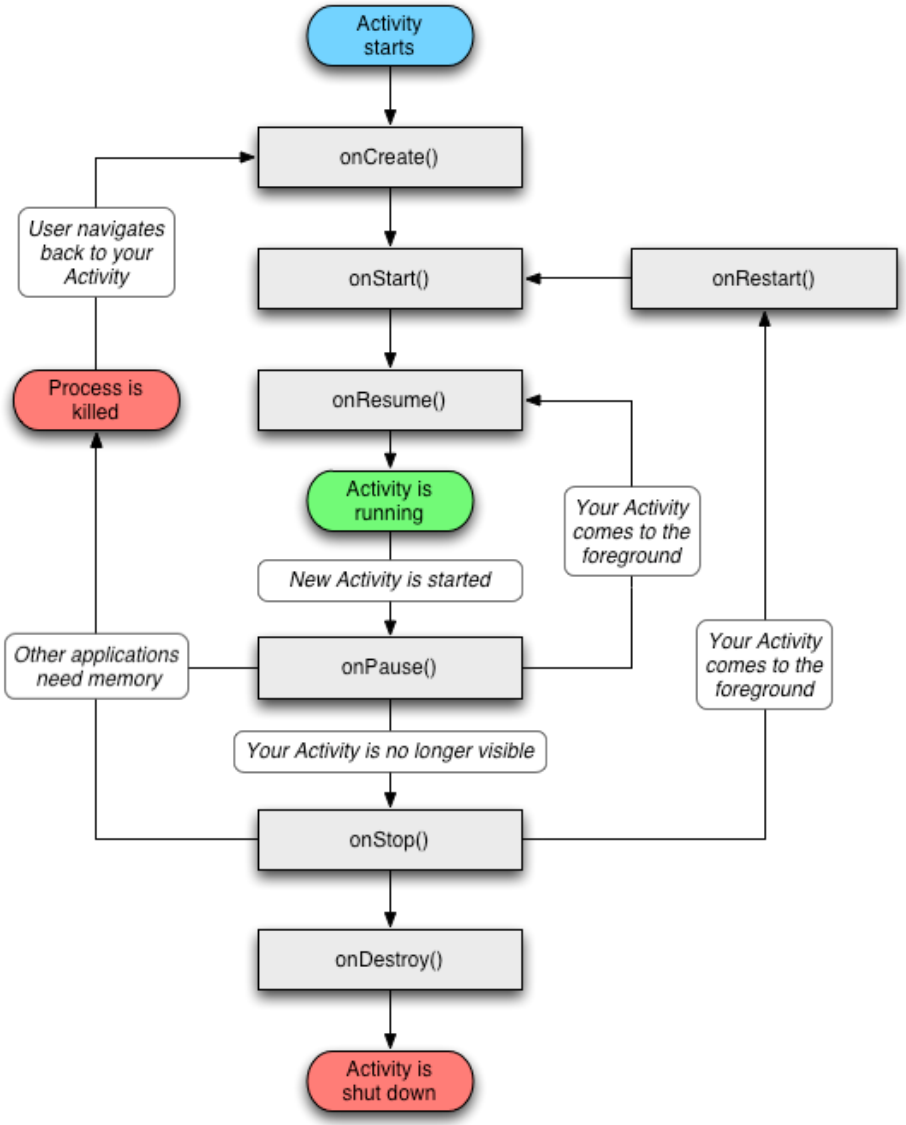
The life cycle of activities



The life cycle of activities

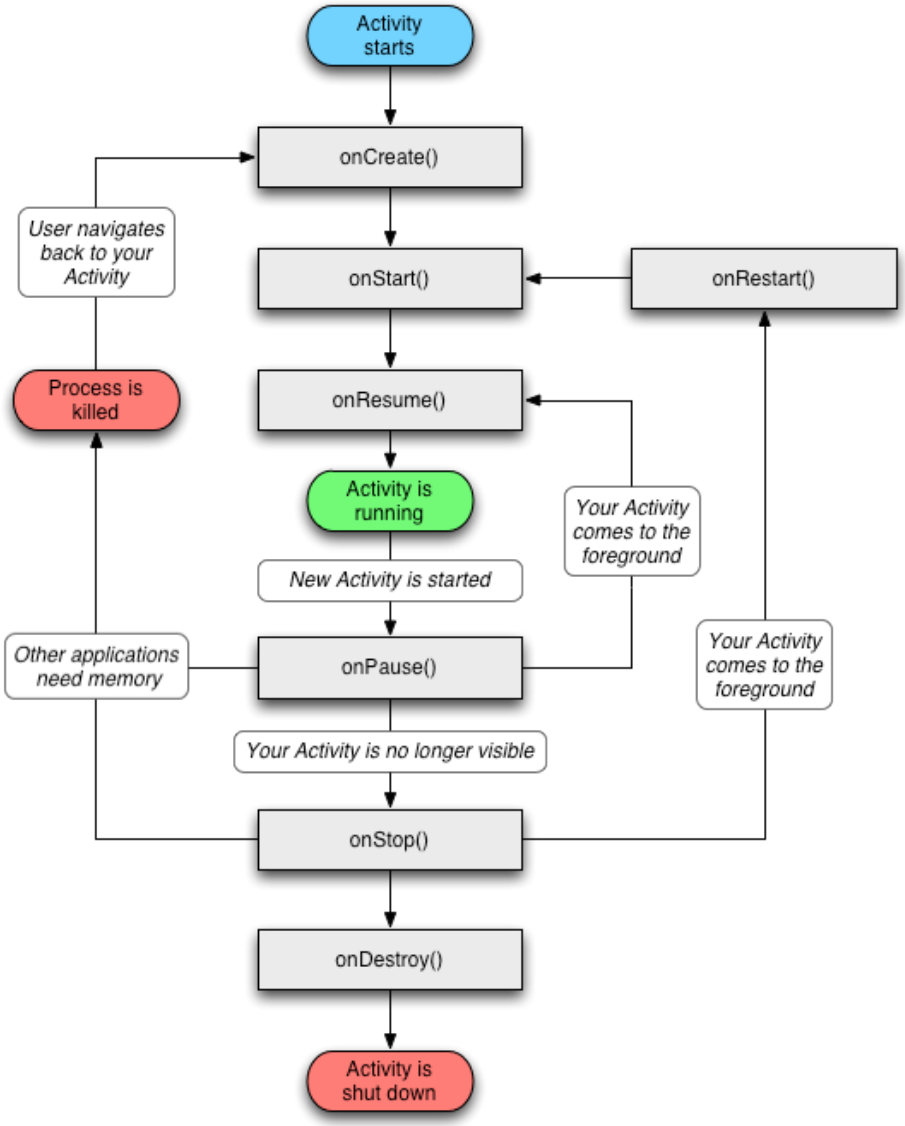


The life cycle of activities



Visible and in foreground

The life cycle of activities



onCreate ()

- It is invoked when the activity is created
- It sets the initial state of the activity
  - Super.onCreate calls ()
  - Establishes the interface with setContentView ()
  - Gets reference to views or elements of the user interface
  - Configures the views as required

onCreate ()

```
public class MainActivity extends Activity {

        private String end TAG = "UbicacionenMapa"; @Override


        protected void onCreate (Bundle savedInstanceState) {
                super.onCreate (savedInstanceState); setContentView
                (R.layout.activity_main);

                Toast.makeText (this, R.string.string_onCreate, Toast.LENGTH_SHORT) .show (); end EditText text =

                (EditText) findViewById (R.id.location); end BUTTONs = (Button) findViewById (R.id.map_button);

                boton.setOnClickListener (new Button.OnClickListener () {


                        @Override
                        public void onClick (View v) {
                                try { String address = texto.getText (). ToString ();

                                        direccion.replace address = ( ',' + ');
                                        Intent geoIntent = new Intent (android.content.Intent.ACTION_VIEW,
                                                        Uri.parse ( "geo: 0.0 q =" + address)); startActivity
                                (geoIntent); } Catch (Exception e) {

                                        Log.e (TAG e.ToString ()); }}}); }
```

onRestart ()

- It invoked if the activity has stopped, but it will start again.

- Typical actions:
  - Any special processing required only after the activity to have stopped

onStart ()

- It called when the activity will be visible
- Typical actions:
  - Log own procedures only the visible state
  - Load or reset the status of a persistent application

onResume ()

- It called when the activity is visible and will interact with the user
- Typical actions:
  - procedures of the foreground

onPause ()

- It called when the activity will lose the foreground, because another activity happens to foreground
- Typical actions:
  - Detention procedures of the foreground
  - Store persistent state

onStop ()

- It called when the activity is no longer visible to the user
  - (It can be restarted later)
- Typical actions:
  - Save the state of the activity
- **Important:** may not be invoked if Android kill the application

OnDestroy ()

- It is invoked when the activity will destroy
- typical actions
  - Free resources Activity

- **Important:** may not be invoked if Android kill the application

Methods of lifecycle activities

```
@Override
protected void onStart () {
        super.onStart ();
        Log.i (TAG (String) getText (R.string.string_onStart)); Toast.makeText (this, R.string.string_onStart,
    Toast.LENGTH_SHORT) .show (); }


@Override
protected void onRestart () {
        super.onRestart ();
        Log.i (TAG (String) getText (R.string.string_onRestart)); Toast.makeText (this, R.string.string_onRestart,
    Toast.LENGTH_SHORT) .show (); }


@Override
protected void onResume () {
        super.onResume ();
        Log.i (TAG (String) getText (R.string.string_onResume)); Toast.makeText (this, R.string.string_onResume,
    Toast.LENGTH_SHORT) .show (); }
```

Methods of lifecycle activities

```
@Override
protected void onPause () {
        super.onPause ();
        Log.i (TAG (String) getText (R.string.string_onPause)); Toast.makeText (this, R.string.string_onPause,
    Toast.LENGTH_SHORT) .show (); }


@Override
protected void onStop () {
        super.onStop ();
        Log.i (TAG (String) getText (R.string.string_onStop)); Toast.makeText (this, R.string.string_onStop,
    Toast.LENGTH_SHORT) .show (); }


@Override
protected void OnDestroy () {
        super.onDestroy ();
        Log.i (TAG (String) getText (R.string.string_onDestroy)); Toast.makeText (this, R.string.string_onDestroy,
    Toast.LENGTH_SHORT) .show (); }
```

Initiation program activities

- Create an Intent object that specifies the activity to start

- Intent pass the object we have created methods such as:

  - startActivity ()
  - StartActivityForResult ()
    - The invoked activity calls a callback method when finished to return a result

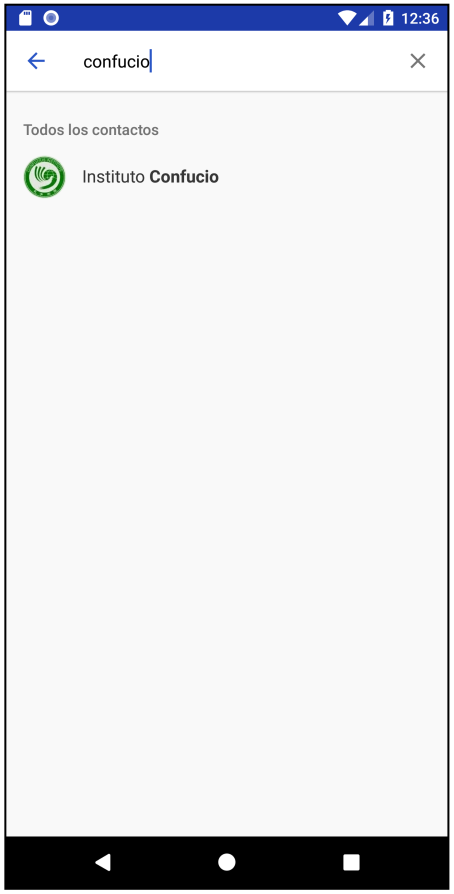
Initiation of activities

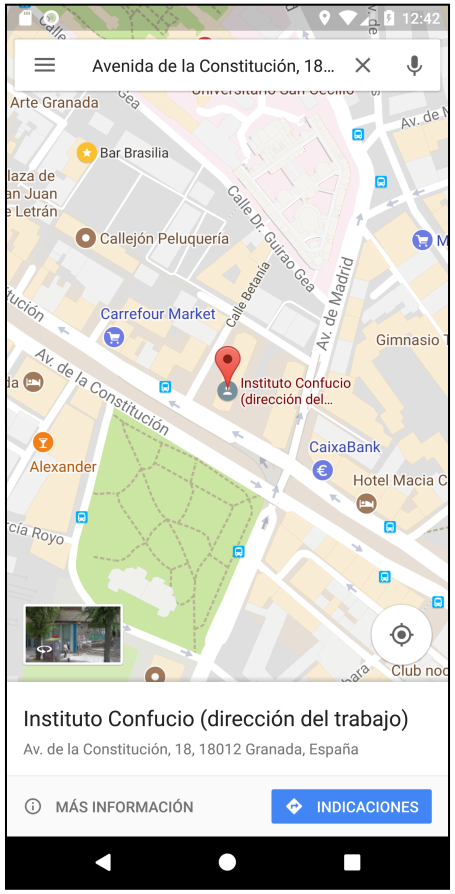
```
boton.setOnClickListener (new Button.OnClickListener () {
        @Override
        public void onClick (View v) {
                try { String address = texto.getText (). ToString ();

                    direccion.replace address = ( ',' + ');
                    Intent geoIntent = new Intent (android.content.Intent.ACTION_VIEW,
                                Uri.parse ( "geo: 0.0 q =" + address)); startActivity
                (geoIntent); } Catch (Exception e) {

                    Log.e (TAG e.ToString ()); }}});
```

ContactosenMapa





ContactosenMapa

```
public class MainActivity extends AppCompatActivity {

        private static final String DATA_MIMETYPE = ContactsContract.Data.MIMETYPE; private static final Uri
        DATA_CONTENT_URI = ContactsContract.Data.CONTENT_URI; private static final String DATA_CONTACT_ID =
        ContactsContract.Data.CONTACT_ID;

        private static final String CONTACTS_ID = ContactsContract.Contacts._ID; private static final Uri CONTACTS_CONTENT_URI
        = ContactsContract.Contacts.CONTENT_URI;

        private static final String STRUCTURED_POSTAL_CONTENT_ITEM_TYPE =
                        ContactsContract.CommonDataKinds.StructuredPostal.CONTENT_ITEM_TYPE; private static final
        String STRUCTURED_POSTAL_FORMATTED_ADDRESS =
                        ContactsContract.CommonDataKinds.StructuredPostal.FORMATTED_ADDRESS;

        private static final int PICK_CONTACT_REQUEST = 0; static String TAG =
        "Contacts in Map";

...
```

ContactosenMapa

```
... @Override

        protected void onCreate (Bundle savedInstanceState) {
                super.onCreate (savedInstanceState); setContentView
                (R.layout.activity_main);
                Log.i (TAG (String) getText (R.string.string_onCreate)); Toast.makeText (this, R.string.string_onCreate,
                Toast.LENGTH_SHORT) .show ();

                end BUTTONs = (Button) findViewByld (R.id.boton);

                boton.setOnClickListener (new Button.OnClickListener () {
                        @Override
                        public void onClick (View v) {
                                try { Intent intent = new Intent (Intent.ACTION_PICK, CONTACTS_CONTENT_URI);

                                        startActivityForResult (intent, PICK_CONTACT_REQUEST); } Catch (Exception
                                e) {
                                        //Do nothing } } }); }
```

Activity.setResult ()

- Initiated activity could set the result to return invoking Activity.setResult () method
  - public void setResult end (int resultCode)
  - final public void setResult (int resultCode, Intent data)
- resultCode (integer)
  - RESULT_CANCELED
  - RESULT_OK
  - RESULT_FIRST_USER
  - (we can add custom code)

onActivityResult

```
@Override
protected void onActivityResult (int requestCode, int resultCode, Intent data) {
    if (resultCode == == Activity.RESULT_OK && requestCode PICK_CONTACT_REQUEST) {
        ContentResolver cr = getContentResolver ();
        Cursor cursor = cr.query (data.getData (), null, null, null, null);

        if (null! = && cursor.moveToFirst cursor ()) {
            String id = cursor.getString (cursor.getColumnIndex (CONTACTS_ID)); String where = DATA_CONTACT_ID + "? =
            AND" + DATA_MIMETYPE + "="; String [] = new String whereParameters [] {id,
            STRUCTURED_POSTAL_CONTENT_ITEM_TYPE}; Cursor addrCur = cr.query (DATA_CONTENT_URI, null, where,
            whereParameters, null);
```

onActivityResult

```
if (null! = addrCur addrCur.moveToFirst && ()) {
            String formattedAddress = addrCur.getString (addrCur
                        . getColumnIndex (STRUCTURED_POSTAL_FORMATTED_ADDRESS));
            if (null! = formattedAddress) {
                formattedAddress = formattedAddress.replace ( '', '+'); Intent geoIntent = new Intent
                (android.content.Intent.ACTION_VIEW,
                        Uri.parse ( "geo: 0.0 q =" + formattedAddress)); startActivity
            (geoIntent); }}


        if (null! = addrCur)
            addrCur.close (); }

    if (null! = cursor)
        cursor.close (); }}
```

Configuration changes

- Device settings can be changed during application execution
  - Keyboard
  - Language
  - Orientation
  - Etc.
- When configuration changes occur, Android often kill and restart the current activity

Configuration changes

- The resumption of activity should be fast enough

- If required we can:
  - Retain an object that contains important status information during the configuration change
  - Manually manage change configuration

Retain a Java object

- Those whose data (re) calculation is expensive can be stored in an object to accelerate configuration changes.

- Redefining the method
  onRetainNonConfigurationInstance () to build and return the object containing configuration.

  - It is called between onStop () and OnDestroy ()

- GetLastNonConfigurationInstance call () in onCreate () to retrieve the stored object

IMPORTANT: Methods "deprecated" in favor of others in the Fragment class (the study)

manual reconfiguration

- We can prevent the system restart activity
- Declare configuration changes that can manage activity in the AndroidManifest.xml file

```
<Activity android: name = "MiActividad"
  android: configChanges = "orientation | screensize | keyboardHidden"
  . . . >
```

- When the change in the configuration
  - It is called the onConfigurationChanged () method
  - an object that specifies the new device configuration is passed as a parameter.