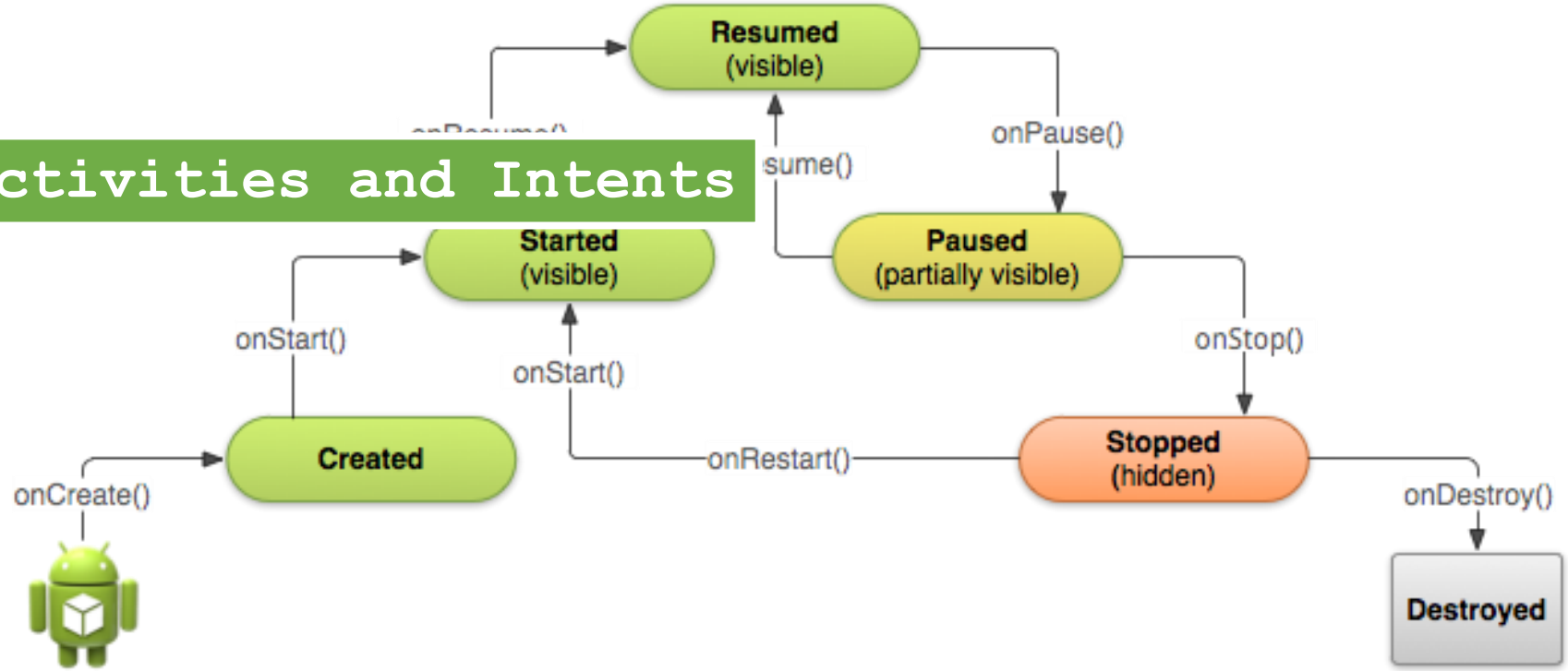
Pemrograman Mobile

3 SKS | Semester 7 | S1 Sistem Informasi

Nizar Rabbi Radliya
nizar@email.unikom.ac.id

Lesson 3

Activities and Intents



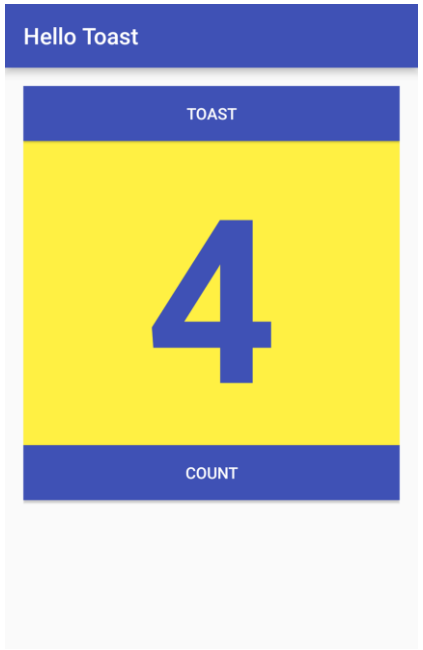
What is an Activity?

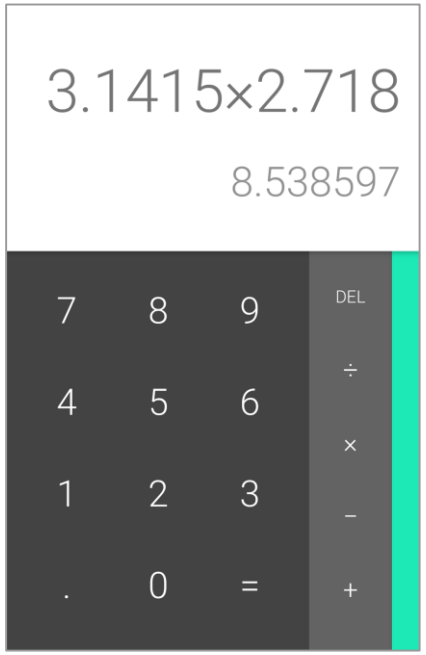
- An Activity is an application component
- Represents one window, one hierarchy of views
- Typically fills the screen, but can be embedded in other activity or a appear as floating window
- Java class, typically one activity in one file

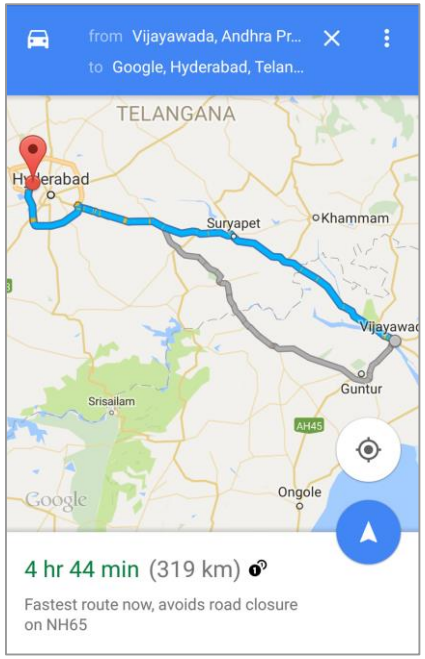
What does an Activity do?

- Represents an activity, such as ordering groceries, sending email, or getting directions
- Handles user interactions, such as button clicks, text entry, or login verification
- Can start other activities in the same or other apps
- Has a life cycle—is created, started, runs, is paused, resumed, stopped, and destroyed

Examples of activities







Apps and activities

- Activities are loosely tied together to make up an app
- First activity user sees is typically called "main activity"
- Activities can be organized in parent-child relationships in the Android manifest to aid navigation

Layouts and Activities

- An activity typically has a UI layout
- Layout is usually defined in one or more XML files
- Activity "inflates" layout as part of being created

Implement new activities

1. Define layout in XML
2. Define Activity Java class

   extends AppCompatActivity
3. Connect Activity with Layout

   Set content view in onCreate()
4. Declare Activity in the Android manifest

1. Define layout in XML

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Let's Shop for Food!" />
</RelativeLayout>
```

2. Define Activity Java class

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }
}
```

3. Connect activity with layout

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Resource is layout in this XML file

4. Declare activity in Android manifest

Main Activity needs to include intent to start from launcher icon

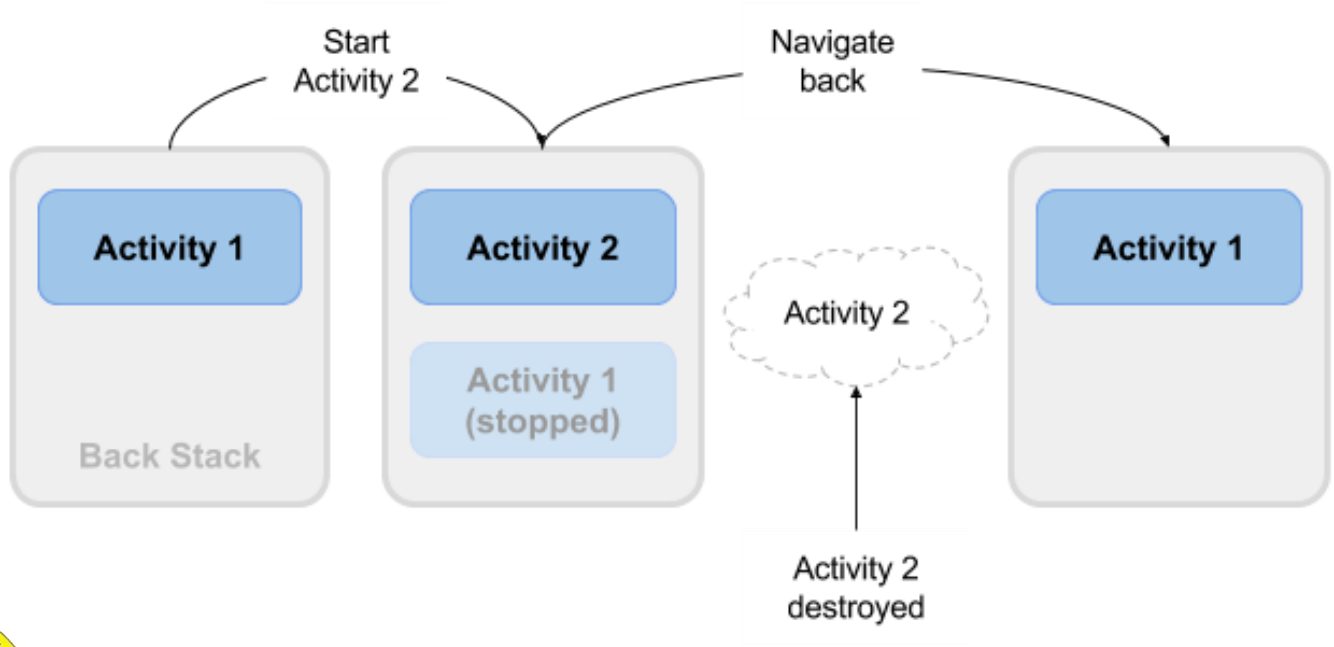
```
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

What is the Activity Lifecycle?

- The set of states an activity can be in during its lifetime, from when it is created until it is destroyed

More formally:

- A directed graph of all the states an activity can be in, and the callbacks associated with transitioning from each state to the next one



Activity states and app visibility

- Created (not visible yet)
- Started (visible)
- Resume (visible)
- Paused (partially invisible)
- Stopped (hidden)
- Destroyed (gone from memory)

State changes are triggered by user action, configuration changes such as device rotation, or system action

Callbacks and when they are called

`onCreate(Bundle savedInstanceState)`—static initialization

`onStart()`—when activity (screen) is becoming visible

`onRestart()`—called if activity was stopped (calls onStart())

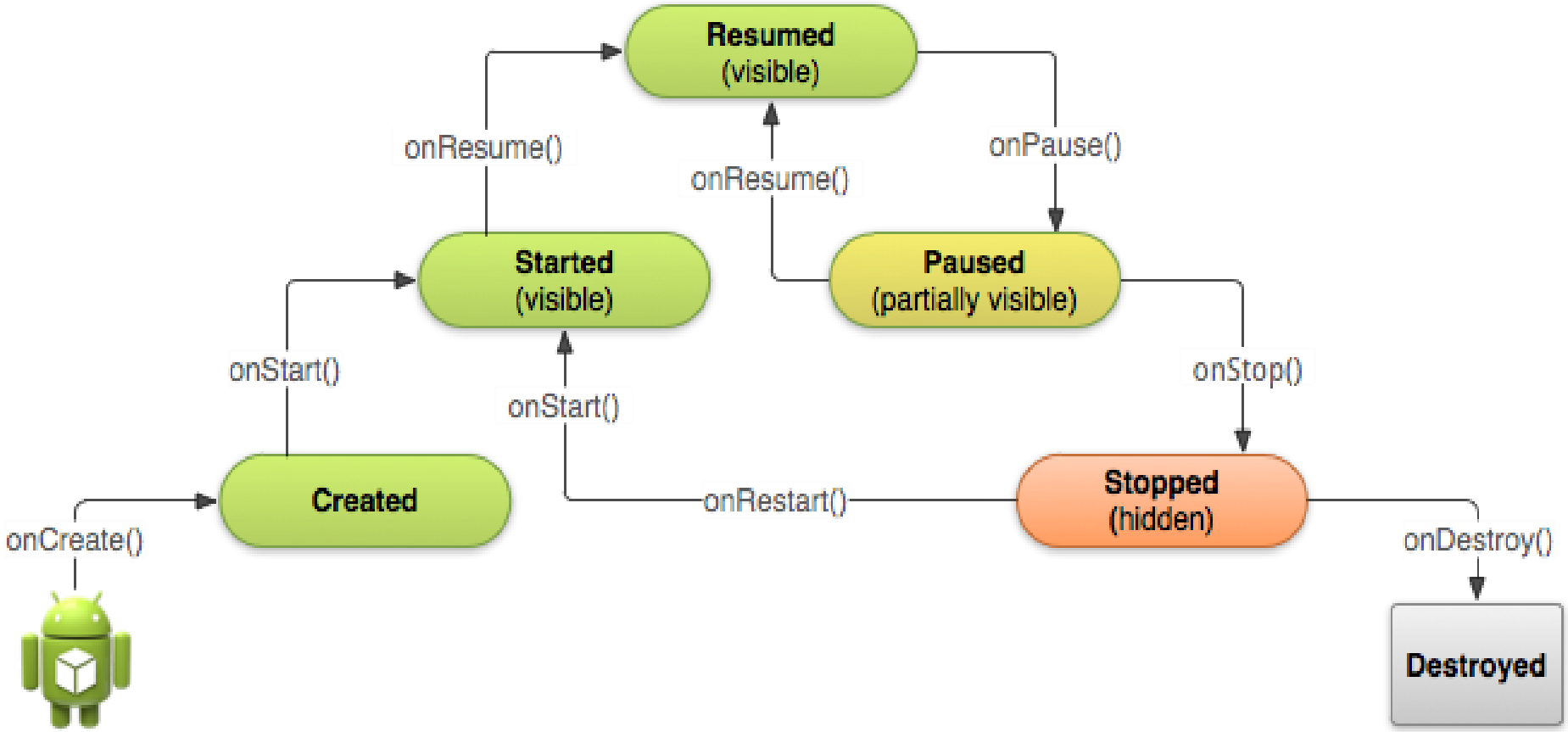
`onResume()`—start to interact with user

`onPause()`—about to resume PREVIOUS activity

`onStop()`—no longer visible, but still exists and all state info preserved

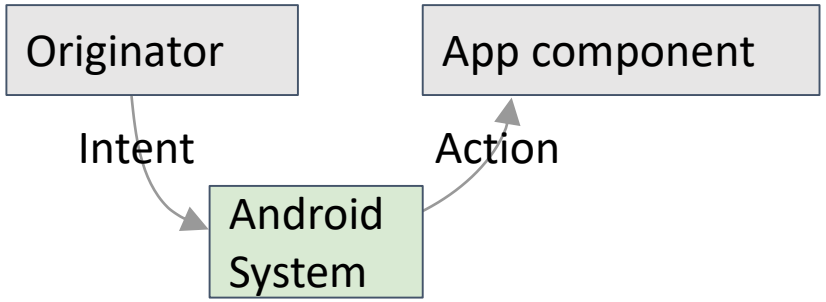
`onDestroy()`—final call before Android system destroys activity

Activity states and callbacks graph



What is an intent?

- An intent is a description of an operation to be performed.
- An Intent is an object used to request an action from another app component via the Android system.



What can intents do?

- Start activities

  A button click starts a new activity for text entry

  Clicking Share opens an app that allows you to post a photo

- Start services

  Initiate downloading a file in the background

- Deliver broadcasts

  The system informs everybody that the phone is now charging

Explicit and implicit intents

Explicit Intent

- Starts a specific activity

  Request tea with milk delivered by Nikita

  Main activity starts the ViewShoppingCart activity

Implicit Intent

- Asks system to find an activity that can handle this request

  Find an open store that sells green tea

  Clicking Share opens a chooser with a list of apps

Start an Activity with an explicit intent

To start a specific activity, use an explicit intent

- Create an intent

```
Intent intent = new Intent(this, ActivityName.class);
```

- Use the intent to start the activity

```
startActivity(intent);
```

Implicit Intents - Examples

Show a web page

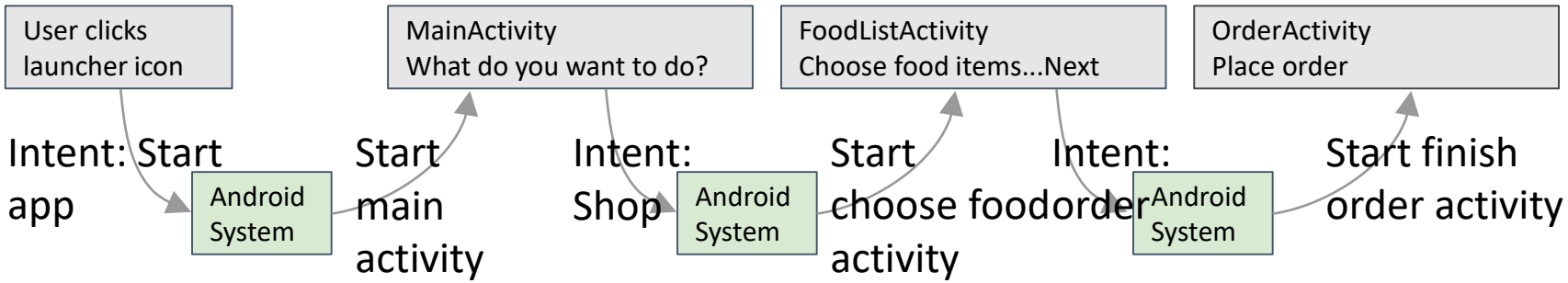
```
Uri uri = Uri.parse("http://www.google.com");
Intent it = new Intent(Intent.ACTION_VIEW,uri);
startActivity(it);
```

Dial a phone number

```
Uri uri = Uri.parse("tel:8005551234");
Intent it = new Intent(Intent.ACTION_DIAL, uri);
startActivity(it);
```

How Activities Run

- All activities are managed by the Android runtime
- Started by an "intent", a message to the Android runtime to run an activity



Two types of sending data with intents

- Data—one piece of information whose data location can be represented by an URI

- Extras—one or more pieces of information as a collection of key-value pairs in a Bundle

Sending and retrieving data

In the first (sending) activity:

- Create the Intent object
- Put data or extras into that intent
- Start the new activity with startActivity()

In the second (receiving) activity:

- Get the intent object the activity was started with
- Retrieve the data or extras from the Intent object

Putting a URI as intent data

```
// A web page URL
intent.setData(
    Uri.parse("http://www.google.com"));

// a Sample file URI
intent.setData(
     Uri.fromFile(new File("/sdcard/sample.jpg")));
```

Put information into intent extras

- `putExtra(String name, int value)`

  ⇒ `intent.putExtra("level", 406);`

- `putExtra(String name, String[] value)`

  ⇒
```
String[] foodList = {"Rice", "Beans", "Fruit"};
intent.putExtra("food", foodList);
```

- `putExtras(bundle);`

  ⇒ if lots of data, first create a bundle and pass the bundle.

- See documentation for all

Sending data to an activity with extras

```
public static final String EXTRA_MESSAGE_KEY =
    "com.example.android.twoactivities.extra.MESSAGE";
Intent intent = new Intent(this, SecondActivity.class);
String message = "Hello Activity!";
intent.putExtra(EXTRA_MESSAGE_KEY, message);
startActivity(intent);
```

Get data from intents

- `getData();`

  ⇒ `Uri locationUri = intent.getData();`
- `int getIntExtra (String name, int defaultValue)`

  ⇒ `int level = intent.getIntExtra("level", 0);`
- `Bundle bundle = intent.getExtras();`

  ⇒ Get all the data at once as a bundle.
- See documentation for all

Returning data to the starting activity

1. Use `startActivityForResult()` to start the second activity
2. To return data from the second Activity:
   - Create a ***new*** Intent
   - Put the response data in the Intent using `putExtra()`
   - Set the result to `Activity.RESULT_OK`

     or `RESULT_CANCELED`, if the user cancelled out
   - call `finish()` to close the activity
3. Implement `onActivityResult()` in first activity

Learn more

- Android Application Fundamentals
- Starting Another Activity
- Activity (API Guide)
- Activity (API Reference)
- Intents and Intent Filters (API Guide)
- Intent (API Reference)
- Navigation
- Managing the Activity Lifecycle
- Pausing and Resuming an Activity
- Stopping and Restarting an Activity
- Recreating an Activity
- Handling Runtime Changes
- Bundle

What's Next?

Testing, Debugging, and Using Support Libraries