THE ACTIVITY LIFECYCLE

STATES OF AN ACTIVITY

- An activity can have four states

STATES OF AN ACTIVITY

- An activity can have four states
    - Active/Running
      The activity is visible to the user

STATES OF AN ACTIVITY

- An activity can have four states
  - Active/Running
    The activity is visible to the user
  - Paused
    Still visible, but have some content on top of it (e.g. a dialog)

STATES OF AN ACTIVITY

- An activity can have four states
  - Active/Running
    The activity is visible to the user
  - Paused
    Still visible, but have some content on top of it (e.g. a dialog)
  - Stopped
    Another activity has replaced the activity

STATES OF AN ACTIVITY

- An activity can have four states
  - Active/Running
    The activity is visible to the user
  - Paused
    Still visible, but have some content on top of it (e.g. a dialog)
  - Stopped
    Another activity has replaced the activity
  - Killable
    If the activity is paused or stopped it can be killed

```java
public class Activity extends ApplicationContext {
     protected void onCreate(Bundle savedInstanceState);

     protected void onStart();

     protected void onRestart();

     protected void onResume();

     protected void onPause();

     protected void onStop();

     protected void onDestroy();
 }
```

```java
public class Activity extends ApplicationContext {
     protected void onCreate(Bundle savedInstanceState);

     protected void onResume();

     protected void onPause();

     protected void onStop();
 }
```

CONFIGURATION CHANGES

- Change of screen orientation, language, input mechanism etc.

CONFIGURATION CHANGES

- Change of screen orientation, language, input mechanism etc.
- All possible types of configuration changes are listed in the `Resources.Configuration` class

CONFIGURATION CHANGES

- Change of screen orientation, language, input mechanism etc.
- All possible types of configuration changes are listed in the **`Resources.Configuration`** class
- Configuration change will trigger the activity lifecycle sequence (**`onPause()`**, **`onStop()`** and **`onDestroy()`**)

CONFIGURATION CHANGES

- Change of screen orientation, language, input mechanism etc.
- All possible types of configuration changes are listed in the `Resources.Configuration` class
- Configuration change will trigger the activity lifecycle sequence (`onPause()`, `onStop()` and `onDestroy()`)
- State needs to be stored and restored!

CONFIGURATION CHANGES

- Change of screen orientation, language, input mechanism etc.
- All possible types of configuration changes are listed in the `Resources.Configuration` class
- Configuration change will trigger the activity lifecycle (`onPause()`, `onStop()` and `onDestroy()`)
- State needs to be stored and restored!
- To store data: Override `onSavedInstanceState(Bundle)` and add data to the Bundle

CONFIGURATION CHANGES

- Change of screen orientation, language, input mechanism etc.
- All possible types of configuration changes are listed in the `Resources.Configuration` class
- Configuration change will trigger the activity lifecycle (`onPause()`, `onStop()` and `onDestroy()`)
- State needs to be stored and restored!
- To store data: Override `onSavedInstanceState(Bundle)` and add data to the Bundle
- To restore data: The data will be available in the bundle passed to `onCreate(Bundle)` or `onRestoreInstanceState(Bundle)`

WHAT IF THE ACTIVITY IS KILLED?

- Same as a with configuration changes! Store data in `onSaveInstanceState(Bundle)`

WHAT IF THE ACTIVITY IS KILLED?

- Same as a with configuration changes! Store data in `onSaveInstanceState(Bundle)`
- ...and restore it in `onCreate(Bundle)` or `onRestoreInstanceState(Bundle)`

NAVIGATING BETWEEN SCREENS

Intent

“An intent is an abstract description of an operation to be performed.”

Intent

“An intent is an abstract descripti
operation to b ed.

IN OTHER WORDS: A COMMAND

Intent.java

- Can be used to start another Activity:

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

Intent.java

- Can be used to start another Activity:

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

- Data can be passed to the activity to be started by adding extras to the intent object:

```
intent.putExtra("key", "value");
```

Intent.java

- Can be used to start another Activity:

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

- Data can be passed to the activity to be started by adding extras to the intent object:

```
intent.putExtra("key", "value");
```

- In the started activity the intent (and its data) can be retrieved using **`getIntent()`**

Intent.java

- Can be used to start another Activity:

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

- Data can be passed to the activity to be started by adding extras to the intent object:

```
intent.putExtra("key", "value");
```

- In the started activity the intent (and its data) can be retrieved using **`getIntent()`**

- The example above shows an explicit intent

IMPLICIT INTENT

- Implicit intents allows another app to handle an action

IMPLICIT INTENT

- Implicit intents allows another app to handle an action
- Example: Call a phone number

```
Uri number = Uri.parse("tel:40201600");
Intent callIntent = new Intent(Intent.ACTION_DIAL, number);
startActivity(callIntent);
```