



Android Views and activities

Topics

Android application components

Android Layouts

Linear layout , relative layout , constraints layout

UI elements

- Text View, Edit Text ,Button , toggle button

Activity and Activity life cycle

Toast Message / Log

Example

Android App components

There are four different types of app components :

1. **Activities** : - a single screen with a user interface
2. **Services** :- run in the background (without user interface)
3. **Broadcast receivers** : enables the system to deliver events to the app outside of a regular user flow
4. **Content providers**:- enable sharing data with other applications

Android Activity

Presented to the user as full-screen windows (screens)

Single, focused thing that the user can do

Almost all activities **interact** with the user

Takes care of creating a window for you in which you can place your UI with **setContentView(View)**

Activity (Android Page/ View)

Activity (Android Page/ View)

User Interface (UI)

- ❖ *buttons , text , text inputs , images ...*
- ❖ *xml file*

Code / Functionality

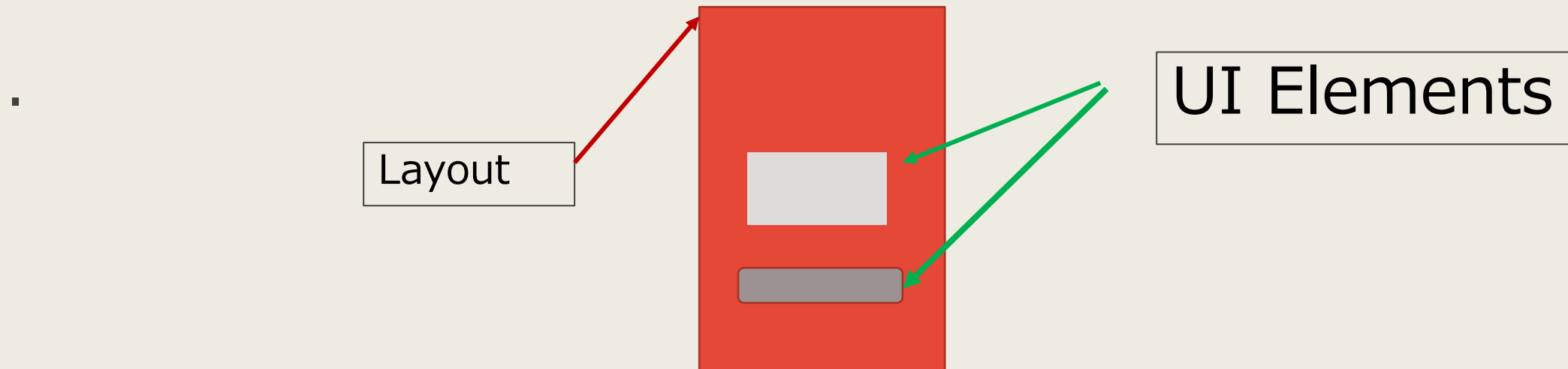
- ❖ performing an action when the user click a button
- ❖ -java /Kotlin

Android User Interface

Android UI contain:

1- **Widgets:** UI elements such as button , Text Views

2- **Layout :** a container that hold the widgets / UI elements .



View Class

View is the base class for *widgets*, which are used to create interactive UI components (buttons, text fields, etc.).

A View occupies a rectangular area on the screen.

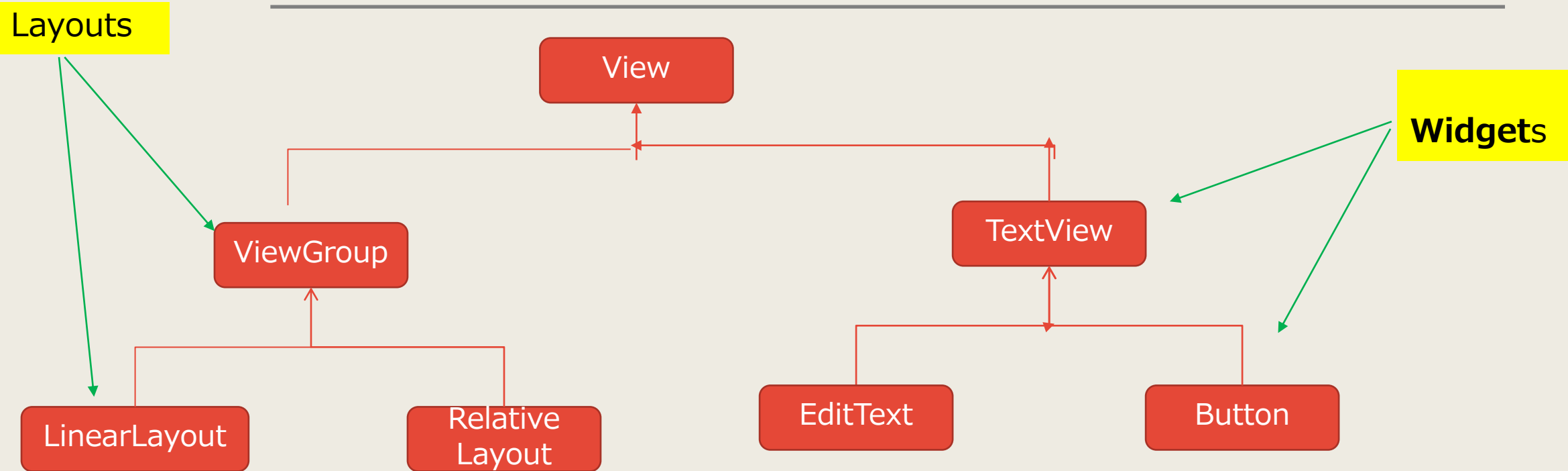
responsible for drawing and event handling.

The ViewGroup subclass is the base class for layouts

Src: <https://developer.android.com/reference/android/view/View>

Android View Hierarchy

All widgets and layout inherit from View



Layouts

There are a few different Layouts available, here is a few of them:

Linear Layout

A layout that arranges its UI elements in a single column or row.

Relative Layout

A layout where the positions can be set relative to other widgets or the parent container .

Constraint Layout

allow you to position a given widget relative to another one. You can constrain a widget on the horizontal and vertical axis

All layouts are subclasses of ViewGroup. See [here](#) for more info.

UI Elements

Text View

Edit Text

Button

Image View

Toggle button

Radio button

Checkboxes

Adding TextView in xml

<TextView

android:id="@+id/my_text_view"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:text="@string/hello_world"

/>

Assign an id

Set width and
height of the view

Getting string from string.xml

EditText

Tag for adding Edit Text

<EditText

Assign an id

android:id="@+id/my_edit_text"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:hint="@string/enter_text"

width and height

/>

Getting value from
string resources

string.xml file

Lets look at string.xml file inside values folder.

This file contain all the string resources

We can add new string resources as well

Changing text programmatically

1- get reference to UI component in layout

```
TextView tv = findViewById<TextView>(R.id.text_view)
```

// changing the text via java code

```
tv.text = "Android is cool";
```

Toast Message

We can display messages to the user by using Toasts

```
Toast.makeText(this, "hello", Toast.LENGTH_LONG).show() ;
```



The Message

Duration

Displaying debug messages in Logcat

We can display debug message in Logcat using Log class

```
Log.d("TAG", " debug message ") ;
```


Four states of Activity

1. **Active / Running** foreground of the screen currently interacting with user
2. **Visible** activity has lost focus but still presented to the user
3. **Stopped or Hidden** completely obscured by another activity
4. **Destroyed** Must be completely restarted and restored to its previous state before displaying to the user

Activity Stack

Activities in the system are managed as **activity stacks**.

When a *new activity* is started, it is usually placed on the top of *the current stack* and becomes the running activity.

The previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

There can be one or multiple activity stacks visible on screen.

Back Stack

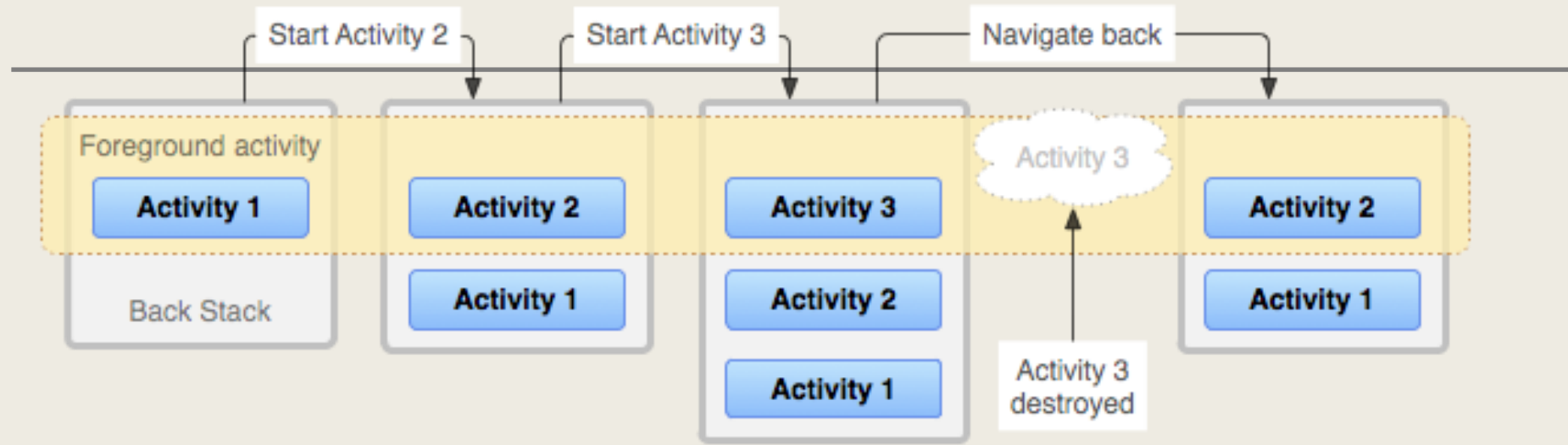
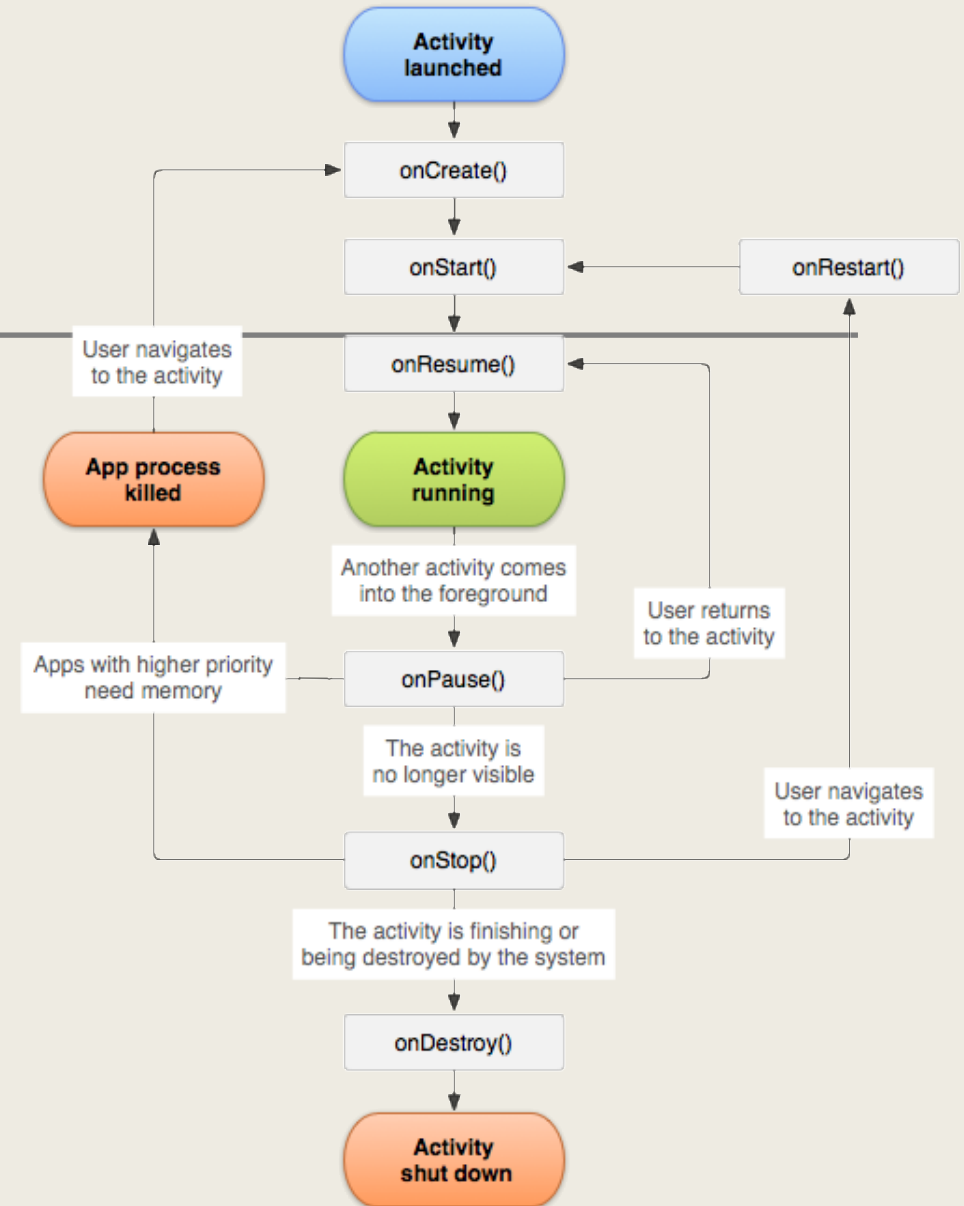


Fig (a) : Example of Activity Stack and Back Stack

Image source: developers.android.com

Activity Life cycle



Android App Life cycle

The **Activity** class provides a core set of six callbacks:

`onCreate()`

`onStart()`

`onResume()`

`onPause()`

`onStop()`

`onDestroy()`.

The system invokes each of these callbacks as an activity enters a new state.

onCreate()

Called when the activity is first created.

This is where you should do all initialization: create views, bind data to lists, etc.

Always followed by onStart().

Invoked only once during the entire activity lifecycle.

onRestart()

Called after your activity has been **stopped**, prior to it being started again.

Always followed by onStart()

onStart()

Called when the activity is becoming **visible** to the user.

onResume()

Called when the activity will start **interacting** with the user.

At this point your activity is at the **top of its activity** stack, with user input going to it.

Always followed by onPause().

onPause()

Called when the activity **loses foreground state**, is **no longer focusable** or **before transition to stopped/hidden** or destroyed state.

onStop()

Called when the activity is no longer visible to the user.

onDestroy()

called **before your activity is destroyed**.

This can happen either because the activity is finishing or because the system is temporarily destroying this instance of the activity to save space.

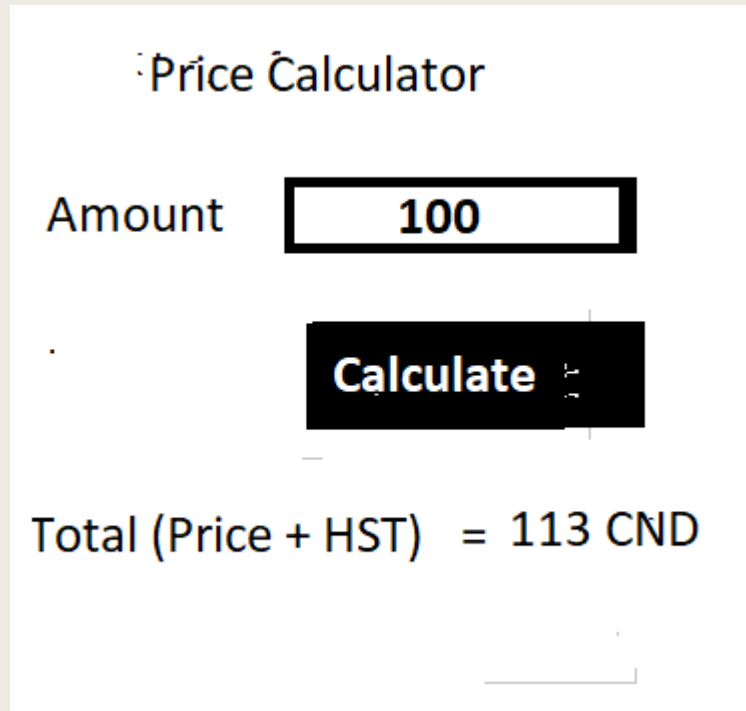
Invoked only once during the entire activity lifecycle.

Examples

Create android application the calculate and display total price

(amount + HST)

HST = 13 %



The screenshot shows a mobile application titled "Price Calculator". It features a text input field labeled "Amount" containing the value "100". Below the input field is a black button with the text "Calculate" in white. At the bottom of the screen, the result is displayed as "Total (Price + HST) = 113 CND".

References

<https://developer.android.com/guide/topics/ui/declaring-layout>

[https://developer.android.com/guide/components/fundamentals?
hl=en](https://developer.android.com/guide/components/fundamentals?hl=en)

<https://developer.android.com/studio/debug/am-logcat>