LẬP TRÌNH ANDROID CƠ BẢN

BÀI 2 : LAYOUT

NỘI DUNG

- 📖 Phần I: Layout trong Android
- 📖 Phần II: Các widget cơ bản
- 📖 Phần III: Hộp thoại thông báo : Toast, Alert Dialog & Custom Dialog

BÀI 2 : LAYOUT

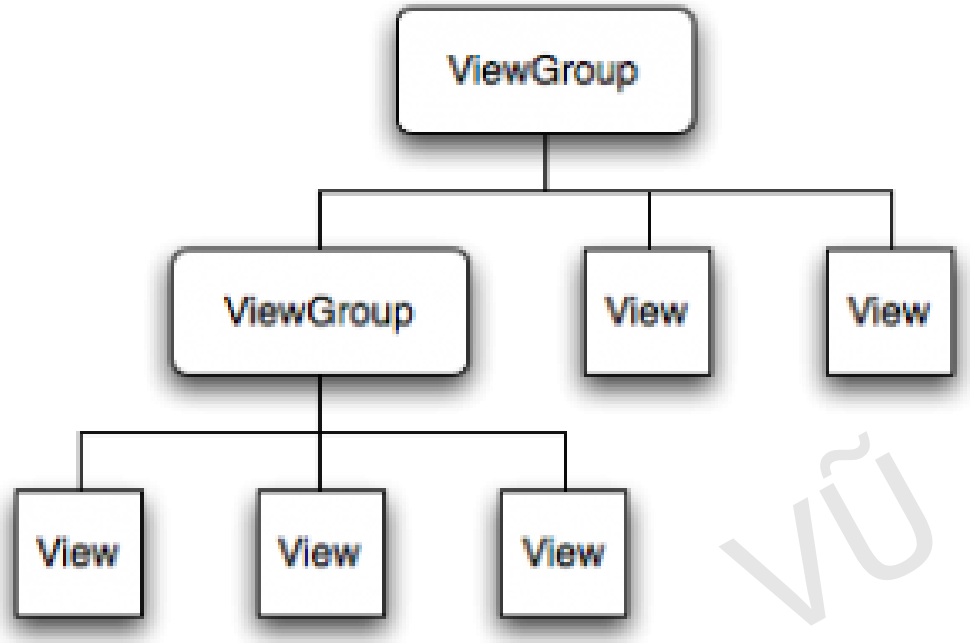
PHẦN I :

LAYOUT TRONG ANDROID

Thiết kế giao diện người dùng

- Càng đơn giản càng tốt
- Dành nhiều thời gian tìm hiểu nhu cầu của khách hàng về giao diện
- Sử dụng điều khiển giao diện chuẩn

Cây phân cấp View (View Hierarchy)

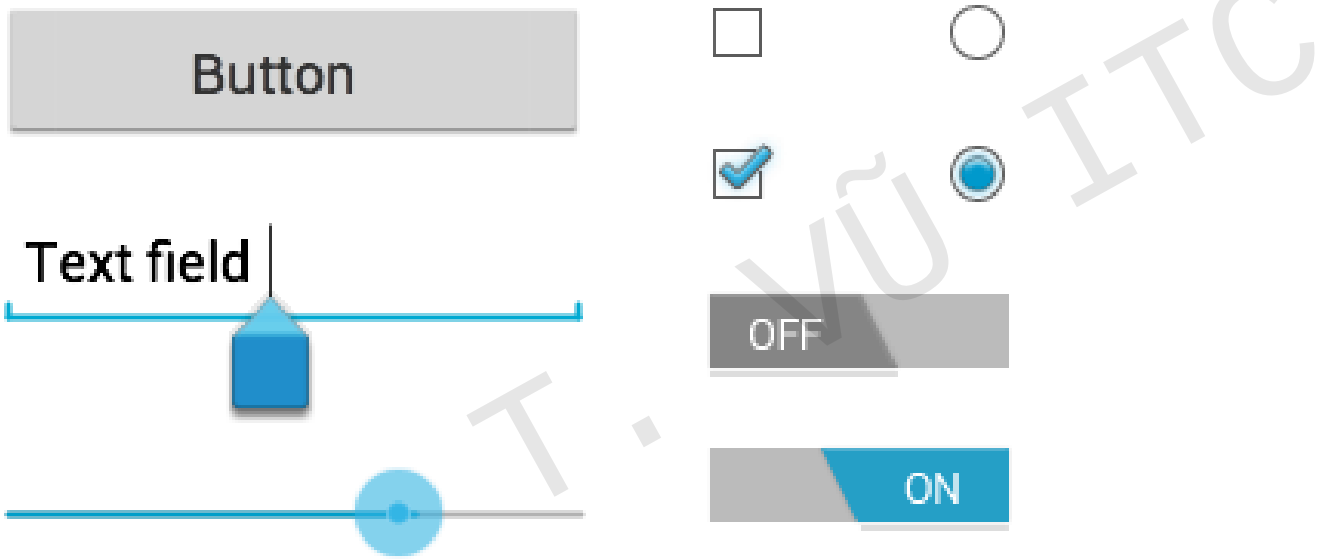


Cây phân cấp View (View Hierarchy)

- View: đơn vị cơ bản của giao diện người dùng
  - Widgets: android.widget.*
  - Là lá của cây phân cấp View
- ViewGroup: định nghĩa layout
  - Nằm trong android.widget.*
  - Định nghĩa nơi chứa các Views (hoặc View Group) con

Ví dụ Widget (view)

- Button
- EditText
- CheckBox và RadioButton
- TextView, ImageView,…



Ví dụ về Layout (ViewGroup)

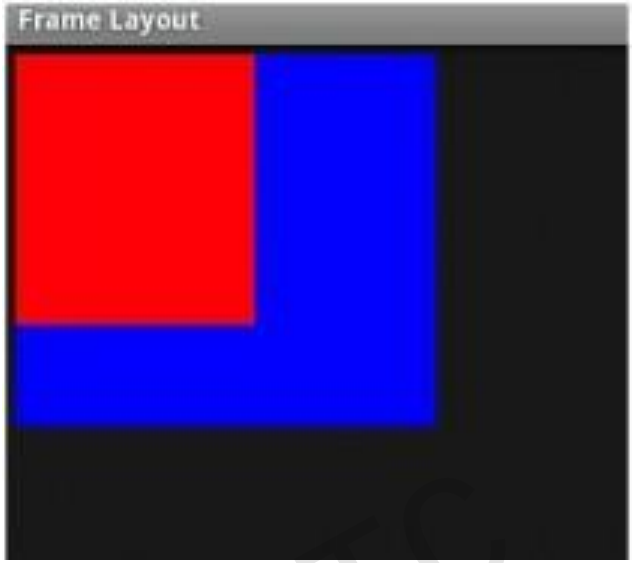
- LinearLayout
- RelativeLayout
- FrameLayout
- TableLayout
- Absolute Layout
- ScrollView Layout
- ....

FRAME LAYOUT

- Là loại Layout đơn giản nhất. Tất cả các phần tử khi thêm vào FrameLayout đều nằm ngay góc trên bên trái của màn hình; chúng ta **không** thể chỉ định một vị trí khác để bố trí các thành phần lên nó.
- View nào vào trước thì nằm phía sau, và ngược lại

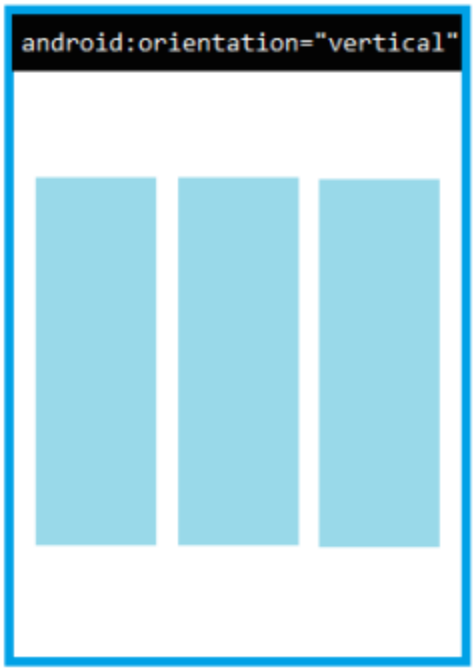
FRAME LAYOUT

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout android:id="@+id/mainlayout"
android:layout_height="fill_parent"
android:layout_width="fill_parent" ">
    <ImageView
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:padding="5px"
    android:src="@drawable/blue"/>
    <ImageView
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:padding="5px"
    android:src="@drawable/red"/>
</FrameLayout>
```



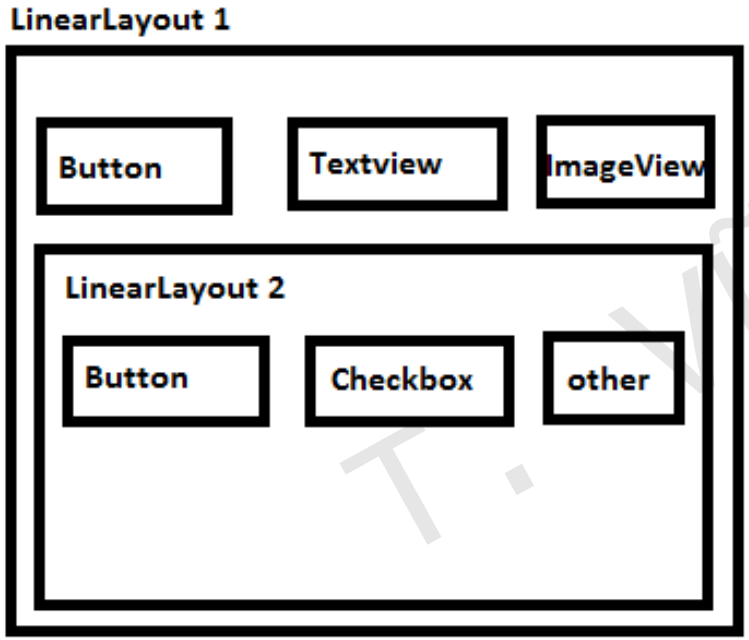
LinearLayout

- Là layout sắp xếp các View con trong nó theo duy nhất một chiều, ngang hoặc dọc dựa vào giá trị của thuộc tính android:orientation
  - Orientation = vertical hoặc horizontal
- Có thể lồng nhiều layout phức tạp



LinearLayout lồng nhau

- LinearLayout lồng nhau là một cách để dễ dàng tạo các giao diện chung
- Chú ý: nếu lồng nhau mà số cấp lớn hơn hoặc bằng 5 sẽ làm cho việc tải giao diện chậm hơn



Trọng lượng của Layout (layout weight)

- **layout weight** cho phép tạo LinearLayout với kích thước tương đối dựa vào trọng số
- Default = 0 – không gian tối thiểu để hiển thị tất cả nội dung

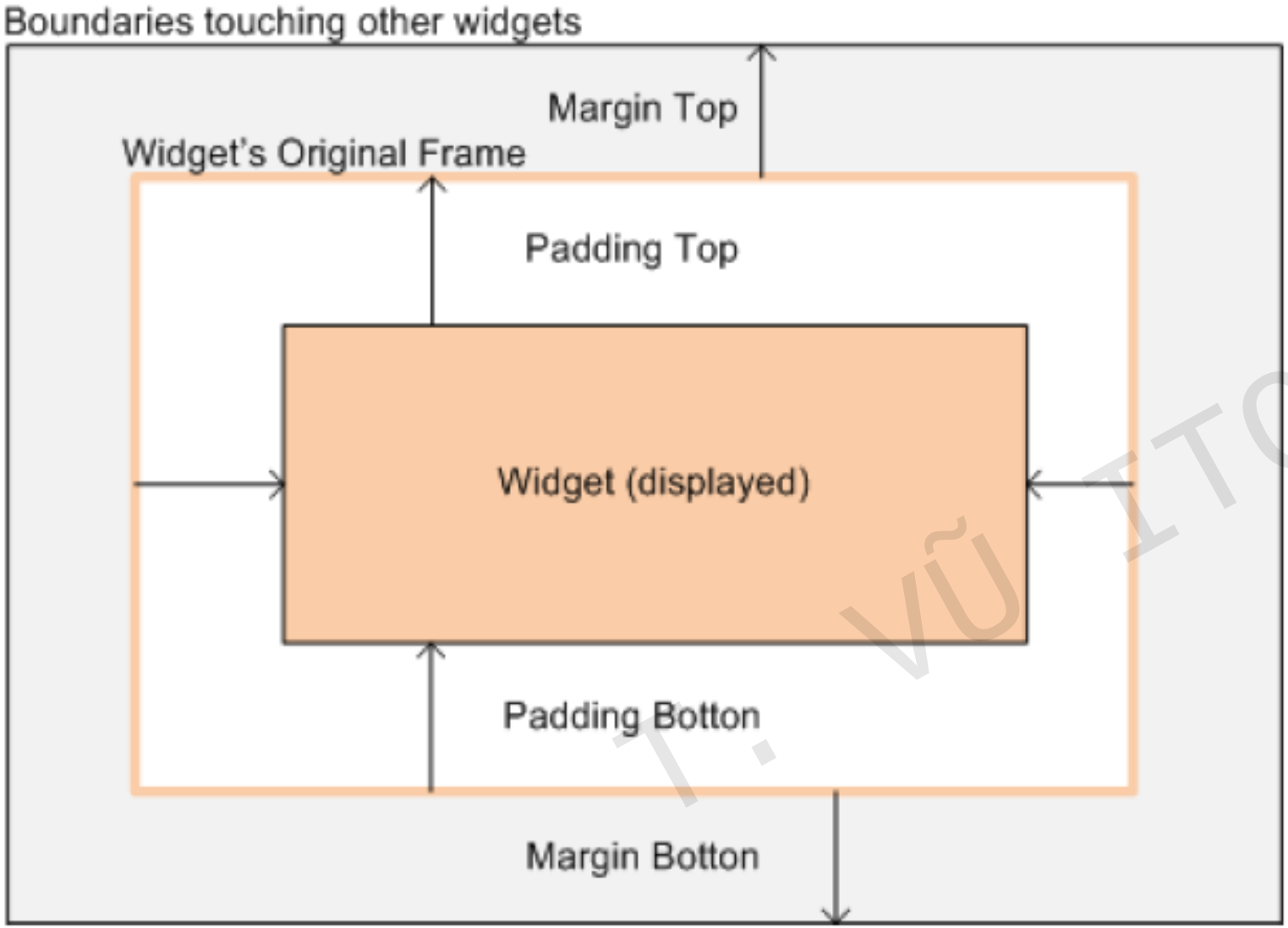
Ví dụ

- Chúng ta sẽ định nghĩa layout cho giao diện sau như thế nào?

- thuộc tính android:layout_gravity chỉ định view sẽ nằm về phía nào trong layout
  - Phân biệt android:gravity chỉ định nội dung văn bản sẽ nằm như thế nào trong view

android:gravity="center"

android:layout_gravity="center"

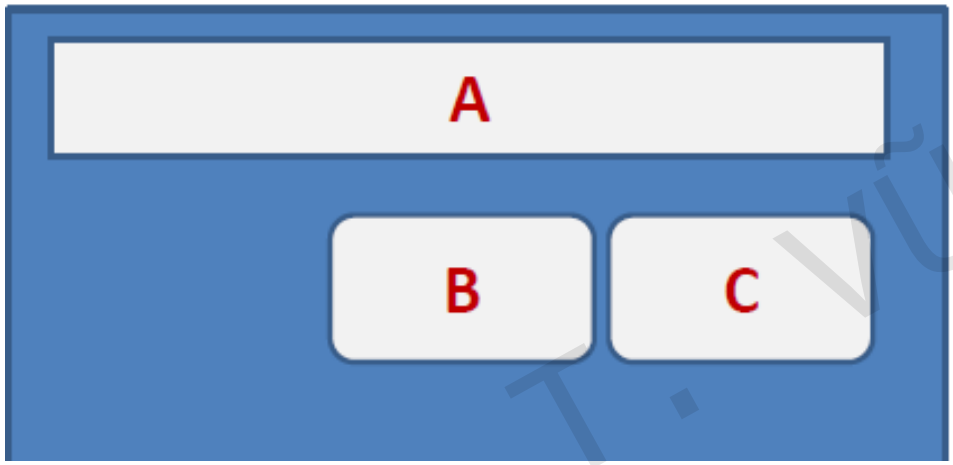

Boundaries touching other widgets
Margin Top
Widget's Original Frame
Padding Top
Widget (displayed)
Padding Botton
Margin Botton

RelativeLayout

- RelativeLayout cho phép sắp xếp các control theo vị trí tương đối giữa các control khác trên giao diện (kể cả control chứa nó).
- Thường ta sẽ dựa vào Id của các control khác để sắp xếp theo vị trí tương đối.
- Do đó khi làm RelativeLayout phải chú ý là đặt Id control cho chuẩn xác, nếu sau khi Layout xong mà lại đổi Id của các control thì giao diện sẽ bị xáo trộn (do đó nếu đổi ID thì phải đổi luôn các tham chiếu khác sao cho khớp với Id mới đổi).

Ví dụ:

- A đứng trên đầu ,
- C bên dưới A và ở phía bên phải,
- B bên dưới A và bên trái C



Một số thuộc tính sắp xếp widget với layout chứa nó:

- **android:layout_alignParentTop:** chỉ ra rằng widget phải được đặt ở đầu của layout mà nó nằm.
- **android:layout_alignParentBottom** đặt ở dưới cùng
- **android:layout_alignParentLeft** đặt ở bên trái
- **android:layout_alignParentRight** : đặt ở bên phải
- **android:layout_centerInParent** : đặt ở trung tâm
- **android:layout_centerHorizontal:** đặt ở trung tâm theo chiều ngang
- **android:layout_centerVertical:** đặt ở trung tâm theo chiều dọc

Một số thuộc tính sắp xếp widget với các widget khác:

- **android:layout_above** chỉ ra rằng widget phải được đặt ở trên của widget tham chiếu.
- **android:layout_below** chỉ ra rằng widget phải được đặt ở dưới của widget tham chiếu.
- **android:layout_toLeftOf** chỉ ra rằng widget phải được đặt ở bên trái của widget tham chiếu.
- **android:layout_toRightOf** chỉ ra rằng widget phải được đặt ở bên phải của widget tham chiếu.

- **android:layout_alignTop:** làm cho top của widget này căn bằng với top của widget tham chiếu
- **android:layout_alignBottom** làm cho cạnh dưới của widget này căn bằng với cạnh dưới của widget tham chiếu
- **android:layout_alignLeft** làm cho cạnh trái của widget này căn bằng với cạnh trái của widget tham chiếu
- **android:layout_alignRight** làm cho cạnh phải của widget này căn bằng với cạnh phải của widget tham chiếu

Để sắp xếp các Widget, ta cần phải làm những bước sau:

- Gán Id cho tất cả các phần tử control (android:id)
- Cú pháp: @+id/… để đặt id cho từng control, ví dụ cho thẻ EditText là android:id = “@+id/editUserName”
- Để bố trí control khác liên quan đến control này, ta cũng sẽ sử dụng giá trị (@+id/…). Ví dụ đặt một control dưới hộp EditText ở trên ta có câu lệnh:

android: layout_below = “@+id/ediUserName”

RELATIVELAYOUT

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:paddingLeft="16dp"
   android:paddingRight="16dp" >
   <EditText
      android:id="@+id/name"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:hint="@string/reminder" />
   <Spinner
      android:id="@+id/dates"
      android:layout_width="0dp"
      android:layout_height="wrap_cotent"
      android:layout_below="@id/name"
      android:layout_alignParentLeft="true"
      android:layout_toLeftOf="@+id/times" />
   <Spinner
      android:id="@id/times"
      android:layout_width="96dp"
      android:layout_height="wrap_content"
      android:layout_below="@id/name"
      android:layout_alignParentRight="true" />
<Button
      android:layout_width="96dp"
      android:layout_height="wrap_content"
      android:layout_below="@id/times"
      android:layout_alignParentRight="true"
      android:text="@string/done" />
</RelativeLayout>
```

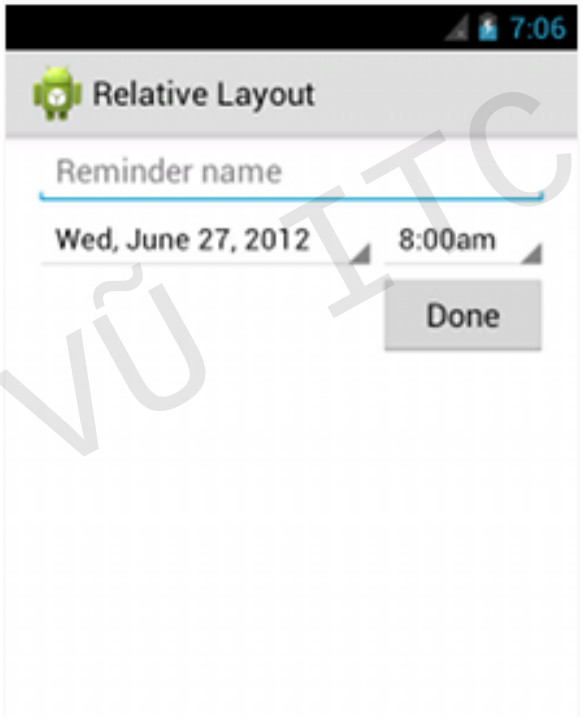


TABLE LAYOUT

- là một viewgroup dùng để nhóm các view con thành các dòng và cột.
- Table layout không hiển thị lưới phân chia dòng cột
- Thuộc tính TableRow chỉ định số dòng, nhưng không có chỉ định số cột. Số cột phụ thuộc vào số lượng view thêm vào
- Kích thước cột phụ thuộc vào độ rộng của view nó chứa

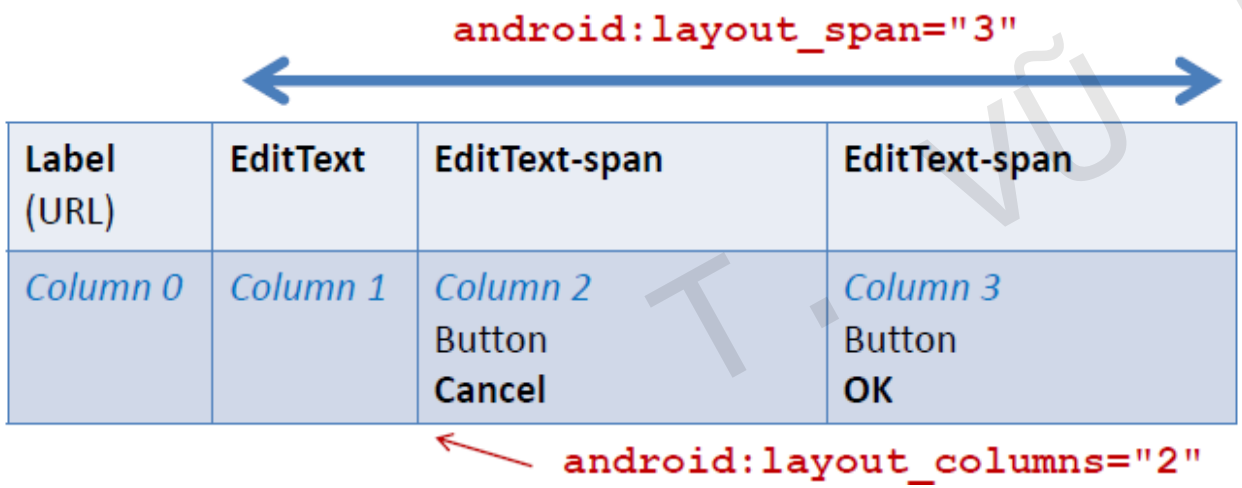
0		1	
0		1	2
0	1	2	3

TABLE LAYOUT

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
  xmlns:android="http://schemas.android.com/apk/
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:stretchColumns="*">
      <TableRow>
          <TextView android:text="Open..."
              android:padding="3dip" />
          <TextView android:text="Ctrl-O"
              android:gravity="right"
              android:padding="3dip" />
      </TableRow>
      <TableRow>
          <TextView android:text="Save As..."
             android:padding="3dip" />
          <TextView android:text="Ctrl-Shift-S"
              android:gravity="right"
              android:padding="3dip" />
      </TableRow>
</TableLayout>
```

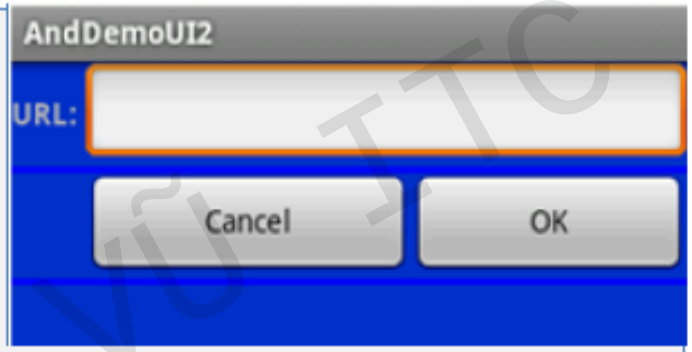
- Tuy nhiên, ta có thể trộn cột bằng thuộc tính layout_span

```
<TableRow>
    <TextView android:text="URL:" />
    <EditText
    android:id="@+id/entry"
    android:layout_span="3" />
</TableRow>
```



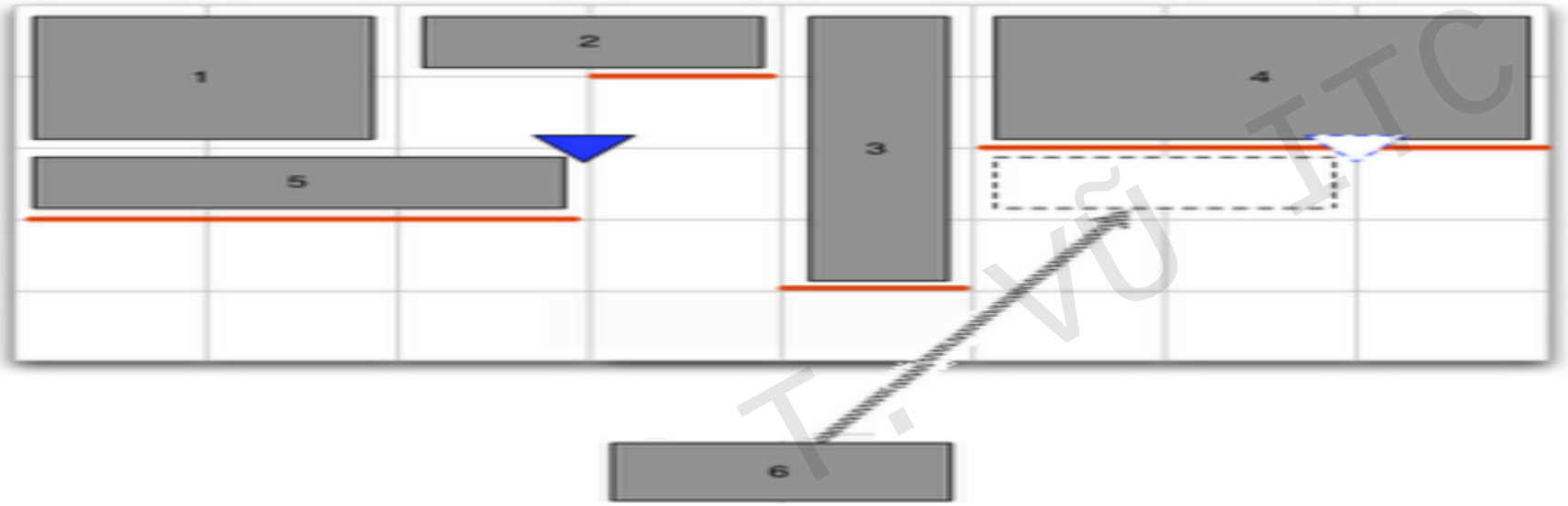
- Thuộc tính stretch_columns kéo dãn view trong cell bằng với kích thước của cột nếu view có kích thước < cột. Giá trị có thể là số 1,2,3 (cột 1,2,3) hoặc "*" (tất cả các cột)
- Ví dụ:

```
...
<TableLayout
android:id="@+id/myTableLayout"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="#ff0033cc"
android:orientation="vertical"
android:stretchColumns ="2,3,4"
xmlns:android="http://schemas.android.com/apk/res/android"
>
```



GRID LAYOUT

- Là dạng layout theo lưới, chia không gian thành các dòng và cột, giao giữa dòng và cột gọi là ô (cell)
- Không giống TableLayout, Grid cho phép trộn các ô lại với nhau theo chiều đứng hoặc ngang



GRID LAYOUT

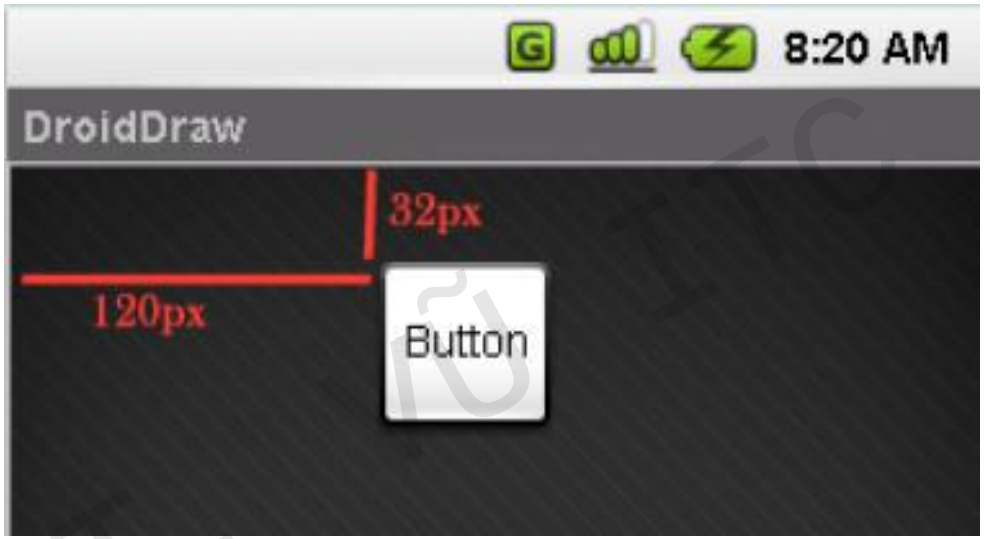
- Layout_column: chỉ định số cột
- Layout_row: chi định số dòng
- Layout_columnSpan: trộn cột
- Layout_rowSpan: trộn dòng
- Layout_rowWeight: thành phần con chia nhau lấp đầy không gian còn lại theo tỉ lệ trọng lượng của chúng

ABSOLUTE LAYOUT

- Xác định vị trí các view trên layout dựa vào tọa độ (x,y) của chúng

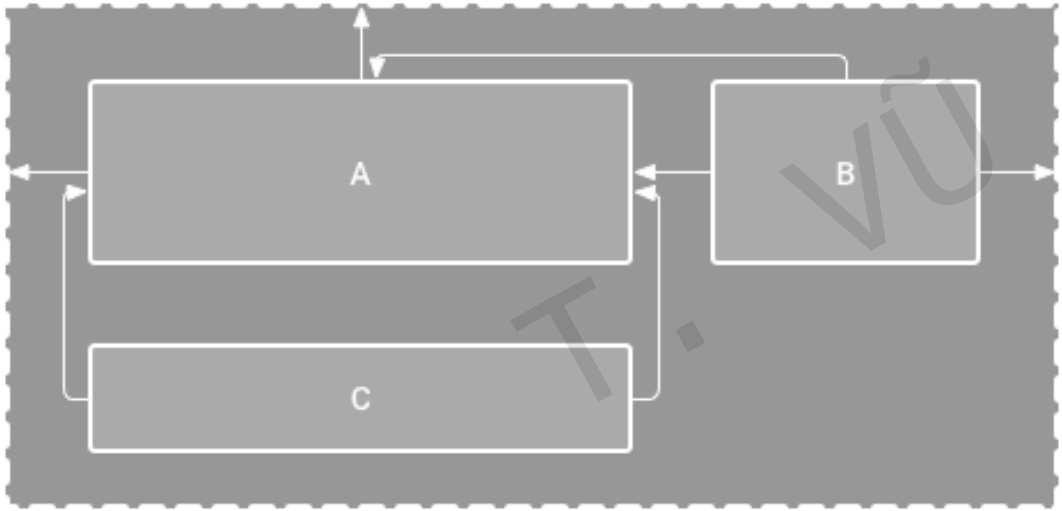
```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
        android:id="@+id/myAbsoluteLayout"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        xmlns:android="http://schemas.android.

<Button
         android:id="@+id/myButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        android:layout_x="120px"
        android:layout_y="32px"
        >
</Button>
</AbsoluteLayout>
```



CONSTRAINT LAYOUT

- Giúp định vị, sắp xếp các View con dựa trên sự ràng buộc liên hệ giữa các View với nhau
- Mỗi view trong **ConstraintLayout** để định vị được chính xác cần tối thiểu 2 ràng buộc, một theo phương ngang (X) và một theo phương đứng (Y)

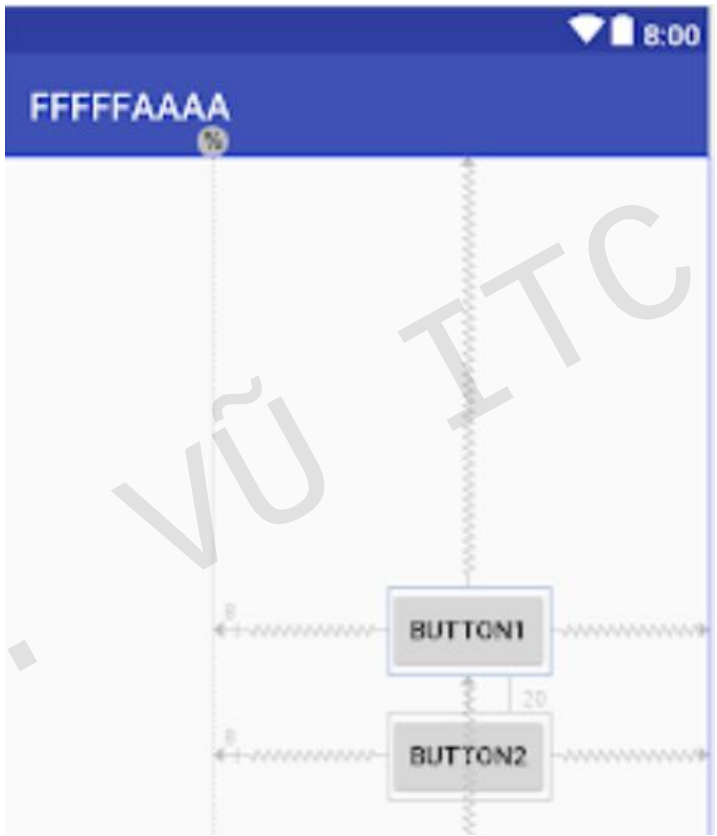


❑ Một số thuộc tính cơ bản

Ràng buộc	Ý nghĩa ràng buộc
layout_constraintLeft_toLeftOf	Rang buộc cạnh trái của phần tử tới phần tử
layout_constraintLeft_toRightOf	Bên trái với bên phải của phần tử chỉ ra
layout_constraintRight_toLeftOf	Bên phải với bên trái
layout_constraintRight_toRightOf	Phải với phải
layout_constraintTop_toTopOf	Cạnh trên với cạnh trên
layout_constraintTop_toBottomOf	Cạnh trên nối với cạnh dưới
layout_constraintBottom_toTopOf	Dưới với trên
layout_constraintBottom_toBottomOf	Dưới với dưới
layout_constraintBaseline_toBaselineOf	Trùng Baseline
layout_constraintStart_toEndOf	Bắt đầu - Kết húc
layout_constraintStart_toStartOf	Bắt đầu - Bắt đầu
layout_constraintEnd_toStartOf	Cuối với bắt đầu
layout_constraintEnd_toEndOf	Cuối với cuối

- Phần tử Guideline: Ta có thể một đường kẻ ẩn trong ConstraintLayout nằm ngang hoặc đứng nó như là một View con để các View khác ràng buộc đến nếu muốn

```
<android.support.constraint.Guideline
    android:id="@+id/guideline_1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    app:layout_constraintGuide_percent="0.3" />
```



BÀI 2 : LAYOUT

PHẦN II :
CÁC WIDGET CƠ BẢN

1. TextView

- Trong Android, một label được sử dụng là TextView.
- Một TextView dùng để hiển thị thông tin và không cho phép người dùng chỉnh sửa.
- Một số thuộc tính của control:
  - Cần thiết lập id cho control.
  - layout_width, layout_height cũng nên thiết lập cho control (bắt buộc)
  - Để thay đổi màu nền dùng background, thay đổi màu chữ dùng textColor…

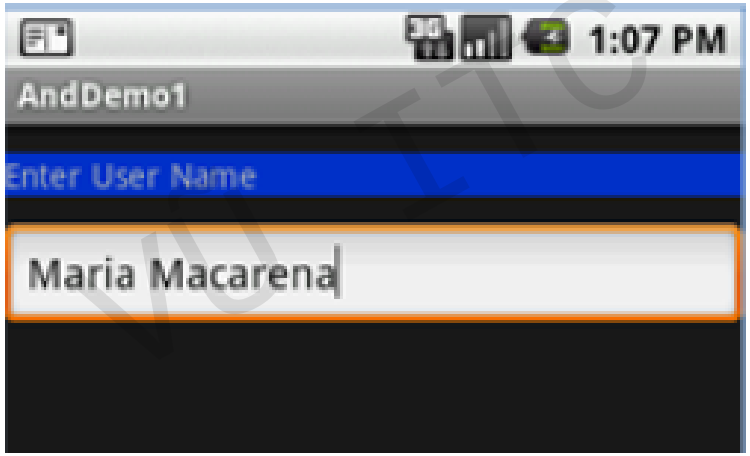
CÁC WIDGET CƠ BẢN


```
<TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="HELLO ANDROID"
        android:textStyle="bold"
        android:textSize="25sp"
        android:background="#334455"
        android:textColor="#FFF"
        android:padding="5"
        android:layout_margin="5"
        android:fontFamily="tahoma"
        android:gravity="center_horizontal"
       />
```

2. EditText

- Thẻ EdtitText (hoặc textBox) là một trường hợp mở rộng của TextView, cho phép cập nhật, chỉnh sửa dữ liệu

```
<EditText
android:id="@+id/txtUserName"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:textSize="18sp"
android:autoText="true"
android:capitalize="words"
android:hint="First Last Name"
>
</EditText>
```



Một số thuộc tính:

- **inputType**: chỉ định cách nhập liệu
  - Text: nhập văn bản đơn thuần
  - textAutoCorrect: sửa lỗi chính tả.
  - textCapWords: Chữ đầu mỗi từ thành hoa
  - Number: chỉ nhập số
  - …
- **android:password**="**true**“: hiện password

- Lấy control theo Id:

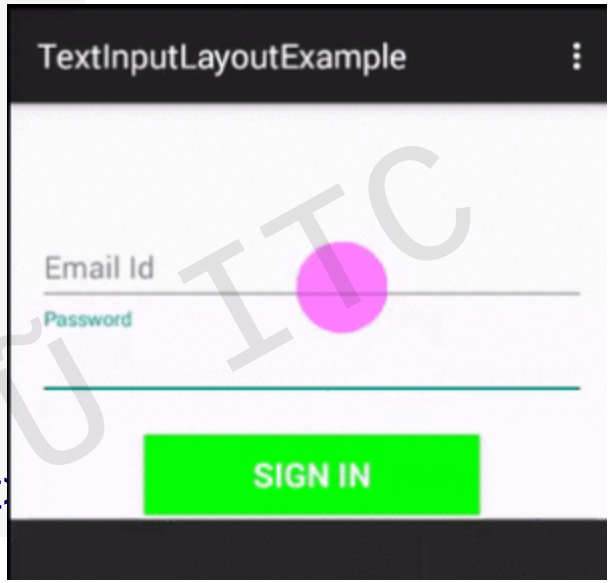
  EditText txtbox=(EditText) findViewById(R.id.editText1);
- Lấy dữ liệu bên trong EditText:

  String msg=txtBox.getText().toString()

TextInputLayout (Floating Label Editext)

- Là 1 dạng khác của Editext có sẵn nhãn gợi ý

```
<android.support.design.widget.TextInputLayout
android:id="@+id/simpleTextInputLayout"
android:layout_width="wrap_content"
android:layout_height="wrap_content">
    <EditText
    android:id="@+id/simpleEditText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="Password" />
    </android.support.design.widget.TextInputLayout
```



3. Button

- Thẻ Button cho phép người dùng thực hiện một hành động click vào nút. Sự kiện click có 2 loại:
  - Click
  - Long Click
- Một button có thể chứa text và icon.
- Với text, sử dụng thẻ button:

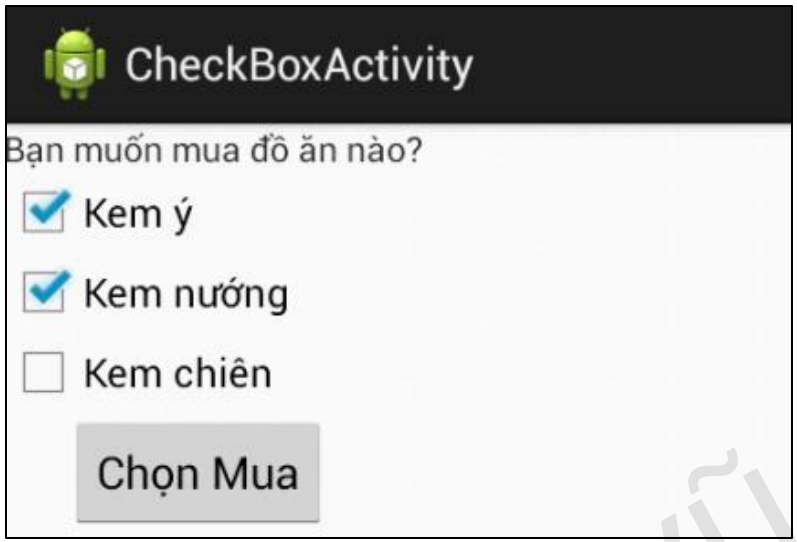
```
<Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/button_text“
        android:click=“xuly“
        ... />
```

4. Image

- ImageView và ImageButton thường được dùng để nhúng hình ảnh cho ứng dụng.
- Cả hai thẻ này đều tương tự như TextView và Button.
- Các thẻ này đều sử dụng thuộc tính android:src hoặc android:background để hiển thị ảnh.
- Ảnh được sử dụng sẽ được tham chiếu tới thư mục /res/*drawable/*
- ImageButton là một lớp con của ImageView và thêm một số chuẩn của Button để thực hiện các sự kiện Click

5. Checkbox

- Checkbox: có hai trạng thái: checked & unchecked.
- Ví dụ:



6. RadioButton

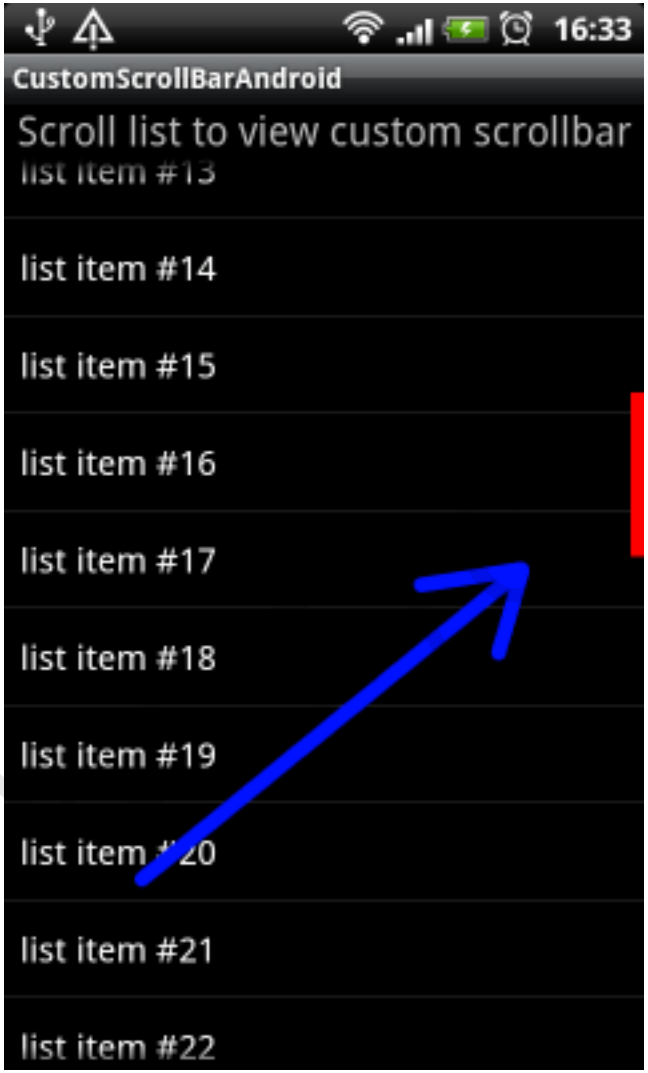
- Cũng có hai trạng thái: checked & unchecked.
- Sử dụng <RadioGroup> để nhóm các radioButton có cùng nhóm lại với nhau

```
<RadioGroup
    android:id="@+id/radioGroup1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="25dp"
    android:layout_marginTop="39dp" >
    <RadioButton
        android:id="@+id/radrathailong"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true"
        android:text="Rất hài lòng" />
    <RadioButton
        android:id="@+id/radhailong"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hài lòng" />
    <RadioButton
        android:id="@+id/radtamchapnhan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tạm chấp nhận" />
    <RadioButton
        android:id="@+id/radthayghe"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Thấy ghê" />
</RadioGroup>
```

7. Scroll View

- **ScrollView:** Nếu dữ liệu vượt quá kích thước của màn hình, chúng ta có thể dùng ScrollView để cuộn dữ liệu

Các widget cơ bản



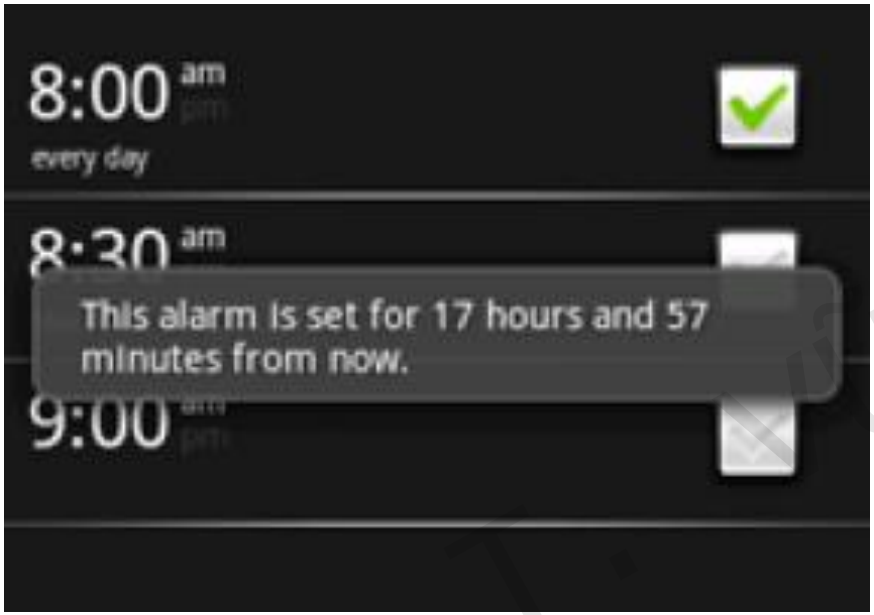
BÀI 2 : LAYOUT

PHẦN III :

HỘP THOẠI THÔNG BÁO

TOAST NOTIFICATION

- Là 1 dạng thông báo hiện lên trên màn hình trong một khoảng thời gian nhất định
- Không thể tương tác
- Được tạo ra từ 1 Activity hoặc Service



TOAST NOTIFICATION

```
Toast toast=Toast.makeText(StylesActivity.this, "text",
                           Toast.LENGTH_SHORT);
toast.setGravity(Gravity.CENTER, 0, 0);
toast.show();
```

- Short form

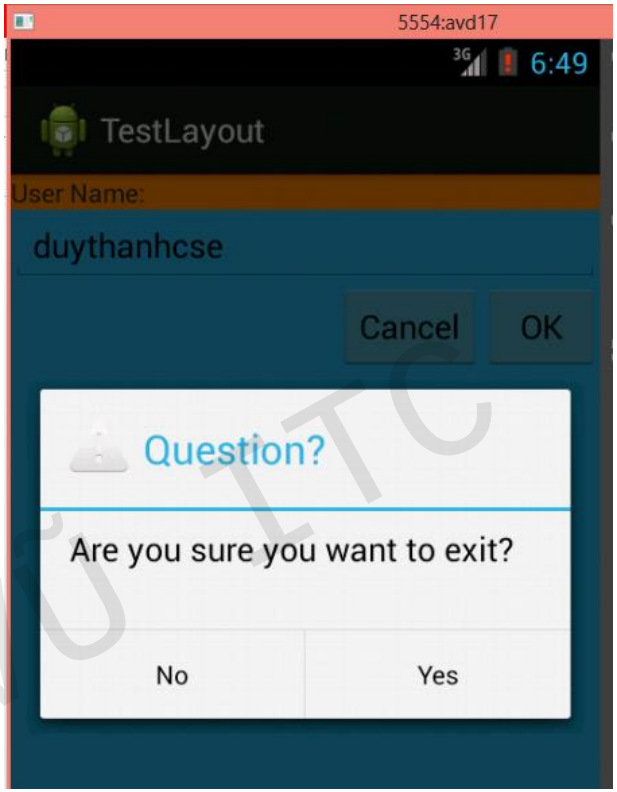
```
Toast.makeText(context, text, duration).show();
```

- **Context**: Application Context hoặc Activity context

- Thời gian hiển thị thông báo (**duration**):
  - **Toast.LENGTH_SHORT** : 2 giây
  - **Toast.LENGTH_LONG :** 3.5 giây

ALERT DIALOG

- Cũng là 1 dạng hiển thị thông báo có điều kiện, có thể là:
  - Xác nhận (confirm): Yes/No Message
  - Hiển thị tiến trình
  - Chọn 1 item từ danh sách
  - …
- Sử dụng lớp AlertDialog.Builder
- Phương thức:
  - SetTitle: tiêu đề
  - setMessage: thông báo
  - setIcon: biểu tượng
  - setCancelable: true/false
  - setNegativeButton: thêm nút cancel
  - setPositiveButton: thêm nút ok
  - setNeutralButton: 1 nút khác
  - setOnCancelListener



ALERT DIALOG

```
AlertDialog.Builder builder=new AlertDialog.Builder(
        MainActivity.this,
        AlertDialog.THEME_HOLO_LIGHT);
builder.setMessage("Are you sure you want to exit?");
builder.setTitle("Question?");
builder.setIcon(android.R.drawable.ic_dialog_alert);
builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {

    @Override
    public void onClick(DialogInterface dialog, int which) {
        // TODO Auto-generated method stub
        MainActivity.this.finish();
    }
});
builder.setNegativeButton("No", new DialogInterface.OnClickListener() {

    @Override
    public void onClick(DialogInterface dialog, int which) {
        // TODO Auto-generated method stub
        dialog.cancel();
    }
});
AlertDialog dialog=builder.create();
dialog.show();
```