Fpt University

FPT POLYTECHNIC

THIẾT KẾ GIAO DIỆN TRÊN ANDROID

Bài 2: Tổng quan về UI (tiếp)

www.poly.edu.vn

Nội dung bài học

2. Các Layout cơ bản (tiếp)
   - Table layout
   - Relative Layout
   - Absolute Layout
   - ScrollView Layout
3. Style and Themes

2. Các layout cơ bản

- **TableLayout**
  - Cho phép sắp các control theo dạng lưới (dòng và cột).
  - Các cột có thể thu nhỏ hoặc dãn rộng tùy thuộc vào nội dung chứa.
  - TableLayout làm việc với các TableRow .
  - TableLayout sẽ xem dòng nào có số lượng control nhiều nhất để xác định rằng nó có bao nhiêu cột (lấy dòng có số lượng control nhiều nhất làm số cột chuẩn).

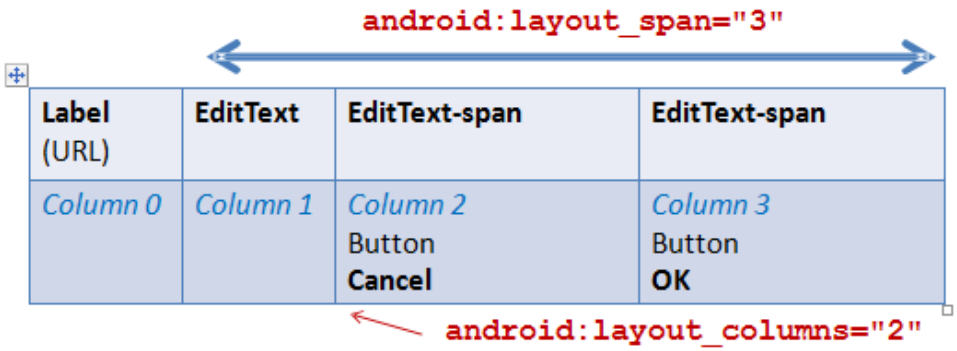
0		1	
0		1	2
0	1	2	3

2. Các layout cơ bản

- **TableLayout (tiếp)**
  - Dùng **layout_span** để trộn các cột: thẻ EditText sẽ được trộn bởi 3 cột ở hàng đầu như hình dưới.

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

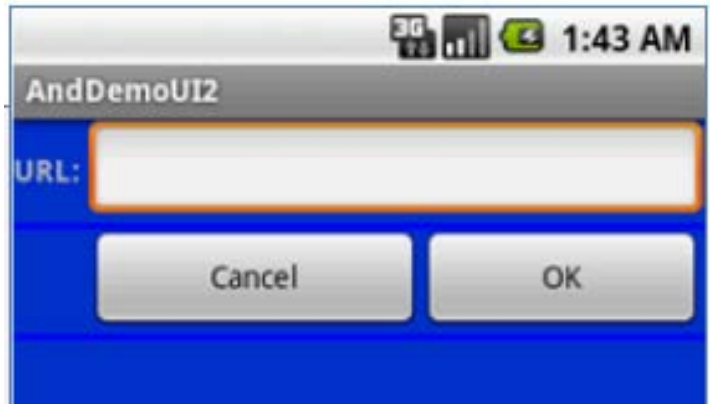
  - Dùng **layout_column** để di chuyển vị trí của control đến một cột nào đó trên 1 dòng. Các cột sẽ đánh số bắt đầu từ 0.



2. Các layout cơ bản

- **TableLayout (tiếp)**
  - Nếu để mặc định mỗi cột sẽ tự động dãn theo kích cỡ của các control mà nó chứa.
  - Ta có thể dùng thuộc tính **stretchColumns** để dãn đều các control, các cell (ta thường dùng dấu "*"):
  - Ví dụ dưới: cột 2, 3,4 sẽ được dãn đều như nhau

```
<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"
>
```

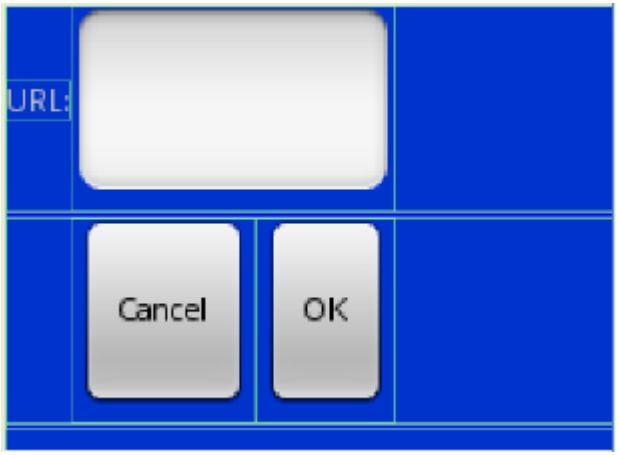


2. Các layout cơ bản

- **TableLayout (tiếp)**

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
android:id="@+id/myTableLayout"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="#ff0033cc"
android:orientation="vertical"
xmlns:android="http://schemas.android.com/apk/res/android"
>
<TableRow>
<TextView android:text="URL:" />
<EditText android:id="@+id/ediUrl“
android:layout_span="3"/>
</TableRow>
<View
android:layout_height="3dip"
android:background="#0000FF" />
<TableRow>
<Button android:id="@+id/cancel"
android:layout_column="2“
android:text="Cancel" />
<Button android:id="@+id/ok"
android:text="OK" />
</TableRow>
<View android:layout_height="3dip"
android:background="#0000FF" />
</TableLayout>
```

…



2. Các layout cơ bản

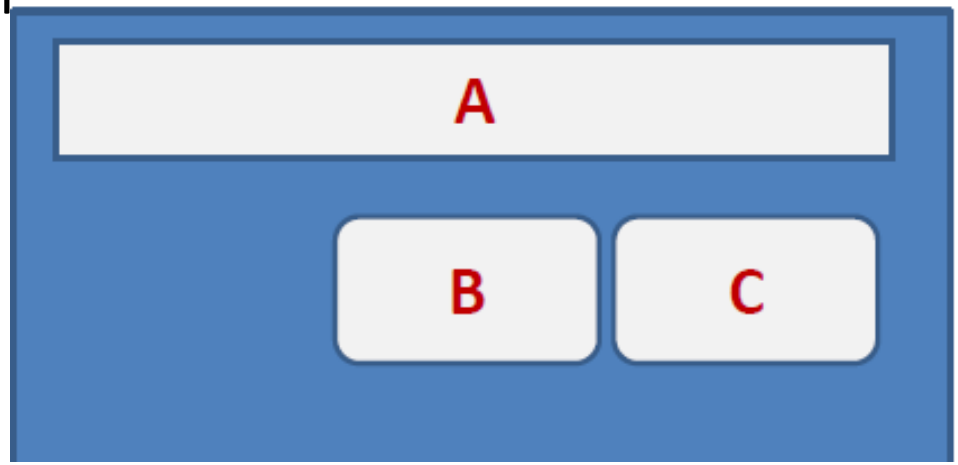
- **RelativeLayout (tiếp)**
  - RelativeLayout cho phép sắp xếp các control theo vị trí tương đối giữa các control khác trên giao diện (kể cả control chứa nó).
  - Thường ta sẽ dựa vào Id của các control khác để sắp xếp theo vị trí tương đối.
  - Do đó khi làm RelativeLayout phải chú ý là đặt Id control 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).

2. Các layout cơ bản

- **RelativeLayout (tiếp)**

  Ví dụ cho thấy

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



2. Các layout cơ bản

- **RelativeLayout (tiếp)**

  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

2. Các layout cơ bản

- **RelativeLayout (tiếp)**

Một số thuộc tính sắp xếp widget với các widget hoặc control 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.

2. Các layout cơ bản

- **RelativeLayout (tiếp)**
  - **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

2. Các layout cơ bản

- **RelativeLayout (tiếp)**

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

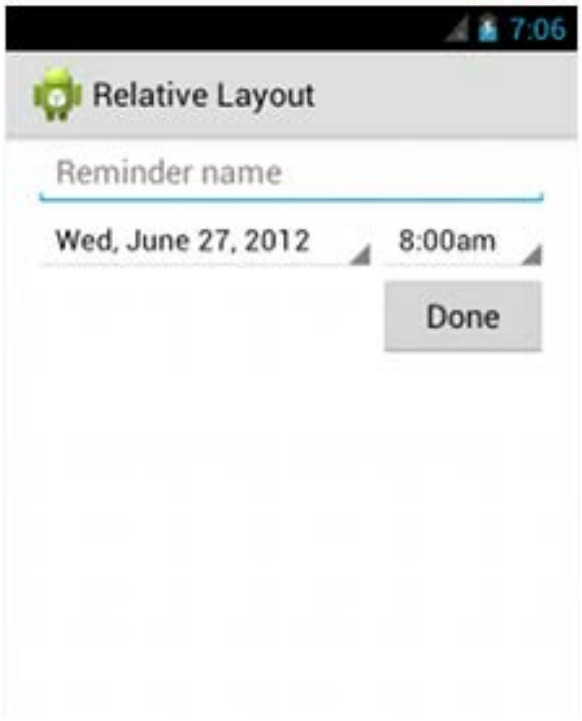
- Gán Id cho tất cả các phần tử control (android:id)
- Cú pháp: @+id/… để đặt id cho từng control, ví dụ cho thẻ EditText là android:id = “@+id/editUserName”
- Để bố trí control khác liên quan đến control này, ta cũng sẽ sử dụng giá trị (@+id/…). Ví dụ đặt một control dưới hộp EditText ở trên ta có câu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

android: layout_below = “@+id/ediUserName”

2. Các layout cơ bản

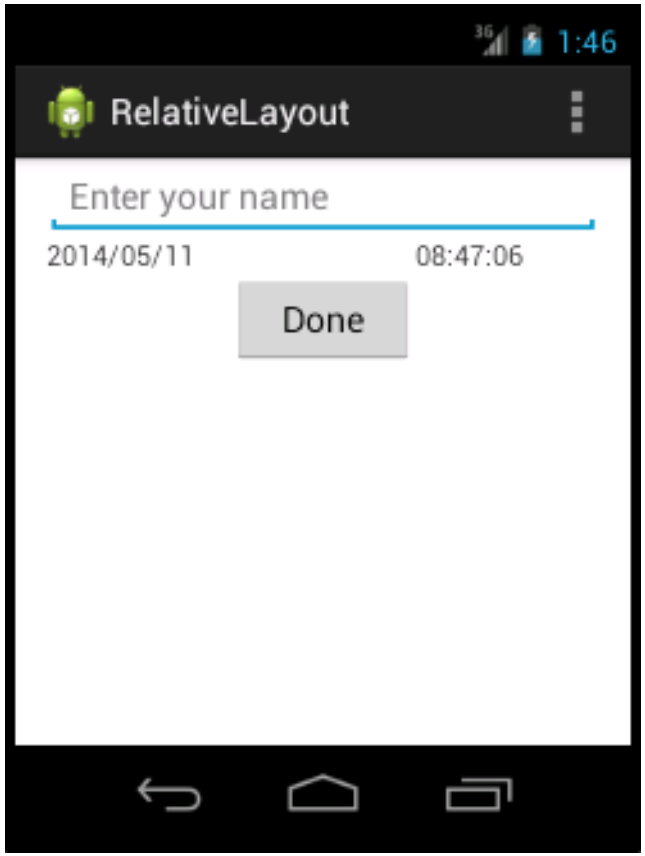
- **RelativeLayout (tiếp)**

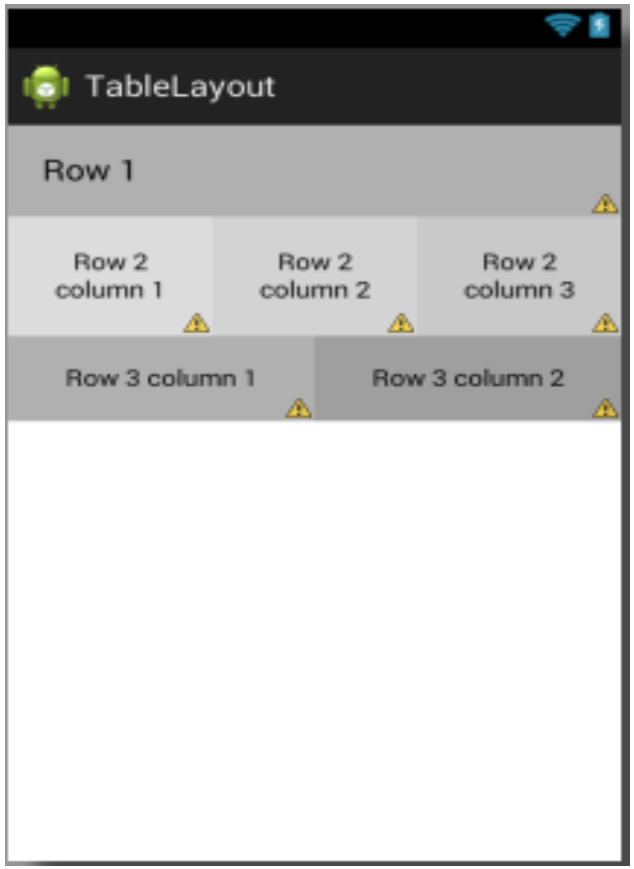
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android
"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:paddingLeft="16dp"
   android:paddingRight="16dp" >
   <EditText
      android:id="@+id/name"
      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_content"
      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>
```



2. Các layout cơ bản

- **Demo → TableLayout và RelativeLayout**

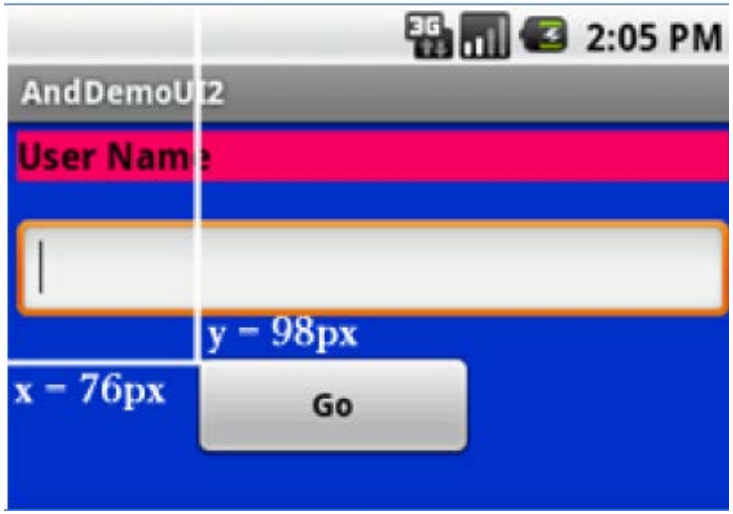




2. Các layout cơ bản

- **AbsoluteLayout**
  - Layout cho phép xác định vị trí chính xác của control dựa vào tọa độ x/y.
  - Absolute layout sẽ ít linh hoạt và bố trí cố định các control hơn so với các loại layout khác.

```
<Button
        android:layout_widt ="120dip"
        android:text="Go"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:id="@+id/btnGo"
        android:layout_x =“76px"
        android:layout_y =“98px"
/>
```



2. Các layout cơ bản

- **AbsoluteLayout(tiếp)**

```
<?xml version="1.0" encoding="utf-8"?>
< AbsoluteLayout
      android:id="@+id/myLinearLayout"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:background="#ff0033cc"
      android:padding="4dip"
      xmlns:android="http://schemas.android.
      com /apk/res/android“ >
<TextView
      android:id="@+id/tvUserName"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:background="#ffff0066"
      android:text="User Name"
      android:textSize="16sp"
      android:textStyle="bold"
      android:textColor="#ff000000"
      android:layout_x= "0dip"
      android:layout_y= "10dip" >
    </TextView>
<EditText
      android:id="@+id/etName"
      android:layout_width="fill_parent"
      android:layout_height="wrap_conte
      nt" android:textSize="18sp“
      android:layout_x ="0dip"
      android:layout_y ="38dip" >
      </EditText>
<Button
      android:layout_width ="120dip”
      android:text="Go"
      android:layout_height="wrap_conte
      nt" android:textStyle="bold"
      android:id="@+id/btnGo“
      android:layout_x="100dip"
      android:layout_y= "170dip"/>
 </AbsoluteLayout>
```

2. Các layout cơ bản

- **ScrollView Layout**
  - Khi cần hiển thị nhiều dữ liệu lên màn hình, ta sử dụng ScrollView.
  - Nhờ có thanh trượt và thanh cuộn, người dùng có thể nhìn thấy phần còn lại của dữ liệu trên màn hình bằng cách trượt sang hoặc cuộn xuống.
  - Điều này tương tự như xem một trang web chứa nội dung lớn, muốn xem nội dung bên dưới ta phải sử dụng thanh cuốn để xuống dưới.

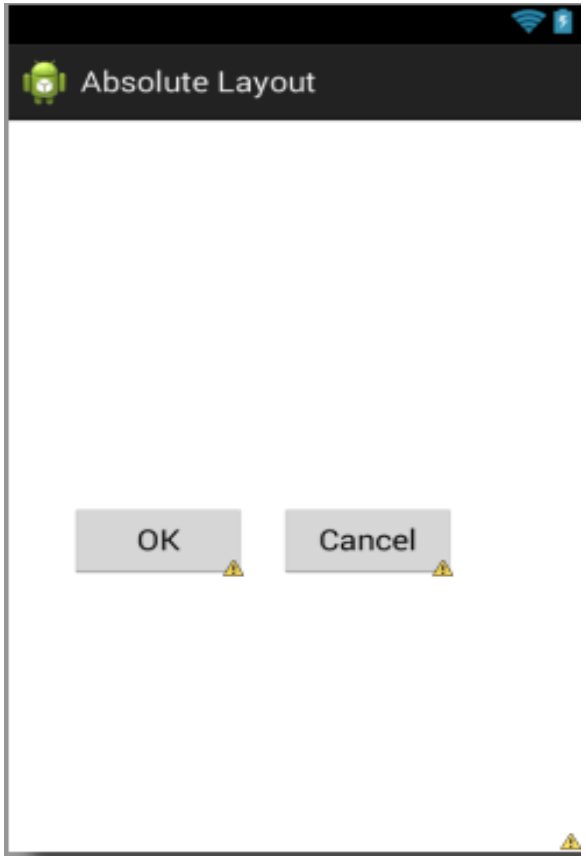
2. Các layout cơ bản

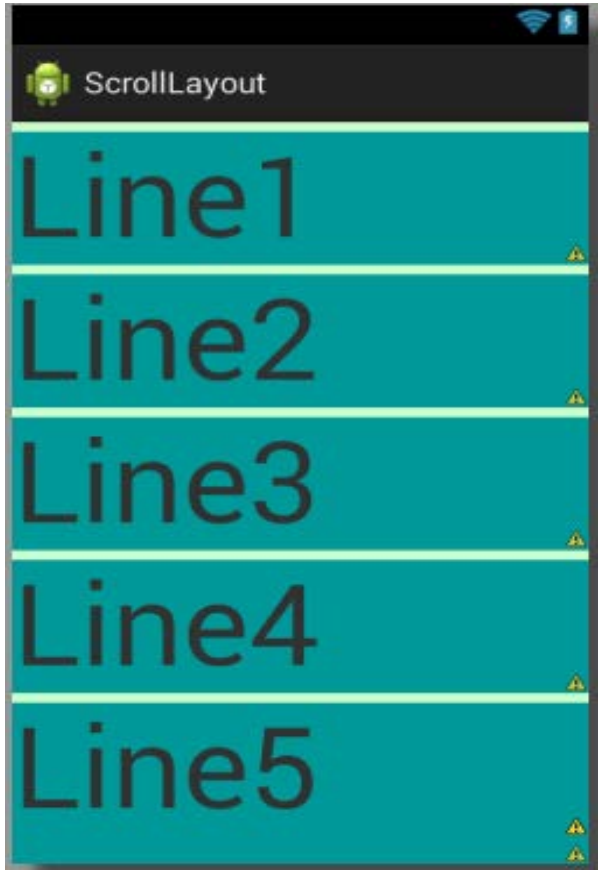
- **ScrollView Layout**

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView android:id="@+id/myScrollView1"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="#ff009999"
xmlns:android="http://schemas.android.com/apk/res/android">
<LinearLayout
android:id="@+id/myLinearLayoutVertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical">
<View
android:layout_width="fill_parent"
android:layout_height="6dip"
android:background="#ffccffcc" />
<TextView
android:id="@+id/textView2"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Line2"
android:textSize="70dip"/>
<View
android:layout_width="fill_parent"
android:layout_height="6dip"
android:background="#ffccffcc" />
<TextView
android:id="@+id/textView3"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Line3" android:textSize="70dip" />
 <View
android:layout_width="fill_parent"
android:layout_height="6dip"
android:background="#ffccffcc" />
<TextView
android:id="@+id/textView4"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Line4" android:textSize="70dip" />
<View
android:layout_width="fill_parent"
android:layout_height="6dip"
android:background="#ffccffcc" />
<TextView android:id="@+id/textView5"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Line5" android:textSize="70dip" />
</LinearLayout> </ScrollView>
```

2. Các layout cơ bản

- **Demo → AbsoluteLayout và ScrollViewLayout**

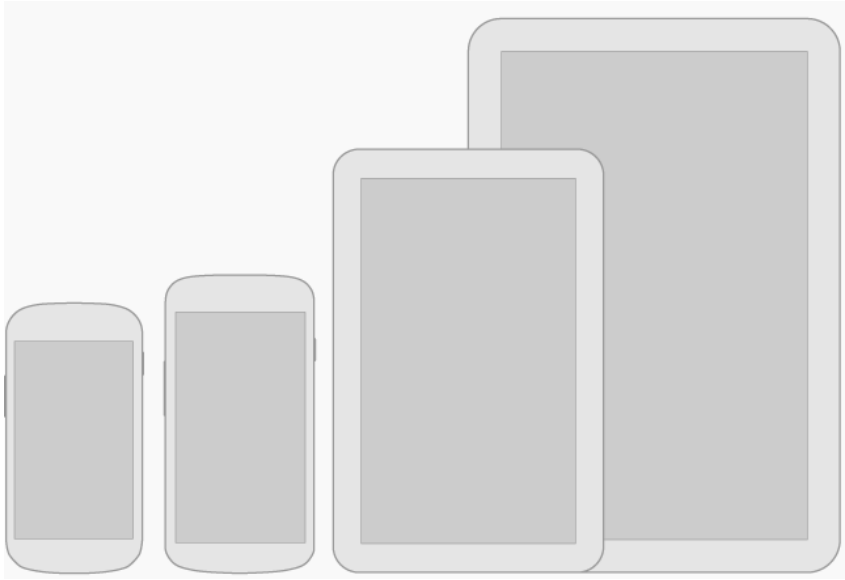




3. Style

Devices và Displays

- Android là nền tảng hệ điều hành được sử dụng trên rất nhiều thiết bị lớn nhỏ khác nhau như: tablet, phone.
- Khi thiết kế cần phải đảm bảo flexible (linh hoạt): dãn hoặc nén layout tùy theo độ rộng và cao của màn hình, vì vậy xu hướng thiết kế hiện nay là multi-pane layout (layout đáp ứng cho nhiều kiểu giao diện)



3. Style

Color

- Chọn màu sắc phù hợp với ứng dụng sẽ tăng hiệu quả sử dụng. Chú ý trong thiết kế vì người mù màu thường không phân biệt được màu đỏ và xanh lá cây.

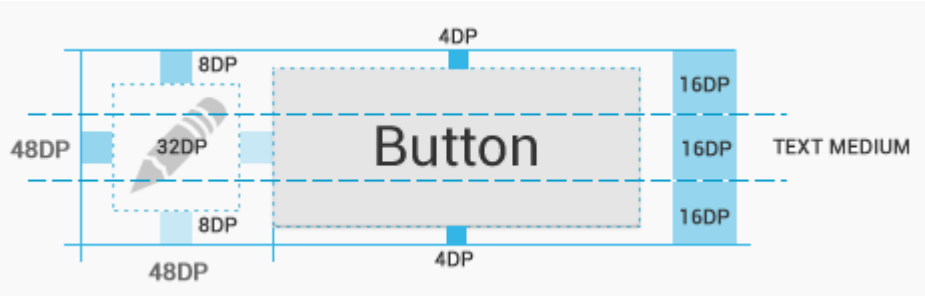
Palettle

- Mỗi một màu sẽ có một dải màu từ tối đến sáng cho người thiết kế lựa chọn

3. Style

Metrics và Grids

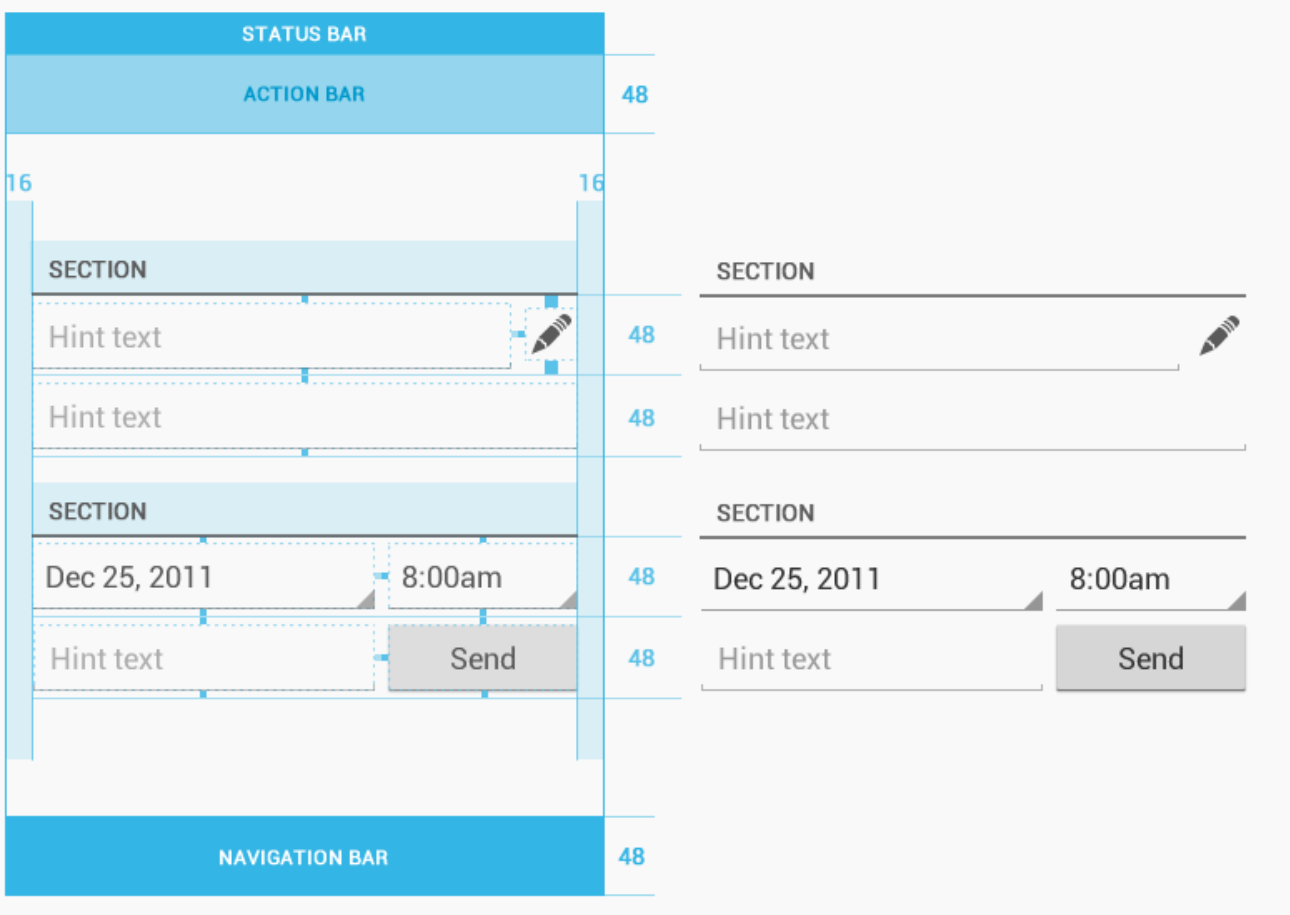
- Các thiết bị khác nhau không chỉ khác nhau về kích thước vật lý mà còn về mật độ màn hình(DPI). Thông thường khi thiết kế, một ứng dụng phải đáp ứng cho nhiều kiểu màn hình khác nhau. Vì vậy để linh hoạt cần hướng tới thiết kế các control với kích thước phù hợp cho nhiều màn hình.
- Sử dụng kích thước 48dp (khoảng 9mm): phù hợp với thao tác chạm vào màn hình của đối tượng.
- Khoảng cách giữa các control nên là 8dp.



3. Style

Metrics và Grids

- Một ví dụ về thiết kế bố trí màn hình:



3. Style & Theme

Thiết kế Style

- Android cung cấp theme Holo Light và Holo Dark để lựa chọn khi xây dựng ứng dụng
- Một style là một bộ các attribute/value được khai báo sẵn để apply vào look & feel (một các nói khác về GUI) của view
- Style có thể được kế thừa bằng cách thêm thuộc tính parent = "@android: style/…"
- Thêm thuộc tính style vào view để apply
- Để thực hiện style hoặc theme, tạo một tệp XML trong thư mục /res/values/ của mỗi một project. Ví dụ về một tệp xác định style như sau: style.xml

3. Style & Theme

```
<?xml version="1.0" encoding="utf-8"?>
    <resources>
      <style name="CodeFont" parent="@android:style/TextAppearance.Medium">
        <item name="android:layout_width">fill_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColor">#00FF00</item>
        <item name="android:typeface">monospace</item>
      </style>
    </resources>
```

Để gán thuộc tính style cho phần tử ví dụ với các dòng text sẽ thực

hiện khai báo: *style="@style/CodeFont"*

```
<TextView
        style="@style/CodeFont"
        android:text="@string/hello" />
```

Tương đương với đoạn code sau

```
<TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textColor="#00FF00"
        android:typeface="monospace"
        android:text="@string/hello" />
```

3. Style & Theme

Theme

- Theme là một style mà nó apply cho toàn bộ activity hoặc thậm chí toàn application
- Thêm thuộc tính andoid:theme vào activity trong manifest
- Khi một style được apply thành theme thì toàn bộ các thuộc tính được khai báo trong style sẽ ghi đè giá trị mặc định của các view trong Activity

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <style name="MyTheme" parent="android:Theme.Light">
        <item name="android:windowNoTitle">true</item>
        <item name="android:windowBackground">@color/translucent_red</item>
        <item name="android:listViewStyle">@style/MyListView</item>
    </style>

    <style name="MyListView" parent="@android:style/Widget.ListView">
        <item name="android:listSelector">@drawable/ic_menu_home</item>
    </style>

</resources>
```

3. Style & Theme

Theme

- Bạn có thể chỉ ra các thuộc tính của theme qua câu lệnh :?android:attr . Câu lệnh này có nghĩa là đang tham chiếu tới thuộc tính style của theme hiện tại.
- Ví dụ ?android:attr/listPreferredItemHeight nghĩa là: sử dụng giá trị được xác định bởi thuộc tính được gọi là listPreferredItemHeight trong theme hiện tại.

```
<Button
        android:id="@+id/Button01"
        style="?android:attr/buttonBarButtonStyle"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Show" />
```

3. Style & Theme

Fonts

- Để thêm font vào ứng dụng trong android, thực hiện các bước sau:
  - Tạo thư mục /fonts/ trong thư mục /assets.
  - Copy fonts cần dùng vào /fonts/
  - Sử dụng code Java để sử dụng font cho UI widget cần hiển thị.
- Sử dụng font copy trong xml:

```
<TextView android:id="@+id/textView4"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Line4"
android:textSize="70dip"
android: typeface=""serif"/>
```

3. Style & Theme

Fonts

- Sử dụng code Java:

```
TextView tvCustom= (TextView) findViewById (R.id.custom);
Typeface myNewFace= Typeface.createFromAsset(
                    getAssets(), “fonts/Jokerman.TTF”);
tvCustom.setTypeface(myNewFace);
```

3. Style & Theme

- Demo → Style & Theme

Tổng kết buổi học

- Các Layout cơ bản (tiếp)
  - Table layout
  - Relative Layout
  - Absolute Layout
  - ScrollView Layout
- Style and Themes


Kết thúc!