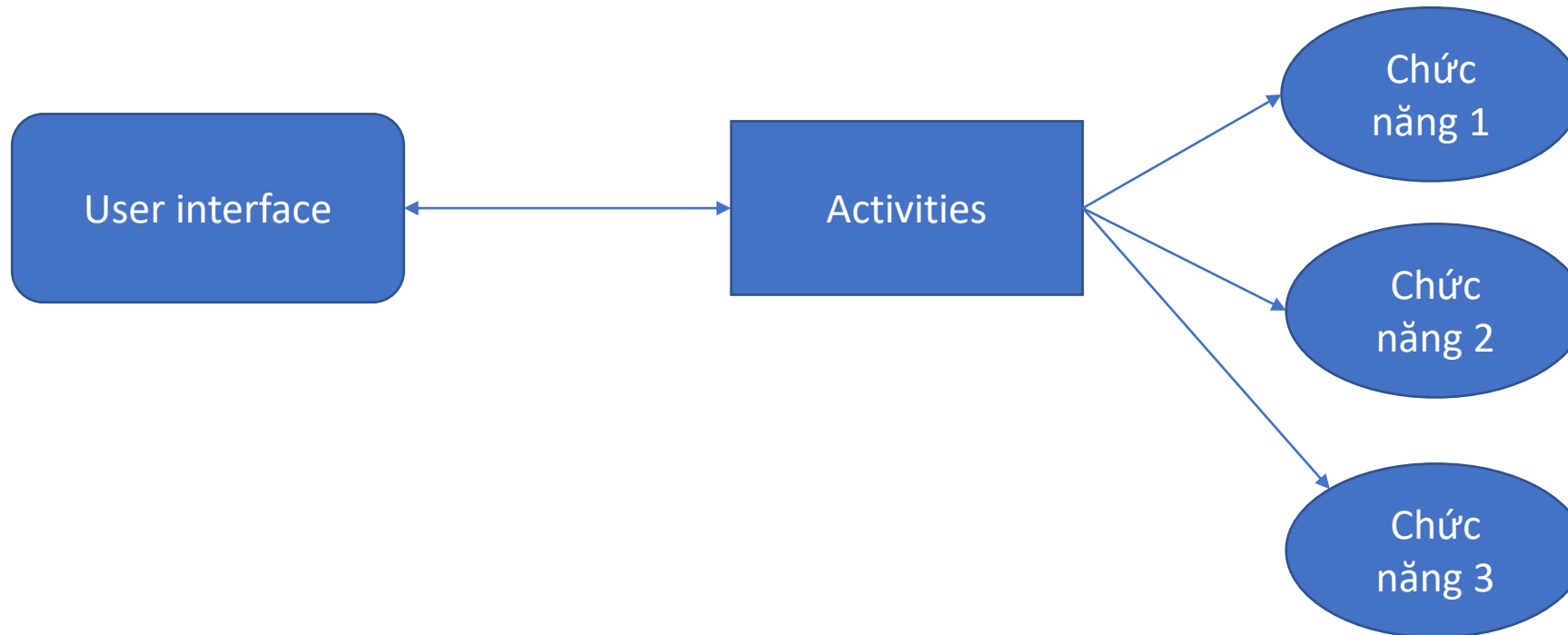


Lập trình thiết bị di động

Ts. Lâm Chí Nguyễn

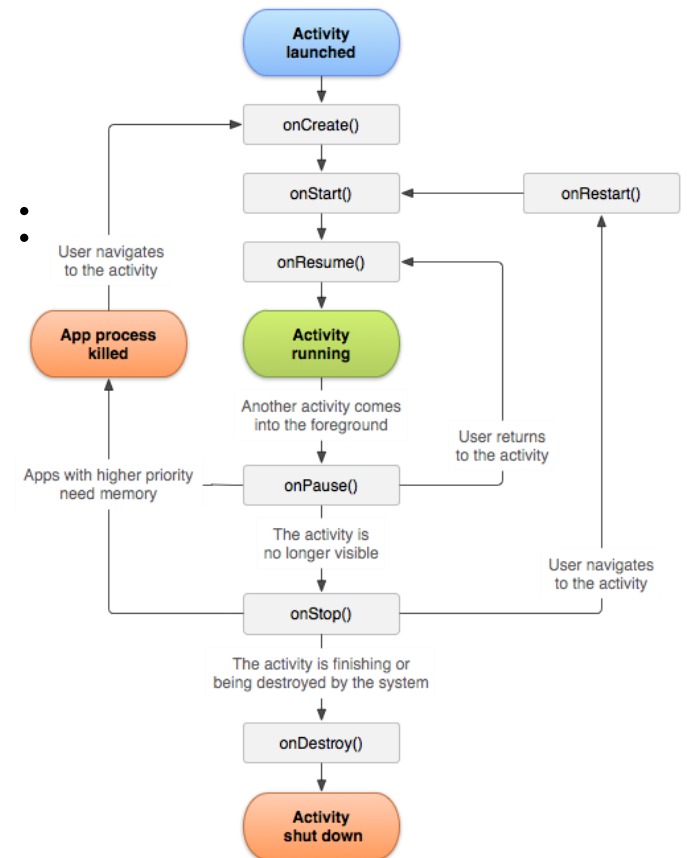
Chương 3 - Activities

- Activity là một thành phần cần thiết của một ứng dụng di động
- Activity là nơi cung cấp khả năng giao tiếp giữa giao diện người dùng và thực hiện chức năng của ứng dụng.



Chương 3 - Activities

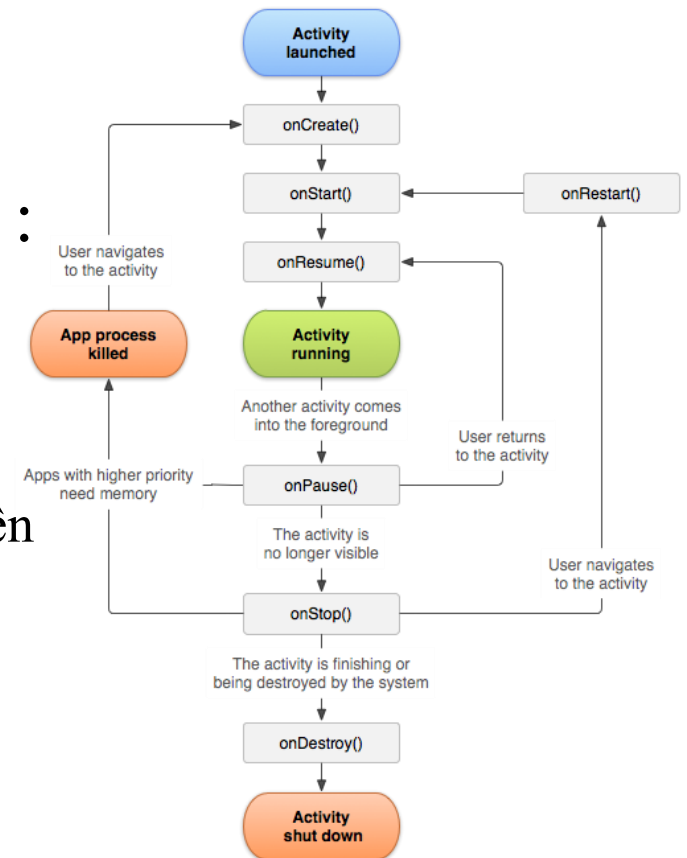
- Các bước cần thiết để có được một activity trong project :
 - Cài đặt một lớp thừa kế từ lớp Activity
 - Cấu hình trong AndroidManifest.xml
 - Thiết kế Layout , giao diện tương tác với người dung
 - Thiết kế các sự kiện event (các chuyển hướng)
 - Thiết kế các chức năng (nếu cần thiết)



Chương 3 - Activities

- Các bước cần thiết để có được một activity trong project :
 - Cài đặt một lớp thừa kế từ lớp Activity
 - Cài đặt các hàm thừa kế : onCreate, onStart().....
 - Cần gọi hàm trùng tên của lớp cha (Activity).
 - Thiết kế các đoạn chương trình thực hiện khi tương ứng với các sự kiện

```
public class WelcomeActivity extends Activity {  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
    }  
    protected void onStart() {  
        super.onStart();  
    }  
    protected void onRestart() {  
        super.onRestart();  
    }  
    protected void onResume() {  
        super.onResume();  
    }  
    protected void onPause() {  
        super.onPause();  
    }  
    protected void onStop() {  
        super.onStop();  
    }  
    protected void onDestroy() {  
        super.onDestroy();  
    }  
}
```



Chương 3 - Activities

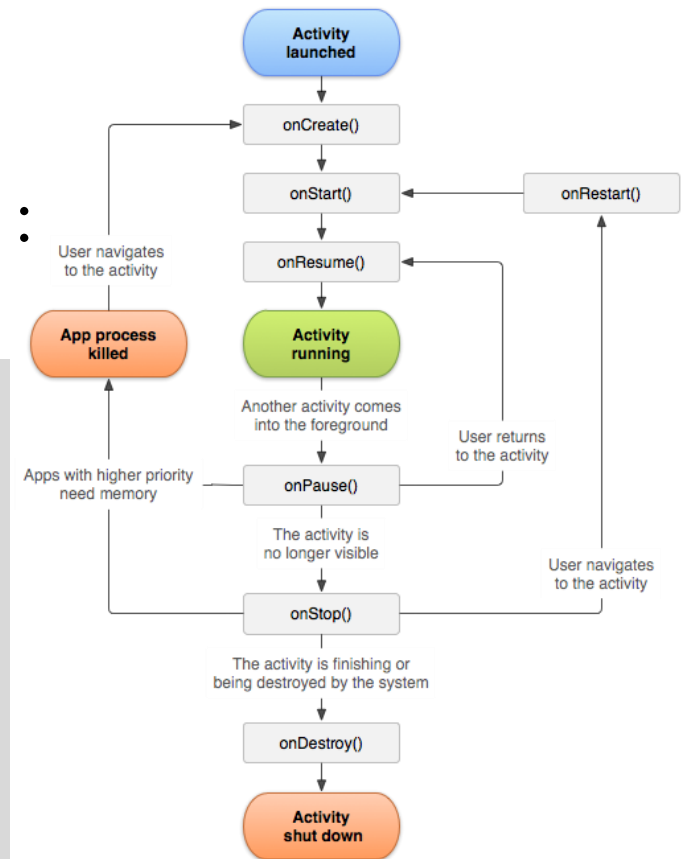
- Các bước cần thiết để có được một activity trong project :
 - Các activities cần được khai báo trong AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.mylogin">

    <application
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyLogin">

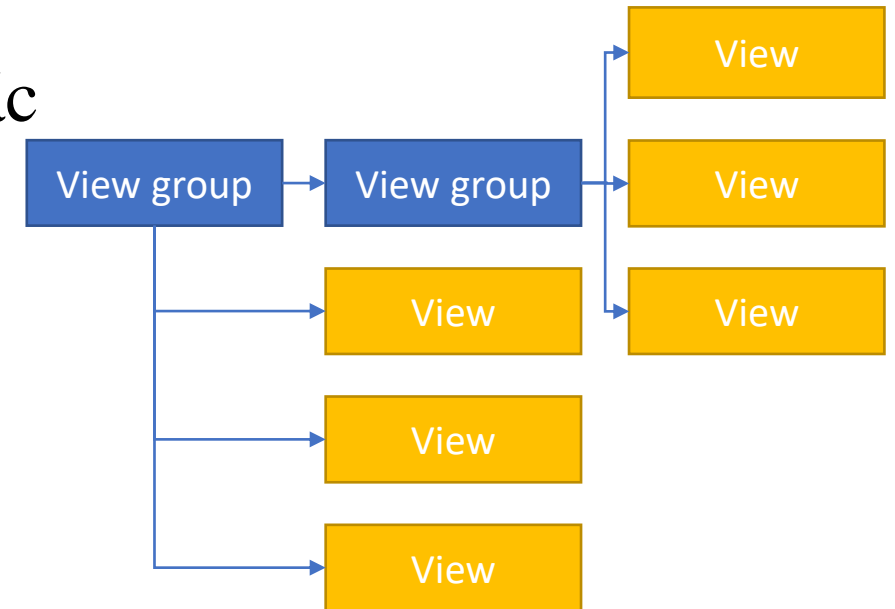
        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="YOUR_API_KEY" />

        <activity
            android:name=".WelcomeActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```



View và ViewGroup

- View là một phần tử cơ bản, cho phép tương tác với người dùng :
 - Xem thông tin TextView
 - Nhập dữ liệu EditText
 - Yêu cầu thực hiện chức năng Button
 - Hiển thị trạng thái ProgressBar
 - Hiển thị hình ảnh ImageView
- ViewGroup : tập hợp các view và viewgroup khác
 - Layout : LinearLayout, TableLayout, ...
 - ToolBar
 - ScrollView
 - TabHost
 - Table Row



Layout

- Layout được lưu trữ trong project dưới dạng XML.
- Công cụ thiết kế Layout gồm 03 cách :
 - Code : Thay đổi chỉnh sửa trên tập tin XML
 - Design : Thay đổi chỉnh sửa bằng giao diện đồ họa kéo thả
 - Split : kết hợp code và design
- Có 06 loại layout thường được sử dụng :
 - LinearLayout
 - TableLayout
 - GridLayout
 - Framelayout
 - RelativeLayout
 - ConstraintLayout

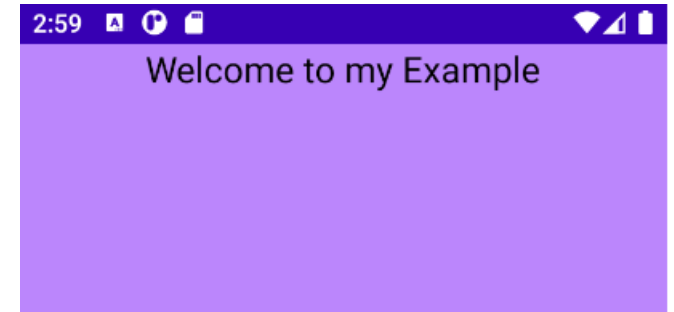
LinearLayout

- Các phần tử được liệt kê tuần tự theo hướng qua
 - Orientation = (horizontal, vertical)
- Xác định kích thước qua :
 - layout_width = (match_parent, wrap_content, 100sp)
 - layout_height = (match_parent, wrap_content, 100sp)
- Các thuộc tính còn lại tìm hiểu qua giao diện design

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Welcome to my Example"
        android:textSize="20sp"
        android:textColor="@color/black">
    </TextView>

</LinearLayout>
```



TableLayout

- Cho phép thiết kế giao diện trình bày dạng bảng hai chiều
- Sử dụng thẻ <TableRow> để thêm dòng
- Các phần tử trên một dòng được phối về các cột theo thứ tự từ trái sang phải
- Để một phần tử có thể trình bày trên nhiều ô cùng một dòng sử dụng thuộc tính *layout_span*
- Có thể kết hợp TableLayout và các layout cơ bản khác. Ngoại trừ các RelativeLayout, ConstraintLayout
-

TableLayout

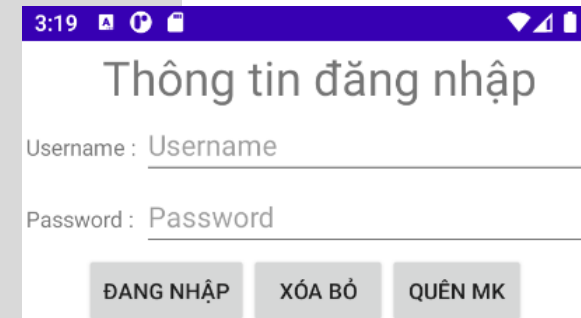
```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_height="wrap_content" android:layout_width="match_parent"
    android:layout_margin="5px">
    <TableRow >
        <TextView
            android:text="Thông tin đăng nhập"
            android:textSize="30dp"                android:layout_span="2"
            android:gravity="center"/>

    </TableRow>
    <TableRow>
        <TextView        android:text="Username : "        android:layout_width="wrap_content"/>
        <EditText        android:hint="Username"        android:layout_width="300dp"/>
    </TableRow>
    <TableRow>
        <TextView android:text="Password : "        android:layout_width="wrap_content"/>
        <EditText android:hint="Password"/>
    </TableRow>
    <LinearLayout android:gravity="center" android:orientation="horizontal">
        <Button        android:layout_width="wrap_content"        android:layout_height="wrap_content"
            android:text="Đăng nhập"/>

        <Button        android:layout_width="wrap_content"        android:layout_height="wrap_content"
            android:text="Xóa bỏ"

        />

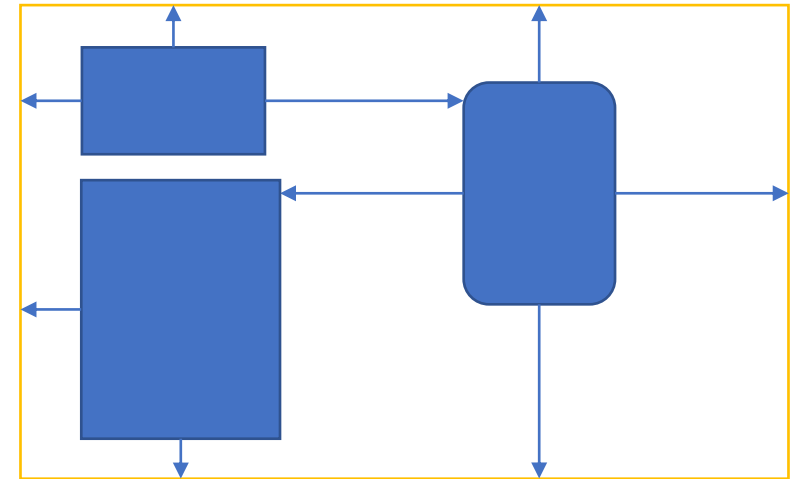
        <Button        android:layout_width="wrap_content"        android:layout_height="wrap_content"
            android:text="Quên MK"/>
    </LinearLayout>
</TableLayout>
```



The screenshot shows a mobile application interface with a dark blue header bar at the top displaying the time 3:19 and various status icons. The main title of the screen is 'Thông tin đăng nhập' (Login Information). Below the title, there are two input fields: 'Username : Username' and 'Password : Password'. At the bottom of the form, there are three buttons: 'ĐANG NHẬP' (Login), 'XÓA BỎ' (Cancel), and 'QUÊN MK' (Forgot Password).

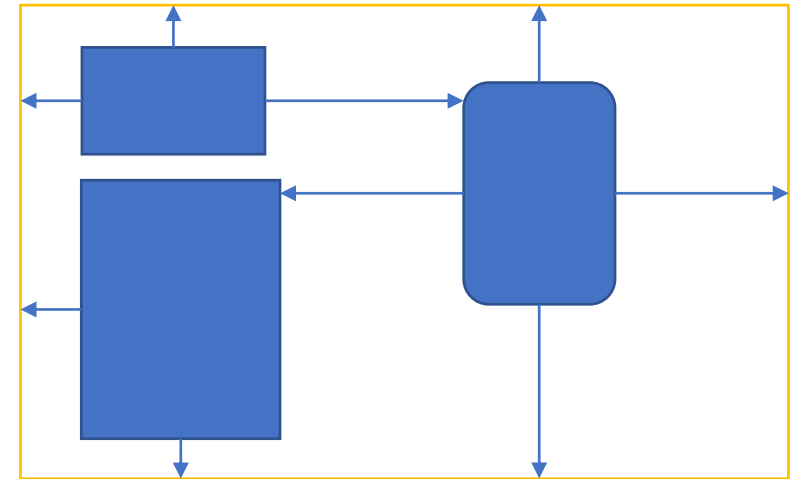
ConstraintLayout

- Cho phép thiết kế giao diện thích ứng với thay đổi kích thước, thiết bị,
- Quá trình thiết kế layout cần xác định các ràng buộc :
- Các ràng buộc :
 - Vị trí : Top, Bottom, Start, End
 - `app:layout_constraintTop_toTopOf`
 - `app:layout_constraintEnd_toStartOf`
 - Đường cơ sở - Baseline
 - `layout_constraintBaseline_toBaselineOf`
 - Canh lề - Alignment : Thay đổi vị trí tương đối
 - Guideline
 - Barrier
- <https://developer.android.com/codelabs/constraint-layout#0>



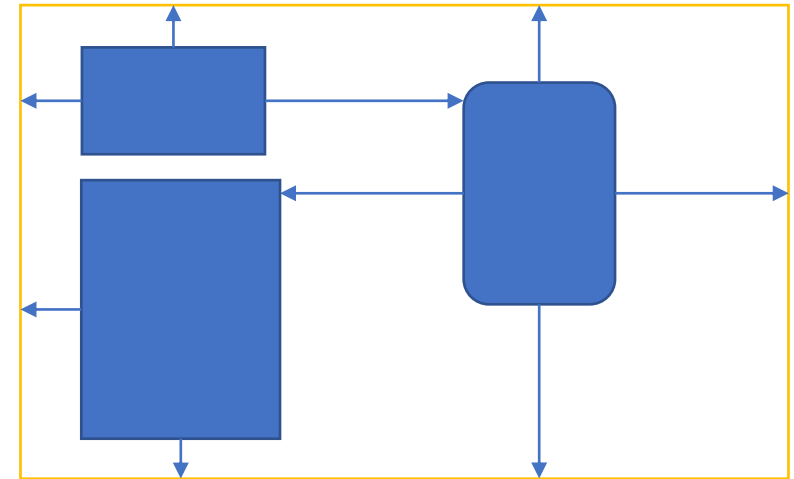
ConstraintLayout

- Cho phép thiết kế giao diện thích ứng với thay đổi kích thước, thiết bị,
- Quá trình thiết kế layout cần xác định các ràng buộc :
- Các ràng buộc :
 - Guideline
 - `<androidx.constraintlayout.widget.Guideline>`
 - `android:orientation = "vertical", "Horizontal"`
 - `app:layout_constraintGuide_begin = "70dp"`
 - Barrier
 - `<androidx.constraintlayout.widget.Barrier>`
 - `app:constraint_referenced_ids = "id view"`
 - `app:barrierDirection="top", "bottom", "left", "right"`
- <https://developer.android.com/codelabs/constraint-layout#0>



ConstraintLayout

- Ràng buộc theo chuỗi chain : Horizontal chain, Vertical_chain
- Kết nối các phần tử thành một chuỗi các mắt xích.
- Các phân phối tương đối giữa các phần tử :
 - `app:layout_constraintHorizontal_chainStyle= ""`
 - `app:layout_constraintVertical_chainStyle= ""`
- ChainStyle:
 - Spread
 - Spread inside
 - Pack
 - Pack with bias



View - Components

- View là một phần tử cơ bản, cho phép tương tác với người dùng :
 - Xem thông tin TextView
 - Nhập dữ liệu EditText
 - Yêu cầu thực hiện chức năng Button
 - Hiển thị trạng thái ProgressBar
 - Hiển thị hình ảnh ImageView
 - ...
- Để thiết kế các View theo yêu cầu cá nhân :
 - Thay đổi thuộc tính tại layout.xml
 - Thay đổi thuộc tính tại design
 - Thay đổi thuộc tính thông qua câu lệnh
- Để có thể nhận được đối tượng phần mềm trong code java :
 - findViewById(R.id.name);

View - Components

- Thuộc tính bắt buộc :
 - `android:id="@+id/text1"`
 - `android:layout_width="match_parent"`
 - `android:layout_height="wrap_content"`
- Các thuộc tính quan trọng nhóm view text:
 - `android:text="Welcome to my Example"`
 - `android:textSize="20sp"`
 - `android:textColor="@color/black"`
 - `android:gravity="center"`
- Các thuộc tính quan trọng nhóm progressBar
 - `android:min="100"`
 - `android:max="200"`
 - `android:progress="150"`

Event

- Là sự tác động của người dung lên một phần tử : view, Viewgroup
- Lập trình sự kiện là phát triển các hàm chức năng sẽ thực hiện khi một sự kiện xảy ra.
 - Lập trình sự kiện qua hàm
 - Lập trình sự kiện inline
 - Lập trình sự kiện độc lập

Event

- Là sự tác động của người dung lên một phần tử : view, Viewgroup
- Lập trình sự kiện là phát triển các hàm chức năng sẽ thực hiện khi một sự kiện xảy ra.
 - Lập trình sự kiện qua hàm thành viên của activity
 - Lập trình sự kiện inline
 - Lập trình sự kiện độc lập

event

- Lập trình sự kiện qua hàm thành viên của activity

```
public void onClick1(View view){  
    EditText edit = findViewById(R.id.description);  
    edit.setText("click on button");  
}
```

- Thay đổi thuộc tính của view có sự kiện cần tương tác

```
<Button  
    android:id="@+id/button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Button"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="@+id/line3"  
    app:layout_constraintTop_toBottomOf="@+id/description"  
    android:onClick="onClick1"  
>
```

- Sự kiện được gắn với một hoặc nhiều đối tượng View

event

- Lập trình sự kiện inline

```
Button button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        EditText edit = findViewById(R.id.description);
        edit.setText("click on button");
    }
});
```

- Không thay đổi thuộc tính onclick
- Sự kiện được gắn kết khi chương trình thực thi
- Sự kiện chỉ dùng riêng cho một đối tượng cụ thể

event

- Lập trình sự kiện độc lập
 - Tạo lớp cài đặt giao diện EventListener (ví dụ OnClickListener)

```
public class MyOnClick implements View.OnClickListener {
    Activity context;
    public MyOnClick(Activity act){
        this.context = act;
    }
    @Override
    public void onClick(View view) {
        int id = view.getId();
        if (id==R.id.button) {
            //action for button
            EditText edit = context.findViewById(R.id.description);
            edit.setText("Click on Button");
        }
    }
}
```

event

- Lập trình sự kiện độc lập
 - Tạo đối tượng event trên activity cần thiết
 - Đặt lại event của View

```
Button button = findViewById(R.id.button);
```

```
MyOnClick onClick = new MyOnClick(this);
```

```
button.setOnClickListener(onClick);
```

- Sự kiện có thể được dùng cho nhiều đối tượng view
- Được sử dụng để lập trình các chức năng có liên quan hoặc không liên quan đến Activity

event

- Một số sự kiện cơ bản trên một View:
 - OnClickListener
 - OnLongClickListener
 - OnKeyListener
 - onTouchListener
 - OnCreateContextMenuListener
- Một số component có riêng mình các listener.
 - DatePickerDialog . OnDateSetListener
 - CalendarView . OnDateChangeListener