ProgressBar trong Android

Khi truyền dữ liệu qua activity khác thì sẽ dùng phương thức startActivity và truyền vào đó một intent , tuy nhiên nếu truyền dữ liệu qua activity khác mà muốn lắng nghe kết quả trả về từ activity đó thì chúng ta sẽ dùng phương thức **startActivityForResult()** và truyền vào đó 2 tham số là intent và requestCode. Phương thức này start một Activity và nhận kết quả trả về từ Activity đó thông qua phương thức đó là **onActivityResult()**.

À tất nhiên là phải **@Override** lại hàm **onActivityResult()** này ở Activity mà mình muốn lắng nghe kết quả trả về từ một Activity khác. Nếu không ghi đè phương thức này thì cũng chả làm sao cả, đơn giản là chúng ta sẽ không lấy được kết quả trả về thôi.

***Một ví dụ điển hình :***

Khi bạn mở một ứng dụng yêu cầu đăng nhập tài khoản, thông thường ở màn hình **Đăng nhập** thường sẽ nó thêm nút **Đăng ký**, khi bạn click vào nút đăng ký thì sẽ được chuyển đến màn hình **Đăng ký**, nếu bạn đăng ký thành công thì sẽ được chuyển về lại màn hình **Đăng nhập** và ở đây tự động ứng dụng sẽ điền sẵn tài khoản và mật khẩu mà bạn vừa đăng kí vào form đăng nhập luôn mà không cần phải nhập lại tài khoản, mật khẩu nữa.

Như vậy bạn thấy là bạn đã truyền username và mật khẩu từ màn hình **Đăng nhập** -> **Đăng ký**, và ở màn hình **Đăng ký** sẽ thao tác một số thứ để trả về cho màn hình **Đăng nhập** kết quả gì đó.

Đó gọi là ***truyền dữ liệu và lắng nghe kết quả trả về***

1. **resultCode** và **requestCode** trong ActivityForResult có ý nghĩa gì?

**requestCode** : request - yêu cầu, code - mật hiệu , vậy requestCode là một giá trị int như là mật hiệu khi khi gửi dữ liệu từ một Activity sang một Activity khác , dựa vào thằng **requestCode** này này mà nó biết dữ liệu được yêu cầu từ thằng Activity nào

Ví dụ: ActA gửi dữ liệu đi cho ActB và ActC và cả ActB và ActC đều trả về dữ liệu cho ActA thì dựa vào thằng **requestCode** để biết nó được gửi từ đâu.

**resultCode**: result – kết quả, code – mật hiệu, khi một activity gửi dữ liệu về thì gọi hàm **setResult()** và truyền vào resultCode, và đây mà cách mà activity nhận sẽ lấy dữ liệu cho đúng trong trường hợp gửi về nhiều result và nó cũng cho biết được Intent gửi về tương ứng là gì.

Ví dụ: ActA gửi đi cho ActB số A và số B , ActB nhận dữ liệu từ ActA và thực hiện 2 phép toán là phép trừ và phép cộng, sau đó Actb trả về cho ActA với **resultCode** phép cộng là 1 và phép trừ là 2 , như vậy ActA dựa vào **resultCode** này để biết được kết quả trả về là cộng hay là trừ.

**Cần chú ý :**

Dữ liệu chỉ có thể được trả về khi bạn gọi hàm **finish().** Cần phải gọi hàm **setResult()** trước khi gọi **finish(),** nếu không kết quả sẽ không được trả về.

2. **RESULT\_OK** và **RESULT\_CANCELED** ?

Đặt resultCode là RESULT\_OK để thể hiện là có chứa kết quả trả về.

Đặt resultCode là RESULT\_CANCELED để thể hiện là không trả về data.

3. Các bước sử dụng **ActivityForResult** ?

Ví dụ Activity1 – LoginAct start Activity2 – RegisterAct và lắng nghe kết quả trả về từ Activity này.

Ở LoginAct gọi phương thức **startActivityForResult()** để đi đến RegisterAct và lắng nghe kết quả trả về.

Ở RegisterAct gọi phương thức **setResult()** để đặt kết quả trả về là **RESULT\_OK** (có trả về kết quả) hay là **RESULT\_CANCELED** (không trả về kết quả) và gọi hàm **finish()** để hủy Activity này và trở về LoginAct

Ở LoginAct phải ghi đè phương thức **onActivityResult()** để nhận kết quả trả về từ RegisterAct.

ProgressBar trong Android

**ProgressBar**là một thanh tiến trình thể hiện trạng thái của một hoạt động, tác vụ đang diễn ra đến đâu, bao lâu thì kết thúc. ProgressBar thường có dạng thanh bar nằm ngang, và có thể được custom theo nhiều dạng khác nhau.

**Thuộc tính thường dùng của ProgressBar**

ProgressBar là lớp con của **View**, Do đó nó kế thừa tất cả các thuộc tính của **View**

**1. android:max:**Thuộc tính dùng để định nghĩa giá trị tối đa mà tiến trình có thể đạt được. Giá trị là số nguyên: 100, 200. Ví dụ thiết lập cho ProgressBar có giá trị tối đa là 100

<ProgressBar

android:id="@+id/simpleProgressBar"

style="?android:attr/progressBarStyleHorizontal"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:max="100" /><!-- Thiết lập giá trị tối đa cho ProgressBar -->

**2. android:progress:**Thuộc tính này dùng để định nghĩa giá trị mặc định cho tiến trình. Giá trị này trong khoảng từ 0 đến max và phải là số nguyên. Ví dụ sau thiết lập**max=100** và**progress=50**

<ProgressBar

android:id="@+id/simpleProgressBar"

style="?android:attr/progressBarStyleHorizontal"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:max="100"

android:progress="50"/><!--// 50 default progress value-->

**3. android:progressDrawable:**Thuộc tính này thường được sử dụng để tùy chính lại **ProgressBar**. Trong ví dụ sau chúng ta sẽ tùy chỉnh lại một **ProgressBar**:

**Bước 1:**Thêm code vào main\_activity.xml hoặc main.xml

<ProgressBar

android:id="@+id/simpleProgressBar"

style="?android:attr/progressBarStyleHorizontal"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:max="100"

android:progress="50"

android:progressDrawable="@drawable/custom\_progress"/><!--custom progress drawable for progress mode-->

**Bước 2 :** Tạo tập tin custom\_progress.xml trong thư mục drawable:

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

<layer-list xmlns:android="http://schemas.android.com/apk/res/android" >

<item>

<shape>

<gradient

android:endColor="#fff"

android:startColor="#1f1"

android:useLevel="true" />

</shape>

</item>

</layer-list>

**4. android:indeterminate:**Nếu bạn biết giá trị công việc của bạn phải làm thì bạn chỉ cần sử dụng ProgressBar để thể hiện giá trị hiện tại. Còn nếu không biết giá trị chính xác thì bạn có thể sử dụng **android:indeterminate=”true”** trong XML hoặc progressBar.setIndeterminate(true) trong code. Lúc này **ProgressBar**sẽ ở chế độ Indeterminate và nó sẽ không hiển thị quá trình làm việc.