|  |  |
| --- | --- |
|  | |
| BÀI TẬP | |
|  | |
|  | Lập trình viên trên thiết bị di động  Lập trình di động trên  Android  Module 02:  Kiến trúc và xây dựng ứng dụng Android |

1. TỔNG QUAN VỀ LẬP TRÌNH ANDROID

VÀ MÔI TRƯỜNG PHÁT TRIỂN

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Cài đặt và sử dụng Android, cập nhật các phiên bản API.  Tạo được một Android Project.  Tạo và sử dụng Android Virtual Devices (Emulator). |

* 1. Thiết lập môi trường lập trình Android

Đề bài:

Hãy thiết lập môi trường lập trình Android, cài đặt phần mềm giả lập hệ điều hành Android (Genymotion), xây dựng và chạy ứng dụng đầu tiên để kiểm tra môi trường lập trình.

Mục tiêu:

* Biết và cài đặt được môi trường để lập trình Android.
* Biết cập nhật các phiên bản API.
* Tạo được một Android Project.
* Tạo và sử dụng được Android Virtual Devices hoặc hệ điều hành Android giả lập (Genymotion).

Gợi ý thực hiện:

* Tải và cài đặt Java JDK tại địa chỉ *http://java.sun.com/javase/downloads/index.jsp*.
* Tải và cài đặt Android Studio tại địa chỉ *http://developer.android.com/sdk/index.html.*
* Cập nhật các phiên bản API.
* Cài đặt phần mềm giả lập hệ điều hành Android: Genymotion.
* Tạo một Project đầu tiên để kiểm tra môi trường lập trình.
* Tạo một hệ điều hành giả lập bằng Android Virtual Devices (Emulator) hoặc Genymotion để chạy ứng dụng đầu tiên.

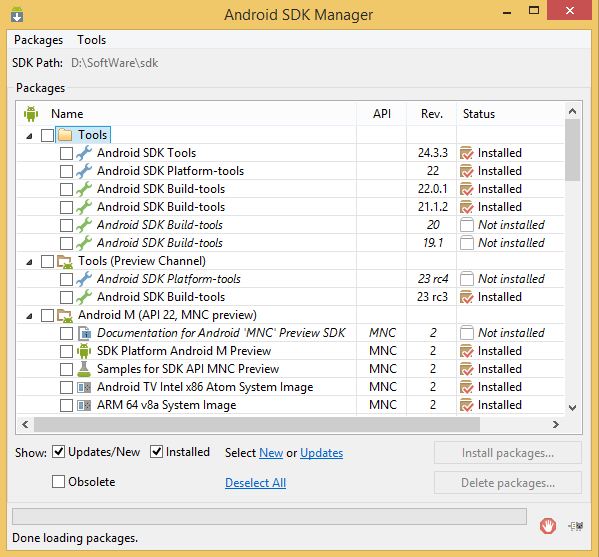
Hướng dẫn chi tiết:

1. Cài đặt Java JDK:

* Tải và cài đặt Java JDK tại địa chỉ: http://java.sun.com/javase/downloads/index.jsp (cài đặt Java JDK trước nhất và nên chọn bản mới nhất).
* Tải và cài đặt Android Studio tại địa chỉ: http://developer.android.com/sdk/index.html.

1. Cập nhật các phiên bản API.

* Tùy vào danh sách API được tải về máy nó sẽ hiện ra hết ở đây. Nếu máy tính của bạn có kết nối Internet, bạn có thể cập nhật API theo cách sau:
  + Bấm vào biểu tượng  trên thanh toolbar, màn hình Android SDK Manager sẽ hiện thị ra như bên dưới:



*Hình 1.1 . Cập các nhật phiên bản API.*

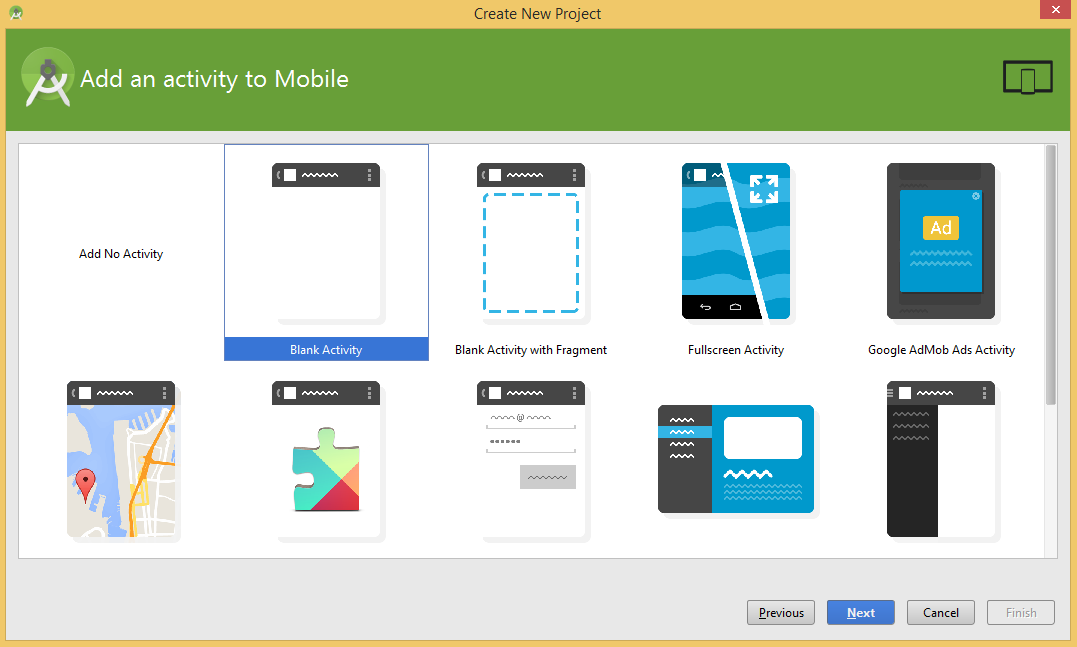
* + Ở hình trên, những gói nào chưa được cài đặt thì có dòng chữ Not installed, bạn chọn vào những gói cần cập nhật và nhấn nút Install packages ở góc phải dưới cùng để cập nhật chúng.

1. Cấu hình và chạy hệ điều hành Android giả lập

* Ở phần này, chúng sẽ tìm hiểu cách tạo và sử dụng hệ điều hành Android giả lập - Emulator từ Android Virtual Devices Manager (AVD Manager).
* Android Virtual Devices (AVD), cho phép lập trình viên tạo và quản lí các thiết bị ảo. Để tạo thiết bị ảo chúng ta khởi động AVD (nhấn vào biểu tượng ), ta chọn vào , một hộp thoại mới hiện ra và chọn máy ảo → chọn phiên bản API → đặt tên (ngoài ra còn có thể thiết lập lại cấu hình máy).
* Ngoài ra, chúng ta còn có thể tạo hệ điều hành Android giả lập bằng phần mềm Genymotion. Chi tiết cài đặt và sử dụng, bạn có thể tham khảo trên trang: http://www.genymotion.com.

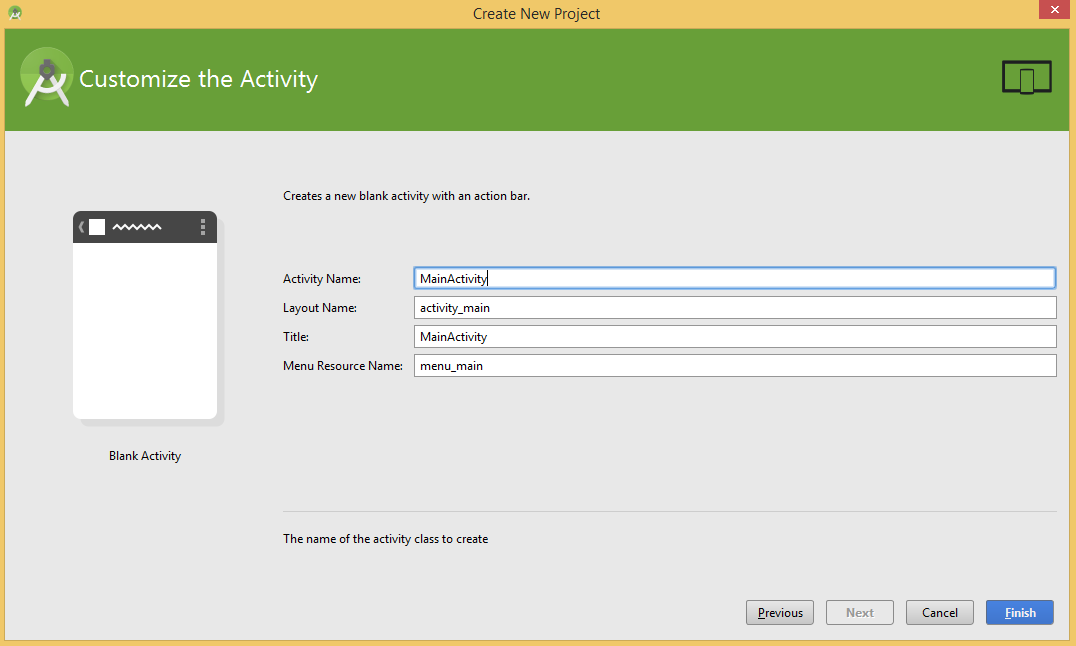
1. Tạo ứng dụng Android đầu tiên

* Chạy chương trình Android Studio.
* Chọn menu File → New → Android Application Project. Xuất hộp thoại với các thông tin sau:
  + Company Domain: là tên của đề án.
  + Application Name: là tên ứng dụng mà chúng ta sẽ tạo.
  + Package Name: là tên định danh của gói phần mềm được hệ thống lưu trữ trên thiết bị khi cài đặt ứng dụng. Thường sẽ được đặt theo tên của một tổ chức hoặc cá nhân tạo ra ứng dụng. Bắt buộc phải có 2 trường dữ liệu trở lên và ngăn nhau bởi dấu chấm. Ở đây, tôi đặt là “vn.cscom.xinchao”.
  + Nhấn Next để qua hộp thoại kế tiếp.
  + Minimum SDK: phiên bản thấp nhất được chỉ định để chạy ứng dụng. Một số các phương thức API sẽ được gọi bổ sung trong phần thư viện hỗ trợ.
  + Nhấn Next để qua hộp thoại kế tiếp.



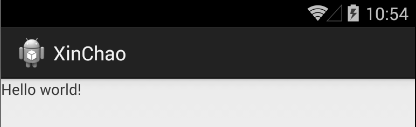
*Hình 1.2. Chọn loại giao diện.*

* Hộp thoại tiếp theo, gồm:
  + Activity Name: tên lớp chứa mã nguồn để xử lý giao diện.
  + Layout Name: tên tập tin giao diện.
  + Title: tên chủ để cho màn hình.
  + Menu Resource Name: tên tập tin thiết kế menu.



*Hình 1.3. Đặt tên cho các tập tin chính cần xử lý.*

* Để chạy ứng dụng này, chúng ta nhấn vào biểu tượng  . Và đây sẽ màn hình ứng dụng đầu tiên của bạn:



*Hình 1.4. Ứng dụng “Hello World” đầu tiên.*

1. CÁC THÀNH PHẦN ỨNG DỤNG ANDROID

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Hiểu được vòng đời của một Activity.  Hiểu và biết cách xây dựng, hiểu được vai trò và ý nghĩa của các phương thức quản lý trạng thái cho Activity.  Phân biệt được giữa Foreground Lifetime và Visible Lifetime. |

* 1. Sử dụng các phương thức đặt các trạng thái cho Activity

Đề bài:

* Viết chương trình sử dụng các phương thức đặt các trạng thái cho Activity.

Mục tiêu:

* Hiểu và biết cách sử dụng các phương thức đặt các trạng thái cho Activity.
* Biết cách viết mã lệnh ở vị trí phù hợp để phục vụ cho việc xây dựng các ứng dụng.

Gợi ý thực hiện:

* Tạo mới Android Application Project, đặt tên là TrangThaiActivity.
* Mở tập tin MainActivity.java tiến hành thêm các phương thức đặt các trạng thái cho Activity.
* Ghi các trạng thái vào **Log** để theo dõi.

Hướng dẫn chi tiết:

* Tạo mới Android Project, đặt tên là TrangThaiActivity (phần Create Activity nên chọn là EmptyActivity).
* Mở tập tin MainActivity.java, chúng ta tiến hành thêm các phương thức đặt các trạng thái và ghi vào Log để theo dõi như sau:

**public** **class** MainActivity **extends** Activity {

@Override

**protected** **void** onRestart() {

**super**.onRestart();

Log.*d*("MainActivity", "onRestart - Màn hình được khởi tạo lại.");

}

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.Layout.*activity\_main*);

Log.*d*("MainActivity",

"onCreate - Màn hình đang khởi tạo giao diện.");

}

@Override

**protected** **void** onStart() {

**super**.onStart();

Log.*d*("MainActivity", "onStart - Màn hình được khởi chạy.");

}

@Override

**protected** **void** onResume() {

**super**.onResume();

Log.*d*("MainActivity",

"onResume - Màn hình đang trong trạng thái

có thể nhận tương tác.");

}

@Override

**protected** **void** onPause() {

**super**.onPause();

Log.*d*("MainActivity",

"onPause - Màn hình dừng trạng thái có thể

nhận tương tác.");

}

@Override

**protected** **void** onStop() {

**super**.onStop();

Log.*d*("MainActivity",

"onStop - Màn hình đang hoạt động ở chế độ ngầm.");

}

@Override

**protected** **void** onDestroy() {

**super**.onDestroy();

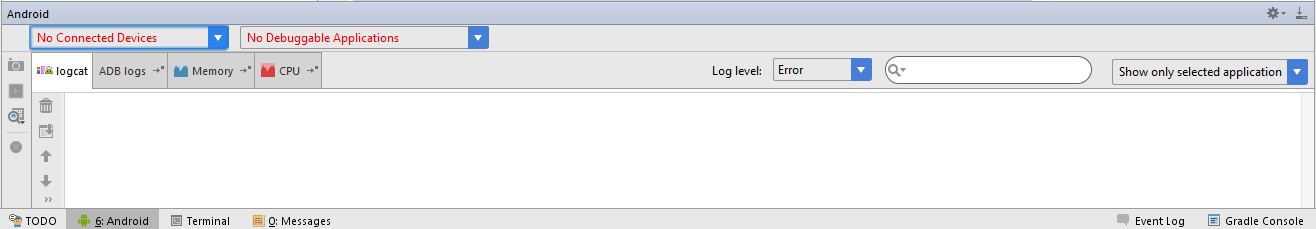
Log.*d*("MainActivity", "onDestroy - Màn hình bị đóng hoàn toàn.");

}

}

*Chú ý:*

* Chúng ta có thể tự viết (ghi đè) các phương thức đặt các trạng thái cho Activity hoặc phát sinh tự động bằng cách nhấp phải chuột vào lớp MainActivity.java → Generate → Override Methods… → chọn các phương thức đặt trạng thái.
* Thực hiện chạy ứng dụng, tiến hành quan sát trạng thái ứng dụng từ công cụ Logcat. Nếu không thấy có thể vào thanh công cụ ADT, nhấn Alt + 6.



*Hình 2.1. Trạng thái ứng dụng hiển thị trong LogCat.*

* 1. Phân biệt Foreground Lifetime và Visible Lifetime

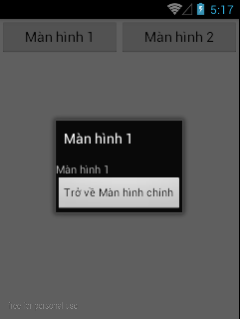
Đề bài:

* Tạo 3 Activity đặt tên là: ManHinhChinh, ManHinh1, ManHinh2. Yêu cầu:
  + Giao diện màn hình chính như sau:



*Hình 2.2. Màn hình chính.*

* + Khi ManHinh1 được kích hoạt thì nó sẽ nằm phía trên ManHinhChinh - vẫn thấy được màn hình chính (Hình 2 – 2).



*Hình 2.3.* *Màn hình chính.*

* + Khi ManHinh2 hiển thị thì nó sẽ chiếm toàn bộ màn hình - không thể thấy được màn hình chính.
* Bắt các sự kiện khi Activity chuyển đổi trạng thái và thông báo lên màn hình Android.

Mục tiêu:

* Hiểu sâu thêm về các phương thức đặt trạng thái trong Activity.
* Phân biệt được giữa Foreground Lifetime và Visible Lifetime.

Gợi ý thực hiện:

* Tạo mới Android Application Project, đặt tên là VongDoiActivity.
* Tạo 2 activity đặt tên lần lượt như sau: ManHinh1 và ManHinh2.
* Thiết kế giao diện cho ManHinhChinh, ManHinh1, ManHinh2 như yêu cầu của đề bài (sử dụng Button).
* Viết đoạn mã lệnh xử lý trong lớp ManHinhChinh.java.

Hướng dẫn chi tiết:

1. Tạo mới Android Application Project đặt tên là VongDoiActivity.
2. Tạo thêm 2 Activity đặt tên là ManHinh1 và ManHinh2

* Nhấp chuột phải vào thư mục layout nằm trong thư mục res:



*Hình 2.4. Tạo Activity - ManHinh1 - 1.*

* Xuất hiện Hộp thoại New. Ta đặt tên cho tập tin giao diện vừa tạo, rồi nhấn finish.
* Tương tự ta tạo tiếp màn hình 2.
* Tạo 2 lớp ở cùng thư mục với lớp ManHinhChinh.java có tên là ManHinh1.java và ManHinh2.java:

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.Layout.*man\_hinh\_1*);

}

* Tiếp theo là phần khai báo trong Manifest.xml, trong cặp thẻ application khai báo thêm nội dung như sau:

<activity

android:name=*".ManHinh1"*

android:label=*"@string/title\_activity\_man\_hinh1"*>

</activity>

<activity

android:name=*".ManHinh2"*

android:label=*"@string/title\_activity\_man\_hinh2"*>

</activity>

1. Thiết kế giao diện ManHinhChinh, ManHinh1, ManHinh2 theo yêu cầu của đề bài.

* Vào thư mục res/ layout:
  + Chỉnh sửa tập tin man\_hinh\_chinh.xml với nội dung như sau:

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:id=*"@+id/LinearLayout1"*

android:Layout\_width=*"match\_parent"*

android:Layout\_height=*"match\_parent"*

android:orientation=*"horizontal"*

tools:context=*"${relativePackage}.${activityClass}"* >

<Button

android:id=*"@+id/button1"*

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

android:Layout\_weight=*"1"*

android:text=*"Màn hình 2"* />

<Button

android:id=*"@+id/button2"*

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

android:Layout\_weight=*"1"*

android:text=*"Màn hình 1"* />

</LinearLayout>

* + Chỉnh sửa tập tin man\_hinh\_1.xml với nội dung như sau:

<TextView

android:id=*"@+id/textView1"*

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

android:Layout\_alignParentTop=*"true"*

android:Layout\_centerHorizontal=*"true"*

android:text=*"Màn hình 1"* />

<Button

android:id=*"@+id/btntrove"*

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

android:Layout\_alignParentLeft=*"true"*

android:Layout\_alignParentRight=*"true"*

android:Layout\_below=*"@+id/textView1"*

android:text=*"Trở về Màn hình chính"* />

* Để màn hình 1 có thể hiển thị ở dạng Dialog, ta vào tập tin AndroidManifest.xml thêm dòng lệnh sau:

<activity

android:name=*".ManHinh1"*

android:label=*"@string/title\_activity\_man\_hinh1"*

**android:theme=*"@*android*:style/Theme.Dialog"* >**

</activity>

1. Viết các câu lệnh xử lý trong lớp ManHinhChinh.java với nội dung như sau:

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.Layout.*man\_hinh\_chinh*);

Button btn1 = (Button) findViewById(R.id.*button1*);

Button btn2 = (Button) findViewById(R.id.*button2*);

btn1.setOnClickListener(**new** View.OnClickListener() {

@Override

**public** **void** onClick(View v) {

Intent intent = **new** Intent(ManHinhChinh.**this**, ManHinh1.**class**);

startActivity(intent);

}

});

btn2.setOnClickListener(**new** View.OnClickListener() {

@Override

**public** **void** onClick(View v) {

Intent intent = **new** Intent(ManHinhChinh.**this**, ManHinh2.**class**);

startActivity(intent);

}

});

}

*Giải thích sơ lược về mã lệnh:*

* Phương thức findViewById để trả về View theo đúng Id truyền vào, mọi điều khiển trong Android đều kế thừa từ View, ở đây ta ép kiểu về cho đúng kiểu Button.
* Phương thức startActivity dùng để mở một Activity, chú ý là Activity này được đưa vào một Intent với cách khai báo như trên.

*Chú ý:*

* Chúng ta cũng có thể thiết lập các thuộc tính và thiết kế giao diện cho các màn hình bằng cách kéo điều khiển Button từ Palette → Form Widget. Và thiết lập các thuộc tính cho chúng từ phần Properties. Nhưng mục đích của bài tập này là giúp cho chúng ta hiểu về vòng đời Activity, phân biệt được Foreground Lifetime và Visible Lifetime nên tạm thời chúng ta không đi sâu vào các Layout, các điều khiển, các xử lý của các điều khiển cũng như Intent – Những phần này sẽ được tìm hiểu sâu hơn ở những bài sau.

1. Thực thi chương trình và thực hiện các thao tác sau:

* Ở màn hình chính, bấm nút ManHinh1 để mở màn hình 1 (Activity ManHinh1) và quan sát hiện tượng xảy ra (chú ý quan sát xem những phương thức nào được gọi).
* Từ màn hình 1, chọn nút “Trở về” và quan sát hiện tưởng xảy ra (chú ý quan sát xem những phương thức được gọi là những phương thức nào).
* Từ màn hình chính, bấm nút ManHinh2 để mở màn hình 2 (Activity ManHinh2) và quan sát hiện tượng xảy ra (xem những phương thức nào được gọi).
* Bấm nút “Back” của thiết bị (điện thoại) và quan sát hiện tượng (chú ý xem phương thức nào được gọi).

1. GIAO DIỆN NGƯỜI DÙNG VÀ XỬ LÝ SỰ KIỆN

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Hiểu và biết cách sử dụng các Layout cơ bản.  Hiểu và biết cách sử dụng các điều khiển cơ bản: TextView, EditText, Button, CheckBox, ImageView,....  Thiết lập được các sự kiện cho các điều khiển trong Android.  Biết và có thể lập trình sự kiện trong Android bằng nhiều cách.  Biết cách sử dụng, định dạng và truy xuất được các loại Layout cơ bản: Frame Layout, Linear Layout, Table Layout, Relative Layout, Absolute Layout.  Tạo được giao diện đúng yêu cầu, đẹp với Layout và các điều khiển cơ bản. |

* 1. Tài nguyên ứng dụng giao diện (Layout)

Đề bài:

* Sử dụng Frame Layout thiết kế giao diện như sau:



*Hình 3.1. Giao diện với Frame Layout.*

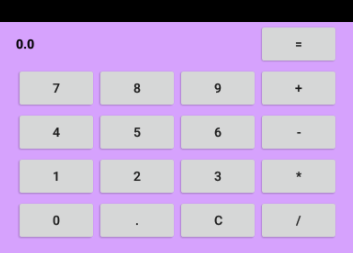
* + Ở hình trên, mục số 1 và số 2 là 2 hình ảnh có màu đỏ và màu vàng được đặt trong thư mục res/ drawable của Project.
* Sử dụng Linear Layout thiết kế giao diện như sau:

|  |  |
| --- | --- |
| *Hình 3.2. Giao diện với Linear Layout – Horizontal.* | *Hình 3.3. Giao diện với Linear Layout – Vertical.* |



*Hình 3.4. Giao diện với Linear Layout(Sử dụng Horizontal và Vertical).*

* Sử dụng Table Layout thiết kế giao diện như sau:



*Hình 3.6. Giao diện với Table Layout.*

* Sử dụng Relative Layout thiết kế giao diện như sau:



*Hình 3.7. Relative Layout.*

Mục tiêu:

* Biết cách sử dụng các loại tài nguyên giao diện phù hợp.
* Tạo được các giao diện theo yêu cầu.

Gợi ý thực hiện:

* Tạo 1 Android Application Project.
* Tạo Layout và kết nối Layout vào Activity.
* Đặt các điều khiển lên Layout vừa tạo và thiết lập, tùy chỉnh các thông số của các Layout để được giao diện như ý muốn.

Hướng dẫn chi tiết:

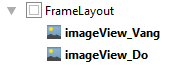
1. Tạo 1 Android Application Project
2. Tạo Layout và kết nối Layout vào Activity

* Tạo một Layout mới có tên “layout\_cua\_tui”.
* Khi chúng ta tạo một Android Application Project, mặc định sẽ có một Activity được tạo và chạy đầu tiên khi thực thi ứng dụng. Và ứng với một Activity sẽ có một Layout đi kèm – Layout mặc định. Để đổi Layout mặc định bằng Layout mới tự tạo, chúng ta vào lớp Activity.java và sửa lại dòng lệnh setContentView (dòng lệnh này dùng để kết nối Layout vào Activity) như sau:

setContentView(R.Layout.***layout\_cua\_tui***);

1. Đặt các điều khiển vào Layout và thiết lập, tùy chỉnh các thông số của các Layout

* Sử dụng Frame Layout thiết kế giao diện ở hình 3.1
  + Chúng ta đặt các điều khiển vào Layout như sau:

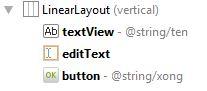


*Hình 3.8. Đặt các điều khiển vào FrameLayout.*

* + Và thiết lập thông số cho hình ở mục số 1 và mục số 2 như sau:

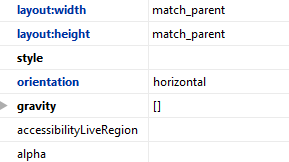
|  |  |
| --- | --- |
| *Hình 3.9. Thiết lập thông số cho hình ở mục 1.* | *Hình 3.10. Thiết lập thông số cho hình ở mục 2.* |

* Sử dụng Linear Layout để thiết kế giao diện ở hình 3.2 và hình 3.3
  + Đặt các điều khiển vào Linear Layout như sau:



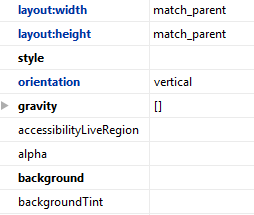
*Hình 3.11. Đặt các điều khiển vào Linear Layout.*

* + Ở hình 3.2 , các điều khiển được sắp sếp theo hướng từ trái qua phải.
  + Chúng ta thiết lập các thuộc tính cho Linear Layout như sau:



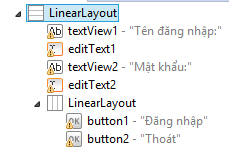
*Hình 3.12. Thiết lập các thuộc tính cho Linear Layout 1.*

* + Ở hình 3.3, các điều khiển được sắp xếp theo hướng từ trên xuống dưới. Chúng ta thiết lập các thuộc tính cho Linear Layout như sau:



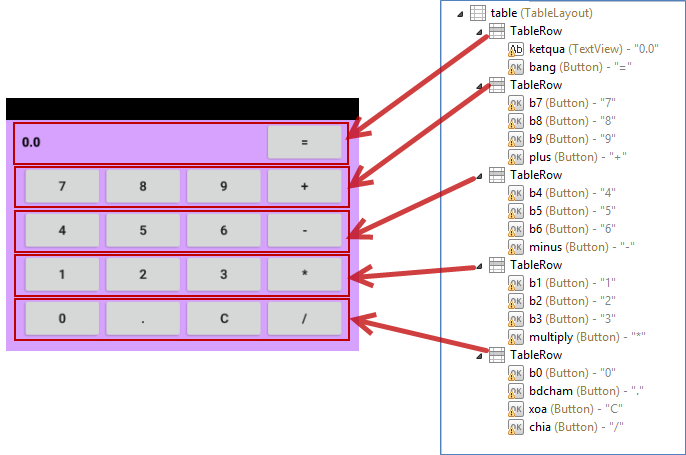
*Hình 3.13. Thiết lập các thuộc tính cho Linear Layout 2.*

* + Ở hình 3.4, chúng ta kết hợp sử dụng cả 2 thuộc tính Horizontal và Vertical để tạo giao diện. Thứ tự sắp xếp Layout và các điều khiển như sau:



*Hình 3.14. Thứ tự sắp xếp Layout và các điều khiển.*

* Sử dụng Table Layout để thiết kế giao diện ở hình 3.5
  + Chúng ta đặt các điều khiển như sau:



*Hình 3.15. Đặt các điều khiển vào Table Layout.*

* + Và chúng ta thiết lập các điều khiển với các thông số như sau:
    - Thiết lập thông số cho TextView ketqua:

<TextView

android:id=*"@+id/ketqua"*

android:layout\_span=*"3"*

android:text=*"0.0"*

android:textColor=*"#000000"*

android:textStyle=*"bold"* />

* + - Thiết lập thông số cho các Button ở TableRow thứ 2:

<TableRow

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

<Button

android:id=*"@+id/b7"*

android:text=*"7"* />

<Button

android:id=*"@+id/b8"*

android:text=*"8"* />

<Button

android:id=*"@+id/b9"*

android:text=*"9"* />

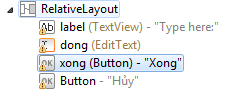
<Button

android:id=*"@+id/plus"*

android:text=*"+"* />

</TableRow>

* + - Các thông số cho các Button ở các TableRow còn lại, bạn có thể tự thiết lập tương tự.
* Sử dụng Relative Layout để thiết kế giao diện ở hình 3.6
  + Chúng ta đặt các điều khiển như sau:



*Hình 3.16. Đặt các điều khiển vào Relative Layout.*

* + Thiết lập thông số cho EditText (edt\_Dong) như sau:

<EditText

android:id=*"@+id/edt\_Dong"*

**android:Layout\_width=*"fill\_parent"***

**android:Layout\_height=*"wrap\_content"***

android:Layout\_below=*"@id/label"* />

* + Thiết lập thông số cho nút “Xong” như sau:

<Button

android:id=*"@+id/xong"*

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

**android:Layout\_alignParentRight=*"true"***

**android:Layout\_below=*"@id/edt\_Dong"***

**android:Layout\_marginLeft=*"10dip"***

**android:text=*"Xong"*** />

* + Thiết lập thông số cho nút “Hủy” như sau:

<Button

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

**android:Layout\_alignTop=*"@id/huy"***

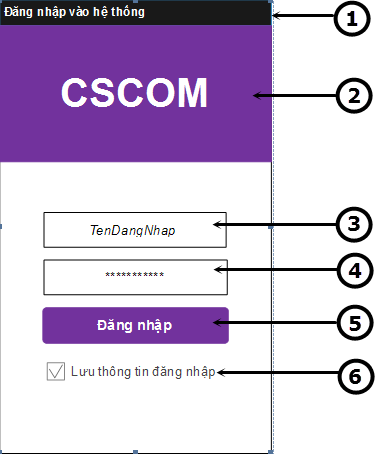
**android:Layout\_toLeftOf=*"@id/huy"***

**android:text=*"Hủy"*** />

* 1. Tạo giao diện đơn giản với Layout và các điều khiển cơ bản

Đề bài:

* Sử dụng Layout và các điểu khiển cơ bản để tạo giao diện như sau:



*Hình 3.17. Màn hình đăng nhập.*

* Trong đó:
  + Số 1 và số 2: tiêu đề và tên màn hình - sử dụng TextView để tạo.
  + Số 3: dùng để nhập tên đăng nhập – sử dụng EditText để tạo và đặt tên id là edt\_TenDangNhap.
  + Số 4: dùng để nhập mật khẩu – sử dụng EditText để tạo và đặt tên id là edt\_MatKhau.
  + Số 5: khi người dùng chọn vào nút này, nó sẽ xử lý thông tin đăng nhập mà người dùng đã nhập ở số 3 và 4 – sử dụng Button để tạo và đặt tên id là btn\_DangNhap.
  + Số 6: nếu người dùng bấm chọn phần này, thông tin đăng nhập sẽ được lưu cho những lần sau, không cần phải nhập thông tin đăng nhập lại – sử dụng Checkbox để tạo và đặt tên id là chkb\_LuuDangNhap.
* Nhưng ở bài tập này, chúng ta chỉ dừng lại ở việc thiết kế giao diện theo yêu cầu và thiết lập các thuộc tính cho các điều khiển.

Mục tiêu:

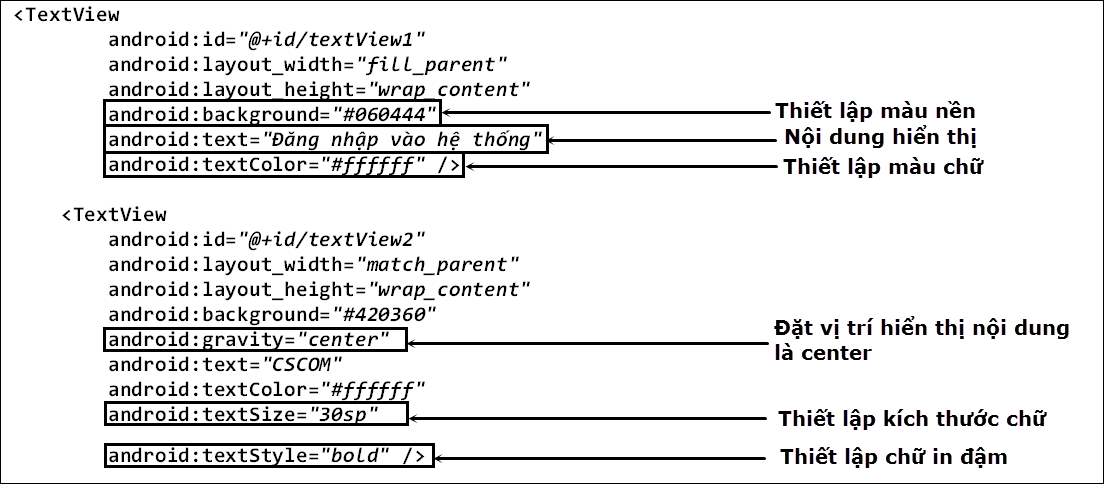
* Hiểu ý nghĩa và cách sử dụng các điều khiển cơ bản để thiết kế giao diện.
* Biết cách phối hợp sử dụng giữa Layout và các điều khiển để tạo được giao diện theo yêu cầu, đẹp.

Gợi ý thực hiện:

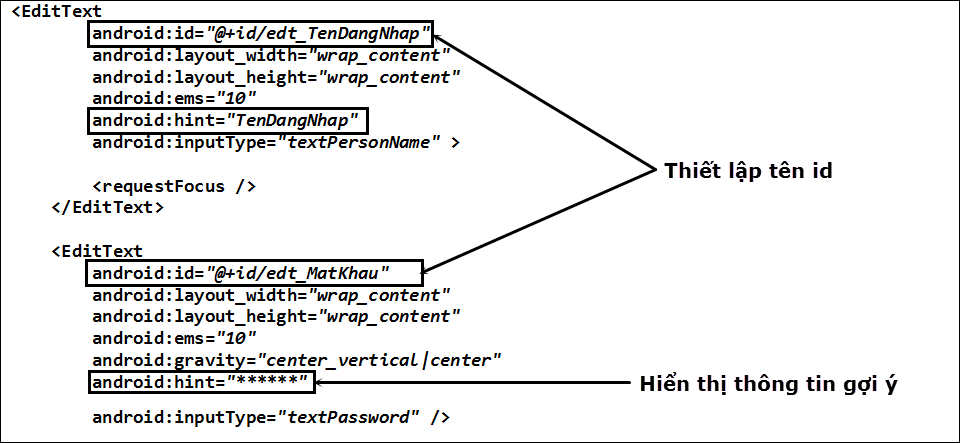
* Tạo mới Android Application Project đặt tên là DangNhap.
* Sử dụng Layout, TextView, EditText, CheckBox và Button thiết kế giao diện như hình 3.17 (có thiết lập id cho các điều khiển).
* Thiết lập các thuộc tính cần thiết cho các điều khiển trên.

Hướng dẫn chi tiết:

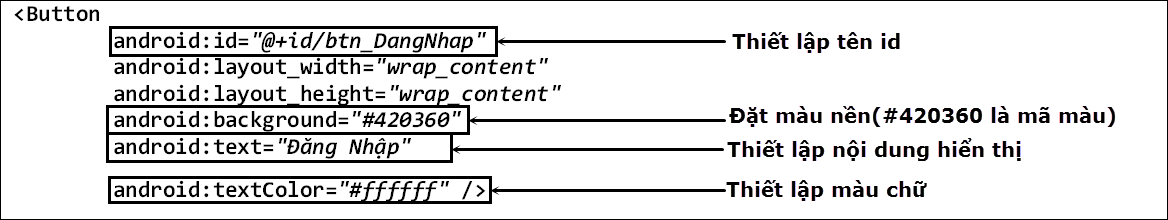
* Sau khi tạo mới Android Application Project đặt tên là DangNhap. Ta tiến hành thiết kế giao diện cho Màn hình đăng nhập theo yêu cầu. Trong đó, ta sử dụng:
  + ***TextView***: dùng để hiển thị các thông tin mà không cho phép người dùng chỉnh sửa. Ở đây, ta sử dụng TextView để thiết lập các câu, chữ cần hiển thị như: đăng nhập vào hệ thống, CSCOM (số 1 và 2 đã được đánh dấu trên hình 3.17).
  + ***EditText***: cho phép nhập và chỉnh sửa dữ liệu (chúng ta cũng có thể khóa chức năng cho chép chỉnh sửa thông tin nhập vào EditText bằng cách chỉnh sửa code trong tập tin java hoặc trong tập tin .xml). Ở bài này, ta sử dụng EditText để nhập tên đăng nhập (TenDangNhap) và mật khẩu (số 3 và 4 đã được đánh dấu trên hình 3.17).
  + ***Button***: dùng để thiết lập sự kiện khi người dùng chọn. Vì vậy, ta sử dụng Button để tạo nút Đăng nhập (số 5 đã được đánh dấu trên hình 3.17).
  + ***CheckBox***: cho phép chúng ta chọn một hay nhiều đối tượng cùng một lúc. Ở bài này, ta sử dụng CheckBox chỉ để chọn 1 thông tin là có lưu thông tin đăng nhập hay không (số 6 đã được đánh dấu trên hình 3.17).
* Sử dụng TextView từ Palette → Form Widget để tạo các thông tin hiển thị. Ở đây, chúng ta sử dụng 2 TextView để hiển thị thông tin cho số 1 và số 2. Ngoài ra, chúng ta có thể tạo và thiết lập các thuộc tính của TextView trong tập tin dang\_nhap.xml như sau:



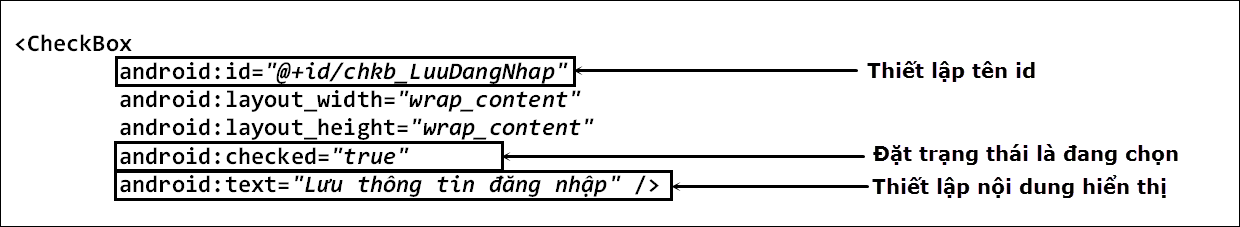
* Tương tự vậy, chúng ta có thể thiết lập các thuộc tính cho EditText trong tập tin dang\_nhap.xml như sau:



* + Và các thuộc tính cho Button như sau:



* + Các thuộc tính cho CheckBox:



* Do ở bài này, TextView chỉ dùng để hiển thị những thông tin, không có thiết lập sự kiện cho các TextView này nên chúng ta có thể không đặt tên id cho chúng. Còn các điều khiển còn lại, cần phải đặt tên id để thiết lập sự kiện cho chúng ở bài sau.
* *Chú ý:* 
  + Chúng ta cũng có thể thiết lập các thuộc tính cho Layout và các điều khiển bằng cách chọn Layout hoặc điều khiển cần thiết lập thuộc tính và thiết lập các thuộc tính của nó trong phần Properties hoặc trong tập tin dang\_nhap.xml.
  1. Thiết lập sự kiện cho các điều khiển cơ bản

Đề bài:

* Thiết lập sự kiện cho TextView, EditText, Button và CheckBox trên màn hình Đăng nhập. Yêu cầu thực hiện:
  + Nhập: tên đăng nhập, mật khẩu, trạng thái của CheckBox (CheckBox có được chọn hay không?).
  + Xử lý: kiểm tra tên đăng nhập có phải là “Admin” và mật khẩu có phải là “Admin123”.
  + Xuất: nếu kiểm tra tên đăng nhập và mật khẩu đã đúng thì có 1 Toast hiển thị thông báo “Đăng nhập thành công!” , ngược lại Toast sẽ hiển thị thông báo “Thông tin đăng nhập chưa đúng! Mời bạn nhập lại!”.

Mục tiêu:

* Biết cách thiết lập sự kiện cho các điều khiển trong Android (bài tập này hướng dẫn cách lấy các điều khiển và thiết lập sự kiện cho Button).

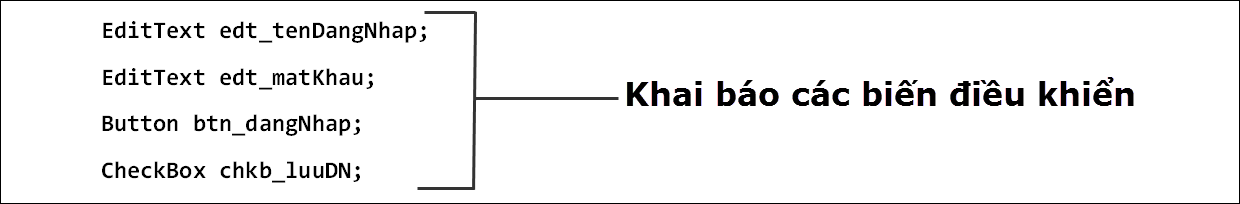
Gợi ý thực hiện:

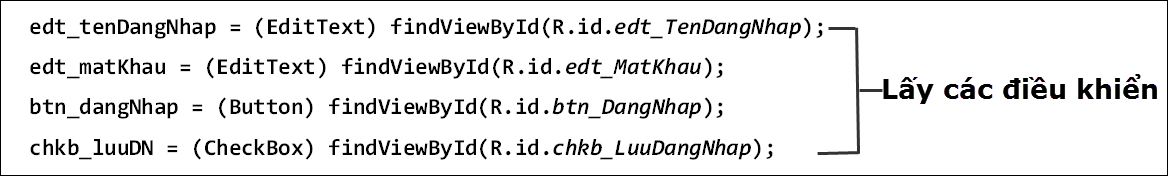
* Tiếp tục với Project ở Bài tập 3.2, sau khi hoàn thành việc thiết kế giao diện và thiết lập các thuộc tính cho các điều khiển, từ các id đã được thiết lập, ta viết code lấy các điều khiển và thiết lập sự kiện cho chúng trong lớp DangNhapActivity.java.

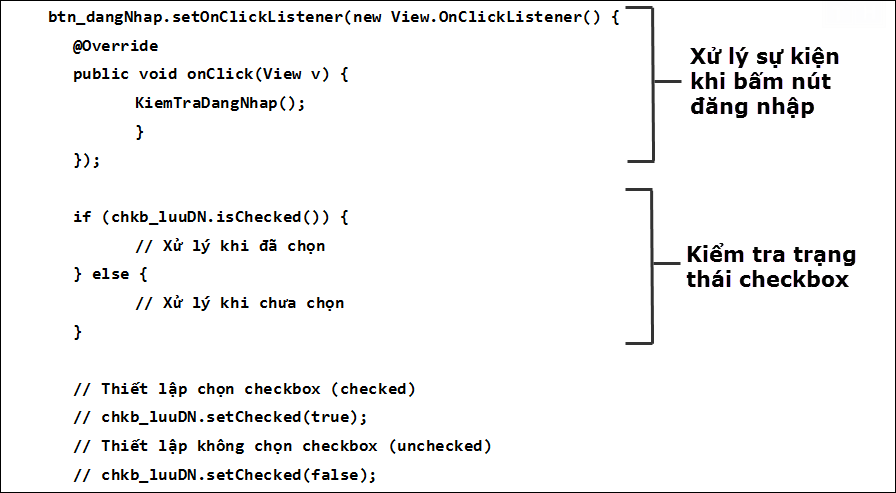
Hướng dẫn chi tiết:

1. Tạo giao diện, đặt tên id và thiết lập các thuộc tính cho các điều khiển (đã làm ở Bài tập 3.2).
2. Thiết lập sự kiện cho các điều khiển để thực hiện các yêu cầu xử lý.

* Vào tập tin DangNhapActivity.java, viết các câu lệnh để lấy các điều khiển như sau:





Tiếp theo, chúng ta viết các câu lệnh thiết lập sự kiện cho các điều khiển như sau: 

* + Phương thức kiểm tra đăng nhập:

**public** **void** KiemTraDangNhap() {

**//Lấy thông tin trên EditText**

String tendangnhap = edt\_tenDangNhap.getText().toString();

String matkhau = edt\_matKhau.getText().toString();

**//Kiểm tra thông tin nhập vào**

**if** (tendangnhap == "Admin" && matkhau == "Admin123") {

**//Toast thông báo thành công**

Toast.makeText(**this**, "Đăng nhập thành công!", Toast.LENGTH\_LONG).show();

} **else** {

Toast.makeText(**this**,"Thông tin đăng nhập chưa đúng! Mời bạn nhập lại!!",

Toast.LENGTH\_LONG).show();

}

}

* 1. Thiết lập sự kiện cho các điều khiển bằng 6 cách

Đề bài:

* Thiết lập sự kiện cho nút “Đăng nhập” trên màn hình Đăng nhập bằng 6 cách: Onclick in XML, Inline anonymous listener, Activity is listener, Listener in variable, Explicit listener class, View Subclassing.

Mục tiêu:

* Biết cách thiết lập sự kiện cho các điều khiển bằng 6 cách.
* Hiểu và nhận ra ưu điểm và khuyết điểm của mỗi cách thiết lập sự kiện.

Gợi ý thực hiện:

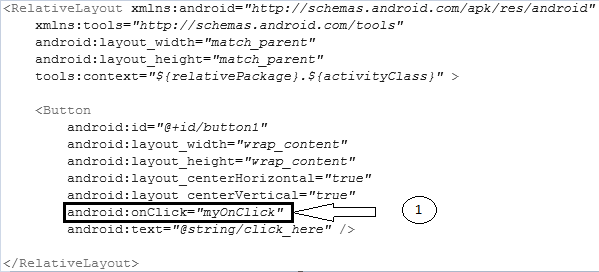
* Onclick in XML:

1. Kéo Button thả vào Layout cần thiết kế, sau đó thêm thuộc tính andorid:onClick= “myOnClick” trong thẻ Button vừa kéo vào.
2. Vào lớp java kế thừa từ Activity và chứa Layout vừa thiết kế trên, chúng ta thêm phương thức myOnClick và có đối số là View. Sau đó, viết nội dung cần xử lý khi nhấn lên Button. Chú ý:
   * Nếu nhiều Button trong Layout có cùng một một phương thức là “myOnClick” thì ta phải lấy đối số View ra kiểm tra id xem thử thuộc Button nào.
   * Có thể để mỗi Button có một phương thức khác nhau.

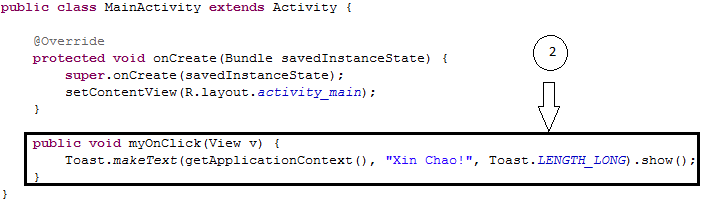
* ***Inline anonymous listener***: Sau khi “findViewById” trả ra đối tượng Button, chúng ta lấy đối tượng đó truy xuất đên phương thức “setOnClickListener” và đối số bên trong là một Interface “OnClickListener”, khi tạo mới Interface thì có Override lại phương thức “onClick”, chúng ta sẽ viết code xử lý trong phương thức đó.
* ***Activity is listener***: Trong lớp java có kế thừa Activity, chúng ta “implements” thêm Interface “OnClickListener”. Sau đó, chúng ta Override lại phương thức “onClick” - nội dung xử lý đều viết trong phương thức này. Nhưng để Button nào xử lý trong phương thức “onClick” này thì ta phải lấy Button đó truy xuất đến phương thức “setOnClickListener” và truyền đối số vào là “this”(tức là chính lớp mà có implements Interface OnClickListener). Chú ý: Nếu nhiều Button đều dùng thì ta phải lấy đối số View trong phương thức onClick để kiểm tra id xem thuộc Button nào để xử lý.
* ***Listener in variable***: Đầu tiên, chúng ta tạo biến của Interface và khởi tạo “new”. Sau khi khởi tạo thì có Override lại phương thức “onClick” - nội dung xử lý viết trong phương thức này. Để sử dụng cho Button nào thì ta chỉ cần lấy Button đó truy xuất đến phương thức “setOnClickListener” và truyền đối số là biến Interface vừa tạo ở trên.
* ***Explicit listener class***: Nếu dữ liệu xử lý quá lớn thì ta xử lý theo cách này. Tức là tạo 1 class riêng có “implements” Interface “OnClickListener” rồi tất cả xử lý viết trong phương thức “onClick” trong class này.
* ***View Subclassing***: cách này thì ta phải override phương thức “performClick” của chính Button. Tức là sau khi khởi tạo Button thì ta phải override lại và viết xử lý trong đó.

Hướng dẫn chi tiết:

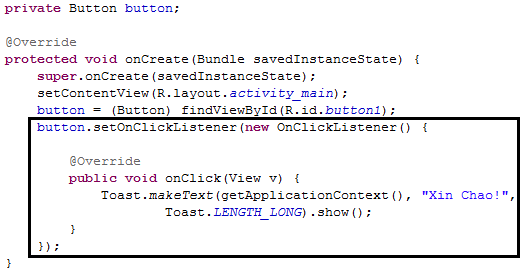
* ***Onclick in XML:***
  + Tạo Button trong file XML như sau:



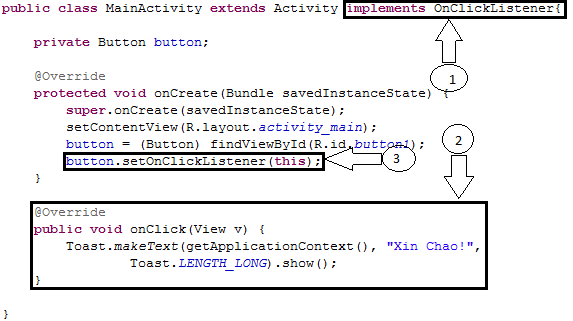
* + - Giải thích: (1) là đặt tên phương thức(myOnClick) để bắt sự kiện khi click lên Button này.
  + Vào class Activity viết code như sau:



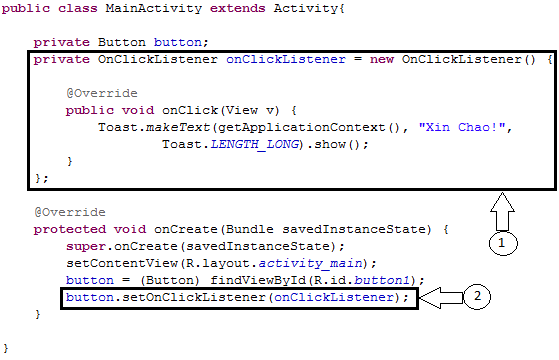
* + - Giải thích: (2) là phương thức để bắt lại sự kiện nhấn lên Button và viết dòng code xử lý để nhận biết đã bắt được.
* ***Inline anonymous listener***: trong file class Activity ta viết như sau:



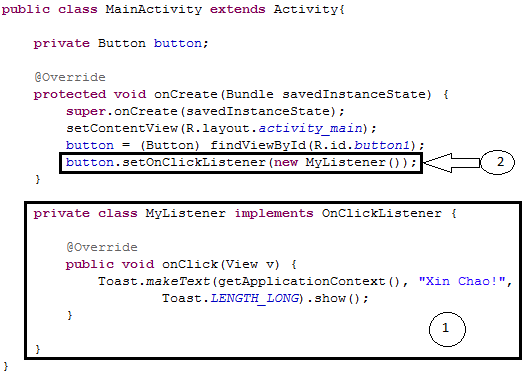
* + Giải thích: ta dùng button truy xuất đến phương thức “setOnClickListener” và truyền vào 1 Interface để bắt lại sự kiện khi nhấn. Bên trong Interface có phương thức “onClick” dùng để viết các xử lý.
* ***Activity is listener*:** trong file class Activity ta viết như sau:



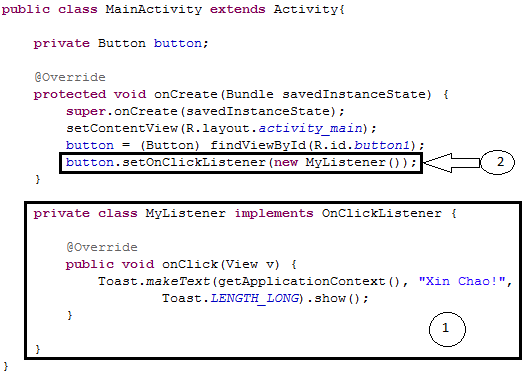
* + Giải thích: (1) là implements lại Interface OnClickListener. (2) là override lại phương thức của Interface và viết code xử lý trong phương thức này. (3) là cài đặt sự kiện khi nhấn vào Button sẽ xử lý như thế nào.
* ***Listener in variable***:



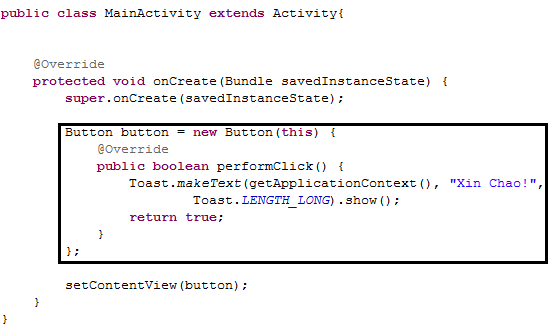
* + Giải thích: (1) là tạo biến onClickListener của Interface. Nhưng khi tạo new lên thì phải override lại phương thức và viết nội dung cần xử lý. (2) là cài đặt sự kiện khi nhấn vào Button sẽ xử lý như thế nào.
* ***Explicit listener class:***



* + Giải thích: (1) là tạo lớp thực thi Interface OnClickListener. Sau đó, override lại phương thức của Interface OnClickListener rồi viết xử lý trong đó. (2) là cài đặt sự kiện khi nhấn vào Button sẽ xử lý như thế nào.



* ***View Subclassing:***



* + Giải thích: Khởi tạo new một Button, sau đó cho override lại phương thức performClick để viết xử lý khi nhấn vào Button này.
  1. Tạo giao diện với ImageView

Đề bài:

* Sử dụng các điều khiển cơ bản để tạo giao diện như sau:



*Hình 3.18. Màn hình Chỉnh sửa thông tin cá nhân.*

* Yêu cầu thực hiện:
  + Mục số 1: sử dụng TextView để tạo.
  + Mục số 2: sử dụng EditText để tạo. Người dùng sẽ nhập các thông tin vào đây.
  + Mục số 3: sử dụng ImageView để tạo. Người dùng sẽ thay đổi hình đại diện của mình ở đây. Cho phép người dùng chọn ảnh từ thiết bị và hiển thị lên màn hình.
  + Mục số 4: sử dụng Button để tạo.
  + Nhập: số chứng minh nhân dân (CMND), số điện thoại, địa chỉ.
  + Xử lý:
    - Khi bấm nút “Cập nhật”: kiểm tra các thông tin nhập vào có thông tin nào còn trống hay không? Số CMND và số điện thoại phải là số. Số CMND phải đúng 9 số.
    - Khi bấm vào nút “Làm lại”: các thông tin đang hiển thị trên các EditText sẽ được xóa trống, dấu trỏ nhập nằm ở EditText dùng để nhập số chứng minh nhân dân.
    - Khi bấm vào mục số 3 (hình đại diện) sẽ cho phép người dùng thay đổi hình đại diện.
    - Xuất: nếu kiểm tra các thông tin đều đúng thì có 1 Toast để hiển thị thông báo “Cập nhật thông tin thành công!”, ngược lại Toast sẽ hiển thị thông báo “Thông tin cập nhật chưa đủ hoặc chưa đúng! Mời bạn cập nhật lại!”.

Mục tiêu:

* Biết các thiết kế giao diện theo yêu cầu với Layout và các điều khiển cơ bản.
* Biết sử dụng các thuộc tính của các đối tượng View.

Gợi ý thực hiện:

* Sử dụng TextView, EditText, Button và ImageView để tạo các giao diện theo yêu cầu.
* Lấy điều khiển và thiết lập sự kiện cho các Button, EditText, ImageView.

Hướng dẫn chi tiết:

* Vận dụng những kiến thức đã biết về TextView, EditText và Button để làm bài tập này. Ngoài ra, chúng ta sẽ tìm hiểu thêm về ImageView. (Phần này chỉ hướng dẫn về ImageView, các điều khiển khác bạn có thể tự tạo giao diện với nó dự vào kiến thức đã biết).

1. Tạo các giao diện theo yêu cầu.

* Chọn ImageView từ Palette → Form Widget để vào vị trí ở mục số 3. Ngoài ra, chúng ta có thể tạo và thiết lập các thuộc tính của ImageView, đặt tên id cho ImageView là img\_HinhDaiDien trong tập tin cap\_nhat\_thong\_tin\_ca\_nhan.xml tương tự như sử dụng TextView, EditText hoặc Button.

1. Lấy điều khiển và thiết lập sự kiện cho ImageView.

* Tương tự các điều khiển khác, chúng ta khai báo biến điều khiển trong lớp CapNhatTTCN.java (Activty – Màn hình Chỉnh sửa thông tin cá nhân) như sau:

ImageView img\_hinhDaiDien;

* Tiếp theo, chúng ta cũng lấy ImageView tương tự như với các điều khiển khác:

Img\_hinhDaiDien = (ImageView) findViewById(R.id.img\_HinhDaiDien);

* Thiết lập sự khiện khi người dùng bấm vào hình đại diện (ImageView) để thay đổi ảnh đại diện:

img\_hinhDaiDien.setOnClickListener(**new** View.OnClickListener() {

@Override

**public** **void** onClick(View v) {

Intent intent = **new** Intent(Intent.ACTION\_GET\_CONTENT);

intent.setType("image/\*");

Intent chon = Intent.createChooser(intent, "Chọn một ảnh đại diện");

startActivityForResult(chon, 0);

}

});

* Đặt ảnh hiển thị sau khi người dùng chọn ảnh:

@Override

**protected** **void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {

**super**.onActivityResult(requestCode, resultCode, data);

**if**(requestCode == 0 && resultCode == Activity.RESULT\_OK)

{

img\_hinhDaiDien.setImageURI(data.getData());

}

}

* Kiểm tra khi bấm nút “Cập nhật” và xử lý khi người dùng bấm vào nút “Làm lại”.
  1. Bài tập làm thêm

Đề bài:

* Sử dụng các điều khiển cơ bản để thiết kế giao diện như sau:



*Hình 3.19. Màn hình Thêm khách hàng.*

Yêu cầu thực hiện:

* Nhập: tên khách hàng, số điện thoại của khách hàng, địa chỉ của khách hàng.
* Xử lý:
  + Khi bấm vào nút “Thêm”: kiểm tra các thông tin nhập vào có thông tin nào còn trống hay không? Số điện thoại phải là số. Số CMND phải đúng 9 số.
  + Khi bấm vào nút “Làm lại”: các thông tin đang hiển thị trên các EditText sẽ được xóa trống, dấu trỏ nhập nằm ở EditText dùng để nhập tên khách hàng.
  + Khim bấm vào hình đại diện, chương trình sẽ cho phép người dùng thay đổi hình đại diện.
* Xuất: nếu kiểm tra các thông tin nhập vào đều đúng thì có 1 Toast để hiển thị thông báo “Thêm khách hàng thành công!”, ngược lại Toast sẽ hiển thị thông báo “Thêm thất bại!”.
* *Chú ý*: nên đặt tên các điều khiển đúng chuẩn (Ví dụ: tên EditText để nhập số CMND nên đặt là edt\_CMND).

Mục tiêu:

Gợi ý thực hiện:

* Vận dụng những kiến thức đã được biết, sử dụng TextView, EditText, Button, ImageView để tạo các giao diện theo yêu cầu.
* Thiết lập sự kiện cho các Button, EditText, ImageView.

Hướng dẫn chi tiết:

1. TÀI NGUYÊN ỨNG DỤNG TRONG ANDROID

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Biết và làm việc được với các tài nguyên ứng dụng cơ bản trong Android như: String, Bool, Color, Dimen, Integer, Integer Array, Typed Array và áp dụng vào trong việc xây dựng các ứng dụng trong Android. |

* 1. Làm quen với các tài nguyên ứng dụng cơ bản trong Android

Đề bài:

* Sử dụng tài nguyên String và truy xuất đến tài nguyên vừa tạo để hiển thị nội dung lên TextView, EditText, Button và CheckBox cho các màn hình giao diện.

Mục tiêu:

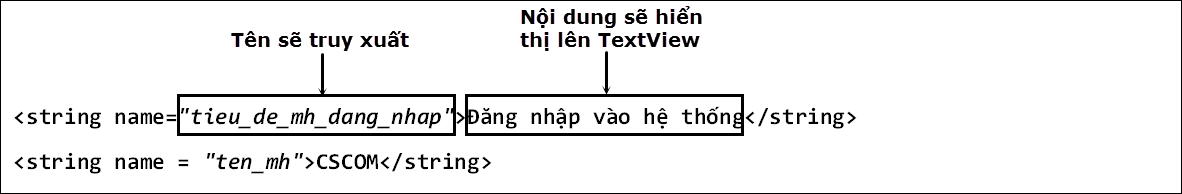
* Hiểu được ý nghĩa và cách sử dụng của các tài nguyên cơ bản.

Gợi ý thực hiện:

* Vào tập tin string.xml để định nghĩa các String trong thư mục res/ values.
* Truy xuất đến tập tin string.xml để sử dụng các tài nguyên String.

Hướng dẫn chi tiết:

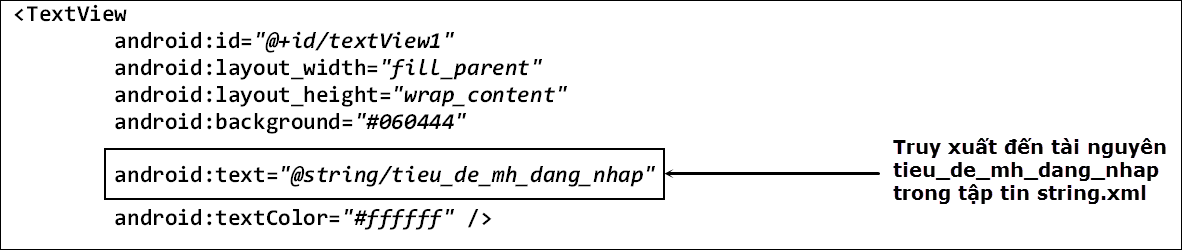
1. Vào tập tin string.xml để định nghĩa các String trong thư mục res/ values như sau:



* Tương tự như trên, ta định nghĩa tiếp các String để hiển thị nội dung lên EditText, Button và CheckBox.

1. Truy xuất đến tập tin string.xml để sử dụng các tài nguyên String đã định nghĩa

* *Cách 1:* Chỉnh lại nội dung trong tập tin .xml
  + *Ví dụ:* chỉnh lại nội dung trong tập tin dang\_nhap.xml kế thừa từ bài 3 – Bài tập 3.2 (Màn hình đăng nhập) như sau:



* *Cách 2*: Viết lệnh trong tập tin Activity.java để truy xuất
  + Ví dụ: Chỉnh lại câu lệnh trong tập tin DangNhapActivity.java kế thừa từ bài 3 – Bài tập 3.2 (Màn hình đăng nhập) như sau:



* Tương tự như trên, ta định nghĩa tiếp các String để hiển thị nội dung lên các điều khiển còn lại.
  1. Sử dụng các tài nguyên String

Đề bài:

* Viết chương trình dùng để hỗ trợ đa ngôn ngữ cho ứng dụng.

Mục tiêu:

* Biết cách quản lý các tài nguyên chuỗi và cách dùng tài nguyên chuỗi.
* Xây dựng được các ứng dụng hổ trợ nhiều ngôn ngữ.

Gợi ý thực hiện:

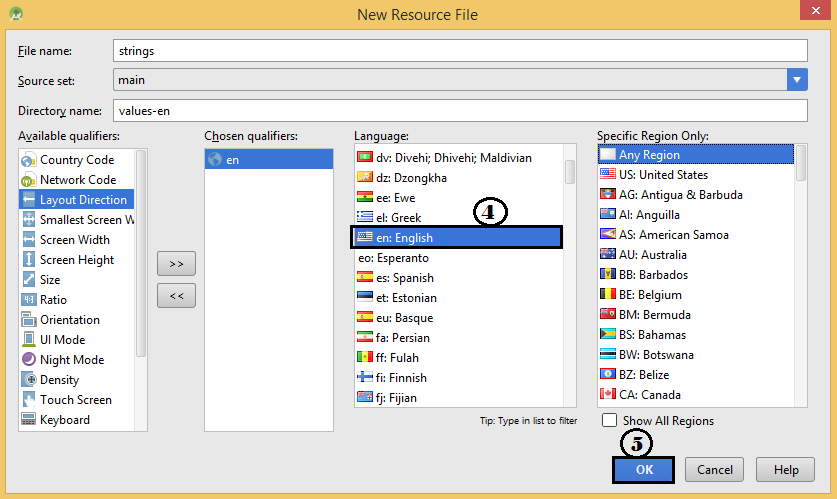
* Tạo một tập tin .xml và thiết lập ngôn ngữ muốn hỗ trợ.
* Vào setting chuyển đổi ngôn ngữ muốn sử dụng (những ngôn ngữ đã được thiết lập), chương trình sẽ tự lấy đúng tài nguyên XML ngôn ngữ mà chúng ta đã tạo.

Hướng dẫn chi tiết:

1. Tạo Android XML File và thiết lập ngôn ngữ muốn hỗ trợ

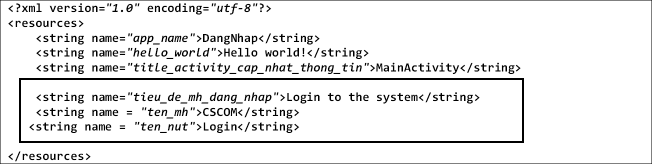
* Nhấp chuột phải vào thư mục values → New → Values resource file:





*Hình 4.1. Hướng dẫn sử dụng tài nguyên String hỗ trợ đa ngôn ngữ - 1.*

* Ở hình 4 -3, mục số 2 là chọn mã quốc gia, ở đây chọn mã là “en” có nghĩa là chọn ngôn ngữ tiếng Anh. Còn mục số 3 là thư mục sẽ được nối kèm ở sau thư mục values.
* Vào tập tin values.xml trong thư mục values-en vừa tạo và chỉnh sửa nội dung tập tin để tạo các tài nguyên string hỗ trợ ngôn ngữ tiếng Anh.
  + *Ví dụ:* Viết các câu lệnh trong tập tin values.xml trong thư mục values-en để hổ trợ ngôn ngữ tiếng Anh cho màn hình đăng nhập như sau:



1. Vào mục setting của thiết bị (điện thoại) chuyển đổi ngôn ngữ sang tiếng Anh. Sau đó, chạy ứng dụng để xem kết quả.

* Thực hiện tương tự như vậy để hỗ trợ các ngôn ngữ khác cho ứng dụng mà bạn muốn.
  1. Sử dụng tài nguyên Color

Đề bài:

* Sử dụng tài nguyên Color để tạo màu chữ cho tên tiêu đề, màu của EditText, màu cho các Button trong các màn hình giao diện.

Mục tiêu:

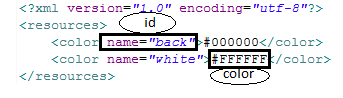
* Biết cách quản lý tài nguyên màu và cách dùng tài nguyên màu.

Gợi ý thực hiện:

* Tạo một tập tin mau.xml trong thư mục res/ values.
* Viết đoạn mã lệnh thiết lập các tài nguyên màu sắc trong tập tin mau.xml vừa tạo.
* Truy xuất đến các tài nguyên trong tập tin mau.xml tương tự như sử dụng tài nguyên string.

Hướng dẫn chi tiết:

1. Click chuột phải vào thư mục values --> New --> Values resource.
2. Đặt tên cho tập tin là “color” --> Ok.
3. Viết những mã màu cần dùng vào như sau:



* *Giải thích:* “id” dùng để truy xuất đến lấy mã màu “color”

1. Vào tập tin .xml (Layout) và lớp .java truy xuất đến → chạy chương trình xem kết quả.
   1. Sử dụng tài nguyên Dimen

Đề bài:

* + Viết chương trình giúp chữa lỗi màn hình bị mất một phần khi xoay thiết bị xoay nằm ngang.

Mục tiêu:

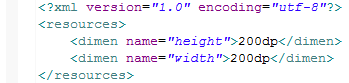
* Biết cách quản lý các tài nguyên về kích thước, các kiểu đơn vị tính trong Android và các dùng.

Gợi ý thực hiện:

* Tạo một tập tin kichthuoc.xml trong thư mục res/ values.
* Tạo các tài nguyên để thiết lập kích thước màn hình theo chiều dọc và chiều ngang trong tập tin kichthuoc.xml vừa tạo.
* Truy xuất đến các tài nguyên trong tập tin kichthuoc.xml tương tự như sử dụng tài nguyên string.

Hướng dẫn chi tiết:

1. : Tạo tương tự như tài nguyên String.
2. Viết các câu lệnh như sau:



1. Vào tập tin .xml (Layout) và lớp .java truy xuất đến → chạy chương trình xem kết quả.
2. INTENT

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Hiểu, xây dựng và truy xuất được Intent, Intent Filter trong Android.  Phân biệt được Intent và Intent Filter. |

* 1. Xây dựng Intent để mở một Actitvity khác nhưng không kiểm soát kết quả trả về (Implicit Intent)

Đề bài:

* Sử dụng các điều khiển cơ bản để tạo giao diện như sau:



*Hình 5.1. Giao diện màn hình chọn chức năng.*

Yêu cầu thực hiện:



*Hình 5.2. Yêu cầu thực hiện giao diện màn hình chọn chức năng.*

* Xây dựng màn hình Đăng nhập sao cho khi người dùng đăng nhập thành công, ứng dụng sẽ mở màn hình Chọn chủ đề (Hình 5. 1) cho phép người dùng chọn các chủ đề (các chức năng).
* Mục số 1: cho phép người dùng thoát khỏi ứng dụng khi người dùng chọn vào nút này (nút “Thoát”).
* Mục số 2: khi người dùng bấm vào hình biểu tượng ở mục này, ứng dụng sẽ mở màn hình mới cho phép người dùng cập nhật thông tin cá nhân vào màn hình Cập nhật thông tin cá nhân.

Mục tiêu:

* Biết cách chuyển giữa các màn hình với nhau.

Gợi ý thực hiện:

* Tạo giao diện theo yêu cầu: mục số 1 – dùng Button để tạo, các mục còn lại dùng ImageView và TextView để tạo.
* Viết mã lệnh thiết lập sự kiện khi người dùng bấm nút “Thoát”.
* Xây dựng Intent để mở màn hình cập nhật thông tin cá nhân khi người dùng bấm vào hình biểu tượng ở mục số 2.

Hướng dẫn chi tiết:

1. Tạo giao diện theo yêu cầu
2. Xử lý sự kiện khi người dùng bấm nút “Thoát”
3. Xây dựng Intent để mở màn hình mới

* Mã lệnh xây dựng Intent đặt tên là chuDeIntent để mở màn hình Chọn chủ đề (ChonChuDeActivity) từ màn hình đăng nhập (DangNhapActivity):

Intent chuDeIntent =**new** Intent(DangNhapActivity.**this**, ChonChuDeActivity.**class**);

startActivity(chuDeIntent ); *// Mở chuDeIntent đã được truyền vào*

* Tương tự như vậy, chúng ta có thể xây dựng Intent để mở màn hình Cập nhật thông tin cá nhân đã xây dựng ở Bài 3 – câu 3.7 khi người dùng bấm vào hình biểu tượng ở mục Cá nhân (mục số 2).
  1. Xây dựng Intent để mở một Actitvity khác và có kiểm soát kết quả trả về (Explicit Intent)

Đề bài:

* Chuẩn bị 2 màn hình: Thêm khách hàng và màn hình Thông tin khách hàng thêm mới. Viết chương trình sao cho:
  + Sau khi đã kiểm tra thêm thành công, ứng dụng sẽ mở ra một màn hình mới – tạm gọi là màn hình thông tin khách hàng thêm mới để hiển thị thông tin của khách hàng vừa được nhập ở màn hình Thêm khách hàng. Ở màn hình Thông tin khách hàng thêm mới có nút “Trở về” – khi người dùng bấm vào nút này sẽ trở lại màn hình Thêm khách hàng.

Mục tiêu:

* Biết gửi dữ liệu đi khi chuyển 2 màn hình với nhau.

Gợi ý thực hiện:

* Tạo một Actitvity để làm màn hình hiển thị thông tin khách hàng vừa được thêm mới có có 2 phần: phần 1 là các TextView để chứa các thông tin sẽ hiển thị và phần 2 là Button để tạo nút “Trở về”.
* Viết các câu lệnh xây dựng Intent để mở và chuyển dữ liệu từ màn hình Thêm khách hàng sang màn hình Thông tin khách hàm thêm mới.
* Viết mã lệnh xử lý khi người dùng bấm nút “Trở về” – từ màn hình Thông tin khách hàng thêm mới trở về màn hình Thêm mới khách hàng.

Hướng dẫn chi tiết:

1. Tạo một Activity để làm màn hình hiển thị thông tin khách hàng

* Dựa vào kiến thức ở những bài trước để làm bước này.

1. Viết mã lệnh xây dựng Intent để mở và chuyển dữ liệu từ màn hình Thêm khách hàng sang màn hình Thông tin khách hàng thêm mới như sau:

* Xử lý xây dựng Intent trong lớp ThemKhachHang.java

*//Tạo Intent để mở Activity ThongTinKHThemMoi*

Intent myIntent=**new** Intent(ThemKhachHang.**this**, ThongTinKHThemMoi.**class**);

*//Khai báo Bundle*

Bundle bundle=**new** Bundle();

*//Lấy nội dung từ các EditText*

String ten = edtTen.getText().toString();

String sodt = edtSDT.getText().toString();

String diachi = edtDiaChi.getText().toString();

*//Đưa dữ liệu riêng lẻ vào Bundle*

bundle.putString("tenkh", ten);

bundle.putString("sodt", sodt);

bundle.putString("diachi", diachi);

*//Đưa Bundle vào Intent*

myIntent.putExtra("goicuatui", bundle);

*//Mở Activity ThongTinKHThemMoi*

startActivity(myIntent);

* *Chú ý:* Để truyền và nhận dữ liệu là dạng đối tượng thì chúng ta sử dụng 2 phương thức: putSerializable và getSerialiazable.
  + Xử lý lấy dữ liệu từ Activity Thêm khách hàng truyền qua trong lớp ThongTinKHThemMoi.java

*//Lấy intent gọi Activity này*

Intent goiIntent=getIntent();

*//Có intent rồi thì lấy Bundle dựa vào MyPackage*

Bundle goi = goiIntent.getBundleExtra("goicuatui");

*//Có Bundle rồi thì lấy các thông số dựa vào tenkh, sodt, diachhi*

String ten = goi.getString("tenkh");

String sodt = goi.getString("sodt");

String diachi = goi.getString("diachi");

*//Tiến hành xử lý hiển thị thông tin lên TextView*

txtTen = ten.setText(ten);

txtSoDT = sodt.setText(sodt);

txtDiaChi = diachi.setText(diachi);

1. Viết mã lệnh xử lý nút “Trả về” để chuyển từ màn hình Thông tin khách hàng thêm mới sang màn hình Thêm mới khách hàng bằng cách tạo Intent không có kiểm soát kết quả trả về (Implicit Intent) tương tự như ở Câu 5.1.
2. Thực thi và quan sát (chú ý phần code xây dựng Intent lúc chạy chương trình).

1. QUẢN LÝ ASSET – SHARED PREFERENCES – BỘ NHỚ THIẾT BỊ

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Hiểu và thực hiện các thao tác: đọc, ghi, xóa dữ liệu ở bộ nhớ trong, ở bộ nhớ ngoài và với SharePreferences.  Tạo được Shared Prefrence Change Listeners. |

* 1. Sử dụng SharePreferences để lưu thông tin đăng nhập

Đề bài:

* Xây dựng màn hình Đăng nhập sao cho khi người dùng chọn vào Checkbox ứng dụng sẽ lưu lại thông tin đăng nhập (tên đăng nhập và mật khẩu) để ở những lần đăng nhập tiếp theo người dùng không phải đăng nhập lại.

Mục tiêu:

* Hiểu sâu hơn về SharePreferences và thao tác lưu trữ dữ liệu được với SharePreferences.

Gợi ý thực hiện:

* Viết mã lệnh xử lý lưu thông tin đăng nhập (tên đăng nhập và mật khẩu) với SharePreferences trong lớp DangNhapActivity.java.
* Viết các phương thức lưu và đọc trạng thái để trong 2 phương thức trạng thái là onPause() và onResume().

Hướng dẫn chi tiết:

* Xử lý lưu thông tin đăng nhập với SharePreferences:
  + Phương thức lưu trạng thái:

@Override

**protected** **void** onPause() {

**super**.onPause();

//gọi phương thức lưu trạng thái ở đây

luuPreferences();

}

**public** **void** luuPreferences()

{

//Tạo đối tượng getSharedPreferences

SharedPreferences pre=getSharedPreferences (prefname, MODE\_PRIVATE);

//Tạo đối tượng Editor để lưu thay đổi

SharedPreferences.Editor editor=pre.edit();

String tendangnhap = edtTenDangNhap.getText().toString();

String matkhau = edtMatKhau.getText().toString();

**boolean** chkb\_LuuDN = chkb\_LuuDangNhap.isChecked();

**if**(!chkb\_LuuDN)

{

//Xóa mọi lưu trữ trước đó

editor.clear();

}

**else**

{

//Lưu vào editor

editor.putString("tendangnhap", tendangnhap);

editor.putString("matkhau", matkhau);

editor.putBoolean("checked", chkb\_LuuDN);

}

//chấp nhận lưu xuống file

editor.commit();

}

* + Phương thức đọc trạng thái đã lưu:

@Override

**protected** **void** onResume() {

**super**.onResume();

//Gọi phương thức đọc trạng thái ở đây

docTrangThaiPreferences();

}

**public** **void** docTrangThaiPreferences()

{

SharedPreferences pre = getSharedPreferences(prefname,MODE\_PRIVATE);

//Lấy giá trị checked ra, nếu không thấy thì giá trị mặc định là false

**boolean** bchk = pre.getBoolean("checked", **false**);

**if**(chkb\_LuuDN)

{

//Lấy tên đăng nhập, mật khẩu, nếu không thấy giá trị mặc định là rỗng

String ten = pre.getString("tendangnhap", "");

String matkhau = pre.getString("matkhau", "");

edtTen.setText(tendangnhap);

edtMatKhau.setText(matkhau);

}

chkb\_LuuDangNhap.setChecked(chkb\_LuuDN);

}

* 1. Lưu dữ liệu vào bộ nhớ trong – bộ nhớ ngoài

Đề bài:

* Xây dựng màn hình Thêm thông tin khách hàng và viết xử lý sao cho khi người dùng thêm thông tin khách hàng thành công, thông tin khách hàng sẽ được:
  + Lưu trữ xuống bộ nhớ trong theo thư mục mặc định của hệ thống.
  + Lưu trữ xuống bộ nhớ trong theo thư mục riêng – do người dùng định nghĩa.
  + Lưu trữ xuống bộ nhớ ngoài (thẻ nhớ – SDCard).
* Tạo màn hình Sửa thông tin khách hàng tương tự như màn hình Thêm khách hàng cho phép sửa số điện thoại và địa chỉ của khách hàng. Với yêu cầu thực hiện:
  + Thông tin về số điện thoại và địa chỉ của khách hàng được đọc lên hiện thị lên EditText (edtSDT và edtDiaChi) từ thư mục mặc định của hệ thống.
  + Thông tin về số điện thoại và địa chỉ của khách hàng được đọc lên hiện thị lên EditText (edtSDT và edtDiaChi) từ thư mục riêng do người dùng định nghĩa.
  + Thông tin về số điện thoại và địa chỉ của khách hàng được đọc lên hiện thị lên EditText (edtSDT và edtDiaChi) từ bộ nhớ ngoài (thẻ nhớ – SDCard).

Mục tiêu:

* Biết đọc, ghi file ở bộ nhớ trong và ngoài.
* Biết cách xin quyền khi dùng đến tài nguyên người dùng.

Gợi ý thực hiện:

* Sử dụng SharePrefeences xử lý lưu thông tin thêm khách hàng mới (tên số điện thoại và địa chỉ) với 3 cách lưu trữ và đọc thông tin đã lưu hiển thị lên EditText từ 3 nơi đã lưu.
* Sử dụng các hàm trong thư viện FileInputStream và FileOutputStream.
* Phải thiết lập các quyền truy cập tong tập tin AndroidManifest.
* Chạy chương trình và mở DDMS để kiểm tra các tập tin đã lưu.

Hướng dẫn chi tiết:

1. Xử lý lưu tữ tập tin

* Lưu thông tin vào bộ nhớ trong theo thư mục mặc định của hệ thống:

**private** **void** GhiBoNhoTrong1() {

String dulieu = edtTenKH.getText().toString() + edtSoDT.getText().toString() +

edtDiaChi.getText().toString();

FileOutputStream fos;

**try** {

fos = openFileOutput(taptinluu, Context.MODE\_PRIVATE);

fos.write(dulieu.getBytes());

fos.close();

Toast.makeText(**this**, "Da ghi thanh cong" + dulieu, Toast.LENGTH\_SHORT).show();

} **catch** (IOException e) {

Toast.makeText(**this**, "Da ghi that bai!", Toast.LENGTH\_SHORT).show();

Log.*e*(getClass().getName(), e.toString());

}

}

* Lưu dữ liệu trong thư mục tự định nghĩa:

**private** **void** GhiBoNhoTrong2() {

File thumuc = getDir("ThuMucTuTao", Context.MODE\_PRIVATE);

**try** {

File f = **new** File(thumuc, taptinluu2);

FileOutputStream fos = **new** FileOutputStream(f);

String dulieu = edtTenKH.getText().toString() + edtSoDT.getText().toString() +

edtDiaChi.getText().toString();

fos.write(dulieu.getBytes());

fos.flush();

fos.close();

Toast.makeText(**this**, "Da ghi thanh cong", Toast.LENGTH\_SHORT).show();

} **catch** (Exception e) {

Toast.makeText(**this**, "Da ghi that bai", Toast.LENGTH\_SHORT).show();

Log.*e*(getClass().getName(), e.toString());

}

}

* Lưu dữ liệu vào bộ nhớ ngoài (thẻ nhớ – SDCard):

**private** **void** GhiVaoThe() {

**try** {

File f = **new** File(Environment.getExternalStorageDirectory().getPath(),tentaptin2);

FileOutputStream fos = **new** FileOutputStream(f);

OutputStreamWriter osw = **new** OutputStreamWriter(fos);

String dulieu = edtTenKH.getText().toString() + edtSoDT.getText().toString() +

edtDiaChi.getText().toString();

osw.append(dulieu);

osw.close();

fos.close();

Toast.makeText(**this**, "Da ghi vao the nho thanh cong! ",Toast.LENGTH\_SHORT).show();

} **catch** (Exception e) {

Toast.makeText(**this**, "Ghi vao the nho that bai!",Toast.LENGTH\_SHORT).show();

Log.*e*(getClass().getName(), e.toString());

}

}

1. Thiết lập các quyền tuy cập trong tập tin AndroidManifest

* Để có thể ghi và đọc trên SDCard thì phải khai báo 2 permission trong AndroidManifest như sau:

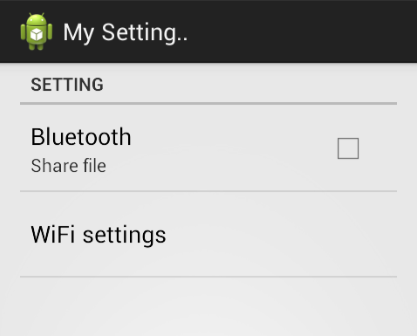
<uses-permission android:name=”android.permission.WRITE\_EXTERNAL\_STORAGE” />

<uses-permission android:name=”android.permission.READ\_EXTERNAL\_STORAGE” />

1. Chạy chương trình và mở DDMS để kiểm tra các tập tin đã lưu.
   1. Tạo Shared Preference Change Listeners

Đề bài:

* Tạo giao diện cấu hình màu nền cho ứng dụng bằng cách tạo Shared Preference Change Listeners. Với giao diện như sau:



*Hình 6.1. Giao diện cấu hình màu nền cho ứng dụng.*

Mục tiêu:

* Hiểu và làm được Shared Preference Change Listeners.
* Nắm được mục đích của Shared Preference Change Listeners.
* Có thể ứng dụng để tạo Interface truyền và nhận dữ liệu giữa các lớp, đối tượng với nhau.

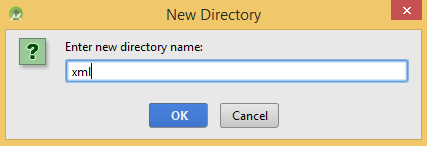
Gợi ý thực hiện:

* Từ một Project Android bất kỳ (nên sử dụng Project Android ở những bài tập trước) chúng ta tạo thêm một Layout PreferenceScreen.
* Tạo giao diện cấu hình màu nền theo yêu cầu đề bài với Layout PreferenceScreen vừa tạo.
* Tạo Activity cho Layout PreferenceScreen vừa tạo kế thừa từ PreferenceActivity.
* Cấu hình trong tập tin AndoidManifest.

Hướng dẫn chi tiết:

1. Tạo Layout PreferenceScreen

* Click chuột phải vào thư mục res → New → Directory. Đặt tên cho thư mục là “xml”.



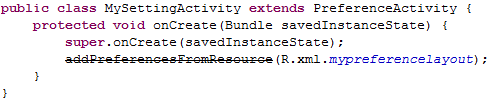
*Hình 6.2. Hướng dẫn tạo Layout PreferenceScreen.*

* Tạo tập tin mypreferencelayout.xml cho thư mục “xml” có nội dung như sau:

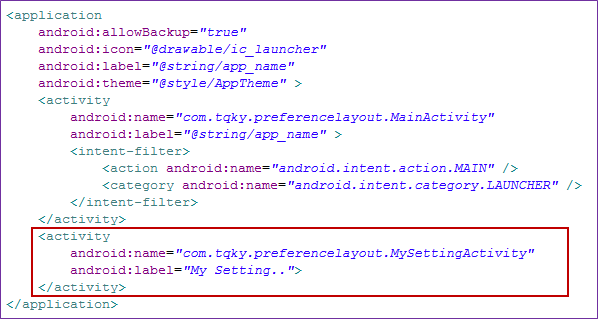


1. Tạo Activity kế thừa PreferenceActivity:

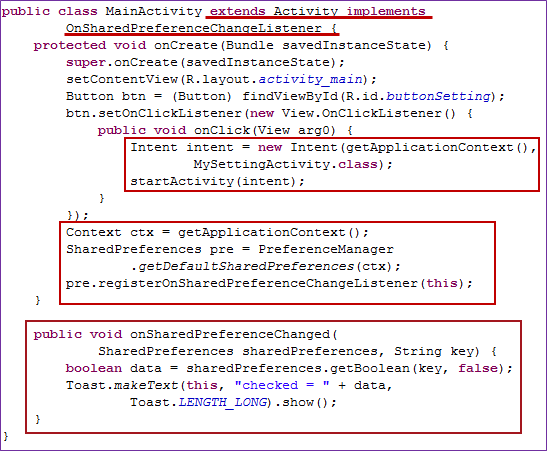
* Tạo tập tin class tên MySettingActivity có code như sau:



* Khai báo trong AndroidManifest:



1. Tạo SharedPreferences và lắng nghe dữ liệu trả về từ SharedPreferences thông qua Interface “OnSharedPreferenceChangeListener”:



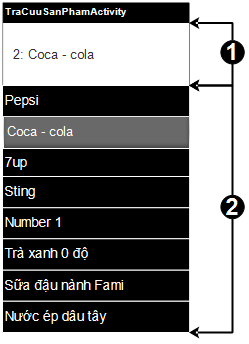
1. Chạy chương trình và xem kết quả.
2. CÁC ĐIỀU KHIỂN HIỂN THỊ DANH SÁCH

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Biết được các loại Adapter và cách dùng.  Biết Custom lại Adapter.  Hiểu được nguyên tắc hoạt động của Adapter khi Custom. |

* 1. Sử dụng ListView

Đề bài:

* Sử dụng ListView để tạo giao diện Tra cứu sản phẩm như sau:



* + 1. . *Sử dụng ListView với mảng dữ liệu định sẵn.*

Yêu cầu thực hiện:

* Mục số 1: 1 TextView để hiển thị vị trí và giá trị được chọn – Khi chọn một dòng trên ListView, chương trình sẽ hiển thị vị trí và giá trị của dòng đó lên TextView này.
* Mục số 2: 1 ListView hiển thị danh sách tên các sản phẩm (các mặt hàng nước giải khát) ở 4 trường hợp:
  + Mảng dữ liệu được định sẵn.
  + Mảng dữ liệu được lưu trong tập tin .xml.
  + Sử dụng ArrayList để lưu dữ liệu.
  + Sử dụng dưới dạng ListActivity (kế thừa từ lớp ListActivity).

Mục tiêu:

* Hiểu được Adapter là gì.
* Biết cách sử dụng Adapter mặc định.

Gợi ý thực hiện:

* Sử dụng 1 TextView và 1 ListView để tạo giao diện.
* Viết câu lệnh xử lý: tạo mảng để chứa dữ liệu về các loại sản phẩm (mảng định sẵn, mảng lưu trong tập tin .xml, mảng tạo bởi ArrayList) , khai báo và lấy điều khiển ListView, tạo mảng Adapter vào gán dữ liệu vào trong nó, đưa dữ liệu vào ListView, thiết lập sự kiện cho ListView khi chọn vào một dòng trên Listview.

Hướng dẫn chi tiết:

1. Đưa dữ liệu cần hiển thị lên ListView vào một mảng (mảng định sẵn, mảng lưu trong tập tin .xml, mảng tạo bởi ArrayList).

* Trường hợp 1:
  + Dựa vào kiến thức đã học ở Bài 5 – Module 1, chúng ta có thể tạo mảng kiểu String để lưu tên cách sản phẩm.
* Trường hợp 2: Lưu mảng dữ liệu trong tập tin .xml
  + Chúng ta tạo mảng dữ liệu để hiển thị trên listview thông qua một tập tin xml bằng cách tạo mới một tập tin ds\_sanpham.xml trong /res /values với nội dung sau:

<string-array name=*"sanphamarray"*>

<item >Pepsi</item>

<item >Coca - Cola</item>

<item >7up</item>

<item >Sting</item>

<item >Number 1</item>

<item >Trà xanh 0 độ</item>

<item >Sữa đậu nành Fami</item>

<item >Nước ép dâu tây</item>

</string-array>

* Trường hợp 3: Mảng được tạo bởi ArrayList.
  + Dựa vào kiến thức đã học ở Bài 6 – Module 1, chúng ta có thể tạo mảng kiểu String để lưu tên cách sản phẩm.
* Trường hợp 4: Sử dụng dưới dạng ListActivity (kế thừa từ lớp ListActivity)

1. Tạo một ListView trên giao diện.

* Kéo vào một ListView trong thẻ tab Composite và đặt tên id cho ListView là lvDSSanPham hoặc khai báo 1 cặp thẻ <listview> trong file giao diện xml:

<ListView

android:id=*"@+id/lvDSSanPham"*

android:Layout\_width=*"match\_parent"*

android:Layout\_height=*"wrap\_content"* >

</ListView>

1. Tạo một đối tượng ArrayAdapter để liên kết giữ ListView và mảng hoặc danh sách dữ liệu.

**// Gán dữ liệu vào mảng adapter**

ArrayAdapter<String> adapter ;

adapter = **new** ArrayAdapter<String>(**this**,android.R.Layout.simple\_list\_item\_1, sanphamarray);

**// Đổ dữ liệu từ adapter vào listview**

lvDSSanPham.setAdapter(adapter);

1. Thiết lập sự kiện cho ListView

lvDSSanPham.setOnItemClickListener(**new** AdapterView.OnItemClickListener() {

@Override

**public** **void** onItemClick(AdapterView<?> arg0, View arg1, **int** arg2, **long** arg3) {

**//Các câu lệnh xử lý**

}

});

* Trường hợp sử dụng dưới dạng ListActivity (kế thừa từ lớp ListActivity), chúng ta thiết lập sự kiện cho ListView như sau:

@Override

**protected** **void** onListItemClick(ListView l, View v, **int** position, **long** id) {

**super**.onListItemClick(l, v, position, id);

**//Các câu lệnh xử lý**

}

* 1. Tạo Custom ListView

Đề bài:

* Hãy tùy biến lại ListView để tạo giao diện hiển thị danh sách các sản phẩm có dạng như sau:



* + 1. . *Tạo Custom ListView*.

Mục tiêu:

* Hiểu được nguyên tắc hoạt động của Adapter.
* Biết cách Custom Adapter.

Gợi ý thực hiện:

* Sau khi đã có ListView trên giao diện, chúng ta có thể tạo Custom Layout cho ListView như sau:

1. Tạo thêm một Layout cho một item của ListView.
2. Tạo lớp Custom Adapter kế thừa từ lớp ArrayAdapter.
3. Tạo một lớp dùng để quản lý dữ liệu.
4. Hiển thị dữ liệu lên ListView.

Hướng dẫn chi tiết:

1. Tạo thêm một Layout cho một item của ListView

* Tạo thêm 1 Layout cho item của ListView và đặt tên là ds\_san\_pham.xml với cấu trúc như sau:

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:Layout\_width=*"match\_parent"*

android:Layout\_height=*"match\_parent"*

android:orientation=*"horizontal"*

android:padding=*"10dp"* >

<ImageView

**android:id=*"@+id/imgvHinhMHSP"***

android:Layout\_width=*"80dp"*

android:Layout\_height=*"80dp"*

android:contentDescription=*"@null"*

**android:src=*"@drawable/flag\_pepsi"* />**

<LinearLayout

android:Layout\_width=*"120dp"*

android:Layout\_height=*"wrap\_content"*

android:orientation=*"vertical"*

android:Layout\_marginLeft=*"20dp"*

android:padding=*"10dp"* >

<TextView

**android:id=*"@+id/tvTenSP"***

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

android:text=*"@null"*

android:textColor=*"#357B6A"*

android:textAppearance=*"?android:attr/textAppearanceMedium"*/>

<TextView

**android:id=*"@+id/tvGiaSP"***

android:Layout\_width=*"wrap\_content"*

android:Layout\_height=*"wrap\_content"*

android:text=*"@null"*

/>

</LinearLayout>

</LinearLayout>

1. Xây dựng đối tượng SanPham bằng cách tạo lớp SanPham.java

**public** **class** SanPham {

**private** String tenSP;

**private** String giaSP;

**private** **int** img\_id;

**public** String getTenSP() {

**return** tenSP;

}

**public** **void** setTenSP(String tenSP) {

**this**. tenSP = tenSP;

}

**public** String getGiaSP() {

**return** giaSP;

}

**public** **void** setGiaSP(String giaSP) {

**this**. giaSP = giaSP;

}

**public** **int** getImg\_id() {

**return** img\_id;

}

**public** **void** setImg\_id(**int** img\_id) {

**this**.img\_id = img\_id;

}

**public** SanPham(String tenSP, String giaSP, **int** img\_id)

{

**this**. tenSP = tenSP;

**this**. giaSP = giaSP;

**this**.img\_id = img\_id;

}

**public** SanPham () {

}

}

1. Tạo một lớp SanPhamLuuTru.java dùng để quản lý dữ liệu về các sản phẩm

**public** **class** SanPhamLuuTru {

**//Tạo mảng ds\_sanpham kiểu String để lưu tên các sản phẩm**

**public** **static** String[] *ds\_sanpham* = **new** String[] { "Pepsi", "7up", "Coca - Cola", "Sting", "Number 1", "Trà xanh 0 độ", "Sữa đậu nành Fami", "Nước ép dâu tây" };

**//Tạo mảng ds\_gia\_sanpham kiểu String để lưu giá tiền của các sản phẩm**

**public** **static** String[] *ds\_gia\_sanpham* = **new** String[] { "165000/ thùng", "160000/ thùng", "177000/ thùng", "180000/ thùng", "160000/ thùng", "145000/ thùng", "150000/ thùng"};

**//Tạo mảng img\_ids kiểu String để lưu id hình của các sản phẩm**

**public** **static** **int**[] *img\_ids* = **new** **int**[] {

R.drawable.flag\_pepsi,R.drawable.flag\_7up,

R.drawable.flag\_coca, R.drawable.flag\_sting,

R.drawable.flag\_number1, R.drawable.flag\_traxanh0do,

R.drawable.flag\_suafami, R.drawable.flag\_nuocepdau };

**private** **static** ArrayList<SanPham> *dssanpham*;

**public** **static** ArrayList<SanPham> getDSSanPham() {

**if** (*dssanpham* == **null**) {

SanPhamLuuTru sanphamLuuTru = **new** SanPhamLuuTru();

sanphamLuuTru.initDSSanPham();

}

**return** *dssanpham*;

}

**private** **void** initDSSanPham() {

dssanpham = **new** ArrayList<SanPham>();

**for** (**int** i = 0; i < ds\_sanpham.length; i++) {

SanPham sp = **new** SanPham(ds\_sanpham[i], ds\_gia\_sanpham[i], img\_ids[i]);

dssanpham.add(sp);

}

}

1. Tạo lớp SanPhamAdapter.java kế thừa từ lớp ArrayAdapter

**public** **class** SanPhamAdapter **extends** ArrayAdapter<SanPham> {

Context context;

**int** resId;

ArrayList<SanPham> dssanpham;

**//Lấy về những giá trị được truyền từ MainActivity**

**public** SanPhamAdapter(Context context, **int** resId, ArrayList<SanPham> dssanpham){

**super**(context, resId, dssanpham);

**this**.context = context;

**this**.resId = resId;

**this**. dssanpham = dssanpham;

}

@Override

**public** View getView(**int** position, View convertView, ViewGroup parent){

**/\* position: là vị trí của sản phẩm trong danh sách**

**convertView: dùng để lấy về các control của mỗi item**

**parent: chính là datasource được truyền vào từ MainActivity \*/**

View view = **null**;

LayoutInflater inflater =

(LayoutInflater)context.getSystemService(Context.LAYOUT\_INFLATER\_SERVICE);

view = inflater.inflate(resId, **null**);

**//Lấy các điều khiển**

TextView tvTenSP = (TextView) view.findViewById(R.id. tvTenSP);

TextView tvGiaSP = (TextView) view.findViewById(R.id. tvGiaSP);

ImageView imgvHinhMHSP = (ImageView) view.findViewById(R.id. imgvHinhMHSP);

**//Lấy sản phẩm ở vị trí thứ position**

SanPham sp = dssanpham.get(position);

**if** (sp != **null**) {

**//Đưa thông tin lên TextView và hình lên ImageView**

tvTenSP.setText(sp.getTenSP());

tvGiaSP.setText(sp.getGiaSP());

imgvHinhMHSP.setImageResource(sp.getImg\_id());

}

**/\* View là dạng tham chiếu đối tượng, nên mọi sự thay đổi của các đối tượng bên trong View thì nó cũng biết sự thay đổi đó trả về view này, tức là trả luôn về các thông số mới mà ta vừa thay đổi \*/**

**return** view;

}

}

1. Tạo và setAdapter cho ListView trong MainActivity

ListView lv\_dsSanPham = (ListView)findViewById(R.id.lvDSSanPham);

ArrayList<SanPham> dssanpham = SanPhamLuuTru.getDSSanPham();

**SanPhamAdapter sanphamAdapter = new SanPhamAdapter(this,R.Layout.Layout\_cua\_tui,dssanpham);**

**lv\_dsSanPham.setAdapter(sanphamAdapter);**

* Ngoài ra, còn có một cách khác để tạo Custom Layout cho ListView đó là sử dụng ViewHolder. Tuy hơi phức tạp hơn nhưng sử dụng ViewHolder là cách điều chỉnh Layout hay và tối ưu về mặt hiệu năng. Bạn có thể tìm hiểu về kỹ thuật này trên trang: *http://developer.android.com*.
  1. Tạo Spinner và AutoComplexTextView trong Android

Đề bài:

* Tạo giao diện hiển thị danh sách các sản phẩm có dạng như sau:

|  |
| --- |
| * + 1. *. Giao diện tra cứu sản phẩm với AutoCompleteTexView* |
| * + 1. *. Giao diện tra cứu sản phẩm với Spinner* |

* Tạo thêm Spinner để lọc theo thể loại sản phẩm và AutoComlexTextView tìm kiếm sản phẩm. Khi chọn vào 1 item Spinner thì dữ liệu trên ListView thay đổi phù hợp với item đang chọn. Còn khi nhập chữ trên AutoComplexTextView xong, nhấn tìm thì hiển thị sản phẩm đó lên ListView.

Mục tiêu:

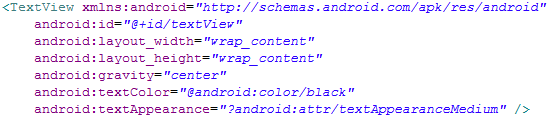
* Biết cách dụng Spinner.
* Hiểu được cách tự hoàn thành đoạn text nhập của AutoComplexTextView và biết cách dùng.

Gợi ý thực hiện:

* Kéo thả Spinner và AutoComplexTextView vào Layout.
* Tạo ArrayAdapter cho Spinner và AutoComplexTextView(dữ liệu là tất cả các tên sản phẩm). Sau đó, chúng ta setAdapter cho 2 đối tượng này.
* Ở Spinner truy xuất đến phương thức “setOnItemSelectedListener” để bắt lại vị trí mà người dùng đang chọn để hiển thị danh sách sản phẩm lên ListView cho đúng.

Hướng dẫn chi tiết:

1. Thiết kế giao diện, tạo đối tượng Sản phẩm, tạo lớp lưu trữ sản phẩm, lớp SanPhamAdapter giống như bài tập 7.2
2. Thêm phần thiết kế giao diện cho dòng giao diện của Spinner như sau:



1. Viết các câu lệnh xử lý trong lớp MainActivity tương tự như ở bài tập 7.2 nhưng thêm những câu lệnh xử lý cho Spinner và AutoComplexTextView như sau:

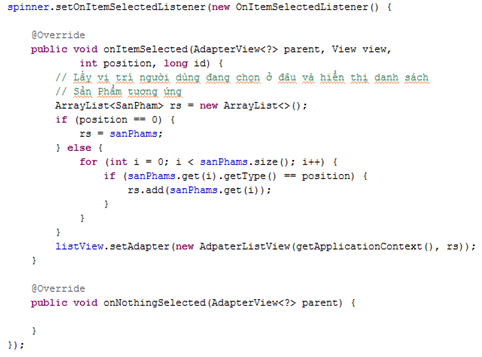
* Lấy diều khiển cho Spinner và AutoComplexTextView:



* Thiết lập chế độ hiển thị cho AutoComplexTextView và Spinner:



* Xử lý khi người dùng chọn một dòng trên Spinner:



1. XÂY DỰNG GIAO DIỆN VỚI FRAGMENT - VẤN ĐỀ ACTIVITY VÀ FRAGMENT

|  |  |
| --- | --- |
| **🖰** | Mục tiêu:  Hiểu và tạo được Fragment.  Áp dụng được các kiến thức và kỹ năng về Fragment vào việc xây dựng các ứng dụng Android. |

* 1. Tạo và sử dụng Fragment 1

Đề bài:

* Tạo 2 Fragment:
  + Fragment 1 được dùng khi màn hình thiết bị nằm ngang.
  + Fragment 2 được dùng khi màn hình thiết bị nằm dọc.
* Yêu cầu:
  + Khi xoay màn hình thiết bị nằm ngang sẽ hiển thị giao diện như hình 1, còn khi xoay màn hình thiết bị nằm dọc sẽ hiển thị giao diện như hình 2.

|  |  |
| --- | --- |
| * + 1. *. Màn hình khi thiết bị nằm ngang.* | * + 1. . *Màn hình khi thiết bị nằm dọc.* |

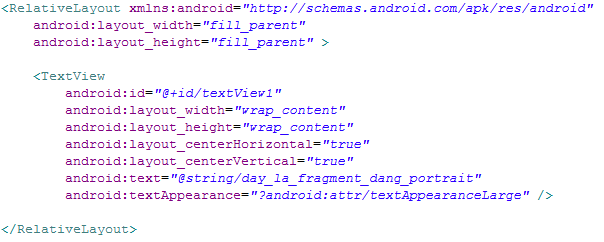
Mục tiêu:

* Hiểu được khái niệm Fragment.
* Biết cách tạo và dùng Fragment.
* Ứng dụng của Fragment.

Gợi ý thực hiện:

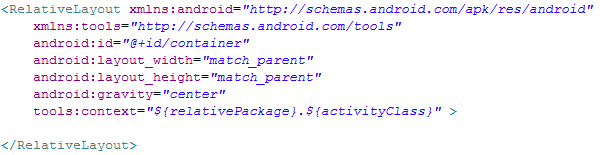
* Tạo 1 tập tin .xml là giao diện của Fragment để trong thư 2 thư mục layout-port và layout-land.
* Trong tập tin .xml main tạo 1 ViewGroup để chứa giao diện Fragment.
* Tạo lớp kế thừa Fragment và cài đặt giao diện là tập tin .xml vừa tạo ở trên.
* Trong lớp main (.xml), để Fragment lên ViewGroup để chứa giao diện Fragment.

Hướng dẫn chi tiết:

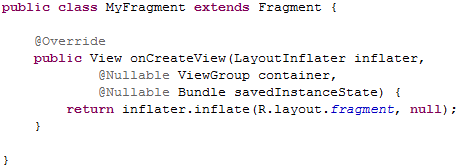
1. Tạo thư mục “layout-land” trong thư mục res.
2. Tạo tập tin fragment.xml (Layout) trong thư mục “layout” và “layout-land”: 

* *Chú ý:* trong thư mục “Layout” thuộc tính text của TextView là “đây là fragment dạng Portrait”, còn trong thư mục “Layout-land” thuộc tính text của TextView là “đây là fragment dạng Landscape”.

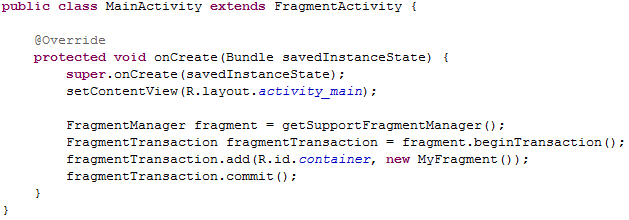
1. : Trong tập tin fragment.xml, ta thêm thuộc tính id như sau:



1. Tạo lớp “MyFragment.java” kết thừa Fragment:



1. Lớp “MainActivity.java”:



1. Chạy và xem kết quả.
   1. Tạo và sử dụng Fragment 2

Đề bài:

* Sử dụng Fragment để xây dựng chương trình với yêu cầu như sau:
  + Khi xoay màn hình thiết bị nằm ngang sẽ hiển thị giao diện như hình 8.3. Khi người dùng nhấn chọn vào 1 dòng trên ListView sẽ hiển thị nội dung của dòng đó như hình 8.4 (ListView và phần hiển thị nội dung cùng nằm trên 1 màn hình).

|  |  |
| --- | --- |
| * + 1. *. Giao diện khi thiết bị nằm ngang* | * + 1. . *Giao diện khi người dùng chọn 1 dòng trên ListView.* |

* Còn khi xoay màn hình thiết bị nằm dọc sẽ hiển thị giao diện như hình 8.5. Khi người dùng nhấn chọn vào 1 dòng trên ListView sẽ hiển thị nội dung của dòng đó ở màn hình khác như hình 8.6.

|  |  |
| --- | --- |
| * + 1. *. Giao diện khi thiết bị nằm dọc 1.* | * + 1. *. Giao diện khi thiết bị nằm dọc 2.* |

Mục tiêu:

* Biết cách gửi dữ liệu giữa Activity và Fragment thông qua Interface.

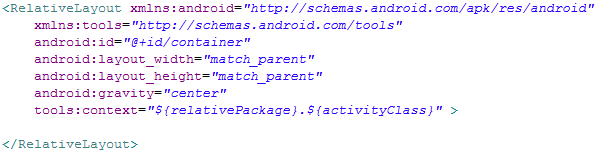
Gợi ý thực hiện:

* Tạo 2 Layout cho màn hình nằm ngang và đứng. Màn hình nằm ngang thì có 1 ListView bên trái và Fragment bên phải. Còn màn hình đứng thì 2 Layout: 1 Layout chứa ListView và 1 Layout chứa Fragment.
* Tạo Interface để truyền nhận dữ liệu giữa Activity và Fragment .

Hướng dẫn chi tiết:

1. Thiết kế giao diện:

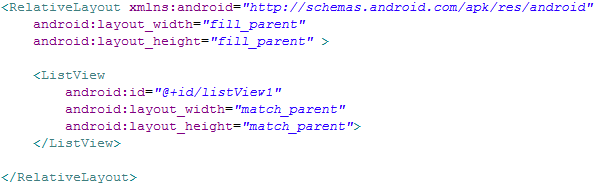
* Thiết kế Layout “activity\_main.xml”:



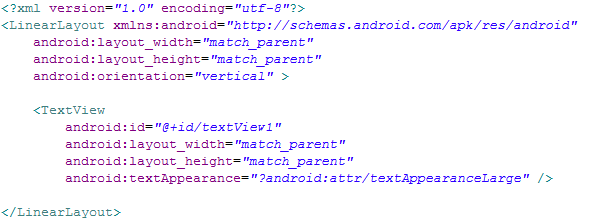
* Thiết kế Layout “my\_fragment” trong thư mụ res/Layout-land:



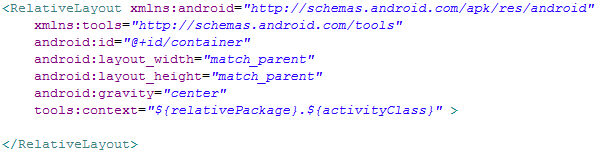
* Thiết kết Layout “my\_fragment” trong thư mục res/Layout-port:



* Thiết kế Layout “my\_fragment\_detail.xml” trong thư mục res/layout:



* Thiết kết Layout “activity\_main.xml” trong thư mục res/layout:



* Thiết kế Layout “item\_listview.xml” trong thư mục res/layout:

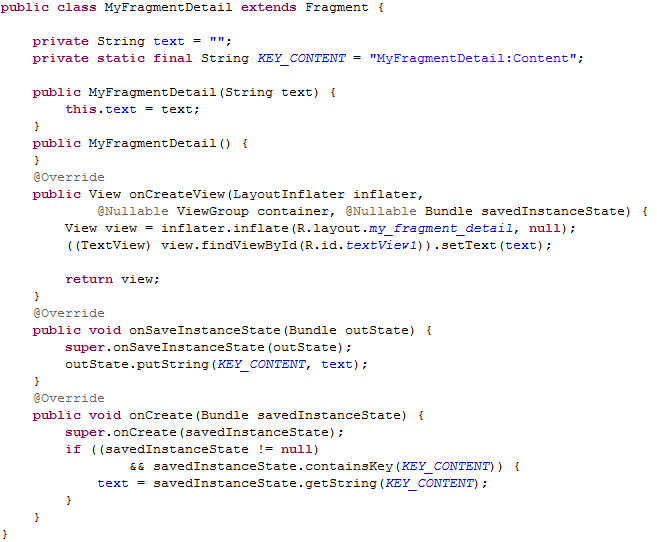


1. Viết câu lệnh trong tập tin .java:

* Viết lệnh trong lớp “SanPham.java”:



* Viết lệnh trong lớp “MyFragmentDetail.java”:



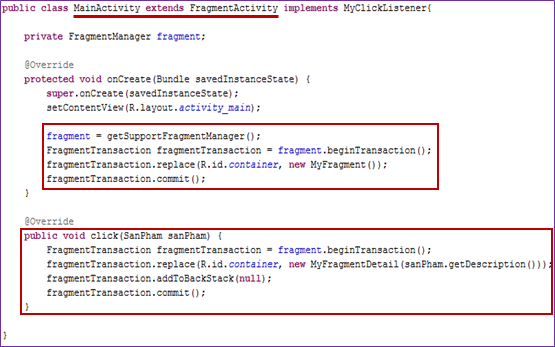
* Viết các câu lệnh trong lớp “MyFragment.java”:

|  |
| --- |
|  |

* Viết các câu lệnh trong lớp “AdapterListView.java”:

|  |
| --- |
|  |

* Viết các câu lệnh trong lớp “MainActivity.java”:



1. Chạy và xem kết quả.