Giao diện người dùng và xử lý sự kiện

GV: TRƯƠNG BÁ THÁI

Email:truongbathai@tdc.edu.vn

ĐT: 0932.577.765

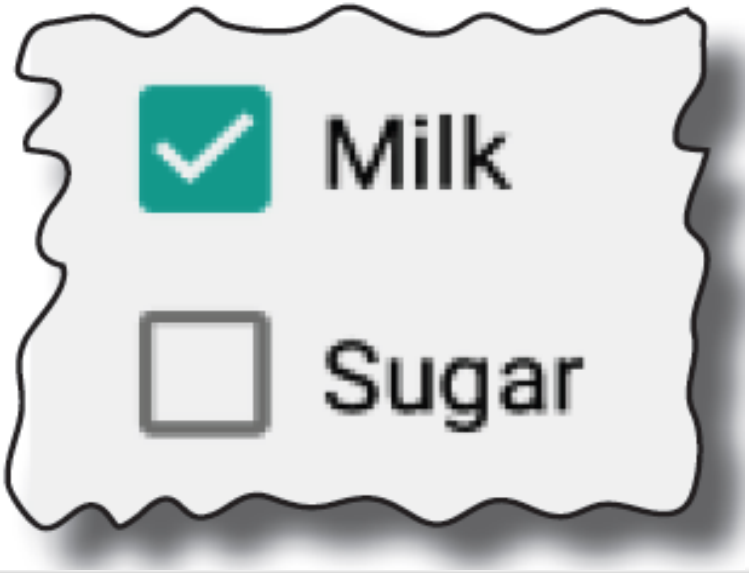
MỤC TIÊU THỰC HIỆN

- Xây dựng được giao diện cho ứng dụng
- Xử lý được các sự kiện cho ứng dụng
- Sử dụng thành thạo IDE Android Studio để viết chương trình Android
- Hình thành thói quen thiết kế chương trình theo tiếp cận Top-Down

Checkbox

Checkbox

- **CheckBox** là thành phần thể hiện trạng thái chọn (checked) hoặc không chọn (unchecked) **CheckBox** thường dùng khi người dùng có nhiều lựa chọn và được phép chọn một hoặc nhiều lựa chọn cùng lúc



Một vài phương thức thường sử dụng

Phương thức	Ý nghĩa
public boolean **isChecked**()	True nếu CheckBox là checked, ngược lại false
public void **setChecked**(boolean status)	Thay đổi trạng thái của CheckBox

Thuộc tính thường dùng của CheckBox

- **android:id:** Là thuộc tính duy nhất của **CheckBox**
- **android:checked:** Checked là thuộc tính của **CheckBox** dùng để set trạng thái của CheckBox. Giá trị là true hoặc false**.** Chúng ta cũng có thể set trạng thái của CheckBox bên Java Code bằng cách dùng phương thức **setChecked**(boolean status)
- **android:gravity:** Thuộc tính này thường sử dụng để canh nội dung trong **CheckBox**: **left, right, center, top, bottom, center_vertical, center_horizontal**

Thuộc tính thường dùng của CheckBox

- **android:text:** Thuộc tính text dùng hiển thị nội dung trong một **CheckBox**.
- **android:textSize:** Thuộc tính textSize xác định kích thước nội dung văn bản của **CheckBox**.
- **android:textStyle:** Thuộc tính xác định loại văn bản của **CheckBox**, thông thường có các loại văn bản:**bold, italic và normal.**

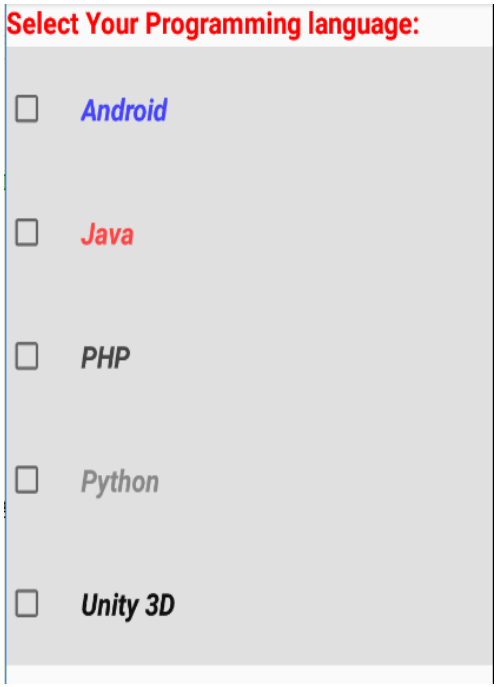
Thuộc tính thường dùng của CheckBox

- **android:background**: Thuộc tính này xác định màu nền cho CheckBox.
- **android:padding**:Thuộc tính này xác định khoảng cách từ đường viền của CheckBox với nội dung nó chứa:left, right, top or bottom.

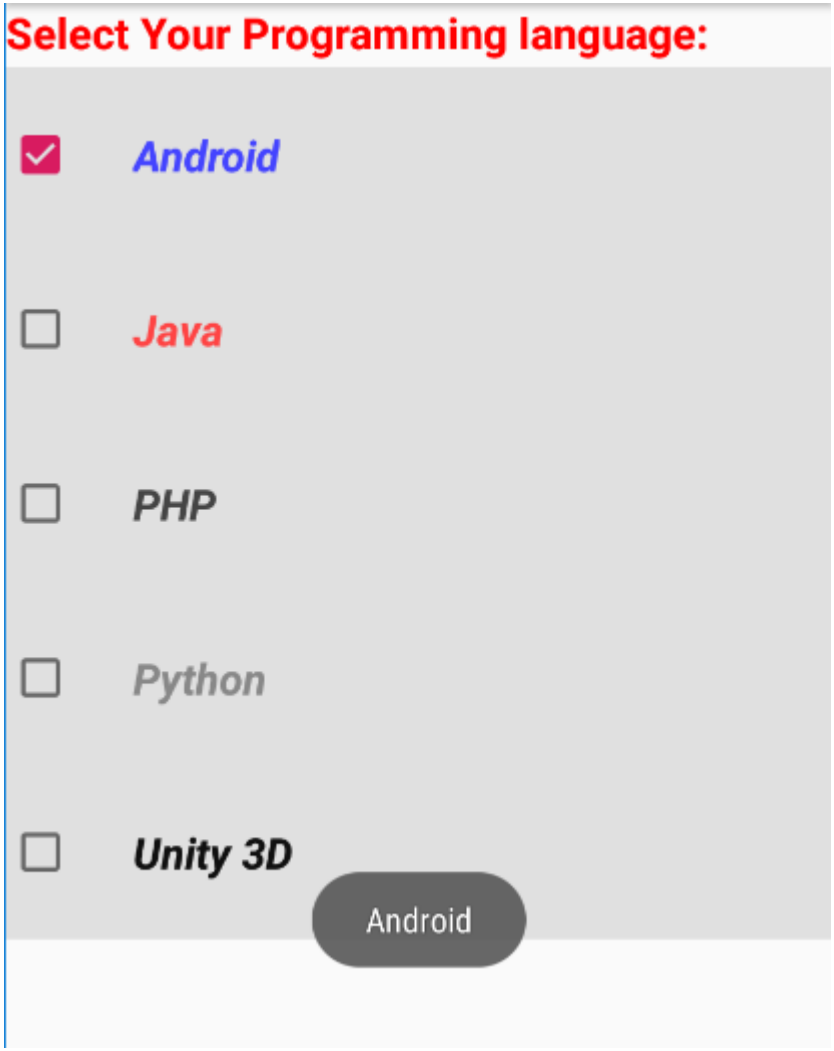
```
<CheckBox
    android:id="@+id/chksimpleCheckBox"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Attribute Of Check Box"
    android:textColor="#44f"
    android:textSize="20sp"
    android:textStyle="bold|italic"
    android:checked="true"
    android:padding="30dp"/>
```

Ví dụ

- Chọn 1 ngôn ngữ lập trình trên các đối tượng **CheckBox**. Khi người sử dụng click lên **CheckBox** sẽ hiển thị ngôn ngữ vừa chọn, thông qua việc sử dụng đối tượng **TOAST**.




Select Your Programming language:
Android
Java
PHP
Python
Unity 3D


Select Your Programming language:
Android
Java
PHP
Python
Unity 3D
Android

Bước 1

- Tạo một project tên là DemoCheckBox: **File → New → Android Application Project** điền các thông tin → **Next →Finish**

Bước 2:

- Vào thư mục **res/values** bổ sung **string.xml**

```
<resources>
    <string name="app_name">CheckBoxExample</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
    <string name="android">Android</string>
    <string name="java">Java</string>
    <string name="php">PHP</string>
    <string name="python" >Python</string>
    <string name="unity">Unity 3D</string>
</resources>
```

Bước 3:

- Mở **res → layout → xml (hoặc) activity_main.xml**

Bước 4:

- Mở **app → src -> MainActivity.java** và thêm code. Khi click vào **CheckBox** sẽ lấy các giá trị của **CheckBox**, sau dùng đối tượng **TOAST** để hiển thị

```
public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
    //Khai báo các widget
    CheckBox android, java, python, php, unity3D;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }
```

```
private void setControl() {
    android = (CheckBox) findViewById(R.id.chkAndroid);
    java = (CheckBox) findViewById(R.id.chkJava);
    python = (CheckBox) findViewById(R.id.chkPython);
    php = (CheckBox) findViewById(R.id.chkPHP);
    unity3D = (CheckBox) findViewById(R.id.chkUnity);
}

private void setEvent() {
    android.setOnClickListener(this);
    java.setOnClickListener(this);
    python.setOnClickListener(this);
    php.setOnClickListener(this);
    unity3D.setOnClickListener(this);
}
```

```
@Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.chkAndroid:
                if (android.isChecked())
                    Toast.makeText(getApplicationContext(), "Android",
Toast.LENGTH_LONG).show();
                break;
            case R.id.chkJava:
                if (java.isChecked())
                    Toast.makeText(getApplicationContext(), "Java",
Toast.LENGTH_LONG).show();
                break;
        }
    }
}
```

RadioButton

RadioButton

- Radiobutton thường được đưa ra 2 hoặc nhiều hơn hai phần tử trong đó người dùng chỉ được chọn một phần tử, để làm được như vậy chúng ta cần nhóm chúng vào một nhóm đó chính là GroupRadiobutton để khi người dùng chọn thì chỉ chọn được duy nhất.

Một vài phương thức thường sử dụng:

Phương thức	Ý nghĩa
public boolean **isChecked**()	True nếu CheckBox là checked, ngược lại false
public void **setChecked**(boolean status)	Thay đổi trạng thái của CheckBox

Thuộc tính thường dùng của RadioButton

- **android:id:** Là thuộc tính duy nhất của RadioButton
- **android:checked**: checked là thuộc tính của RadioButton dùng để set trạng thái của CheckBox. Chúng ta cũng có thể set trạng thái của RadioButton bên Java Code bằng cách dùng phương thức **setChecked**(boolean status).

- **android:gravity:** Thuộc tính này thường sử dụng để canh nội dung trong RadioButton: left, right, center, top, bottom, center_vertical, center_horizontal.
- **android:text:** thuộc tính text dùng hiển thị nội dung trong một RadioButton. Chúng ta có thể set thuộc tính này trong tập tin xml hoặc java code
- **android:textColor:**Thuộc tính này dùng xác định màu chữ, dạng màu chữ: “#argb”, "#rgb”, “#rrggbb”, hoặc “#aarrggbb”.

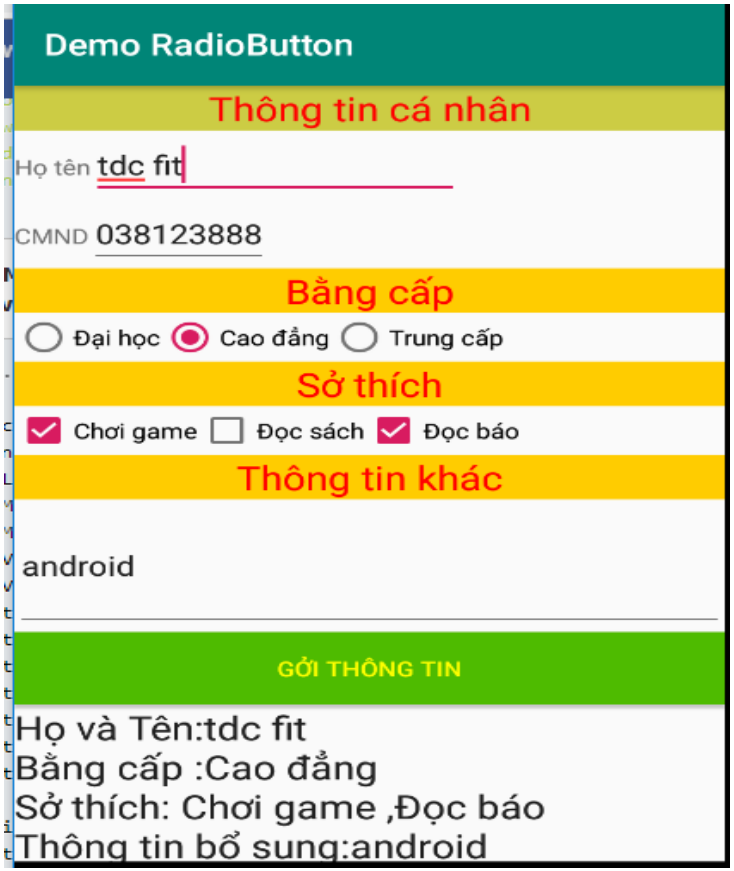
- **android:textSize**: Thuộc tính textSize xác định kích thước nội dung văn bản của RadioButton. Chúng ta có thể đặt kích thước văn bản theo: sp (scale independent pixel) hoặc dp (density pixel).
- **android:textStyle**:Thuộc tính xác định loại văn bản của RadioButton, thông thường có các loại văn bản:bold, italic và normal. Nếu chúng ta muốn sử nhiều hơn một loại văn bản thì phải thêm phép toán hoặc "|" vào giữa các loại văn bản.

- **android:background:** Thuộc tính này xác định màu nền cho RadioButton.
- **android:padding:** Thuộc tính này xác định khoảng cách từ đường viền của RadioButton với nội dung nó chứa: left, right, top or bottom.

```
<RadioButton
    android:id="@+id/rdisimpleRadioButton"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text ="TDC IT"
    android:textSize="40sp"
    android:textColor="#f00"
    android:textStyle="bold|italic"
    android:checked="true"
    android:padding="30dp"/>
```

Ví dụ:

- Xây dựng ứng dụng khảo sát đơn giản sau



Yêu cầu

- Tên người không được để trống và phải có ít nhất 3 ký tự
- Chứng minh nhân dân chỉ được nhập kiểu số và phải có đúng 9 chữ số
- Bằng cấp mặc định sẽ chọn là Đại học

- Sở thích phải chọn ít nhất 1 chọn lựa
- Thông tin bổ sung có thể để trống
- Khi bấm gửi thông tin, chương trình sẽ hiển thị toàn bộ thông tin cá nhân cho người sử dụng biết thông qua TextView:

Bước 1:

- Tạo một project tên là DemoRadioButton: **File → New →Android Application Project** điền các thông tin → **Next →Finish**

Bước 2:

- Vào thư mục **res/values** bổ sung **string.xml**

```
<resources>
    <string name="app_name">Demo RadioButton</string>
    <string name="action_settings">Settings</string>
    <string name="information">Thông tin cá nhân</string>
    <string name="firstname">Họ tên</string>
    <string name="id">CMND</string>
    <string name="degree">Bằng cấp</string>
    <string name="university">Đại học</string>
    <string name="college">Cao đẳng</string>
    <string name="college1">Trung cấp</string>
    <string name="favorate">Sở thích</string>
    <string name="game">Chơi game</string>
    <string name="book">Đọc sách</string>
    <string name="newspaper">Đọc báo</string>
    <string name="addinformation">Thông tin khác</string>
    <string name="btninsert">Gởi thông tin</string>
</resources>
```

Bước 3:

- Mở **res→layout→xml**

Bước 4:

- Mở **app →src →MainActivity.java** và thêm code. Khi click vào Button **Gởi thông tin**, các thông tin được hiển thị qua **TextView.**

```
public class MainActivity extends AppCompatActivity {
    //Khai báo các widget
    EditText edtFirstName, edtID, edtInformation;
    CheckBox chkGame, chkBook, chkNewspaper;
    TextView txtResult;
    Button btnInsert;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }
```

```
private void setControl() {
    edtFirstName  = (EditText) findViewById(R.id.edtFirstName);
    edtID         = (EditText) findViewById(R.id.edtID);
    edtInformation = (EditText) findViewById(R.id.edtAddInformation);
    chkGame       = (CheckBox) findViewById(R.id.chkGame);
    chkBook       = (CheckBox) findViewById(R.id.chkBook);
    chkNewspaper  = (CheckBox) findViewById(R.id.chkNewspaper);
    btnInsert     = (Button)   findViewById(R.id.btninsert);
    txtResult     = (TextView) findViewById(R.id.txtResult);
}
```

```
private void setEvent() {
    btnInsert.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub
            showInformation();
        }
    });
}
```

```
// Lấy thông tin người sử dụng nhập vào
private void showInformation(){
    try{
        String message ="";
        //Kiểm tra họ và tên
        String firstname  =  edtFirstName.getText().toString().trim();
        if(firstname.length()<=3){
            edtFirstName.selectAll();
            edtFirstName.requestFocus();
            Toast.makeText(getApplication(), "Họ và tên không nhỏ hơn 3 ký tự",
Toast.LENGTH_LONG).show();
            return;
        }
        message +="Họ và Tên:" +firstname +"\n";
```

```
String strID  =  edtID.getText().toString().trim();
        if(strID.length()!=9){
            edtID.selectAll();
            edtID.requestFocus();
            Toast.makeText(getApplication(), "Số chứng minh nhân dân phải 9
số", Toast.LENGTH_LONG).show();
            return;
        }
        String degree;
        RadioGroup rdigroupDegree  =  (RadioGroup)
findViewById(R.id.rdiGroupDegree);
        int id =  rdigroupDegree.getCheckedRadioButtonId();
        if(id==-1){
            Toast.makeText(getApplication(), "Xin vui lòng chọn bằng cấp",
Toast.LENGTH_LONG).show();
            return;
        }
```

```
RadioButton radDegree= (RadioButton) findViewById(id);
        degree =  radDegree.getText()+"";
        message += "Bằng cấp :"+ degree +"\n";
        String favorate="";
        if(chkGame.isChecked())
           favorate+= chkGame.getText()+ " ,";
        if(chkBook.isChecked())
           favorate+= chkBook.getText()+ " ,";
        if(chkNewspaper.isChecked())
           favorate+=chkNewspaper.getText()+ " ";
        if(favorate.trim().length()==0){
           Toast.makeText(getApplication(), "Xin vui một sở thích",
Toast.LENGTH_LONG).show();
           return;
        }
        else
           message+="Sở thích: "+ favorate +"\n";
```

```
if(edtInformation.getText().toString().trim().length()!=0)
            message += "Thông tin bổ sung:"
+edtInformation.getText().toString().trim();
            txtResult.setText(message);
        }
        catch(Exception e){
            Log.d("1", e.getMessage());
        }
    }
}
```

Image

ImageView

- **ImageView** là một view sử dụng để hiển thị ảnh, mà nguồn ảnh có thể là một file ảnh trên ứng dụng, trên thiết bị hoặc từ **URL**.

Thuộc tính thường dùng của ImageView

- **android:id:** Là thuộc tính duy nhất của **ImageView**
- **android:src:** là thuộc tính chứa hình ảnh cần hiển thị
- **android:background:** Thuộc tính này xác định màu nền cho **ImageView**
- **android:padding:** Thuộc tính này xác định khoảng cách từ đường viền của **ImageView** với nội dung nó chứa: **left, right, top or bottom.**

- **paddingRight**: thiết khoảng cách bên phải của ImageView.
- **paddingLeft**: thiết khoảng cách bên trái của ImageView.
- **paddingTop**: thiết khoảng cách phía trên của ImageView.
- **paddingBottom**: thiết khoảng cách phía bên dưới của ImageView.
- **padding**: thiết khoảng cách tất cả 4 phía của ImageView.

- **android:scaleType:** ScaleType là thuộc tính xác định các thức mà hình ảnh sẽ được scale như thế nào để phù hợp với view của chúng ta. ImageView có thể hiển thị image theo nhiều cách khác nhau phụ thuộc vào các giá trị của thuộc tính scaleType. Giá trị của scaleType :**fit_xy, center_crop, fitStart v.v .**

```
<ImageView
    android:id="@+id/simpleImageView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#000"
    android:src="@drawable/tdc"
    android:padding="30dp"
    android:scaleType="fitXY"
    />
```

Ví dụ:

- Xây dựng ứng dụng hiển thị hình ảnh lên một **ImageView**. Để xem hình người sử dụng click vào 2 **Button** trên màn hình

Bước 1:

- Tạo một project tên là DemoImages: **File → New → Android Application Project** điền các thông tin → **Next →Finish.**

Bước 2:

- Vào thư mục **res/values** bổ sung **string.xml**

```
<resources>
    <string name="app_name">DemoImage</string>
    <string name="btndemo1">Xem hình 1</string>
    <string name="btndemo2">Xem hình 2</string>
</resources>
```

Bước 3:

- Mở **res → layout → xml (hoặc) activity_main.xml**

Bước 4:

- Mở **app → src -> MainActivity.java** và click vào xem hình 1 thì sẽ hiển thị hình 1. Click vào xem hình thì sẽ hiển thị hình 2.

```
public class MainActivity extends AppCompatActivity {
    //Khai báo các widget
    ImageView imgDemo;
    Button btnDemo1, btnDemo2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }
```

```
private void setControl() {
    imgDemo    = (ImageView) findViewById(R.id.imgDemo);
    btnDemo1= (Button)   findViewById(R.id.btnDemo1);
    btnDemo2= (Button)findViewById(R.id.btnDemo2);
}
```

```
    private void setEvent() {
        btnDemo1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                imgDemo.setImageResource(R.drawable.tdc);
            }
        });

        btnDemo2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                imgDemo.setImageResource(R.drawable.fit);
            }
        });
    }
}
```

ToggleButton

ToggleButton

- **ToggleButton** có thể nói là dạng đặc biệt của button, nó có 2 trạng thái cơ bản là **check** và **not check**. Khi ở trạng thái check chúng ta click nó sẽ chuyển sang trạng thái not check và ngược lại

Thuộc tính quan trọng:

- **textOn**: trạng thái nút đang bật
- **textOff**: trạng thái nút đang tắt

Một vài phương thức thường sử dụng

Phương thức	Ý nghĩa
CharSequence **getTextOff**()	Trả về một chuỗi khi giá trị trạng thái button là unchecked
CharSequence **getTextOn**()	Trả về một chuỗi giá trị khi trạng thái button là checked
public void **setChecked**(boolean status)	Thay đổi trạng thái của ToggleButton
public Boolean **isChecked**()	Kiểm tra trạng thái của ToggleButon, có 2 giá trị true hoặc false

Thuộc tính thường dùng của ToggleButton

- **android:id:** Là thuộc tính duy nhất của **ToggleButton**
- **android:checked: Checked là thuộc tính của ToggleButton dùng để set trạng** thái của **ToggleButton**. Chúng ta cũng có thể set trạng thái của **ToggleButton** bên Java Code bằng cách dùng phương thức setChecked(boolean status)

- **android:textOn Và android:textOff:** thuộc tính textOn được sử dụng để hiển thị câu thông báo khi ở trạng thái checked. Chúng ta có thể set textOn trong XML, hoặc trong Java Class
- **android:textColor:** Thuộc tính này dùng xác định màu chữ, dạng màu chữ: “#argb”, "#rgb”, “#rrggbb”, hoặc “#aarrggbb”

- **android:textSize:** Thuộc tính textSize xác định kích thước nội dung văn bản của **ToggleButton**. Chúng ta có thể đặt kích thước văn bản theo: sp(scale independent pixel) hoặc dp(density pixel)
- **android:textStyle:** Thuộc tính xác định loại văn bản của **ToggleButton**, thông thường có các loại văn bản:**bold, italic và normal.** Nếu chúng ta muốn sử nhiều hơn một loại văn bản thì phải thêm phép toán hoặc "|"

- **android:background:** Thuộc tính này xác định màu nền cho **ToggleButton**
- **android:padding:** Thuộc tính này xác định khoảng cách từ đường viền của **ToggleButton** với nội dung nó chứa: **left, right, top or bottom.**
- **android:drawableBottom, android:drawableTop,android: drawableRight và android:drawableLeft:** Các thuộc tính này hiển thị hình trong res/drawable theo các hướng: **bottom, top, right và left** nội dung văn bản của **ToggleButton**.

Ví dụ:

- Trong ví dụ này chúng ta sẽ làm app có một **ToggleButton**. Khi người sử dụng click **ToggleButton** sẽ hiện thị trạng thái hiện tại của **ToggleButton** qua đối tượng TOAST

Bước 1:

- Tạo một project tên là DemoToggleButton**. File → New → Android Application Project** điền các thông tin → **Next →Finish.**

Bước 2:

- Mở **res → layout → xml (hoặc) activity_main.xml**

Bước 3:

- Mở **app → src -> MainActivity.java** và thêm code. Khi click vào **ToogleButton** sẽ lấy chuỗi trạng thái của nó bằng cách dùng phương thức **getText()**, sau đó dùng đối tượng **TOAST** để hiển thị.

```
public class MainActivity extends AppCompatActivity {
    //Khai báo các widget
    ToggleButton tglbtnSimple;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }
    private void setControl() {
        tglbtnSimple = (ToggleButton) findViewById(R.id.simpleToggleButton);
    }
```

```
    private void setEvent() {
        tglbtnSimple.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                Toast.makeText(getApplication(), "Trạng thái đã chọn " +
tglbtnSimple.getText(), Toast.LENGTH_LONG).show();
            }
        });
    }
}
```

Switch

Switch

- Switch: đối tượng nút bấm hai trạng thái “bật” và “tắt”, có thể thao tác bằng cách trượt ngón tay trên đối tượng.

Một vài phương thức thường sử dụng

Phương thức	**Ý nghĩa**
public boolean **isChecked**()	Trạng thái hiện tại của Switch
public void **setChecked**(boolean status)	Thay đổi trạng thái của Switch

Thuộc tính thường dùng của Switch

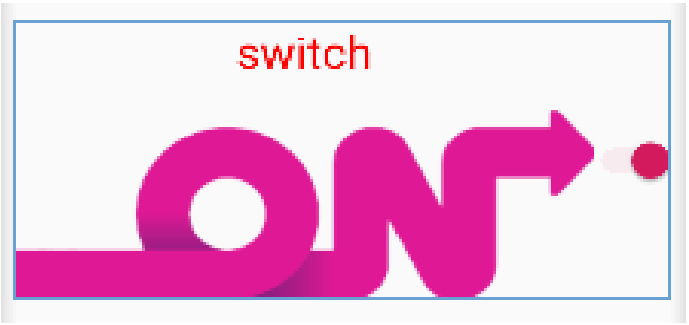
- **android:id:** Là thuộc tính duy nhất của **Switch**
- **android:checked:** checked là thuộc tính của **Switch** dùng để set trạng thái của **Switch**. Chúng ta cũng có thể set trạng thái của **Switch** bên Java Code bằng cách dùng phương thức setChecked(boolean status)

- **android:text:** thuộc tính text dùng hiển thị nội dung trong một **Switch**. Chúng ta có thể set thuộc tính này trong tập tin xml hoặc java code.
- **android:gravity:** Thuộc tính này thường sử dụng để canh nội dung trong **CheckBox**: **left, right, center, top, bottom, center_vertical, center_horizontal.**
- **android:textOn và android:testOff:** thuộc tính textOn được sử dụng để hiển thị câu thông báo khi Switch ở trạng thái checked. Chúng ta có thể set textOn trong XML, hoặc trong Java Class.

- **android:textColor:** Thuộc tính này dùng xác định màu chữ, dạng màu chữ: “#argb”, "#rgb”, “#rrggbb”, hoặc “#aarrggbb”
- **android:textSize:** Thuộc tính textSize xác định kích thước nội dung văn bản của **Switch**. Chúng ta có thể đặt kích thước văn bản theo: sp(scale independent pixel) hoặc dp(density pixel).
- **android:textStyle:** Thuộc tính xác định loại văn bản của **Switch**, thông thường có các loại văn bản:**bold, italic và normal.** Nếu chúng ta muốn sử nhiều hơn một loại văn bản thì phải thêm phép toán hoặc "|" vào giữa các loại văn bản.
- **android:background:** Thuộc tính này xác định màu nền cho **Switch**

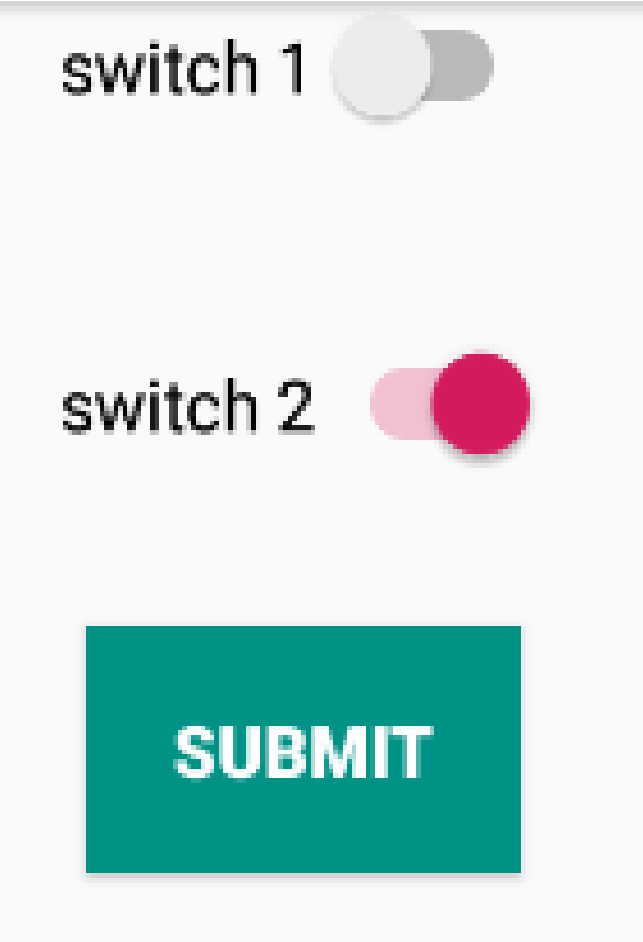
- **android:padding:** Thuộc tính này xác định khoảng cách từ đường viền của **Switch** với nội dung nó chứa: **left, right, top or bottom.**
- **android:drawableBottom, android:drawableTop,android: drawableRight và android:drawableLeft:** Các thuộc tính này hiển thị hình trong res/drawable theo các hướng: **bottom, top, right và left** nội dung văn bản của Switch.

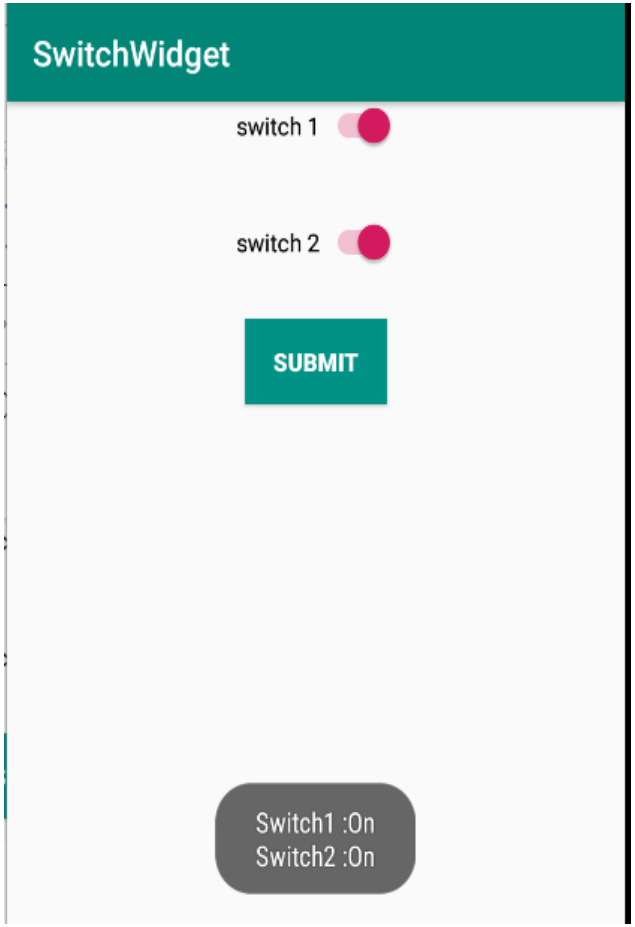
```
<Switch
    android:id="@+id/simpleSwitch"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:checked="true"
    android:drawableBottom="@drawable/on"
    android:gravity="center"
    android:text="switch"
    android:textColor="#f00"
    android:textOff="Off"
    android:textOn="On"
    android:textSize="25sp" />
```



Ví dụ:

- Xây dựng ứng dụng có 2 **Switch** và một **Button**. Khi người sử dụng chọn **Switch và** click **Button** sẽ hiện thị trạng thái hiện tại của **Switch** qua đối tượng TOAST


switch 1
switch 2
SUBMIT


SwitchWidget
switch 1
switch 2
SUBMIT
Switch1 :On
Switch2 :On

Bước 1:

- Tạo một project tên là Demo**Switch**: **File → New → Android Application Project** điền các thông tin → **Next →Finish.**

Bước 2:

- Vào thư mục **res/values** bổ sung **string.xml**

```
<resources>
    <string name="app_name">SwitchWidget</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
    <string name="switch1">Switch 1</string>
    <string name="switch2">Switch 2</string>
    <string name="btnsubmit">Submit</string>
</resources>
```

Bước 3:

- Mở **app → src -> MainActivity.java** và thêm code. Khi click Submit hiển thị trạng thái

```
public class MainActivity extends AppCompatActivity {
    //Khai báo các widget
    Switch simpleSwitch1, simpleSwitch2;
    Button submit;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }

    private void setControl() {
        simpleSwitch1 = (Switch) findViewById(R.id.simpleSwitch1);
        simpleSwitch2 = (Switch) findViewById(R.id.simpleSwitch2);
        submit = (Button) findViewById(R.id.btnSubmit);
    }
```

```
private void setEvent() {
        submit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String statusSwitch1, statusSwitch2;
                if (simpleSwitch1.isChecked())
                    statusSwitch1 = simpleSwitch1.getTextOn().toString();
                else
                    statusSwitch1 = simpleSwitch1.getTextOff().toString();
                if (simpleSwitch2.isChecked())
                    statusSwitch2 = simpleSwitch2.getTextOn().toString();
                else
                    statusSwitch2 = simpleSwitch2.getTextOff().toString();
                Toast.makeText(getApplicationContext(), "Switch1 :" + statusSwitch1
+ "\n" + "Switch2 :" + statusSwitch2, Toast.LENGTH_LONG).show(); // display
the current state for switch's
            }
        });
    }
}
```

ScrollView

ScrollView

- ScrollView là một dạng đặc biệt của FrameLayout ở chỗ nó cho phép người dùng di chuyển qua một danh sách các quan điểm mà chiếm nhiều không gian hơn màn hình vật lý

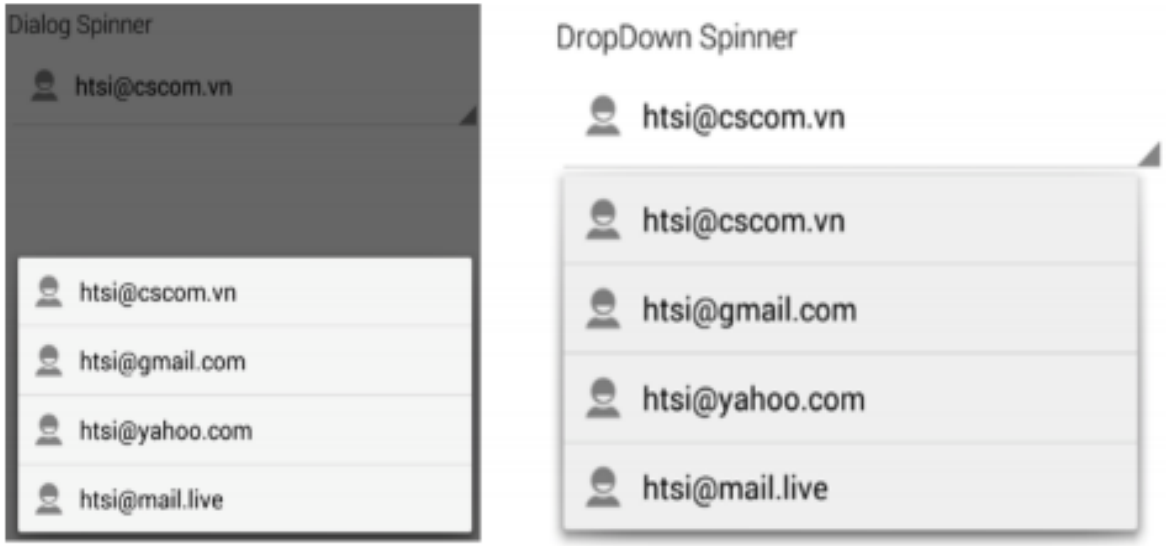
Thuộc tính thường dùng của ScrollView

- **android:id:** Là thuộc tính duy nhất của **ScrollView**.
- **android:scrollbars:** Thuộc tính này được sử dụng hiển thị thanh cuộn trong **ScrollView,** giá trị có thể là **"vertical, horizontal"**. Mặc định cuộn theo thẳng đứng

Spinner

Spinner

- Spinner là đối tượng điều khiển hiển thị một danh mục ở một thời điểm, người dùng có thể lựa chọn một trong nhiều danh mục để hiển thị. Bao gồm hai chế độ hiển thị pop-up lựa chọn (spinnerMode)



Thuộc tính XML quan trọng:

- **spinnerMode**: dialog | dropdown
- **prompt**: string
- **popupBackground**: drawable | color

Một số phương thức quan trọng:

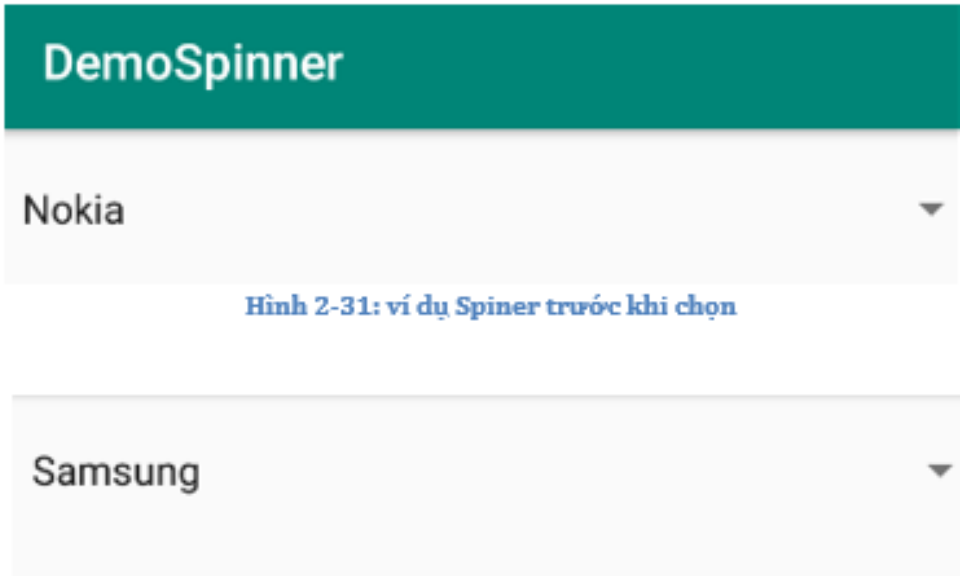
- **setAdapter**(SpinnerAdapter)
- **setPrompt**(CharSequence) – setPrompt(int resId) (Dialog Mode)
- **setPopupBackgroundResource**(int)
- **setPopupBackgroundDrawable**(Drawable)

Các sự kiện thường dùng trong Spinner

- **setOnItemSelectedListener** Sự kiện này xảy ra khi người dùng click chọn **Spinner**
  - **onItemSelected**: gọi phương thức này được gọi khi có một sự kiện chọn item nào đó.
  - **onNothingSelected**: phương thức này được gọi khi click vào Spinner mà không chọn item nào cả.

Ví dụ:

- Xây dựng Spinner hiển thị danh sách các mặt hàng, khi chọn một mặt hàng sẽ hiển thị thông báo.



Hình 2-31: ví dụ Spiner trước khi chọn

Bước 1:

- Tạo một project tên là DemoSpinner: **File → New → Android Application Project** điền các thông tin → **Next →Finish.**

Bước 2:

- Mở **res → layout → xml (hoặc) activity_main.xml**

Bước 3:

- Mở **app → src -> MainActivity.java** và thêm code. Khởi tạo **Spinner.** Tiếp theo, tạo 1 mảng dữ liệu và kết nối **Spinner** thông qua đối tượng **ArrayAdapter.** Trong bước này chúng ta cũng thiết lập sự kiện **setOnItemSelectedListener(new OnItemSelectedListener())** cho **Spinner**

```
public class MainActivity extends AppCompatActivity {
    //Khai báo các widget
    String mobilePhones[] = {"Nokia", "Samsung", "Iphone", "HTC",
"BPhone", "MobileStar"};
    Spinner spnMobile;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }

    private void setControl() {
        spnMobile = (Spinner) findViewById(R.id.spinner1);
    }
```

```
private void setEvent() {
        ArrayAdapter adapter = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, mobilePhones);
adapter.setDropDownViewResource(android.R.layout.select_dialog_singlechoice);
        spnMobile.setAdapter(adapter);
        spnMobile.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> arg0, View arg1,
                                int arg2, long arg3) {
                // TODO Auto-generated method stub
                Toast.makeText(getApplication(), "Bạn đã chọn mục:" +
mobilePhones[arg2], Toast.LENGTH_LONG).show();
            }
```

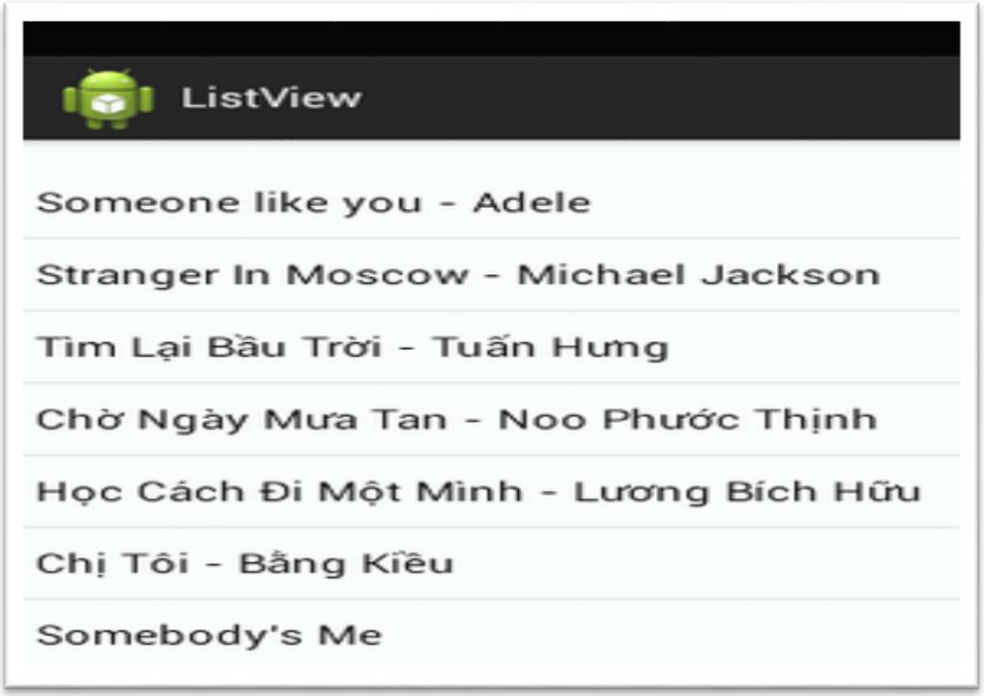
```
        @Override
        public void onNothingSelected(AdapterView<?> arg0) {
            // TODO Auto-generated method stub
            Toast.makeText(MainActivity.this, "bạn chưa chọn",
Toast.LENGTH_SHORT).show();
        }
    });


}
}
```

ListView

ListView

- ListView là một dạng điều khiển nâng cao có chức năng hỗ trợ hiển thị dữ liệu theo dạng từng dòng. ListView cần một đối tượng Adapter để quản lý và giúp hiển thị dữ liệu.



Thuộc tính thường dùng của ListView

- **android:id:** Là thuộc tính duy nhất của **ListView**.
- **android:divider:** Thuộc tính này có thể là một image hay màu dùng để phân chia giữa các dòng trong **ListView.**
- **android:dividerHeight:** Thuộc tính này xác định chiều cao của thuộc tính **android:divider.**

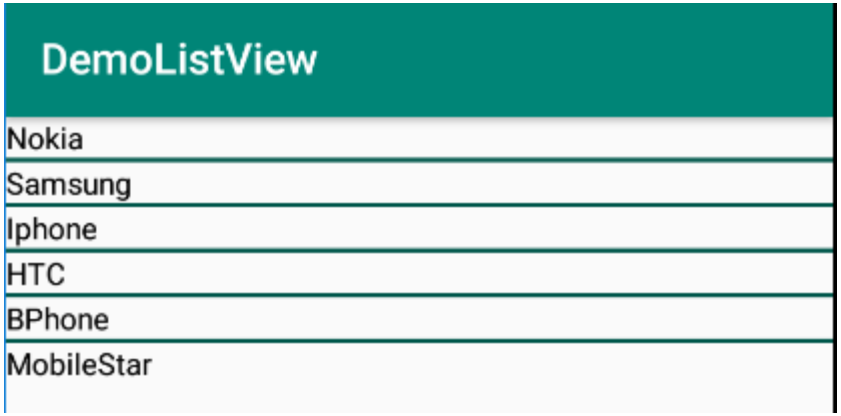
- **android:listSelector**: Thuộc tính này thường được sử dụng để thiết lập màu hoặc hình dòng được chọn trong listView. Thường nó sử dụng màu cam hoặc màu xanh dương, nhưng chúng ta cũng có thể thiết lập lại màu hoặc image cho ListView

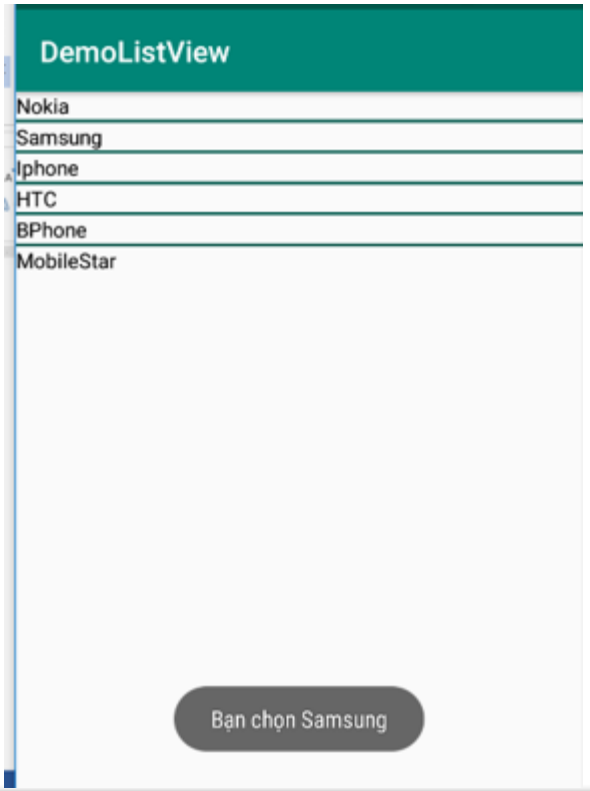
Sự kiện thường dùng trong ListView

- **setOnItemClickListener:** Sự kiện này xảy ra khi người dùng click **lên ListView.**
- **setOnItemLongClickListener:** Sự kiện được gắn cho ListView Item, khi nhấn lâu từ 2.5 tới 3 giây thì sự kiện này sẽ sảy ra

Ví dụ:

- Xây dựng ứng dụng gồm hiển thị danh sách sản phẩm sử dụng ListView**. Khi chọn sản phẩm sẽ hiển thị thông báo**





Bước 1:

- Tạo một project tên là DemoListView: **File → New → Android Application Project** điền các thông tin → **Next →Finish.**

Bước 2:

- Mở **res → layout → xml (hoặc) activity_main.xml**

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ListView
        android:id="@+id/simpleListView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:divider="@color/colorPrimaryDark"
        android:dividerHeight="2dp" />
</LinearLayout>
```

Bước 3:

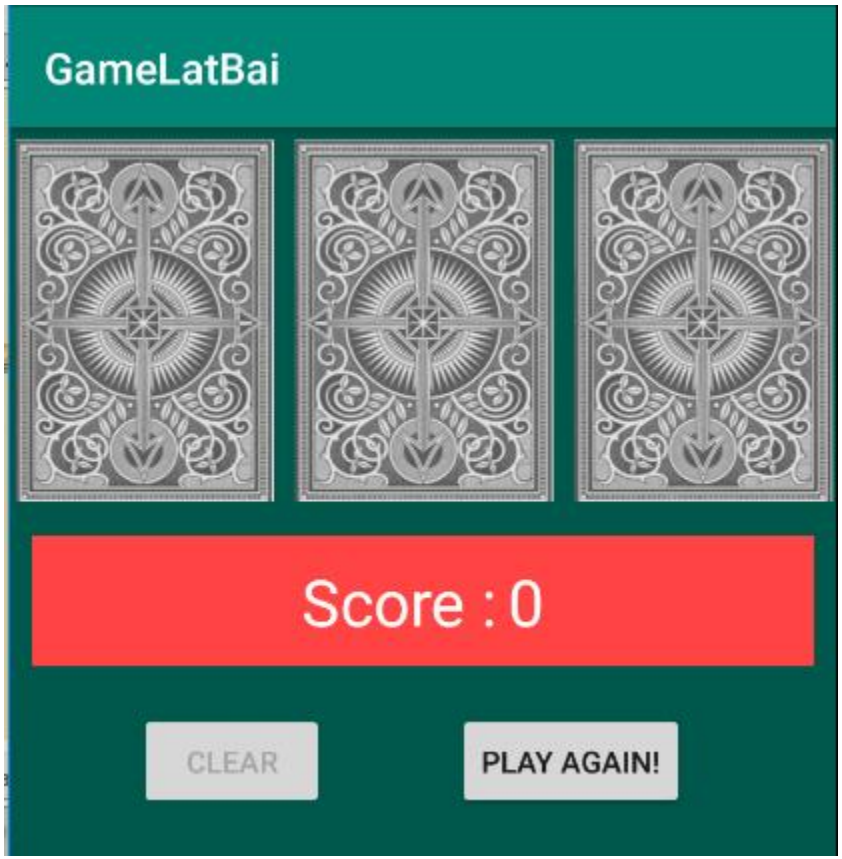
- Mở **app → src -> MainActivity.java** và thêm code, tạo 1 mảng dữ liệu và kết nối **ListView** thông qua đối tượng **ArrayAdapter.**

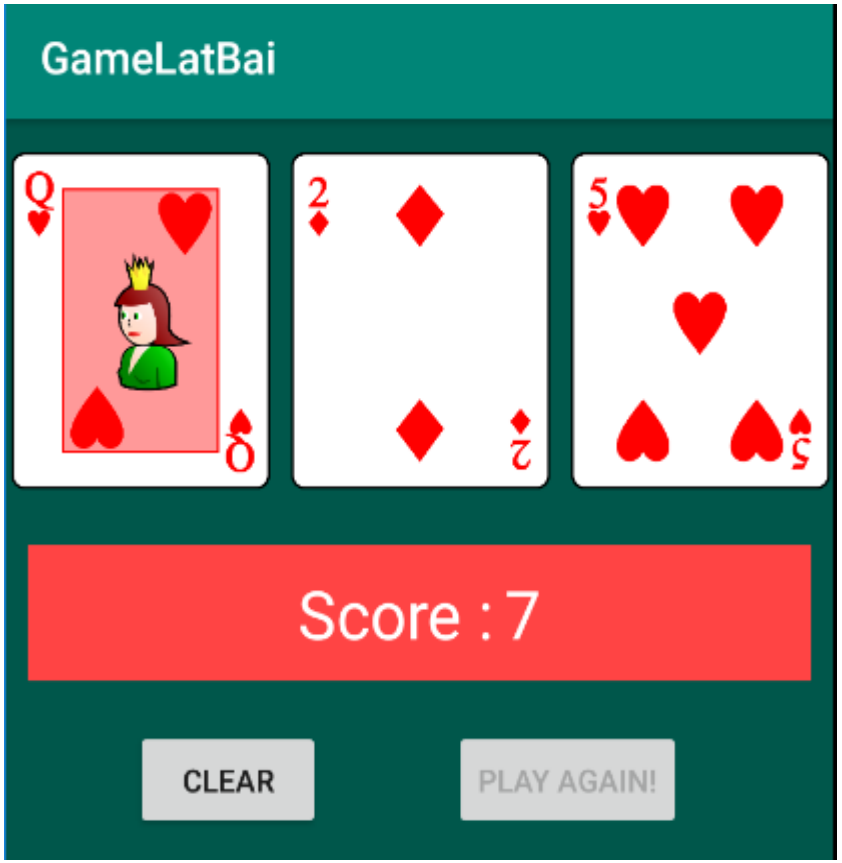
```
public class MainActivity extends AppCompatActivity {
    //Khai báo các widget
    String mobilePhones[] = {"Nokia", "Samsung", "Iphone", "HTC", "BPhone",
"MobileStar"};
    ListView simpleList;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setControl();
        setEvent();
    }
    private void setControl() {
        simpleList = (ListView) findViewById(R.id.simpleListView);
    }
```

```
    private void setEvent() {
        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(this,
R.layout.activity_listview, R.id.textView, mobilePhones);
        simpleList.setAdapter(arrayAdapter);
        simpleList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position,
long id) {
                Toast.makeText(MainActivity.this, "Bạn chọn " +
mobilePhones[position], Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

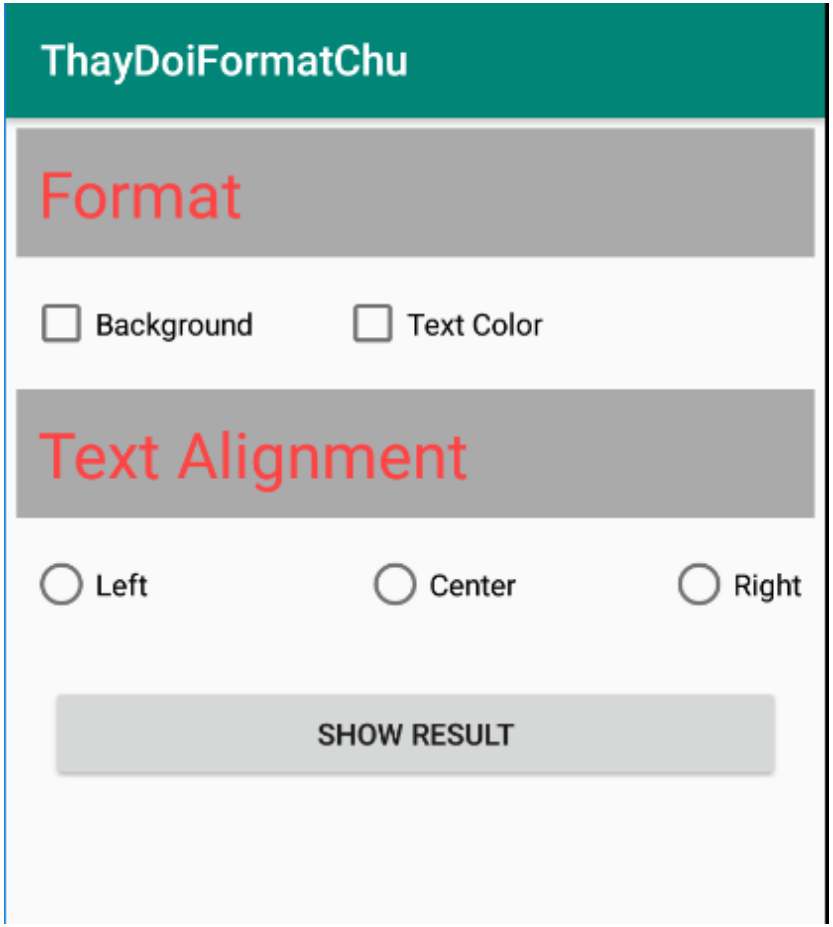
Bài Tập

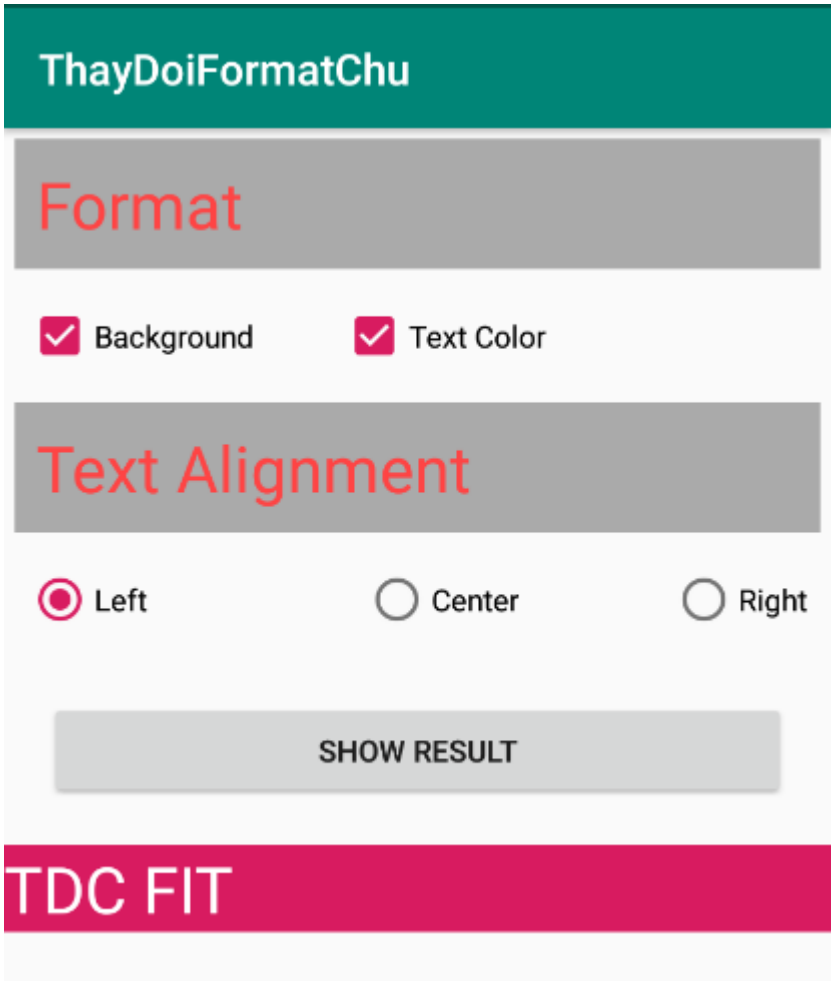
Bài 3:Lật bài và tính điểm



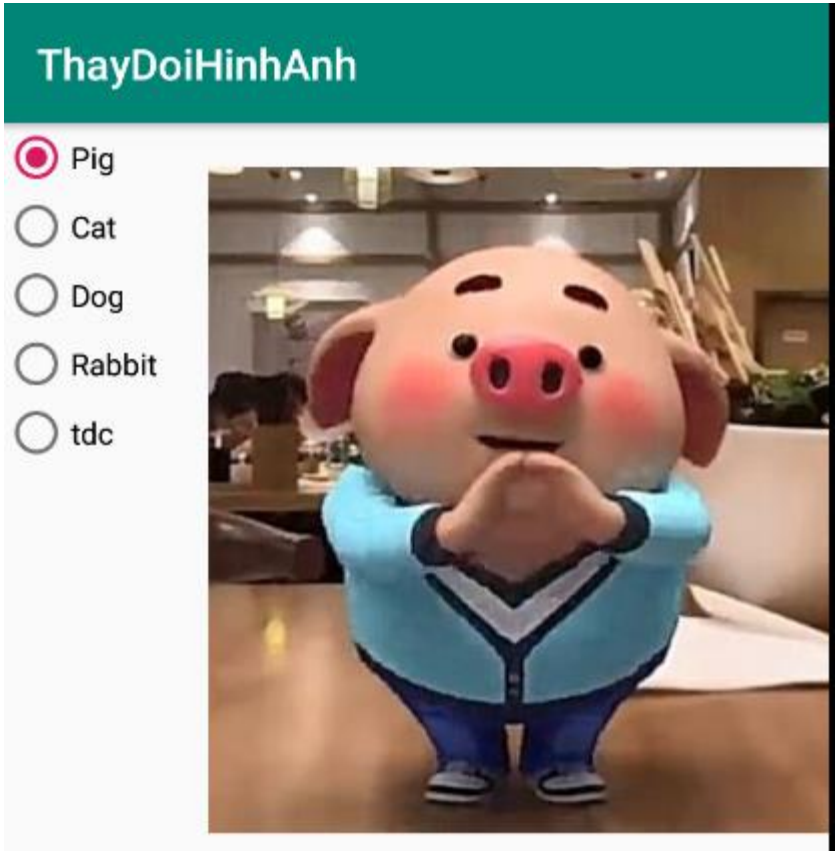


Bài 4: Thay đổi font chữ

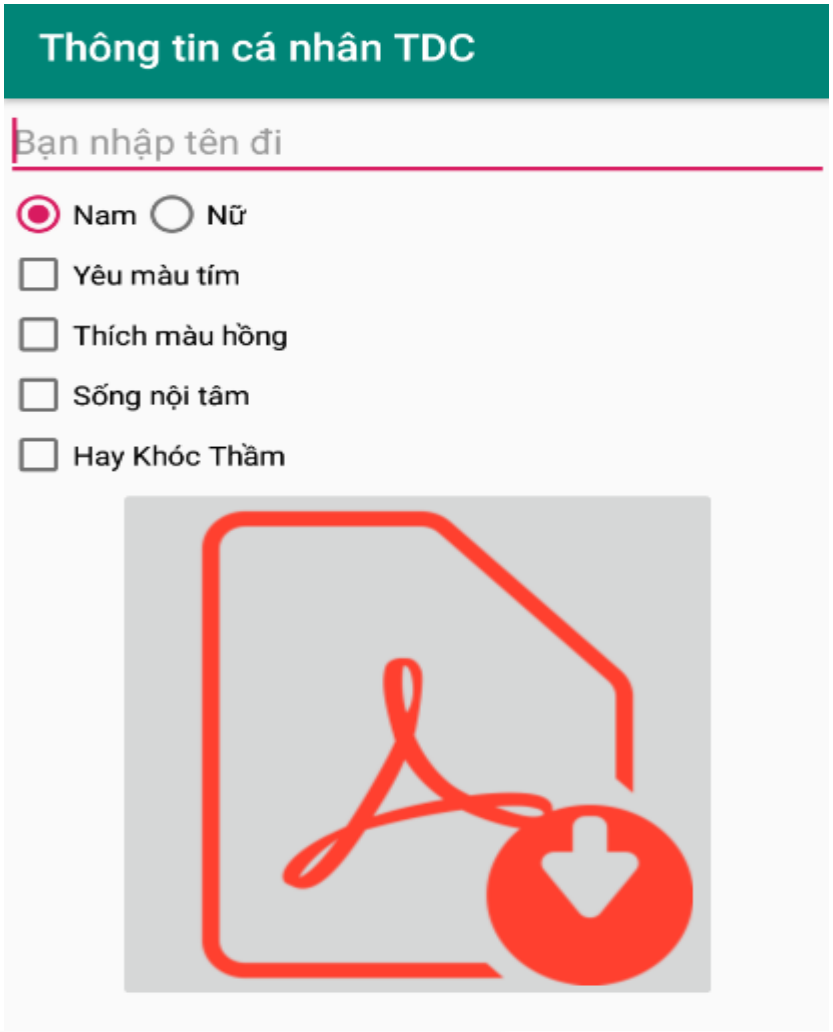


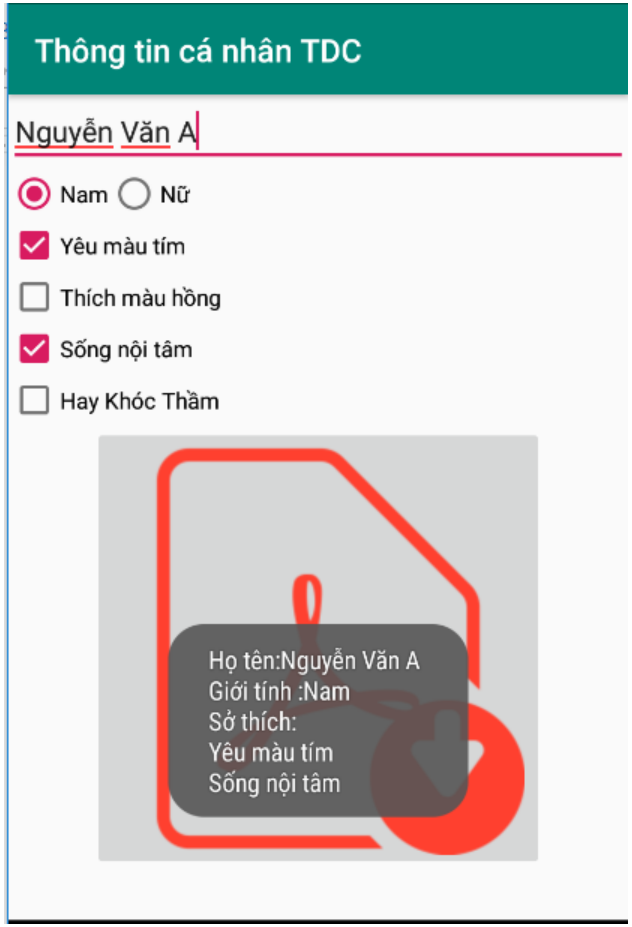


Bài 5: Thay đổi hình

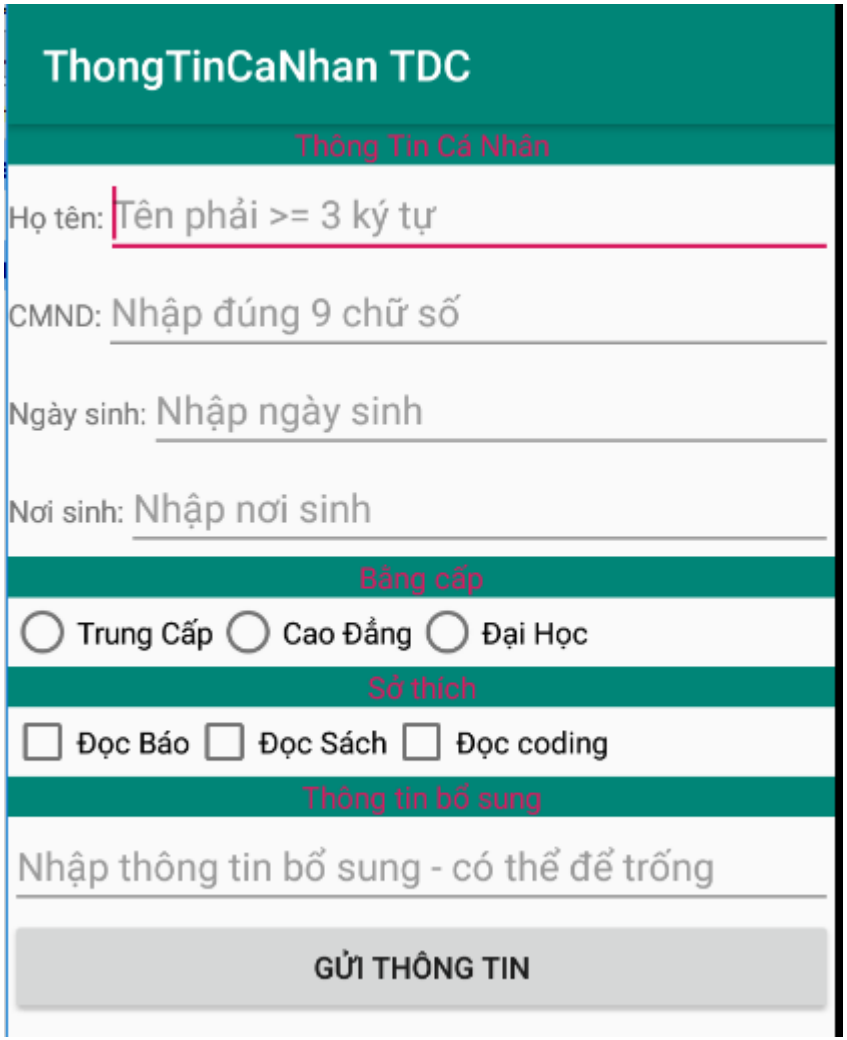


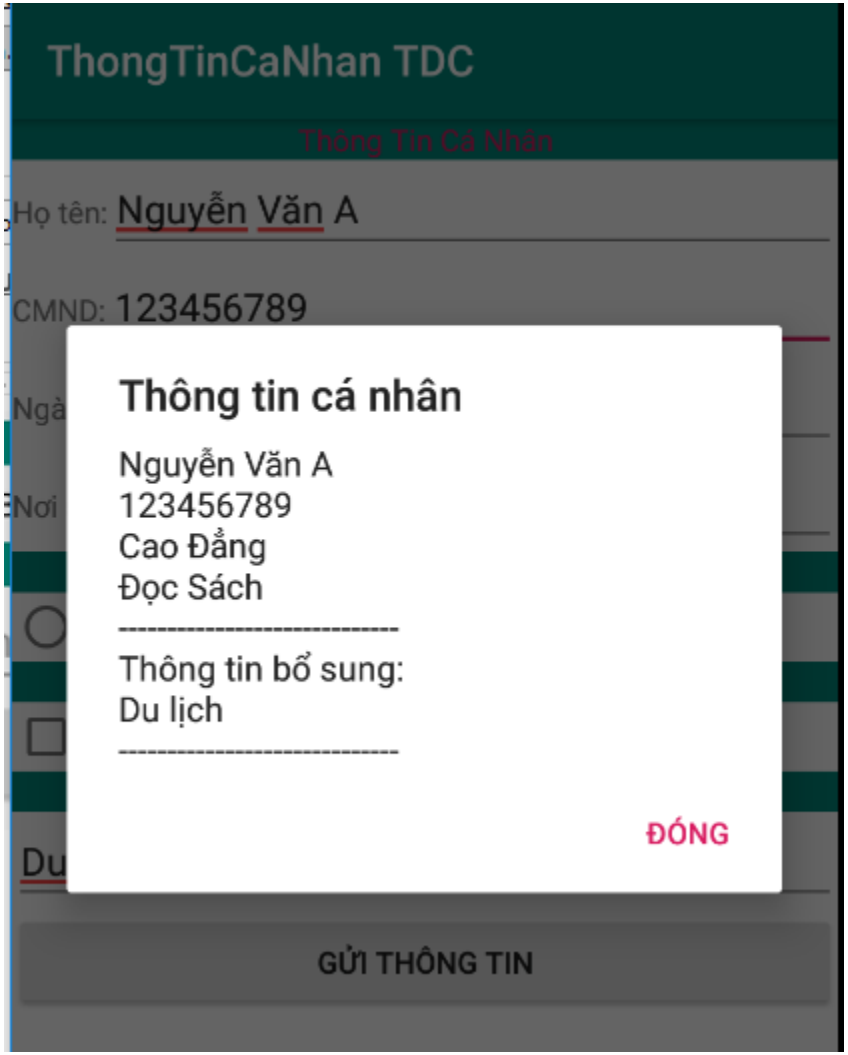
Bài 6: Thu thập thông tin cá nhân



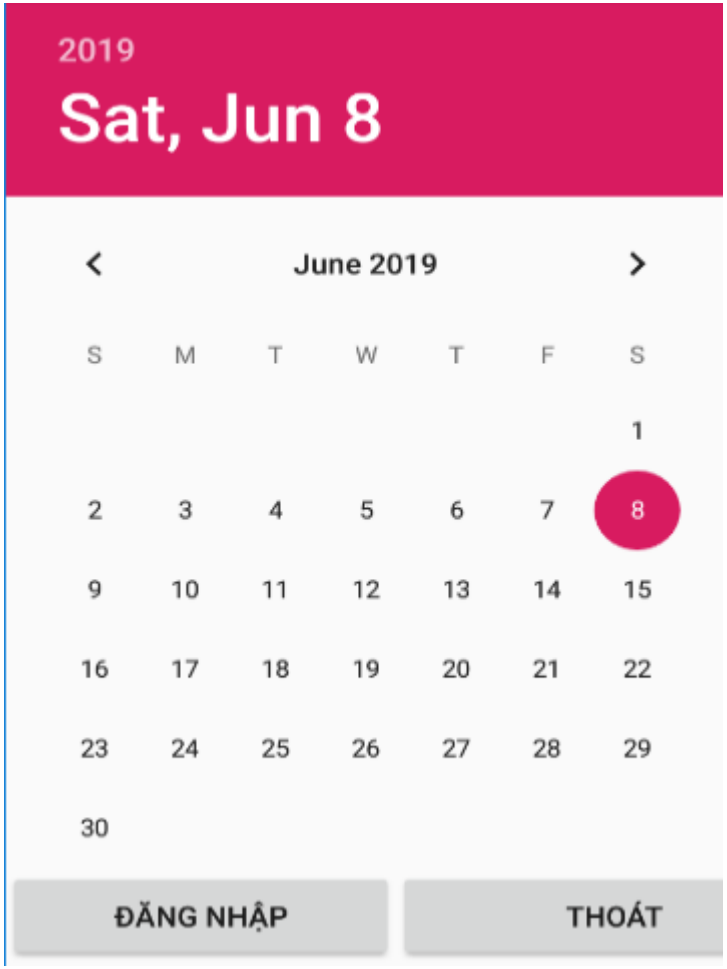


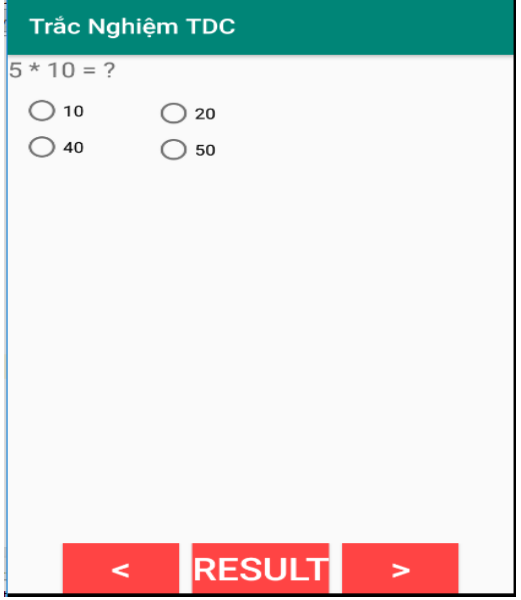
Bài 7:Thông tin cá nhân

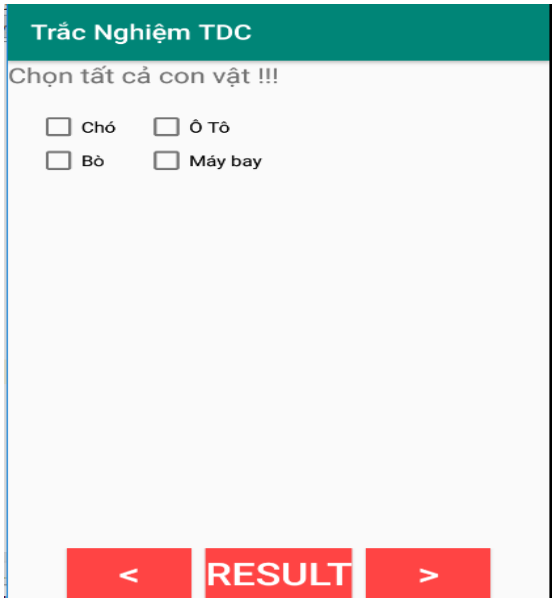


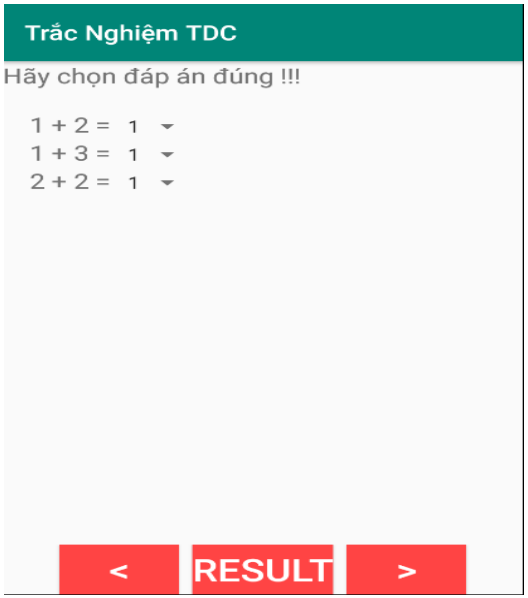


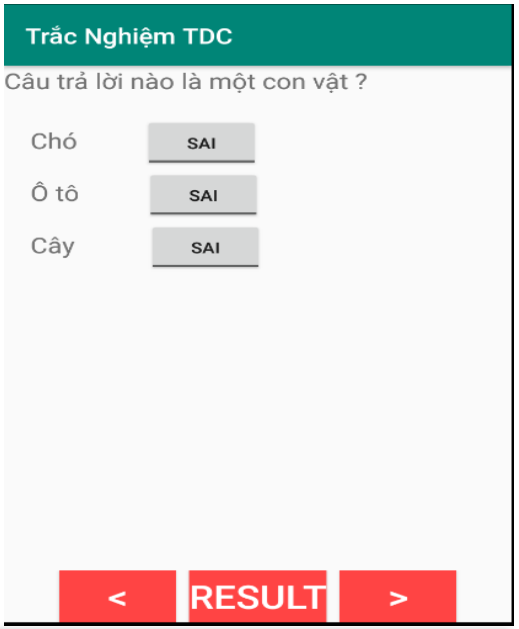
Bài 8: Xây dựng trắc nghiệm

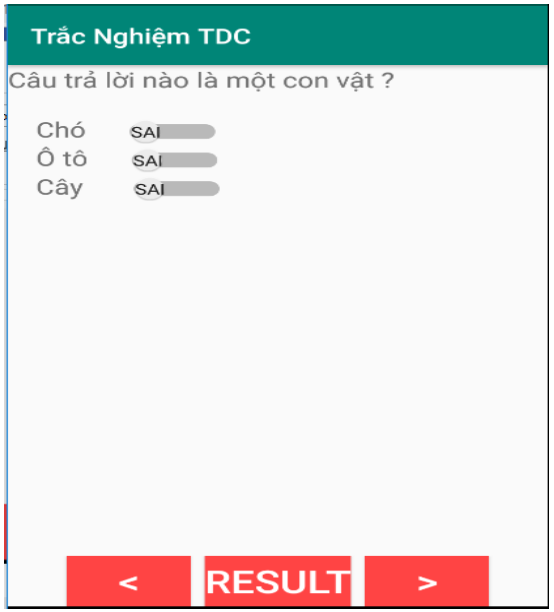


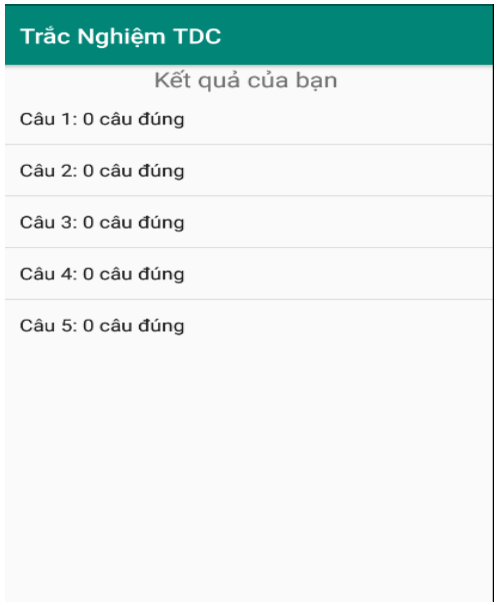

Trắc Nghiệm TDC
5 * 10 = ?
10
20
40
50
<
RESULT
>


Trắc Nghiệm TDC
Chọn tất cả con vật !!!
Chó
Ô Tô
Bò
Máy bay
<
RESULT
>


Trắc Nghiệm TDC
Hãy chọn đáp án đúng !!!
1 + 2 = 1
1 + 3 = 1
2 + 2 = 1
<
RESULT
>


Trắc Nghiệm TDC
Câu trả lời nào là một con vật ?
Chó
SAI
Ô tô
SAI
Cây
SAI
<
RESULT
>


Trắc Nghiệm TDC
Câu trả lời nào là một con vật ?
Chó
SAI
Ô tô
SAI
Cây
SAI
<
RESULT
>


Trắc Nghiệm TDC
Kết quả của bạn
Câu 1: 0 câu đúng
Câu 2: 0 câu đúng
Câu 3: 0 câu đúng
Câu 4: 0 câu đúng
Câu 5: 0 câu đúng

CẢM ƠN TẤT CẢ ĐÃ LẮNG NGHE