

## THỰC HÀNH BUỔI 2 – BÀI SỐ 3

### LẬP TRÌNH GIAO DIỆN NGƯỜI DÙNG MỨC THẤP

#### I. NỘI DUNG CẦN ÔN TẬP

- Vẽ một đối tượng đơn giản
- Xử lý sự kiện người dùng tương tác lên đối tượng mức thấp
- Làm cho đối tượng di chuyển
- Lập trình luồng trong giao diện mức thấp
- Lập trình với SurfaceView

#### II. THỰC HÀNH

Trong buổi thực hành này, sinh viên mở 1 project mới tên là **Bai\_3\_Ten\_MSSV** với **MainActivity.java** và layout tương ứng là **activity\_main.xml**, lần lượt tạo ra các Class và thực hiện các nội dung sau:

**Class 1:** Lập trình 1 lớp Java vẽ các đối tượng cơ bản (tên là **VeCoBan.java**) là kế thừa của lớp **View** thực hiện vẽ các đối tượng như sau: vẽ chuỗi ký tự (hay văn bản) hiển thị tên chủ đề “LẬP TRÌNH GIAO DIỆN MỨC THẤP” và thông tin kích thước màn hình, vẽ một hình **cung** (arc), một **hình tròn** (circle), một **đường** (line), **hình chữ nhật** (rectangle), một **chuỗi ký tự nằm nghiêng lên 45°**, một **ảnh bitmap**,... như hình 2.1.

**Ghi chú:** Cách mở 1 lớp Java mới, giả sử đang để cây thư mục của project ở chế độ Android, chọn thư mục java → Chọn thư mục có tên là Package name → Ấn chuột phải → Chọn New → Chọn Java class → Khai báo tên lớp mới → ... Sau khi hoàn thành lớp java, để chạy thử, ta sửa lại tập tin MainActivity.java như sau (để chạy thử lớp VeCoBan.java).

**Lưu ý:** Chưa cần lập trình Layout với tập tin xml. Để hiển thị giao diện mức thấp với lớp VeCoBan, ta khai báo 1 thể hiện của lớp VeCoBan và thay giao diện xml (R.layout.activity\_main) bằng thể hiện này (phần được highlight).

```
package com.example.b2_1;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main); // Tạm thời cho lệnh này thành chú giải
        VeCoBan ve = new VeCoBan(this); // Khai báo 1 thể hiện của lớp VeCoBan
        setContentView(ve);
    }
}
```

Kết quả chạy thử như hình 2.1.

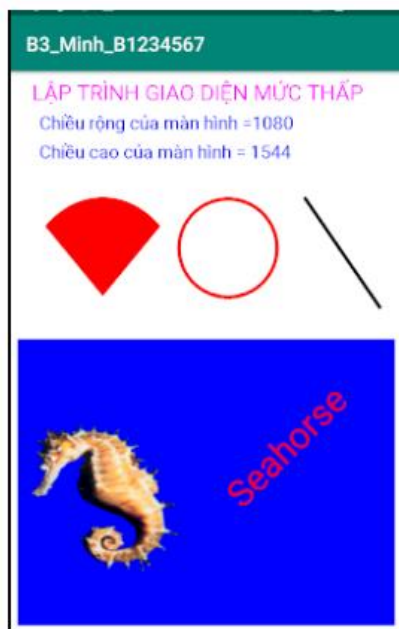
### Class 2: Lập trình cho đối tượng di chuyển

Lập trình 1 lớp Java tạo đối tượng di chuyển (đặt tên là **ChuyenDong.java**): sử dụng hai ảnh có sẵn (có thể tải về hình các quả cầu **internet.png** và **network.png** từ internet) lập trình vẽ đối tượng (ví dụ ảnh **internet.png**) di chuyển ngẫu nhiên. Nếu đối tượng chạm vào cạnh của màn hình thì đổi chiều chuyển động đồng thời đổi sang ảnh khác (ví dụ ảnh **networking**).

Sửa lại tập tin MainActivity (thay lớp VeCoBan bằng lớp ChuyenDong, kết quả như hình 2.2.

### Class 3: Xử lý sự kiện tương tác của người dùng

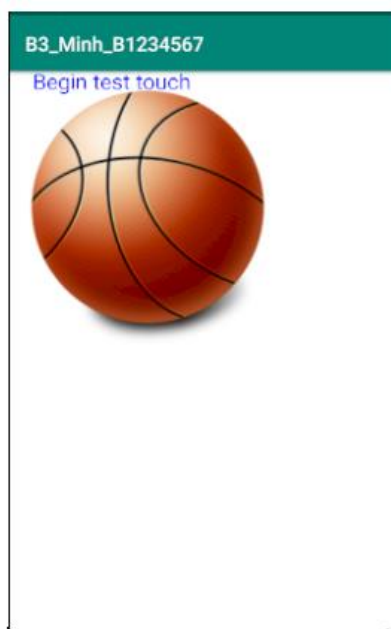
Lập trình tạo lớp xử lý sự kiện tương tác người dùng, hoạt động như sau: vẽ một ảnh có sẵn ra màn hình, chẳng hạn một quả bóng, giao diện như hình 2.3, khi người dùng chạm ngón tay vào quả bóng thì sẽ hiện lên chuỗi ký tự “ACTION Down”, khi ngón tay chạm-giữ và di chuyển (rê) trên màn hình thì quả bóng sẽ di chuyển theo và hiện chữ “ACTION Move”, khi ngón tay nhấc khỏi màn hình thì hiện dòng chữ “ACTION Up”.



Hình 2.1



Hình 2.2



Hình 2.3

### Class 4: Lập trình luồng với lớp SurfaceView

Phát triển một ứng dụng game đơn giản như sau: vẽ một đối tượng hình ảnh (chẳng hạn hình con rệp có tên tập tin là **bug.png**) di chuyển ngẫu nhiên trên canvas (với một hình nền tùy chọn), dòng chữ “Time: 6” cho biết thời gian chơi còn lại (giây), như hình 2.4. Khi người chơi chạm vào đối tượng bug thì nó sẽ biến mất rồi xuất hiện lại ở vị trí ban đầu và tiếp tục di chuyển, khi đó được cộng một điểm. Sau 10 giây nếu từ 3 điểm trở lên thì hiển thị kết quả cho

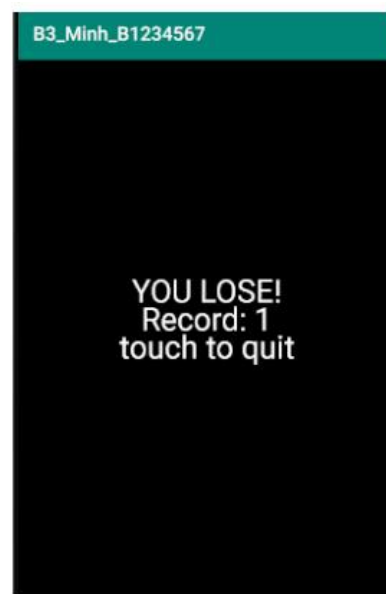
biết đã thắng với số điểm đạt được (trong một màn hình mới) như hình 2.5, ngược lại hiển thị kết quả thua như hình 2.6.



Hình 2.4



Hình 2.5



Hình 2.6

### Lớp MainActivity (Nội dung 5): Lập trình Layout cho MainActivity.

- **Lập trình LinearLayout** gồm 1 Framlayout con và 1 TableLayout con như hình 2.7. khi mới khởi động, FramLayout hiển thị 1 text “BÀI THỰC HÀNH BUỔI 2” ở phía trên và mở rộng đến gần cuối màn hình, dùng thuộc tính

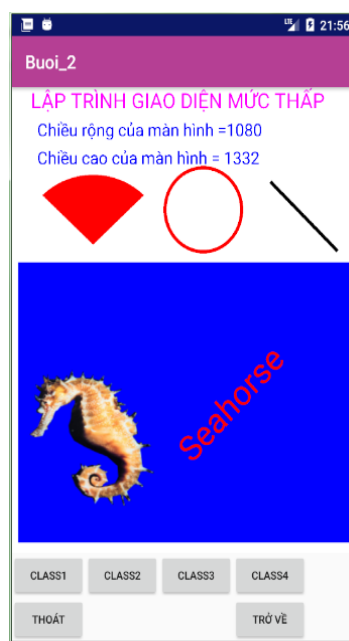
**android:layout\_height="0dp"**

và **android:layout\_weight="1"**

1 TableLayout gồm 2 hàng, 4 cột chứa 6 Button và 2 TextView rỗng.



Hình 2.7



Hình 2.8



Hình 2.9

- **Lập trình tương tác**, mở MainActivity và lập trình xử lý tương tác của người dùng lên các button: khi button CLASS1 được ấn thì Class1 chạy, hiển thị lên FrameLayout như hình 2.8, khi button CLASS2 được ấn thì Class2 chạy, hiển thị lên FrameLayout như hình 2.9, khi button CLASS3 được ấn thì Class3 chạy, hiển thị lên FrameLayout như hình 2.10, khi button CLASS4 được ấn thì Class1 chạy, hiển thị lên FrameLayout như hình 2.11, khi button TRỞ VỀ được ấn thì khởi động lại ứng dụng, khi button THOÁT được ấn thì ứng dụng kết thúc.



Hình 2.10



Hình 2.11

## HƯỚNG DẪN

SV thực hiện các bước như sau:

**Bước 1:** Tạo ra một project (Trong ví dụ này được đặt tên là **B3\_Tên SV\_MSSV**) với một Activity chính (MainActivity).

**Bước 2:** Xây dựng **VeCoBan**, đây là một lớp kế thừa lớp View để thực thi hoạt động vẽ. Trong lớp này ta lần lượt khai báo các đối tượng cần thiết như: đối tượng kiểu Paint (kế thừa lớp Paint trong gói graphics); đối tượng Canvas (lớp Canvas trong gói graphics) để vẽ trên đó; ghi đè lên phương thức onDraw.

Để hiển thị giao diện người dùng mức thấp ta khai báo lớp này trong MainActivity.

### **CLASS1:** Xây dựng lớp VeCoBan

Nhấp chuột phải vào thư mục có tên Package name của project (chứa lớp MainActivity) → chọn New → Chọn Java class → chọn và nhập vào tên lớp → check chọn public.

**Để lập trình lớp VeCoBan, SV cần tham khảo phần lý thuyết sau đây:**

Giáo trình lập trình cho thiết bị di động

Lớp **VeCoBan** kế thừa lại các phương thức cũng như các thuộc tính của lớp **View**. Trong đó sẽ cài đặt lại lệnh khởi tạo cho lớp **VeCoBan**:

```
public VeCoBan(Context context) { super(context); }
```

và ghi đè phương thức **onDraw()**, đây chính là phương thức vẽ ra giao diện.

Phương thức khởi tạo **public VeCoBan(Context context)** chỉ nhận vào một tham số là một đối tượng **Context**. Tham số này cho phép truy xuất đến các đối tượng cũng như các dịch vụ của hệ thống. Đối tượng Context hỗ trợ việc vẽ một giao diện ra màn hình thiết bị.

Để vẽ ta phải tạo một đối tượng **Paint**, dùng để định nghĩa màu sắc, style... Nói một cách đơn giản thì đối tượng paint đóng vai trò như một biến có thể gán và sử dụng nhiều giá trị khác nhau, nếu muốn thay đổi giá trị ta chỉ việc gán lại chứ không cần phải khai báo mới.

Lớp **VeCoban** đã dùng lớp **RectF()** và phương thức **drawOval**, **drawArc**, **drawCircle**, **drawLine**, **drawRect**, **drawText**, **drawBitmap** để vẽ các hình cung, hình tròn, đường thẳng, hình chữ nhật, text, text xoay theo một phương xác định và một ảnh được lưu trong thư mục **res\drawable**. Thư viện **android.graphics.Canvas** hỗ trợ nhiều phương thức để vẽ các hình dạng khác nhau, chúng ta có thể tham khảo tại URL:

<http://developer.android.com/reference/android/graphics/Canvas.html>

Phần sau đây giải thích một số phương thức đã được sử dụng:

### (1) Vẽ một cung

Để vẽ 1 cung ta thực hiện các bước sau:

- Khai báo đối tượng hình chữ nhật **rectF**, kế thừa lớp **RectF**, với lệnh:

```
RectF rectF = new RectF(float left, float top, float right, float bottom);
```

Đối số vào gồm 4 tọa độ kiểu **float** cho 1 hình chữ nhật tương ứng với 4 cạnh của nó (trái, đỉnh, phải, đáy). Lưu ý (tọa độ trái <= tọa độ phải và tọa độ đỉnh <= tọa độ đáy).

- Vẽ hình oval nằm trong đối tượng chữ nhật **rectF**, với lệnh:

```
c.drawOval(rectF, paint);
```

- Xác định các thuộc tính **color** của đối tượng **Paint** đã khai báo, với các lệnh:

```
paint.setColor(int color);
```

với **color** là các màu đã được định nghĩa như **RED**, **BLUE**, **GREEN**, **CYAN**, **MAGENTA**, **YELLOW**, **BLACK**, **WHITE**,...

Hoặc định nghĩa một màu mới bằng phương thức **public void setARGB (int a, int r, int g, int b)**, các đối số **a**, **r**, **g**, **b** lần lượt là độ bão hòa màu và 3 màu cơ bản lấy giá trị là **int** từ 0 đến 255.



- Xác định các thuộc tính **style** của đối tượng **Paint** đã khai báo, với các lệnh:

```
paint.setStyle(Style.style);
```

trong đó **style** lấy các giá trị là **FILL** (đầy màu, liền nét), **STROKE** (bên trong không tô màu, nét gián đoạn), hoặc **FILL\_AND\_STROKE**. Mặc định là vẽ liền nét và bên trong không tô màu.

- Vẽ một cung với phương thức **drawArc**

**public void drawArc (RectF oval, float startAngle, float sweepAngle, boolean useCenter, Paint paint)**

Phương thức này vẽ một cung nằm trong hình oval đã khai báo. Các đối số vào còn lại lần lượt là góc bắt đầu (startAngle), góc quét (sweepAngle) theo chiều kim đồng hồ để vẽ nên cung, nếu đối số useCenter có giá trị là true thì cung sẽ chứa tâm của hình oval () , nếu có giá trị là false thì không chứa tâm (). Nếu startAngle có giá trị âm hoặc  $\geq 360$ , thì góc bắt đầu sẽ là modulo 360. Nếu sweepAngle  $\geq 360$ , thì hình cung sẽ lấp đầy hình oval. Nếu sweepAngle  $< 0$  thì sweepAngle là modulo 360. Cung được vẽ theo chiều kim đồng hồ, góc  $0^0$  sẽ tương ứng với 3 giờ trên đồng hồ, paint là đối tượng Paint đã được khai báo.

## (2) Vẽ hình tròn

Sau khi đặt các thuộc tính color và style cho đối tượng paint (nếu muốn thay đổi so với thuộc tính đã đặt trước), ta dùng phương thức sau:

**drawCircle(float cx, float cy, float radius, Paint paint)**

Các đối số vào lần lượt là toạ độ (cx, cy) của tâm và bán kính radius của hình tròn lấy giá trị là số float, paint là đối tượng Paint đã được khai báo.

## (3) Vẽ hình chữ nhật

Sau khi đặt các thuộc tính color và style cho đối tượng paint (nếu muốn thay đổi so với thuộc tính đã đặt trước), ta dùng phương thức sau:

**addRect(float left, float top, float right, float bottom, Paint paint);**

Các đối số vào lần lượt là toạ độ cạnh trái(left), cạnh trên (top), cạnh phải (right), cạnh đáy (bottom) của hình chữ nhật lấy giá trị là số float, paint là đối tượng Paint đã được khai báo.

## (4) Vẽ text

Để vẽ Text ta dùng phương thức:

**public void drawText (String text, float x, float y, Paint paint)**

Với (x,y) là toạ độ bắt đầu của text. Ngoài ra a có thể đặt cỡ chữ bằng lệnh sau:

**paint.setTextSize(size);** với size là số nguyên qui định cỡ chữ.

## (5) Vẽ text theo một hướng

Cũng dùng phương thức **drawText** như vẽ text bình thường, để xoay theo một hướng ta dùng thêm phương thức **rotate** trước khi gọi phương thức drawText. Phương thức rotate có 2 cú pháp như sau:

**public void rotate (float degrees)** , tham số degrees là góc quay tính theo độ.

Hoặc: **public final void rotate (float degrees, float px, float py)** , có thêm 2 đối số:

**px** Tọa độ x cho điểm trục (không đổi bởi phép quay)

**py** Tọa độ y cho điểm trục (không đổi bởi phép quay)

## (6) Vẽ một ảnh được lưu trong thư mục res\drawable



Trước tiên ta phải chép ảnh muốn vẽ (định dạng png) vào thư mục drawable và khai báo 1 đối tượng ảnh trong resources bằng phương thức getResources với cú pháp như sau:

**Resources res = this.getResources();**

Tiếp theo là khai báo 1 đối tượng Bitmap với cú pháp như sau:

**Bitmap anh = BitmapFactory.decodeResource(res, R.drawable.imagename);**

Với **imagename** là tên của file ảnh muốn vẽ (đã chép vào thư mục drawable).

Sau cùng dùng phương thức drawBitmap để vẽ ảnh, cú pháp như sau:

**public void drawBitmap (Bitmap anh, Rect scr, RectF dst, Paint paint)**

Các tham số là:

**anh** Ảnh bitmap cần vẽ, đã khai báo trước.

**Rect** Tập hợp con các ảnh bitmap được vẽ, có thể là null.

**RectF** Hình chữ nhật mà ảnh bitmap sẽ được vẽ vừa khít trong đó.

**paint** Là đối tượng Paint đã khai báo, có thể là null.

### (7) Phương thức public void restore ()

Dùng để lưu tất cả những gì đã vẽ trước đó cho đến khi phương thức này được gọi.

**Ghi chú:** Không cần xây dựng Layout với tập tin XML, và cũng không cần điều chỉnh tập tin XML Layout mẫu của ADT. Vì ứng dụng sẽ sửa dụng đối tượng canvas để vẽ trên đó.

Sinh viên tham khảo tập tin **VeCoBan.java** hoàn chỉnh như sau:

```
package com.example.b3_minh_b1234567;

import android.content.Context;
import android.content.res.Resources;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.RectF;
import android.view.View;

public class VeCoBan extends View {
    private final Paint paint=new Paint();
    public VeCoBan(Context context){super(context);}
    @Override
    protected void onDraw (Canvas canvas){
        super.onDraw(canvas);
        int xcanvas = canvas.getWidth(); //Lấy chiều ngang của màn hình.
        int ycanvas = canvas.getHeight(); // Lấy chiều dọc của màn hình.
        // cho canvas màu trắng
        paint.setColor(Color.WHITE);
        canvas.drawPaint(paint);
    }
}
```

```

//Vẽ text "LẬP TRÌNH GIAO DIỆN MỨC THẤP"
paint.setColor(Color.MAGENTA);
paint.setTextSize(60);
canvas.drawText("LẬP TRÌNH GIAO DIỆN MỨC THẤP", 60, 80, paint);

// Hiển thị kích thước màn hình
paint.setColor(Color.BLUE);
paint.setTextSize(50);
String Wcanvas = Integer.toString(xcanvas);
canvas.drawText("Chiều rộng của màn hình =" + Wcanvas, 80, 160, paint);
String Hcanvas = Integer.toString(ycanvas);
canvas.drawText("Chiều cao của màn hình =" + Hcanvas, 80, 240, paint);

// Khai báo 1 hình chữ nhật để vẽ cung trong đó
RectF rectF = new RectF(30, ycanvas/4-40, xcanvas/2-60, ycanvas/2+120);
paint.setColor(Color.WHITE);
canvas.drawOval(rectF, paint); //Hình Oval nằm trong hình chữ nhật chứa cung sắp vẽ
paint.setColor(Color.RED);
paint.setStyle(Paint.Style.FILL);
canvas.drawArc(rectF, -135, 90, true, paint); // Vẽ cung

//Vẽ circle
paint.setColor(Color.RED);
paint.setStyle(Paint.Style.STROKE);
paint.setStrokeWidth(10);
canvas.drawCircle(xcanvas/2+60, ycanvas/4+100, xcanvas/8, paint);

//Vẽ line
paint.setColor(Color.BLACK);
paint.setStyle(Paint.Style.STROKE);
paint.setStrokeWidth(10);
canvas.drawLine(xcanvas/2+xcanvas/4, ycanvas/4-40, xcanvas-60, ycanvas/2-120, paint);

//Vẽ hình chữ nhật đầy màu
paint.setColor(Color.BLUE);
paint.setStyle(Paint.Style.FILL);
canvas.drawRect(20, ycanvas/4+350, canvas.getWidth()-20, canvas.getHeight()-20, paint);

// Vẽ rotated text "Seahorse"
paint.setColor(Color.RED);
canvas.rotate(-45, xcanvas/4, ycanvas/4+450); // Quay canvas -45 độ
paint.setStyle(Paint.Style.FILL);
paint.setTextSize(100);
canvas.drawText("Seahorse", xcanvas/4, 3*ycanvas/4+200, paint);
//canvas.restore();
canvas.rotate(45, xcanvas/2, ycanvas/4+450); // Quay canvas trở lại (thay hàm restore)

//Vẽ image
Resources res = this.getResources();
Bitmap bit = BitmapFactory.decodeResource(res, R.drawable.seahorse);
float d=bit.getWidth(); //the width of the bitmap
float xstar=xcanvas/8; //the coordinates of the left edge of the rectangle
float ystar=3*ycanvas/4; //the coordinates of the top edge of the rectangle
RectF rectF2 = new RectF(xstar, ystar, xstar+d, ystar+d); //defined rectangle

```



```

        canvas.drawBitmap(bit,null,rectF2,paint);
    }
}

```

### Khai báo lớp VeCoBan trong MainActivity

Để chương trình được thực thi, cần phải khai báo và gọi lớp VeCoBan trong lớp Activity :

```

package com.example.b2_1;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main); // Tạm thời cho lệnh này thành chú giải
        VeCoBan ve = new VeCoBan(this);
        setContentView(ve);
    }
}

```

### Bước 3: Xây dựng lớp CLASS2 và chạy thử

**CLASS2:** Lập trình cho đối tượng di chuyển: sử dụng hai ảnh có sẵn (có thể tải về hình các quả cầu **internet.png** và **network.png** từ internet) để lập trình vẽ đối tượng (ví dụ ảnh internet.png) di chuyển ngẫu nhiên. Nếu đối tượng chạm vào cạnh của màn hình thì đổi chiều chuyển động đồng thời đổi sang ảnh khác (ví dụ ảnh networking).

#### Các bước thực hiện:

- (1) Chép 2 tập tin ảnh định dạng png vào thư mục **res\drawable** của project (chú ý: tên tập tin mở đầu bằng chữ thường)..
- (2) Xây dựng một **lớp kế thừa lớp View** để vẽ đối tượng chuyển động như mô tả ở trên (Trong ví dụ này được đặt tên là **ChuyenDong**).
- (3) Khai báo lớp **ChuyenDong** trong MainActivity.
- (4) Chạy thử và debug.

**Xây dựng lớp ChuyenDong:** Thực hiện các bước tương tự như trong Bài 1. Mã nguồn của lớp này như sau (*SV cần tìm hiểu kỹ các câu lệnh, có chú giải trực tiếp trong chương trình, nên nhập từng lệnh thay vì copy và paste*):

```

package com.vn.doanhoaminh.buoi_2;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;

```

Giáo trình lập trình cho thiết bị di động

```

import android.graphics.Paint;
import android.view.View;

public class ChuyenDong extends View{
    private float ballSpeedX = 4; // Tốc độ chuyển động.
    private float ballSpeedY = 7;
    private float x = 10; //Toạ độ ban đầu.
    private float y = 10;
    private float diameter; // Đường kính
    Paint paint = new Paint();
    Bitmap bitmap=BitmapFactory.decodeResource(getResources(), R.drawable.network);
    public ChuyenDong(Context context) {
        super(context);
    } //lệnh constructor
    @Override
    protected void onDraw(Canvas canvas) {
        canvas.drawPaint(paint);
        canvas.drawBitmap(bitmap, x, y, paint );
        diameter = bitmap.getWidth();
        update();
        invalidate(); // Buột vẽ lại.
    }
    private void update() {
        // Lấy toạ độ mới (x,y) để vẽ quả địa cầu chuyển động ngẫu nhiên.
        x += ballSpeedX;
        y += ballSpeedY;
        //Nếu chạm biên thì đổi ảnh và đổi chiều chuyển động.
        if (x<=0 || (x>=getWidth()-diameter)) {
            ballSpeedX = -ballSpeedX;
            bitmap=BitmapFactory.decodeResource(getResources(), R.drawable.network);
        }
        if (y<=0 || (y>=getHeight()-diameter)) {
            ballSpeedY = -ballSpeedY;
            bitmap=BitmapFactory.decodeResource(getResources(), R.drawable.internet);
        }
    }
}

```

### Khai báo lớp ChuyenDong trong Activity chính

Để chương trình được thực thi, cần phải khai báo và gọi lớp ChuyenDong trong lớp MainActivity. Mã của lớp MainActivity hoàn chỉnh như sau:

```

package com.example.b2_2;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.os.Bundle;

public class B2_2_MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main); // Tạm thời cho lệnh này thành chú giải
        View chuyenDong=new ChuyenDong(this);
        setContentView(chuyenDong);
    }
}

```

Giáo trình lập trình cho thiết bị di động

**Ghi chú:** Chưa cần xây dựng Layout với tập tin XML, và cũng không cần điều chỉnh tập tin XML Layout mẫu của ADT. Vì ứng dụng sẽ sửa dụng đối tượng canvas để vẽ trên đó.

### Giải thích thêm về lớp ChuyenDong:

Trong lớp MoveView ta xây dựng phương thức pudate() với 2 chức năng sau:

- Cập nhật toạ độ của đối tượng với vận tốc cho trước với các lệnh:

$x += \text{ballSpeedX}$ ;  $\text{ballSpeedX}$  là tốc độ chuyển động theo phương X.

$y += \text{ballSpeedY}$ ;  $\text{ballSpeedY}$  là tốc độ chuyển động theo phương Y.

- Thay đổi ảnh khi đối tượng chạm vào biên của canvas. Nếu  $x \leq 0$  có nghĩa là quả cầu chạm biên trái và nếu  $x \geq$  (chiều rộng của màn hình – đường kính của quả cầu) có nghĩa là quả cầu chạm biên phải. Khi đó, đổi chiều chuyển động ngang (vận tốc ngang đổi dấu) và chọn lại đối tượng là ảnh network.png. Tương tự, Nếu  $y \leq 0$  có nghĩa là quả cầu chạm biên trên và nếu  $y \geq$  (chiều cao của màn hình – đường kính của quả cầu) có nghĩa là quả cầu chạm biên dưới. Khi đó, đổi chiều chuyển động dọc (vận tốc dọc đổi dấu) và chọn lại đối tượng là ảnh internet.png.
- Phương thức khởi tạo đối tượng Canvas và được ghi đè để: khai báo đối tượng Paint, khai báo đối tượng Bitmap; gọi phương thức update để cập nhật toạ độ vẽ đối tượng, chiều chuyển động và hình ảnh của đối tượng, gọi phương thức invalidate() để vẽ lại làm cho đối tượng chuyển động.

### Bước 4: Xây dựng lớp CLASS3 và chạy thử

#### Class 3: Xử lý sự kiện tương tác của người dùng

Lập trình phát triển một ứng dụng như sau: vẽ một ảnh có sẵn ra màn hình, chẳng hạn một quả bóng, khi người dùng chạm ngón tay vào điểm nào trên màn hình thì quả bóng sẽ chuyển đến điểm đó và khi ngón tay chạm-giữ và di chuyển (rê) trên màn hình thì quả bóng sẽ di chuyển theo. Đồng thời khi ngón tay chạm xuống thì sẽ hiện dòng chữ “ACTION Down”, khi ngón tay rê đi thì hiện dòng chữ “ACTION Move”, khi ngón tay nhấc khỏi màn hình thì hiện dòng chữ “ACTION Up”.

#### Hướng dẫn:

- (1) Tạo một lớp java kế thừa lớp View, đặt tên là **TuongTac.java** thực hiện chức năng đã được mô tả như trên.

Mã nguồn hoàn chỉnh của lớp PanelTouch.java như sau:

```
package com.vn.doanhoaminh.buoi_2;

import android.view.View;
import android.content.Context;
import android.graphics.*;
import android.view.MotionEvent;

public class TuongTac extends View {
    public final Paint paint = new Paint();
    public final Canvas c = new Canvas();
    public Bitmap bitmap;
```

Giáo trình lập trình cho thiết bị di động

```

public float x=50;
public float y=50;
public int duongkinh;
public String st=" Begin test touch";
public TuongTac(Context context) {
    super(context);
    // TODO Auto-generated constructor stub
}
@Override
protected void onDraw(Canvas c) {
    // TODO Auto-generated method stub
    super.onDraw(c);
    paint.setColor(Color.WHITE);
    c.drawPaint(paint);
    paint.setColor(Color.BLUE);
    Bitmap bitmap = BitmapFactory.decodeResource(getResources(),
R.drawable.network);
    c.drawBitmap(bitmap, x, y, paint);
    duongkinh=bitmap.getWidth();
    paint.setTextSize(60);
    c.drawText(st, x, y, paint);
}

public boolean onTouchEvent(MotionEvent event) {
    switch (event.getAction()) {
        case MotionEvent.ACTION_DOWN:
            x = event.getX() - (duongkinh/2);
            y = event.getY() - (duongkinh/2);
            st= "ACTION Down";
            break;
        case MotionEvent.ACTION_MOVE:
            x = event.getX() - (duongkinh/2);
            y = event.getY() - (duongkinh/2);
            st="ACTION Move";
            break;
        case MotionEvent.ACTION_UP:
            x = event.getX() - (duongkinh/2);
            y = event.getY() - (duongkinh/2);
            st="ACTION Up";
            break;
    }
    invalidate();
    return true;
}
}

```

### Giải thích thêm về lớp TuongTac.java :

(x,y)



Toạ độ (x,y) của vị trí ngón tay chạm vào màn hình cảm ứng được lấy bởi các hàm getX() và getY(). Quả bóng được vẽ trong một khung chữ nhật có cạnh bằng đường kính của nó và toạ độ của góc trái trên là (x,y), xem hình 3.3. Vì vậy khi điểm chạm của ngón tay nằm sát cạnh phải hoặc cạnh đáy của màn hình thì quả bóng sẽ bị khuất không còn nhìn thấy. Để

tránh hiện tượng này ta phải chỉnh lại tọa độ vẽ quả bóng bằng cách lấy tọa độ ngón tay trừ đi bán kính quả cầu. Vì vậy tọa độ vẽ quả bóng được xác định như sau:

$$x = \text{event.getX()} - (\text{duongkinh}/2);$$

$$y = \text{event.getY()} - (\text{duongkinh}/2);$$

- (2) Khai báo lớp **TuongTac.java** trong **Activity** chính và chạy thử (tương tự như các mục trên).

### Bước 5: Xây dựng lớp CLASS4 và chạy thử

#### Class4: Lập trình luồng với lớp SurfaceView

Phát triển một ứng dụng game đơn giản như sau: một đối tượng hình ảnh (chẳng hạn hình con rệp có tên tập tin là **bug.png**) di chuyển ngẫu nhiên trên canvas (với một hình nền tùy chọn), xem hình 2.4. Khi người dùng chạm vào đối tượng bug thì nó sẽ biến mất rồi xuất hiện lại ở vị trí ban đầu và tiếp tục di chuyển, khi đó người chơi được cộng một điểm. Sau 15 giây nếu từ 5 điểm trở lên thì hiển thị kết quả cho biết đã thắng với số điểm đạt được (trong một màn hình mới) như hình 2.5, ngược lại hiển thị kết quả thua như hình 2.6.

### Hướng dẫn

- (1) Tạo ra 3 Java Class có tên là **GamePanel**, **ThreadView**, **Bug** trên cùng gói tên miền của ứng dụng (trong bước này, ta chỉ khai báo tên lớp, lớp được kế thừa và phương thức khởi tạo, sẽ lập trình hoàn thành trong các bước sau). Trong đó, **GamePanel** là lớp chính thực hiện game, hiển thị giao diện của game (mức thấp) và cho phép người chơi tương tác. Lớp **ThreadView** và **Bug** hỗ trợ cho các hoạt động khi lớp **GamePanel** được khởi chạy. Lớp **ThreadView** tạo ra 1 luồng và 1 bề mặt để lớp **GamePanel** thể hiện trong thời gian chạy. Lớp **Bug** được tạo ra để “vẽ con bọ” và hình nền trên canvas.
- (2) Chép tập tin ảnh **bug.png** và tập tin **background.png** (tìm trên trang web <https://elcit.ctu.edu.vn/enrol/index.php?id=2444> hoặc trên mạng tập tin ảnh tương tự, định dạng png) vào một trong các thư mục **drawable** (ví dụ res\drawable).
- (3) Lập trình lớp **ThreadView**. SV tham khảo mã nguồn của lớp này như sau:

```
package com.vn.doanhoaminh.buoi_2;
import android.graphics.*;
import android.view.*;

public class ThreadView extends Thread{
    private GamePanel mpanel; // Khai báo 1 thể hiện của lớp Gamepanel
    private SurfaceHolder mholder; // Khai báo 1 đối tượng quản lý giao diện mức thấp
    private boolean mrun = false; // Khai báo biến điều kiện cho vòng lặp while

    //Constructor
    public ThreadView (GamePanel panel){
        mpanel = panel;
        mholder = mpanel.getHolder();
    }
}
```

```

    }
    public void setRunning(boolean run){
        mrun = run;
    }
    @Override
    public void run() {
        Canvas canvas = null;
        while (mrun){
            canvas = mholder.lockCanvas(); // Hàm lockCanvas () tạo một canvas để vẽ
            if(canvas != null)
            {
                mpanel.doDraw(canvas); //Thực hiện hành động trên canvas (hiển thị)
                mholder.unlockCanvasAndPost(canvas); //Hoàn thành việc vẽ trên canvas
            }
        }
    }
}

```

(4) Lập trình lớp **Bug** với các chức năng vẽ đối tượng chuyển động như mô tả trong đề bài, SV tham khảo mã nguồn của lớp này như sau:

```

package com.vn.doanhoaminh.buoi_2;
import android.content.Context;
import android.content.res.Resources;
import android.graphics.*;

public class Bug {
    Bitmap background;
    Bitmap bitmap;
    int dia;
    int x = 70;
    int y = 70;
    int w, h;
    int sX = 30;
    int sY = 30;
    public Bug(Context context){
        Resources res = context.getResources();
        bitmap = BitmapFactory.decodeResource(res,R.drawable.bug);
        background = BitmapFactory.decodeResource(res, R.drawable.background);
    }
    public void onDraw(Canvas c)
    {
        Paint paint = new Paint();
        paint.setColor(Color.BLACK);
        dia = bitmap.getWidth();
        w = c.getWidth();
        h = c.getHeight();
    }
}

```



```

        c.drawPaint(paint);
        c.drawBitmap(background, 0, 0, paint);
        c.drawBitmap(bitmap, x, y, paint);
        x += sX;
        y += sY;
        //Neu cham bien thi quay lai va doi Ball
        if (x<=0 || (x>=w- dia)) {
            sX = -sX;
        }
        if (y<=0 || y>=h-dia) {
            sY = -sY;
        }
    }
}

```

(5) Lập trình hoàn thành lớp **GamePanel**, SV tham khảo mã nguồn của lớp này như sau:

```

package com.vn.doanhoaminh.buoi_2;

import android.annotation.SuppressLint;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Paint.Align;
import android.view.MotionEvent;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

public class GamePanel extends SurfaceView implements SurfaceHolder.Callback{
    Bitmap bitmap;
    int count = 20;
    Time time = new Time();
    ThreadView thread;
    int dia;
    int kq = 0;
    int x, y;
    int X;
    int Y;
    Bug bug;
    float dis;

    public GamePanel(Context context){
        super(context);
        bug = new Bug(context);
        bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.bug);
    }
}

```

Giáo trình lập trình cho thiết bị di động

```

        //touched = MediaPlayer.create(context, R.raw.touched);
        dia = bitmap.getWidth();
        getHolder().addCallback(this);
        thread = new ThreadView(this);
        time.start();
    }
    @SuppressWarnings("WrongCall")
    public void doDraw(Canvas c){
        if(count > 0){
            bug.onDraw(c);
            Paint tpaint = new Paint();
            tpaint.setColor(Color.BLACK);
            c.drawText("Time: " + count, 10, 10, tpaint);
        }
        else if (count <= 0){
            if(kq >= 5){
                Paint spaint = new Paint();
                spaint.setColor(Color.BLACK);
                c.drawPaint(spaint);
                spaint.setColor(Color.WHITE);
                spaint.setTextSize(30);
                spaint.setTextAlign(Align.CENTER);
                c.drawText("YOU WIN!", c.getWidth()/2, c.getHeight()/2 - 60, spaint);
                c.drawText("Record: " + kq, c.getWidth()/2, c.getHeight()/2, spaint);
                spaint.setTextAlign(Align.CENTER);
                c.drawText("touch to quit", c.getWidth()/2, c.getHeight()/2 + 60, spaint);
            }
            else if(kq < 5){
                Paint spaint = new Paint();
                spaint.setColor(Color.BLACK);
                c.drawPaint(spaint);
                spaint.setColor(Color.WHITE);
                spaint.setTextSize(30);
                spaint.setTextAlign(Align.CENTER);
                c.drawText("YOU LOSE!", c.getWidth()/2, c.getHeight()/2 - 60, spaint);
                c.drawText("Record: " + kq, c.getWidth()/2, c.getHeight()/2, spaint);
                spaint.setTextAlign(Align.CENTER);
                c.drawText("touch to quit", c.getWidth()/2, c.getHeight()/2 + 60, spaint);
            }
        }
    }

    public void surfaceChanged(SurfaceHolder holder, int format, int width,
        int height) {
        // TODO Auto-generated method stub
    }

    public void surfaceCreated(SurfaceHolder holder) {
        // TODO Auto-generated method stub
        if(!thread.isAlive()){

```

```

        thread = new ThreadView(this);
        thread.setRunning(true);
        thread.start();
    }
}

public void surfaceDestroyed(SurfaceHolder holder) {
    // TODO Auto-generated method stub
    if(thread.isAlive()){
        thread.setRunning(false);
    }
}

public class Time extends Thread {
    @Override
    public void run(){
        while (count > 0){
            try{
                count--;
                sleep(1000);
            }

            catch (InterruptedException e){
                e.printStackTrace();
            }
        }
    }

    public boolean onTouchEvent(MotionEvent event){
        int eventaction = event.getAction();
        X = (int)event.getX();
        Y = (int)event.getY();
        switch (eventaction){
            case MotionEvent.ACTION_DOWN:{
                int centerX = bug.x + dia;
                int centerY = bug.y + dia;
                dis = (float) Math.sqrt((((centerX-X)*(centerX-X)) + (centerY-Y)*(centerY-Y)));
                if(count > 0){
                    if(dis < dia){
                        kq++;
                        bug.x = 100;
                        bug.y = 100;
                    }
                }
                if(count <= 0)
                {
                    System.exit(0);
                }
            }
        }
    }
}

```

```

        default:
        break;
    }
    return super.onTouchEvent(event);
}
}

```

(6) Khai báo lớp **GamdPanel** trong lớp **MainActivity** và chạy thử

### Bước 6: Lập trình hoàn chỉnh MainActivity (Nội dung 5)

(1) Lập trình giao diện, mở tập tin activity\_main.xml và lập trình với mã nguồn như sau:

```

<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:id="@+id/framelayout">
        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:gravity="center"
            android:text="BÀI THỰC HÀNH BUỔI 2"
            android:textColor="#00f"
            android:textSize="24sp"
            android:textStyle="bold" />

    </FrameLayout>

    <TableLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:gravity="center_horizontal">
        <TableRow>
            <Button
                android:id="@+id/class1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:text="Class1"
                android:textSize="12sp" />

            <Button
                android:id="@+id/class2"

```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Class2"
        android:textSize="12sp" />
    <Button
        android:id="@+id/class3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Class3"
        android:textSize="12sp" />
    <Button
        android:id="@+id/class4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Class4"
        android:textSize="12sp" />
</TableRow>
<TableRow>
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="THOÁT"
        android:textSize="12sp" />
    <TextView />
    <TextView />
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="TRỞ VỀ"
        android:textSize="12sp" />
</TableRow>
</TableLayout>
</LinearLayout>

```

(2) Mở tập tin MainActivity và lập trình tương tác người dùng lên các button với mã nguồn như sau:

```

package com.example.b3_minh_b1234567;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.SurfaceView;
import android.view.View;
import android.widget.Button;

```

Giáo trình lập trình cho thiết bị di động

```

import android.widget.FrameLayout;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(final Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final FrameLayout hienthi = (FrameLayout)findViewById(R.id.framelayout);
        Button bai1 = (Button) findViewById(R.id.class1);
        Button bai2 = (Button) findViewById(R.id.class2);
        Button bai3 = (Button) findViewById(R.id.class3);
        Button bai4 = (Button) findViewById(R.id.class4);
        Button thoat = (Button)findViewById(R.id.button1);
        Button trove = (Button)findViewById(R.id.button2);
        final VeCoBan ve = new VeCoBan(this);
        final ChuyenDong chuyenDong = new ChuyenDong(this);

        bai1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                hienthi.addView(ve);
            }
        });

        bai2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                hienthi.addView(chuyenDong);
            }
        });

        final TuongTac tuongTac = new TuongTac(this);
        bai3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                hienthi.addView(tuongTac);
            }
        });

        final SurfaceView view = new GamePanel(this);
        bai4.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                hienthi.addView(view);
            }
        });
    }
}

```



```
thoat.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        System.exit(0);  
    }  
});  
  
trove.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        startActivity(new Intent(getApplicationContext(), MainActivity.class));  
    }  
});  
}  
}
```

(3) Chạy thử trên máy ảo.