# Touch

## Mục tiêu

Nhận biết được sự kiện chạm trên View

Sử dụng sự kiện

## Ví dụ Single touch

1. Tạo ra một class mới tên Single Touch và cho lớp này kế thừa từ View, sau đó import class và phát sinh hàm tạo.

```
public class SingleTouch extends View{
	public SingleTouch(Context context) {
		super(context);
	}
}
```

2. Trong class vừa tạo khai báo 2 biến toàn cục là Paint (bút vẽ) và Path (đường vẽ)

```
Paint paint=new Paint();
Path path=new Path();
```

3. Trong hàm tạo khởi tạo giá trị cho biến Paint như sau:

```
paint.setAntiAlias(true);//khu rang cua
paint.setStrokeWidth(6f);//do day
paint.setColor(Color.RED);//gan mau
paint.setStyle(Paint.Style.STROKE);
paint.setStrokeJoin(Paint.Join.ROUND);
```

4. Override lên hàm onDraw để vẽ.

```
@Override
protected void onDraw(Canvas canvas) {
	super.onDraw(canvas);
	canvas.drawPath(path, paint);
}
```

5. Override lên hàm *onTouchEvent*. Trong hàm đầu tiên lấy ra tọa độ X,Y chính là vị trí chạm. Sau đó xét nếu chạm tay (down) thì nhảy paint đến vị trí đó, nếu di chuyển (move) vẽ từ vị trí trước đến vị trí move, nhả tay ra (up) thì không làm gì

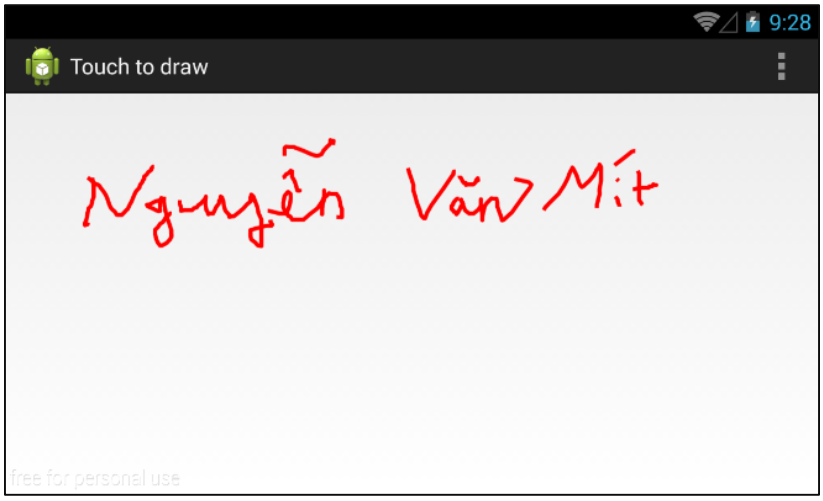
nữa. Cuối cùng gọi invalidate() để vẽ lại (chú ý hàm invalidate và postInvalidate). Cuối cùng nhớ trả về true (bỏ trả về mặc định đi).

```
@Override
public boolean onTouchEvent(MotionEvent event) {
	float eventX=event.getX();
	float eventY=event.getY();
	switch(event.getAction())
	{
	case MotionEvent.ACTION_DOWN:
		path.moveTo(eventX,eventY);
		break;
	case MotionEvent.ACTION_UP:
		break;

	case MotionEvent.ACTION_MOVE:
		path.lineTo(eventX,eventY);
		break;
	default : return false;

	}
	invalidate();
	return true;//super.onTouchEvent(event);
}
```

6. Quay lại file java chính. Trong hàm onCreate sửa lại hàm setContextView như sau:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //setContentView(R.layout.activity_main);
    SingleTouch st=new SingleTouch(this);
    setContentView(st);
}
```

7. Chạy ứng dụng, nhấn và rê chuột




## Viết chương trình vẽ và duy chuyển vòng tròn khi chạm các ngón tay (Multitouch)

Tạo lớp MultiTouch như sau:

1. Khai báo các biến toàn cục gồm hai ArrayList để chứa tập tọa độ x,y của các ngón tay khi chạm và một ArrayList isTouch để chứa trạng thái touch.

```
Paint paint;
Path path;
ArrayList<Float> x;
ArrayList<Float> y;
ArrayList<Boolean> isTouch;
```

2. Hàm *onTouchEvent* (xem thêm ghi chú bên trong hàm): ta phải chú ý các diểm sau:
   - Lớp **MotionEventCompat** nhưng thư viện hỗ trợ là android.support.v4.view.MotionEventCompat;
   - Để đơn giản chúng ta sử dụng hàm *getActionIndex* để lấy index của sự kiện index là vị trí của các ngón tay.

     Ví dụ: chạm 1 ngón chúng ta có index=0, chạm thêm ngón thứ 2 index=1.....
   - Mỗi lần chạm lấy dữ liệu tọa độ gán vào biến istouch, khi di chuyển chúng ta phải cập nhật lại tọa độ cho các điểm chạm.
   - Mỗi lần chạm xong thì sự kiện UP cập nhật lại index, nên in luôn được cập nhật khi bỏ chạm.

```
@Override
public boolean onTouchEvent(MotionEvent event) {
int action = MotionEventCompat.getActionMasked(event);
```

```
int index = MotionEventCompat.getActionIndex(event);
switch(action)
{
	case MotionEvent.ACTION_DOWN:
		istouch.add(true);
		x.add(event.getX(index));
		y.add(event.getY(index));
		break;
	case MotionEvent.ACTION_POINTER_DOWN:
		istouch.add(true);
		x.add(event.getX(index));
		y.add(event.getY(index));
		break;

	case MotionEvent.ACTION_MOVE:
		int count=event.getPointerCount();
		for(int i=0;i<count;i++)
		{
			index=i;
			x.set(index,event.getX(index));
			y.set(index,event.getY(index));
		}
		break;

	case MotionEvent.ACTION_UP:
		istouch.remove(index);
		x.remove(index);
		y.remove(index);
		break;
	case MotionEvent.ACTION_POINTER_UP:
		istouch.remove(index);
		x.remove(index);
		y.remove(index);
		break;

	default: return false;
}
invalidate();
// bao co thay doi va yeu cau ve lai view
//neu ve lai view tu UI-thread dung  invalidate()
//neu ve lai tu non-UI Thread dung view.postInvalidate()
return true;
```

```
        //return super.onTouchEvent(event);
}
```

3. Cuối cùng là hàm onDraw. Hàm này được chạy liên tục mỗi lần chạy sẽ duyệt tất cả các vị trí chạm đã được lưu trữ và vẽ hình tròn tại vị trí đó.

```
@Override
protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);
        for(int i=0;i<x.size();i++)
                if(istouch.get(i)==true)
                        canvas.drawCircle(x.get(i), y.get(i),50f, paint);
}
```

4. Chạy lại ứng dụng chạm 3 ngón tay và rê