1. **View subclassing**

**\*Khái niệm:**

"View subclassing" trong Android là kỹ thuật tạo lớp con tùy chỉnh từ lớp View hoặc các View con (như TextView, ImageView, Button…) để mở rộng hoặc thay đổi hành vi mặc định của các thành phần giao diện.

**\*Sử dụng subclass View khi:**

+ Muốn vẽ giao diện tùy chỉnh (custom drawing).

+ Cần xử lý touch riêng biệt (gesture hoặc thao tác đặc biệt).

+ Cần tạo một View đặc biệt mà không có sẵn trong Android.

+ Tái sử dụng một thành phần giao diện có logic riêng.

|  |  |
| --- | --- |
| **CODE** | **RUN** |
| MainActivity.java package com.example.test;  import android.os.Bundle; import androidx.appcompat.app.AppCompatActivity;  public class MainActivity extends AppCompatActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  } } | **Sau click** |
| MyCustomView.javapackage com.example.test;  import android.content.Context; import android.graphics.Canvas; import android.graphics.Color; import android.graphics.Paint; import android.util.AttributeSet; import android.view.MotionEvent; import android.view.View;  public class MyCustomView extends View {   private Paint paint;   public MyCustomView(Context context) {  super(context);  init();  }   public MyCustomView(Context context, AttributeSet attrs) {  super(context, attrs);  init();  }   private void init() {  paint = new Paint();  paint.setColor(Color.*RED*);  }   @Override  protected void onDraw(Canvas canvas) {  super.onDraw(canvas);  canvas.drawCircle(getWidth() / 2f, getHeight() / 2f, 100, paint);  }   @Override  public boolean onTouchEvent(MotionEvent event) {  if (event.getAction() == MotionEvent.*ACTION\_DOWN*) {  paint.setColor(Color.*GREEN*);  invalidate(); *// vẽ lại* return true;  }  return false;  }  } |
| Activity\_main.xml <FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">   <com.example.test.MyCustomView  android:layout\_width="200dp"  android:layout\_height="200dp"  android:layout\_gravity="center"  android:background="#CCCCCC" /> </FrameLayout> |

1. **Tập tin trên di động**

**1. Khái niệm**

Tập tin trong môi trường di động là đơn vị lưu trữ dữ liệu do hệ điều hành quản lý. Trong lập trình di động, tập tin được dùng để:

+ Lưu dữ liệu người dùng (văn bản, hình ảnh, âm thanh…);

+ Lưu thiết lập cấu hình;

+ Lưu tài nguyên cục bộ phục vụ cho ứng dụng.

**\*Đặc điểm của tập tin trên di động:**

+ Thường được lưu nội bộ (internal storage) hoặc ngoại vi (external storage – thẻ nhớ).

+ Bị sandbox hóa: Mỗi ứng dụng chỉ được truy cập các tập tin của chính nó, trừ khi có cấp quyền đặc biệt.

+ Tối ưu bộ nhớ: Hệ điều hành di động thường giới hạn dung lượng, vì vậy quản lý file hiệu quả rất quan trọng.

## ****2. Các loại tập tin trên di động****

Tập tin trên thiết bị di động có thể được chia thành 2 loại lớn dựa trên chức năng và vị trí lưu trữ:

### A. ****Phân loại theo chức năng****

| **Loại tập tin** | **Mô tả** |
| --- | --- |
| **Tập tin cấu hình** | Lưu các thông số cấu hình (ví dụ: SharedPreferences dạng XML) |
| **Tập tin dữ liệu** | Lưu trữ thông tin như ghi chú, danh sách người dùng, v.v. |
| **Tập tin đa phương tiện** | Hình ảnh, video, âm thanh, tài liệu được ứng dụng sử dụng hoặc tạo ra |
| **Tập tin cache** | Dữ liệu tạm thời để tăng tốc hoạt động ứng dụng |
| **Tập tin cơ sở dữ liệu** | Dạng như SQLite (\*.db), lưu trữ dữ liệu có cấu trúc |

### B. ****Phân loại theo vị trí lưu trữ****

#### 1. ****Internal Storage (Bộ nhớ trong)****

Tập tin chỉ có ứng dụng đó truy cập được.

An toàn hơn và tự động xóa khi gỡ cài đặt app.

#### 2. ****External Storage (Bộ nhớ ngoài – thẻ nhớ hoặc bộ nhớ dùng chung)****

Có thể chia sẻ giữa các ứng dụng (cần quyền truy cập).

Ví dụ: lưu ảnh chụp từ camera, tải file từ mạng.

## III. ****Mô hình Networking trên di động (Client - Server)****

### ****1. Khái niệm****

Mô hình **Client - Server** là mô hình mạng trong đó:

**+ Client** (ứng dụng di động) gửi yêu cầu (request) đến **Server** (máy chủ).

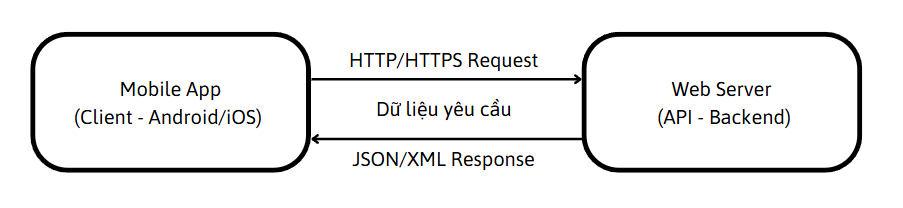
**+ Server** xử lý yêu cầu và trả về kết quả (response) cho client.

**\*Trên di động, mô hình này được sử dụng để:**

**+** Gửi và nhận dữ liệu qua Internet.

+ Giao tiếp với các API (RESTful API).

+ Đồng bộ dữ liệu, xác thực người dùng, lấy dữ liệu từ server (như sản phẩm, thời tiết...).



| **Thành phần** | **Vai trò** |
| --- | --- |
| **Client (App di động)** | Gửi yêu cầu (thường là HTTP GET, POST) đến server, xử lý phản hồi. |
| **Server (Web/API)** | Nhận request, xử lý logic (truy xuất CSDL, xác thực, v.v.), trả về kết quả. |
| **Giao thức truyền thông** | Thường dùng: **HTTP/HTTPS** |
| **Định dạng dữ liệu** | Phổ biến nhất là **JSON** (hoặc XML cũ hơn) |