TRƯỜNG ĐẠI HỌC THỦY LỢI

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

Description automatically generated with low confidence

**BÁO CÁO MÔN HỌC**

PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG

**ĐỀ TÀI: APP QUẢN LÝ CHI TIÊU**

Giáo viên hướng dẫn: Phạm Huy Thông

Sinh viên thực hiện: Nguyễn Hoàng Sơn

Nguyễn Thành Hưng

Trần Thanh Thảo - 2051063511

Trần Việt Phương

Vũ Thanh Sơn

Lớp học phần: 62TH-NB

Nhóm thực hiện: Nhóm 9

**Hà Nội, tháng 6 năm 2023**

**MỤC LỤC**

[**CHƯƠNG 1: GIỚI THIỆU** 2](#_Toc138499503)

[**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT** 4](#_Toc138499504)

[1. Các component chính trong android 4](#_Toc138499505)

[1.1 Activity 4](#_Toc138499506)

[1.2 Content Provider 4](#_Toc138499507)

[2. Các thành phần giao diện được sử dụng trong ứng dụng 5](#_Toc138499508)

[2.1 View 5](#_Toc138499509)

[2.2 LinearLayout 5](#_Toc138499510)

[2.3 FrameLayout 6](#_Toc138499511)

[2.4 Button 6](#_Toc138499512)

[2.5 ImageView 8](#_Toc138499513)

[2.6 Listview 8](#_Toc138499514)

[2.7 TextView 9](#_Toc138499515)

[2.8 EditText 9](#_Toc138499516)

[2.9 Activity và Intent 10](#_Toc138499517)

[3. Thành phần database 13](#_Toc138499518)

# **CHƯƠNG 1: GIỚI THIỆU**

Trong thế giới ngày nay, quản lý chi tiêu cá nhân là một nhiệm vụ quan trọng để đảm bảo tài chính cá nhân ổn định và đạt được mục tiêu tài chính. Tuy nhiên, việc theo dõi và kiểm soát chi tiêu hàng ngày có thể trở nên phức tạp và đòi hỏi sự tỉ mỉ và đầu tư thời gian. May mắn thay, sự phát triển của công nghệ đã mang lại những giải pháp hiệu quả và tiện lợi cho việc quản lý chi tiêu cá nhân thông qua các ứng dụng di động.

Trong báo cáo này, nhóm chúng em xin giới thiệu một ứng dụng quản lý chi tiêu tuyệt vời có tên "Quản lý chi tiêu cá nhân". Ứng dụng này không chỉ giúp người dùng theo dõi chi tiêu hàng ngày một cách thuận tiện, mà còn cung cấp nhiều tính năng hữu ích để phân loại, phân tích và quản lý ngân sách cá nhân. Với giao diện người dùng thân thiện và tính năng linh hoạt, Quản lý chi tiêu cá nhân đã trở thành một công cụ quan trọng để giúp người dùng duy trì sự cân đối tài chính và đạt được mục tiêu tài chính cá nhân.

Trong phần tiếp theo của báo cáo, nhóm chúng em sẽ trình bày chi tiết về tính năng và lợi ích của ứng dụng, bao gồm việc theo dõi chi tiêu hàng ngày, quản lý khoản chi/thu, thống kê. Ngoài ra cũng sẽ nhấn mạnh về sự dễ sử dụng và hiệu quả của ứng dụng này trong việc quản lý chi tiêu cá nhân.

# **CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

## **Các thành phần chính trong android**

### **1.1 Activity**

Trong ứng dụng Android, Activity đóng vai trò đặc biệt quan trọng, là nơi giúp người dùng tương tác trực tiếp với ứng dụng, ví dụ như gọi điện thoại, chụp ảnh, gửi e-mail hoặc xem bản đồ.

Activity được coi là xương sống của một ứng dụng Android, một ứng dụng có thể có một hoặc nhiều Activity (bất kì ứng dụng nào cũng cần có ít nhất 1 Activity).

Activity có thể hiển thị ở chế độ toàn màn hình, dạng cửa sổ hoặc với một kích thước nhất định

Một Activity có thể gọi đến một Activity khác, Activity được gọi đến sẽ tương tác với người dùng tại thời điểm được gọi tới.

Một ứng dụng bên ngoài có thể gọi tới bất kỳ Activity nào trong ứng dụng (nếu được cấp quyền). Ví dụ: Một ứng dụng chụp ảnh sau khi chụp ảnh xong, sẽ gửi yêu cầu để start một activity có chức năng soạn e-mail trong ứng dụng email nhằm mục đích gửi ảnh vừa chụp đi.

### **1.2 Content Provider**

Content Provider là một thành phần giúp các một ứng dụng có thể đọc và ghi dữ liệu từ một file hoặc từ SQLite của một ứng dụng khác trong cùng một hệ thống. Bất kỳ ứng dụng nào có quyền (permisson) đều có thể truy xuất, chỉnh sửa dữ liệu của một ứng dụng khác.

Content Provider được chia thành 2 loại:

Native Content Provider: Là những Content Provider có sẵn, được tạo ra bởi hệ thống, ví dụ như Contacts, Message, …

Custom Content Provider: Bao gồm các Content Provider được tạo ra bởi các developer phụ thuộc vào đặc điểm của từng ứng dụng.

### **1.3 Service**

Là một thành phần của ứng dụng các tác vụ chạy ngầm dưới hệ thống nhằm thực hiện một nhiệm vụ nào đó.

**1.4 Broadcast receive**: Để nhận bản tin quảng bá cơ chế phát đi các sự kiện

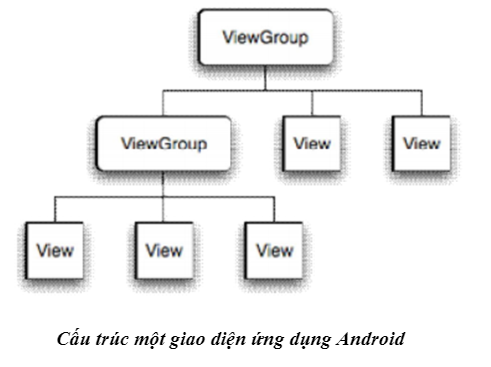
## **Các thành phần giao diện được sử dụng trong ứng dụng**

### **View**

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và ViewGroup. Có nhiều kiểu View và ViewGroup. Mỗi một kiểu là một hậu duệ của class View và tất cả các kiểu đó được gọi là các Widget.

Tất cả mọi widget đều có chung các thuộc tính cơ bản như là cách trình bày vị trí, background, kích thước, lề, … Tất cả những thuộc tính chung này được thể hiện hết ở trong đối tượng View.

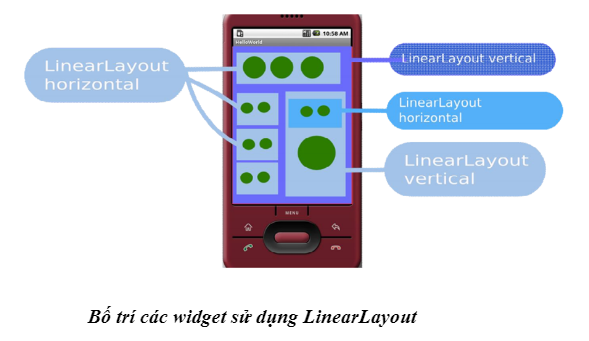
Trong Android Platform, các screen luôn được bố trí theo một kiểu cấu trúc phân cấp như hình dưới. Một screen là một tập hợp các Layout và các widget được bố trí có thứ tự. Để thể hiện một screen thì trong hàm onCreate của mỗi Activity cần phải được gọi một hàm là setContentView(R.layout.main); hàm này sẽ load giao diện từ file XML lên để phân tích thành mã bytecode.



### **LinearLayout**

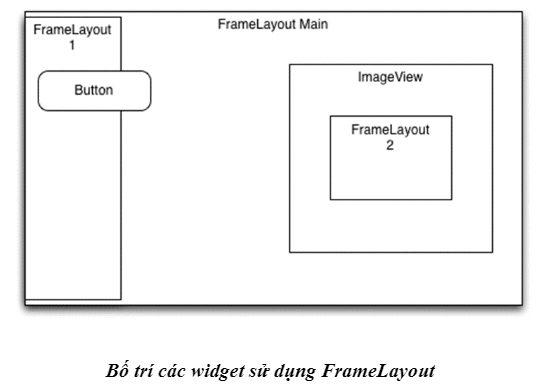
LinearLayout được dùng để bố trí các thành phần giao diện theo chiều ngang hoặc chiều dọc nhưng trên một line duy nhất mà không xuống dòng.

LinearLayout làm cho các thành phần trong nó không bị phụ thuộc vào kích thước của màn hình. Các thành phần trong LinearLayout được dàn theo những tỷ lệ cân xứng dựa vào các ràng buộc giữa các thành phần.



### **FrameLayout**

FrameLayout được dùng để bố trí các đối tượng theo kiểu giống như là các Layer trong Photoshop. Những đối tượng nào thuộc Layer bên dưới thì sẽ bị che khuất bởi các đối tượng thuộc Layer nằm trên. Frame Layer thường được sử dụng khi muốn tạo ra các đối tượng có khung hình bên ngoài chẳng hạn như contact image button.



### **Button**

Sở dĩ widget button được giới thiệu đầu tiên trong số các widget khác là vì đây là đối tượng có thể nói là được dùng nhiều nhất trong hầu hết các ứng dụng Android.

Để thiết kế giao diện với một button ta có 2 cách như sau:

#### **Thiết kế bằng XML**

|  |
| --- |
| <Button       android:id=*"@+id/Button1"*       android:layout\_width=*"wrap\_content"*       android:layout\_height=*"wrap\_content"*       android:onClick=*"clickHere"*       android:text=*"Click Here"* /> |

Thuộc tính android:onClick="clickHere" được dùng để nắm bắt sự kiện click vào button. Khi sự kiện click button xảy ra thì phương thức “touchMe” được khai báo trong thẻ thuộc tính sẽ được gọi. Nếu trường hợp phương thức “touchMe” chưa được khai báo trong file mã nguồn tương ứng thì sẽ phát sinh một exception. Ngược lại, phương thức "clickHere" sẽ nhận được một đối tham biến là đối tượng View nơi đã phát sinh ra sự kiện. Đối tượng View này có thể ép kiểu trực tiếp sang kiểu Button vì thực chất nó là một button.

|  |
| --- |
| public void clickHere (View v){  Button me = (Button) v;  me.setText(“Clicked”);  } |

#### **Thiết kế bằng code**

Nếu không đòi hỏi phải custom lại một widget thì chúng ta không cần phải sử dụng tới code. Trong một số trường hợp bắt buộc chúng ta phải custom các widget để cho phù hợp với hoàn cảnh. Chẳng hạn như trong game, các menu hay các nút điều khiển,…

Ví dụ để khai báo một Button trong code ta làm như sau:

|  |
| --- |
| Button cmdButton = new Button(this);  cmdButton.setText(“Click Here”);  cmdButon.setOnClickListener(…); |

Để custom một widget nào đó ta phải tạo một class kế thừa từ class Widget muốn custom, sau đó sử dụng hàm draw để vẽ lại widget đó như một Canvas.

Ví dụ: canvas.drawPicture(Picture.createFromStream(...));

### **ImageView**

ImageView được dùng để thể hiện một hình ảnh. Nó cũng giống như ImageButton, chỉ khác là không có hình dáng của một cái button.

Ví dụ:

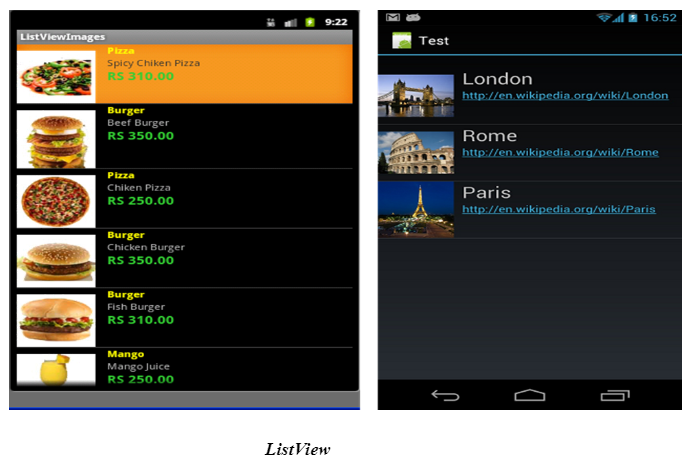
|  |
| --- |
| ImageView iv = new ImageView(this);  iv.setImageResource(R.drawable.icon);  <ImageView  android:id="@+id/ImageView01"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" /> |

### **Listview**

Được sử dụng để thể hiện một danh sách các thông tin theo từng cell. Mỗi cell thông thường được load lên từ một file XML đã được cố định trên đó số lượng thông tin và loại thông tin cần được thể hiện.

Để thể hiện được một list thông tin lên một screen thì cần phải có 3 yếu tố chính:

* **Data Source:** Data Source có thể là một ArrayList, HashMap hoặc bất kỳ một cấu trúc dữ liệu kiểu danh sách nào.
* **Adapter:** Adapter là một class trung gian giúp ánh xạ dữ liệu trong Data Source vào đúng vị trí hiển thị trong ListView. Chẳng hạn, trong Data Source có một trường name và trong ListView cũng có một TextView để thể hiện trường name này. Tuy nhiên, ListView sẽ không thể hiển thị dữ liệu trong Data Source lên được nếu như Adapter không gán dữ liệu vào cho đối tượng hiển thị.
* **ListView:** ListView là đối tượng để hiển thị các thông tin trong Data Source ra một cách trực quan và người dùng có thể thao tác trực tiếp trên đó.



### **TextView**

TextView ngoài tác dụng là để hiển thị văn bản thì nó còn cho phép định dạng nội dung bằng thẻ html.

Ví dụ:

|  |
| --- |
| TextView textView = (TextView) findViewById(R.id.textView);  CharSequence styledText = Html.fromHtml("<i>This</i> is some <b>styled</b> <s>text</s>");  textView.setText(styledText); |

Nội dung TextView cũng có thể được định dạng bằng thẻ html ngay trong XML

### **EditText**

Trong Android đối tượng EditText được sử dụng như một TextField hoặc một TextBox.

|  |
| --- |
| <EditText  android:id="@+id/EditText01"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textStyle="bold"  android:textSize="20dip"  android:textColor="#000000"  android:text="Hello Android!"  android:singleLine="true"  android:inputType="textCapWords"/> |

Các thuộc tính cần chú ý sử dụng EditText đó là:

*android:inputType = "…"* sử dụng để xác định phương thức nhập cho EditText. Chẳng hạn như khi bạn muốn một ô để nhập password hay một ô để nhập Email thì thuộc tính này sẽ làm điều đó.

*android:singleLine = “true”* EditText của bạn sẽ trở thành một TextField, ngược lại sẽ là TextBox.

### **Activity và Intent**

#### Activity

Activity là một thành chính của một ứng dụng Android, được dùng để hiển thị một màn hình và nắm bắt các hoạt động xảy ra trên màn hình đó. Khi làm việc với Activity cần nắm bắt được một số kiến thức cơ bản như sau:

● Chu kỳ sống của một Activity

● Tạo menu và dialog

● Khởi động một Activity

Để khởi động một Activity ta sử dụng Intend sẽ tìm hiểu kỹ hơn ở phần b. Tuy nhiên, trong phần này tôi sẽ hướng dẫn cách chuyển giữa các Intend theo 2 loại:

|  |
| --- |
| Intent intent = new Intent(this, B.class);  startActivity(intent); |

● Khai báo không tường minh

Cung cấp các thao tác cần làm gì với loại dữ liệu nào, hệ thống sẽ tìm đến activity tương ứng để khởi động.

VD: Để xem thông tin một contact nào đó trong Activity của ứng dụng Contact trong Android ta chỉ đến dữ liệu contact và chỉ đến Activity View contact như sau:

|  |
| --- |
| Intent i = new Intent();  i.setAction(Intent.ACTION\_VIEW);  i.setData(Uri.withAppendedPath(  android.provider.Contacts.People.CONTENT\_URI, "1));  startActivity(i); |

a, Tính liên lạc giữa 2 activity

Khi chuyển sang một Activity khác ta có thể gửi kèm dữ liệu trong intend đó như sau:

|  |
| --- |
| intend.putExtra(“key1”, “value1”);  intend.putExtra(“key2”, 23); |

Bên phía Activity được khởi động hay được chuyển đến, có thể lấy dữ liệu được gửi như sau:

|  |
| --- |
| getIntend().getExtra().getString(“key1”);  getIntend().getExtra().getInt(“key2”); |

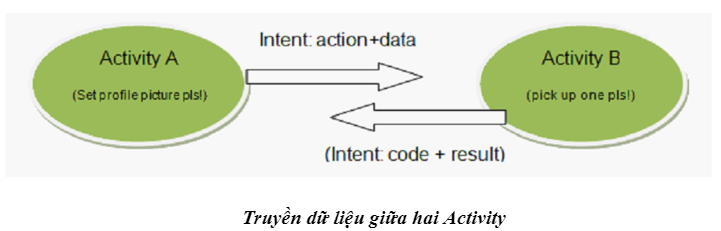
b, Task

Android là một hệ điều hành đa tiến trình. Khi lập trình trên nền tảng Android thì tiến trình là một vấn đề cần phải được chú ý nhiều nhất. Mặc dù Android hỗ trợ đa tiến trình nhưng trên một thiết bị di động với cấu hình thấp mà chúng ta quá lạm dụng tiến trình thì sẽ rất tốn bộ xử lý điều này cũng đồng nghĩa với việc bạn đang biến ứng dụng của bạn trở thành một thứ phần mềm tiêu thụ điện năng.

#### Intent

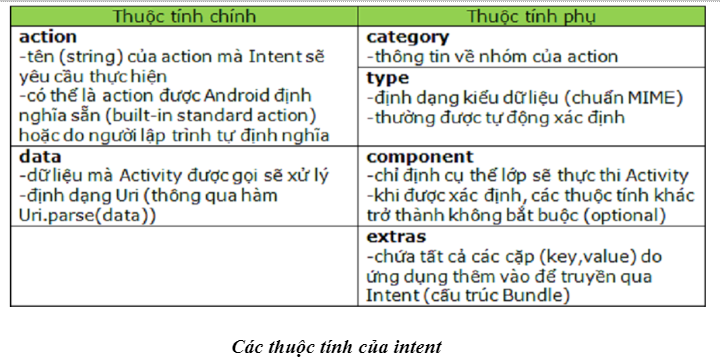
Khái niệm Intent:

* Là một cấu trúc dữ liệu mô tả cách thức, đối tượng thực hiện của một Activity.
* Là cầu nối giữa các Activity: ứng dụng Android thường bao gồm nhiều Activity, mỗi Activity hoạt động độc lập với nhau và thực hiện những công việc khác nhau. Intent chính là người đưa thư, giúp các Activity có thể triệu gọi cũng như truyền các dữ liệu cần thiết tới một Activity khác. Điều này cũng giống như việc di chuyển qua lại giữa các Forms trong lập trình Windows Form.



Dữ liệu của Intent:

* Intent về cơ bản là một cấu trúc dữ liệu, được mô tả trong lớp android.content.Intent
* Các thuộc tính của một đối tượng Intend:

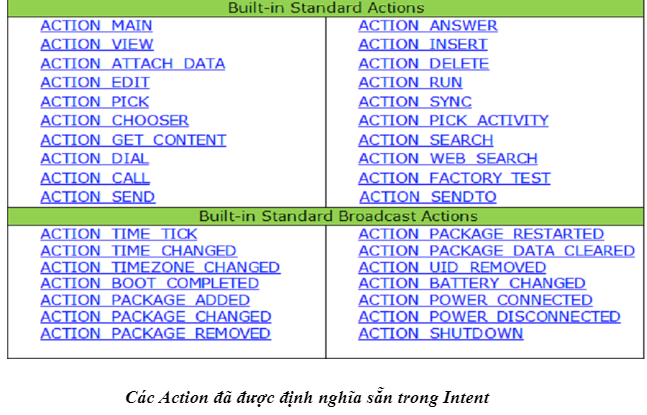


Các Action được định nghĩa sẵn:

Dưới đây là những hằng String đã được định nghĩa sẵn trong lớp Intent. Đi kèm với nó là các Activity hay Application được xây dựng sẵn sẽ được triệu gọi mỗi khi Intent tương ứng được gửi (tất nhiên khi được cung cấp đúng data).

VD: Gọi tới một số điện thoại:

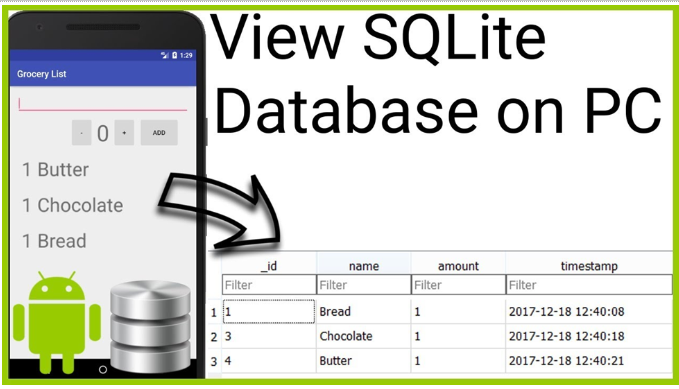
|  |
| --- |
| Intent dialIntent = new Intent(Intent.ACTION\_DIAL, Uri.parse("tel:123456"));  startActivity(dialIntent); |



## **Thành phần database**

SQLite là một dạng CSDL tương tự như Mysql, PostgreSQL... Đặc điểm của SQLite là gọn, nhẹ, đơn giản. Chương trình gồm 1 file duy nhất vỏn vẹn chưa đến 500kB, không cần cài đặt, không cần cấu hình hay khởi động mà có thể sử dụng ngay. Dữ liệu database cũng được lưu ở một file duy nhất. Không có khái niệm user, password hay quyền hạn trong SQLite database.

SQLite không thích hợp với những hệ thống lớn nhưng ở quy mô vừa tầm thì SQLite phát huy uy lực và không hề yếu kém về mặt chức năng hay tốc độ. Với các đặc điểm trên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm v..v.. và là sự lựa chọn phù hợp cho những người bắt đầu học database. Hiện nay thì SQLite đã được ứng dụng vào smartphone như iPhone và Android để lưu trữ dữ liệu.



Giao diện SQLite Manager

CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG