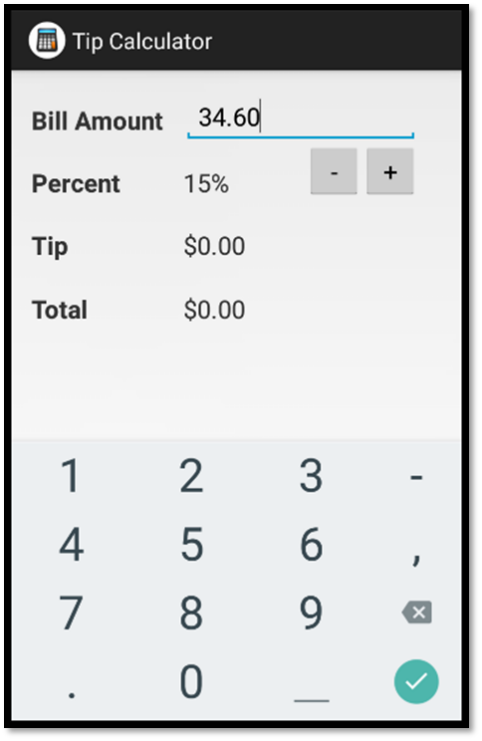
|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC THỦY LỢI  **KHOA CNTT** |  |

**BÀI THỰC HÀNH SỐ 01**

Học phần: Phát triển ứng dụng cho thiết bị di động

**Dự án**: Tip Calculator



**Mục tiêu**: Sinh viên làm quen với việc tạo giao diện ứng dụng Tip Calculator trên Android với Java đơn giản sử dụng Android Studio

|  |
| --- |
| Tiền boa, còn được gọi là tiền "tip" hay "gratuity", là một khoản tiền nhỏ được khách hàng trả thêm cho nhân viên phục vụ để thể hiện sự hài lòng với dịch vụ họ nhận được. Văn hóa tiền boa phổ biến ở nhiều quốc gia, đặc biệt là ở các nước phương Tây. Mức độ tiền boa được mong đợi có thể thay đổi tùy thuộc vào quốc gia, loại hình dịch vụ và chất lượng dịch vụ.   * **Hoa Kỳ**: Mức tiền boa thông thường ở Hoa Kỳ là từ 15% đến 20% tổng hóa đơn. Một số người có thể boa cao hơn cho dịch vụ đặc biệt tốt hoặc thấp hơn cho dịch vụ kém. * **Canada**: Mức tiền boa thông thường ở Canada là từ 15% đến 18% tổng hóa đơn. * **Vương quốc Anh**: Mức tiền boa thông thường ở Vương quốc Anh là từ 10% đến 12% tổng hóa đơn. Tuy nhiên, việc boa không bắt buộc và nhiều người không boa. * **Châu Âu**: Mức tiền boa ở Châu Âu có thể thay đổi tùy thuộc vào quốc gia. Ví dụ, ở Pháp, mức tiền boa thông thường là từ 10% đến 15% tổng hóa đơn, trong khi ở Ý, mức tiền boa thường là từ 5% đến 10%. * **Châu Á**: Tiền boa không phổ biến ở nhiều quốc gia Châu Á. Tuy nhiên, ở một số quốc gia như Nhật Bản và Hàn Quốc, khách hàng có thể để lại một khoản tiền nhỏ (thường là 1000 yên hoặc 1000 won) để thể hiện sự đánh giá cao.   Bài toán Tip Calculator là một bài toán đơn giản được sử dụng để tính toán số tiền boa cần trả cho nhân viên phục vụ. Bài toán thường bao gồm ba biến đầu vào:   * **Tổng hóa đơn (Bill Total):** Số tiền cần thanh toán cho dịch vụ. * **Tỷ lệ phần trăm tiền boa (Tip Percentage):** Mức phần trăm tổng hóa đơn mà khách hàng muốn boa. * **Số lượng người ăn (Number of People):** Số người chia sẻ hóa đơn.   Công thức để tính toán số tiền boa là:  **Tip Amount = (Bill Total \* Tip Percentage) / 100**  Ví dụ:   * Tổng hóa đơn: 100 USD * Tỷ lệ phần trăm tiền boa: 20% * Số lượng người ăn: 2   **Số tiền boa = (100 USD \* 20%) / 100 = 20 USD**  **Tổng số tiền thanh toán = Bill Total + Tip Amount = 100 USD + 20 USD = 120 USD**  **Mỗi người ăn cần thanh toán = Total Payment / Number of People = 120 USD / 2 = 60 USD** |

**HƯỚNG DẪN THỰC HIỆN**

**Bước 1: Thiết lập Môi trường lập trình** (*~~Bỏ qua nếu đã thực hiện~~*)

* Tải xuống và cài đặt **Android Studio** từ trang web chính thức của Android.
* Khởi chạy Android Studio và tạo một **New Project**.
* Chọn **Empty View Activity** hoặc **No Activity** và đặt tên cho ứng dụng, ví dụ: **TipCalculator**.
* Đảm bảo rằng ngôn ngữ lập trình được lựa chọn là **Java**.

Tạo **New Project** và chọn **Empty View Activity**

A screenshot of a computer

Description automatically generated

Đặt tên cho ứng dụng là **TipCalculator** và chọn ngôn ngữ là **Java**

**A screenshot of a computer

Description automatically generated**

**Bước 2: Thiết kế Giao diện người Dùng (UI)**

* Trong **Project** pane, mở tệp **activity\_main.xml** nằm trong thư mục **layout**.
* Sử dụng **Design Editor**, bạn có thể kéo và thả các thành phần UI như **EditText**, **TextView**, **Button**, v.v. từ **Palette** vào màn hình của bạn.
* Đặt các **id** cho mỗi thành phần UI để có thể tham chiếu trong code Java, ví dụ: **etBillAmount**, **tvTip**, **tvTotal**.

|  |
| --- |
| **Nên chọn kiểu Layout nào cho giao diện đề bài cho?**  Khi xây dựng giao diện cho một **Activity** trong Android, có nhiều loại **Layout** mà bạn có thể sử dụng, mỗi loại có những đặc điểm và mục đích sử dụng riêng. Dưới đây là một số **Layout** phổ biến mà bạn có thể cân nhắc để xây dựng giao diện cho ứng dụng Tip Calculator:   1. **LinearLayout**:    * Đây là layout đơn giản nhất, nơi các thành phần (child views) được sắp xếp theo chiều dọc hoặc ngang.    * Đối với Tip Calculator, bạn có thể sắp xếp các thành phần theo chiều dọc để tạo một danh sách từ trên xuống dưới: nhập số tiền, chọn tip, và hiển thị tổng số tiền. 2. **RelativeLayout**:    * Cho phép bạn định vị các thành phần liên quan đến các đối tượng khác trong layout hoặc liên quan đến cha mẹ layout.    * Đây là lựa chọn tốt nếu bạn cần đặt một số thành phần ở các vị trí cụ thể so với nhau, như đặt tổng số tiền ngay dưới số tiền tip. 3. **ConstraintLayout**:    * Là một layout linh hoạt và mạnh mẽ mà cho phép bạn tạo ra các giao diện phức tạp và đáp ứng mà không cần nhiều layout lồng nhau.    * Bạn có thể tạo các ràng buộc giữa các thành phần với nhau hoặc với cạnh của layout để xác định vị trí của chúng.    * Đối với Tip Calculator, **ConstraintLayout** là lựa chọn tốt nhất bởi vì nó sẽ cho phép bạn dễ dàng canh chỉnh và điều chỉnh vị trí của các thành phần để phù hợp với mọi kích thước màn hình. 4. **FrameLayout**:    * Đơn giản và thường được dùng để chứa một thành phần duy nhất hoặc làm container cho fragment.    * Không thường được dùng làm layout chính cho một **Activity** phức tạp, nhưng bạn có thể dùng nó để chứa các thành phần khác trong một phần nhất định của màn hình. 5. **TableLayout**:    * Phân chia màn hình thành hàng và cột, giống như một bảng.    * Có thể dùng cho một ứng dụng như Tip Calculator nếu bạn muốn một cách trình bày dữ liệu rõ ràng theo kiểu bảng. 6. **GridLayout**:    * Một phiên bản mạnh mẽ hơn của **TableLayout**, hỗ trợ cả rowspan và colspan, cho phép bạn tạo các grid phức tạp hơn.    * Nó hữu ích cho việc tạo một bố cục giao diện tổ chức theo một grid. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1. Mở file activity\_main.xml**   * Trong Android Studio, trên cửa sổ **Project** ở bên trái, mở thư mục **res** và sau đó mở thư mục **layout**. * Nhấp đúp vào **activity\_main.xml** để mở nó trong trình chỉnh sửa.     **2. Chọn chế độ Design**   * Ở góc dưới của trình chỉnh sửa, bạn có thể chuyển đổi giữa **Code** và **Design**. Chọn **Design** để kéo và thả các thành phần.   **3. Xóa các thành phần mặc định**   * Nếu có bất kỳ thành phần nào đã được đặt trong layout, bạn có thể chọn chúng và nhấn phím **Delete** để xóa.   **4. Chuyển đổi ConstraintLayout thành Relatvie Layout làm layout chính**   * Tại khung **Component Tree**, chuột phải vào layout mặc định (có kiểu Constraint Layout) > nhấp chọn **Convert view** .. > Chọn **Relative Layout**.      * Kết quả trông như thế này:  |  | | --- | | <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:paddingLeft="@dimen/activity\_horizontal\_margin"  android:paddingRight="@dimen/activity\_horizontal\_margin"  android:paddingTop="@dimen/activity\_vertical\_margin"  android:paddingBottom="@dimen/activity\_vertical\_margin" >  ...  <!-- Thêm các TextView và Button khác tương tự như trên dựa vào id và các thuộc tính định vị -->  </RelativeLayout> |     **5. Thêm TextView cho Bill Amount**   * Kéo **TextView** vào layout và đặt **id** cho nó, ví dụ: **@+id/** **billAmountLabel**.  |  | | --- | | <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/bill\_amount\_label"  android:id="@+id/billAmountLabel"  android:textSize="20sp"  android:textStyle="bold"  android:layout\_alignParentTop="true"  android:layout\_alignParentLeft="true"  android:layout\_alignParentStart="true"  android:paddingTop="10dp" /> |       **5. Thêm EditText cho Bill Amount**   * Kéo **EditText** vào layout và đặt **id** cho nó, ví dụ: **@+id/** **billAmountEditText**. * Thêm hint để người dùng biết nơi nhập số tiền hóa đơn: **android:hint="Bill Amount"**. * Đặt kiểu nhập số cho **EditText**: **android:inputType="numberDecimal"**.  |  | | --- | | <EditText  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:inputType="numberDecimal"  android:ems="8"  android:id="@+id/billAmountEditText"  android:text="@string/bill\_amount"  android:textSize="20sp"  android:layout\_alignTop="@+id/billAmountLabel"  android:layout\_toRightOf="@+id/billAmountLabel"  android:layout\_toEndOf="@+id/billAmountLabel"  android:layout\_marginLeft="16dp" /> |       **4. Thêm TextView cho tiêu đề Percent và các Button để điều chỉnh phần trăm**   * Bạn sẽ cần một **TextView** để hiển thị tiêu đề "Percent", sau đó hai **Button** để tăng và giảm phần trăm.  |  | | --- | | <!-- The tip percent -->  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/tip\_percent\_label"  android:id="@+id/percentLabel"  android:textSize="20sp"  android:textStyle="bold"  android:layout\_alignBottom="@+id/percentDownButton"  android:layout\_alignParentLeft="true"  android:layout\_alignParentStart="true" />      <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/tip\_percent"  android:id="@+id/percentTextView"  android:layout\_alignTop="@+id/percentLabel"  android:layout\_alignLeft="@+id/billAmountEditText"  android:layout\_alignStart="@+id/billAmountEditText"  android:textSize="20sp" />      <Button  android:layout\_width="45dp"  android:layout\_height="45dp"  android:text="@string/decrease"  android:id="@+id/percentDownButton"  android:layout\_below="@+id/billAmountEditText"  android:layout\_toLeftOf="@+id/percentUpButton"  android:layout\_toStartOf="@+id/percentUpButton"  android:textSize="20sp" />      <Button  android:layout\_width="45dp"  android:layout\_height="45dp"  android:text="@string/increase"  android:id="@+id/percentUpButton"  android:layout\_alignTop="@+id/percentDownButton"  android:layout\_alignRight="@+id/billAmountEditText"  android:layout\_alignEnd="@+id/billAmountEditText"  android:textSize="20sp" /> |   **5. Thêm TextView cho Tip và Total**   * Kéo và thả hai **TextView**, một cho Tip và một cho Total.  |  | | --- | | <!-- The tip amount -->  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/tip\_amount\_label"  android:id="@+id/tipLabel"  android:textSize="20sp"  android:textStyle="bold"  android:layout\_marginTop="23dp"  android:layout\_below="@+id/percentDownButton"  android:layout\_alignParentLeft="true"  android:layout\_alignParentStart="true" />      <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/tip\_amount"  android:id="@+id/tipTextView"  android:layout\_alignTop="@+id/tipLabel"  android:layout\_alignLeft="@+id/percentTextView"  android:layout\_alignStart="@+id/percentTextView"  android:textSize="20sp" />      <!-- The total amount -->  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/total\_amount\_label"  android:id="@+id/totalLabel"  android:textSize="20sp"  android:textStyle="bold"  android:layout\_below="@+id/tipLabel"  android:layout\_alignParentLeft="true"  android:layout\_alignParentStart="true"  android:layout\_marginTop="24dp" />      <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@string/total\_amount"  android:id="@+id/totalTextView"  android:textSize="20sp"  android:layout\_alignTop="@+id/totalLabel"  android:layout\_alignLeft="@+id/tipTextView"  android:layout\_alignStart="@+id/tipTextView" /> |   **6. Định dạng và Stylize các thành phần**   * Sử dụng thuộc tính **android:padding** để tạo khoảng cách bên trong giữa các thành phần và nội dung của chúng. (kèm ảnh code từ phần ở trên) * Thiết lập **android:layout\_margin** để tạo khoảng cách giữa các thành phần và các thành phần xung quanh. (kèm ảnh code từng phần ở trên) * Bạn cũng có thể thêm **android:textSize** để thay đổi kích thước chữ (kèm ảnh code từng phần ở trên), và **android:textColor** để thay đổi màu sắc của văn bản.   **7. Kích thước và Khoảng cách**   * Điều chỉnh **android:layout\_width** và **android:layout\_height** của các thành phần để chúng phù hợp với thiết kế của bạn. * Cân nhắc sử dụng **wrap\_content**, **match\_parent**, hoặc giá trị cụ thể (ví dụ: **dp**).   (kèm ảnh code từng phần ở trên)  **8. Kiểm tra và Xem trước**   * Sử dụng tính năng xem trước trong Android Studio để xem giao diện của bạn trên các loại thiết bị và kích cỡ màn hình khác nhau. * Đảm bảo rằng giao diện của bạn trông đẹp mắt và các thành phần được sắp xếp hợp lý trên tất cả các thiết bị.   **9. Lưu và Chạy thử nghiệm**   * Lưu thay đổi của bạn và chạy ứng dụng trên emulator hoặc thiết bị thực để kiểm tra xem giao diện có hoạt động như mong đợi không.   Khi hoàn thành các bước trên, bạn sẽ có một giao diện người dùng cơ bản cho ứng dụng Tip Calculator của mình. Có thể bạn sẽ cần phải điều chỉnh một số ràng buộc hoặc thuộc tính để chúng trông giống như mẫu bạn cung cấp. |

### Bước 3: Thêm logic xử lý trong Java

* Mở file **MainActivity.java**.
* Khai báo các biến cho mỗi thành phần UI mà bạn đã thêm trong **activity\_main.xml**.
* Trong phương thức **onCreate()**, bạn sẽ liên kết các biến này với UI bằng **findViewById()**
* Thêm bộ lắng nghe sự kiện click cho nút tính toán
* Xây dựng phương thức **calculateTip()** để tính tiền tip và tổng cộng

|  |
| --- |
| package com.murach.tipcalculator;  import java.text.NumberFormat;  import android.app.Activity;  import android.os.Bundle;  import android.view.KeyEvent;  import android.view.View;  import android.view.View.OnClickListener;  import android.view.inputmethod.EditorInfo;  import android.widget.Button;  import android.widget.EditText;  import android.widget.TextView;  import android.widget.TextView.OnEditorActionListener;  import android.content.SharedPreferences;  import android.content.SharedPreferences.Editor;  public class TipCalculatorActivity extends Activity  implements OnEditorActionListener, OnClickListener {  // define variables for the widgets  private EditText billAmountEditText;  private TextView percentTextView;  private Button percentUpButton;  private Button percentDownButton;  private TextView tipTextView;  private TextView totalTextView;  // define the SharedPreferences object  private SharedPreferences savedValues;  // define instance variables that should be saved  private String billAmountString = "";  private float tipPercent = .15f;  @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.main\_activity);  // get references to the widgets  billAmountEditText = (EditText) findViewById(R.id.billAmountEditText);  percentTextView = (TextView) findViewById(R.id.percentTextView);  percentUpButton = (Button) findViewById(R.id.percentUpButton);  percentDownButton = (Button) findViewById(R.id.percentDownButton);  tipTextView = (TextView) findViewById(R.id.tipTextView);  totalTextView = (TextView) findViewById(R.id.totalTextView);  // set the listeners  billAmountEditText.setOnEditorActionListener(this);  percentUpButton.setOnClickListener(this);  percentDownButton.setOnClickListener(this);  // get SharedPreferences object  savedValues = getSharedPreferences("SavedValues", MODE\_PRIVATE);  }  @Override  public void onPause() {  // save the instance variables  Editor editor = savedValues.edit();  editor.putString("billAmountString", billAmountString);  editor.putFloat("tipPercent", tipPercent);  editor.commit();  super.onPause();  }  @Override  public void onResume() {  super.onResume();  // get the instance variables  billAmountString = savedValues.getString("billAmountString", "");  tipPercent = savedValues.getFloat("tipPercent", 0.15f);  // set the bill amount on its widget  billAmountEditText.setText(billAmountString);  // calculate and display  calculateAndDisplay();  }  public void calculateAndDisplay() {  // get the bill amount  billAmountString = billAmountEditText.getText().toString();  float billAmount;  if (billAmountString.equals("")) {  billAmount = 0;  }  else {  billAmount = Float.parseFloat(billAmountString);  }  // calculate tip and total  float tipAmount = billAmount \* tipPercent;  float totalAmount = billAmount + tipAmount;  // display the other results with formatting  NumberFormat currency = NumberFormat.getCurrencyInstance();  tipTextView.setText(currency.format(tipAmount));  totalTextView.setText(currency.format(totalAmount));  NumberFormat percent = NumberFormat.getPercentInstance();  percentTextView.setText(percent.format(tipPercent));  }  @Override  public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {  if (actionId == EditorInfo.IME\_ACTION\_DONE ||  actionId == EditorInfo.IME\_ACTION\_UNSPECIFIED) {  calculateAndDisplay();  }  return false;  }  @Override  public void onClick(View v) {  switch (v.getId()) {  case R.id.percentDownButton:  tipPercent = tipPercent - .01f;  calculateAndDisplay();  break;  case R.id.percentUpButton:  tipPercent = tipPercent + .01f;  calculateAndDisplay();  break;  }  }  } |

**Bước 4: Kiểm thử ứng Dụng**

* Sử dụng AVD Manager trong Android Studio để tạo một **Android Virtual Device** và chạy ứng dụng của bạn trên nó \*hoặc sử dụng điện thoạt thật để kiểm thử
* Nhập giá trị vào **EditText** cho số tiền hóa đơn và bấm vào nút tính toán để xem kết quả.

**Bước 5: Tối ưu và Debug**

* Kiểm tra các trường hợp nhập sai và cải thiện bằng cách thêm thông báo lỗi thân thiện với người dùng.
* Sử dụng **Logcat** để theo dõi và sửa các lỗi khi chạy ứng dụng.

**Bước 6: Xây dựng và chia sẻ APK**

* Trong Android Studio, chọn **Build** > **Build Bundle(s) / APK(s)** > **Build APK(s)** để tạo APK cho ứng dụng.
* Chia sẻ file APK với người khác hoặc tải lên Google Play Store.