# **COS30017 – Software Development for Mobile Devices**

## **Formative Assignment 2**

### **Task 1**

Java Code

package com.example.task1;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.content.ContextCompat;  
  
import android.content.res.Configuration;  
import android.graphics.Color;  
import android.media.MediaPlayer;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
  
public class MainActivity extends AppCompatActivity {  
 private Button btn\_sneeze,btn\_blow,btn\_meds;  
 private View inner,outer;  
 static int *state* = 2;  
 static int *health*=10;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 initUI();  
 }  
 private void sneeze(){  
 final MediaPlayer mp=MediaPlayer.*create*(this,R.raw.*sneeze2*);  
 mp.start();  
 }  
 private void blow(){  
 final MediaPlayer mp=MediaPlayer.*create*(this,R.raw.*blow\_nose*);  
 mp.start();  
 }  
 private void meds(){  
 final MediaPlayer mp=MediaPlayer.*create*(this,R.raw.*slurp*);  
 mp.start();  
 }  
 private void initUI(){  
 btn\_sneeze=findViewById(R.id.*button\_sneeze*);  
 btn\_blow=findViewById(R.id.*button\_blow*);  
 btn\_meds=findViewById(R.id.*button\_take\_medication*);  
 outer=findViewById(R.id.*vertical\_outer*);  
 inner=findViewById(R.id.*viewLinearLayout*);  
  
  
  
 Log.*d*("health",String.*valueOf*(*health*));  
 btn\_sneeze.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view) {  
 sneeze();  
 *state*=0;  
 *health*--;  
 health();  
 }  
 });  
  
  
 btn\_blow.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view) {  
 blow();  
 *state*=1;  
  
 }  
 });  
  
 btn\_meds.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view) {  
 if(*health*>=10){  
 Log.*d*("message","Health is at >=10");  
 }  
 else{  
 meds();  
 if(*health*+2>10){  
 *health*++;  
 health();  
 }else {  
 *health* += 2;  
 health();  
 }  
 }  
 }  
 });  
 }  
  
 @Override  
 public void onConfigurationChanged(Configuration newConfig) {  
 super.onConfigurationChanged(newConfig);  
 if(*state*==0){  
 blow();  
 *state*=1;  
 }else{  
 sneeze();  
 *state*=0;  
 *health*--;  
 health();  
 }  
  
 }  
 private void checkHealth(){  
 if(*health*>5 && *health*<=7){  
 outer.setBackgroundColor(ContextCompat.*getColor*(this,R.color.*light\_blue*));  
 inner.setBackgroundColor(ContextCompat.*getColor*(this,R.color.*light\_blue*));  
 }else if(*health*<=5 && *health*>0){  
 outer.setBackgroundColor(ContextCompat.*getColor*(this,R.color.*red*));  
 inner.setBackgroundColor(ContextCompat.*getColor*(this,R.color.*red*));  
 btn\_sneeze.setEnabled(true);  
 }else if(*health*==0){  
 Log.*d*("message","Health is empty, please take medication");  
 *health*=0;  
 btn\_sneeze.setEnabled(false);  
 }else{  
 outer.setBackgroundColor(ContextCompat.*getColor*(this,R.color.*white*));  
 inner.setBackgroundColor(ContextCompat.*getColor*(this,R.color.*white*));  
 }  
 }  
 private void printHealth(){  
 Log.*d*("health",String.*valueOf*(*health*));  
 }  
 private void health(){  
 printHealth();  
 checkHealth();  
 }  
}

Android Manifest XML

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.task1">  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme">  
 <activity android:name=".MainActivity"  
 android:configChanges="orientation|screenSize">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

### **Task 2**

String externalization separates string values from its implementation and uses a reference of the string value in the implementation. This assists in localization as you do not need to rebuild the application to accommodate non-English languages and only need to translate the string values in stored in the string file.

A screenshot of a social media post

Description automatically generated 

It is seen above that Task 2.2 has been localized to Japanese from English. This is done by switching the Android app’s main language from English to Japanese by accessing Settings > System > Languages & input.

Screenshot that the string value is referenced in the layout file

A screenshot of a cell phone

Description automatically generated

### **Task 3**

A close up of food

Description automatically generated

A screenshot of a cell phone

Description automatically generated

The Constraint layout as well as Linear layout is used in the structuring of the app’s frontend. ScrollView was employed to make the app scrollable because not all UI elements are able to fit the initial screen size. A Toast message is used to display the total price when the Order button is pressed.

For the backend, the initialization of the interactable UI elements were separated into various methods based on UI type. The principle of separation of concern was employed in the various initialization methods, and various support methods created to prevent code redundancy. The only design pattern employed was the Adapter design pattern, or in this case the ArrayAdapter, which was used to adapt the array of strings to the Spinner UI’s dropdown menu. Comments were included to describe the functions and variables used, this takes into consideration code maintainability. Furthermore, a HashMap data structure was used to store the prices of the food items as well as the values to the size of the burger.

Aside from the initialization methods for the various UI elements, various support methods such as check(), updatePrice(), checkSize(), clearCheckBox(), checkSelection(), checkVeg(), vegChecked() and vegUnchecked() were created for code reusability. Below is a list of each of the mentioned methods functions:

* check() – used to check what type of meat is chosen and the price of the meat to be added to the total price.
* updatePrice() – update the TextView of the total price.
* checkSize() – used to check the size chosen for the burger and multiply the total price by the type of size chosen.
* clearCheckBox() – used to clear the checkboxes when the reset button is pressed.
* checkSelection() – check if three vegetables were selected, if true, mute the unselected checkbox, or unmute it when the selection of vegetables drops to 2.
* checkVeg() – check the number of vegetables chosen and apply the promotion if the number of vegetables chosen is 3.
* vegChecked() – adds the price of the vegetable to the total price, decreases the value for selection, increases the value of numOfVeg, calls the checkSelection method and checkVeg method.
* vegUnchecked() – decreases the price of the vegetable from the total price, increases the value for the selection, decreases the value of numOfVeg, calls the checkSelection method, and calls the checkVeg method.

Code

package com.example.burgerqueen;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.util.Log;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Button;  
import android.widget.CheckBox;  
import android.widget.RadioButton;  
import android.widget.RadioGroup;  
import android.widget.Spinner;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import java.lang.reflect.Array;  
import java.util.ArrayList;  
import java.util.HashMap;  
  
public class MainActivity extends AppCompatActivity{  
 //Spinner fields and related variables  
 private Spinner spinner;  
 private ArrayAdapter<String> adapter;  
 String[] size;  
 static String *sizeDefault*="Small";  
  
 //Button UI variables  
 private Button order,reset;  
 private RadioGroup radio\_group;  
 private RadioButton radiobutton;  
  
 //Total price variables  
 private TextView price;  
 static double *initialPrice*=0.00;  
  
 //Meat section variables  
 static Boolean *isMeatPicked*=false;  
 static double *meat\_price*; //used to change back total price if meat is unpicked  
 // and apply new price for new meat selected  
 static Boolean *isProteinPicked*=false; //check if meat section is selection, it is required  
  
 //Vegetable section variables  
 //static String[] checkBox\_ID={"lettuce","tomato","pickle","onion","cheese","mayo","mustard"};  
 static String[] *checkBox\_fiber\_ID*;  
 static String[] *checkBox\_junk\_ID*;  
 //static CheckBox[] checkBoxes=new CheckBox[checkBox\_ID.length];  
 static CheckBox[] *checkBoxes\_fiber*;  
 static CheckBox[] *checkBoxes\_junk*;  
 static int *selection*=3;  
 static int *numOfVeg*=0;  
 static boolean *vegPromo*=false;/\*vegPromo is used to check if promo is hit\*/  
  
 //Hashmaps  
 private HashMap<String,Double> meat;  
 private HashMap<String,Double> fiber;  
 private HashMap<String,Double> junk;  
 private HashMap<String,Double> sizeHash;  
  
 public MainActivity() {  
 }  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 //Call initMain to initialize all UI elements  
 initMain();  
 }  
 private void initMain(){  
 //Initialization individual UI elements and systems of the program.  
 initPrice();  
 initSpinner();  
 initRadioGroup();  
 initCheckBox();  
 initButton();  
 }  
 private void initPrice(){  
 price=findViewById(R.id.*total*);  
 price.setText(String.*format*("RM%.2f",*initialPrice*));  
  
 //HashMap Meat  
 meat=new HashMap<>();  
 meat.put("Beef",4.50);  
 meat.put("Fish",4.00);  
 meat.put("Chicken",3.00);  
 meat.put("Egg",2.00);  
  
 //HashMap Fiber  
 fiber=new HashMap<>();  
 fiber.put("Veg",0.50);  
  
 //HashMap Junk  
 junk=new HashMap<>();  
 junk.put("Cheese",1.00);  
 junk.put("Mayonaise",0.50);  
 junk.put("Mustard",0.70);  
  
 //HashMap Size  
 sizeHash=new HashMap<>();  
 sizeHash.put("Regular",1.2);  
 sizeHash.put("Small",1.0);  
 sizeHash.put("Large",1.3);  
 sizeHash.put("Gigantic",1.5);  
  
  
  
 }  
 private void initButton(){  
 order=findViewById(R.id.*order*);  
 reset=findViewById(R.id.*reset*);  
  
 //ClickListener  
 order.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view) {  
 if(!*isProteinPicked*){  
 Toast.*makeText*(MainActivity.this,"Please select a meat",Toast.*LENGTH\_SHORT*).show();  
 }  
 else{  
 Toast.*makeText*(MainActivity.this,"Your total is "+String.*format*("RM%.2f",*initialPrice*), Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 });  
 reset.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view) {  
 //Reset radio\_group  
 radio\_group.setOnCheckedChangeListener(null);  
 radio\_group.clearCheck();  
 *isMeatPicked*=false;  
 *meat\_price*=0.00;  
 *isProteinPicked*=false;  
  
 //Reset price  
 *initialPrice*=0.00;  
 updatePrice();  
  
 //Reset vegetable section  
 *selection*=3;  
 *vegPromo*=false;  
 clearCheckBox();  
  
 //Reset spinner  
 *sizeDefault*="Small";  
 spinner.setSelection(0);  
 spinner.setClickable(true);  
 spinner.setEnabled(true);  
 *numOfVeg*=0;  
  
 //Reinitialize  
 initMain();  
 }  
 });  
 }  
  
  
  
 private void initCheckBox(){  
 //initialize fiber and junk checkbox arrays(testing)  
 *checkBox\_fiber\_ID*=getResources().getStringArray(R.array.*fiber*);  
 *checkBox\_junk\_ID*=getResources().getStringArray(R.array.*junk*);  
 *checkBoxes\_fiber*=new CheckBox[*checkBox\_fiber\_ID*.length];  
 *checkBoxes\_junk*=new CheckBox[*checkBox\_junk\_ID*.length];  
 for(int i=0;i<*checkBox\_fiber\_ID*.length;i++){  
 int temp=getResources().getIdentifier(*checkBox\_fiber\_ID*[i],"id",getPackageName());  
 *checkBoxes\_fiber*[i]=findViewById(temp);  
 *checkBoxes\_fiber*[i].setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View v) {  
 if(((CheckBox) v).isChecked()){  
 vegChecked();  
 }else{  
 vegUnchecked();  
 }  
 }  
 });  
 }  
 for(int i=0;i<*checkBox\_junk\_ID*.length;i++){  
 int temp=getResources().getIdentifier(*checkBox\_junk\_ID*[i],"id",getPackageName());  
 *checkBoxes\_junk*[i]=findViewById(temp);  
 *checkBoxes\_junk*[i].setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View v) {  
 if(((CheckBox)v).isChecked()){  
 Double temp=junk.get(((CheckBox) v).getText().toString());  
 if(temp!=null){  
 *initialPrice*+=temp;  
 updatePrice();  
 }  
  
 }else{  
 Double temp=junk.get(((CheckBox) v).getText().toString());  
 if(temp!=null){  
 *initialPrice*-=temp;  
 updatePrice();  
 }  
 }  
 }  
 });  
 }  
 }  
 private void initRadioGroup(){  
 radio\_group=findViewById(R.id.*meat\_rg*);  
 radio\_group.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {  
 @Override  
 public void onCheckedChanged(RadioGroup radioGroup, int i) {  
 *isProteinPicked*=true;  
 int id=radio\_group.getCheckedRadioButtonId();  
 radiobutton=findViewById(id);  
 String result=radiobutton.getText().toString();  
 //check() is used to check for the meat selected and apply the price  
 // or detract the price accordingly  
 check(result);  
 //isMeatPicked is to confirm that selection has been done before, this is to allow reselection  
 *isMeatPicked*=true;  
 }  
 });  
  
 }  
 private void initSpinner(){  
 size=getResources().getStringArray(R.array.*sizes*);  
 spinner=findViewById(R.id.*spinner*);  
 adapter=new ArrayAdapter<String>(this,android.R.layout.*simple\_spinner\_item*,size);  
 adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);  
 spinner.setAdapter(adapter);  
 spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {  
 @Override  
 public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {  
 *sizeDefault*=spinner.getSelectedItem().toString();  
 //checkSize() is used to check the size selected and apply  
 // the appropriate percentage of increase to the total price  
 checkSize();  
 }  
  
 @Override  
 public void onNothingSelected(AdapterView<?> adapterView) {  
 //if nothing is selected, this is too ensure that the default size is "Small"  
 *sizeDefault*="Small";  
 }  
 });  
  
 }  
  
  
 private void check(String result){  
 //if true, it will subtract meat\_price from the initialPrice  
 // to revert the totalPrice to 0.00 and apply the new price based on the meat selected.  
 if(*isMeatPicked*){  
 *initialPrice*-=*meat\_price*;  
 *initialPrice*+=meat.get(result);  
 *meat\_price*=meat.get(result);  
 updatePrice();  
 }else {  
 //meat\_price is used to subtract later on if isMeatPicked is true.  
 *initialPrice*+=meat.get(result);  
 *meat\_price*=meat.get(result);  
 updatePrice();  
 }  
 }  
 private void updatePrice(){  
 price.setText(String.*valueOf*(String.*format*("RM%.2f",*initialPrice*)));  
 }  
  
 private void checkSize(){  
 //checks for the size selected, and apply the correct percentage for price increase.  
 for(int i=0;i<size.length;i++){  
 if(*sizeDefault*.equals("Small")){  
 continue;  
 }else{  
 *initialPrice*\*=sizeHash.get(*sizeDefault*);  
 spinner.setClickable(false);  
 spinner.setEnabled(false);  
 updatePrice();  
 break;  
 }  
 }  
  
 }  
 private void clearCheckBox(){  
 for(int i=0;i<*checkBoxes\_fiber*.length;i++){  
 *checkBoxes\_fiber*[i].setChecked(false);  
 *checkBoxes\_fiber*[i].setClickable(true);  
 *checkBoxes\_fiber*[i].setEnabled(true);  
 }  
 for(int i=0;i<*checkBoxes\_junk*.length;i++){  
 *checkBoxes\_junk*[i].setChecked(false);  
 *checkBoxes\_junk*[i].setClickable(true);  
 *checkBoxes\_junk*[i].setEnabled(true);  
 }  
 }  
  
 private void checkSelection(){  
 for(int i=0;i<*checkBoxes\_fiber*.length;i++){  
 if(*selection*==0){  
 if(!*checkBoxes\_fiber*[i].isChecked()){  
 *checkBoxes\_fiber*[i].setClickable(false);  
 *checkBoxes\_fiber*[i].setEnabled(false);  
 }  
 }else{  
 *checkBoxes\_fiber*[i].setEnabled(true);  
 *checkBoxes\_fiber*[i].setClickable(true);  
 }  
 }  
 }  
  
 private void checkVeg(){  
 if(*numOfVeg*==3){  
 *vegPromo*=true;  
 *initialPrice*-=0.50;  
 updatePrice();  
 }  
 else{  
 if(*vegPromo*){  
 *initialPrice*+=0.50;  
 *vegPromo*=false;  
 updatePrice();  
 }  
 }  
 }  
 private void vegChecked(){  
 *initialPrice*+=fiber.get("Veg");  
 updatePrice();  
 *selection*--;  
 *numOfVeg*++;  
 checkSelection();  
 checkVeg();  
 }  
 private void vegUnchecked(){  
 *initialPrice*-=fiber.get("Veg");  
 updatePrice();  
 *selection*++;  
 *numOfVeg*--;  
 checkSelection();  
 checkVeg();  
 }  
}