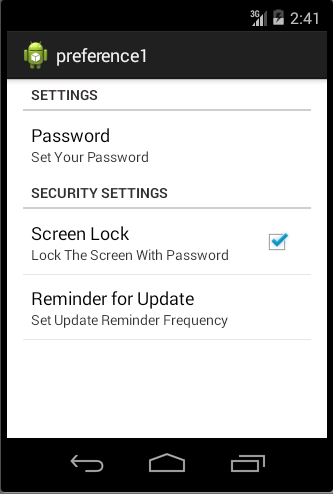
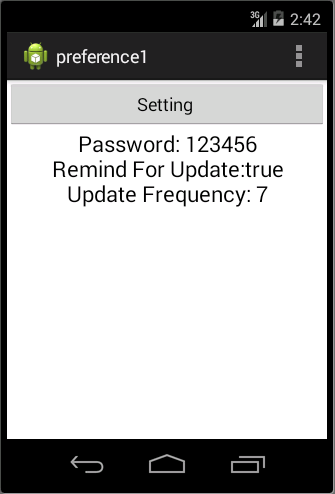
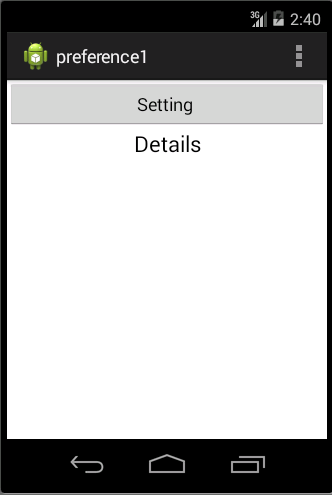
**Practical 9 (Preference and File)**

**Question 1 (Preference)**

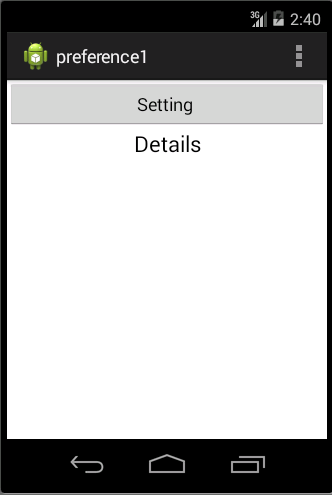


****

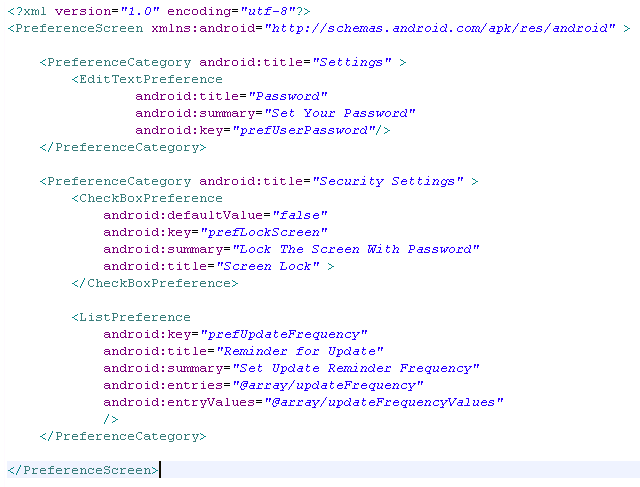


1. Your **activity\_main.xml** file should have something similar to xml below: **(p/s: You do not need to follow the xml as it might be different in each design.)**

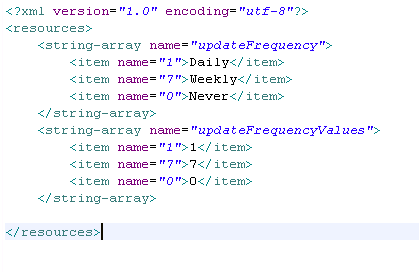




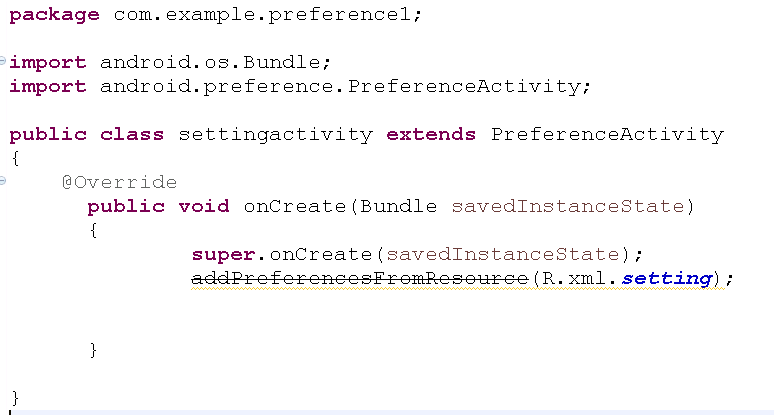
1. Create another file called **setting.xml in /res/xml.** You should create the xml folder by yourself. This setting.xml is for preference activity.



1. Create another file called **arrays.xml in /res/values.** This is the array we use in the list for the preference activity.



1. Create a class called **settingactivity.java**.



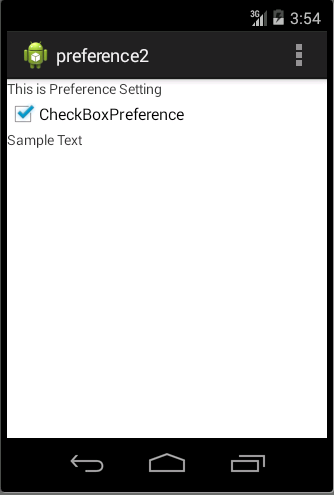
1. In your **MainActivity.java**, code the following code segment to provide function to the button created in activity\_main.xml.

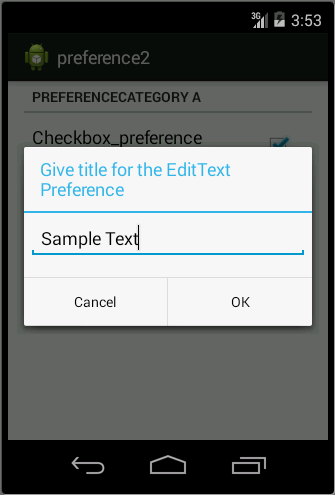


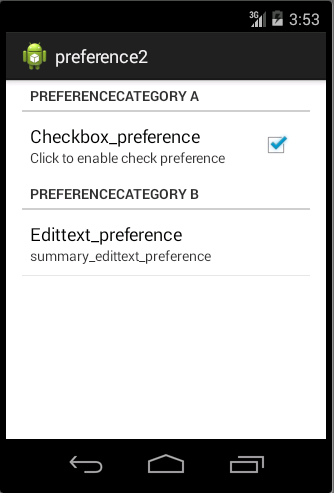
1. Make sure that you add the settingactivity in your Android Manifest.
2. After make sure all the coding appropriate done, you may compile and run your coding .

**Question 2 (Preference)**

1. In this question, you are required to create an app with the following interface.



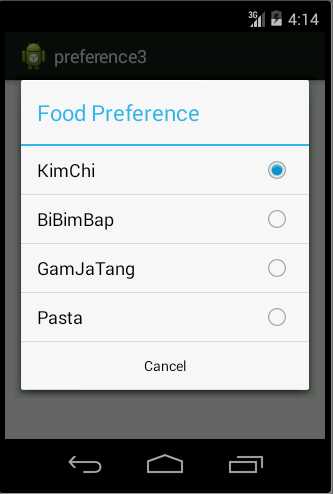


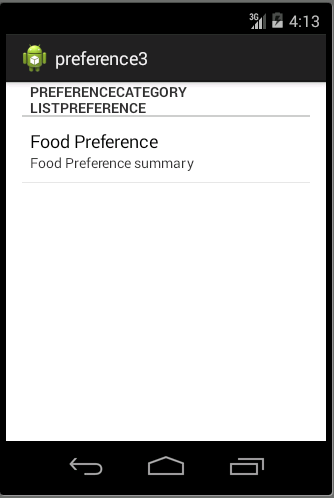


1. In this question, you should create a preference activity which have checkbox preference and also edittext preference.
2. You should create a fragment for this question. When the user click on the edittext preference, you should then create a dialog to get the text for the edittext.
3. After all the setting from user, you should then display the layout which shows the settings from user.

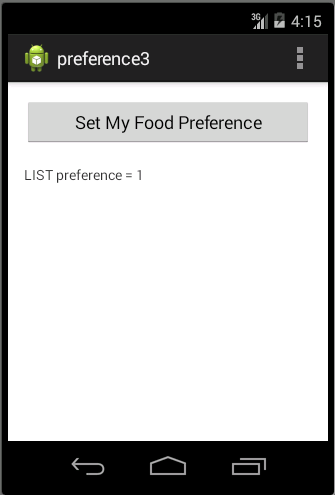
**Question 3(Preference)**

1. In this question, you are required to create an app with the following interface.









1. In this question, you should start with an activity with ONE button. When user click on the button, you should then create an preference activity which support a listpreference.
2. You can create any FOUR type of food as the element in the list. The selection from user should then display under the button.
3. You should create a fragment for this question.

**Notification.Builder**

Notification creation

A notification builder object must include the following as minimum requirement:

1. A small icon, set by setSmallIcon()
2. A title, set by setContentTitle()
3. A detail text, set by setContentText()

For example:

 Notification mBuilder = new Notification.Builder(mContext)  
         .setContentTitle(“Sample notification”)  
         .setContentText(“This is first notification from Builder”)  
         .setSmallIcon(R.drawable.new\_mail)  
         .setLargeIcon(aBitmap)  
         .build();

Steps to create a notification using Notification.Builder

1. **Create Notification Builder**

In this step, you should create a notification builder object and set its properties. You may refer to <http://developer.android.com/reference/android/app/Notification.Builder.html> for more properties of the object. You should at least set **small Icon**, **title** and **detail text** for every builder object.

1. **Attach Actions**

This is an optional part and required if you want to attach an action with the notification. Usually, pending intent is used. **setContentIntent()** used to set the action. For example:

Intent resultIntent = new Intent(this, ResultActivity.class);  
...  
// Because clicking the notification opens a new ("special") activity, there's  
// no need to create an artificial back stack.  
PendingIntent resultPendingIntent =  
    PendingIntent.getActivity(  
    this,  
    0,  
    resultIntent,  
    PendingIntent.FLAG\_UPDATE\_CURRENT  
);

PendingIntent resultPendingIntent;  
...  
mBuilder.setContentIntent(resultPendingIntent);

1. **Issue the notification**

**NotificationManager.notify()** is used to send your notification. Make sure the build() method called before the notify method. For example:

int mNotificationId = 001;  
// Gets an instance of the NotificationManager service  
NotificationManager mNotifyMgr =   
        (NotificationManager) getSystemService(NOTIFICATION\_SERVICE);  
// Builds the notification and issues it.  
mNotifyMgr.notify(mNotificationId, mBuilder.build());

**Question 4**

Create an app to create notification. In this question, you should create an interface to allow user to set their notification title, subject and also content. Once the user click on the button, the notification should be created. **p/s: Use notification.builder to do this.**

