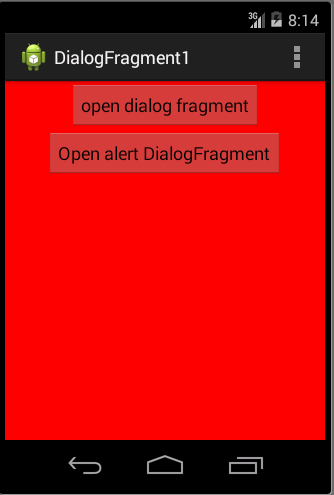
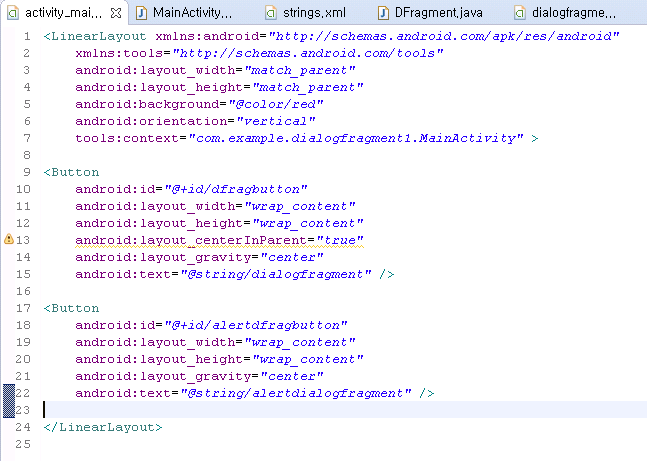
**Practical 6 (DialogFragment)**

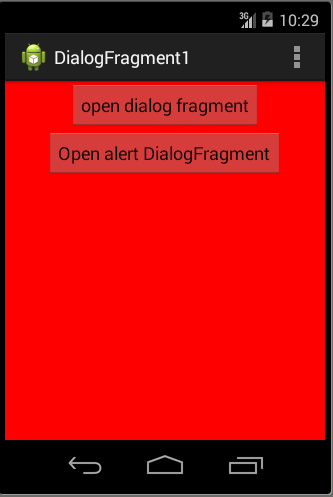
**Question 1 (Basic DialogFragment)**

1. In the /layout directory, open the **activity\_main.xml** and design the interface as below using graphical layout.

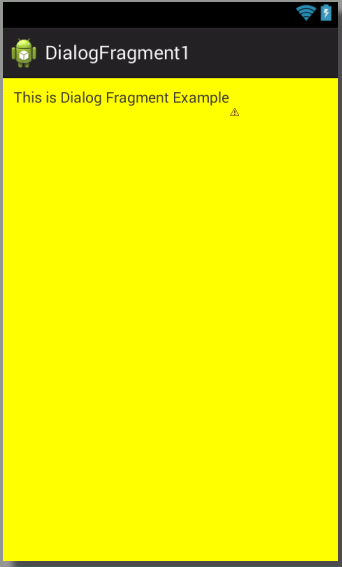


1. Your **activity\_main.xml** file should have something similar to xml below: **(p/s: You do not need to follow the xml as it might be different in each design.)**





1. Create another file called **dialogfragment.xml in /res/layout.** You should create the interface which similar to the following.

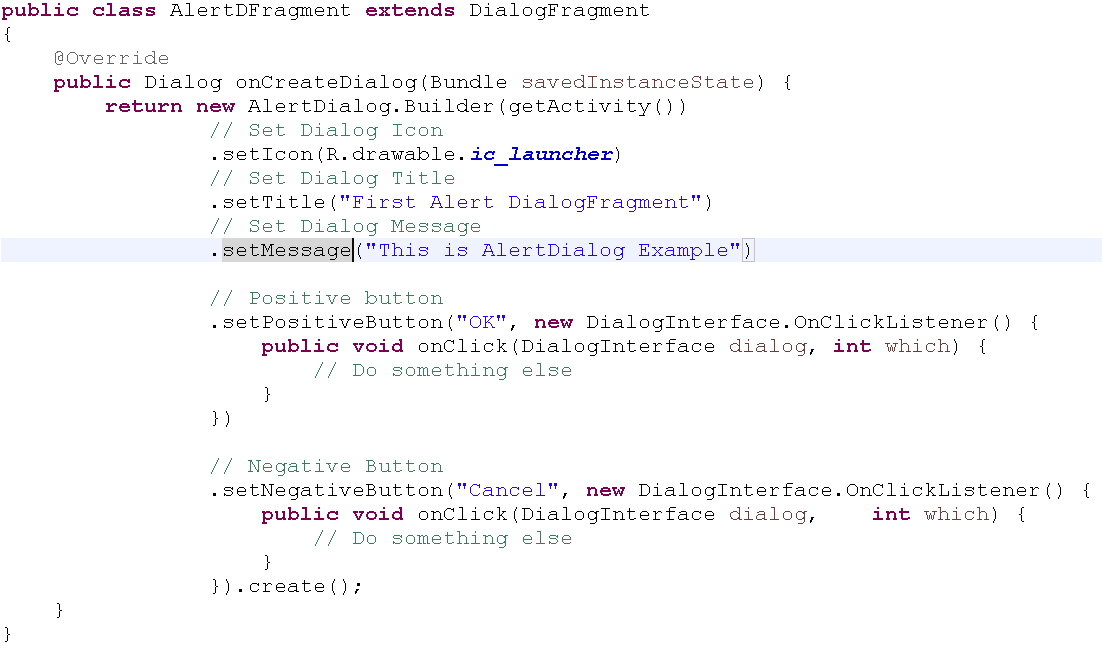




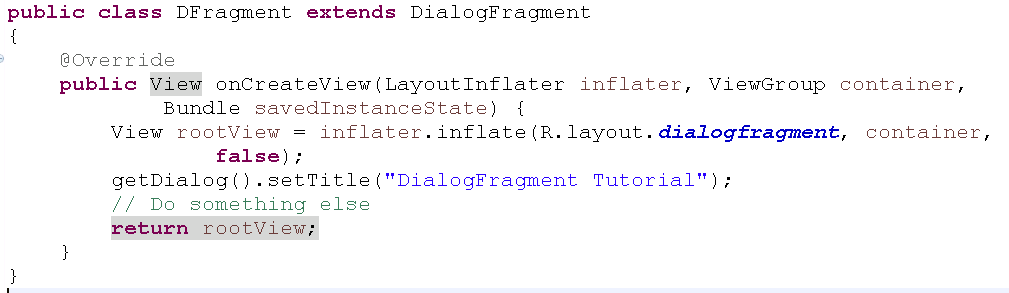
1. In your **MainActivity.java**, code the following code segment to provide function to the buttons created in activity\_main.xml.



1. Create a class called **AlertDFragment.java** for AlertDialog.



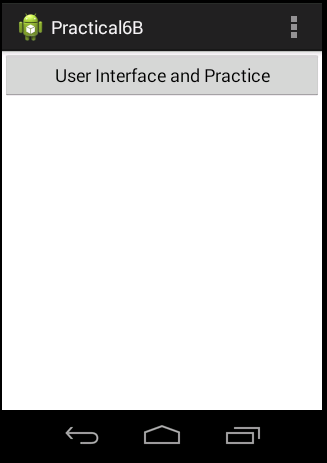
1. Create another class called **DFragment.java** which extends from DialogFragment.



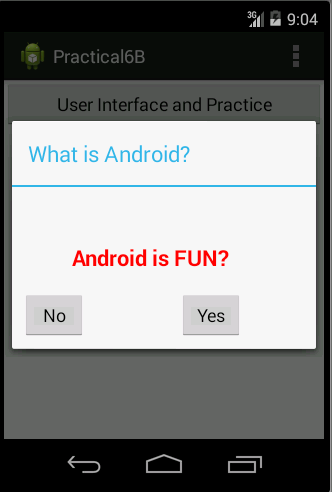
1. After make sure all the coding appropriate done, you may compile and run your coding .

**Question 2 (DialogFragment)**

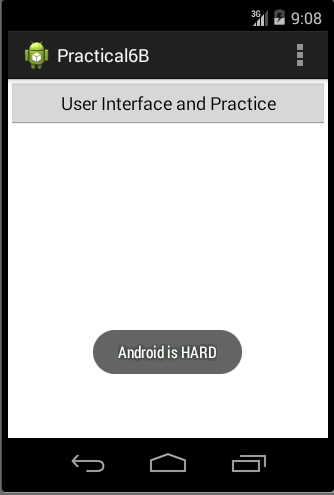
1. In this question, you are required to create an app with the following interface.



1. When user click on the button, the following DialogFragment should be created. In this part, you should create a xml file (layout) to design the fragment which similar to the sample output as below: User should answer the question by choosing YES/NO button.



1. Once user click on one of the button, the **DialogFragment** should be close and generate a Toast message which as below: If the user clicks on **YES** button, **“Android is EASY”** generated, else **“Android is HARD”** created.



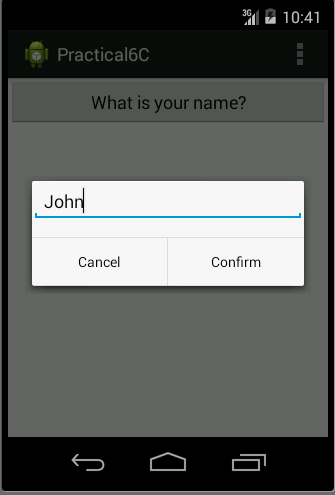


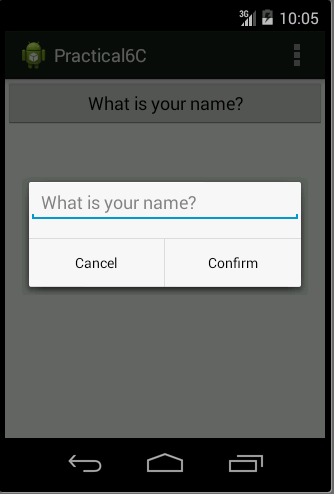
**Question 3(Fragment)**

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

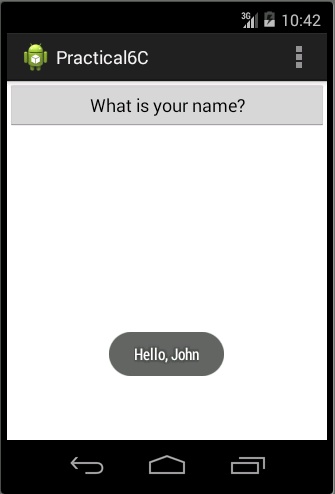


1. When user click on the button, the following DialogFragment should be created. In this part, you should create a xml file (layout) to design the fragment which similar to the sample output as below: User should enter their name and click on **Confirm** button to continue or **Cancel** button to cancel.





1. Once user click on one of the button, the **DialogFragment** should be close and generate a Toast message which as below: “Hello, (name you entered).”



**Question 4(TimerDialog)**

1. In the /layout directory, open the **activity\_main.xml** and design the interface as below using graphical layout.



1. Create a class TimerDialogFragment with the following code

**public** **class** TimerDialogFragment **extends** DialogFragment

{

Handler mHandler ;

**int** mHour;

**int** mMinute;

**public** TimerDialogFragment(Handler h){

/\*\* Getting the reference to the message handler instantiated in MainActivity class \*/

mHandler = h;

}

@Override

**public** Dialog onCreateDialog(Bundle savedInstanceState){

/\*\* Creating a bundle object to pass currently set time to the fragment \*/

Bundle b = getArguments();

/\*\* Getting the hour of day from bundle \*/

mHour = b.getInt("set\_hour");

/\*\* Getting the minute of hour from bundle \*/

mMinute = b.getInt("set\_minute");

TimePickerDialog.OnTimeSetListener listener = **new** TimePickerDialog.OnTimeSetListener() {

@Override

**public** **void** onTimeSet(TimePicker view, **int** hourOfDay, **int** minute) {

// **TODO** Auto-generated method stub

mHour = hourOfDay;

mMinute = minute;

/\*\* Creating a bundle object to pass currently set time to the fragment \*/

Bundle b = **new** Bundle();

/\*\* Adding currently set hour to bundle object \*/

b.putInt("set\_hour", mHour);

/\*\* Adding currently set minute to bundle object \*/

b.putInt("set\_minute", mMinute);

/\*\* Adding Current time in a string to bundle object \*/

b.putString("set\_time", "Set Time : " + Integer.*toString*(mHour) + " : " + Integer.*toString*(mMinute));

/\*\* Creating an instance of Message \*/

Message m = **new** Message();

/\*\* Setting bundle object on the message object m \*/

m.setData(b);

/\*\* Message m is sending using the message handler instantiated in MainActivity class \*/

mHandler.sendMessage(m);

}

};

/\*\* Opening the TimePickerDialog window \*/

**return** **new** TimePickerDialog(getActivity(), listener, mHour, mMinute, **false**);

}

}

1. In your MainActivity.java, do the following coding.

**public** **class** MainActivity **extends** ActionBarActivity {

**int** mHour = 15;

**int** mMinute = 15;

/\*\* This handles the message send from TimerDialogFragment on setting Time \*/

Handler mHandler = **new** Handler(){

@Override

**public** **void** handleMessage(Message m){

/\*\* Creating a bundle object to pass currently set Time to the fragment \*/

Bundle b = m.getData();

/\*\* Getting the Hour of day from bundle \*/

mHour = b.getInt("set\_hour");

/\*\* Getting the Minute of the hour from bundle \*/

mMinute = b.getInt("set\_minute");

/\*\* Displaying a short time message containing time set by Timer dialog fragment \*/

Toast.*makeText*(getBaseContext(), b.getString("set\_time"), Toast.***LENGTH\_SHORT***).show();

}

};

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_main***);

/\*\* Click Event Handler for button \*/

OnClickListener listener = **new** OnClickListener() {

@Override

**public** **void** onClick(View v) {

/\*\* Creating a bundle object to pass currently set time to the fragment \*/

Bundle b = **new** Bundle();

/\*\* Adding currently set hour to bundle object \*/

b.putInt("set\_hour", mHour);

/\*\* Adding currently set minute to bundle object \*/

b.putInt("set\_minute", mMinute);

/\*\* Instantiating TimePickerDialogFragment \*/

TimerDialogFragment timePicker = **new** TimerDialogFragment(mHandler);

/\*\* Setting the bundle object on timer fragment \*/

timePicker.setArguments(b);

/\*\* Getting fragment manger for this activity \*/

FragmentManager fm = getFragmentManager();

/\*\* Starting a fragment transaction \*/

FragmentTransaction ft = fm.beginTransaction();

/\*\* Adding the fragment object to the fragment transaction \*/

ft.add(timePicker, "time\_picker");

/\*\* Opening the TimePicker fragment \*/

ft.commit();

}

};

/\*\* Getting an instance of Set button \*/

Button btnSet = (Button)findViewById(R.id.***btnSet***);

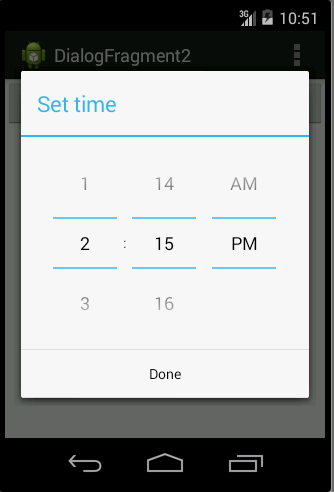
/\*\* Setting click event listener for the button \*/

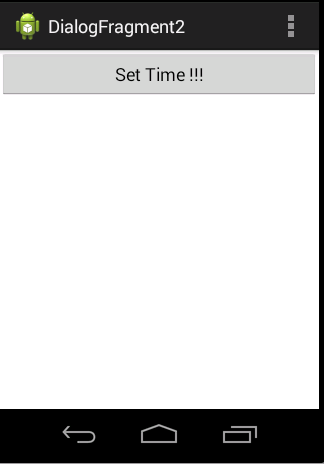
btnSet.setOnClickListener(listener);

}

1. Compile and run your application. You should get the following output.







**Question 5**

1. Base on the example from question 4, create a **DatePickerDialog** and create a Toast for the date you pick from the dialog. Hints: You may refer to developer.android for the DatePickerDialog.

http://developer.android.com/reference/android/app/DatePickerDialog.html

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:orientation=*"vertical"* >

<Button

android:id=*"@+id/button1"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"selectFrag"*

android:text=*"Fragment 1"* />

<Button

android:id=*"@+id/button2"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:onClick=*"selectFrag"*

android:text=*"Fragment 2"* />

<fragment

android:id=*"@+id/fragment\_place"*

android:name=*"com.example.samplefragment1.FragmentOne"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"* />

</LinearLayout>

1. Create a file called fragment\_one.xml in /res/layout folder. Use the following settings for the UI.

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:background=*"@color/blue"*

android:orientation=*"vertical"* >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:layout\_weight=*"1"*

android:text=*"This is Fragment 1"*

android:textStyle=*"bold"* />

</LinearLayout>

1. Create a file called fragment\_two.xml in /res/layout folder. Use the following settings for the UI.

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:background=*"@color/yelow"*

android:orientation=*"vertical"* >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:text=*"This is Fragment 2"*

android:textStyle=*"bold"* />

</LinearLayout>

1. Create a class called FragmentOne.java under /src folder. Use the following source code.

**public** **class** FragmentOne **extends** Fragment

{

**public** View onCreateView(LayoutInflater inflater,ViewGroup container, Bundle savedInstanceState)

{

//Inflate the layout for this fragment

**return** inflater.inflate(R.layout.***fragment\_one***, container, **false**);

}

}

1. Create a class called FragmentTwo.java under /src folder. Use the following source code

**public** **class** FragmentTwo **extends** Fragment

{

**public** View onCreateView(LayoutInflater inflater,ViewGroup container, Bundle savedInstanceState)

{

//Inflate the layout for this fragment

**return** inflater.inflate(R.layout.***fragment\_one***, container, **false**);

}

}.

1. In the MainActivity.java, override the OnCreate() function as below:

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_main***);

}

**public** **void** selectFrag(View view)

{

Fragment fr;

**if**(view == findViewById(R.id.***button2***))

{

fr = **new** FragmentTwo();

}**else**

{

fr = **new** FragmentOne();

}

FragmentManager fm = getFragmentManager();

FragmentTransaction fragmentTransaction = fm.beginTransaction();

fragmentTransaction.replace(R.id.***fragment\_place***, fr);

fragmentTransaction.commit();

}

1. After make sure all the coding appropriate done, you may compile and run your coding .