ViewPager and Dialog-12

In this assignment, you will create a new activity to host CrimeFragment. This activity's layout will consist of an instance of ViewPager. Adding a ViewPager to your UI lets users navigate between list items by swiping across the screen to "page" forward or backward through the crimes. You will also add a dialog in which users can change the date of a crime. Pressing the date button in CrimeFragment will present this dialog on Lollipop and later.

1. Go to GitHub and create a new repository for your assignment. Also add your instructor as a collaborator.

2. Open your CriminalIntent app in Android Studio. Ensure you have committed your previous lab and submitted the commit ID before moving forward. Under the VCS > Git > Remotes menu item, remove your old remote and add your new repository remote.

3. CrimePagerActivity will be a subclass of AppCompatActivity. It will create and manage the ViewPager. Create a new class named CrimePagerActivity. Make its superclass AppCompatActivity and set up the view for the activity.

   ```
   public class CrimePagerActivity extends AppCompatActivity {
       @Override
       protected void onCreate(Bundle savedInstanceState) {
           super.onCreate(savedInstanceState);
           setContentView(R.layout.activity_crime_pager);
       }
   }
   ```

4. The layout file does not yet exist. Create a new layout file in res/layout/ and name it activity_crime_pager. Make its root view a ViewPager and give it the attributes shown below. Notice that you must use ViewPager's full package name (androidx.viewpager.widget.ViewPager). You use ViewPager's full package name when adding it to the layout file because the ViewPager class is from the support library. Unlike Fragment, ViewPager is only available in the support library; there is not a "standard" ViewPager class in a later SDK.

   ```
   <?xml version="1.0" encoding="utf-8"?>
   <androidx.viewpager.widget.ViewPager
       android:id="@+id/crime_view_pager"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       xmlns:android="http://schemas.android.com/apk/res/android"/>
   ```

5. In CrimePagerActivity, set the ViewPager's pager adapter and implement its getCount() and getItem(int) methods.

```
public class CrimePagerActivity extends AppCompatActivity {
   private ViewPager mViewPager;
   private List<Crime> mCrimes;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_crime_pager);

      mViewPager = (ViewPager) findViewById(R.id.crime_view_pager);

      mCrimes = CrimeLab.get(this).getCrimes();
      FragmentManager fragmentManager = getSupportFragmentManager();
      mViewPager.setAdapter(new FragmentStatePagerAdapter(fragmentManager) {

         @Override
         public Fragment getItem(int position) {
            Crime crime = mCrimes.get(position);
            return CrimeFragment.newInstance(crime.getId());
         }

         @Override
         public int getCount() {
            return mCrimes.size();
         }
      });
   }
}
```

6. Now you can begin the process of decommissioning CrimeActivity and putting CrimePagerActivity in its place. First, add a newIntent method to CrimePagerActivity along with an extra for the crime ID.

```
public class CrimePagerActivity extends AppCompatActivity {
  private static final String EXTRA_CRIME_ID =
        "com.bignerdranch.android.criminalintent.crime_id";

  private ViewPager mViewPager;
  private List<Crime> mCrimes;

  public static Intent newIntent(Context packageContext, UUID crimeId) {
    Intent intent = new Intent(packageContext, CrimePagerActivity.class);
    intent.putExtra(EXTRA_CRIME_ID, crimeId);
    return intent;
  }

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_crime_pager);

    UUID crimeId = (UUID) getIntent()
        .getSerializableExtra(EXTRA_CRIME_ID);
    ...
  }
}
```

7. Now, you want pressing a list item in CrimeListFragment to start an instance of CrimePagerActivity instead of CrimeActivity. Return to CrimeListFragment.java and modify CrimeHolder.onClick(View) to start a CrimePagerActivity.

```
private class CrimeHolder extends RecyclerView.ViewHolder
    implements View.OnClickListener {
  ...
  @Override
  public void onClick(View view) {
    ~~Intent intent = CrimeActivity.newIntent(getActivity(), mCrime.getId());~~
    Intent intent = CrimePagerActivity.newIntent(getActivity(), mCrime.getId());
    startActivity(intent);
  }
}
```

8. You also need to add CrimePagerActivity to the manifest so that the OS can start it. While you are in the manifest, remove CrimeActivity's declaration. To accomplish this, you can just rename the CrimeActivity to CrimePagerActivity in the manifest.

```
<manifest ...>
   ...
   <application ...>
   ...
      <activity
~~         android:name=".CrimeActivity"~~
         android:name=".CrimePagerActivity">
      </activity>
   …
```

9. Finally, to keep your project tidy, delete CrimeActivity.java from the project tool window.

10. Run CriminalIntent. Press Crime #0 to view its details. Then swipe left and right to browse the crimes. Notice that the paging is smooth and there is no delay in loading. By default, ViewPager loads the item currently onscreen plus one neighboring page in each direction so that the response to a swipe is immediate. You can tweak how many neighboring pages are loaded by calling setOffscreenPageLimit(int).

11. At the end of CrimePagerActivity.onCreate(Bundle), find the index of the crime to display by looping through and checking each crime's ID. When you find the Crime instance whose mId matches the crimeId in the intent extra, set the current item to the index of that Crime.

```
public class CrimePagerActivity extends AppCompatActivity {
   ...
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      ...
      FragmentManager fragmentManager = getSupportFragmentManager();
      mViewPager.setAdapter(new FragmentStatePagerAdapter(fragmentManager) {
         ...
      });

      for (int i = 0; i < mCrimes.size(); i++) {
         if (mCrimes.get(i).getId().equals(crimeId)) {
            mViewPager.setCurrentItem(i);
            break;
         }
      }
   }
}
```

12. Run CriminalIntent again. Selecting any list item should display the details of the correct Crime. And that is it. Your ViewPager is now fully armed and operational.

13. Add a string resource for your dialog.

```
<resources>
   ...
   <string name="crime_solved_label">Solved</string>
   <string name="date_picker_title">Date of crime:</string>

</resources>
```

14. Create a new class named DatePickerFragment and make its superclass DialogFragment. Be sure to choose the support library's version of DialogFragment: androidx.fragment.app.DialogFragment. DialogFragment includes the following method:

```
public Dialog onCreateDialog(Bundle savedInstanceState)
```

15. The FragmentManager of the hosting activity calls this method as part of putting the DialogFragment onscreen. In DatePickerFragment.java, add an implementation of onCreateDialog(Bundle) that builds an AlertDialog with a title and one OK button. (You will add the DatePicker widget later.) Be sure that the version of AlertDialog that you import is the AppCompat version: androidx.appcompat.AlertDialog.

```
public class DatePickerFragment extends DialogFragment {
   @Override
   public Dialog onCreateDialog(Bundle savedInstanceState) {
      return new AlertDialog.Builder(getActivity())
         .setTitle(R.string.date_picker_title)
         .setPositiveButton(android.R.string.ok, null)
         .create();
   }
}
```

16. In this implementation, you use the AlertDialog.Builder class, which provides a fluent interface for constructing an AlertDialog instance. First, you pass a Context into the AlertDialog.Builder constructor, which returns an instance of AlertDialog.Builder. Next, you call two AlertDialog.Builder methods to configure your dialog:

```
public AlertDialog.Builder setTitle(int titleId)
public AlertDialog.Builder setPositiveButton(int textId, DialogInterface.OnClickListener listener)
```

17. In CrimeFragment, add a constant for the DatePickerFragment's tag. Then, in onCreateView(…), remove the code that disables the date button and set a View.OnClickListener that shows a DatePickerFragment when the date button is pressed.

```
public class CrimeFragment extends Fragment {

   private static final String ARG_CRIME_ID = "crime_id";
   private static final String DIALOG_DATE = "DialogDate";
   ...
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
         Bundle savedInstanceState) {
      ...
      mDateButton = (Button) v.findViewById(R.id.crime_date);
      mDateButton.setText(mCrime.getDate().toString());
      mDateButton.setEnabled(false);
      mDateButton.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            FragmentManager manager = getFragmentManager();
            DatePickerFragment dialog = new DatePickerFragment();
            dialog.show(manager, DIALOG_DATE);
         }
      });

      mSolvedCheckBox = (CheckBox) v.findViewById(R.id.crime_solved);
      ...
      return v;
   }
}
```

18. Run CriminalIntent and press the date button to see the dialog.

19. In the project tool window, create a new layout resource file named dialog_date.xml and make its root element DatePicker. This layout will consist of a single View object – a DatePicker – that you will inflate and pass into setView(…). Configure the DatePicker as shown below:

DatePicker

```
xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/dialog_date_picker"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:calendarViewShown="false"
```

20. In DatePickerFragment.onCreateDialog(Bundle), inflate this view and then set it on the dialog.

```
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {
  View v = LayoutInflater.from(getActivity())
    .inflate(R.layout.dialog_date, null);

  return new AlertDialog.Builder(getActivity())
    .setView(v)
    .setTitle(R.string.date_picker_title)
    .setPositiveButton(android.R.string.ok, null)
    .create();
}
```

21. Run CriminalIntent. Press the date button to confirm that the dialog now presents a DatePicker. As long as you are running Lollipop or later, you will see a calendar picker.

22. To get data into your DatePickerFragment, you are going to stash the date in DatePickerFragment's arguments bundle, where the DatePickerFragment can access it. Creating and setting fragment arguments is typically done in a newInstance() method that replaces the fragment constructor. In DatePickerFragment.java, add a newInstance(Date) method.

```
public class DatePickerFragment extends DialogFragment {

  private static final String ARG_DATE = "date";

  private DatePicker mDatePicker;

  public static DatePickerFragment newInstance(Date date) {
    Bundle args = new Bundle();
    args.putSerializable(ARG_DATE, date);

    DatePickerFragment fragment = new DatePickerFragment();
    fragment.setArguments(args);
    return fragment;
  }
  ...
}
```

23. In CrimeFragment, remove the call to the DatePickerFragment constructor and replace it with a call to DatePickerFragment.newInstance(Date).

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
        Bundle savedInstanceState) {
    ...
    mDateButton = (Button)v.findViewById(R.id.crime_date);
    mDateButton.setText(mCrime.getDate().toString());
    mDateButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            FragmentManager manager = getFragmentManager();
            ~~DatePickerFragment dialog = new DatePickerFragment();~~
            DatePickerFragment dialog = DatePickerFragment
                .newInstance(mCrime.getDate());
            dialog.show(manager, DIALOG_DATE);
        }
    });
    ...
    return v;
}
```

24. In onCreateDialog(Bundle), get the Date from the arguments and use it and a Calendar to initialize the DatePicker.

```
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {
    Date date = (Date) getArguments().getSerializable(ARG_DATE);

    Calendar calendar = Calendar.getInstance();
    calendar.setTime(date);
    int year = calendar.get(Calendar.YEAR);
    int month = calendar.get(Calendar.MONTH);
    int day = calendar.get(Calendar.DAY_OF_MONTH);

    View v = LayoutInflater.from(getActivity()).inflate(R.layout.dialog_date, null);

    mDatePicker = (DatePicker) v.findViewById(R.id.dialog_date_picker);
    mDatePicker.init(year, month, day, null);

    return new AlertDialog.Builder(getActivity()).setView(v)
            .setTitle(R.string.date_picker_title)
            .setPositiveButton(android.R.string.ok, null)
            .create();
}
```

25. In CrimeFragment.java, create a constant for the request code and then make CrimeFragment the target fragment of the DatePickerFragment instance.

```
public class CrimeFragment extends Fragment {
   private static final String ARG_CRIME_ID = "crime_id";
   private static final String DIALOG_DATE = "DialogDate";

   private static final int REQUEST_DATE = 0;
   ...
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
         Bundle savedInstanceState) {
      ...
      mDateButton.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            FragmentManager manager = getFragmentManager();
            DatePickerFragment dialog = DatePickerFragment
                  .newInstance(mCrime.getDate());
            dialog.setTargetFragment(CrimeFragment.this, REQUEST_DATE);
            dialog.show(manager, DIALOG_DATE);
         }
      });
      ...
      return v;
   }
}
```

26. In DatePickerFragment, create a private method that creates an intent, puts the date on it as an extra, and then calls CrimeFragment.onActivityResult(…).

```
public class DatePickerFragment extends DialogFragment {
   public static final String EXTRA_DATE = "com.bignerdranch.android.criminalintent.date";

   private static final String ARG_DATE = "date";
   ...
   @Override
   public Dialog onCreateDialog(Bundle savedInstanceState) {
      ...
   }
   private void sendResult(int resultCode, Date date) {
      if (getTargetFragment() == null) {
         return;
      }
      Intent intent = new Intent();
      intent.putExtra(EXTRA_DATE, date);

      getTargetFragment().onActivityResult(getTargetRequestCode(), resultCode, intent);
   }
}
```

27. Let's use sendResult(…) method. When the user presses the positive button in the dialog, you want to retrieve the date from the DatePicker and send the result back to CrimeFragment. In onCreateDialog(…), replace the null parameter of setPositiveButton(…) with an implementation of DialogInterface.OnClickListener that retrieves the selected date and calls sendResult(…).

```
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {
   ...
   return new AlertDialog.Builder(getActivity()).setView(v)
      .setTitle(R.string.date_picker_title)
      .setPositiveButton(android.R.string.ok, null);
      .setPositiveButton(android.R.string.ok,
            new DialogInterface.OnClickListener() {
               @Override
               public void onClick(DialogInterface dialog, int which) {
                  int year = mDatePicker.getYear();
                  int month = mDatePicker.getMonth();
                  int day = mDatePicker.getDayOfMonth();
                  Date date = new GregorianCalendar(year, month, day).getTime();
                  sendResult(Activity.RESULT_OK, date);
               }
      })
      .create();
}
```

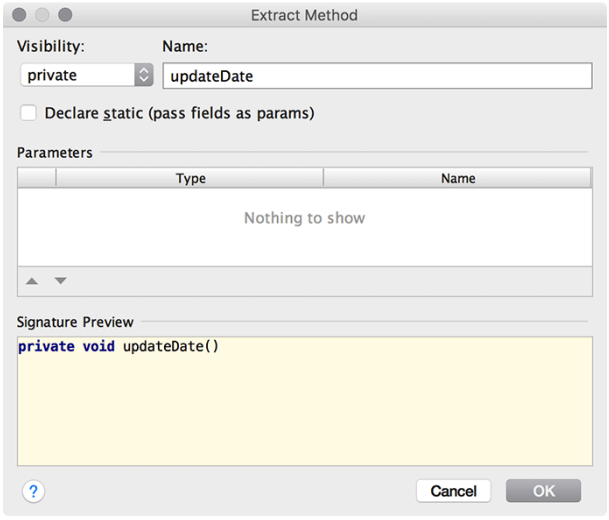
28. In CrimeFragment, override onActivityResult(…) to retrieve the extra, set the date on the Crime, and refresh the text of the date button.

```
public class CrimeFragment extends Fragment {
   ...
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
         Bundle savedInstanceState) {
      ...
   }

   @Override
   public void onActivityResult(int requestCode, int resultCode, Intent data) {
      if (resultCode != Activity.RESULT_OK) {
         return;
      }

      if (requestCode == REQUEST_DATE) {
         Date date = (Date) data
               .getSerializableExtra(DatePickerFragment.EXTRA_DATE);
         mCrime.setDate(date);
         mDateButton.setText(mCrime.getDate().toString());
      }
   }
}
```

29. The code that sets the button's text is identical to code you call in onCreateView(…). To avoid setting the text in two places, encapsulate this code in a private updateDate() method and then call it in onCreateView(…) and onActivityResult(…). You could do this by hand or you can have Android Studio do it for you. Highlight the entire line of code that sets mDateButton's text.

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
   if (resultCode != Activity.RESULT_OK) {
      return;
   }

   if (requestCode == REQUEST_DATE) {
      Date date = (Date) data
            .getSerializableExtra(DatePickerFragment.EXTRA_DATE);
      mCrime.setDate(date);
      mDateButton.setText(mCrime.getDate().toString());
   }
}
```

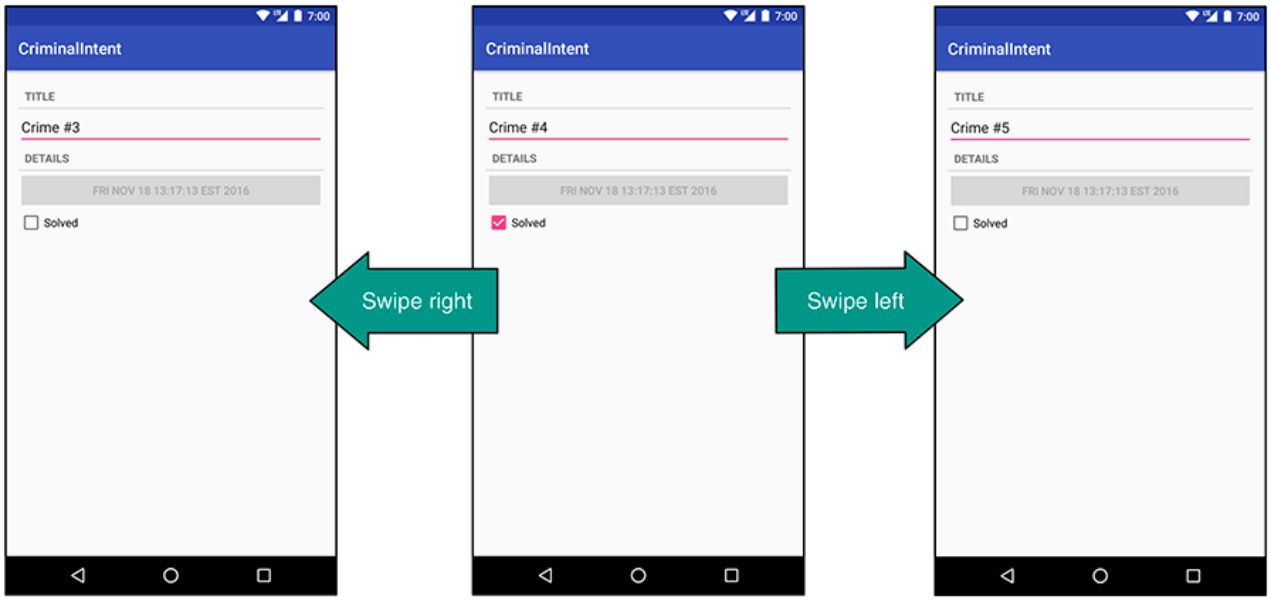
30. Right-click and select Refactor → Extract → Method...

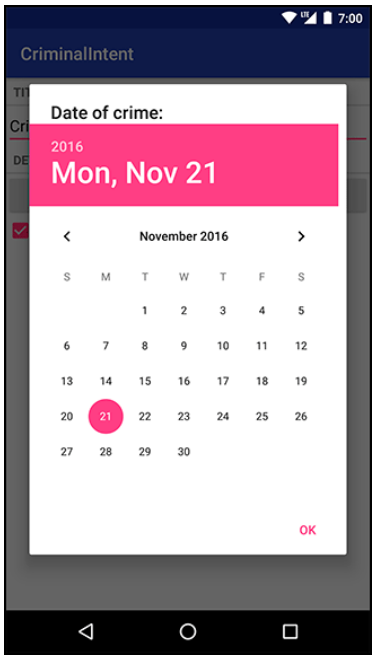


31. Make the method private and name it updateDate. Click OK and Android Studio will tell you that it has found one other place where this line of code was used. Click Yes to allow Android Studio to update the other reference, then verify that your code is now extracted to a single updateDate() method.

```
public class CrimeFragment extends Fragment {
    ...
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment_crime, container, false);
        ...
        mDateButton = (Button) v.findViewById(R.id.crime_date);
        updateDate();
        ...
    }
    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (resultCode != Activity.RESULT_OK) {
            return;
        }
        if (requestCode == REQUEST_DATE) {
            Date date = (Date) data.getSerializableExtra(DatePickerFragment.EXTRA_DATE);
            mCrime.setDate(date);
            updateDate();
        }
    }
    private void updateDate() {
        mDateButton.setText(mCrime.getDate().toString());
    }
}
```

Final App


7:00
CriminalIntent
TITLE
Crime #3
DETAILS
FRI NOV 18 13:17:13 EST 2016
Solved
Swipe right
CriminalIntent
TITLE
Crime #4
DETAILS
FRI NOV 18 13:17:13 EST 2016
Solved
Swipe left
CriminalIntent
TITLE
Crime #5
DETAILS
FRI NOV 18 13:17:13 EST 2016
Solved


7:00
CriminalIntent
Date of crime:
2016
Mon, Nov 21
November 2016
S M T W T F S
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
OK