Mobile App Development

Exercises: Todo List ViewPager App

The Todo List App exercise implemented a base App with the TodoListActivity and TodoActivity controllers and their respective fragments. This adhered to the principle of dynamically building the view with fragment transations. It also demonstrated good practice by using a Bundle for the `todoId` in the fragment rather than accessing the `intent` directly in the Activity; hence, decoupling the fragment with its own `argument bundle` and therby making it reusable.

This exercise add a new ViewPager to the App which allows for navigating between list items by swiping accross the screen to go forward and backward through the detail of list items.

You could either replicate the `TodoListApp` or create a new git branch to the existing `TodoListApp` implementation. If in doubt, then recreating the App may be good practice.

The source code can be found at https://github.com/ebbi/TodoListApp/. Please use this as reference only. The following instructions will build the same git branch leading to the final prototype App with a toolbar.

See the following documentation on the ViewPager
https://developer.android.com/training/animation/screen-slide.html

list view	dettail view
	

Todo List ViewPager App

Create a git branch for implementing the ViewPager

```
git checkout master
git status

git branch viewpager
git checkout viewpager
git status
```

Create a new Activity class and a ViewPager layout file

```
Create a new res/layout/activity_todo_pager file and insert the following viewpager layout:
```

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.view.ViewPager
  xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/todo_view_pager"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
</android.support.v4.view.ViewPager>
```

Create the controller **Activity class with** the following code:

```
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentStatePagerAdapter;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;

import java.util.List;
import java.util.UUID;

public class TodoPagerActivity extends AppCompatActivity {

    private static final String EXTRA_TODO_ID = "todo_id";

    private ViewPager mViewPager;
    private List<Todo> mTodos;

    public static Intent newIntent(Context packageContext, UUID todoId){
        Intent intent = new Intent(packageContext, TodoPagerActivity.class);
        intent.putExtra(EXTRA_TODO_ID, todoId);
        return intent;
    }

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

        UUID todoId = (UUID) getIntent().getSerializableExtra(EXTRA_TODO_ID);

        mViewPager = findViewById(R.id.todo_view_pager);

        mTodos = TodoModel.get(this).getTodos();

        FragmentManager fragmentManager = getSupportFragmentManager();
        mViewPager.setAdapter(new FragmentStatePagerAdapter(fragmentManager) {
            @Override
            public Fragment getItem(int position) {
                Todo todo = mTodos.get(position);
                return TodoFragment.newInstance(todo.getId());
            }

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

        for (int i = 0; i < mTodos.size(); i++){
```

```
                if (mTodos.get(i).getId().equals(todoId)) {
                    mViewPager.setCurrentItem(i);
                    break;
                }
            }
        }
    }
```

Similar to the `RecyclerViewer`, the `ViewPager` requires a `PagerAdapter`. The detail can be complex and we use the `FragmentStatePagerAdapter`, a subclass of `PagerAdapter` to handle the detail and we only need to overide two simple methods for `getCount()` and `getItem(int)`. Essentially, when the `getItem(int)` is called, it will return a todo fragment to display the todo at that poistion.

By default, the `ViewPager` shows the first item in the `displayAdapter`, the `for` loop finds the selected todo index and sets it to be the current item.

Note, the `newIntent` method with the extra for the `todoId` and the change to the `TodoPagerActivity`

The `TodoListFragment` needs to be updated to start an instance of the `TodoPagerActivity` and not the previous intent which was to start an instance of the `TodoActivity`. Update the `onClick` event for `TodoHolder` to the following intent.

```
// Intent intent = TodoActivity.newIntent(getActivity(), mTodo.getId());
  Intent intent = TodoPagerActivity.newIntent(getActivity(), mTodo.getId());
```

Finally, edit the `manifest.xml` file and update it so the OS activity manager can start the `TodoPagerActivity`

```
<activity
    android:name=".TodoPagerActivity"
    android:parentActivityName=".TodoListActivity">
</activity>

Run the App
```

The TodoActivity has been replaced by TodoPagerActivity in the manifest file; as such the TodoActivity is now decomissioned for the new UI. Selecting any todoItem should display the item and you can swipe back and forth.

Submit and merge the toolbar git branch

```
git add .
git commit -am "ViewPager complete"

git checkout master
git merge viewpager
git status
```

For challenge, try adding a `first` and `last` button to the navigation.