Dublin City University
National Institute for Digital Learning
Open Education Unit

Module: SDA: Software development mobile applicaton A

Programme(s):

BSc in Information Technology
HDIP in SCi in Software Development

Assignment 4 2018-2019

Software Development A

Module Code: SDA

Module Title:	Software Development A: Mobile Application Development
Assignment Type:	Coding
Assignment Title:	Assignment 4

Assignment Introduction

The purpose of this assignment is threefold

- To familiarize students with Advanced User Interface features and Design Patterns i.e. Fragments, ToolBars, Menus
- To familiarize students with basic data storage e.g. shared preferences
- To prepare students for their mobile application project

This assignment serves to bring together the different skills and knowledge obtained from previous assignments while also testing new features, and your research process. The application that you will develop is a Clothing Order Application.

Specific Requirements

- Review the assignment instructions and perform additional research required to complete the application.
- Provide formatted and commented code in your Java and XML files, as well as Java Doc comments
- Provide code that is internationalised i.e. no hard-coded strings in your **XML** layout files.

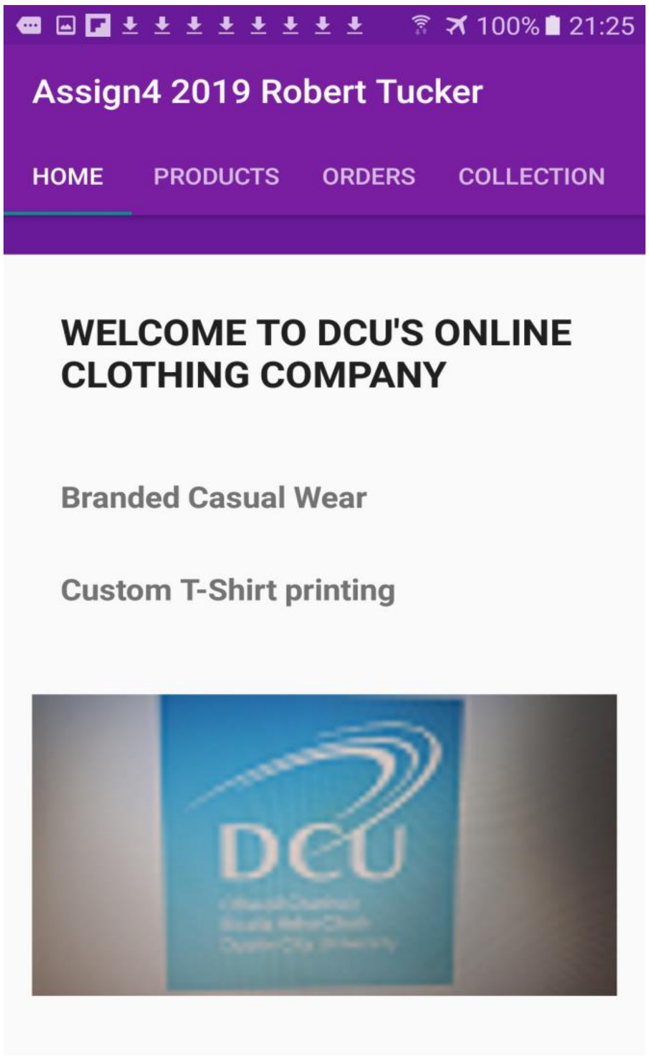
Deliverables

- A zipfile that contains your android studio project, with all your files i.e. a zip up of your project
- The zipfile will be submitted through Loop and all your project code should be checked into your **assign4** folder in your repository on GitHub.
- A screencast of between 5-10 minutes duration posted to your DCU YouTube account summarising the working and your understanding of the SDA sample application 'actionTabsFragPager'

- A screencast of less than 5 minutes duration posted to your DCU YouTube account of your completed app showing:
  - the app working in an emulator
  - describing and showing visually any changes and design choices made
- A well laid-out document containing all appropriate links, screenshots, resources used, and description of issue resolution and outstanding bugs as if delivering to a client
- A git repository link from which your project can be cloned and built without issue in Android Studio – you should verify this. Your GitHub repository must contain all necessary file e.g. gradle, xml, java, png files etc.

  N.B. A zipfile only of the project pushed to github will get no marks for the git section and

  N.B github repository clones missing java files or gradle files which do not build and run automatically will not be assessed functionally.



WELCOME TO DCU'S ONLINE CLOTHING COMPANY

Branded Casual Wear

Custom T-Shirt printing

Figure 1: A tabbed layout and the HOME Fragment

Assignment Instructions:

Purpose:

To show understanding of using tabs, fragments, toolbars and menu's by studying the '**actionTabsFragPager'** demo code.

To provide an application that builds upon your Assignment 3 application which -

1. uses up-to-date Android Design Patterns (ToolBar, Swiping and Tabs)
2. advanced User Interface features and
3. shares data between fragments.

You can style this app as you choose, i.e. background colours, text colour and size, positions of views etc but you are only required to follow the same style/theme as your delivered Assignment 3 app.

A summary of key points of the application you are to develop follows:

1. The application is titled: "Assign 4 2018 <Your Name>"
2. The first screen will contain a toolbar and 4 tabs entitled:

   HOME, PRODUCTS, ORDERS, COLLECTION

3. Each tab will be associated with a 'Fragment' java class and a layout xml file.
4. The app will launch with the HOME tab presented to the user.
5. The application can be navigated in two ways; tapping tabs on the toolbar and/or by swiping the screens
6. An example HOME tab/fragment is shown in Figure 1 above, however you should adapt this as you see fit to suit your Assignment 3 application style, theme, colour, image etc to provide an introductory screen to your Assignment 3 App.
7. The PRODUCT tab is the same content and layout as the Product List Activity in Assignment 3 except using a tab and fragment this time.
8. The ORDER tab is the same content and layout as the Order T-Shirt Activity in Assignment 3 except using a tab and fragment this time.
9. The COLLECTION tab contains a scrollable list of collection location information. Each item in the list contains 3 entries - a Store area e.g. 'Swords', an address and a phone number
10. You can reuse and adapt the Product List layout/adapter code with 3 fields (but no icon required), however you are free to present the Collection list in whatever way you wish in keeping with your app theme etc noting that you will need to be able to extract the location name when the user taps on an element in the same way as for the toast message in the Product list

11. You should include a way to keep track of items selected in the PRODUCT tab and the COLLECTION tab. - **Use SharedPreferences** to achieve this. (See Unit 13 Storage – Using Internal Storage)

    When a user taps on a product in the list on the PRODUCT tab a toast message pops up saying '<product-name> added' and this product detail is added to your stored list to be included in the order email.

    When a user taps on a location in the COLLECTION tab a toast message pops up saying '<location> selected for collection' and this location is added to your stored list to be included in the order email.

12. If no collection location was selected a delivery address is expected in the relevant ORDERS tab view.

13. The ORDERS tab also allows (as before) to take a photo and have a custom T-Shirt printed with your order. This time if a photo is taken it will be loaded into the ImageView object on the ORDERS screen but it is optional for the user to do this.

14. You are allowed to use the actionTabsFragPager demo code, which will provide you with a template for creating tabs and swiping views.
    Be sure to credit this source in your class files.

Tips and Hints:

1. To access your views in a fragment using **findViewById**, your code will look something like this.

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,  Bundle savedInstanceState)
{
        View root = inflater.inflate(R.layout.products_fragment, container, false);
        ListView listView = (ListView) root.findViewById(R.id.listview_products);   ….
        etc….
        return root
}
```

2. You may also have to change how you implemented your listeners especially if you used the **onClick** attribute in your XML: and change calls from "**this**" to getActivity(), as "**this**" is now a fragment (and not an activity). Use onClickListener type methods

3. Check the Manifest file has the following values:

```
<application
        android:theme="@style/AppTheme">
          <more properties>
    <activity
                android:theme="@style/AppTheme.NoActionBar">
              <more properties>
```

Mark Weightings

	Fail	Pass	Good	V Good	Excellent
Code Functionality 40%	The application does not run. The required functionality is not implemented	Code compiles and runs, Attempts have been made to implement some of the required functionality but the app does not meet all of the required functionality, the UI is missing some core features. There are bugs in the code.	Attempts made to implement all/most of the required functionality but app does not perform exactly as described, or the app does not provide all the components necessary for the required functionality. No attempt made to style the app There are some minor bugs in the code.	All documented functionality implemented and application looks professional: stylish with a consistent look and feel Student successfully integrated all the required components into their application (excluding the summary in preference fragment) The required functionality could be written in a more concise and effective way or by using different classes or methods. There are some minor bugs in the code, which have been identified, but not yet resolved. (Tell me briefly about these in your class comments in your code i.e. start of the file).	The application performs as expected. The UI is styled in a professional and consistent manner All required functionality implemented in an efficient way. The most effective Android/Java classes are used to implement the required features. Student displayed a thorough understanding of how to use appropriately the different Android classes and features Evidence of research outside core functionality in relation to making app bug free.

Code Style and structure and layout, theme, style, colour, images **20%**	No comments in code. Poorly formatted. The naming of variables and/or methods are too cryptic, inconsistent, or do not follow any convention. Strings hardcoded No debug information present Poor design	Code is missing appropriate comments . Variables inconsistently named. The code is inconsistently formatted. Some debug information present but their statements not informative. Methods are too long. Code is not very maintainable Basic design Colors not following guidelines	Code is commented, but some comments too long, and take from code comprehension Code lacks elegance in that more lines are used than necessary to perform tasks. Code not easy to read Some strings are hardcoded. Some debug information present, but statements could be more informative. Methods are too long. Design pattern used but not uniform	Code is commented and contains appropriate debug information. Variables are consistently named, and no hard-coded strings are present but code lacks elegance. The code is not consistently simple, concise and succinct. Consistent design pattern and color guide	Code is commented (where appropriate) and formatted. Comments are short and informative. Code is concise and well written All names are easy to understand, consistent (in both wording and capitalization) Code is elegant i.e. it is clean, simple, easy to read and succinct. Code is easy to maintain No hard-coded strings present Error handling code present. The code is very maintainable. Consistent and professional design patterns used
Screencasts 20% **(10% each)**	No accompanying screencast link sent.	screencast incomplete, does not summarise the purpose and work undertaken in the labs	Screencasts are adequate, but are not clear and concise	Screencasts are informative, and well structured. It is clear, students understood the work undertaken and the purpose of the lab.	Screencasts are concise, well presented and well structured. It is evident that the student understood the required functionality associated with list views and Adapters. The 'Product' and the 'Order T-shirt' activities are explained using a clear, concise and informative voice
10% GITHUB	No code pushed to Github account and noguihub account created	Account created on github but no code submitted	Account created on github and partial code submitted to github account	Full code submitted to github regularly.	Full code and good commit messages submitted to github **regularly** or other evidence of git feature use
10% Document with resources screenshots issues etc	No document provided	Document provided with little info and no structure	Well structured document with partial info	Well structured document with nearly all required info	**Well structured document with all relevant info and well presented**