



College of Engineering, Construction and Living Sciences  
Bachelor of Information Technology  
IN721: Design and Development of Applications for Mobile Devices  
Level 7, Credits 15  
**Practical 01: Touch & Input**

### Assessment Table

Assessment Activity	Weighting	Learning Outcomes	Assessment Grading Scheme	Completion Requirements
Practicals	25%	1, 3, 4	CRA	Cumulative
Language Translator	20%	1, 3, 4	CRA	Cumulative
Wishlist	25%	1, 3, 4	CRA	Cumulative
Exams 1-5	30%	2, 3, 4	CRA	Cumulative

### Conditions of Assessment

This assessment will need to be completed by Friday, 12 June 2020.

### Pass Criteria

This assessment is criterion-referenced with a cumulative pass mark of 50%.

### Submission Details

You must submit your program files via **GitHub Classroom**. Here is the link to the repository you will be using for your submission – <https://classroom.github.com/a/ifyWTPlw>. For ease of marking, please submit the marking sheet with your name & student id number via **Microsoft Teams** under the **Assignments** tab.

### Authenticity

All parts of your submitted assessment must be completely your work and any references must be cited appropriately.

## Policy on Submissions, Extensions, Resubmissions & Resits

The school's process concerning **Submissions, Extensions, Resubmissions and Resits** complies with Otago Polytechnic policies. Students can view policies on the Otago Polytechnic website located at <https://www.op.ac.nz/about-us/governance-and-management/policies>.

### Extensions

Please familiarise yourself with the assessment due dates. If you need an extension, please contact your lecturer before the due date. If you require more than a week's extension, a medical certificate or support letter from your manager may be needed.

### Resubmissions

Students may be requested to resubmit an assessment following a rework of part/s of the original assessment. Resubmissions are completed within a short time frame (usually no more than 5 working days) and usually must be completed within the timing of the course to which the assessment relates. Resubmissions will be available to students who have made a genuine attempt at the first assessment opportunity. The maximum grade awarded for resubmission will be C-.

## Learning Outcomes

At the successful completion of this course, students will be able to:

1. Implement complete, non-trivial, industry-standard mobile applications following sound architectural and code-quality standards.
2. Explain relevant principles of human perception and cognition and their importance to software design.
3. Identify relevant use cases for a mobile computing scenario and incorporate them into an effective user experience design.
4. Follow industry standard software engineering practice in the design of mobile applications.

## Assessment Overview

In this practical, you will complete a series of tasks covering today's lecture. This practical is worth 1% of the final mark for the Design and Development of Applications for Mobile Devices.

### Task 1

- In your **activity\_main.xml**, add the following widgets & dimension:
  - An **email** edit text that is constrained 16dp from the top, start & end of the the screen's edge
  - A **password** edit text that is constrained 16dp from the bottom of the **email** edit text & 16dp from the start & end of the screen's edge
  - A button that is constrained 16dp from the bottom of the **password** edit text & 16dp from the start & end of the screen's edge
  - A text view that is constrained 16dp from the bottom of the button & 16dp from the start & end of the screen's edge
  - Change the width of each widget to 0dp/match parent
- In your **MainActivity.kt**, add the following code:
  - An inner class called **ClickMeButtonOnClickHandler** which implements the **View.OnClickListener** interface. If you have not implemented the interface's members, click on the red light bulb then on **Implement members**
  - A dialog will display all the members associated to the interface - this interface only has one public method called **onClick()**. Click the **OK button**. If you have followed along carefully, the **onClick()** method body should be generated
  - In the **onClick()** method body, you will write code which somewhat replicates the functionality of a login form. Checking if the **email** & **password** edit text widgets are blank will be sufficient for this task
  - If either edit text widget is blank, prompt the user with an error. Don't the Java **setError()** method. Instead, use the Kotlin **error** property
  - If both edit text widgets are not blank, set the text view value to the email & password edit text value. Again, don't use the Java **setText()** method. Instead, use the Kotlin equivalent.

### Task 2

- Extend your **activity\_main.xml** by adding the following widgets & dimensions:
  - A divider that is constrained 16dp from the bottom of the text view & 16dp from the start & end of the screen's edge
  - A radio group containing three instrument radio buttons - I have chosen guitar, piano & trumpet. Adding radio buttons to a radio group can be fiddly...be patient. The radio group is constrained 16dp from the bottom of the divider & 16dp from the start & end of the screen's edge
  - A spinner that is constrained 16dp from the bottom of the radio group & 16dp from the start & end of the screen's edge. You will populate the spinner with the months of the year
  - A button that is constrained 16dp from the bottom of the spinner & 16dp from the start & end of the screen's edge
  - A text view that is constrained 16dp from the bottom of the button & 16dp from the start & end of the screen's edge. When you click the button, it should output the value from the radio group & spinner
  - Once again, change the width of each widget to 0dp/match parent

- Extend your **MainActivity.kt** by adding the following code:
  - An inner class called **EnrolButtonOnClickListener** which implements the **View.OnClickListener** interface. Like you did in task 1, click on the red light bulb then on **Implement members**
  - In the **onClick()** method body, you will write code which outputs the value of the radio group & spinner. Make sure you check if a radio button has been checked. If you don't, your application will stop/crash. Of course, you can set a radio button to be checked in your **activity\_main.xml** or at **runtime**
  - Handle all errors appropriately with a **Toast** message
  - Create a new array adapter & populate your spinner with the months of the year. Don't use `arrayOf`, instead, create a string array in **arrays.xml**

## Expected Output

- In this **practicals** directory, I have included a **expected-output** directory containing videos for each practical. I prefer to use videos instead of images
- This course is about being a creative. Don't feel like you have to always replicate my expected output...I just want make sure you complete the core functionality of each task

## Submission

- Create a new branch named 01-checkpoint within your practicals GitHub repository
- Create a new pull request and assign Grayson-Orr to review your submission
- Deadline: Friday, 12 June at 5pm

**Note:** Please don't merge your own pull request.