

6COSSC004W Mobile Native Application Development**Coursework 1 – iPhone Application (2020/21)**

Module leader	PHILIP TRWOGA
Unit	Coursework 1
Weighting:	40%
Qualifying mark	30%
Description	iPhone Unit Converter Application
Learning Outcomes Covered in this Assignment:	<p>This assignment contributes towards the following Learning Outcomes (LOs):</p> <p>LO1 understand language features and programming practice required for native development</p> <p>LO2 apply industry standard tools for design and development</p> <p>LO3 communicate and defend work by both written and oral means</p>
Handed Out:	8/02/2021
Due Date	16/03/2021
Expected deliverables	<p>Submit on Blackboard a zip file containing:</p> <p>Complete XCode solution - Swift language only</p>
Method of Submission:	<p>Electronic submission on BB via a provided link close to the submission time. The file you upload should have the following naming format:</p> <p>E.g. 6COSC004W_StudentID_firstName_lastName.zip</p>
Type of Feedback and Due Date:	<p>Formative feedback will be provided during tutorial sessions. Verbal feedback on the submitted CW will be provided during the CW presentation/viva. Students are encouraged to record this feedback at this time. Feedback is due by the 30/03/2021. Feedback shall also be given on the Blackboard.</p>

Note: All marks will remain provisional until formally agreed by an Assessment Board.

Assessment regulations

See the Assessment guidelines <https://www.westminster.ac.uk/current-students/guides-and-policies/assessment-guidelines>

for a clarification of how you are assessed, penalties and late submissions, **what constitutes plagiarism etc.**

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website <http://www.westminster.ac.uk/study/current-students/resources/academic-regulations>

Note: By submitting the work through Blackboard you are acknowledging that this is solely your own work. Any code which is not created by you MUST be clearly commented as such. Any code discovered to not have been created by you will mean that the work will be submitted to academic standards for a potential assessment offence, which may result in a zero mark in the component or whole module. This is an individual coursework so do not share code or work collaboratively with any other person in the creation of your solution. You are allowed to use code from lectures or tutorials.

6COSC004W Mobile Native Development Coursework 1 – Specification - iPhone Utility Converter App

This coursework is worth 40% of the module mark.

Please read this document **carefully** and read all footnotes. Raise any issues about this coursework early with your lecturer/tutor.

Synopsis

You are to create a unit converter application for the iPhone. The app shall convert **weights, temperature, liquid volumes, distance and speed**. It also allows the user to save the last five conversion for each unit category. The user can also select the display accuracy of the conversions.

Conversion Requirements

R1: The software shall convert between the following mass units:

Weight:

Kg, grams, ounces, pounds, stone-pounds¹

R2: The software shall convert between the following temperature units:

Temperature:

Celsius, Fahrenheit, Kelvin

R3: The software shall convert between the following length units:

Length:

metre, km, mile, cm, mm, yard, inch

R4: The software shall convert between the following units of speed:

Speed:

metres/sec, km/hour, miles/hour, nautical miles/hour (knot)

R5: The software shall convert between the following units of volume:

Volume liquid:

UK gallon, litre, UK pint, fluid ounce, millilitre

¹ Stone pounds means that weight should be displayed in stones with any remainder in pounds. For example, 162 pounds would be 11st – 8lb (11 stones and 8 pounds).

Design

A potential UI design for the app is shown below in figure 1. Note that you are free to design this app as you wish but it must meet all requirements and constraints.



Figure 1 Concept design showing main view and a custom keyboard. Note a tabbed based application is appropriate for this app² though you are free to create a different navigation design.

Operation

To use the application the user first selects one of the conversion types from the array of buttons at the top of the view. As soon as the user begins to type numbers on the keyboard the conversions shall appear in the other text fields³. If the user wishes to save the current conversion, then they must select the save icon on the

² See the supplementary notes on Blackboard – Study Materials – Extra Content

³ Hints: the best event/action to use on the **UITextField** to achieve this is **editingChanged** as this is called when any character is entered into a field.

You may wish to use a scroll view otherwise some text fields may not be accessible.

toolbar. The user can change the accuracy of the displayed conversion via the settings view.

Keyboard Requirements

The app shall have a **custom** keyboard and shall have keys 0-9, delete last entry key, decimal point, and a negative key (only used for temperatures)⁴.

History View Requirements

Figure 2 shows a potential design for the history view that is reached via a segue from the main view or from a tab bar icon.

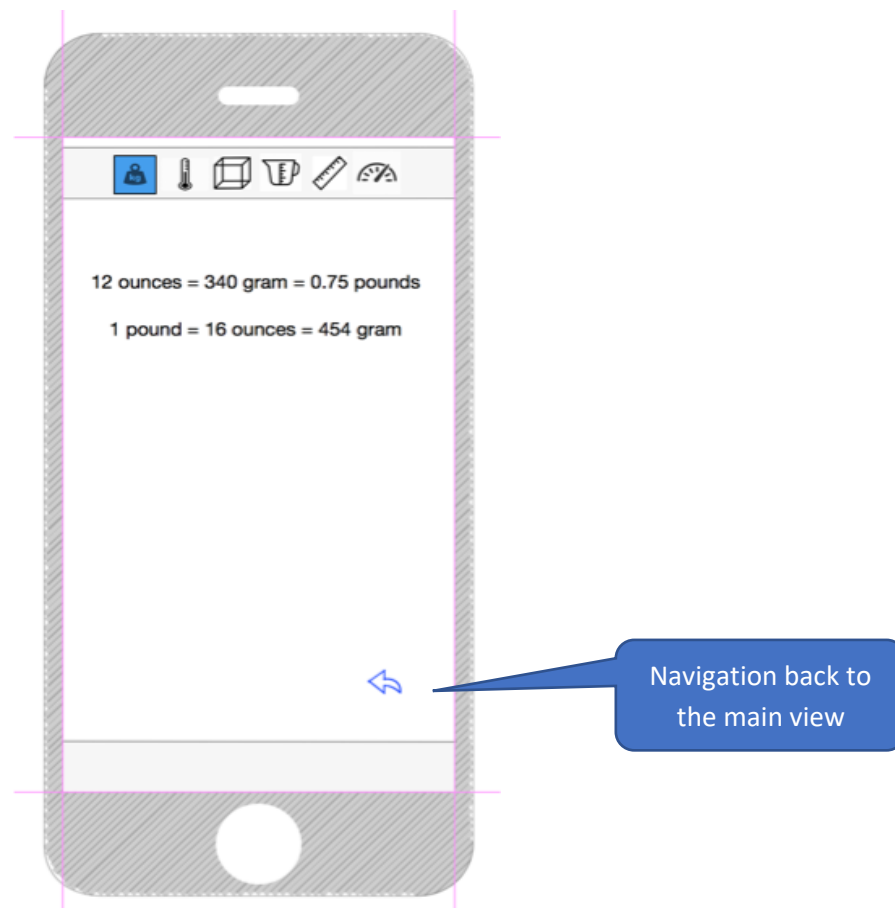



Figure 2 History view – note each unit has its own history, the sketch shows the weight category history. Note that you need to show length, weights, temp, liquids and speed.

⁴ Note you can use the decimal system keyboard for 50% of the available marks for this section.

History Requirements

- R6: The user shall be able to save the current conversion for the current unit.
- R7: The history view shall display the last 5 conversions for each conversion unit.
- R8: After 5 saves the oldest conversion shall be removed from the list (first in - first out).
- R9: The user shall be able to select the unit category in the History view*.
- R10: The user shall be able to navigate back to the main view from the History view.

Note that saving the last conversion is not automatic, and the user must select a save icon/button (for example ) and this button can be located on the toolbar or elsewhere on the view⁵.

*For example, this can be achieved by selecting one of the button icons shown in figure 2.

Settings View Requirements

R11: The user shall be able to set the precision of the display from two decimal places to four.

For example, 2.13, 2.127, 2.1268. Note that numbers to correctly rounded.

Non-functional requirements

- 1) The app shall render well in all iPhone from 8 to iPhone 12 series
- 2) The app shall only support portrait mode.
- 3) All strings must be well formatted and correctly spelt.
- 4) All units must be correctly labelled.
- 5) All conversion calculations shall be correct
- 6) Conversion history must be persistent, even when the app has been closed (this means killed and not just backgrounded)

⁵ Note that you can use **UserDefaults** for this purpose see:

<https://developer.apple.com/documentation/foundation/userdefaults>

Indicative Mark Scheme⁶

Criteria	Maximum component mark
Navigation handling (i.e., tab bar view, buttons and navigation) – for high marks this must all work correctly and responsively.	15
Keyboard – for high marks all keys including the negative must be present and it must be a custom keyboard. Half marks for using a system decimal keyboard.	20
Conversions – 10 marks for weight conversion and 5 marks each for; volume, length, speed and temperature. Must be complete and correct for high marks.	30
Conversion history – save the conversion on demand for each category. High marks for correct string formatting, correct saving of user data, and persistent storage and good useability.	20
Settings – high marks for good usability and correctness	15
Total	100

Note for high marks your work must also meet the non-functional requirements

⁶ Note that you need to present your work in a viva after the due-by-date for marking during, in which you will need to demonstrate your understanding of your code – **note your mark may be reduced if you do not demonstrate good understanding and this is at the discretion of the marking tutor**. Note also that best marks are awarded for efficient and well-written code (good coding standards) and also for the correctness of each section (meets the specified requirement exactly). Note that the viva is **compulsory** element of the coursework and you may receive a **zero mark** if you do not attend a viva and a maximum of 30%.

Note that each requirement in the indicative marks section has 4 marking categories (also see note on Usability below):

- 1) Complete and correct – available marks (as shown above)
- 2) Minor defect – approx. 70% of available marks (as shown above)
- 3) Major defect – approx. 10-50% of available marks (as shown above)
- 4) Not done or some non-functioning code – 0-10% of available marks (as shown above)

End of document