ITECH2000 Mobile Development Fundamentals

Module 2 Lab/Tute Exercises

Part 1 – Review Questions

Q1. What is the differences between a Text Box and a Label?

Q2. What are some of the properties that are common to Text Box and Label components?

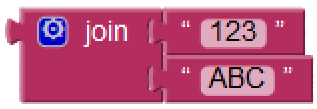
Q3. What type of component can display a graphic that you have created in photoshop, paint, or other graphics editing program?

Q4. What is meant by the term 'event' in terms of AppInventor? Give an example in relation to Button components.

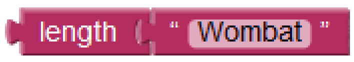
Q5. What is an action block, in relation to components in AppInventor? Give an example in relation to the Label component.

Q6. If each of the following were attached to a 'set lblOutput .Text to:' block in turn, what would end up being displayed in the label on the screen of the app (in other words, <u>what do they evaluate to</u>) ?

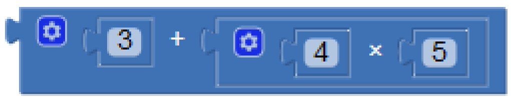
a.



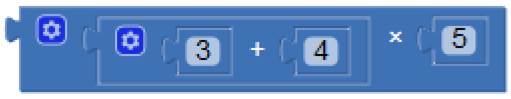
b.



c.

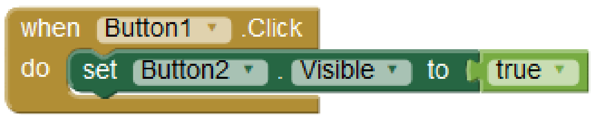


d.

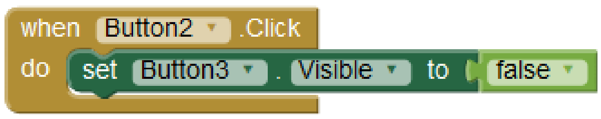


Q7. Assume there is an app with three button components, named *Button1*, *Button2*, *Button3*,

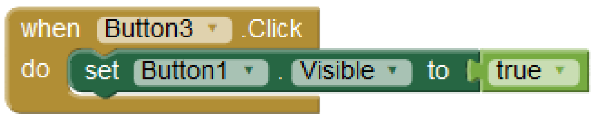
The following code blocks exist in the blocks editor view for the screen:



```
when Button1 .Click
do  set Button2 . Visible to true
```



```
when Button2 .Click
do  set Button3 . Visible to false
```



```
when Button3 .Click
do  set Button1 . Visible to true
```

a. If only button 2 is visible when the app begins, can the user cause button 1 to become visible? If so, how?
b. If only button 3 is visible when the app begins, can the user cause button 2 to become visible? If so, how?

Q8. Write (by hand on paper – see below) a code block sequence to fulfil the following requirements:

There is a textbox named *txtQuantityAdult* into which the user will enter a number specifying how many adult-priced tickets to watch a movie they want to buy.

There is a textbox named *txtQuantityChild* into which the user will enter a number specifying how many child-priced tickets to watch a movie they want to buy.

There is a button named *btnCalculate*, which when clicked will report into a label named *lblResult* the phrase "Your purchase will cost " followed by the total cost of the tickets based on the following rates: adult tickets are $17 each, and child tickets are $11 each.

Notes about writing AppInventor code by hand:

- Draw lines showing the extent and conclusion of all cradles. The content of the cradle should all be indented compared to the text that is part of the cradle-block itself.
- For any expressions inside of other expressions (such as in math expression blocks), wrap the contained expression in square brackets.
- Indicate that something is attached to another block, by using the colon after the name of the attachment point.

For example, the code provided for Button1 in question 7 would be written out in the following way:

```
when Button1 . Click
do:
|      set Button2 . Visible to:  true
|_____________________________________
```

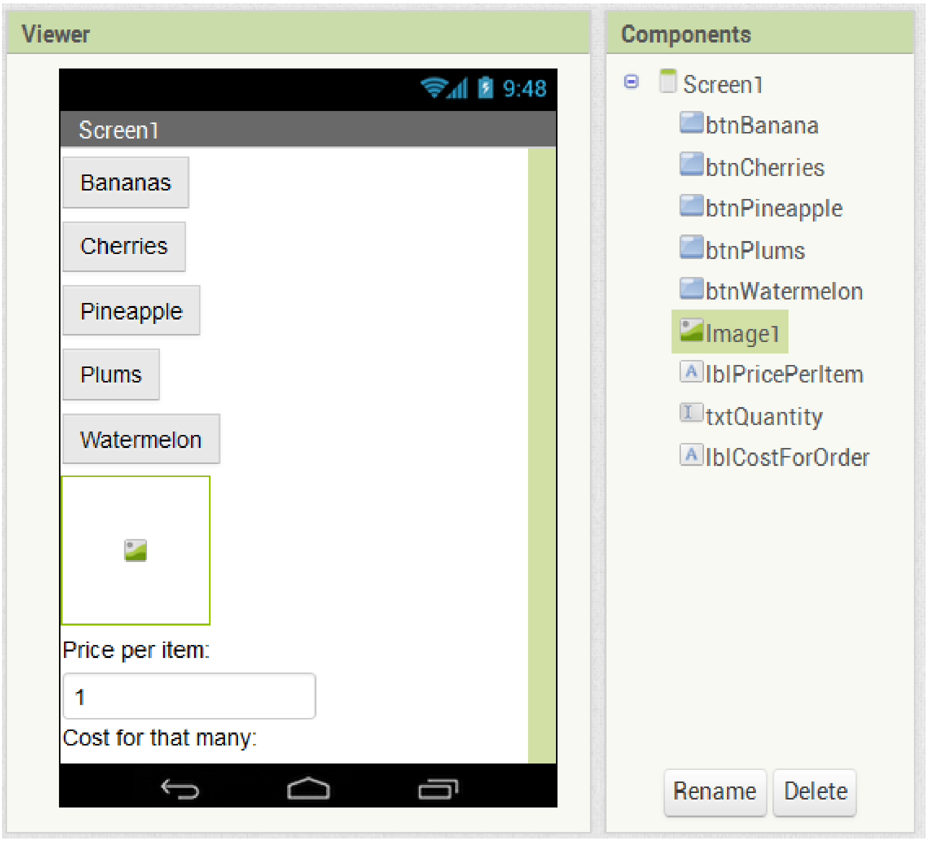
Part 2 – AppInventor Programming Tasks

Task 1 – Fruit Selector App

For this task you will make a simple app that provides a mechanism for selecting different items of fruit for sale at a greengrocer.

If working on your own computer, you may find you first need to install the Android Emulator, which can be obtained from the web page at: http://appinventor.mit.edu/explore/ai2/setup-emulator

a. Design the user interface to look like the following image (on the left side), and ensure that the names of the components are matching as shown on the right side. (Note that there are some changes to properties for some components – explained below).

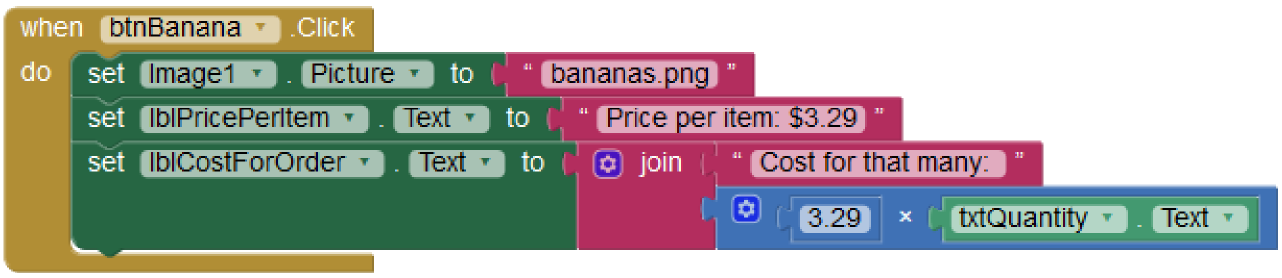


b. You need to change the properties of the Image component so that its Width is exactly 100 pixels, and its Height is exactly 100 pixels.

c. Ensure that the textbox will display the number 1 by default when the app starts, by setting its *Text* property.

d. Upload the supplied image files as the media files for this app (download them from Moodle – they are contained in a single zip file (compressed folder), which you will need to extract so that the images can be individually added to your app).

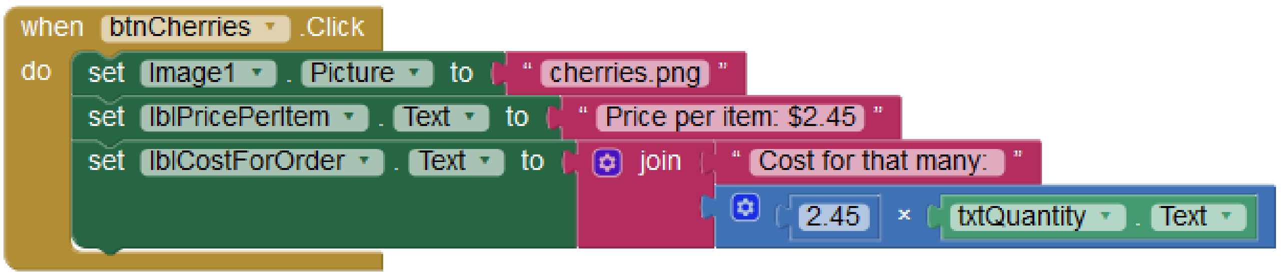
e. Switch to the Blocks view, and add the following code block sequence for the “Bananas” button’s Click event:



```
when btnBanana .Click
do  set Image1 . Picture to " bananas.png "
    set lblPricePerItem . Text to " Price per item: $3.29 "
    set lblCostForOrder . Text to join " Cost for that many: "
                                       3.29 × txtQuantity . Text
```

Once you’ve added that it would be a good point to check if the button behaves correctly, before you proceed on to further code. Try each of 1, 2 and 3 as the quantities being bought, to confirm the reported price is correct. (You may also like to try for invalid input of blank, and note what AppInventor does when something unexpected is input.)

f. Again in the Blocks view, add the following code block sequence for the “Cherries” button’s Click event:



```
when btnCherries .Click
do  set Image1 . Picture to " cherries.png "
    set lblPricePerItem . Text to " Price per item: $2.45 "
    set lblCostForOrder . Text to join " Cost for that many: "
                                       2.45 × txtQuantity . Text
```

Once you have added this, check that it also works, and also check that the “Bananas” button is still working as well.

g. Continue on to make the other three buttons respond appropriately to being clicked, by having code sequences similar to the previous ones. Use the following price information:

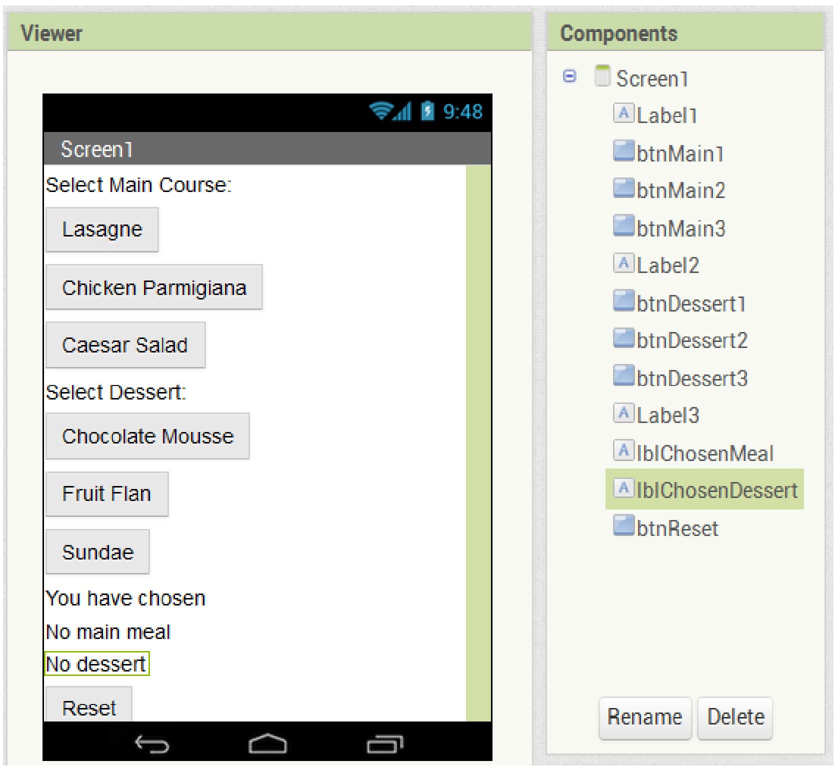
Pineapples:	7.67 each
Plums:	1.78 each
Watermelon:	12.11 each

Check that they calculate correctly for several input values, and confirm that the earlier buttons still work correctly.

h. What output does it produce if you try to buy 10 of some item? Does it show two digits for the cents part? If not, look for a block that will ensure the number is displayed showing 2 decimal places under all circumstances (Hint: a math block will help).

Task 2 – Meal Selector App

Design a new app to have the components as shown below (on left hand side) and ensure that the names of the components are matching as shown on the right side:



The app needs to behave as follows:

- Initially, only the main courses will be able to be selected. The Desserts and the heading before them should not even be visible to the user, but the two messages at the bottom should be showing.
- When the user chooses one of the main courses, the other two should be disabled from being clickable, and all the desserts should be now shown (as well as their heading). Also, the message at the bottom should be updated to say which main meal has been chosen.

  HINT: there are code blocks in the 'Logic' palette which say "true" and "false" which can be used to indicate whether something should be visible or enabled, or not. We will look at these in more detail next week.

  SUGGESTION: You should try to get one button to work first. Then try to get the RESET button to work (see below) before you make other buttons do things.

- When the user chooses one of the desserts, the other two should be made invisible, and the message at the bottom should be updated to say which dessert has been chosen.
- When the reset button is pressed, the app should return to the way it was initially, i.e. only the main courses being visible and all of them able to be selected, and the messages at the bottom be saying that no selections are made yet.

Extension Task

For either of the two apps, use a HorizontalArrangement layout component to get more than one button/component to appear in the same row as each other.