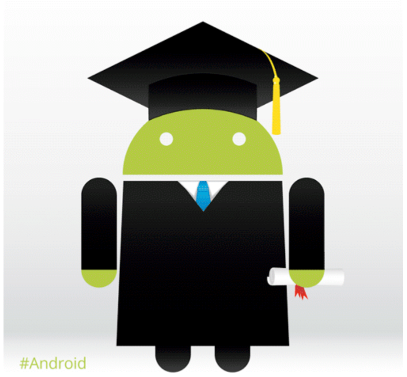
ICS4U Summative Project

Overview:

Your summative goal is to develop an Android app using Java and Android Studio that demonstrates your mastery of the concepts covered this year.



Creativity is encouraged but to keep your project manageable aim to create an app with a single Activity (screen) and basic widgets; for example, consider building an app that is (a little) more complex than your magic number or coin counter app.

There are two "milestones" for your summative:

1. **May 16th:** You will submit a 1 page (printed) proposal that includes:
   a. A **paragraph** (or two) describing what your app will do. What problem does it help solve?
   b. A **design diagram** that shows the basic layout and widgets in your app, including any **ID** and **string resource names** (see the *TipCalculator* example in **5-3**). All names should follow a consistent pattern / style.

   This will also be the **last chance** to demo any problem set exercises.

2. **May 29th**: You will (electronically) submit your final code, that must:
   a. Be ***your*** original work!
   b. **Match your design diagram** as closely as possible.
   c. Follow the design pattern we have been using for our apps in class.
   d. Make effective/efficient use of Java and the Android API.
   e. Meet all of our coding style guidelines (JavaDocs <u>**not**</u> required).
   f. Use **regular inner classes** for event handling (i.e., not defined within the **onCreate()** method).
   g. Include an **onPause()** and **onResume()** method to save the state of important instance variables.
   h. Lock the orientation of the app, and ensure that it functions with our Nexus 7 (2012) screen size -- I will test your app on this device.
   i. Customize the icon for the app.
   j. Include at least one new interactive widget that you have learned on your own. Your app should be intuitive to use.

   Be prepared to give a short (~5 minute) demo of your final app to the class.

ICS4U Summative: Marking Rubric

App: __________________ **Student Name:** __________________

Marking Criteria	Level (Cat)	Comments
Practice Apps (5-2 #2, 5-3 #3, 5-4 #1, 5-5 #2 or 3)	(A)	
APP DESIGN: Due (printed) May 16th		
On time, printed, and clearly written app description with neat and complete design diagram (including well-named ID and resource names). Design demonstrates careful planning **before** coding starts.	(T)	
FINAL APP CODE SUBMISSION AND CLASS PRESENTATION: May 29th		
Code structure follows the design pattern we have learned in class, and closely matches your design plan/diagram.	(K)	
All style guidelines have been applied correctly and code is well commented (widget IDs, string resource names, Java variable names, file headers, method headers, and tricky code).	(C)	
Code is efficient, makes effective use of Java and the Android API.	(T)	
App is bug-free, orientation is locked, looks good on Nexus 7 (2012) device, has customized icon, and is intuitive to use.	(A)	
Properly designed **onCreate()**, **onPause()**, and **onResume()** methods. Correctly implemented regular inner classes for event handling.	(A)	
App includes at least one interactive widget that you have learned about yourself.	(K)	
Overall level of effort compared to peers, and difficulty level of app design.	(T)	
Well-prepared, engaging, class presentation of key features of the app.	(C)	
Overall Mark:	/ 40	

ICS4U Summative - “Plan B”

This is an alternative (less open-ended) project option in case you find creating an Android App to be too difficult or time-consuming. This option requires less creativity and does **not** require a proposal.

Extend your **SuperPaint** drawing application from your last assignment, by adding additional features to it:

1. Have a drawing area, completely mouse driven, with a “Clear” button that will reset the drawing area (and clear any “Undo” and “Redo” operations).
2. Have “Undo” and “Redo” buttons that reverse previous operation(s) or redo undone operation(s), respectively. This time, use one or more data structures from the Java Collections Framework.
3. Have a pull-down list to select a shape (line, oval, rectangle), and a checkbox to specify whether to fill it or not (applied to ovals and rectangles only).
4. Have a checkbox to specify whether to paint using a gradient, and two buttons that allow the user to choose a first and second colour in the gradient. If the checkbox is unchecked, the first colour is used as a solid colour for drawing (and filling). ***Hint:** see the **Graphics2D** and **GradientPaint** classes.*
5. Have a text field for entering line width. ***Hint:** see the **BasicStroke** class.*
6. Have a checkbox to specify whether to draw a dashed or solid line, and a text field for entering stroke dash length.
7. Have a “File” menu with options:
   a. “About” – displays a dialog with your name etc..
   b. “Preferences” – displays a preferences dialog (see #8)
   c. “Exit” – terminates the application
8. Have a separate “Preferences” window, accessed from the “File” menu where the user can set default values for shape type, fill on/off, gradient on/off, first gradient colour, second gradient colour, line width, dash length, and dashed line on/off. All preference data should be saved in a configuration file so that it is maintained between sessions.
9. Add **at least one** other custom feature. Some ideas: add additional shapes like round rectangles or arcs; use a JDesktopFrame and JInternalFrame to allow the user to create multiple separate drawings in separate child windows; use a JInternalFrame to make a “floating” toolbar containing most of the GUI components that allow the user to determine the characteristics of the shape to be drawn.
10. Meet all of our coding style guidelines, **including** complete JavaDocs.

Be prepared to give a short (~5 minute) demo of your final app to the class.

ICS4U Summative ("Plan B"): Marking Rubric

Student Name: ____________________

Marking Criteria	Level (Cat)	Comments
Practice Android Apps **(5-2 #2, 5-3 #3, 5-4 #1, 5-5 #2 or 3)**	**(A)**	
APP DESIGN: Not Applicable		
On time, printed, and clearly written app description with neat and complete design diagram (including well-named ID and resource names). Design demonstrates careful planning **before** coding starts.	0 **(T)**	• Not applicable to paint program
FINAL CODE SUBMISSION AND CLASS PRESENTATION: May 29th		
All requirements properly implemented in program design.	**(T)**	
All style guidelines have been applied correctly and code is well commented, including **complete** JavaDocs.	**(C)**	
Well-designed classes / methods, effective encapsulation, wise choice of JCF data structure(s). Intuitive, easy to use GUI.	**(K)**	
Code is bug-free, efficient/non-redundant, and makes good use of the Java API.	**(A)**	
Effective event and exception handling techniques.	**(A)**	
App includes at least one custom feature/widget that you have learned about yourself.	**(K)**	
Overall level of effort compared to peers, and difficulty level of custom feature(s).	**(T)**	
Well-prepared, engaging, class presentation of key features of the program.	**(C)**	
Overall Mark:	**/ 40**	