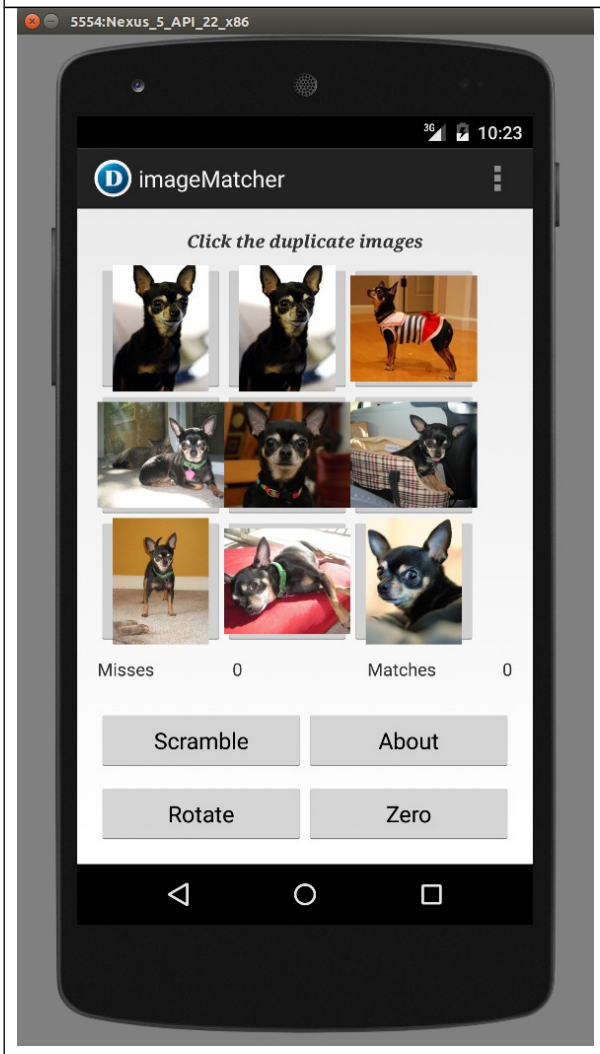
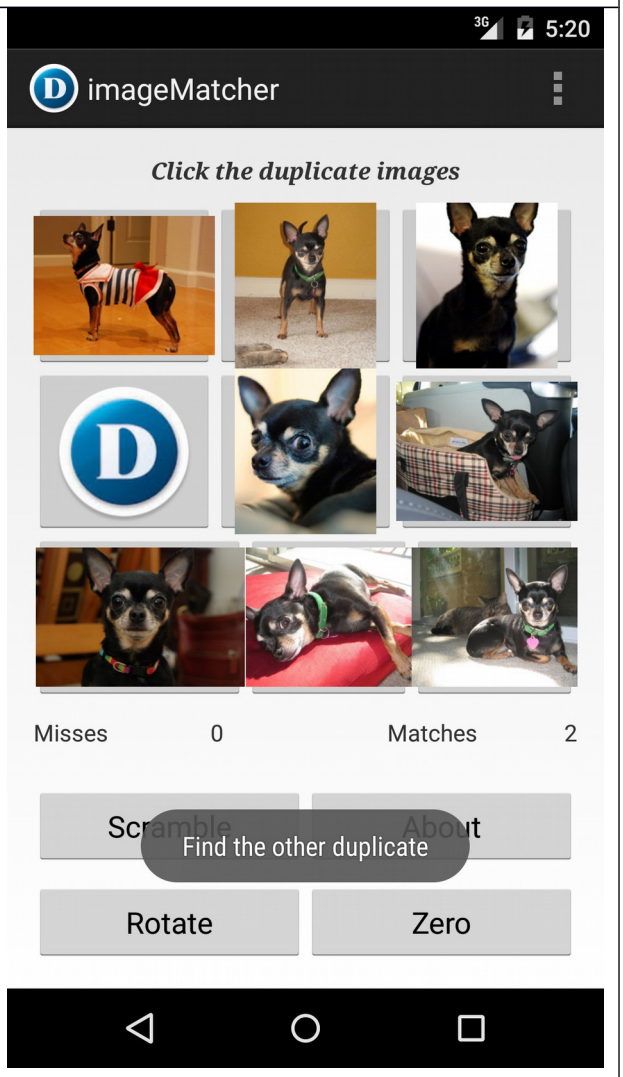
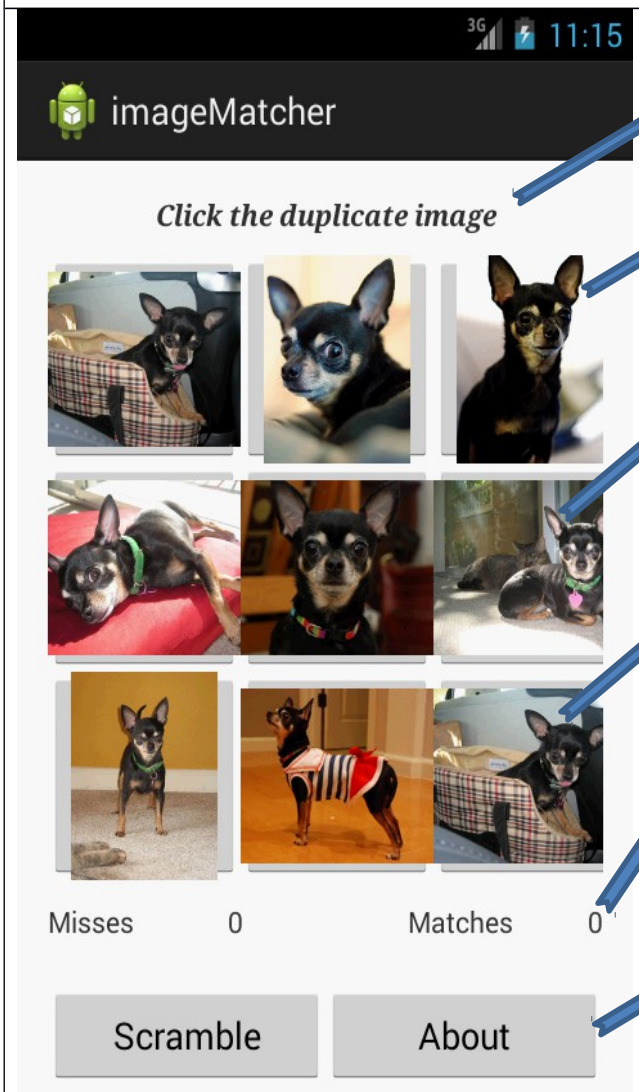
The goal of this assignment is to create a simple *Image Match Game App* as illustrated below. The users are expected to select the duplicate images. When started the app will display 9 images randomly, one of which is a duplicate. You will keep two counters hits and misses. The scramble button will reshuffle the images, the about button will open a new activity with information about the app. The counters will be maintained after the app is killed.

Default UI (main layout)	First of duplicates selected. (toast showing)
	

1. You will need 8 similar images and 1 image to indicate that it is a selected duplicate. In eclipse copy the images to the first drawable folder. Find small images, you may keep the images sizes frozen width and height or if you want a challenge you can have android expand them. But be certain the game fills the whole screen.
2. You are responsible for the logic to select a random duplicate and to scramble/shuffle the images to populate your layout so they are displayed in a random order using setImageResource(R.drawable.*imagename*) and to keep track of the duplicates. Hint: (use any of arrays, R.drawable.imagename et al are integers, Random class, Array class, Collection class…)
3. Image Click:
    a. If clicked image is a duplicate:
        i. add one to the match count
        ii. set the image to not clickable
        iii. change the image to your 9th image (duplicate selected image)
        iv. If it is the first duplicate clicked: pop up a Toast “Find the other duplicate.”
        v. If it is the second duplicate clicked: pop up an alert dialog with one button with text “You win! Click on Scramble to play again.”
    b. If clicked image is not a duplicate:
        i. add one to the not matched count
        ii. set the image to not clickable
        iii. pop up a Toast
4. Scramble button:
    a. shuffle and redraw the images (a new game)
5. Zero button:
    a. reset the counters
    b. save the counters in shared preferences
6. About button: fire a new Activity:
    a. Create a layout (UI) with
        i. A small paragraph describing how to play
        ii. Credits (who wrote the app)
        iii. Any small image you like
7. Be sure to properly comment your app’s java code, javadoc for class and methods.
8. Use SharedPreferences to
    a. Make sure the miss and match counts are saved and restored when the app is killed and when the cournters are reset.
9. Use the instance state bundle to
    a. Make sure your game state is restored on rotate
10. Reset Counts button: reset all of the counters
11. Rotate button: rotate the screen to landscape if currently portrait and vice versas
12. All of your UI strings, dimensions and colours must be in strings.xml no hard coding.

13. Create the main layout UI. I used a TableLayout below is something similar to mine just to give you an idea. You are free to use what you wish. You may want to use http://www.droiddraw.org/widgetguide.html to help choose and format your widgets.

Layout UI	XML
	<LinearLayout > <TextView /> <TableLayout > <TableRow> <ImageButton /> <ImageButton /> <ImageButton /> </TableRow> <TableRow> <ImageButton /> <ImageButton /> <ImageButton /> </TableRow> <TableRow> <ImageButton /> <ImageButton /> <ImageButton /> </TableRow> </TableLayout> <LinearLayout> <TextView /> <TextView /> <TextView /> <TextView /> </LinearLayout> <TextView/> <LinearLayout> <Button/> <Button/> </LinearLayout> </LinearLayout>

14. Once you have tested and the UI and logic are operating correctly, use localization to make your app bilingual (french and english.) (This is only possible if you have used string resources instead of constants everywhere.) http://developer.android.com/guide/topics/resources/localization.html
15. Optional: Code the app to have a larger pool of images, to give it more variety. If you choose to do this, do so only once the other steps are completed.

Assignment Submission Requirements

1. Do this assignment with your group (teacher assigned.)
2. Be sure to comment your methods and all code thoroughly.
   Use Javadocs for all class files and your methods, for the class javadoc something like:

```
/**
* Assignment 1: Image Matcher
* blah blah blah
*
* @author D. Ritchie
* @author B. Kernighan
* @version
*/
```

3. Your final project should be submitted to Lea/Moodle as a zip file that I can import into eclipse. I will need the apk submitted too.

Learning Objectives

- UI layouts, widgets
- SharedPreferences
- Activity lifecycle
- Drawable resources
- String resources
- Localisation
- Dialogues

See my sample code for reference https://github.com/orgs/Android518-2015