

[**Overview 1**](https://docs.google.com/document/d/1wsmLki_CMUfyLTZk03BMGz5-gRRLpSsNsSgvKx9EqIs/edit#heading=h.td9vhdmkwto)

[**Sample UIs 1**](https://docs.google.com/document/d/1wsmLki_CMUfyLTZk03BMGz5-gRRLpSsNsSgvKx9EqIs/edit#heading=h.phveoaxbonlr)

[**Specification 2**](https://docs.google.com/document/d/1wsmLki_CMUfyLTZk03BMGz5-gRRLpSsNsSgvKx9EqIs/edit#heading=h.jhlg17wapdzw)

[**Collaboration (you may NOT do this solo) 4**](https://docs.google.com/document/d/1wsmLki_CMUfyLTZk03BMGz5-gRRLpSsNsSgvKx9EqIs/edit#heading=h.6beywap6gt52)

[**Learning Objectives 4**](https://docs.google.com/document/d/1wsmLki_CMUfyLTZk03BMGz5-gRRLpSsNsSgvKx9EqIs/edit#heading=h.h1nx00o2ne37)

[**UI Example  with minimal xml 5**](https://docs.google.com/document/d/1wsmLki_CMUfyLTZk03BMGz5-gRRLpSsNsSgvKx9EqIs/edit#heading=h.5ul5v5z3wzmg)****

**Overview**

The goal of this assignment is to create a driver’s quiz about road signs.  You will display  4 road sign images and a description that matches one sign.  The user is to click on the image that matches the description.  They are to be given 2 tries.  If the first selection is incorrect the image is to be replaced with a “wrong” image.   If the first selection is correct or the second incorrect make the next button visible.  You will keep counters for correct / incorrect answers across activities.   You will also keep a counter of quiz attempts,  all three counters will be maintained after the app is killed.

**Sample UIs**

My example UIs are below, you can probably do better ;)

|  |  |
| --- | --- |
| Default UI  (main layout) | Clicked image incorrect: |
| https://lh4.googleusercontent.com/9O-W7QWyME8zl32QzTPBLqy6daRJQjFNEcnLXuRAU4OTfXUCeLjlyD-gwKUfBb9Zl1xxDUWfCxnF8zg7wHU10nVIJ8j5O20UPC64ZCz5BXODVyjFSaU-XFXOoPX984Ee0A | https://lh3.googleusercontent.com/iSAebh1ODvdphdwWv1OAixjzMVwVOhQxbNr3kAOeSQnHZsnUnFI97SqwWxAnjAkym2NkrAcOA3Ml8sFWCodBcyBZm7y91zGDFggSpBk_oJn2A0UQlKvIWj9IavGH6SZ7PA |

**Specification**

1. App name in the Launcher must be DQ followed by your  first names:  DQName1Name2
2. Create and use your own icon
3. Show the user a minimum of 4 questions, one at a time for the quiz.    
   You will need a least 15 images so the answers are not obvious.    
   One text question per image.  
   One hint per image  
   Randomly select the image/question.
4. Maintain all counters between run times (SharedPreferences.)
5. Maintain state in the Bundle as needed for display information
6. You are responsible for the logic to select a random image and it’s text for each question.  Randomly select 3 other images to display with it. Use them to populate your layout so they are displayed in a random order using setBackground() or setImageResource()  Keep track of the correct answer. Hint: Resource class, Random class, Array class, Collection class…
7. UI must show
   1. 4 image buttons
   2. question count ex:   question 1 of 4
   3. counts for correct and incorrect: correct 1 incorrect 0
   4. hint button
   5. about button
   6. next button  (hidden for each new question)
8. Make sure the images scale when the screen is rotated and you can see the whole image (if resolution is off, that is ok, you only need one set of images but they must look centred and fill the view, no gaps unless part of the image)   You may need to crop so they are similar sizes or convert to 9 patch images.       
   Here are some cheat sheets for [views and layouts](https://drive.google.com/drive/folders/0B-CHlg81QPjfMzNGaFY5VURGWEE?usp=sharing)See also [ScaleType Android](https://developer.android.com/reference/android/widget/ImageView.ScaleType.html) & this is pretty good [a visual guide](https://robots.thoughtbot.com/android-imageview-scaletype-a-visual-guide)Here are some description and image sources:

<http://www.drivingtest.ca/quebec-driving-test/>

<https://testdeconnaissances.saaq.gouv.qc.ca/en/>

1. Create and use a unique Icon: <http://romannurik.github.io/AndroidAssetStudio/icons-launcher.html>
2. When the last question is answered, update the saved scores (keep the last two scores) (SharedPreferences)
3. If the app ends before they finish the questions (n.b.  override finish)  When the app is started again use shared preferences to return the user to next question and set the counters. You need counters to implement that, along with an array of  what questions have been answered.
4. Image Buttons actions:
   1. If it is the correct image:
      1. add one to the correct count, update score, update question counter
      2. make all images not clickable
      3. change the image in some way to show it is the correct answer (you may want to try <selector> and/or an alternate image that shows the same with a border) see: <http://developer.android.com/guide/topics/ui/controls/button.html>
      4. enable and make visible the next button
   2. if it is not the correct image (first incorrect):
      1. set it to not clickable
      2. change the image in some way to show it is incorrect
   3. if it is not the correct image (second incorrect):
      1. add one to the incorrect count, update score, update question counter
      2. set all images to not clickable
      3. highlight the correct image in some way  (see point a. iii. )
      4. enable and make the next button visible
5. About button: fire a new Activity:
   1. Create a layout ( UI ) with
      1. A short paragraph describing how to play
      2. Credits (who wrote the app)
      3. Any saved past scores (SharedPreferences)
      4. Any small image you like (ex: your photos)
6. Hint button:  fire an Intent to do a google search using the current question's  description text, maybe prefix it with “road sign”  See Perform a Web Search here <https://developer.android.com/guide/components/intents-common.html#Search>
7. Be sure to comment your methods and all code thoroughly, use Javadocs for class and your methods.
8. Use Logging
9. All of your UI strings must be in strings.xml.
10. Once you have tested and the UI and logic are operating correctly, use localization to make your app bilingual (add french) strings   
    <http://developer.android.com/guide/topics/resources/localization.html>

**Collaboration (you may NOT do this solo)**

1. You will do this assignment with a [teacher assigned partner](https://docs.google.com/spreadsheets/d/1S3rkvnp0wGZkONimtI9jGYbM77eEcgN-g74v6uQF6AI/edit#gid=0).
2. You will be given access to an assigned a repo by the teacher and expected to use the protected workflow from git lab 3 [READ the workflow overview here](https://github.com/Android518-2017/git-workflow)You will have to create a branch from the protected branch master and discuss and agree on merging pull requests.   Most expectations are in the workflow overview but see also [collaboration expectations](https://drive.google.com/open?id=1lo4pILADxl4YcSrMcl0JJDyq4w9tEtgzDTO7Xyr1ZmU).
3. Due in two weeks,  I will clone master once the deadline expires, ensure all of your changes are applied to it via a merged pull request.
4. See collaboration expectations for rules & regs

**Learning Objectives**

UI layouts, widgets

State, SharedPreferences

Activity lifecycle

Drawable resources

String resources

Explicit and Implicit  Intents

Localisation

Collaboration using git

**UI Example  with minimal xml**

Below is my layout overview, just to give you an idea. You are free to use what you wish [layout optimisation](https://developer.android.com/training/improving-layouts/optimizing-layout.html),  [RelativeLayout](https://developer.android.com/guide/topics/ui/layout/relative.html), [ConstraintLayout](https://developer.android.com/training/constraint-layout/index.html) etc.  Just be sure it resizes on rotate.  (Try to make yours prettier than mine :) )

|  |  |
| --- | --- |
| Layout UI | XML |
| https://lh6.googleusercontent.com/ts4B_vFSVK2NPIq6wToBvbOm9uZhYCybH4IOAbyXtl2BRnPQYRycRQkyzmcG3OIvam0ue5gwwQaYYRHpi6xGGH_rhkb6sNNvPCfXPaoPHOV3xgth4SCvkpga63TEaDs1Mwhttps://docs.google.com/drawings/d/s2upyudndeVnY1JXCDWFCRA/image?w=115&h=40&rev=1&ac=1 | <LinearLayout >  (vertical)     <LinearLayout> (horizontal)         <TextView />         <TextView />     </LinearLayout>     <LinearLayout > (horizontal)       <ImageButton/>       <ImageButton/>     </LinearLayout>     <LinearLayout> (horizontal)         <ImageButton />         <ImageButton />     </LinearLayout>     <TextView/>     <TextView/>     <LinearLayout> (horizontal)         <Button />         <Button />     </LinearLayout>     <Button/>  </LinearLayout> |