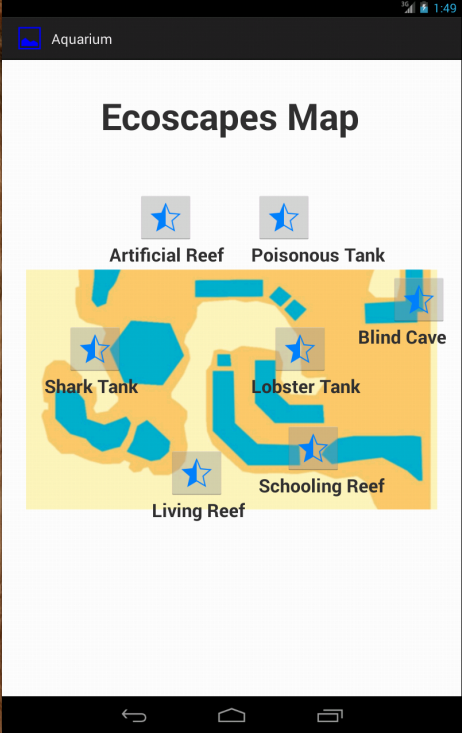
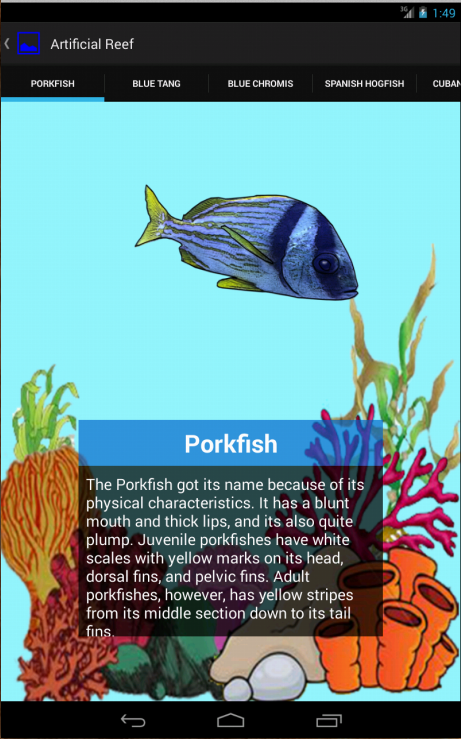
Group 3

Quiz 4

6/20/14

Aquarium App

The Aquarium application opens in start up to a map screen shown on the left. The map screen shows the different tanks in the aquarium section of the museum. The map shows the different tanks in the area. Once clicked the app directs you to the tank page where all the different fishes are listed in the interface. An example is shown on the right. The screen shows the type of fish and a description. This screen can be swiped along to show more fishes or you can click the back button to go back to the map. This app uses fragments and different activities to display the different fishes. The images are all located in the “drawable…” folders and the fragment xml files are located under layout.

 Bugs App

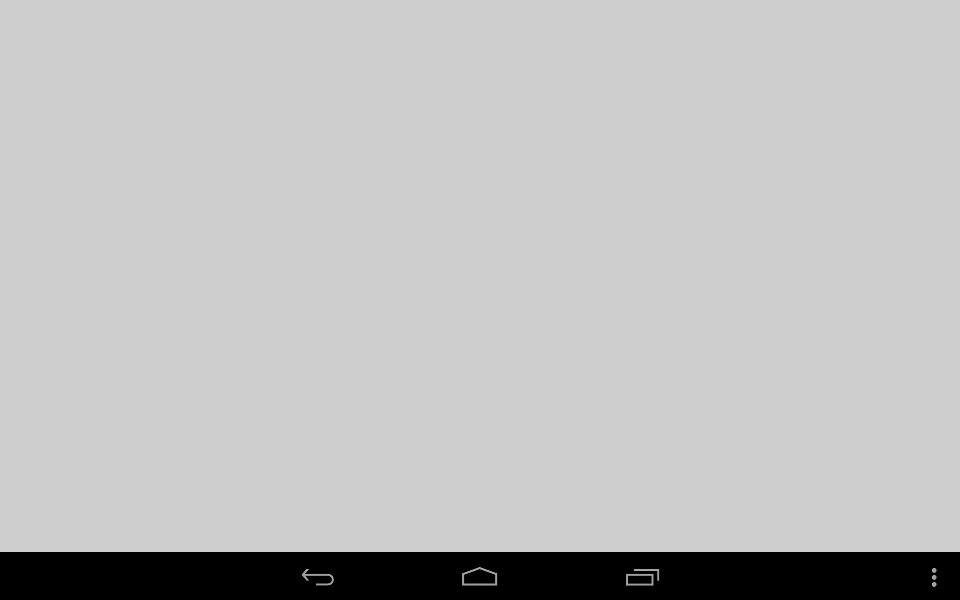
The bugs application is very similar to the aquarium application. The bugs app is different however because it does not open into a floorplan page but goes straight to the pages of the animals. Similarly to the aquarium app, the animals can be seen by swiping the screen to different pages. The bugs app has fewer files and does not make separate java files for each animal like in the aquarium app. The bugs application uses the BugReceiver class and the MainActivity file to accomplish the display. The MainActivity.java uses a series of cases to test and change the BugImageView on the app while the BugReceiver class checks if which intent is currently running and then adds that bundle to the intent: i. Screenshots of the app are shown below.

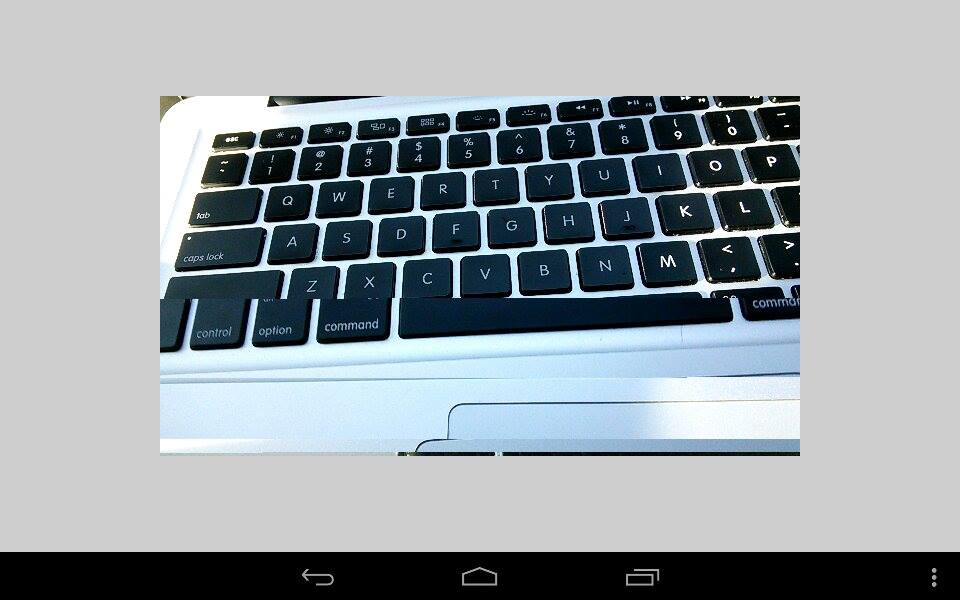
Integration

The integration, or robot detect app uses the different green pixel on a camera to estimate the distance between them. The app begins on start up as a blank screen. Once you click once, you can see the camera, which is running at 10 fps. Once you click, the app takes the picture and puts it on the camera view. The app then goes through all the pixels and selects them and creates a purple highlight upon them. The yellow column on the end forms as the image is being searched through for green pixels. At the end, it takes the highest and lowest y values of green and subtracts the 2 to get the distance between them.

Start up view:



When Camera Turns on from tap:



When Picture is taken and green is replaced to purple:



Distance is calculated