Kevin Hoffman, Sean Kim, Christina Le, Xin Song

CS 2110, Lab 103

Final Report

Due: Friday, December 7, 2012

1.

|  |  |  |
| --- | --- | --- |
| **Story ID** | **Story** | **Done?** |
| 1 | As a user, I want to see a splash screen displayed while the program loads. | Completed |
| 2 | As a user, I want to see my current latitude and longitude and the latitude and longitude of where I need to go. | Completed |
| 3 | As a user, I want to see a Google map showing my current location. | Completed |
| 4 | As a user, I want to see the running time it takes me to conduct the tour. | Completed |
| 5 | As a user, I need an arrow pointing me in the right direction. | Completed |
| 6 | As a user, I want a congratulatory message at the end of the tour. | Completed |
| 7 | As a user, I want I know when I reach each destination. | Completed |
| 8 | As a user, I want to be able to override the current GPS coordinates with another set. | Completed |
| 9 | As a user, I want hints telling me whether I'm getting closer or further away from the next destination. | Completed |

2.

Once the splash screen disappears, the home page appears with a picture of the Rotunda. This home screen shows the user’s current location, via latitude and longitude, and the next destination’s name and coordinates. A blue arrow points in the direction the user must travel to reach the next destination, and a timer begins to store the total time of the tour. Toasts providing hot and cold hints appear every so often to help guide the user. If the user is over 100 meters from the destination, the hint says, “You’re getting cold!” If the user is 100-50 meters from the destination, the hint says, “You’re getting warm!” If the user is 50-10 meters from the destination, the hint says, “You’re getting hot!” If the user is within 10 meters from the destination, the application says, “You have arrived at your destination.” Additionally, the home screen has a map button that allows the user to see their current location denoted by a red pushpin on Google Maps. This secondary screen also displays a back button to enable the user to return to the main screen.

After the user has reached the first destination, this process continues for all seven destinations in the default order of the Rotunda, McIntire, Amphitheater, Alderman, Castle, Old Dorms, and Rice Hall. A simple change in the code would allow the user to choose any number of these destinations and pick the order. Once the user has reached the last destination, a congratulatory message “You have finished the tour!” pops up and the total time is displayed. The application also plays celebratory music, specifically The Boston Pops Orchestra’s “Sleigh Ride.”

For testing purposes, the application allows the user to override the GPS location settings by entering manual coordinates. This would allow the user to “complete” the tour from the convenience of one room.

3.

1. As a user, I want to see a splash screen displayed while the program loads.
   1. Splash screen displays app name and names of all team members
      1. Tested during sprint 1
      2. Passed all tests.
   2. Splash screen remains on screen for 5 seconds
      1. Tested during sprint 1
      2. Passed all tests.
2. As a user, I want to see my current latitude and longitude and the latitude and longitude of where I need to go.
   1. The application displays the GPS coordinates of current location.
      1. Tested during sprint 1
      2. Passed all tests.
   2. The application displays the GPS coordinates of the next location.
      1. Tested during sprint 1.
      2. Passed all tests.
3. As a user, I want to see a Google map showing my current location.

a. Get the GPS coordinates from the application

* + 1. Tested during sprint 1
    2. Passed all tests.

b. Generate google map on the screen for the provided coordinates

i. Tested during sprint 1

ii. Passed all tests.

1. As a user, I want to see the running time it takes me to conduct the tour.
   1. Store running time in application
      1. Tested during Sprint 1
      2. Passed all tests
   2. b. Display running time of tour
      1. Tested during Sprint 1
      2. Passed all tests
2. As a user, I need an arrow pointing me in the right direction.
   1. The screen displays an arrow.
      1. Tested during sprint 2.
      2. Passed all tests.
   2. The arrow points in the correct direction.
      1. Tested during sprint 2.
      2. Passed all tests.
3. As a user, I want a congratulatory message at the end of the tour.

a. The user reaches the end of the tour

* + 1. Tested during sprint 2.
    2. Passed all tests.

b. The screen displays a congratulatory message

i. Tested during sprint 2.

ii. Passed all tests

c. Celebratory music plays

i. Tested during sprint 2.

ii. Passed all tests.

1. As a user, I want I know when I reach each destination.
   1. The application considers that the user has reached their destination within 10 meters of the location.
      1. Tested during sprint 2.
      2. Passed all tests.
   2. The screen displays a notification.
      1. Tested during sprint 2.
      2. Passed all tests.
2. As a user, I want to be able to override the current GPS coordinates with another set.
   1. The application shows current coordinates as the coordinates given by the user.
      1. Tested during sprint 2.
      2. Passed all tests.
   2. The arrow points in a new direction to reflect the user-given coordinates.
      1. Tested during sprint 2.
      2. Passed all tests.
   3. The hints reflect the user-given coordinates
      1. Tested during sprint 2.
      2. Passed all tests.
3. As a user, I want hints telling me whether I'm getting closer or further away from the next destination.

a. The application displays hints based on the user's current location

i. Tested during Sprint 2

ii. Passed all tests

b. The hints are correct

i. Tested during Sprint 2

ii. Passed all tests

The quality assurance went through the whole product developing process. After each sprint, quality of each part of software component designed by individuals of the team was checked Different functionalities of the application were made sure that they can integrated together and work as a whole.

4.

* MainActivity.java is the home screen of the app controlling the tour and giving the user various information like coordinates and direction.
  + startTimer(View v)
    - Method is called when user presses the start button and the timer is started.
  + updateText()
    - Updates the current latitude and longitude on screen.
  + updateNextText()
    - Updates the latitude, longitude, and the building of the next tour stop on screen
  + updateTemp()
    - Displays a toast message whether the user is cold, warm, or hot from the next location
  + setArrow()
    - Calculates where the arrow should be pointing based off where the user is heading and animates the arrow on screen
  + checkDestination()
    - Based on how far away the user is from the next location, it notifies the user if he/she has reached the next tour location, is done with the tour, or sets the “temperature” of the user based on distance
  + setLoc(View v)
    - Called when user presses the Set Location button. It overrides the current location of the user and displays the new coordinates on screen.
  + goMap(View v)
    - Called when user presses the Map button. It launches the Map screen.
* gMapActivity.java is the map screen that shows the user’s current location on a map
  + showMap(double lat, double lng)
    - This method updates the user’s current location on the map given the user’s current latitude and longitude
  + Back(View v)
    - This method is called when the user presses the back button. It takes the user back to the previous activity.
* GpsTracker.java is the class that gets and updates the user’s current location.
  + getLocation()
    - This method checks to see if GPS is enabled and if it is requests a location update.
  + onLocationChanged()
    - This method is automatically called on a successful location update and it sets the current latitude and longitude in the MainActivity.java and calls checkDestination() on the MainActivity.java
* Splash.java is the class that showed our splash screen at program launch for 5 seconds.

Major Data Structures

* ArrayList<ArrayList<String>> locations
  + It stores the tour locations. The inner array list stores the latitude, longitude, and building name of each location.
* int tourStop
  + This field kept track of what stop the user is currently at.
* ImageView arrow
  + This held the image of the arrow that pointed the user where the next tour stop is located.
* MediaPlayer mpAudio
  + This stored and played a song at the end of the tour.

5.

Kevin Hoffman, Software Architect

Wrote the code for getting the user’s current location, the arrow, and checking how far the user is from their destination and what to do based on the distance. I also integrated everyone’s else's code into one project and made sure it integrated correctly.

Sean Kim, GUI Designer

Designed user interface for application. Worked primarily with xml layout files. Coded splash screen activity and xml layout of home screen. Utilized GIMP image editor to create custom splash screen, home screen and launcher icon.

Christina Le, Project Manager

Ensured team met criteria as detailed by the assignment. Kept records of coding progress and updated project backlog and sprint sheets. Wrote the code for timer and display of hot and cold hints.

Xin Song, QA Manager

Ensured that each individual component of our software meets the functionality requirement in every story and passed tests during each sprint, also every component integrated together successfully at the end of the application development. Implemented google map functionality to show current location to user. Wrote the code for end of tour congratulatory message and music.

**Retrospective (will be combined into one)**:

Kevin: The stories helped us break down this big project into smaller more manageable tasks that could easily be divided up between people. I also liked the concept of sprints because it forced you to work on the project incrementally instead of rushing to complete everything at the end. I felt that the entire process makes it easier for a team of developers to complete a project in a reasonable timeframe. Defining acceptance criteria made it easier to test our code and set benchmarks on how far we had gotten. I had previously done Android before but I still learned more about the LocationManager class and using Google Maps. This software design process helped us formulate our plan on how we were going to build our app and implementing our plans.

Sean: I liked how the stories divided up the work in the application but I didn’t like how arbitrary the division was. I wish there was a bit more guidance on what is considered a story and what is considered a task, etc. In theory, using stories seems like a wonderful idea and it worked pretty well but the practice of it still left me feeling unsure of what was what. On the whole, I loved the android project. It was a lot of learning to do in the beginning but I think I learned more during this project than I did in all of my other classes combined. I have a much greater understanding of the android operating system as well as how applications work within the system. The best part about the project is that it relates to the real world so well, seeing how mobile applications are becoming more and more popular. Working in groups was also a great idea seeing as this is how things work in real software companies. Overall, I enjoyed the project.

Christina: The stories were extremely helpful in seeing how the application would be broken down into more manageable parts. At first, the requirements overwhelmed me until we began thoroughly discussing the features as a team. The diverse backgrounds and experience of each team member greatly contributed to our ability to divide and conquer. Within the sprints, the acceptance criteria determined beforehand helped me keep on track in writing code. Also, because the backlogs were stored on a Google doc, I could easily see the current status of each story and sprint. The most difficult part of the project was learning the Android API. As I had zero knowledge prior to beginning the project, I had to extensively research specific classes I needed to implement the features we had determined. However, once I became moderately comfortable with the Android setup, coding and seeing the separate methods mesh together to create the application, I really enjoyed the process.

Xin: As the first Android application developed with my team members, this project strengthened my hand-on programming skills and also deepened my understanding of object-oriented programming by working with different components which the Android API provided. I really enjoyed the whole project process because I saw how Java is used in the real-world app development and implemented on mobile platform such as smart phones. Working as a team, coding was not the most difficult part since adding a functionality through Android API is not like writing something from draft. However, team cooperation contributed greatly to the success of our final product. This includes defining and assigning tasks to team member and keeping record via product and sprint backlogs. Good communication ensured every team member had up-to-date information throughout the process, thus making it easier to combine contributions from each one’s effort. So I really like this project and it was a great chance for me to get a feel of conquering a project efficiently and professionally like in the real world industry.

1. The set of stories in your Product Backlog, including any that were not completed. For each story, give it's status (completed, partially completed, not completed). It's not required that you have succeeded with all the stories that you initially hoped to implement UPDATED 12/6 5:20pm
2. A brief description of how to use your program once we start it, and what it can do. This does not have to be a user-manual. It can be a few paragraphs that explains any special features of the user-interface. We will assume it supports the core functionality, but if there are things it does not, then your description should mention this. This should describe all optional features you implemented. The purpose of this description is to help a grader run your program and see what it does after you demo it to the TAs.
3. A section on testing that you did. This should be based on the acceptance criteria defined for each story. List each of these (grouped by story), and for each state when you tested this one (in which sprint, at the end, never) and what the result was. If testing revealed a code defect, briefly tell us that. (Don't invent results here.) Finally, at the end of this section, write a few sentences about how well your quality assurance / testing process worked for your team. (We're curious.) The goal of this section is for us to see how you used the quality assurance activities that were part of your testing process.
4. A brief description of your program's design. It should list the major classes and for each one say in an English sentence or two what the class' high-level responsibiliy(ies) is(are). Also, for each list the major methods and what they do. (You do not have to list every method and field. No need to document getters, setters, equals, compareTo, etc.) This section also should list the major data structures that are used that are important to your design (e.g. a list of buddies, a map of tour-sites, etc). You can use UML class diagrams, but this is not required - bullet lists and text will be fine. (If you want to use UML, use the Violet UML program that's freely available here on the web. It's also installed in the ITC and 001 labs.) The goal here is to see that you know how to describe a class design effectively to others.
5. A team report, as follows. For each person on the team, list:

* Tasks or areas where the person had the primary responsibility, and tasks or areas where the person assisted.
* What coding activities were done by that person (writing test code counts here too).
* A final process retrospective that discusses what worked well and didn't work well in terms of using stories, acceptance criteria, tasks, etc. as part of planning and executing your development.