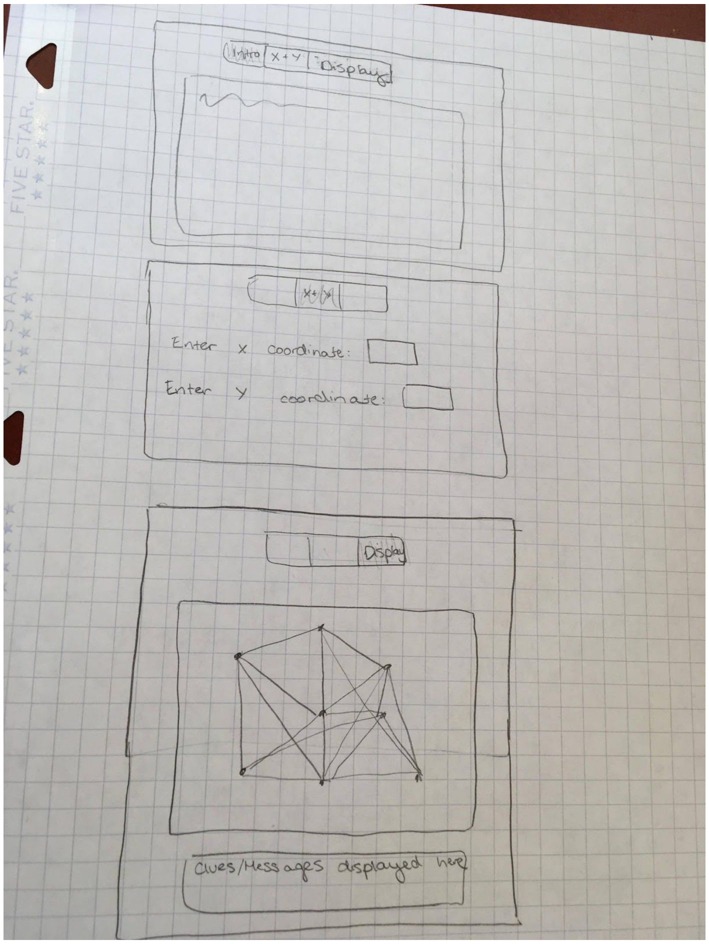
CS 230 FALL 2017

Mona Kashyap, Maggie Ugelstad, and Alie Langston

Geocache Program

User's Manual

This program simulates a geocaching experience. While our original design will be for a desktop, we are hoping to extend this program to be used in an android application. The user's manual will chiefly discuss how a user can interact with the desktop version of our application.



The picture above is the very basic outline of our intended GUI.

The first panel will feature an introduction to the program, explaining what the user should and can do.

The second panel will allow the user to manually enter in their coordinates, which will then be displayed on a map on the third panel. On the third panel, a user will be able to then move their place on the map (which will be displayed by a specific symbol) using the arrow keys on their keyboard. They will only be able to move their symbol along given paths, which will also be displayed on the map. If the user is in the near vicinity of a clue, a message will be displayed on the bottom of the panel stating that they are getting closer. Once they have reached the clue, the clue will be displayed at the bottom of the page.

Challenges will also be available to the user to increase the difficulty of the game. Similar to the cues, challenges will be displayed in the bottom of the third panel.

Technical Report

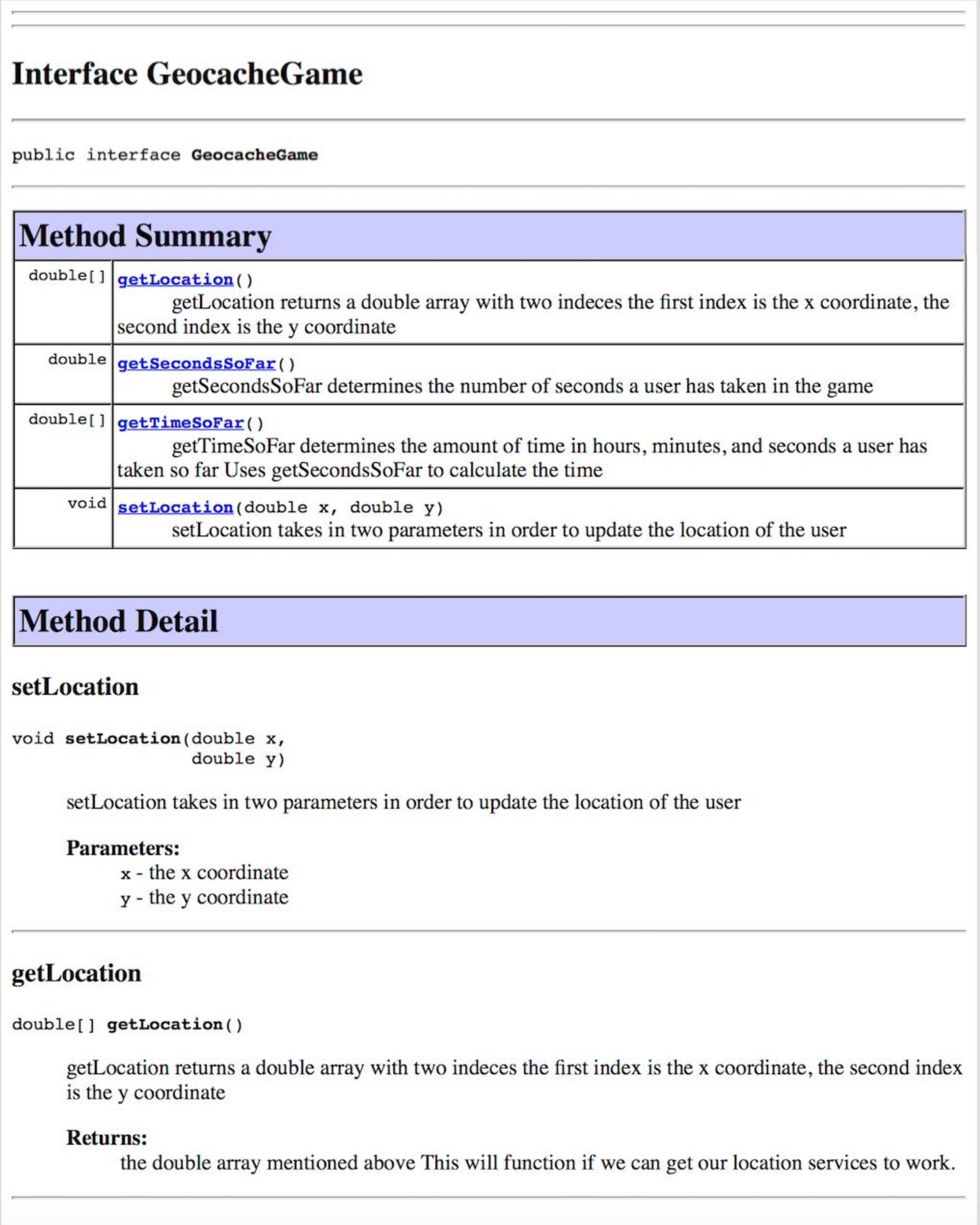
ADTs

We will be using two ADTs in our program.

1. A complete graph called Map that extends Graph from javafoundations package. This will be used to organize the points on the map.
2. LinkedLists, which will be used for two primary functions. The first is to keep track of each node on the map. The second is to keep track of clues associated with each node.

Classes

1. The first class we have defined is GeocacheGame.java. This class provides most of the functionality for the overall game.
2. Map.java is the second class we have defined, and this class will be used to create the structure of how a user will be able to move.
3. Nodes.java is the last class that we have defined so far. The Map object will be made of Nodes, and each Node will have coordinates and clues associated with it.



Interface GeocacheGame

```
public interface GeocacheGame
```

Method Summary

`double[]`	`getLocation()` getLocation returns a double array with two indeces the first index is the x coordinate, the second index is the y coordinate
`double`	`getSecondsSoFar()` getSecondsSoFar determines the number of seconds a user has taken in the game
`double[]`	`getTimeSoFar()` getTimeSoFar determines the amount of time in hours, minutes, and seconds a user has taken so far Uses getSecondsSoFar to calculate the time
`void`	`setLocation(double x, double y)` setLocation takes in two parameters in order to update the location of the user

Method Detail

setLocation

```
void setLocation(double x,
                 double y)
```

setLocation takes in two parameters in order to update the location of the user

Parameters:

`x` - the x coordinate

`y` - the y coordinate

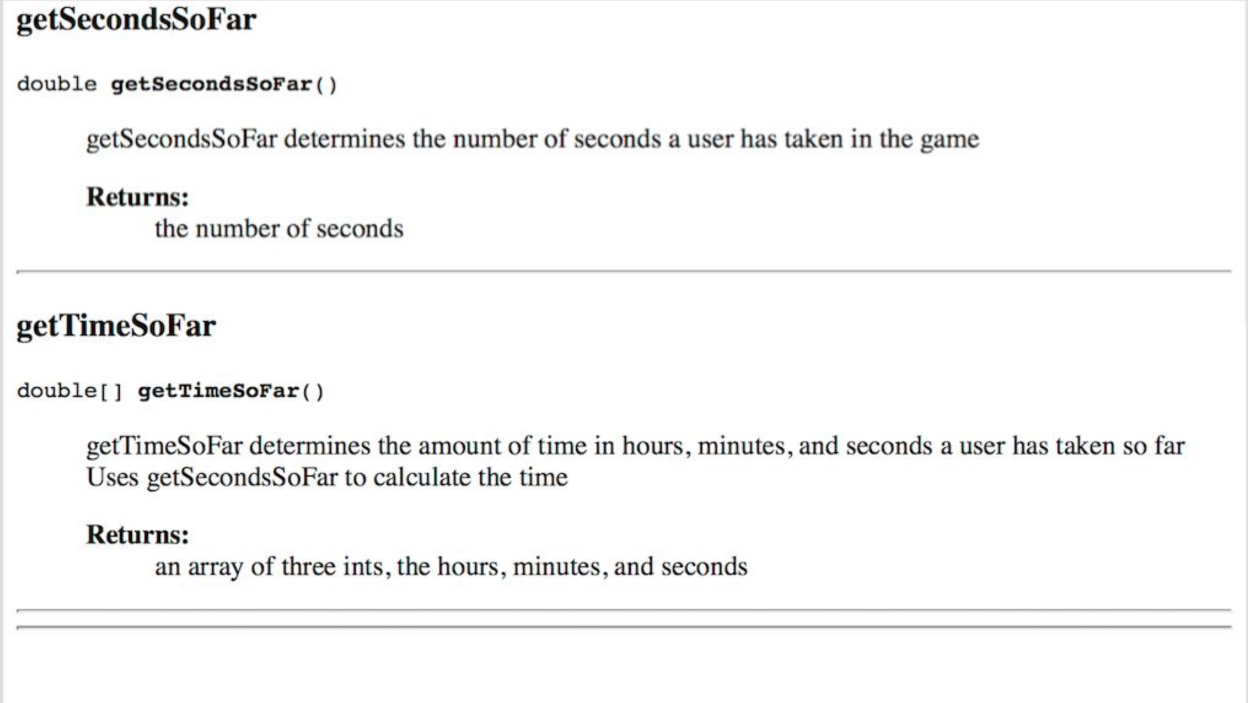
getLocation

```
double[] getLocation()
```

getLocation returns a double array with two indeces the first index is the x coordinate, the second index is the y coordinate

Returns:

the double array mentioned above This will function if we can get our location services to work.



getSecondsSoFar

```
double getSecondsSoFar()
```

getSecondsSoFar determines the number of seconds a user has taken in the game

Returns:
the number of seconds

getTimeSoFar

```
double[] getTimeSoFar()
```

getTimeSoFar determines the amount of time in hours, minutes, and seconds a user has taken so far
Uses getSecondsSoFar to calculate the time

Returns:
an array of three ints, the hours, minutes, and seconds