PROJECT 2 FINAL WRITEUP: GARDEN SIMULATION

Jillian Oestreicher

Software Development 1

Professor Rivas

May 1, 2017

Abstract

This paper will serve to illustrate my final project that I have created over the course of this semester. For this project I created a series of classes that simulate the growth of a garden. I will explain concepts including what this program does, how users interact with it, the physical requirements needed, and the ultimate significance it holds.

Introduction

For project 2, I have decided to create a program that emulates the growth of a garden. I was motivated to create this project because I am interested in objects and inheritance, which I learned a little about in high school. After hearing more in class about this subject I wanted to pursue a program focused around it to better my understanding of it. This project creates a garden of flowers and bushes, allowing for initial user input, and then modifies their rate of growth using a random generation of values.

Detailed System Description

First, I made an interface for items to be planted in a garden. In the interface I created methods to affect the size of the plant to reflect plant growth, as well as to take into account instances of frost and rain and to draw the plant using an x and y location. Then, I wrote a class Flower that implemented PlantInterface. I created instance variables for location, height, width, and growth rate. A constructor takes parameters that indicate the x and y coordinates (as integers) of the base of the flower. I set the initial height of the flower to be 30.0, the width to be 8.0, and the initial growth rate to be .5. For each day the plant grows, the height increases by the

growth rate. Rain increases the growth rate by .1, while frost kills the plant and resets the growth rate, height, and width to 0.0. I then created a test driver to test my Flower class and Interface and to check to make sure it worked. Then I made a FlowerGarden class that represents a collection of plants. It originally contained an ArrayList of type PlantInterface which held flowers. I also created instance variables of length and width. When the grow, rain, frost, or draw message is sent to FlowerGarden, the garden sends a message to all the flowers in the garden. I also added a method plantNewPlants that is called to add several new plants to the garden. The constructor creates the number of flowers whose x and y coordinates are within the garden. I generated the x and y locations for the plants using a random number in the confines of the garden. I then made another driver to compile my program.

I decided I also wanted to add a different kind of plant, so I then made a Bush class. I had to create an abstract class Plant to implement PlantInterface for shared code between the Bush and Flower class, and I extended the classes to Plant instead of PlantInterface. The Bush class was essentially the same as the Flower class, except that it grew in width along with height so it had an extra variable. The Plant class used protected variables so that subclasses could access them. I made a BushGarden like the FlowerGarden, and created another abstract class called Garden to share the code between the gardens. Thus, I had to change FlowerGarden to be extended from Garden. I then made a garden that was half bushes and half flowers called FlowerBushGarden, and a driver called GardenTest to run this garden. The driver asks for an initial input of plants to be planted by the user and then affects them. Then it plants more plants with a randomized number up to 50 to express the growth of the garden. Finally, it outprints the amount of plants grown in the garden.

My UML diagrams are displayed below:

Plant
height: double width: double growth: double xCoord: int yCoord: int
Plant(int, int, double, double) grow(int) rain(days) frost(): void draw(): void toString(): String

Garden
xCoord: int yCoord: int width: int height: int plantNum: int
Garden(int, int, int, int, int) grow(int) rain(int): void frost(): void draw(): void toString(): String plantNewPlants(int): void

Requirements

The physical requirements this project requires includes a computer and a text editor that allows for Java. I used TextWrangler for this project, as well as the Terminal to check my code. I also used class notes and the Intro to Programming textbook as resources for help.

Literature survey

In terms of garden simulations similar to the one I made, I found a journal article explaining the creation of a software simulation that advises gardeners on watering schedules and watering use for their plants. This article was called "SmartGardenWatering: experiences of using a garden watering simulation" and was by professors from the University of Melbourne in Australia. This program had to take into account knowledge of both programming as well as plant management skills. It reminded me of my program, as it emulated plant growth through a software program. Mine was different though, as it took into account instances of frost and rain but did not include watering by human interference. Also, this software application I found is a realistic tool for gardener use, while mine is less realistic and is not necessarily applicable for people to base their actual gardens off of.

User Manual

This system is easy to use. All the user has to do after compiling the GardenTest driver is to input the amount of plants it wants to be initially planted in the garden. The user should not enter a negative number, as is stated in the code, as this is not valid. Also, it is recommended the user enters a whole number as plants are recognized to be planted as whole entities. Once the

user enters their starting amount of flowers, the program will run and outprint the growth of the garden and total amount of plants planted.

Conclusion

In conclusion, my Garden program for this project which imitated the growth of a garden helped me to improve my skills in Java. I learned a lot about objects and inheritance, which are important aspects of this programming language. I am ultimately very proud of my work on this project and believe it showcases what I've learned in Software Development I well.

Bibliograpy

Liang, Y. Daniel. *Introduction to Java Programming: Comprehensive Version*. Boston: Pearson, 2015. Print.

Pearce, Smith, Nansen, and Murphy. “SmartGardenWatering: experiences of using a garden watering simulation.” *OZCHI '09 Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group* (11, 2009): 217-224