Pokémon Safari Zone

CSC 335 – Final Project

All information in this document is subject to change

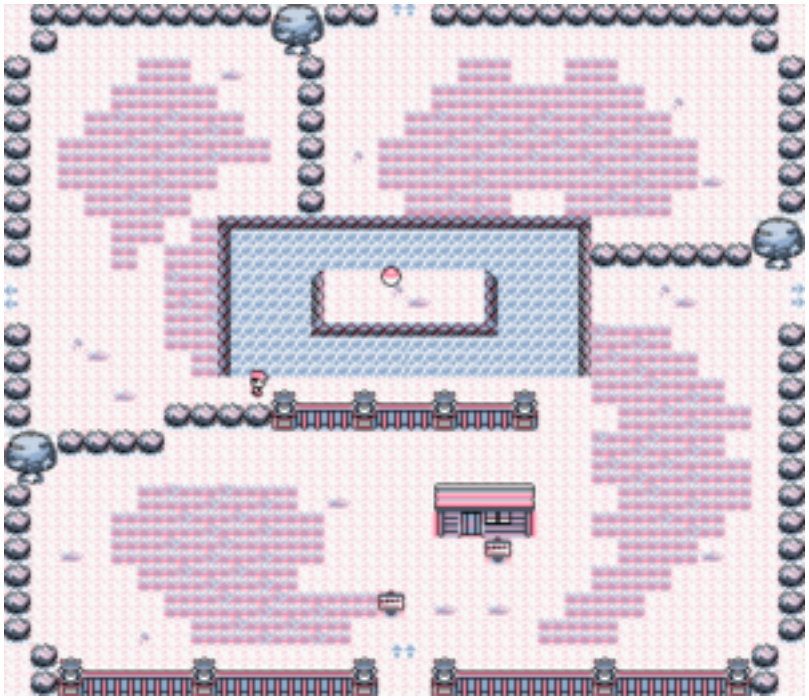

Table of Contents

Overview

If you recall from the classic pokemon games, these games featured a mini-game known as the Safari Zone. Within this zone, trainers receive 30 poke balls as well as the ability to travel 500 steps within the Safari Zone area.

When a wild Pokémon appears, no Pokémon may be sent out to battle it: catching Pokémon here, as in all Safari Zones, requires sheer luck. There are four options in the battle screen: Throw a Safari Ball, throw Bait, throw a Rock, and run away. Throwing Bait makes a Pokémon less likely to run, but makes it harder to catch; while throwing a Rock does the reverse, making it easier to catch but more likely to run. If the player takes too long to catch the Pokémon, it will automatically run away. (Wiki: Kanto Safari Zone)

Trainers will explore this Safari Zone until they have exhausted their steps. After which, some form should display the pokemon captured, items received and the status of the trainer.

Your Task

You are required to create a fully functional Safari Zone Game. To allow the project to be reasonably completed within the time frame, you won't have to implement every feature from every Safari Zone. However, feel free to draw inspiration from past pokemon games. Creativity is encouraged and you can expand on this as much as you want, wow factor points will be given according to your ability to go beyond the project parameters.

In our Safari Zone, the player will be able to select a map, and engage in a battle against and capture pokemon, all the while trying to reach the end of the Safari Zone. Game rules may be tweaked for a multiplayer mode if you feel that new rules are more appropriate – speak with your project manager if you have ideas.

Base Requirements

General

- The game should begin with the ability for the user to select between win conditions.
- The user may or may not choose between maps whether or not the programming group decides to allow transition between maps during gameplay.
- When the win condition is met, users should be able to view the final status of their trainer.
- The player should be able to close your application at any time.
- The player should be able to forfeit the game at any time giving the same behavior as loosing the game.
- The player should be prompted to save the game before closing.

Map

- Maps should be laid out in a grid.
- Maps must contain obstacles that cannot be passed through – the player should not be able to walk across every square!
- There must be at least 2 maps to choose from.
    - Maps do not have to be randomly generated, they can be hard coded beforehand.
- The player cannot see the whole map.
    - Meaning: The screen should show a section of the map centered around the player. When the player moves, this section also moves, like an overhead camera.
    - Example: Map is 100 x 250 squares. But where the user is standing, user only sees 30 x 30 squares. (You can use whatever numbers you want, as long as it fits the other criteria.)
- Map size is up to you but should be large enough for the user to exhaust the amount of steps they begin without travelling to every square. For example, in the original game, you were given 500 steps before you were ejected from the game. Thus, you should have a safari zone that possesses more than 500 squares of walk able area.
- The player must be able to reach every point on the map in 500 steps.
- Transitions between maps DO NOT have to be animated.

Pokémon

- You must be able to encounter 10 unique pokemon of 3 types.
    - 6 Common, 3 Uncommon, and 1 Rare.
- The rate at which you encounter each of these types of pokemon should be different. For example, you are more likely to encounter a common type pokemon as opposed to a rare pokemon.
- In battle, these types of pokemon must also posses unique figures such as:
    - HP
    - Likelihood to run
    - Maximum HP capable of being captured
    - Maximum duration of battle before running
    - Etc.
- Specifically, when you have encountered a pokemon, the behavior of the pokemon will act in accordance with the following description:

> When a wild Pokémon appears, no Pokémon may be sent out to battle it: catching Pokémon here, as in all Safari Zones, requires sheer luck. There are four options in the battle screen: Throw a Safari Ball, throw Bait, throw a Rock, and run away. Throwing Bait makes a Pokémon less likely to run, but makes it harder to catch; while throwing a Rock does the reverse, making it easier to catch but more likely to run. If the player takes too long to catch the Pokémon, it will automatically run away. (Wiki: Kanto Safari Zone)

- Furthermore, these pokemon should be animated both in and out of battle. The trainer should be able to view the pokemon they have captured throughout the game.
- Items should be able to be used on pokemon. For example, restoring HP of a pokemon.

Items

- You must have 3 unique items. (Items that do the same thing, such as restore HP, but by different amounts will be counted as 1 Item.)
- The Safari Ball must be implemented as an item.
- Items must be capable of being added during gameplay, whether that be finding items on the map or winning them in battle.

Trainer

- The trainer shouldn't start out with any pokemon in their possession.
- The trainer at the beginning is given 30 balls and 500 steps.
- When the trainer has exhausted their steps, the game should eject the trainer.
- As the description said earlier, the trainer is only capable of four actions in battle:
    - Throw a Safari Ball
    - Throw a Rock
    - Give Bait
    - Run Away
- During normal gameplay, the trainer should be able to
    - Move about the map
    - Check the pokemon and items the trainer possesses.
    - Use items on both the trainer and the pokemon.
- The trainer may or may not be equipped with an HP. This is based upon the win condition established by the 335 group.

Battles

- Battles should be entirely animated

Win Conditions

- 335 Groups will choose a win condition they see fit for their safari zones.
    - Examples
        - Surviving for a set number of steps
        - Reaching a certain point or landmark
        - Catching a certain number of Pokémon
        - Anything that makes sense to your team (think creatively!)
- You are required to be able to switch between at least two win conditions.

Persistence

- The player should be able to save the game and load a saved game at any time outside of a battle.

Sound

- Your game should have sound effects for all major events as discussed with your project manager.
    - Examples:
        - Throwing a rock

- Throwing bait
- Throwing a Pokéball
- Pokémon entering battle
- Pokémon running away
- Capturing a Pokémon

More Information

For more information on the Safari Zone, along with statistics for game play, I found the wiki site to be incredibly helpful. Yes, I know, it's Wikipedia.

http://bulbapedia.bulbagarden.net/wiki/Kanto_Safari_Zone

Iteration 1: The Overworld

Accept the following Github Classroom assignment. One team member create a new team with your final project team name. Then have the other three team members join that team.

You should now have a private source code repository on GitHub for all team members that can be seen by your project manager.

Tasks

- All team members have added their name to README.md and committed it (all must show up as a committer).
- At least one branch is created and then been pulled into master by a pull request
- Unit tests exist for all classes with 90% or higher code coverage.
- Model package
  - Build and implement an inheritance hierarchy of Pokemon
    - At least one class that is inherited from
    - At least 10 classes inheriting from the abstract class(es)
  - Build and implement an inheritance hierarchy of Items
    - At least one class that is inherited from
    - At least 2 classes inheriting from the abstract class(es)
  - Build and implement a Trainer class + any related subclasses
  - Build and implement a Map class + any related subclasses
    - Should have at least one map implemented (may be hardcoded; randomly generating maps is much more difficult than hardcoding)
    - Map(s) should have some squares that cannot be moved through
  - Most of the model is complete except for encounters
- One win condition is implemented (500 steps is easiest at this point)
- Model is well tested
  - 90% or higher code coverage on all model classes

- Basic GUI
    - Should have graphical GUI showing the map and the trainer
    - Should be able to move around a map using arrow keys on the keyboard
        - **Trainer's movement should be animated**—the trainer should move from one square to another progressively, not instantly appear on the next square
        - No other animations besides this are required
    - Should not be able to move through all squares
    - Game should implement one of the win conditions (again, 500 steps is easiest)
    - No encounters needed at this point! (trainer moves around map and nothing happens. No battle scene, etc)
- Should be able to save the game
    - Should be able to load a saved game
    - Should be able to exit the game

Grading Criteria (100 points) subject to change!!!!

General
____/ 4 Team sent email to all team members and PM listing all tasks and who will complete them
____/ 4 At least two branches were created and pulled into the master branch with a pull request.
____/ 4 All team members have made at least one commit to at least one branch. -1 for any missing team member

Model
____ / 4 Game representation (something that manages all the other models)
____/ 12 Model the Pokemon
____/ 8 Inheritance hierarchy of Items
+4 Abstract class(es) exist and is/are well-implemented
+4 At least 2 inheriting classes, both of which are well-implemented
____/ 10 Trainer class exists and is well-implemented
____/ 10 Map class exists and is well-implemented
____/ 10 Unit tests exist for all classes that are considered part of the model & have 90% or higher code coverage.

GUI
____ / 6 Map is represented with images (ex: sprites)
+2 Images on the map
+4 Not showing the whole map on-screen
____ / 6 The trainer can walk around and collide with obstacles on the map
____ / 10 The trainer's walking from square to square is animated
+4 While walking, the trainer shows a walking sprite
+6 The trainer slides from square to square instead of appearing instantly
____ / 4 General Info: You can see general info like number of steps and Pokemon that

have been caught somewhere on or within the GUI
_____ / 4 Game ends when win condition is met
_____ / 4 Can save, load, and exit the game

Iteration 2: The Battles

In general, this iteration involves implementing the battles, but it also involves finishing the project according to spec as a whole.

Remember: **a "wow factor" is part of the spec**. That is, your project must have something that goes "above and beyond" the requirements. This could be a new feature or a large expansion on an existing feature—anything that makes your PM go "wow!" when they grade your project.

Grading Criteria

- ____/ 12 General
  - ____/ 2 Win condition selection
  - ____/ 2 Map selection/transition
  - ____/ 2 Final status shown at end
  - ____/ 2 Can close
  - ____/ 2 Can forfeit
  - ____/ 2 Save prompt on exit
- ____/ 12 Maps
  - ____/ 1 Grid based
  - ____/ 1 Obstacles
  - ____/ 1 At least 2 maps
  - ____/ 4 Player cannot see entire game
  - ____/ 4 Movement animated smoothly
  - ____/ 1 Maps are comfortable
- ____/ 8 Pokemon
  - ____/ 4 At least 10 unique pokemon belonging to at least 3 unique rarities with the correct minimum distributions with different encounter rates
  - ____/ 4 Items can be applied to Pokemon
- ____/ 8 Items
  - ____/ 3 At least 3 unique items

    - ____/ 2 Safari Ball is an item
    - ____/ 3 Items added during gameplay
- ____/ 8 Trainer
    - ____/ 1 Trainer starts with no Pokemon (or one single default Pokemon, your choice)
    - ____/ 1 Trainer starts with finite steps and balls
    - ____/ 2 Can move
    - ____/ 2 Can check items
    - ____/ 2 Can use items
- ____/ 15 Battles
    - ____/ 5 At least the 4 basic actions in battles
    - ____/ 2 Pokemon transitions animated
    - ____/ 2 Trainer transitions animated
    - ____/ 2 Trainer actions animated
    - ____/ 2 Pokemon actions animated (if applicable)
    - ____/ 2 Transition from map to battle animated (if applicable)
- ____/ 9 Win Conditions
    - ____/ 2 Finite steps condition
    - ____/ 2 Finite balls condition
    - ____/ 3 At least one other condition
    - ____/ 2 Can pursue any condition
- ____/ 10 Persistence
    - ____/ 10 Game state can be saved and loaded when a battle is not occurring
- ____/ 10 Sound
    - ____/ 10 Major events have sounds
- ____/ 12 Wow factor
- ____/ 16 Code Health
    - ____/ 16 90% code coverage for each individual class that should be in the model
- ____/ 5 Grader discretion and unanticipated errors

Task List

Model Updates, Not Including Battles (If Necessary)

- _____ / Trainer
- _____ / Pokemon
- _____ / Items
- _____ / Map

Battles

- _____ / Model implementation
- _____ / GUI implementation
- _____ / Animations

Other

- _____ / Item interface (not a Java interface, but how you see/interact with your items)
- _____ / Caught Pokemon interface (same note)
- _____ / Sound & Music
- _____ / Other Win Condition(s)
- _____ / Unit Tests
- _____ / Wow Factor