Pokémon Safari Zone

CSC 335 Spring 2016 – Final Project
Project Manager: Jeremy Mowery (jermowery@email.arizona.edu)
Authors: Greg DePaul, Jeremy Mowery
All information in this document is subject to change

Table of Contents

Overview

This project is inspired by the Safari Zone of the first generation Pokémon games. In those games the player (trainer) could enter the Safari Zone to catch Pokémon but with slightly modified rules.

In the Safari Zone the player had limited steps (500) and limited pokeballs (30). There were two goals of the Safari Zone. The first was to catch as many Pokémon through random encounters as possible and to reach the house at the end of the map to receive the special item HM 07. Actions in battle with random Pokémon were limited; instead of sending out an owned Pokémon to fight the trainer themselves conducted battle choosing at every turn to either throw a rock (increases the chances of capture and changes of the Pokémon fleeing), throw bait (decreases the chance of the Pokémon fleeing and the chances of capture), throw a pokeball (random chance to capture based on the Pokémon and the number of times bait or rocks have been thrown), or fleeing (ends battle). After a certain number of turns the Pokémon would flee. In this way the Safari Zone was a nonviolent way to capture Pokémon.

In this project you will implement a Safari Zone with the same basic rules described above.

This project will require the following:

- A well tested model for the game including random encounters
- Drawing of graphics (these can be found online)
- Animation with Timers
- Playing music files

The minimum requirements for this project do not include:

- Networking

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 Pokémon 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 Pokémon, 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 Tasks (90% of your grade)

General

- The game should including multiple win conditions that can be pursued at any time.
- The user may or may not choose between maps whether or your team 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 losing the game.
- The player should be prompted to save the game before closing.
- The player should be prompted to load when opening the game if there is a saved state.

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 OR there must be a transition between maps during gameplay.
    - Maps do not have to be randomly generated, they can be hard coded beforehand.
- The player cannot see the whole map at once.
- 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 walkable area.
    - If you are uncertain if your map is too small, it probably is.
- The player must be able to reach every point on the map in 500 steps.
- If the player can transition between maps during gameplay the transitions DO NOT have to be animated.

Pokémon

- You must be able to encounter 10 unique Pokémon of 3 different rarities.
  - At least 6 Common, 3 Uncommon, and 1 Rare.
- The rate at which you encounter each of these types of Pokémon should be different. For example, you are more likely to encounter a common type Pokémon than a rare Pokémon.
- In battle, these types of Pokémon must also possess unique stats such as:
  - HP
  - Initial likelihood to run
  - Maximum HP capable of being captured
  - Maximum duration of battle before running
  - Etc.
- Specifically, when you have encountered a Pokémon, the behavior of the Pokémon 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 Pokémon should be animated both in and out of battle. The trainer should be able to view the Pokémon they have captured throughout the game.
- Items should be able to be used on Pokémon. For example, restoring HP of a Pokémon.

Items

- You must have 3 unique items other than Safari Ball, Rock, and Bait. (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 Pokémon 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
  - Throw Bait
  - Run Away
- During normal gameplay, the trainer should be able to
  - Move about the map
  - Check the Pokémon and items the trainer possesses.
  - Use items on both the trainer and the Pokémon.
- The trainer may or may not have their own HP. This is based upon the win condition established by your team.

Battles

- Battles should be entirely animated
  - All 4 actions need to have a corresponding animation
  - Actions of the Pokémon need to have an animation
  - Transition into and out of battle must be animated

Win Conditions

- At a minimum the game must end when the trainer runs out of steps of Safari Balls, unless otherwise negotiated with your PM.
- You may have any win condition you see fit with approval by your PM some examples include:
  - 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 pursue any win condition.

Persistence

- The player should be able to save the game and load a saved game at any time outside of a battle.
  - Note this can and should be tested in a unit test without writing a file.

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
- Your game should have situation aware music
    - A background song should play for moving around the map
    - A background song should play for the battle
    - The music should switch with the situation in a smooth manner

Code Health

- You must maintain good code health
    - Code coverage of each individual class that belongs in the model package must be 95% or higher
    - MVC should be respected especially with model
    - Tests should not rely on the GUI or on external resources (files, network connections, the system clock, etc...)
    - You should use at least one other branch than master in a nontrivial way

Wow Factor (10% of your grade)

This is by no means an exhaustive list of wow factor features, but merely a list of suggestions for potential extra features. Some of these are much harder to implement than others, and are worth more accordingly. **Talk to your project manager if you have other ideas** to ensure that you receive points for your feature.

- Map Terrain **–** Introduce map terrain that affects how Pokémon function. For example, a Pokémon might run slightly better if they're is in a forest.
- Map Creator – Allow maps to be created by the user and then played in the game. This could also use a GUI to simplify the process
- Map Transitions – Animate the transitions between maps.
- Destroyable Obstacles **–** Obstacles on the map that you must destroy before being able to move over. After being destroyed, you can move over

them freely. Cut and surf are pretty popular in Pokémon games, though these don't have to be equipped to a Pokémon.

- Complex Items – Items beyond the initial requirement of 3 that allow for more complicated interaction with the map. For example, TM to teach techniques such as Cut and Surf.
- Customizable Trainer – Be able to change the clothing / color of the trainer avatar that travels the map.
- Multiuser – User should be able to play on the same map over a network connection. These viewers should be able to see other trainers on the map. These trainers should be able to trade with one another.
- Chat System – Allow users to chat in a multiplayer game.
- Story Mode – Create a story mode to go along with the battles that you are participating in. The more robust the story, the more points you can earn. Discuss with your project manager.
- Complex Animations – Create animation for Pokémon movement and/or Pokémon combat and/or other game actions. Especially noteworthy animation can earn more points. Discuss with your project manager.
- Evolving Pokémon – Create a system where a Pokémon can evolve into a more powerful Pokémon type when certain conditions are met. Water and Fire Stones are some items that perform this feature.
- Working Pokedex – Record the captured and seen Pokémon in the game.
- Additional Maps – Any maps over the original requirement of two will be worth an additional point. This can go along with the "Story Mode" extra credit option discussed above.
- Additional Enemies/Battles – Extending the ability for trainers to battle other trainers. In this case, it is allowed for a trainer to use their Pokémon to battle.
- Day and Night Cycle – Have the time in the game change in some noticeable way with specific impacts on the game.
- Have an executable for your game – Have the ability to produce an executable Jar or plain executable for your game that stands alone.
- Use advanced techniques – Produce better software by utilizing extra libraries, frameworks, or build systems, some examples include Guava (useful utilities like ImmutableList), Mockito (mocking to improve your tests), Dagger (dependency injector), Bazel (build system), Gradle (build system), and others as discussed with your project manager.

More Information

More information on the Safari Zone, along with statistics for game play can he found here: http://bulbapedia.bulbagarden.net/wiki/Kanto_Safari_Zone