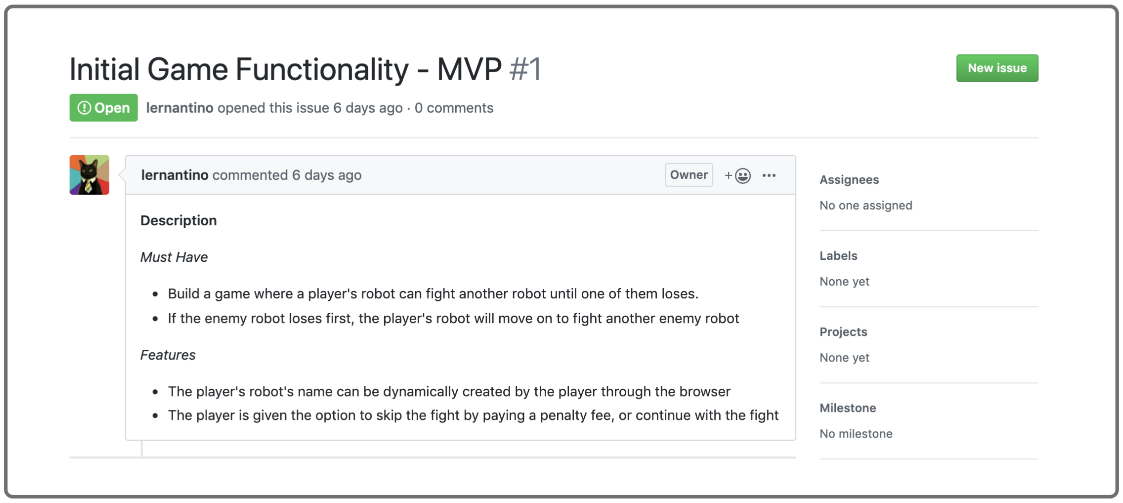
3.2.1 Introduction

In the first lesson, we created a basic `fight()` function that lets us exchange attacks with the enemy robot. However, we can only fight one battle round with one enemy robot. Because we can't determine the winner or even how to win, the game in its current state won't impress the judges.

Let's build on our previous work and increase the Robot Gladiators game's complexity. We can consider it part of an MVP to introduce more enemy robots and exchange multiple rounds of attack until a combatant has been defeated. In fact, this will make it possible to actually play and win the game! In order to give the judges a sense of how the game works, we need to add these features.

Let's look at the GitHub issue again to see where we stand in relation to the MVP:


Initial Game Functionality - MVP #1

New issue

Open lernantino opened this issue 6 days ago · 0 comments

lernantino commented 6 days ago Owner

Description

Must Have

- Build a game where a player's robot can fight another robot until one of them loses.
- If the enemy robot loses first, the player's robot will move on to fight another enemy robot

Features

- The player's robot's name can be dynamically created by the player through the browser
- The player is given the option to skip the fight by paying a penalty fee, or continue with the fight

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone
No milestone

As you can see, we need to incorporate some essential game functions:

- Fighting a robot until defeat
- Fighting another robot after defeat
- Receiving the robot's name from the player
- Allowing players to fight or skip

ON THE JOB

The `fight()` function established a foundation of basic game functionality that we can build upon. A big part of becoming proficient in JavaScript is knowing how to create and use functions as building blocks that work together to accomplish a program's overarching goal.

By the end of this lesson, you will be able to:

- Create a `for` loop to execute multiple statements repeatedly in a concise format.
- Iterate through an array of elements.
- Use conditional statements to control the flow of the application.
- Debug the application using the `debugger` statement and Chrome's DevTools.