Coding Standards

Introduction

Battleboats is a game where players attempt to sink ships placed on a 10x10 grid using bullets. This document outlines coding standards to maintain consistency, readability, and efficiency across the codebase.

Naming Conventions

- Use descriptive names that convey the purpose of functions and variables.
- Follow the CamelCase convention for functions and variables.
- Avoid excessively long names (>24 characters) while ensuring clarity.
- Avoid abbreviations that may obscure meaning.
- Variables representing game elements should have clear names indicating their role (e.g., grid, bullets_left).

Style and Spacing

- Use a two-space tab standard indentation.
- Include comments to explain the purpose of functions and complex logic blocks.
    - Comments for methods follow this structure:
        - def validate_grid_and_place_ship(start_row, end_row, start_col, end_col):
        - # will check the row/column to see if a ship can be placed
        - # * * *
    - Inline comments:
    - game_over = False # game status
- Use spacing between lines of code where necessary for readability.
- Avoid excessive spacing that may hinder code comprehension.

Function Standards

- Functions should have a clear and singular purpose, adhering to the Single Responsibility Principle.
- Function parameters should be descriptive and clearly indicate their purpose.
- Avoid redundant or overly complex logic within functions.
- Ensure consistent error handling and input validation.

Use of Global Variables

- Minimize the use of global variables where possible.
- When global variables are necessary, ensure they are clearly documented and their usage is justified.

Making Pull Requests

- Provide clear and descriptive titles for pull requests summarizing changes.
- Include comments explaining the purpose of functions and complex logic blocks.
- Adhere to all coding standards, including naming conventions and styling.
- Update relevant unit tests for changes being made.
- Assign pull requests to appropriate reviewers based on their expertise.

Reviewing Pull Requests

- Review pull requests focusing on logic, functionality, and adherence to coding standards.
- Ensure code is clear, readable, and maintainable.
- Pay attention to error handling and edge cases.
- Provide constructive feedback to authors and suggest improvements where necessary.

Testing Requirements

- Define comprehensive test cases covering all aspects of the game logic and functionality.
- Test functional requirements such as ship placement, shooting mechanics, and game-over conditions.
- Test non-functional requirements such as performance, scalability, and error handling.