CS 1530 – SPRINT 2 DELIVERABLE

15 FEB 2017

https://github.com/danieljustice/Hnefatafl

Team Scrumbags

Timothy Kang (TimKang95)

George Mattesi (Mattesigp)

Daniel Justice (YuzuFugu)

Zachary Gannon (Gannon93) (Scrum Master)

Accomplishments

For the second sprint (Sprint 2), the team accomplished all features we sought to complete by the provided delivery date, as well as progress on one feature that was defined and placed in the backlog for the third sprint (Sprint 3), initially. The team used Trello to document progress, GitHub to share code, and Slack to instant-message.

During the retrospective for the first sprint, the team defined the following:

- What went well:
  - Scrum meeting went quickly and concisely
  - Planned enough stories to fulfil requirements
  - Division of labor was fair
  - Success knowledge transfer of Gradle applications
  - Learning to play the game
- What didn't go well:
  - Coordinating everyone's vastly different schedules
  - Slack notifications were not being pushed to everyone
- What can we improve:
  - Using @ in Slack to force push to specified people
  - Each member uses their own branch for development
  - Code is peer-reviewed before being merged into master

During the second sprint, the team continued to utilize what the retrospective had identified as successful practices by dividing the labor with all member's agreements and using Slack to communicate. To build upon identified improvements, the group employed the use of @ in Slack to notify potential code reviewers when a member's branch was ready to be reviewed for merging into master.

For the second sprint, the labor was divide as follows:

- Timothy Kang: New game and initial setup development and integration
- George Mattesi: Piece movement, movement validity, and turn enforcing development, integration, and unit testing
- Daniel Justice: Asset creation, integration, and unit testing
- Zachary Gannon: Scrum Master, documentation/write-up, defect testing
- All: Peer review

The division of labor, like the first sprint, was divided judiciously with each member's schedule and capacity in mind during the planning meeting and brought little conflict or disagreement.

User Stories

The team completed all the stories that were scored for the second sprint. The stories completed were related to the core functionality, stability, and maintainability of the game. The total velocity for the second sprint was 28 points. The following are the completed user stories, in order of priority:

- **Functionality: New Game (4 story points)**
  - As a user
  - I want to create a new game or reset the board
  - So that the game can start with a fresh board
- **Functionality: Pieces Can Move (4 story points)**
  - As a user
  - I want to move pieces on the board
  - So that the game can operate as intended
- **Functionality: Turns (2 story points)**
  - As a user
  - I want to restrict player movement with turns
  - So that players may not interfere with another's pieces
- **Functionality: Sprites (4 story points)**
  - As a developer
  - I want to display sprites for game pieces
  - So that I may provide a better aesthetic to the game
- **Code Quality (2 story points)**
  - As a developer
  - I want to write or cleanse code to be succinct and readable
  - So that other members may follow and update code
- **Documentation (2 story points)**
  - As a developer
  - I want to write commented code
  - So that the program may be easily maintained by others
- **Unit Tests (2 story points)**
  - As a developer
  - I want to write unit test for computational methods
  - So that the game functionality may be testable
- **Write Up (8 story points)**
  - As a development team
  - We want to provide an end-of-sprint summary to the customer
  - So that the customer can be informed of progress on their product

User Stories cont.

The following stories were discussed while planning for the second sprint, but were not critical enough to score and delegate resources to the second sprint, as the stories were not crucial to the customer's requirements for the second sprint.

Backlog for future sprints:

- **Functionality: Legal Moves**
  - As a user
  - I want to ensure all pieces move legally (like Rooks)
  - So that no player can cheat and gain an advantage
- **Functionality: Save Game**
  - As a user
  - I want to save the current game state
  - So that I may preserve the game state
- **Functionality: Load Game**
  - As a user
  - I want load a saved game state
  - So that I may proceed in a saved game

Decisions

The team planned the second sprint’s user stories based mostly upon the code requirements supplied by the customer and the first sprint’s retrospective results.

The stories with the higher priority were those related to the core functionality of the game, such as game start, board setup, piece movement, and turn enforcing. These stories were placed at a higher priority because of time sensitivity and complexity, which also increased their respective story point scores. The scores given to stories also reflected the points given to similar development stories from the first sprint, in that the scores were given more buffer time to better accommodate developer availability and past development time durations.

The stories with the lower priority also had lesser story points as they did not require much development time, such as in-line documentation and ensuring code quality. The write up was given low priority but high story points in that it was planned to be one of the last stories to be completed and was expected to take a larger effort than stories of similar priority.

One story from the backlog that was planned initially for the third sprint, Functionality: Legal Moves, was started with the additional capacity and development of relevant stories from the second sprint, but was not completed during the second sprint due to defect 1 listed below. Therefore, the story is still planned to be completed in a future sprint. The second defect became its own story for a future sprint because it does not detract from the originally defined story for Functionality: New Game, nor the requirements provided by the customer.

Defects

We found the following defects while manual testing. These defects were not fixed as they do not invalidate any requirements provided by the customer for the second sprint and were discovered later in the development testing and review process due to their manual nature.

1. Player can overwrite own pieces

   Reproduction Steps: Start a new game, click a game piece of the team whose turn it is, and click the location of an existing piece of the same team to “move” it.

   Expected Behavior: The game pieces swap places or no move occurs at all.

   Observed Behavior: The piece initially clicked (moving piece) either overwrites the existing piece or is deleted, creating a deficit for that team.

2. New game first player is inconsistent

   Reproduction Steps: Start a new game, play one turn (attackers), click new game button, play one turn (attackers).

   Expected Behavior: Upon the second instance of the game, the attackers should start the game.

   Observed Behavior: Upon the second instance of the game, the defenders start the game on turn 1.