


Chutes and Ladders

Team Triple Threat



William Garlington, Manny Uzoma, Steve Deibert

Project Overview



Game Choice: Chutes and Ladders

- Traditional board game popular among children
- Involves dice rolling and moving player tokens across a game board w/chutes(slides) and ladders

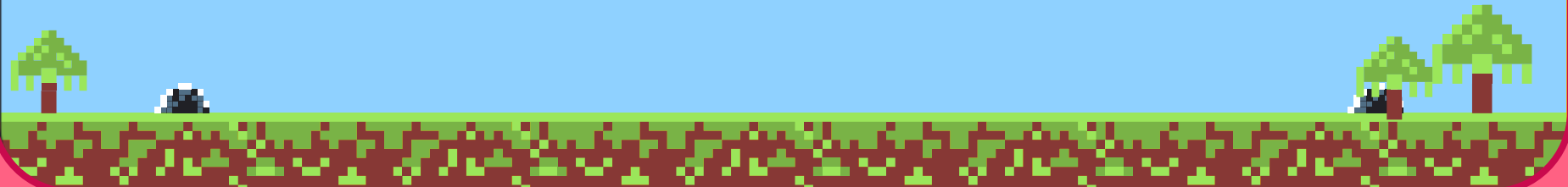
Main Features:

- Graphical representation of the game board with chutes and ladders
- Player tokens moving across the board based on dice rolls
- Interactive roll button for players to take turns
- Game over detection and option to restart or close the game
- User-friendly GUI with intuitive navigation and interaction

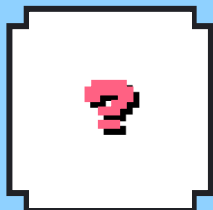


Assumptions:

- Players are familiar with the rules of Chutes and Ladders

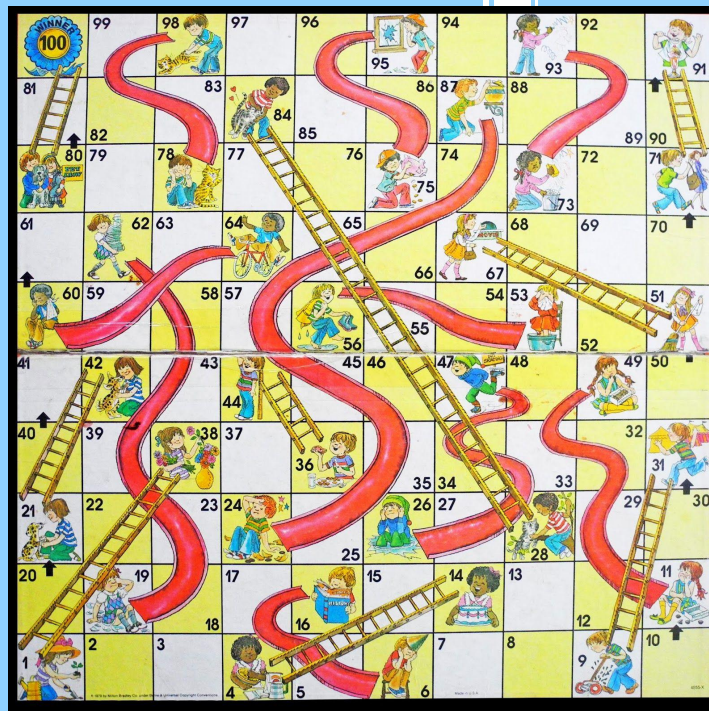


What is Chutes & Ladders ?



Board

- 1-4 Players standard
- A board with 10x10 grid filled with chutes and ladders connecting random tiles
- Ladders, when landed on, move the player upwards to the “top” connected square
- Landing at the “end point” of a chute/ladder has no effect
- Players are board icons reflecting position
- Players roll dice to advance “upwards”
- A player reaching tile 100 wins and ends the game





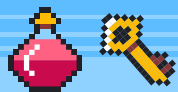
Project Requirements

Functional

- ☐ Game Board Displays
- ☐ Player Movements
- ☐ Turn-Based Gameplay
- ☐ Roll Button Functionality
- ☐ Game Over Detection

Non-Functional

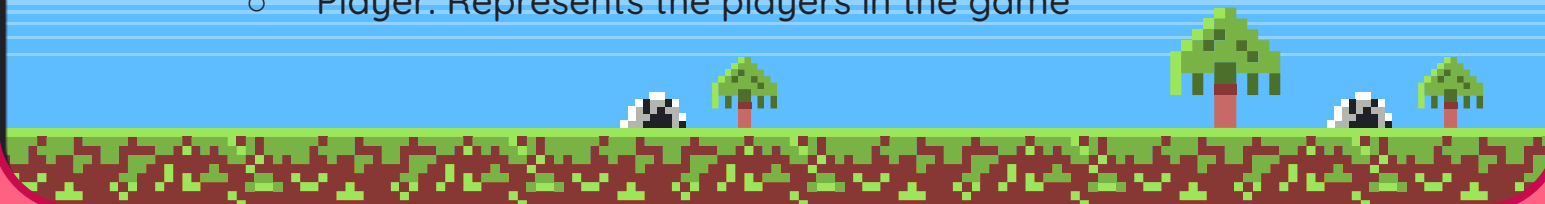
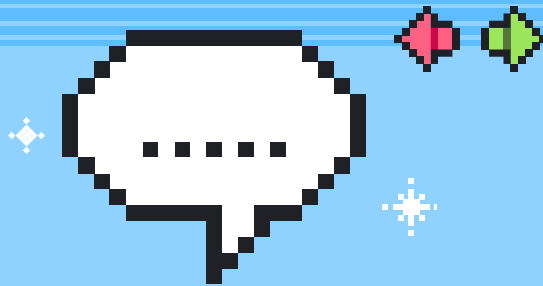
- ☐ Design an intuitive and visually appealing UI
- ☐ Ensure responsiveness and smooth transitions during gameplay
- ☐ Ensure flexibility to accommodate changes and additions
- ☐ Provide clear instructions and feedback to guide users through gameplay
- ☐ Ensure accessibility for players of all ages and abilities



Solutions Approach

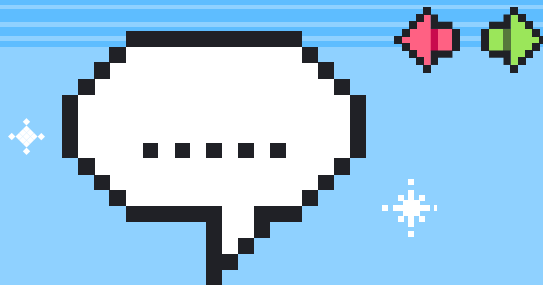
Major Components:

- MVC
 - Control: Manages the game flow and player actions
 - GameBoard: Represents the game board with chutes and ladders
 - Interface: Handles the GUI aspects, including rendering the game board and players
 - Player: Represents the players in the game





continuation

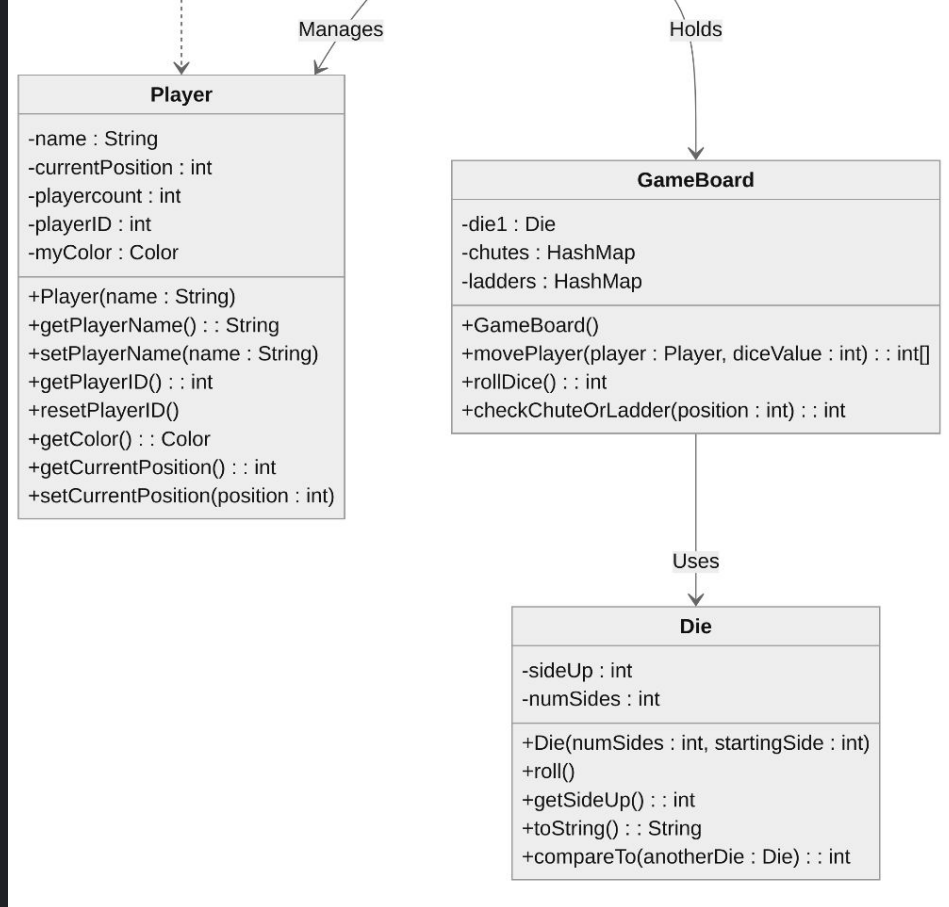
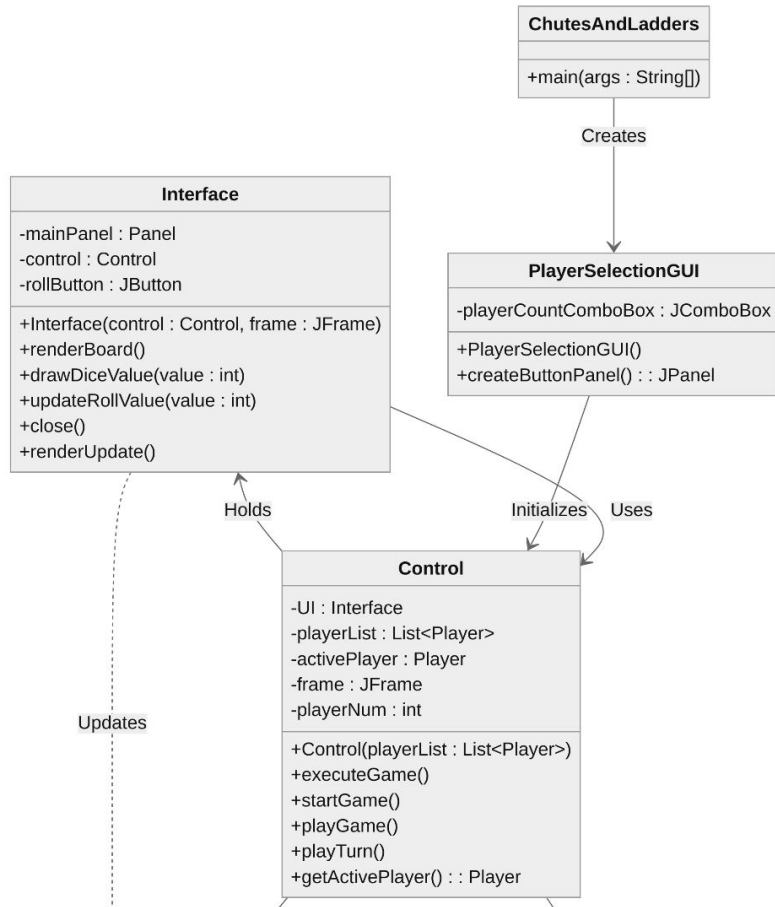


Game/UI Features:

- Graphical representation of the game board with chutes and ladders
- Player rendering on the board to visualize their positions
- Roll button for players to take turns
- Movement from chutes and ladders



UML Diagram



Collaboration Approaches

Discord group chat to delegate tasks and get status updates

Development was always done on separate branches and then merged into main when functionality was confirmed

Branches rarely had conflicts because group members tried to work narrowly within their communicated feature

What we could've done differently?

Group code walkthroughs and programming sessions to keep people on the same page about program flow

Stricter requirements on checking pull requests pre-merge



Testing, Validation, Acceptance

Testing approaches

We considered our project “deliverable” on a functional level when it became possible for multiple players to complete a round

Due to the simpler nature of our project and the heavier emphasis on complex GUI interactions, we didn’t write any unit testing for methods

Testing was mainly done through console logging and visual confirmation of success

LIVE DEMO

yippie

Summary

Biggest lesson learned:

**START CODING
ON TIME**



**TTT
OUT**

thanks - ttt