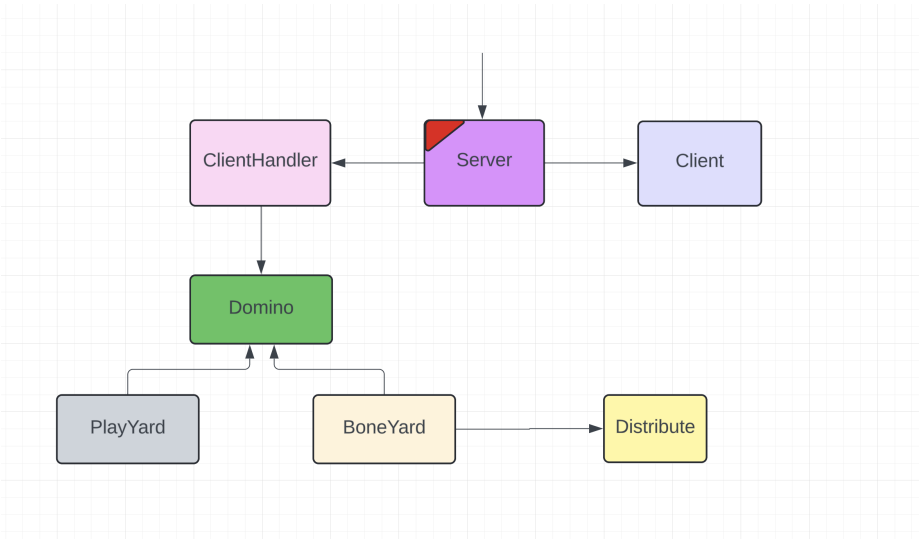
Object Design of Project-5 Networked Domino

Somiyo Rana, Anurag Shrestha, Suman Kafle

1 Proposed Design



2 Description

Server: The Server class acts as the backbone of the networked Domino game, managing all core functionalities related to game logistics. This class initializes the game environment, including setting up a server socket to accept connections from players. Once the server is running, it handles incoming client connections by spawning ClientHandler instances, each dedicated to managing the communication with a specific client. The Server also orchestrates the overall game flow, processes incoming moves from clients, broadcasts game state updates, and determines game outcomes based on the rules of Dominoes.

Client: The Client class serves as the interface through which players interact with the game. It is responsible for establishing a connection to the server, handling user inputs, and displaying game updates. The class sends player commands to the Server and receives responses, which include game state

updates and prompts for player action. This class ensures that players are continuously engaged by providing real-time updates and maintaining an interactive gameplay experience.

ClientHandler: The ClientHandler class is essential for managing individual client connections in a multi-threaded environment. Each instance of this class is responsible for one player, handling all data transmission to and from that client. The ClientHandler processes commands sent by the client, forwards these commands to the server logic to determine the next steps, and sends back the responses. It acts as an intermediary that ensures messages are accurately exchanged between the server and its clients, facilitating a synchronized multiplayer experience.

Domino: The Domino class represents an individual domino piece, which is a fundamental component of the game. Each domino has two ends, each with a number of dots, which are crucial for game play. This class provides methods to access and manipulate the domino's data, such as checking its compatibility with other dominos based on the game rules. It encapsulates all attributes and behaviors specific to a domino, ensuring that the game logic can effectively manage domino interactions.

Boneyard: The BoneYard class manages the collection of unused dominos, serving as the draw pile from which players can draw new dominos when they are unable to make a move. This class is responsible for initializing the dominos at the start of the game and shuffling them to ensure random distribution. During the game, it handles requests from players to draw dominos and keeps track of the remaining pieces to inform players when the draw pile is depleted.

PlayYard: The PlayYard class represents the central playing area where all the action takes place. It manages the layout of dominos that have been played, enforcing the rules about how dominos can be placed in relation to each other. This class checks each player's move for validity and updates the arrangement of dominos on the board. It is pivotal in maintaining the state of the game, determining possible moves, and ensuring that the game progresses according to the established rules.

Distribute: The Distribute class is tasked with the initial distribution of dominos to the players at the start of each game. It ensures that each player receives the correct number of dominos and that the distribution is fair and random. This class interacts with the BoneYard to pull dominos from the unused pool and assign them to players, setting the groundwork for the game dynamics and player strategies.