Pegs and Dice

Olivares Enthusiasts

Evan Delanty | David Sosa | Matt Benson

Project Overview

- What's the game? Pegs and Dice
  - Turn Based Dice Game
- Main Features
  - Rolling
  - Moving pieces
  - Comboing
    - Choosing a combo
    - Re-rolling and comboing again
- Assumptions
  - The player knows how to do basic math
- Limitations
  - Comboing, you can't combo every dice

Game Description

- Where did it come from? Good question!

- What are the rules of the game:
  - Every player gets a board, pegs, and six dice
  - Current player rolls dice and picks a combo
  - Move pegs in accordance with the combo.
  - Turn ends when the player "Farkles"
  - Players will play until a player's top row is filled, that player is declared the winner.

- What does it look like?

Pegs and Dice

- Starting Screen
- Multiplayer
- Singleplayer

Pegs and Dice

Olivares Enthusiasts
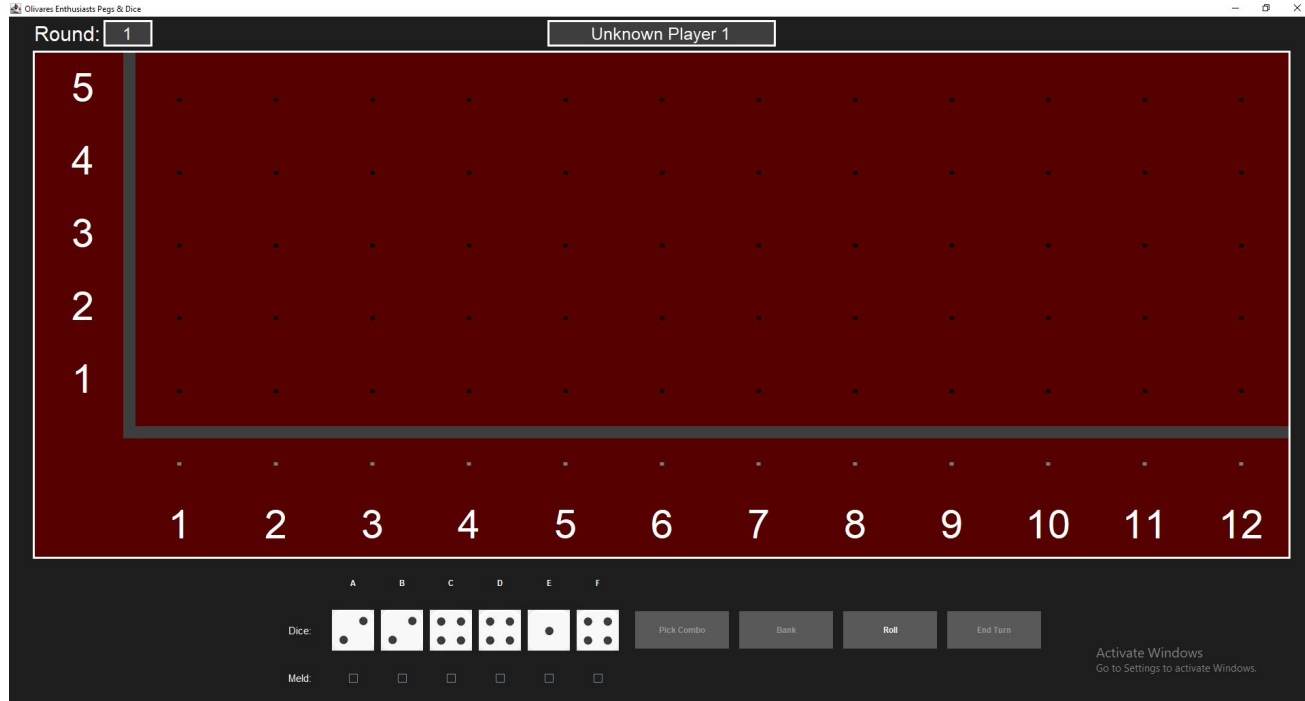
Presents

Pegs & Dice

Enter Names Below
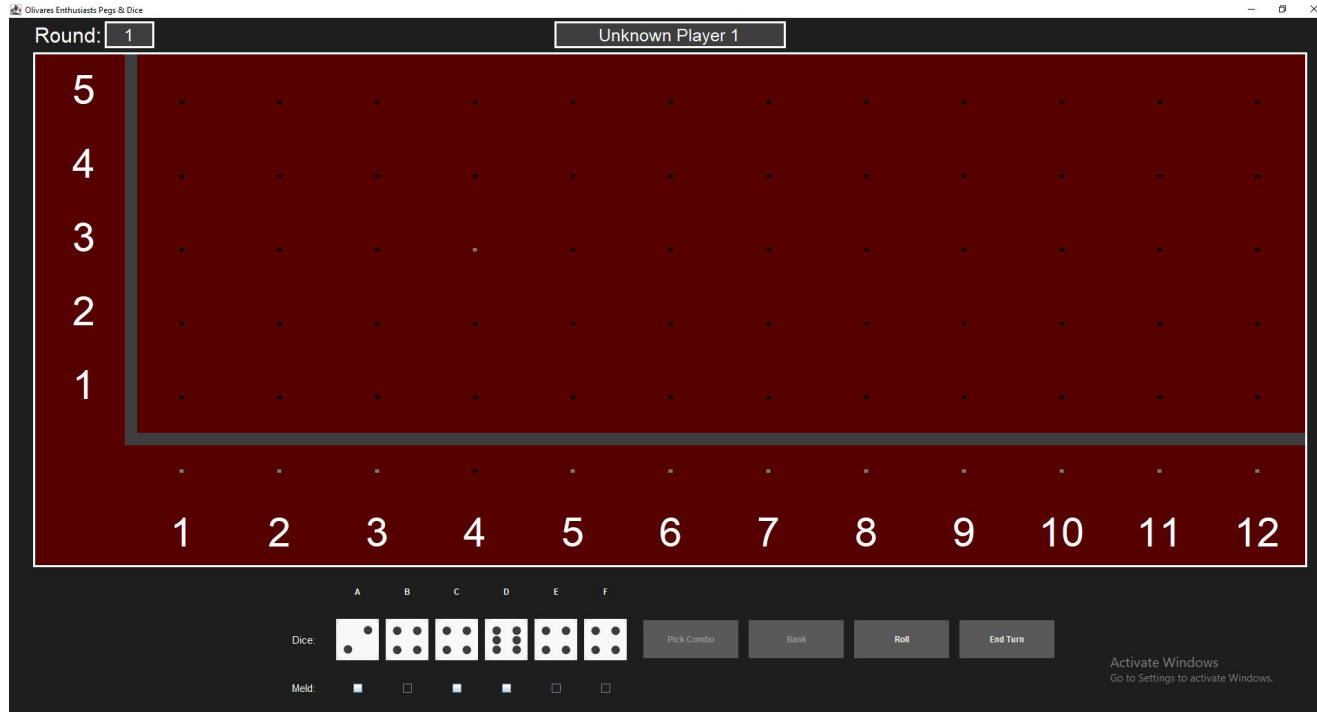
Player 1

Player 2

Unknown Player 1

- Starting Screen
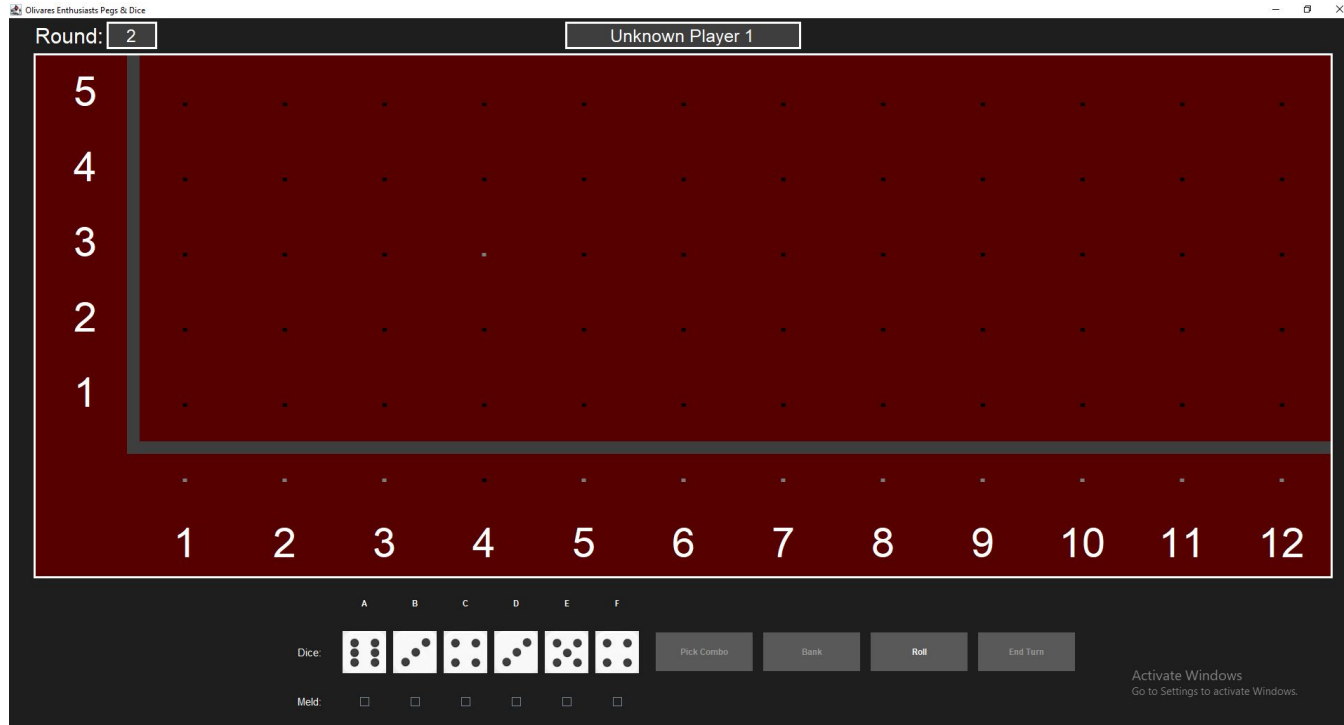- You have names

Pegs and Dice

- Before round starts
- Disabled buttons

Pegs and Dice

In this instance, we chose the dice with the values of 4

After a Round

- Round Count Updated
- Disabled Buttons

Project Requirements

Functional

- A player must be able to simulate a dice roll and be able to choose a combination or number based on their inputs.

- A player must be able to see a live representation of their current board state and be able to see which pegs moved.

- A player must be able to reroll unused dice to try to create additional combinations with the same sum as their initial combo. If a player uses all of their dice of fills a column entirely, they should get a new set of dice and continue their turn with a new combination.

Non-functional

- A player should know exactly when a button is pressed from visual feedback.

- A player should be able to see the results of their input in under 5 seconds of system loading.

- A player should be able to visibly read all text on the screen no matter their screen resolution.
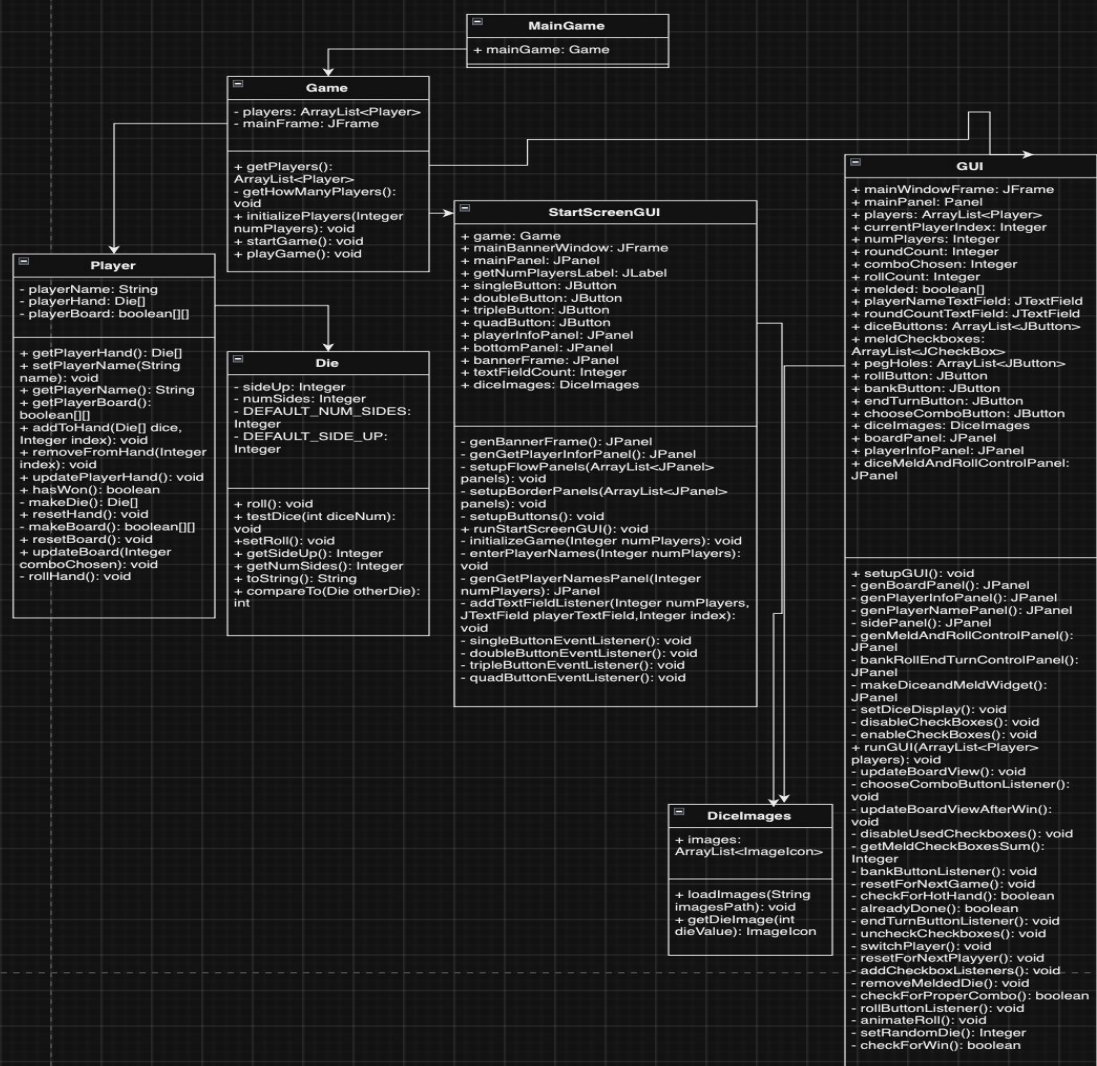
Project Solution Approach

- What are the major components in your solution design?
  - Technical Board (Player Board)
  - Board View
  - Player hand
  - Chosen Combo
- What game/ui features did you really identify and work towards in your take on your game?
  - Have a round counter
  - Have great user feedback
    - Dice rolling
    - Disabled buttons
    - Name changes
  - Keep it simple!

UML Design

MainGame
+ mainGame: Game

Game
- players: ArrayList<Player>
- mainFrame: JFrame

+ getPlayers():
ArrayList<Player>
- getHowManyPlayers():
void
+ initializePlayers(Integer
numPlayers): void
+ startGame(): void
+ playGame(): void

Player
- playerName: String
- playerHand: Die[]
- playerBoard: boolean[][]

+ getPlayerHand(): Die[]
+ setPlayerName(String
name): void
+ getPlayerName(): String
+ getPlayerBoard():
boolean[][]
+ addToHand(Die[] dice,
Integer index): void
+ removeFromHand(Integer
index): void
+ updatePlayerHand(): void
+ hasWon(): boolean
- makeDie(): Die[]
+ resetHand(): void
- makeBoard(): boolean[][]
+ resetBoard(): void
+ updateBoard(Integer
comboChosen): void
- rollHand(): void

Die
- sideUp: Integer
- numSides: Integer
- DEFAULT_NUM_SIDES:
Integer
- DEFAULT_SIDE_UP:
Integer

+ roll(): void
+ testDice(int diceNum):
void
+setRoll(): void
+ getSideUp(): Integer
+ getNumSides(): Integer
+ toString(): String
+ compareTo(Die otherDie):
int

StartScreenGUI
+ game: Game
+ mainBannerWindow: JFrame
+ mainPanel: JPanel
+ getNumPlayersLabel: JLabel
+ singleButton: JButton
+ doubleButton: JButton
+ tripleButton: JButton
+ quadButton: JButton
+ playerInfoPanel: JPanel
+ bottomPanel: JPanel
+ bannerFrame: JPanel
+ textFieldCount: Integer
+ diceImages: DiceImages

- genBannerFrame(): JPanel
- genGetPlayerInforPanel(): JPanel
- setupFlowPanels(ArrayList<JPanel>
panels): void
- setupBorderPanels(ArrayList<JPanel>
panels): void
- setupButtons(): void
+ runStartScreenGUI(): void
- initializeGame(Integer numPlayers): void
- enterPlayerNames(Integer numPlayers):
void
- genGetPlayerNamesPanel(Integer
numPlayers): JPanel
- addTextFieldListener(Integer numPlayers,
JTextField playerTextField,Integer index):
void
- singleButtonEventListener(): void
- doubleButtonEventListener(): void
- tripleButtonEventListener(): void
- quadButtonEventListener(): void

GUI
+ mainWindowFrame: JFrame
+ mainPanel: Panel
+ players: ArrayList<Player>
+ currentPlayerIndex: Integer
+ numPlayers: Integer
+ roundCount: Integer
+ comboChosen: Integer
+ rollCount: Integer
+ melded: boolean[]
+ playerNameTextField: JTextField
+ roundCountTextField: JTextField
+ diceButtons: ArrayList<JButton>
+ meldCheckboxes:
ArrayList<JCheckBox>
+ pegHoles: ArrayList<JButton>
+ rollButton: JButton
+ bankButton: JButton
+ endTurnButton: JButton
+ chooseComboButton: JButton
+ diceImages: DiceImages
+ boardPanel: JPanel
+ playerInfoPanel: JPanel
+ diceMeldAndRollControlPanel:
JPanel

+ setupGUI(): void
- genBoardPanel(): JPanel
- genPlayerInfoPanel(): JPanel
- genPlayerNamePanel(): JPanel
- sidePanel(): JPanel
- genMeldAndRollControlPanel():
JPanel
- bankRollEndTurnControlPanel():
JPanel
- makeDiceandMeldWidget():
JPanel
- setDiceDisplay(): void
- disableCheckBoxes(): void
- enableCheckBoxes(): void
+ runGUI(ArrayList<Player>
players): void
- updateBoardView(): void
- chooseComboButtonListener():
void
- updateBoardViewAfterWin():
void
- disableUsedCheckboxes(): void
- getMeldCheckBoxesSum():
Integer
- bankButtonListener(): void
- resetForNextGame(): void
- checkForHotHand(): boolean
- alreadyDone(): boolean
- endTurnButtonListener(): void
- uncheckCheckboxes(): void
- switchPlayer(): void
- resetForNextPlayyer(): void
- addCheckboxListeners(): void
- removeMeldedDie(): void
- checkForProperCombo(): boolean
- rollButtonListener(): void
- animateRoll(): void
- setRandomDie(): Integer
- checkForWin(): boolean

DiceImages
+ images:
ArrayList<ImageIcon>

+ loadImages(String
imagesPath): void
+ getDieImage(int
dieValue): ImageIcon

Team Collaboration Approaches

- How did we communicate?
  - Communication
    - Text and Group Meetings
  - Git
    - Git branches
    - Pull Requests
    - Code Reviews
  - Lessons Learned
    - Working with a team can be hard
      - It's impossible to know what people are thinking
  - Mainly Individual Coding
    - Communication about overall project design

Testing, Validation, and Acceptance Plan

- What testing approaches did you use?
  - Unit Tests
  - User Tests
    - Friend who created the game, and other people who were interested in the project that we were building
- What's the #1 way you'll be able to say "Our project is deliverable"?
  - It works!
  - We got to all our requirements from our initial report
  - Nearly nailed the UI

Our sketch

- Basically the same
- Button addition

Live Demo Time!

Summary

- We learned a lot
  - Don't expect to just hop into the code and start working
    - There's always a learning curve
  - Good documentation is very important
    - Comments!
    - Variable and Function names
    - Reading others code can be challenging without it
  - Communication is key!
    - Don't work on the same thing