CPSC8710 – web game group project
10/23/2023

Kingdoms: Game and Project Overview

Hazim Alzorgan
halzorg@clemson.edu

Bhushan Marathe
bmarath@clemson.edu

Manan Parikh
mjparik@clemson.edu

Introduction

As we already entered the age of friends meeting up and hanging out on Zoom, Discord, Microsoft Teams, and wide variety of online meeting platforms, it is becoming a necessity to transform our beloved game night games (like Pictionary, charades, …etc.) into online games that can be played through interactive gameplay on such meeting platforms. One of those games and the one we decided to develop is Kingdoms.

Kingdoms is an interactive browser-based we developed in hoping to indulge friend parties into a competitive game of deception, where your knowledge of how each of your friends is key to winning the game. Kingdoms is simply a game where players compete to capture each other's kingdoms through a combination of bluffing strategies with randomness. The game offers a visual representation of kingdom conquest through grid world map overlay.

The main objective of this game is for a player to conquer all other player's kingdoms, proving they are the best friend who even knows how you think. A player captures kingdoms by selecting them during their turn. However, there are some "fake" kingdoms in the selection and selecting them ends a players turn prematurely.

This game is navigated by one user we call "The Moderator" who ensures everyone plays the game fairly and by the rules. The moderator will receive each player's made-up kingdom name and inputs them into the prompted fields, they also get to make up a number of fake kingdom names to confuse the players.

Setup and Initialization:

The game is very simple, it starts by asking the moderator for player names and player kingdom names, and then transitions into the grid capture mode where the main gameplay takes place. We can simplify the game setup as follow:

1. After the moderator starts the game, they are asked to define the number of real kingdoms (which is also the number of players), and the number of the fake kingdoms.
2. Next page will ask the moderator to disable their screenshare as they input the player names and the kingdom names (Future work includes adding a function to block screenshare for this page). On this page, the moderator will input the kingdom names with their corresponding player names that are sent to them privately on the meeting chat, hence the screen share disabling request.
3. Once the total number of the kingdoms and players, as well as their names are decided, the game will display a strategic conquest map in the middle with a grid overlay. On the right side of the map, we have our player names which are iterated through throughout the game. And on the right side of the map, the kingdom name selection buttons are shown where the players kingdom names the moderators fake names are displayed.

4. Each player is assigned a color randomly and the map grids will show those colors accordingly, this way we can see who controls the greatest number of territories on the map. Each player also has a time limit to tell the moderator what kingdom they would like to conquer before their time runs out and the game iterates to the next player.
5. When a kingdom is selected, if it's a real kingdom then it will be captured by the current player, and whoever owned that kingdom before is disqualified. Moreover, all of their previously conquered kingdoms are now under the control of the current player. This is displayed on the grid by colors changing to represent the winning player. If the selected kingdom is a fake kingdom, the player loses their turn, and we move on to the next player.
6. The game will end when one player is left in control of the whole map, after guessing everyone else's kingdom and eliminating them all.

Takeaway: The setup phase was the most important to debug and polish as much as possible as it decided every element in the gameplay, due to many web browsers disabling pop-ups, we elected to include a message in the input phase to temporarily stop screen sharing as it is important that all the names and their assignments were kept secret. There is also an emphasis on the importance of the moderator's role since they are the one in control of the whole game.

The main challenge faced here was the mechanism in which our team decided to create the code, we decided in the end that the best course of action was to meet together and write the code in a joint effort to avoid mismatch.

Game Rules and Mechanics:

1. **Game Start:** Once the game starts, the first round begins with the first player.
2. **Round Mechanics:**
   - The moderator keeps a list of the player names and their kingdom names.
   - Each round lasts 15 seconds.
   - During a player's round, the moderator must click on a kingdom to capture it.
   - A player cannot tell the moderator to pick their original kingdom, if they do, they are disqualified.
   - If the chosen kingdom is real, the player captures it and its representation on the grid is colored with the player's unique color.
   - If the chosen kingdom is fake, a message is displayed, and the player's round ends.
   - If a player captures a kingdom that already belongs to another player, the ownership of all kingdoms of the previous owner is transferred to the capturing player.
3. **Turn Sequence:**
   - After a kingdom is selected, the next player's round begins. The game highlights the active player.
   - After the 15-second round for one player ends, the next player's round begins. The game highlights the active player.

4. **Victory Condition:** The game is won by a player when they manage to capture all the kingdoms. A victory message is displayed, indicating the winner.
5. **Player Removal:** If a player's main kingdom is captured, they are removed from the game.

Takeaway: The gameplay needed to be flawless with as little bugs in the code as possible, to ensure a smooth gameplay experience and a fair competitive gameplay. The main issue we faced at this stage was the overall transition of kingdom takeover, since there are a couple of conditions to walk through before the grid takes any action. This part, as expected, took most of our time to build and tune until we reach a satisfactory gameplay experience. It has been tested on Chrome, Firefox, and Edge with no error messages, this however might be different for other users who use restricted access.

User Interface (UI) Elements:

Figure 1. shows a sample of the main game UI.

1. **Player Section:** Displays the list of players with their respective colors.
2. **Kingdoms Section:** Contains buttons representing each kingdom. The moderator clicks on these buttons to attempt to capture kingdoms.
3. **Grid Overlay:** A visual representation of kingdom ownership. Cells on the grid are colored based on the player who owns the kingdom.
4. **Round Indicator:** Displays the current round number and the time remaining for the current player's round.

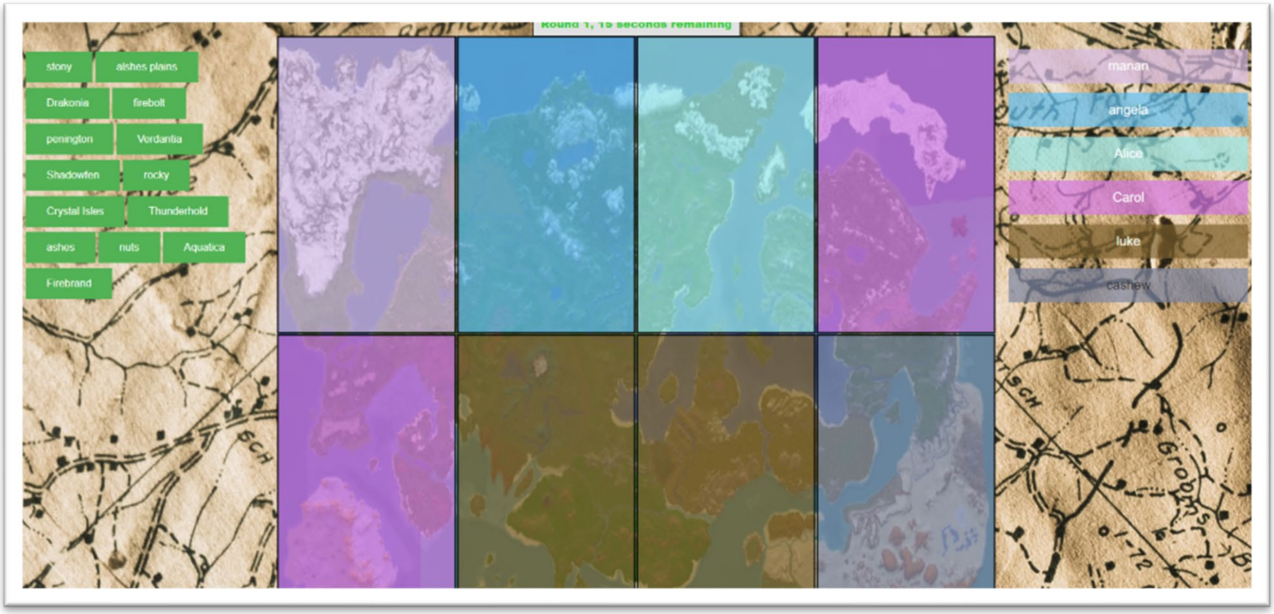

Figure 1 Gameplay UI

Technical Details:

- The game is primarily written in JavaScript, leveraging the DOM for interactivity. Although we were generally more experience with Python, JavaScript was used more in for this purpose, and it was more logical to use it and take advantage of the community help forums for this language.
- It uses local storage to retrieve player names and kingdom names.
- The game employs event listeners to handle user interactions like button clicks.
- The game ensures each player has a unique color through the **getRandomColor()** function.

Final Conclusion

Kingdoms is a fun and interactive game that challenges players to strategize and make decisions quickly. The combination of guessing, randomness, and real-time decision making makes it engaging and fun to play especially with group of close friends.

Creating this game was both intriguing and demanding, especially given the need for seamless page transitions and adept management of storage functions. Interestingly, teamwork presented some challenges. Initially dividing tasks led to misunderstandings among us. We discovered that collaborating in real-time was more efficient and timesaving compared to divvying up tasks, a strategy that doesn't always work in team settings. In essence, this endeavor was enriching for the entire team, as we leveraged our varied backgrounds and continuously learned from one another.