Exercise Wednesday

Intermediate JavaScript (/intermediate-javascript) / Object-Oriented JavaScript (/intermediate-javascript/object-oriented-javascript) / Game of Choice (Two-Day Project)

Text

Goal: The goal of this multi-day project is to exercise your JavaScript programming skills with constructors, prototypes, objects, properties, methods, and DOM manipulation and traversal.

Warm Up

First, carefully read each project description below, and decide which project to tackle with your partner. Then, work through the following questions as part of a pre-coding brainstorming session. Map out a basic approach for the project and get creative!

- What will happen when a user clicks "Play"?
- What objects will you need?
- What key-value pairs will each object need to contain in order for the game to work?
- What functions?
- How will these be triggered throughout the gameplay?
- How will information be collected from the user? How will it be displayed?

- When you have a broad overview of how you want to build your game, identify the simplest behavior, and perhaps what behaviors to tackle after that. Remember to start simple and work one step at a time.

Code

Pick just one prompt below to complete, and then start by completing the warm up.

Pig Dice (Recommended)

Write a program where two users can play Pig dice (https://en.wikipedia.org/wiki/Pig_%28dice_game%29) against each other. Start with your business logic, and once it is completed move onto your user interface logic.

Use test-driven development to create your business logic. Include pseudocode tests in your README. After every passing test, make sure to commit your code.

Make sure that your user interface and business logics are clearly separated, and practice separation of concerns when designing your user interface function.

Further Exploration

- Add options to play one of the other variations of Pig Dice using two or more dice
- Add option to play the computer — easy or hard levels:
  - **Easy**: Computer always stops after the second roll.
  - **Hard**: Computer uses strategy based on current total and rolled dice.

Tic Tac Toe (Difficult)

Note that this project is extra challenging; only begin with this if both you and your partner are fully on board. Or, consider tackling Tic Tac Toe if you finish Pig Dice with time to spare.

Create a Tic Tac Toe (https://en.wikipedia.org/wiki/Tic-tac-toe) game for two players. Start with your business logic, and once it is completed move onto your user interface logic.

Use test-driven development to create your business logic. Include pseudocode tests in your README. After every passing test, make sure to commit your code.

Make sure that your user interface and business logics are clearly separated, and practice separation of concerns when designing your user interface function.

Hints

Consider making four constructors/prototypes: Player, Space, Board, and Game. The objects created from these could include some of the following features:

- A player should know whether it's an X or an O and be able to report that (e.g. player.mark() could return "X" or "O").
- A space should know its coordinates and be able to be marked by a player (e.g., space.mark(player_X)).
- A space should be able to report who it's marked by (e.g. space.marked*by() could return "X" or "O", or it could return a player object — _player1* or *player2*).
- A board should create 9 spaces with the proper coordinates, and tell if there are three in a row marked by the same player. A board should be able to return a space by its coordinates (e.g., board.find(1, 2)).
- A game should create 2 players and a board, be able to move to the next turn, know which player's turn it is, and be able to tell if the game is over or not.

Example

```
let testPlayer = new Player("X");
testPlayer.mark(); // returns "X"

let board = new Board();
let testSpace = board.find(1, 2); // board.find(1,2) return
s a Space object

testSpace.xCoordinate(); // returns 1
testSpace.yCoordinate(); // returns 2

testSpace.mark(testPlayer);
testSpace.markedBy(); // returns testPlayer or "X"

board.gameOver(); // returns a boolean
```

Further Exploration

Give users the option to play the computer at one of two levels: easy or hard.

- **Easy**: Computer randomly selects squares.
- **Hard**: Computer uses strategy (see strategy section in Tic Tac Toe (https://en.wikipedia.org/wiki/Tic-tac-toe)).

Peer Code Review

- Does the application function as expected?
- Is the code well-refactored and easy to follow?
- Do objects drive business logic?
- Are constructors and prototypes used successfully.
- Are pseudocode tests present in the README?

Previous (/intermediate-javascript/object-oriented-javascript/optional-accessing-stylesheets-in-the-cssom)

Next (/intermediate-javascript/object-oriented-javascript/introduction-to-whiteboarding)

Lesson 26 of 33
Last updated March 23, 2023

disable dark mode

Epicodus (http://www.epicodus.com)