Connect 4 - JS Game

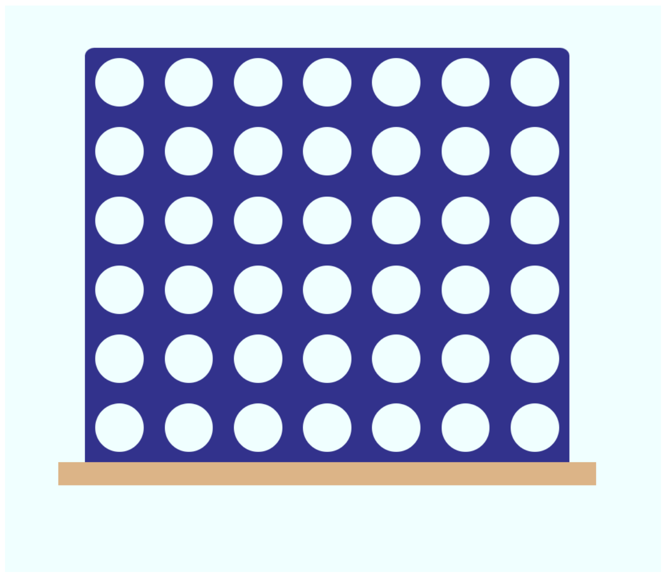
In this project you will implement the famous Connect 4 game.

Overview

Connect Four is a two-player board game where the objective is to be the first to align four of one's own colored discs horizontally, vertically, or diagonally on a grid. Players take turns dropping colored discs from the top of the grid, and the discs fall down occupying the lowest available space within a chosen column. The game requires strategic thinking to block opponents' moves while setting up opportunities to create a winning sequence. The player who successfully connects four discs in a row wins the game.

Task 1: Build the UI

In this task you are required to build the User Interface (UI) of the game. You are to define the base structure (html) and styles (css) of the game. The goal is to maintain an organized layout and css classes.



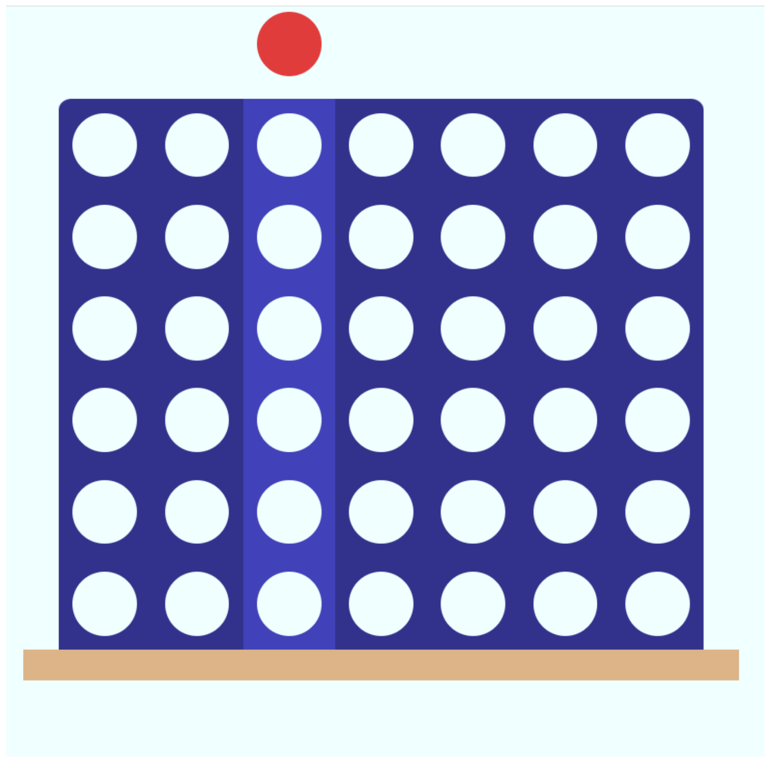
Implementation Details:

- The whole blue **grid** should be a single div with a set width
    - *Hint: come up with a fixed width for a single column, and use it to compute the width of the grid (you can use a css variable and the calc css function to achieve an elegant solution)*
- In the html, define each **row** as a div that has 7 div **cols** inside (replicate it 6 times).
- Define a single **col** to have a nested div for the “circular whole”.
- Use flex-box to achieve control of the layout.
- Create a brownish base to make the board seem to be standing on it.

Task 2: Mouse events

In this task you will add mouse events so that the game feels interactive. When the user’s mouse is over within the x boundaries of the grid, display the current player’s disc red / yellow) over the correct column and highlight the full column. When the user’s mouse is outside the x boundaries of the grid, hide the player’s disc and remove the highlight from the col. When the user clicks, switch the color of the player disc.

This task does not include the dropping of a player disc.

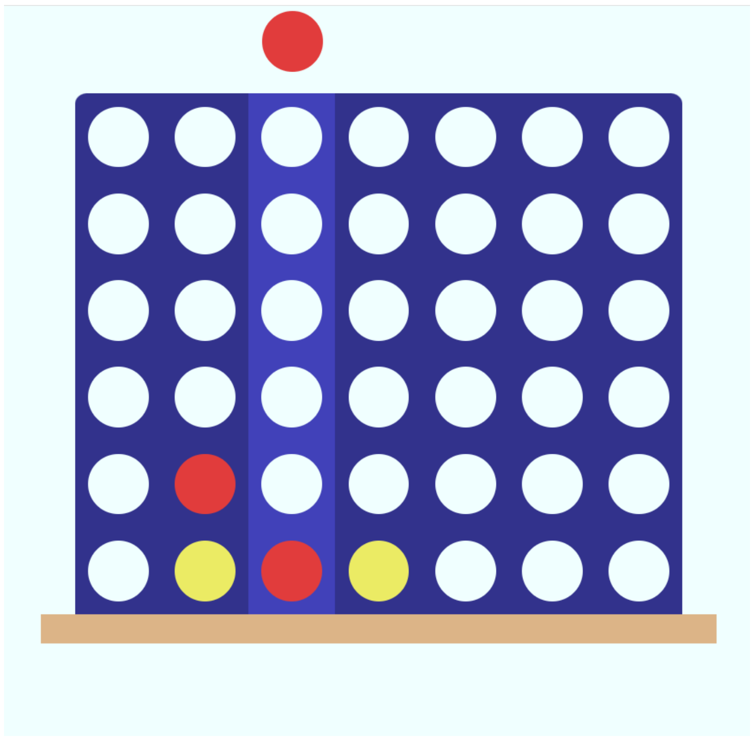


Implementation Details:

- Create script.js and add it to your html file.
- Initialize a 2D array to keep track of the grid's data.
- Define variables to store valuable dom nodes. Some examples are:
    - Grid node: so that you can get the left offset and the full width
    - Cell nodes: so that you can update empty circles
    - Col nodes: to update the blue color to a light blue when the mouse is over the col
- Determine which javascript mouse event(s) can help achieve the task requirements.

Task 3: Game Logic

In this task, you will implement the logic of dropping player discs and computing the grid's state (ongoing, player win, tie).



Implementation Details:

- Keep track of player turns.
- Make sure to define helper functions that are small and concise and avoid large functions that do too much. Have separate functions for the win checks (horizontal, vertical, diagonal).
- Once the game is over, disable dropping any more discs.
- Make sure to synchronize correctly the discs on the UI and the 2D data array

Task 4: Game Controls

In this task you will add UI elements around the main grid to help keep track of games scores, feed messages when the game is over, and display buttons such as new game and reset game.

Implementation Details:

- Display the over score tally of the games player, clearly stating how many wins the red and yellow players have
- Display New Game and Reset buttons (figure out when they each should be disabled/enabled)
- When the game is over display on the screen (in a nice manner) the result of the game. Game over scenarios:
    - Red Wins
    - Yellow Wins
    - Tie

Task 5 (Bonus): Computer Player

In this task, add a button for “Computer Mode” where the yellow player is played by the computer. Add logic to how the computer can decide where to place a token.