



## Front End Technologies Week 11 Coding Assignment

**Points possible:** 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

**Instructions:** In VS Code, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your JavaScript project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

### Coding Steps:

1. Using any of the tools you've worked with so far, create a game of tic-tac-toe.
  - a. A heading should say whether it is X's or O's turn and change with each move made.
  - b. Create a tic-tac-toe grid using your HTML element of choice. When a cell in the grid is clicked, an X or O should appear in that spot depending on whose turn it is.
  - c. A button should be available to clear the grid and restart the game.
  - d. When a player has won, or the board is full and the game results in a draw, a Bootstrap alert or similar Bootstrap component should appear across the screen announcing the winner.

### Screenshots of Code:



HTML

# PROMINEO TECH



# PROMINEO TECH

file Edit Selection View Go Run Terminal Help

EXPLORER

WEEK 1...  
  .gitignore  
  package.json  
  TicTacToe.css  
  TicTacToe.html  
  TicTacToe.js

TicTacToe.html U X

JS TicTacToe.js U

# TicTacToe.css

TicTacToe.html > html > body.bg-info > div.container

```
1  <!DOCTYPE html>
2  <html lang="en">
3  > <head>...
10 </head>
11 <body class="bg-info">
12   <div class="container" >
13     <!-- Heading on page that says who's
14     <h1 id="turnNotifier">Now Playing: <
15     <div class="row " style="height: 70p
16     <div id="winOrDraw" class="alert
17   </div>
18   <!-- I need to set up the grid,
19   <div id="gameBoard">
20     <div class="cell" id="0"></div>
21     <div class="cell" id="1"></div>
22     <div class="cell" id="2"></div>
23     <div class="cell" id="3"></div>
24     <div class="cell" id="4"></div>
25     <div class="cell" id="5"></div>
26     <div class="cell" id="6"></div>
27     <div class="cell" id="7"></div>
28     <div class="cell" id="8"></div>
29   </div>
30   <br>
31   <br>
32   <br>
33   <button class="button" id="resetButt
34 </div>
35 <script src="https://cdn.jsdelivr.net/npm
36 <script src="TicTacToe.js"></script>
37 </body>
38 </html>
```



# PROMINEO TECH

**Java**



# PROMINEO TECH

File Edit Selection View Go Run Terminal Help TicT

EXPLORER ...  
✓ WEEK 11 CODING AAIGN...  
 .gitignore U  
 {} package.json U  
 # TicTacToe.css U  
 <> TicTacToe.html U  
 JS TicTacToe.js U

<> TicTacToe.html U JS TicTacToe.js U X # TicTacToe.css

JS TicTacToe.js > [🔍] cellClicked  
1 const cellSet = Array.from(document.getEleme  
2 //set up and assign some variables and ids  
3 let winOrDraw = document.getElementById("win  
4 let resetButton = document.getElementById("r  
5 let turnNotifier = document.getElementById(''  
6  
7 const xPlayer = "X";  
8 const oPlayer = "O";  
9 let turnMarker;// = xPlayer;  
10  
11 //set all squares to blank (null) at start  
12 const squares = []  
13 // Array(cellSet.length);  
14 // squares.fill(null);  
15  
16 const drawBoard = () => {  
17 cellSet.forEach ((cell, index) => {  
18 let styleString = '';  
19 if (index < 3) {  
20 | styleString += 'border-bottom: 3px  
21 }  
22 if (index % 3 === 0) {  
23 | styleString += 'border-right: 3px  
24 }  
25 if (index % 3 ===2) {  
26 | styleString += 'border-left: 3px  
27 }  
28 if (index > 5) {  
29 | styleString += 'border-top: 3px  
30 }  
31 cell.style = styleString;  
32 winOrDraw.innerText = ""  
33 cell.addEventListener('click', cellC  
34 |



# PROMINEO TECH

<> TicTacToe.html U

JS TicTacToe.js U X

# TicTacToe.css U

JS TicTacToe.js > [?] drawBoard > [?] cellSet.forEach() callback

```
34     });
35   };
36   // empty cells have onclick that will enter X or O based on the t
37   const cellClicked = (position) => {
38     console.log("cell was clicked");
39     let id = position.target.id;
40     //check if a square is already empty, and if no assign the cu
41     if(!squares[id]) {
42       squares[id] = turnMarker;
43       position.target.innerText = turnMarker;
44
45       //check for a win before switching players.
46       //Only need to check current player, because
47       //other player won't win on current player's turn
48       if (winningPlay()) {
49         //turnNotifier.innerText = `${turnMarker} wins!`;
50         winOrDraw.innerText = ` Player ${turnMarker} Wins!`;
51         winOrDraw.style.opacity = "100";
52         return;
53       }
54       //checks for a win, and returns true if so. Even if board
55       if (fullBoardDraw()) {
56         winOrDraw.innerText = `It's a DRAW! No Winner!`;
57         winOrDraw.style.opacity = "100";
58         return;
59       }
60       if (turnMarker === xPlayer) {
61         turnMarker = oPlayer;
62       } else if (turnMarker === oPlayer) {
63         turnMarker = xPlayer;
64       }
65       turnNotifier.innerText = `Now playing: ${turnMarker}`;
66     }
67   }
```



# PROMINEO TECH

<> TicTacToe.html U

JS TicTacToe.js U X

# TicTacToe.css U

JS TicTacToe.js > [e] cellClicked

```
68
69   const winningPlay = () => {
70       if (squares[0]=== turnMarker){
71           if (squares[1] === turnMarker && squares[2] === turnMarker){
72               console.log(`${turnMarker} wins!`);
73               return true;
74           }
75       }
76       if (squares[0]=== turnMarker){
77           if (squares[4] === turnMarker && squares[8] === turnMarker){
78               console.log(`${turnMarker} wins!`);
79               return true;
80           }
81       }
82       if (squares[0]=== turnMarker){
83           if (squares[3] === turnMarker && squares[6] === turnMarker){
84               console.log(`${turnMarker} wins!`);
85               return true;
86           }
87       }
88       if (squares[1]=== turnMarker){
89           if (squares[4] === turnMarker && squares[7] === turnMarker){
90               console.log(`${turnMarker} wins!`);
91               return true;
92           }
93       }
94       if (squares[2]=== turnMarker){
95           if (squares[5] === turnMarker && squares[8] === turnMarker){
96               console.log(`${turnMarker} wins!`);
97               return true;
98           }
99       }
100      if (squares[2]=== turnMarker){
101          if (squares[4] === turnMarker && squares[6] === turnMarker){
```



# PROMINEO TECH

<> TicTacToe.html U

JS TicTacToe.js U X

# TicTacToe.css U

JS TicTacToe.js > [x] winningPlay

```
100  ✓ 11 (squares[2]=== turnMarker){
101  ✓   if (squares[4] === turnMarker && squares[6] === turnMarker
102  ✓   |   console.log(`${turnMarker} wins!`);
103  ✓   |   return true;
104  ✓   |   }
105  ✓   }
106  ✓  if (squares[3]=== turnMarker){
107  ✓   if (squares[4] === turnMarker && squares[5] === turnMarker
108  ✓   |   console.log(`${turnMarker} wins!`);
109  ✓   |   return true;
110  ✓   |   }
111  ✓   }
112  ✓  if (squares[6]=== turnMarker){
113  ✓   if (squares[7] === turnMarker && squares[8] === turnMarker
114  ✓   |   console.log(`${turnMarker} wins!`);
115  ✓   |   return true;
116  ✓   |   }
117  ✓   }
118  }
119  ✓ //the draw was difficult. I couldn't figure out an easy way to ite
120  //or causing a weird issue where it would work fine after the fir
121  //but it would throw up the draw alert on the first click even whe
122  ✓ const fullBoardDraw = () => {
123  if (
124  (squares[0] === xPlayer || squares[0] === oPlayer) &&
125  (squares[1] === xPlayer || squares[1] === oPlayer) &&
126  (squares[2] === xPlayer || squares[2] === oPlayer) &&
127  (squares[3] === xPlayer || squares[3] === oPlayer) &&
128  (squares[4] === xPlayer || squares[4] === oPlayer) &&
129  (squares[5] === xPlayer || squares[5] === oPlayer) &&
130  (squares[6] === xPlayer || squares[6] === oPlayer) &&
131  (squares[7] === xPlayer || squares[7] === oPlayer) &&
132  ✓ (squares[8] === xPlayer || squares[8] === oPlayer) ){
133  return true;
```





# PROMINEO TECH

<> TicTacToe.html U

JS TicTacToe.js U X

# TicTacToe.css U

JS TicTacToe.js > ...

```
135     }
136
137     //if board fills, or player wins (3 in a row, orthogonal or diagonal)
138     //winningRows in the table/array match one of these 8 options, count as win
139     // const winningRows= [
140     // [0, 1, 2],
141     // [0, 4, 8],
142     // [0, 3, 6],
143     // [1, 4, 7],
144     // [2, 5, 8],
145     // [2, 4, 6],
146     // [3, 4, 5],
147     // [6, 7, 8]
148     // ]
149
150     //reset button that starts a new game
151     const newGame = () => {
152         squares.forEach((square, index) => {
153             squares[index] = null;
154         });
155         cellSet.forEach((cell) => {
156             cell.innerText = "";
157         });
158         turnMarker = xPlayer;
159         winOrDraw.style.opacity = "0";
160         winOrDraw.innerText = "";
161         turnNotifier.innerText = `Now playing: ${turnMarker}`;
162     }
163     resetButton.addEventListener('click', newGame);
164
165     //make sure to call the functions after you make it.
166     newGame();
167     drawBoard();
```



# PROMINEO TECH

**CSS stylesheet (not much, was running late)**



# PROMINEO TECH

```
<> TicTacToe.html U    JS TicTacToe.js U    # TicTacToe.css U X
# TicTacToe.css > .cell

1  √ :root {
2    |    --color: #be5a08;
3    |  }
4
5  √ body {
6    |    margin: 0;
7    |    padding: 0;
8    |    color: var(--color);
9    |    font-family: Arial, Helvetica, sans-serif;
10   |  }
11
12  √ * {
13    |    box-sizing: border-box;
14    |  }
15
16  √ h1 {
17    |    text-transform: uppercase;
18    |    font-size: 30px;
19    |  }
20
21  √ .cell {
22    |    height: 120px;
23    |    width: 120px;
24    |    display: flex;
25    |    justify-content: center;
26    |    align-items: center;
27    |    color: var(--color);
28    |    font-size: 80px;
29
30  |  }
31
32  √ #gameBoard {
33    |    width: 360px;
34    |    display: flex;
```



# PROMINEO TECH

```
<> TicTacToe.html U JS TicTacToe.js U # TicTacToe.cs
# TicTacToe.css > .cell
25 justify-content: center;
26 align-items: center;
27 color: var(--color);
28 font-size: 80px;
29
30 }
31
32 #gameBoard {
33     width: 360px;
34     display: flex;
35     flex-wrap: wrap;
36
37 }
38
39 .container {
40     justify-content: center;
41     align-items: center;
42     flex-direction: column;
43     display: flex;
44     height: 100vh;
45     padding: 30px;
46
47 }
48
49 button:hover {
50     cursor: pointer;
51 }
```



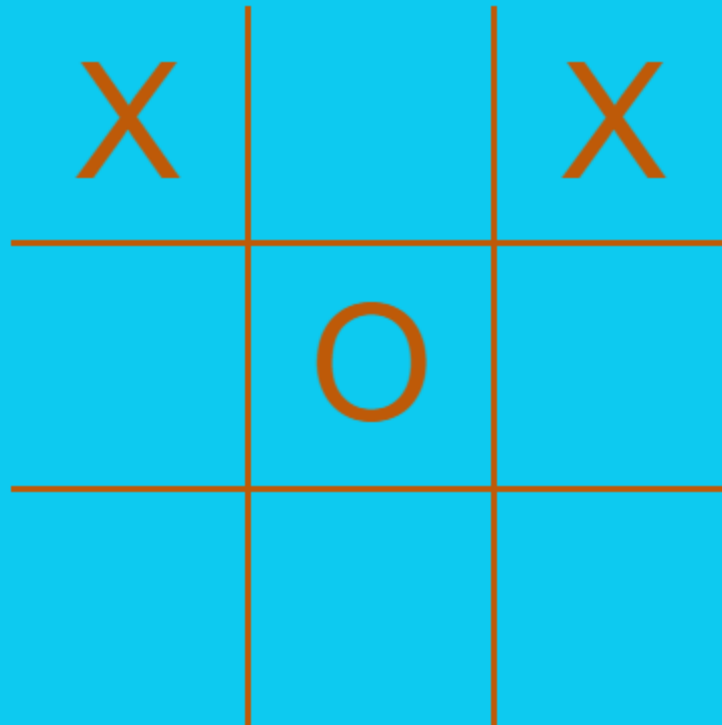
# PROMINEO TECH

**Screenshots of Running Application:**



# PROMINEO TECH

NOW PLAYING: O



Reset Board



# PROMINEO TECH

NOW PLAYING: O

Player O Wins!

X	O	X
X	O	
	O	

Reset Board



# PROMINEO TECH

NOW PLAYING: X

Player X Wins!

X	O	X
X	O	O
X	X	O

Reset Board





# PROMINEO TECH

NOW PLAYING: X

It's a DRAW! No Winner!

X	O	X
X	O	O
O	X	X

Reset Board

URL to GitHub Repository:

<https://github.com/KalonOhm/TicTacToe>