PROJECT 1:
CONNECT 4

DEISE KOREEDA

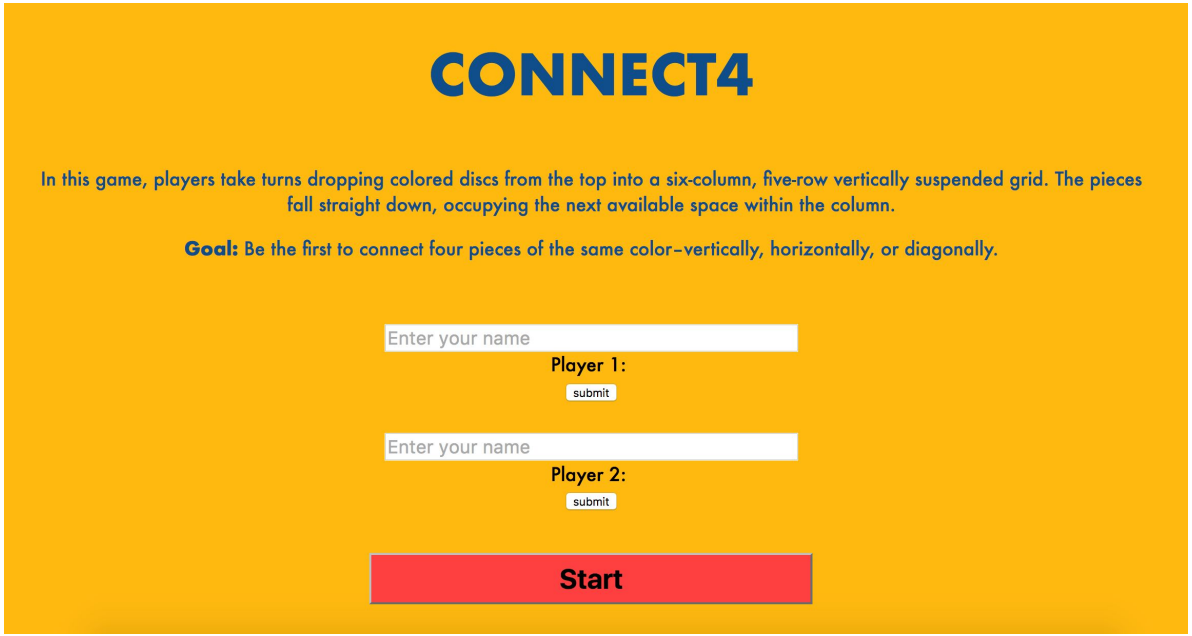
Connect 4: What is it?

- **2 players** have different colored discs: **red or yellow**
- Blue **suspended grid**: **6x5 discs** (width x height)
- The **suspended grid has gaps at the top** through which the **player drops the disc** and **it falls straight down** as far as it can go, **occupying the next available space** within that column
- The player has to use his/her own **discs to block** his/her **opponent**

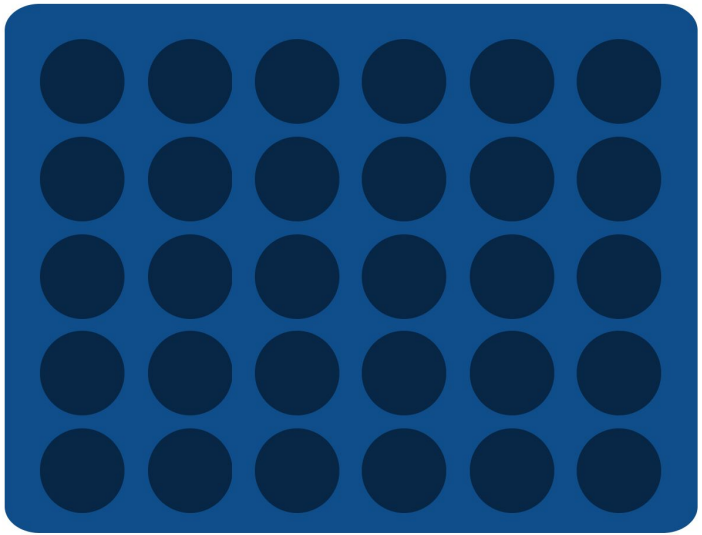
Goal

Be the first to connect 4 pieces in a row, column or diagonally

Landing Page



Game page



Player 1

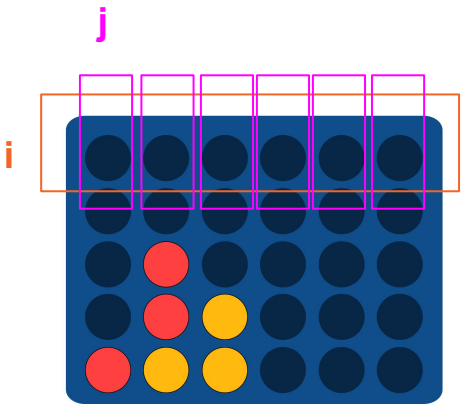
Reset

Foreseen challenges or obstacles:

1. Create **array (row x column)** that will **save the position** of discs
2. Apply **CSS and Javascript to slide down pieces “inside the board”** - from the top of it to an available spot in a particular column in the array **considering distances between discs**
3. Write the **winner logic** with all possibilities considering **vertical, horizontal and diagonal combinations**
4. **Media query** for tablets, cellphones and laptops/desktops

How did I tackle those obstacles?

1. Create **array (row x column)** that will **save the position** of discs

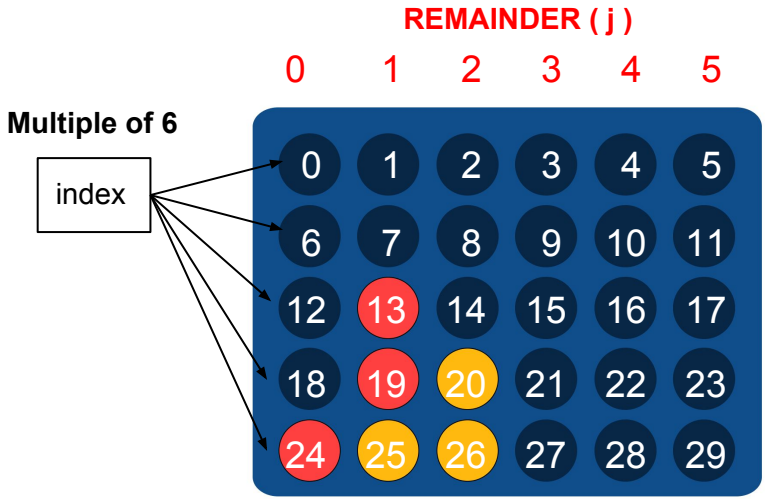



```
//creating board spots where index i represents rows and j represents columns
for (var i = 0; i < row; i++) {
    boardCircles[i] = [];
   for (var j = 0; j < column; j++) {
        boardCircles[i][j] = $('<div>');
        boardCircles[i][j].addClass('circle');
        // console.log('boardCircles -->', boardCircles[i][j]);
        $('.container').append(boardCircles[i][j]);
   }
}
```

How did I tackle those obstacles?

```
$('.circle').click(function() {
    console.log('hi');
    var position = $(this).index();
    console.log('position clicked', position);
    var counter = 0;
    var startPosition = 0;
    var endPosition = 0;

        if(position % 6 === 0) {
            console.log('if statement running');
            var j = 0;
            startPosition = 25;
            endPosition = 25;
            var d = 92;
            // console.log(left);
            if (player === 1) {
                fillAvailableSlots(j, "yellow");
                move(j, startPosition, endPosition, "slide-yellow");
                player += 1;
                currentPlayer = $('#current-player').text("Player 2");
                checkForWinner(boardCircles);
                checkTiedGame(boardCircles);
            } else {
                fillAvailableSlots(j, "red");
                move(j, startPosition, endPosition, "slide-red");
                player = 1;
                currentPlayer = $('#current-player').text("Player 1");
                checkForWinner(boardCircles);
                checkTiedGame(boardCircles);
            }
        }
```

2. Apply **CSS and Javascript to slide down pieces inside the board** - from the top of it to an available spot in a particular column in the array **regarding distances between discs.** *How to select a particular column?*



How did I tackle those obstacles?

2. Apply **CSS and Javascript to slide down pieces inside the board** - from the top of it to an available spot in a particular column in the array **regarding distances between discs**

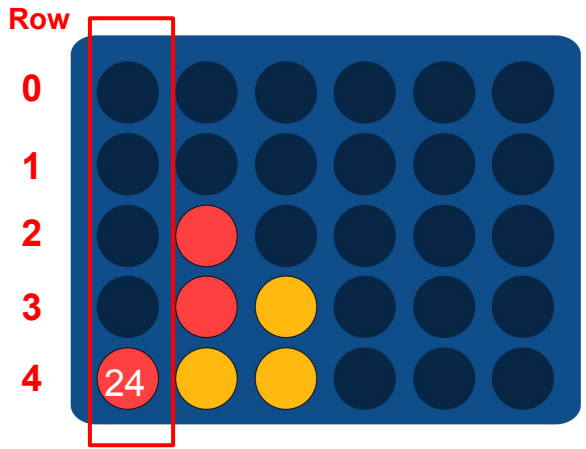
```
function fillAvailableSlots(column, color, numOfSlots) {
    var numOfSlots = 0;
    for (var i = 0; i < row; i++) {
      console.log('varredura');
        if(boardCircles[i][column].hasClass('circle')){
          numOfSlots += 1;
        }
    }
    console.log('number of available slots', numOfSlots);
    if (numOfSlots !== 0) {
      boardCircles[numOfSlots-1][column].removeClass('circle');
      boardCircles[numOfSlots-1][column].addClass(color);
    }
    // else {
    //   boardCircles[numOfSlots-1][column].removeClass('circle');
    //   boardCircles[numOfSlots][column].addClass(color);
    // }
    setTimeout(fillAvailableSlots, 2000);
}
```

Index = 24

Remainder = 0

Column = 0

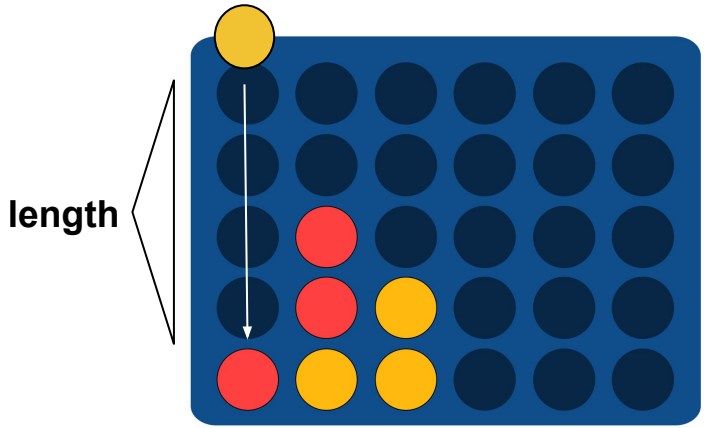
Scan rows within the column 0

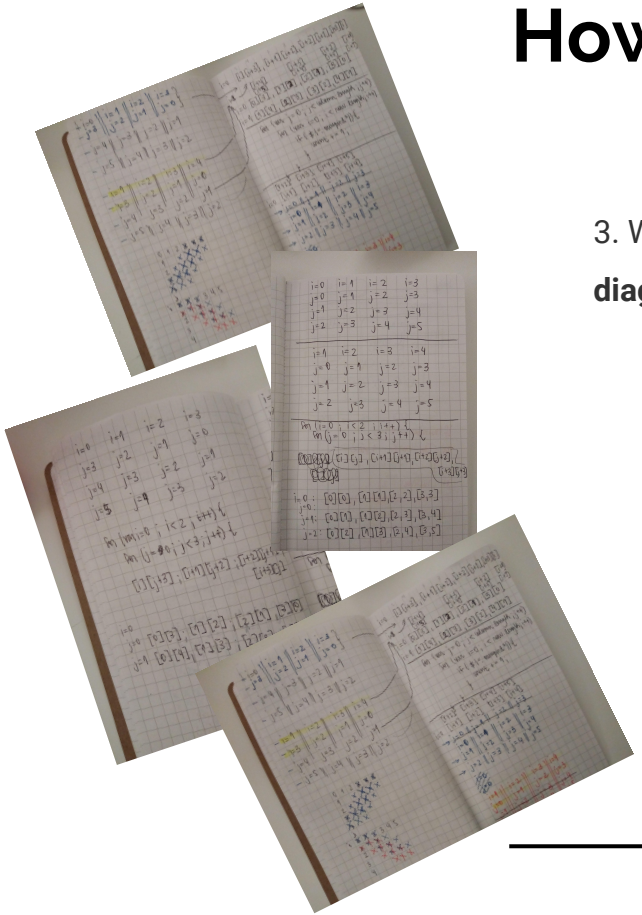


How did I tackle those obstacles?

```
//This function creates a div and slide it down on the board
function move(column, leftStart, leftEnd, color)  {

    //total height of container = 350
    var totalHeight = 370;
    var distanceBtwDivs = 80;
    //create a div
    var slideCircle = $('<div>');
    // slideCircle.addClass('slide-circle');
    //add it a class of color
    slideCircle.addClass(color);
    //set left position to it according to the argument given in the function
    slideCircle.css('left', leftStart);
    //append this div to the container, which represents the board
    $('.container').append(slideCircle);

    //account for number of available slots on the board
    var numOfSlots = 0;
    for (var i = 0; i < row; i++) {
        if(boardCircles[i][column].hasClass('circle')){
          numOfSlots += 1;
        }
    }

    /*if there are 5 available slots then distance to slide piece down on the board
     *will be total height of the container
     */
    if (numOfSlots === 5) {
        slideCircle.animate({top: totalHeight, left: leftEnd}, 200);
        slideCircle.fadeOut(50);
    }
```

2. Apply **CSS and Javascript to slide down pieces inside the board** - from the top of it to an available spot in a particular column in the array **regarding distances between discs**





How did I tackle those obstacles?

3. Write the **winner logic** with all possibilities regarding **vertical, horizontal and diagonal combinations**

```
var checkForWinner = function(circles) {
    // console.log('checkWinner');
    // console.log(circles);

    //scan rows
    for (var j = 0; j < 3; j++) {
      for (var i = 0; i < row; i++) {
        //if statement that checks if yellow has a sequence in any row
        if (circles[i][j].hasClass('yellow') && circles[i][j+1].hasClass('yellow') && circles[i][j+2].hasClass('yellow') && cir
        {
            console.log('checkWinner ---> if statement')
            setTimeout(function(){
              alert ('Yellow Wins!!!');
            }, 250);
        }
        //if statement that checks if red has a sequence in any row
        if (circles[i][j].hasClass('red') && circles[i][j+1].hasClass('red') && circles[i][j+2].hasClass('red') && circles[i][j
        {
            console.log('checkWinner ---> if statement')
            setTimeout(function(){
              alert ('Red Wins!!!');
            }, 250);
        }
      }
    }
```

How did I tackle those obstacles?

3. Write the **winner logic** with all possibilities regarding **vertical, horizontal and diagonal combinations discs**

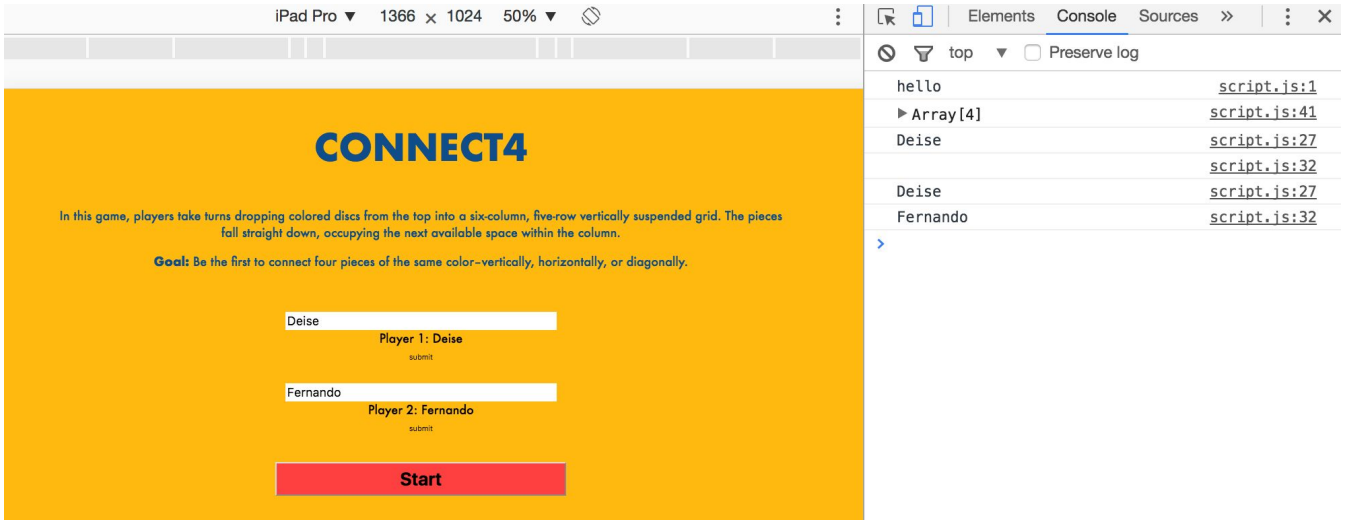
```
var checkTiedGame = function(array) {
    var counter = 0;
    for (var i = 0; i < row; i++) {
        for (var j = 0; j < column; j++) {
            if (array[i][j].hasClass('yellow') || array[i][j].hasClass('red')) {
                counter += 1;
            }
        }
    }
    if (counter === 30) {
      console.log('checkTiedGame---> if statement')
      setTimeout(function(){
        alert ('Try again!!!');
      }, 250);
    }
}
```

How did I tackle those obstacles?

4. Media query for tablets, cellphones and laptops/desktops

Display player's name

```
23  var players = [];
24
25  $(".submit-button").click(function() {
26    var inputName1 = $("#player1").val();
27    console.log(inputName1);
28    players.push(inputName1);
29    $('#yellow').html(inputName1);
30
31    var inputName2 = $("#player2").val();
32    console.log(inputName2);
33    players.push(inputName2);
34    $('#red').html(inputName2);
35
36      $('input[name=player]').each(function() {
37        players.push($(this).val());
38      });
39  });
40
41  console.log(players);
42
```



Thank you!