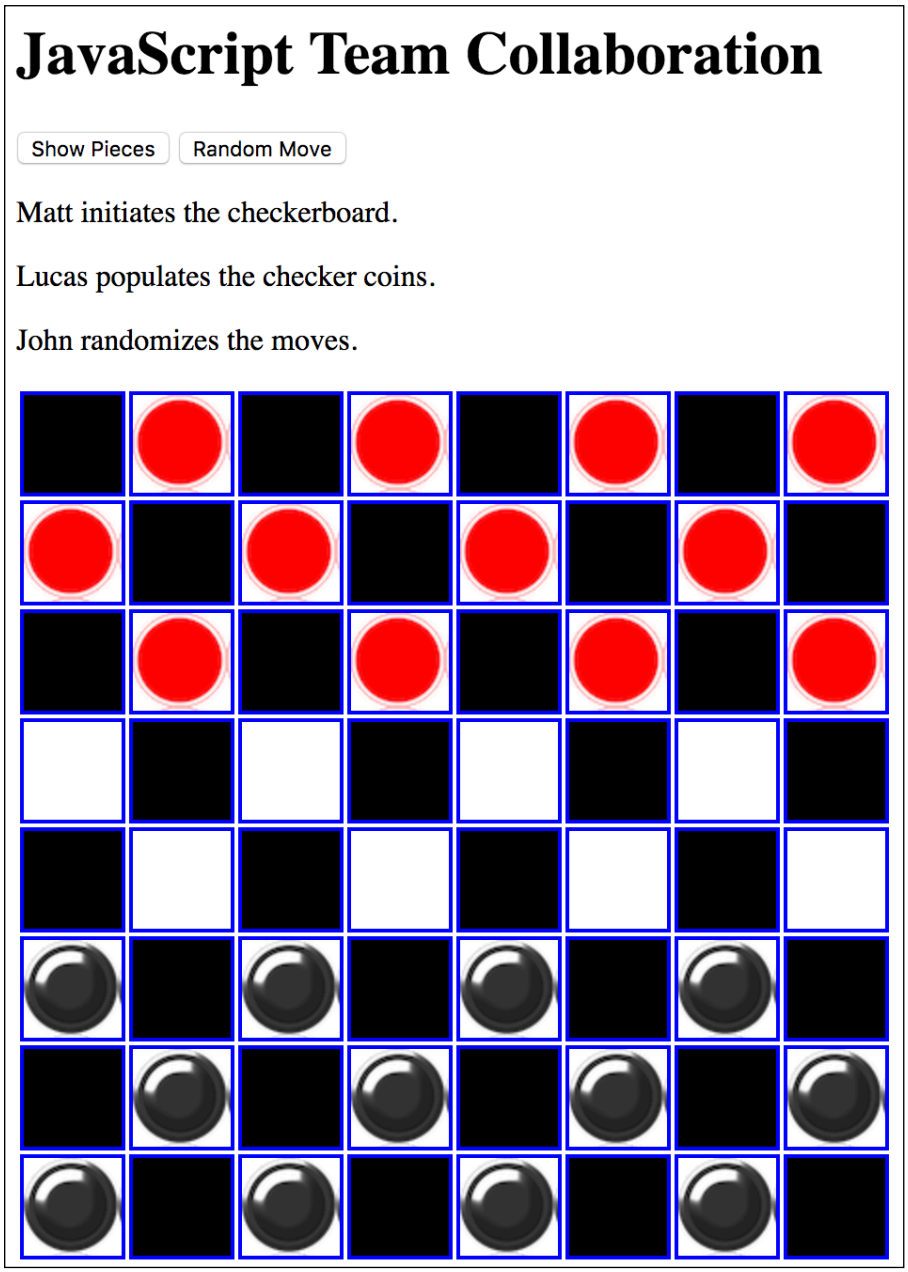
INFO 151: Assignment 4 (15 points)

The objectives of this assignment include:
+ Practicing on the use of **loops, arrays** and **functions** in Javascript programing;
+ Applying **methods and properties** in the **DOM** for dynamic content manipulation;
+ Develop **team collaboration experience** for system development.

The final results of the assignment is showing a checker board with pieces:



Step 0: Team Lead and Preparation

- Appoint one member of your team as the **lead** (member 1) who lays the foundation for the assignment;
- The whole team (member 2 and 3) should assist the lead in the **steps 1 – 4**;
- Step 5 (and step 6) are then to be finished by **team member 2** (and 3 if applicable)

Step 1: Directories/Folders and Files

Please **create a directory a3/** and **the following files**:

- **game.html** as the main page for this assignment
- **css/** directory with **style.css in it** for CSS rules
- **js/** directory with **action.js** in it for Javascript code
- **images/** directory

Step 2: HTML Code (2 point)

Create the **game.html** page and make sure it fully validated after done. Include the following in the HTML:

1. Link to the CSS file at **css/style.css**, similar to:

```
<link rel="stylesheet" type="text/css" href="css/main.css" media="all" />
```

2. Link to the JavaScript file at js/action.js:

```
<script type="text/javascript" src="js/action.js"></script>
```

3. Register a function call to **drawGameBoard(8, 8)** to the body's **onload** event:

```
<body onload="drawGameBoard(8,8);">
```

4. Include the following elements in the body section:

```
<h1>JavaScript Team Collaboration</h1>

<p><span id="member1">Who</span> initiates the checkerboard. </p>
<p><span id="member2">Who</span> populates the checker coins. </p>
<p><span id="member3">who</span> randomizes the moves. </p>

<div id="board">
</div>
```

Step 3: CSS Code (3 points)

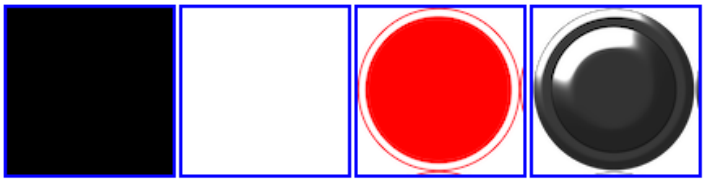
Add CSS selectors and rules to **css/style.css** and make sure all **CSS code is validated** in the end. Include the following CSS selector and rules:

1. Create a **td *type*** selector and make it **100px width**, **100px height**, and with **visible borders**, among others.
2. Create a **black *class*** selector with a *darker* background color;
3. Create a **white *class*** selector with a *lighter* background color;
4. Create a **redpiece *class*** selector with background-image with url("../images/redpiece.png");
5. Create a **blackpiece *class*** selector with background-image url("../images/redpiece.png");

Test the CSS *code by adding the following HTML code to* ***game.html****:*

```
<table>
    <tr>
        <td class="black"></td>
        <td class="white"></td>
        <td class="redpiece"></td>
        <td class="blackpiece"></td>
    </tr>
</table>
```

which should show something similar here:



Make sure you see this before proceeding to Javascript.

Step 4: Team Member #1: Draw the Game Board/Table (5 points)

Add a function to **js/main.js file**:

1. Create a function called **getCellID()** with three parameters **colsPerRow**, **row**, and **col** in the parentheses. Code for this function with comments is provided here for your reading and use:

```
/**
 * To convert the current row and column of a cell into
 * a unique identifier (ID).
 * For example, on a 8x8 grid/board, the 3rd column (index 2)
 * of the 2nd row (index 1) will have the ID of "c10", where:
 * 10 is tge result of 1*8 + 2.
 *
 * @param {int} colsPerRow the number of columns per row;
 * @param {int} r the row index of the current cell/grid;
 * @param {int} c the column index of the current cell/grid;
 * @returns {String} the ID of the cell with prefix "c" such as "c10".
 */
function getCellID(colsPerRow, r, c){
    var id = r*colsPerRow + c;
    return "c" + id;
}
```

2. Create another function called **drawGameBoard()** with two parameters for **rows** and **columns** in the parentheses. This function will draw the checker board/grid. Add statements to do the following in the function block (between curly braces) to:

- Use **document.getElementById("member1")** and assign the name of the team member to it;
- Create a **table** element using **document.createElement("table")** and assign it to a **variable;**
- Reference the div element with id="board" by using **document.getElementById("board")** and assign it to **another variable**;
- Add the **table element** above to the **board div** above using the **appendChild()** method;
- Create a **nested loop** (a "big" loop and a "small" loop inside the big loop):
    - IN the **BIG LOOP**, the number of repetitions is the number of the **rows** in the parameter. Do the following in the loop:
        - Create a **tr** (for a table row) element using the same **document.createElement()** method above;
        - Assign the newly created element to **a variable;**

        - Insert the **tr** element (variable for the row) into the above table using the **appendChild()** method;
    - The SMALL (nested) loop repeats the number of **columns**. In this loop:
        - Create a **td** (for a table data cell) element using the same **document.createElement()** method above;
        - Assign the newly created **td** element to a **variable**;
        - Assign ***"white"*** to the **className** property of the above **td** element/variable;
        - Call **getCellID()** function and pass **8, i, j** (assume i and j are loop indices for row and column) as parameter values to convert the current row and column to an unique ID;
        - Assign the above ID to the **id** property/attribute of the above **td** element/variable;
        - Use **appendChild()** method to add the **td** element/variable into the **tr** element/variable;

You should see an 8x8 table/grid after step 4 is finished.

Step 5: Team Member #2: Show Game Pieces (5 points)

First, **ask team member #1** to add the following in the game.html:

- Additional **script** tag with **src="/~2ndMemberDrexelID/a3/js/action2.js";**
- A new button and event registration:

```
<button onclick="showGamePieces(8,8);">Show Pieces</button>
<button onclick="randomMove();">Random Move</button>
```

For **team member #2**, do the following:

- Create a new **action2.js** under **/a3/js/** subdirectory;
- Add the following array variable with data in your code (**before function**):

```
var pieces = [
    [ 0,-1, 0,-1, 0,-1, 0,-1],
    [-1, 0,-1, 0,-1, 0,-1, 0],
    [ 0,-1, 0,-1, 0,-1, 0,-1],
    [ 0, 0, 0, 0, 0, 0, 0, 0],
    [ 0, 0, 0, 0, 0, 0, 0, 0],
    [ 1, 0, 1, 0, 1, 0, 1, 0],
    [ 0, 1, 0, 1, 0, 1, 0, 1],
    [ 1, 0, 1, 0, 1, 0, 1, 0]
];
```

- Create a function named **showGamePieces()** with **rows** and **columns** parameters. In the function:
    - Use **document.getElementById("member2")** and assign the name of team member #2 to it;
    - Create a nested loop similar to the 2nd part of Step 4. Within the SMALL (nested) loop:
        - Assume i and j are indices for the loops (for row and column), call **getCellID(8, i, j)** to convert row and column to a unique ID;

- Use **document.getElementById()** with the above **ID** to reference the corresponding **td** element and assign it to a **variable**;
- Create a selection structure (if… else…) to assign the proper CSS class name to the **td** element/variable.
    - If **pieces[i][j]** is **lesser than zero**, assign ***"redpiece***" to the **className** property/attribute of the td variable;
    - If **pieces[i][j]** is **greater than zero**, assign ***"blackpiece"*** to the **className** property/attribute of the td variable;

Step 6: Team Member #3: Random Move (5 points)

Complete step 6 **ONLY IF** you have a 3rd member in your team.

Ask team member #1 to add the following in the **game.html**:

- Additional **script** tag with **src="/~3rdMemberDrexelID/a3/js/action3.js";**

For team member #3, create a function **randomMove()** without parameters. In the function:

- Combine **Math.random()** and **Math.floor()** to generate a random integer between 0 and 8.
- Repeat the above step for 4 times to generate four random integers and store them in variables **r1, c1, r2, c2** (We will use the **first two integers** as row and column indices to **move from**, and use the **last two** integers as indices of where to **move to)**.
- Use the first two integers r1 and c1 as indices to look up the value in **pieces[r1][c1]** array;
- Assign the above value to replace **pieces[r2][c2]**;
- Convert r1 and c1 into a unique ID by calling **getCellID(8, r1, c1)**;
- Use **document.getElementById()** with the above ID to reference the corresponding td element and assign its **className** property to "white";
- Convert r2 and c2 into a unique ID by calling **getCellID(8, r2, c2)**;
- Reference the corresponding td element with the above ID and change its **className** according to **"redpiece"** or **"blackpiece",** depending on the value of **pieces[r1][c1].** (This changes **the style of the target cell** to the **style of where the piece moves from**.)
- Assign 0 to **pieces[r1][c1]** at the end.

Scores from ***Step 5 and Step 6 (if there is a 3rd member of the team) will be averaged and then added to the final grade.***

Bonus (+2 points): Show the game board with the black and white **checker patterns**.

In the end, make sure you generate a game board similar to the **figure** and there is **no error** in your code (HTML, CSS, and Javascript).

Submit the URL to your team's **game.html** to Blackboard Learn.