let allPieces = document.querySelectorAll('.pieces')

all pieces have class .pieces even null form 0 to 63

null pieces only have class pieces and nullPiece

let bluePieces = document.querySelectorAll(".bluePiece")

let redPieces = document.querySelectorAll(".redPiece")

**startGameButtonFunc()**

is where we start the game this invokes **addClickeventToPieces()**

after the first run of addClickeventToPieces() this is invoked in **changeWhosTurn()**

**addClickeventToPieces()**

function adds event listener to blue pieces when turn is 0

red pieces when turn is 1

the click runs function **resestWithRemainingPeices**

**resestWithRemainingPeices()**

runs function **pieceYouWantToMove(event)**

also re-runs querySelecter for bluePiece and redPiece class with remaining pieces

bluePieces = document.querySelectorAll(".bluePiece")

redPieces = document.querySelectorAll(".redPiece")

**pieceYouWantToMove**(event)

function get clicked piece and saves the pieces id in variable (activatedPiece)

we give it a class activatedPiece as well

runs **removeEventListenerWhenNotTurn()**

and **checkForOpenSpotsNoJumps()**

**removeEventListenerWhenNotTurn()**

removes the Eventlister "click", resestWithRemainingPeices

Which was initially added in addClickevent function

**checkForOpenSpotsNoJumps()**

function checks for spots we can move dirrecty forward to

and when it finds those pieces gives them class PieceYouCanMoveTo

fills variables (rightForwardOpenPieceUP , leftForwardOpenPieceUp , rightForwardOpenPieceDown

leftForwardOpenPieceDown)

then runs function **checkForJumps()**

**checkForJumps()**

checks if any single jumps are open and gives class if one is found

fills variables (rightForwardUPJump, leftForwardUPJump , rightForwardDownJump

leftForwardDownJump)

runs querySelector on all pieces with class PieceYouCanMoveTo

and gives event listener click with function **movePiece**

**movePiece(event)**

on click sets the pieces id to variable spotWeWantToMoveTO

then checks if spotWeWantToMoveTO id is equal to our pieces with class PieceYouCanMoveTo

(rightForwardOpenPieceUP , leftForwardOpenPieceUp , rightForwardOpenPieceDown

leftForwardOpenPieceDown)

when piece is selected invokes function **resestAfterPieceIsMoved()**

or if no piece is clicked runs function **singleJumpMovePiece()**

**singleJumpMovePiece()**

chekcs if variable spotWeWantToMoveTO is equal to

(rightForwardUPJump, leftForwardUPJump , rightForwardDownJump leftForwardDownJump)

If yes and that piece is clicked gives variable

pieceWeSingleJumped for the piece we are jumping over

and then runs function **resestAfterPieceIsJumpedSingle()**

**resestAfterPieceIsMoved()**

resets varialbes and runs **winGameFunc()**

and switchs classes for SpotWeWantToMoveTO and activatedPiece

rightForwardOpenPieceUP =0

    leftForwardOpenPieceUp =0

    rightForwardOpenPieceDown =0

    leftForwardOpenPieceDown =0

    rightForwardUPJump =0

    leftForwardUPJump =0

    rightForwardDownJump =0

    leftForwardDownJump=0

    spotWeWantToMoveTO =0

**resestAfterPieceIsJumpedSingle()**

changes classes for pieceWeSingleJumped , spotWeWantToMoveTO and activatedPiece

and runs **winGameFunc()**

**winGameFunc()**

checks for win conditions and runs functions

**pieceCount()**

**changeWhosTurn()**

**pieceCount()**

changes inner html of ID Blue and RedPieceCount

to bluePieces.length which is quary for bluePiece class

**changeWhosTurn()**

just changes turn to 1 if currently 0 and 0 if currently 1

invokes **addClickeventToPieces()**

and  **pieceCount()**

**addClickeventToPieces()**

function adds event listener to blue pieces when turn is 0

red pieces when turn is 1

the click runs function **resestWithRemainingPeices**

**... restarts is all**