
Ruby

Jovana Banovic

Ruby

Ruby is a programming language from Japan. Why i choose Ruby is because of its simplicity . Even though it is a simple language it is powerful as well.



1. Intro

Problem:

Create a TicTacToe Game :

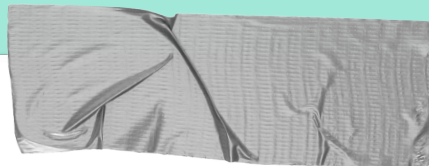
- Create a Board
- Determine all the Win possibilities
- Determine who is going to play
- As the game progressed recreate the board
- Final messages of who won

```
1 class Board
2
3   def initialize
4     @board= Array.new(3) { Array.new(3," ") }
5   end
6
7   def defineframe
8     puts "1 | 2 | 3 |",
9         "4 | 5 | 6 |",
10        "7 | 8 | 9 |"
11     print "\n"
12   end
```

First Function Initialize and Board Frame



This TicTacToe
game is a game
between you and a
friend

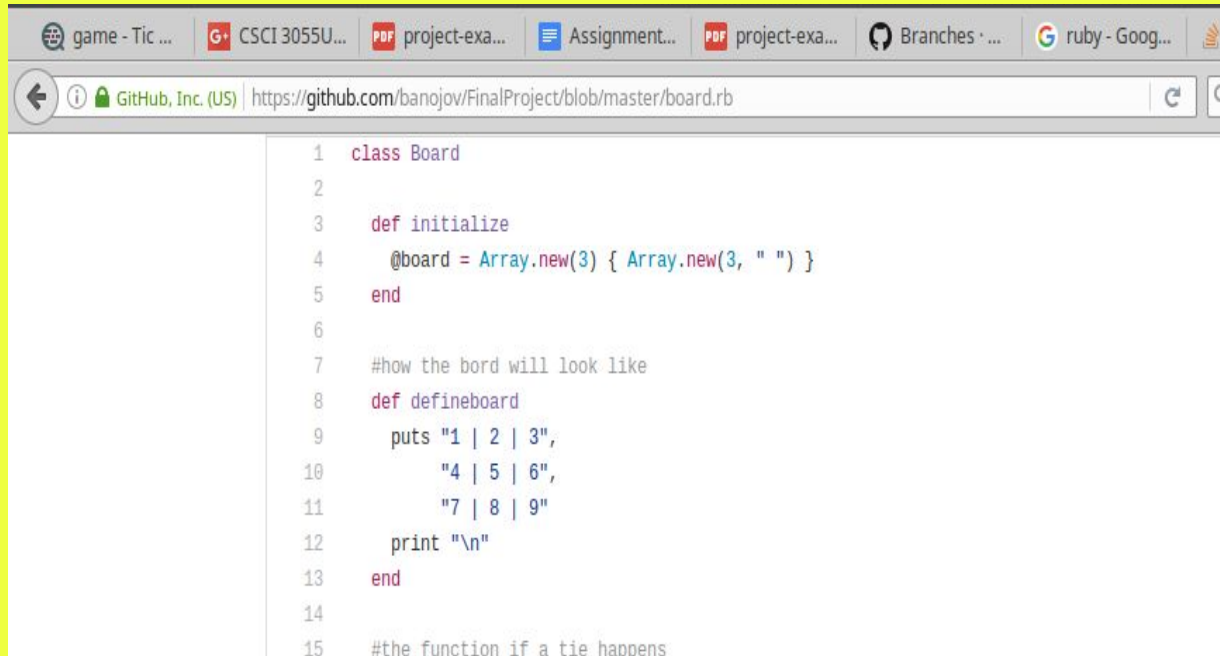


2. Functions


After having a good idea of how the board will look like i proceeded to make a function if a tie occurs,

This was a simple function.

However all the possible wins was the most challenging function since there are multiple ways a player can win



```
1 class Board
2
3   def initialize
4     @board = Array.new(3) { Array.new(3, " ") }
5   end
6
7   #how the bord will look like
8   def defineboard
9     puts "1 | 2 | 3",
10         "4 | 5 | 6",
11         "7 | 8 | 9"
12     print "\n"
13   end
14
15   #the function if a tie happens
```



The board function
was later on
changed to look
more like the
following piece of
code

```

19
20 #print board after each of the players choose their "X" or "O" move
21 def print_Board
22     (0...3).each do |row|
23         print "      "
24         (0...3).each do |col|
25             print @board[row][col]
26             print " | " unless col == 2
27         end
28         print "\n"
29         print unless row == 2
30     end
31     print "\n"
32 end
33 # all the possible wins
34 def find_Winner
35     #diagonal wins
36     if ( @board[0][2] == @board[1][1] && @board[1][1] == @board[2][0] ) ||
37         ( @board[0][0] == @board[1][1] && @board[1][1] == @board[2][2] )
38         return @board[1][1] unless @board[1][1] == " "
39     end
40     #wins on the sides
41     (0...3).each do |i|
42         if @board[i][0] == @board[i][1] && @board[i][1] == @board[i][2]
43             return @board[i][0] unless @board[i][0] == " "
44
45             elsif @board[1][i] == @board[2][i] && @board[0][i] == @board[1][i]
46                 return @board[0][i] unless @board[0][i] == " "
47             end
48         end
49     # the game is tied
50

```



All the possible win
instances

```

64 #let the game begin
65 board = Board.new
66 currentplayer = "X"
67
68 puts "\n" * 100
69 board.defineboard
70
71 while board.find_Winner == "U"
72
73   puts " #{currentplayer}'s turn. Choose a Number!"
74
75   moves = gets.chomp.to_i - 1
76   row = moves / 3
77   col = moves % 3
78
79   if board.remove(currentplayer, row, col)
80     if currentplayer == "X"
81       currentplayer = "O"
82     else
83       currentplayer = "X"
84     end
85   else
86     puts "Invalid move, please select again\n\n"
87   end
88
89   board.print_Board
90 end
91
92 winner = board.find_Winner
93
94 if winner == "C"
95   puts "   PLAYER C WON G A M E"

```



Implementation of
board

Definition of X and O


```
Terminal - jovana@jovana-E5450: ~/Desktop/FinalProject
File Edit View Terminal Tabs Help
1      X |  | 
      |  | 
0's turn. Choose a Number!
2      X | 0 | 
      |  | 
X's turn. Choose a Number!
3      X | 0 | X
      |  | 
0's turn. Choose a Number!
4      X | 0 | X
      0 |  | 
X's turn. Choose a Number!
5      X | 0 | X
      0 | X | 
0's turn. Choose a Number!
6      X | 0 | X
      0 | X | 0
X's turn. Choose a Number!
7      X | 0 | X
      0 | X | 0
      X |  | 
X ' S   W I N
      X | 0 | X
      0 | X | 0
      X |  | 
jovana@jovana-E5450:~/Desktop/FinalProject$
```

Source: theguan.com


Code in action

```
1 def merge_sort(lists)
2   return lists if lists.size <= 1
3
4   middle = lists.size / 2
5   left = lists[0, middle]
6   right = lists[middle, lists.size]
7   merge(merge_sort(left), merge_sort(right))
8 end
9
10 def merge(left, right)
11   sorted = []
12   if left.first <= right.first
13     sorted << left.shift
14   else
15     sorted << right.shift
16   end
17 end
18 sorted.concat(left).concat(right)
19 merge_sort [1,2,3,4,5,6,7,8]
20 end
```

Ruby Merge Sort

```

public void MergeSort(int lowerIndex,int highgerIndex){

    if(lowerIndex<highgerIndex){

        int middle=lowerIndex+(highgerIndex-lowerIndex)/2;
        //sorts the left side of the array

        MergeSort(lowerIndex,middle);
        //right side

        MergeSort(middle+1,highgerIndex);
        //merge both sides

        mergeParts(lowerIndex,middle,highgerIndex);
    }
}

public void mergeParts(int lowerIndex, int highgerIndex, int
middle ){

    for(int i=lowerIndex;i<=highgerIndex;i++){
        tempMergArr[i]=array[i];
    }
}

```

```

int i=lowerIndex;
int j=middle+1;
int k=lowerIndex;

while(i<=middle && j <= highgerIndex){

    if(tempMergArr[i] <= tempMergArr[j]){

        array[k]=tempMergArr[i];
        i++;

    }else{

        array[k]=tempMergArr[j];
        j++;
    }
    k++;

}

if(i<= middle){

    array[k]=tempMergArr[i];
    k++;
}

}
}

```

Java Merge Sort



4. Closing

- We cannot mention Ruby without mentioning Ruby on Rails
- https://www.youtube.com/watch?v=OaDhY_y8WTo&feature=youtu.be
- A good intro to ruby on rails
- Ruby on Rails, is a server-side web application framework written in Ruby under the MIT License. Rails is a model–view–controller framework that provides default structures for a database, a web service, and web pages.



<https://www.tutorialspoint.com/ruby-on-rails/rails-examples.htm>