Tic Tac Toe

Command line games

Kai kai@42.us.org

Summary: The art of building a program.

Contents

Chapter I

Gameboard

- How do we play tic tact toe? It starts with drawing a # ... two sets of perpendicular lines. A grid.

- Decide for yourself how to design a game board. It could look something like this:

```
 x | o |
-----------
   | o | o
-----------
   |   | x
```

- Be careful and plan ahead! As the game progresses, we want to be able to store and display values of o, x, or nil in each square.

- How will you save the data? It could work multiple ways. For example, you could use nine different variables...

```
square_0 = nil;
square_1 = nil;
square_2 = 'x';
square_3 = nil;
...
```

- Or an array...

```
board = Array.new
board[2] = 'x'
```

- Or an array of arrays if you really want.

```
board = Array.new
top_row = Array.new
board[0] = top_row
...
```

- Choose the data structure that will hold your game board, and add it to your program. Let the whole board start "empty" with no x's or o's.

Chapter II

Printing

- When you first start the tictac.rb, it should print a welcome message to the terminal, and the empty game board.

```
?> tictac.rb
Welcome to Intergalactic Tic Tac Toe!
   |   |
-----------
   |   |
-----------
   |   |
```

- Later, you will want to print the same game board with something else in the squares besides empty spaces.
- Create a `method` in your program which takes the state of the board as parameter(s) and prints it out.
- Test it by changing the board state a few times within your program, and printing out each version.

```
def print_board_parameters(board)
        #your method definition here
end

# blank board
print_board(board)

board[2] = ’x’

# after one move
print_board(board)
```

Chapter III

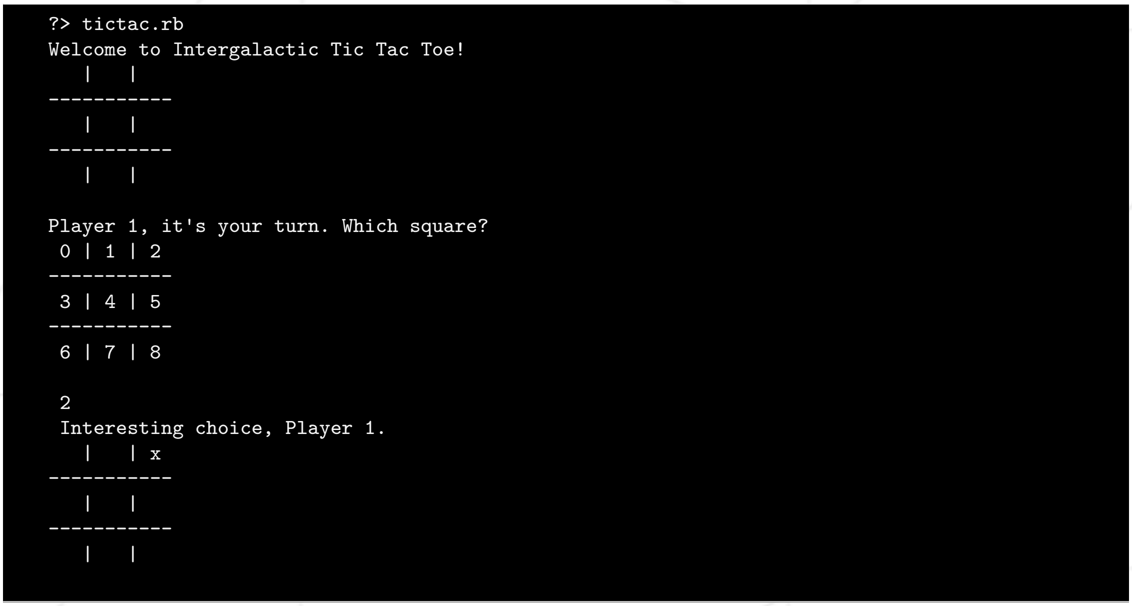
Making the First Move

- OK, that's cool. You can print out different versions of the board if you set them yourself from inside the code.

- Now, have your program change the board based on input from the command line.

- Start with Player 1. Ask them for a move, and read what choice they make. Then update that part of the board to hold an 'x'.

- Print the board before and after.



```
?> tictac.rb
Welcome to Intergalactic Tic Tac Toe!
   |   |
-----------
   |   |
-----------
   |   |

Player 1, it's your turn. Which square?
 0 | 1 | 2
-----------
 3 | 4 | 5
-----------
 6 | 7 | 8

 2
 Interesting choice, Player 1.
   |   | x
-----------
   |   |
-----------
   |   |
```

Chapter IV

Looping

Awesome! You know how to print the gameboard and store whether there is an x or an o in each square. You also have been able to receive a move from a player and make it real.

- In the main body of your program, add a loop which repeatedly asks Player 1 and Player 2 for their moves.

- It should alternate between the two players and print the board after each move.

- For now, the loop can be infinite.

Chapter V

Validity

Validity checking - one of the most important things you will ever need to remember as a programmer.

- To prevent cheating, insert a check to make sure that a players' move is valid.
- A move is not valid if it is outside the game board.
- A move is not valid if someone's x or o is already played there.

Chapter VI

Did I Win?

Looking good! Kudos that you have made it this far.

- Create a method in your program which determines if anyone has won yet.
- This may be the most complex part of the program so far! Remember to check for horizontal, vertical, and diagonal lines.
- Add this check to your loop so that we look for a win after every move.
- Print a congratulatory message when you do find a win!