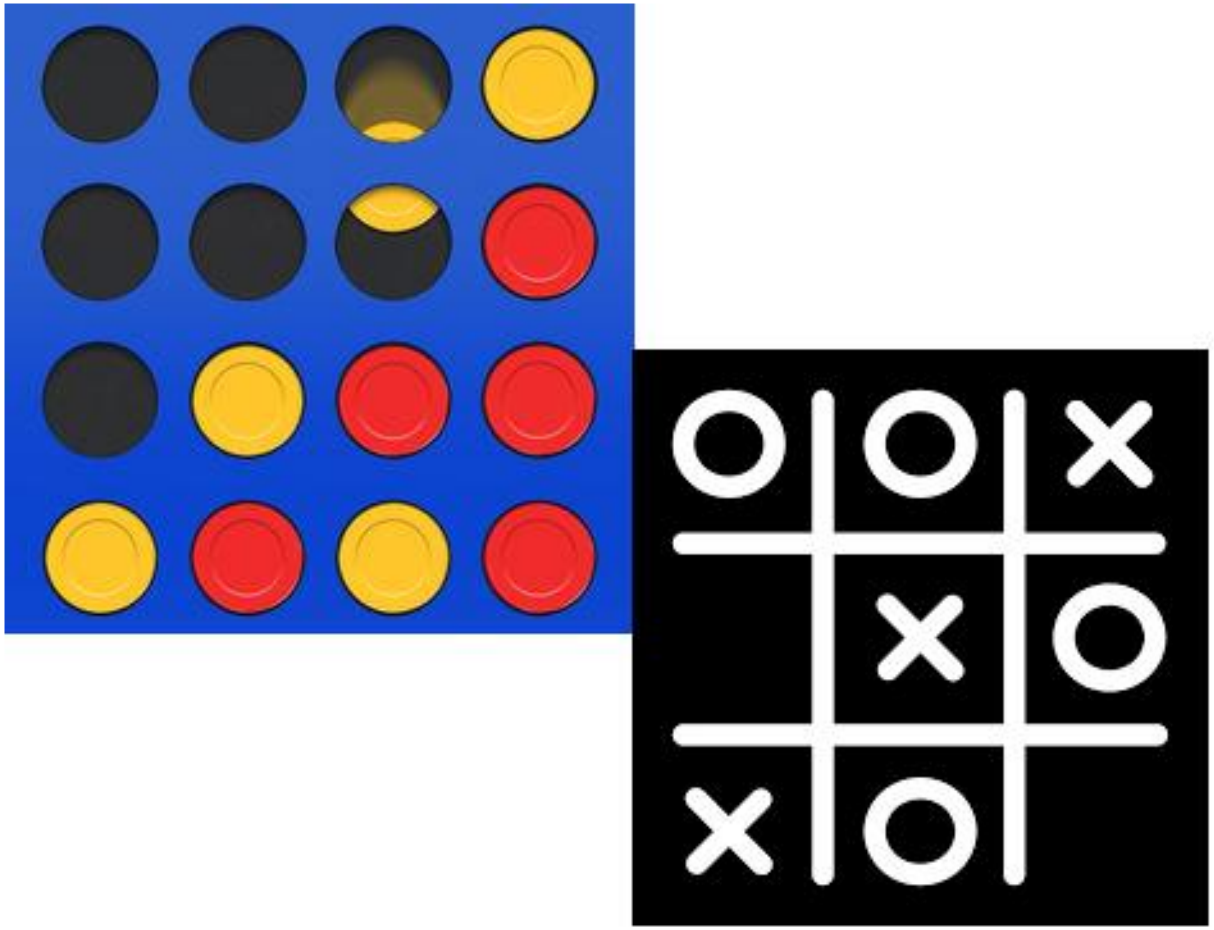
# Programming Fundamentals 2

## Assignment 2 – Grid Based Games App

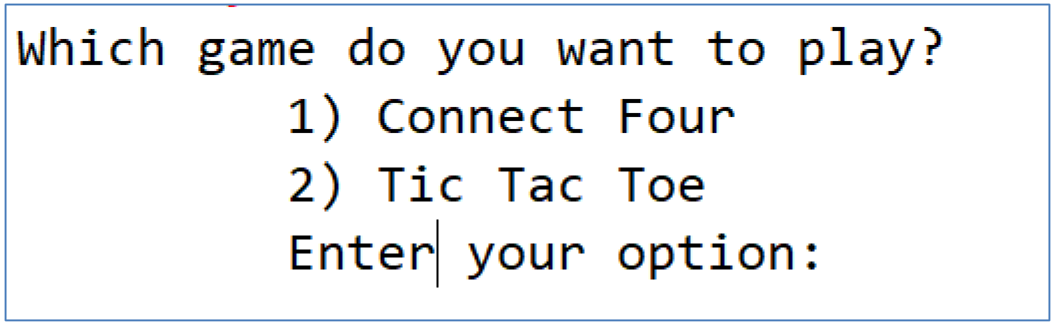
Produced by: Dr. Siobhán Drohan
Mairead Meagher

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
http://www.wit.ie/



Aim of Assignment is to develop a console based game app that allows you to play both TicTacToe and Connect4.



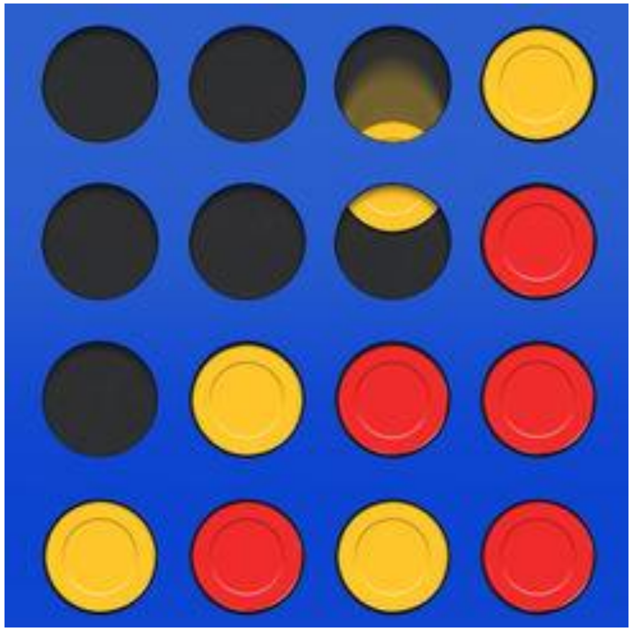
```
Which game do you want to play?
         1) Connect Four
         2) Tic Tac Toe
         Enter your option:
```

Aim of Assignment is to
develop one console based game app
that allows you to play both
TicTacToe and Connect4.

In this assignment, we are particularly assessing the following areas:

- Inheritance
- Polymporhism
- Abstraction

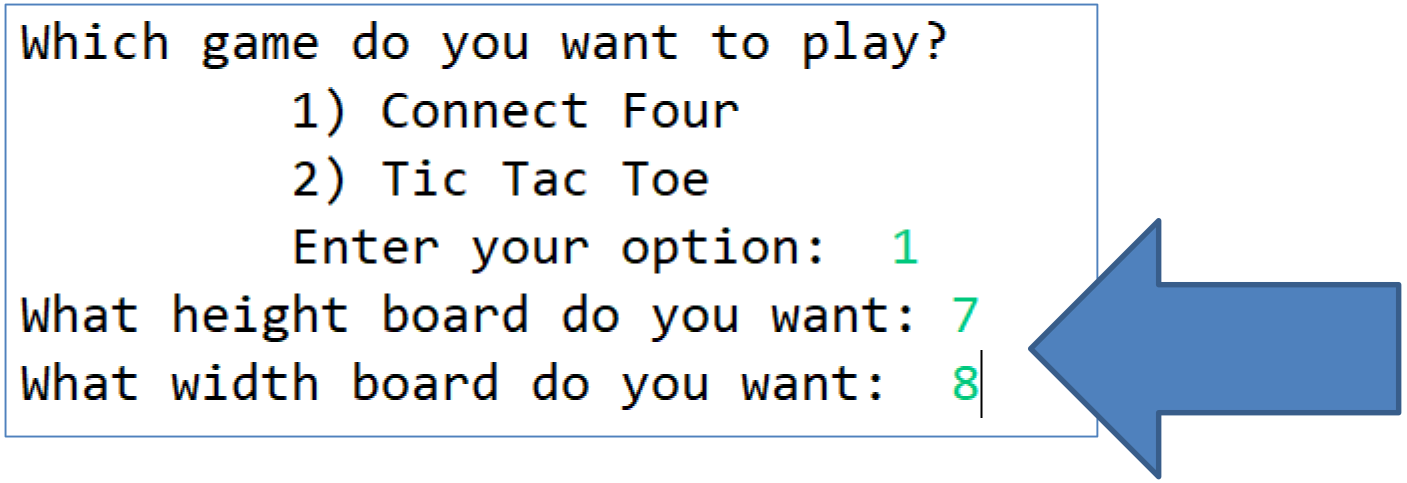
as well as Collections, Persistence and Encapsulation.



Some rules and sample screen shots.

**Note**: our screen shots are very minimal; they are just a prototype.  You don't have to replicate them; you can design your own user experience!

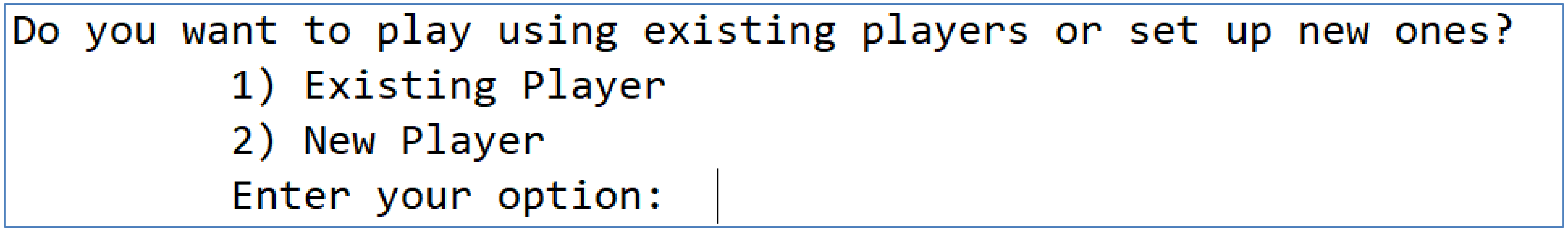
Connect 4 board can be any size (min 4x4).



```
Which game do you want to play?
         1) Connect Four
         2) Tic Tac Toe
         Enter your option:  1
What height board do you want: 7
What width board do you want:  8
```

You can choose to play as:

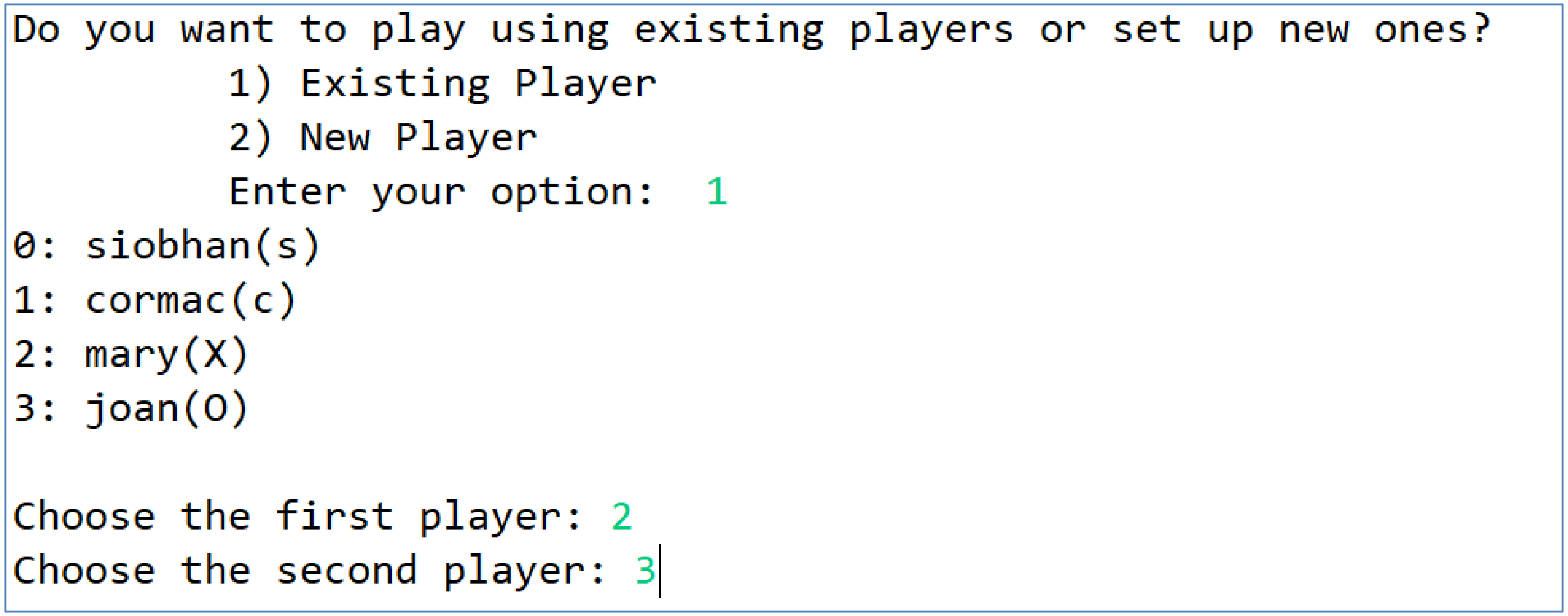
- existing players (loaded from a file) **or**
- new players (which are then added to the file)



```
Do you want to play using existing players or set up new ones?
         1) Existing Player
         2) New Player
         Enter your option:
```

You can choose to play as:

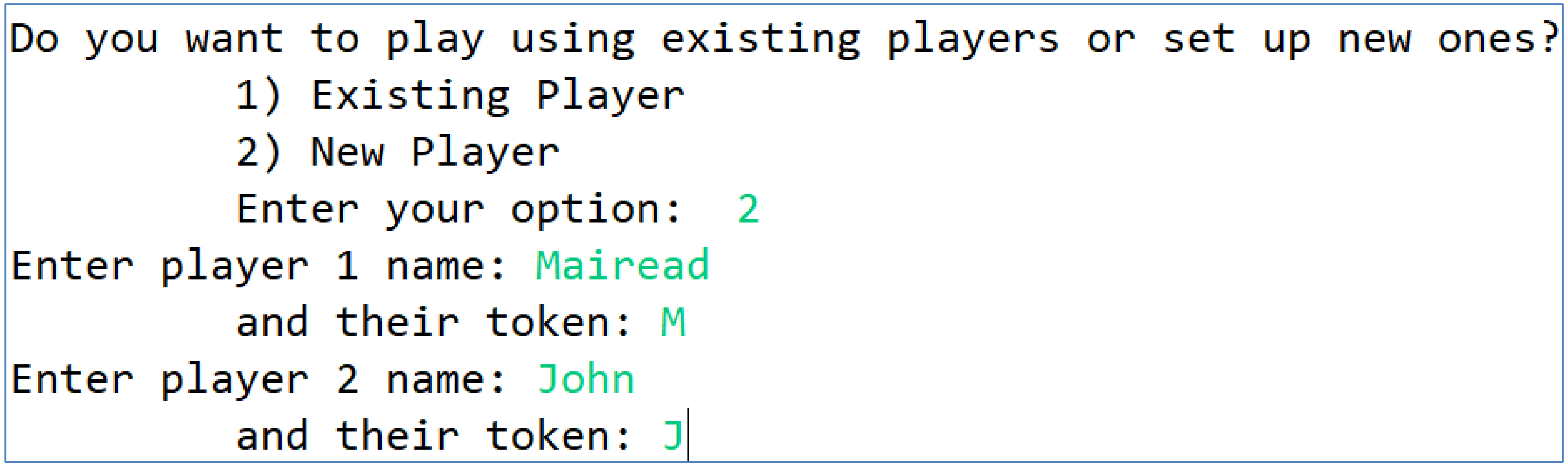
- existing players (loaded from a file) **or**
- new players (which are then added to the file)



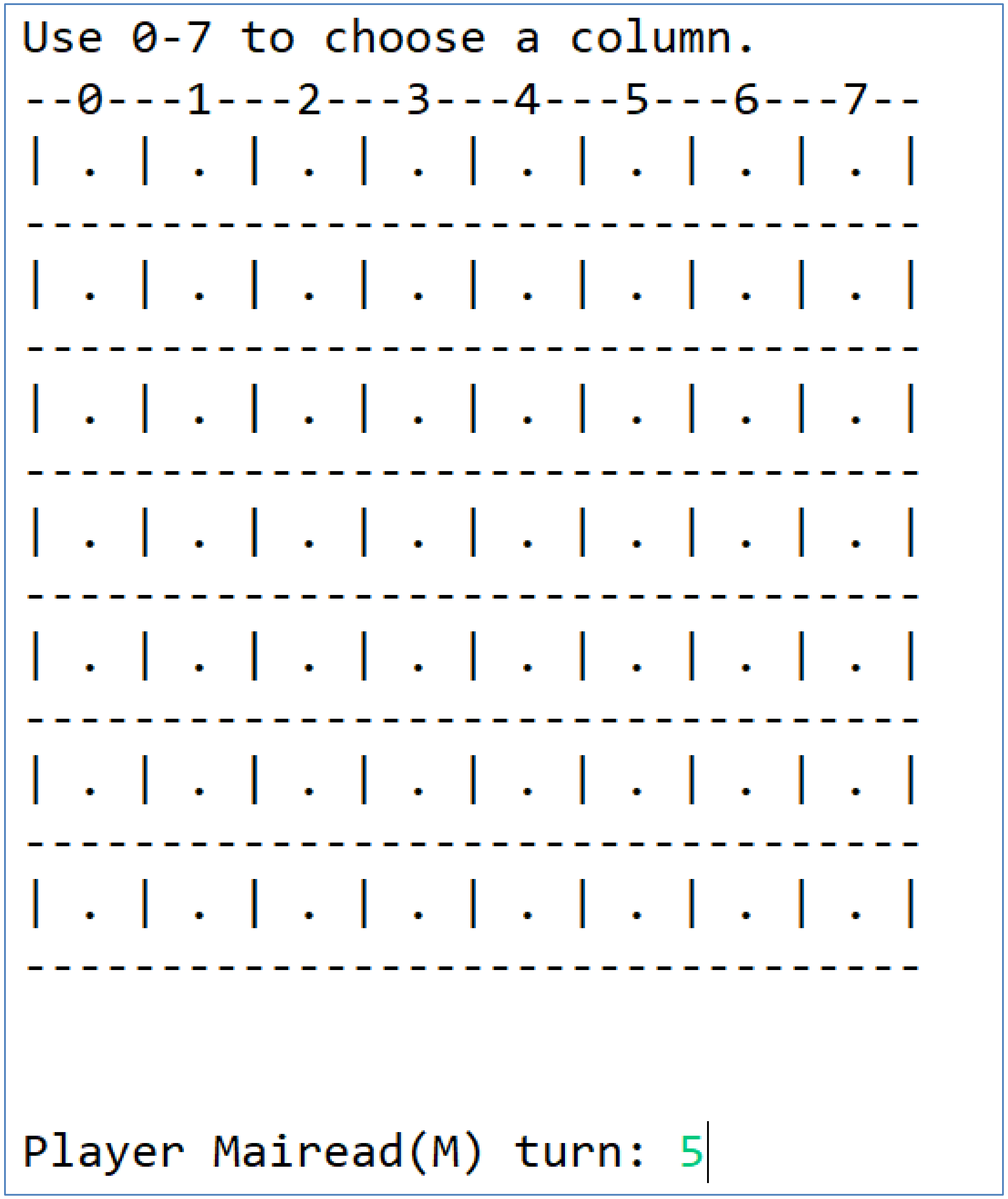
```
Do you want to play using existing players or set up new ones?
          1) Existing Player
          2) New Player
          Enter your option:  1
0: siobhan(s)
1: cormac(c)
2: mary(X)
3: joan(O)

Choose the first player: 2
Choose the second player: 3
```

You can choose to play as:

- existing players (loaded from a file) **or**
- new players (which are then added to the file)



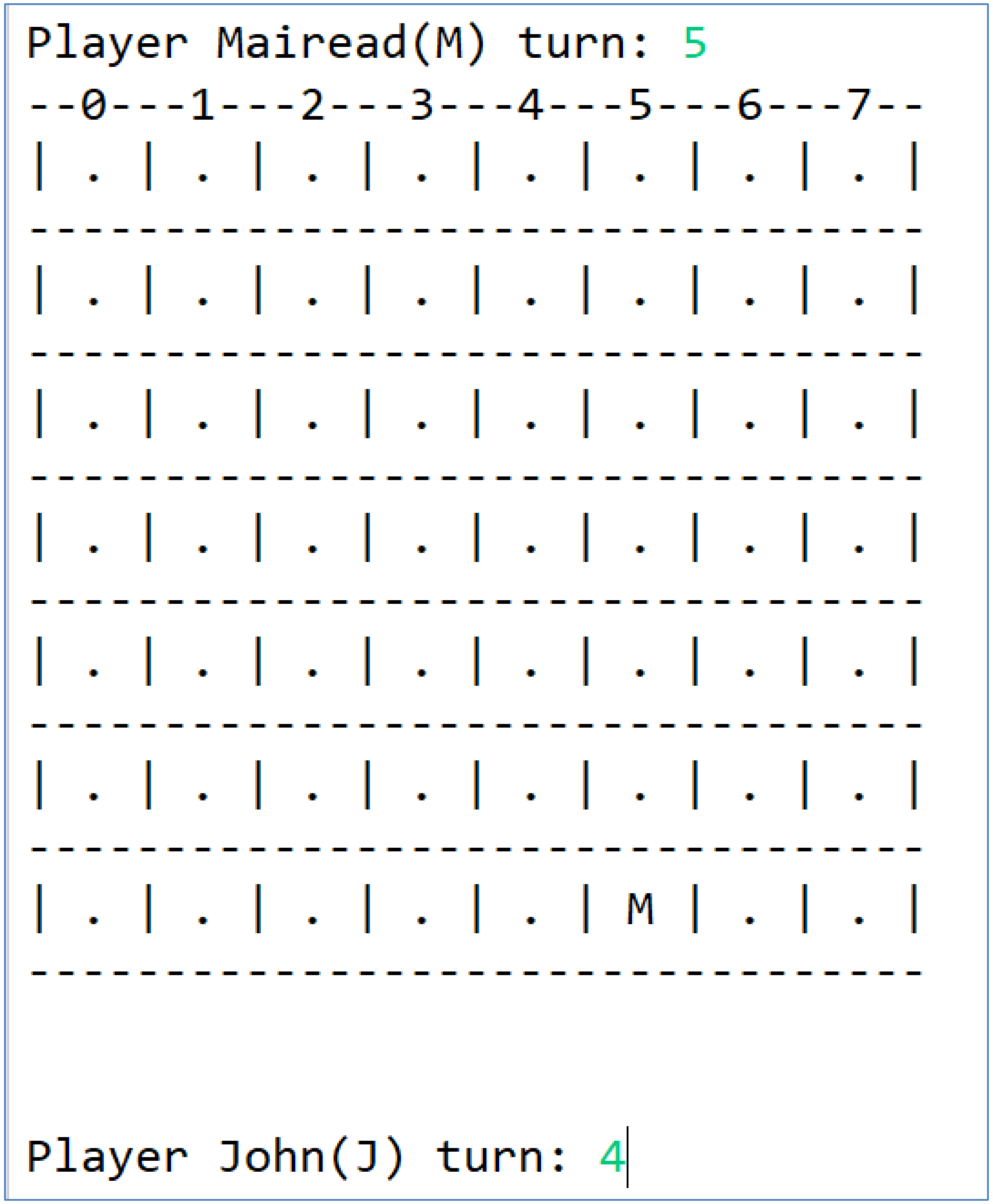
```
Do you want to play using existing players or set up new ones?
         1) Existing Player
         2) New Player
         Enter your option:  2
Enter player 1 name: Mairead
         and their token: M
Enter player 2 name: John
         and their token: J
```



```
Use 0-7 to choose a column.
--0---1---2---3---4---5---6---7--
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------



Player Mairead(M) turn: 5
```

Board is drawn

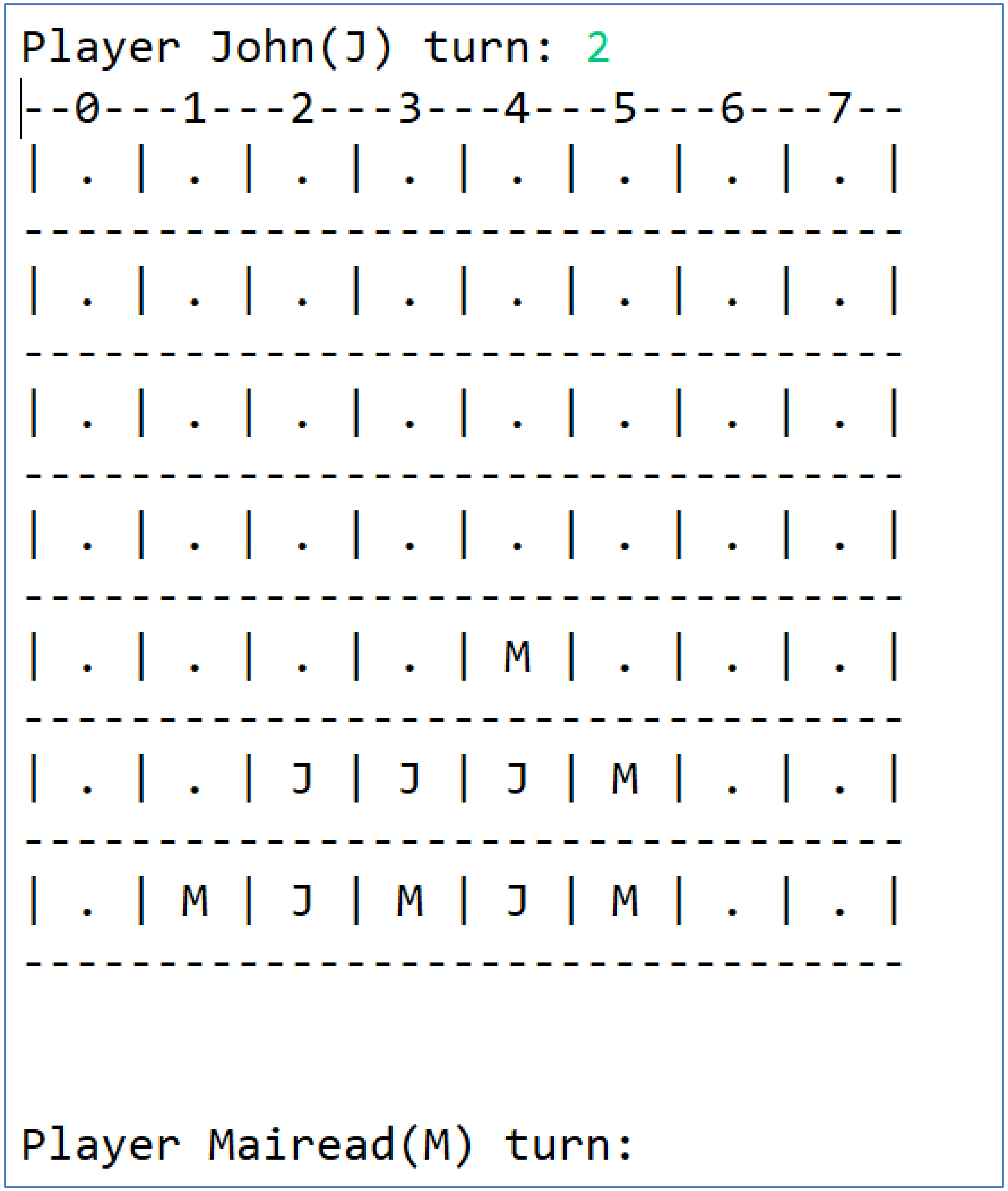
Player 1 takes turn



```
Player Mairead(M) turn: 5
--0---1---2---3---4---5---6---7--
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | M | . | . |
---------------------------------



Player John(J) turn: 4
```

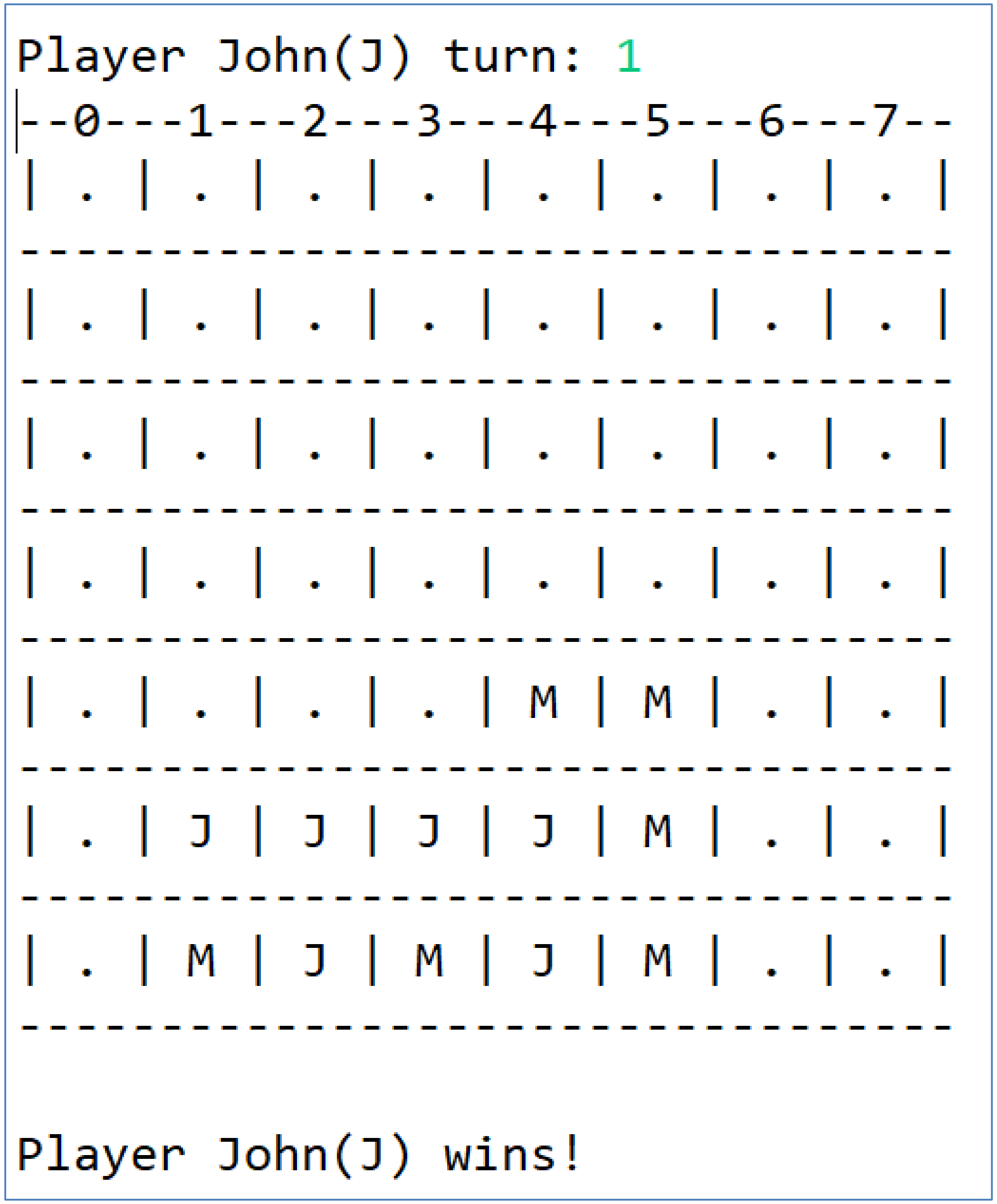
Board is redrawn

Player 2 takes turn



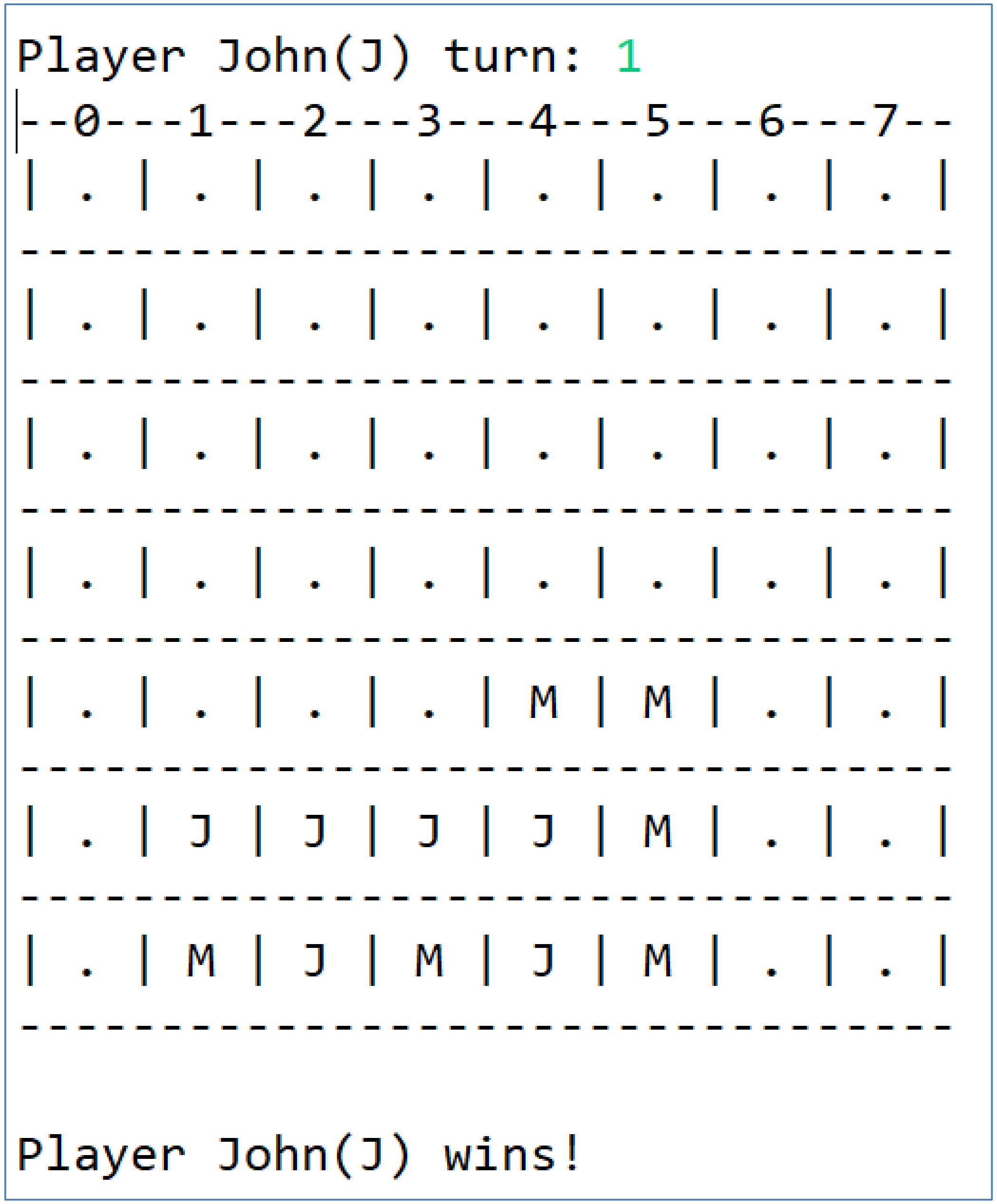
```
Player John(J) turn: 2
--0---1---2---3---4---5---6---7--
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | M | . | . | . |
---------------------------------
| . | . | J | J | J | M | . | . |
---------------------------------
| . | M | J | M | J | M | . | . |
---------------------------------



Player Mairead(M) turn:
```

After a few turns, board might look like this.



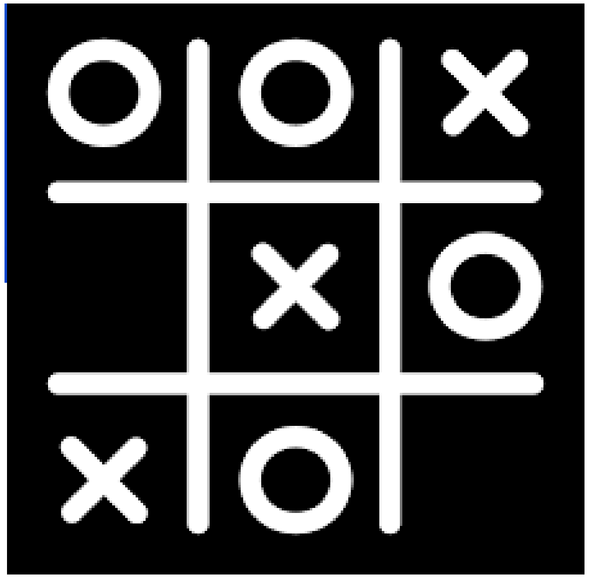
```
Player John(J) turn: 1
--0---1---2---3---4---5---6---7--
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | M | M | . | . |
---------------------------------
| . | J | J | J | J | M | . | . |
---------------------------------
| . | M | J | M | J | M | . | . |
---------------------------------


Player John(J) wins!
```

Player “John” wins.



```
Player John(J) turn: 1
--0---1---2---3---4---5---6---7--
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | . | . | . | . |
---------------------------------
| . | . | . | . | M | M | . | . |
---------------------------------
| . | J | J | J | J | M | . | . |
---------------------------------
| . | M | J | M | J | M | . | . |
---------------------------------


Player John(J) wins!
```

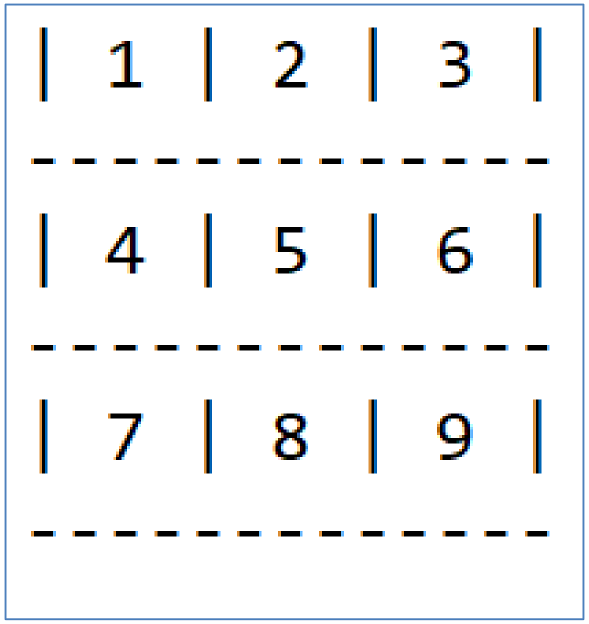
A player wins when they get four in a row, horizontally, vertically or diagonally.



Some rules and sample screen shots.

**Note**: our screen shots are very minimal; they are just a prototype.  You don't have to replicate them; you can design your own user experience!

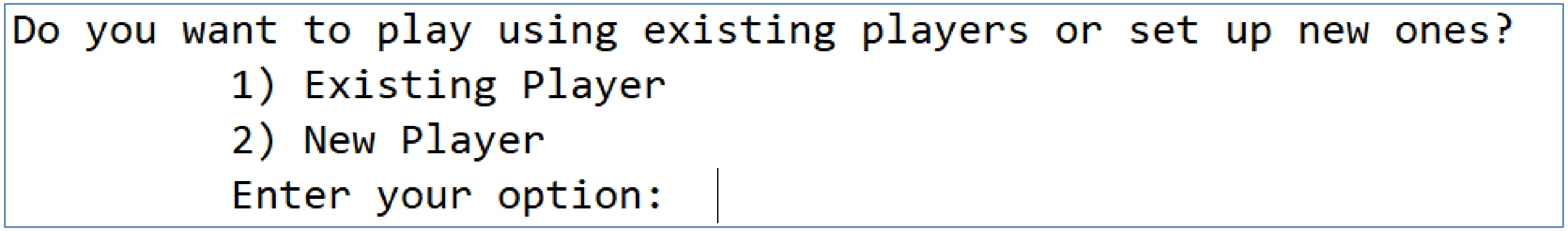
## TicTacToe board must be 3x3.



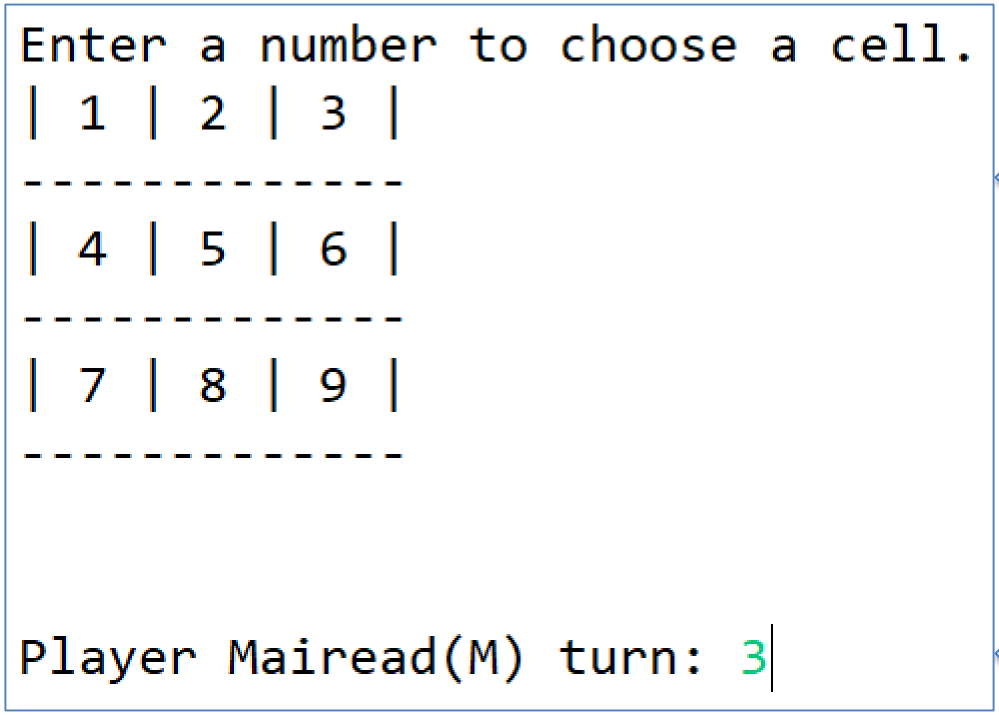
```
| 1 | 2 | 3 |
-------------
| 4 | 5 | 6 |
-------------
| 7 | 8 | 9 |
-------------
```

As with Connect4, you can choose to play as:

- existing players (loaded from a file) **or**
- new players (which are then added to the file)



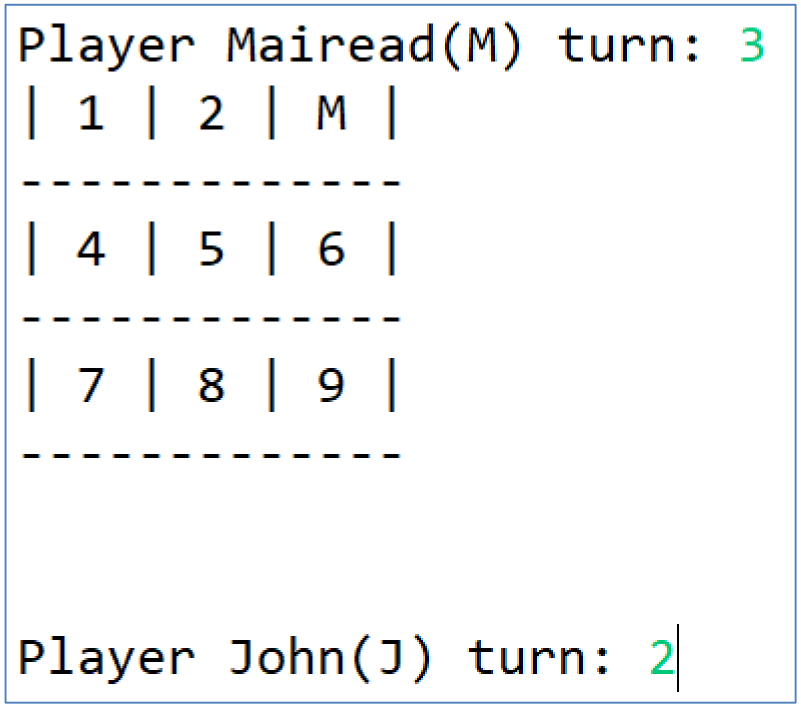
```
Do you want to play using existing players or set up new ones?
         1) Existing Player
         2) New Player
         Enter your option:
```



```
Enter a number to choose a cell.
| 1 | 2 | 3 |
-------------
| 4 | 5 | 6 |
-------------
| 7 | 8 | 9 |
-------------


Player Mairead(M) turn: 3
```

Board is drawn

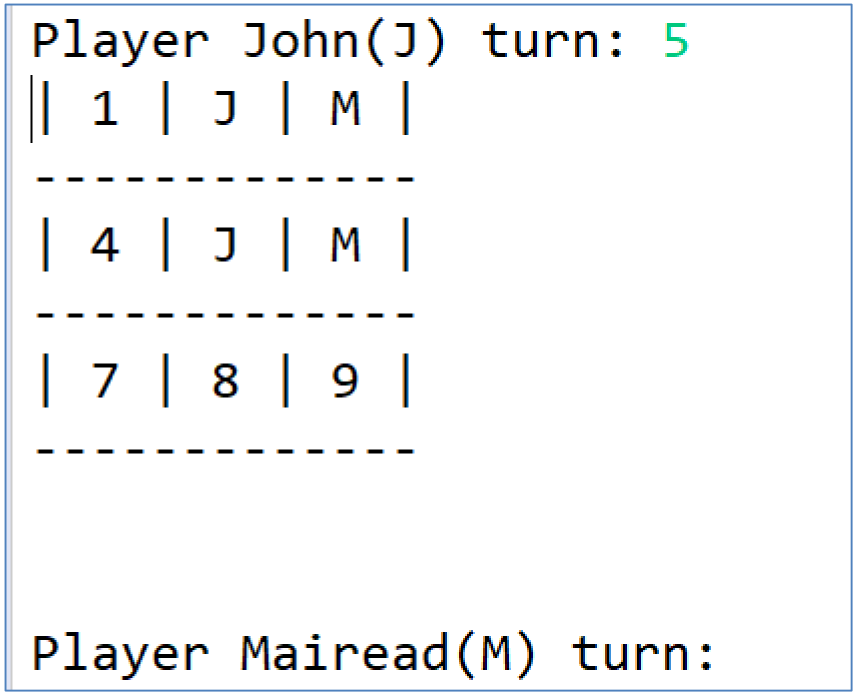
Player 1 takes turn



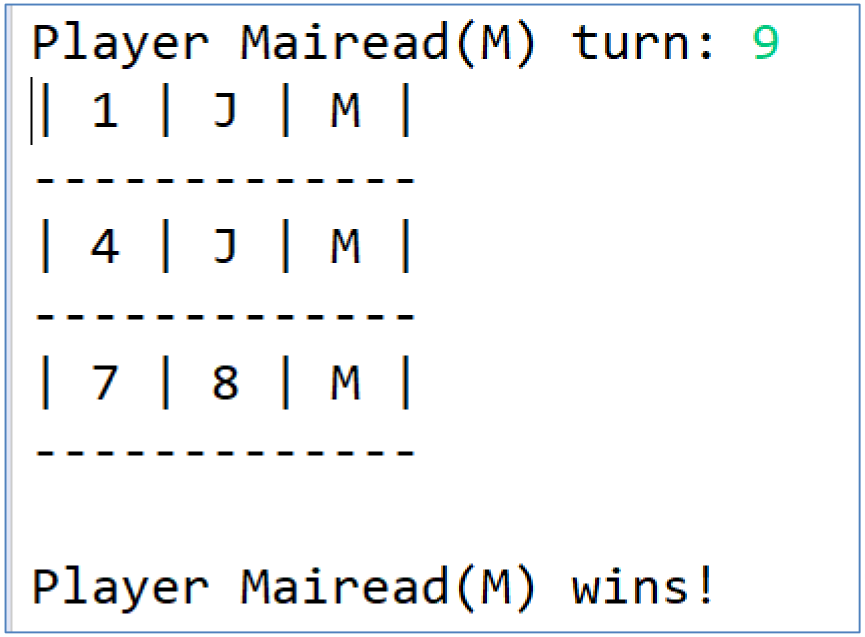
```
Player Mairead(M) turn: 3
| 1 | 2 | M |
-------------
| 4 | 5 | 6 |
-------------
| 7 | 8 | 9 |
-------------



Player John(J) turn: 2
```

Board is redrawn

Player 2 takes turn

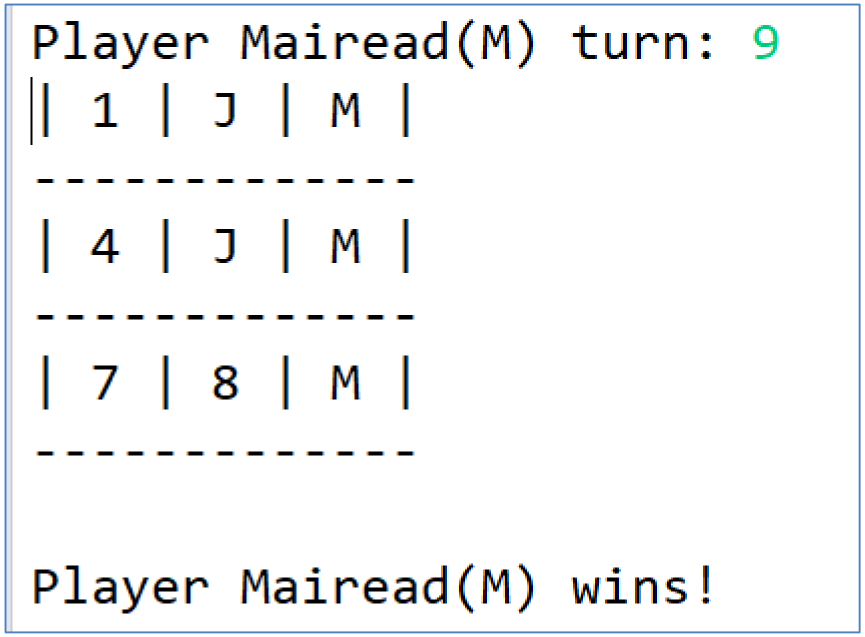

```
Player John(J) turn: 5
| 1 | J | M |
--------------
| 4 | J | M |
--------------
| 7 | 8 | 9 |
--------------


Player Mairead(M) turn:
```

After a few turns, board might look like this.



```
Player Mairead(M) turn: 9
| 1 | J | M |
-------------
| 4 | J | M |
-------------
| 7 | 8 | M |
-------------


Player Mairead(M) wins!
```

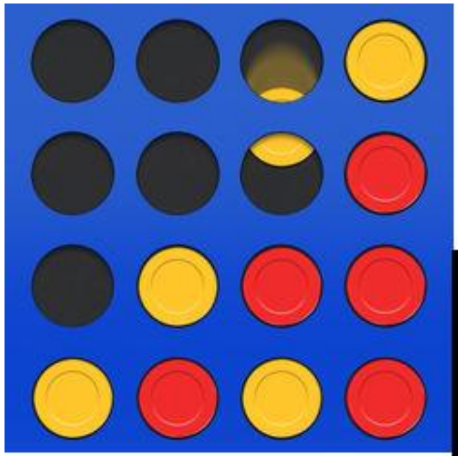
Player “Mairead” wins.

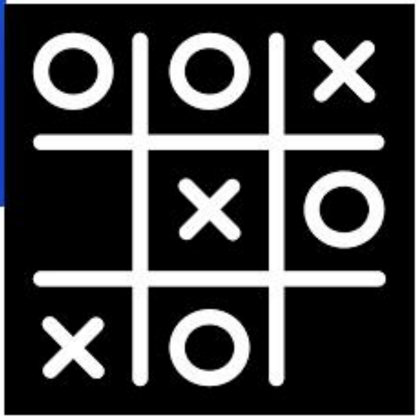


```
Player Mairead(M) turn: 9
| 1 | J | M |
-------------
| 4 | J | M |
-------------
| 7 | 8 | M |
-------------


Player Mairead(M) wins!
```

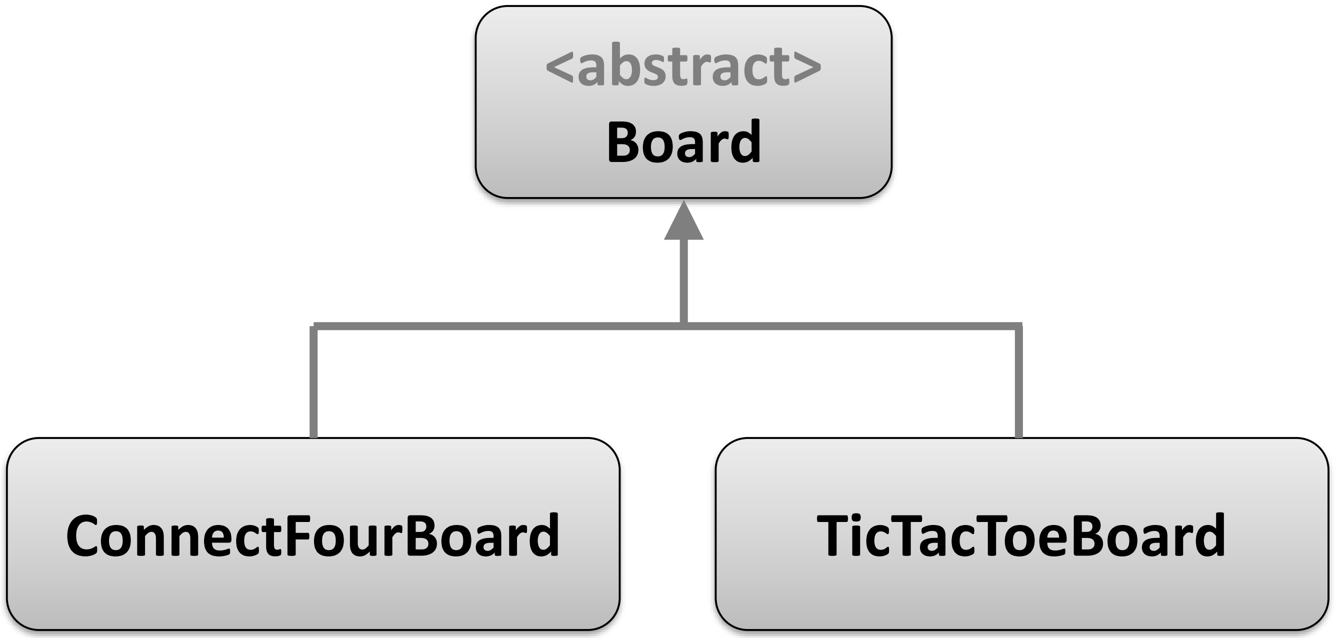
A player wins when they get three in a row, horizontally, vertically or diagonally.

Some Architecture Hints

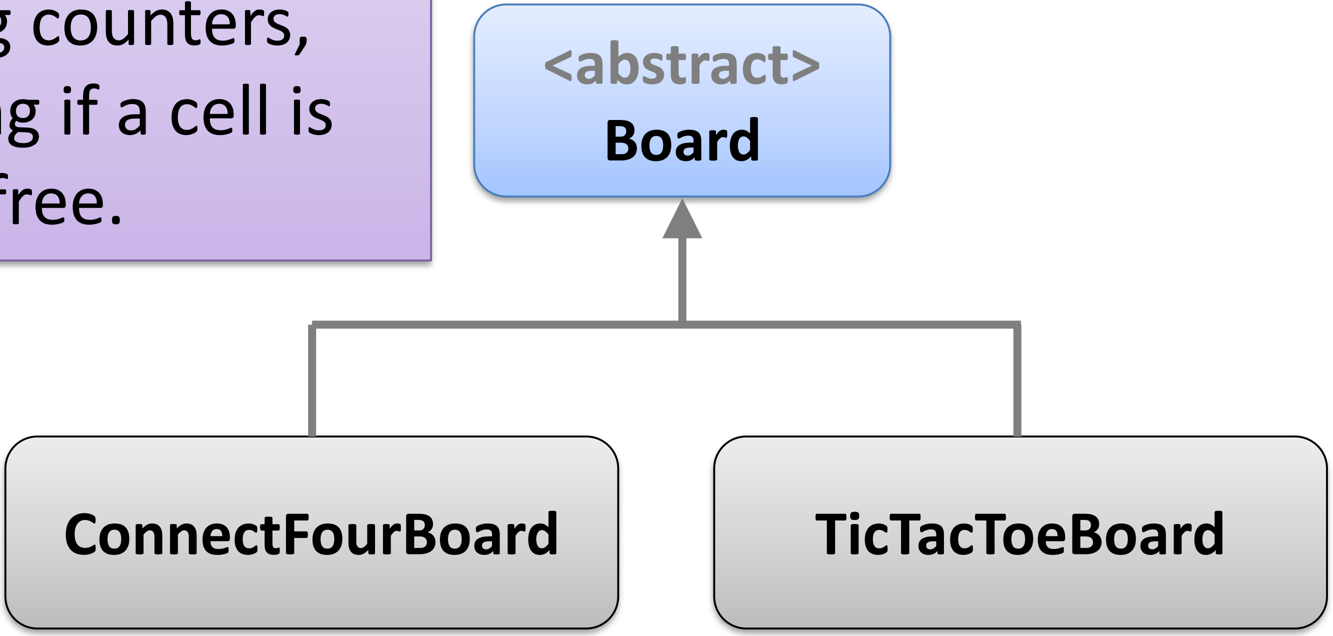




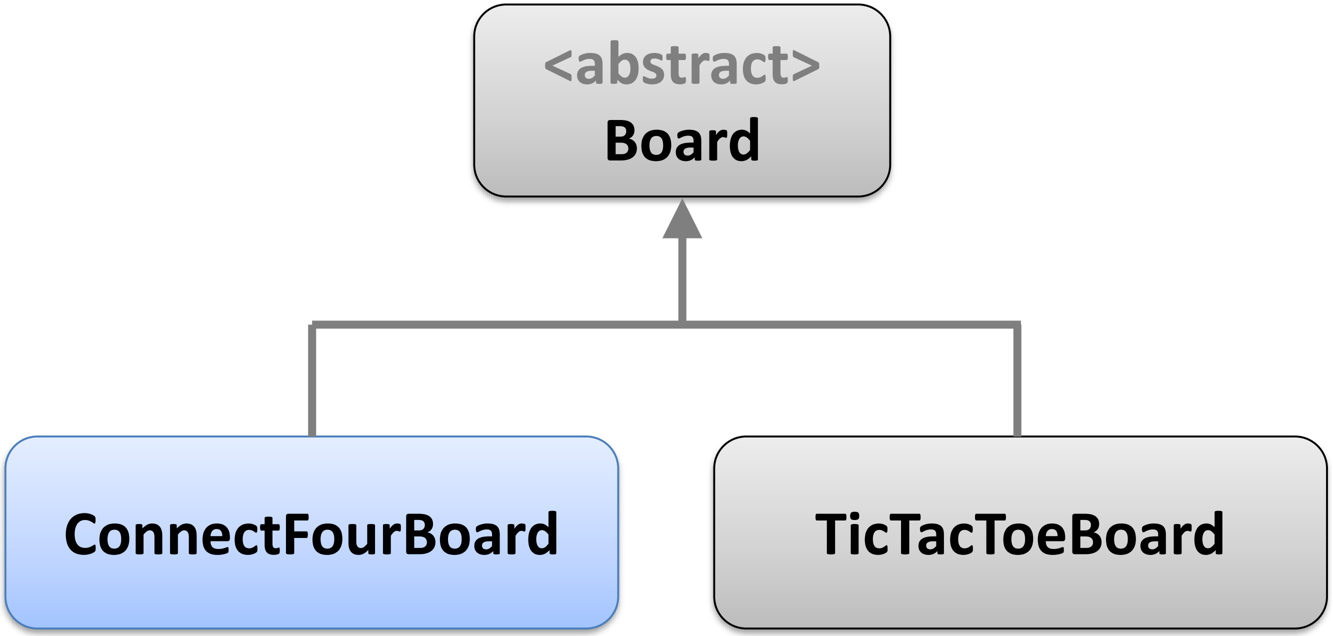
**The Boards!**


<abstract>
Board
ConnectFourBoard
TicTacToeBoard

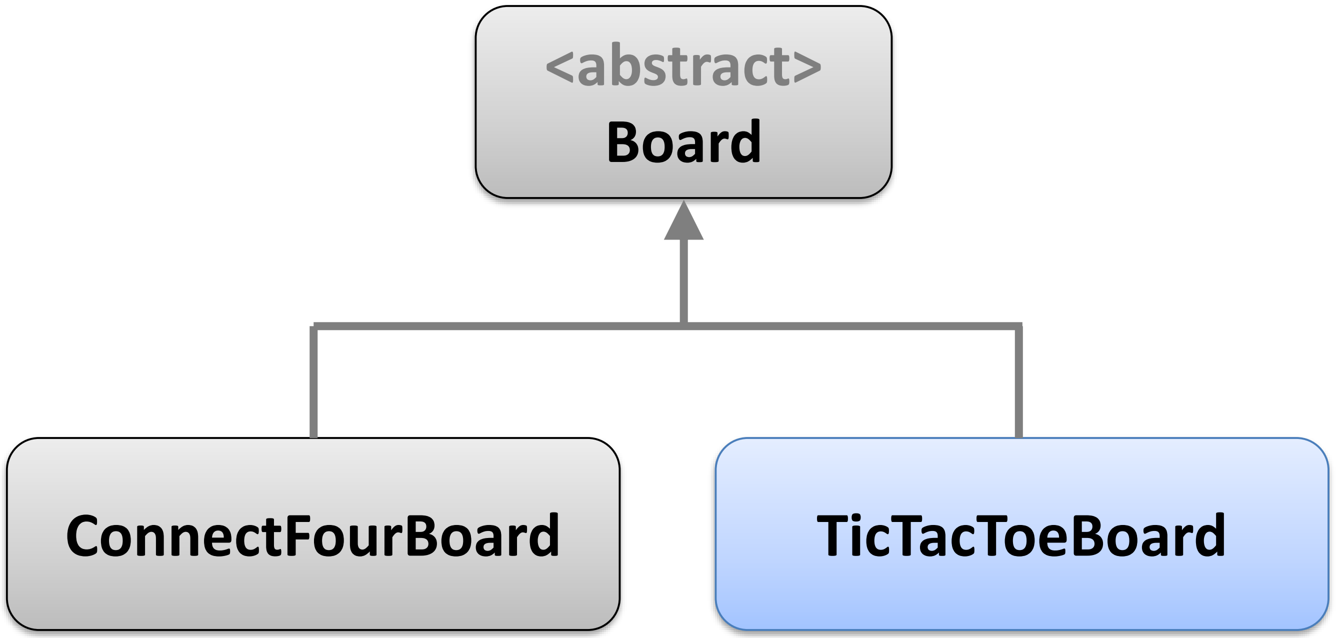
The Board class contains a 2D array. Also, you may have abstract methods for say winning the game, placing counters, checking if a cell is free.



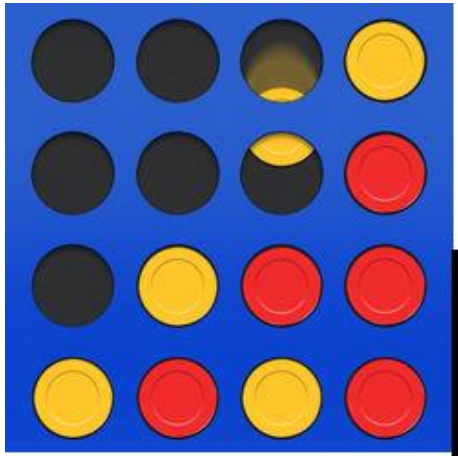
The ConnectFourBoard class manages the behaviour of the Connect4 board. You would provide implementations of any abstract methods here and items specific to the ConnectFour board in here.

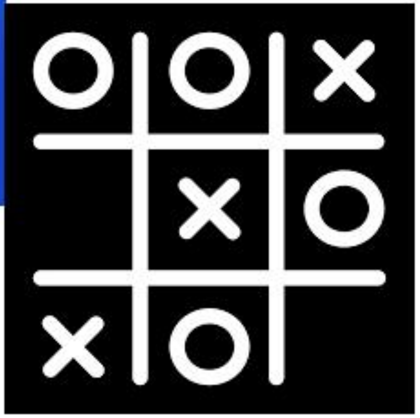


The TicTacToeBoard class manages the behaviour of the TicTacToe board.  You would provide implementations of any abstract methods here and items specific to the TicTacToe board in here.



Some Architecture Hints





**The Players!**

**Player**

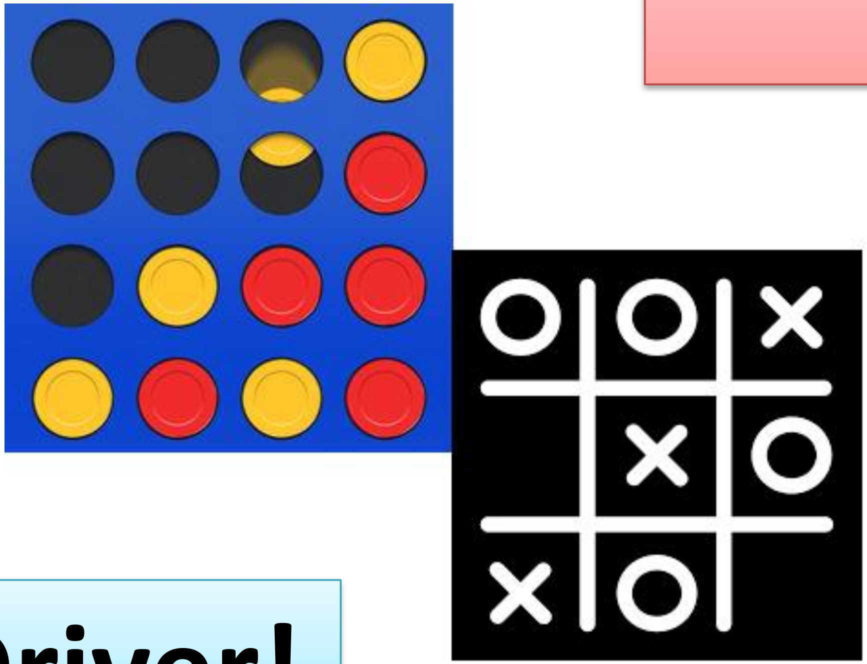
The Player class manages a player's information e.g. name and token.

## PlayerList

The PlayerList class manages the file stored list of players and also the two current players in the game.

```
<object-stream>
  <list>
    <models.Player>
      <name>siobhan</name>
      <token>s</token>
    </models.Player>
    <models.Player>
      <name>cormac</name>
      <token>c</token>
    </models.Player>
    <models.Player>
      <name>mary</name>
      <token>X</token>
    </models.Player>
    <models.Player>
      <name>joan</name>
      <token>O</token>
    </models.Player>
</list>
</object-stream>
```

Some Architecture Hints



The Driver!

**Driver**

The Driver class starts the chosen game, handles player turns and manages the user I/O.

# Any Questions?