Tic-Tac-Toe Kata

The initial SW Craftsmanship assessment consists in developing a Tic-Tac-Toe game adhering to the three rules of Test-Driven Development[1].

Tic Tac Toe - Game Rules:

The kata consists in developing a standard Tic Tac Toe. For reference, these are the rules of the game:
https://en.wikipedia.org/wiki/Tic-tac-toe

UAT Scenarios

To show your expertise is required, you commit your scenarios in the red-green-refactor steps following the implementation roadmap below.

1 – Game Board Creation phase:

```
Game Board Creation…
  | |
-+-+-
  | |
-+-+-
  | |

Board Created.
The game will start with player X
```

2 – Player X won with a vertical line

```
Player X:
X| |
-+-+-
X|O|
-+-+-
X| |O

PLAYER X WON!
```

3 – Player O won with a horizontal line

```
Player O:
X| |X
-+-+-
O|O|O
-+-+-
X| |

PLAYER O WON!
```

4 – Player X won with a diagonal line

```
Player X:
X| |
-+-+-
O|X|
-+-+-
O| |X

PLAYER X WON!
```

[1] Uncle Bob Martin – The Three Rules Of TDD: http://butunclebob.com/ArticleS.UncleBob.TheThreeRulesOfTdd

5 – The game ends with a draw

```
Player X:
X|O|X
-+-+-
O|O|X
-+-+-
X|X|O

THE GAME ENDS WITH A DRAW!
```

Objective:

The system should run in BOT mode (random BOT moves for players X & O), printing on the screen all the player's moves (with a 2-second timeout between each round) until someone wins or the game ends with a draw.

Graduation test scoring system

The kata source must be in GitHub/GitLab/etc. The first commit/push must be an empty directory, which will trigger the graduation test's start.

Scoring sheet:

- *Solo mode*: NO PAIR - NO MOB – NO copy & paste from other sources.
- *Timebox*: 2 hours -- 4 pomodori[2] from the first git push of an empty repository.
- *Notes*: code like in pair programming session storing every pomodoro into a NOTES.md.
- *Prove the progression*: commit the NOTES.md at least at every pomodoro (preferable at every stage Red-Green-Refactor) to show the kata coding progression.
- *Simple design*: NOTES.md must show your simple design approach and how you organized the code between the growth of features VS tech debt and refactoring.
- *Emerging Architecture*: Simple design & emerging architecture should be your approach (no big thinking upfront)
- *Three laws of TDD*: the code must follow a strict TDD way via cycles of Red/Green/Refactor. Code written before a test or that exceeds the test's scope is a failure on this check.
- *Git History*: commit any Red-Green-Refactor cycle to have a readable history in Git.
- *Coverage*: The code coverage for this simple kata must be 100%.
- *Clean Code – Basic refactoring*:
  1. *Documentation as code*: the test suite looks like a book.
  2. *Domain Model I*: the test report shows the Domain Model (*DDD ubiquitous language*[3]).
  3. *Domain Model II*: the code uses the same Domain Model of the tests.
  4. *Well-written prose*: the test report explains the Tic Tac Toe game without further reading.
  5. *Extract until you can't*: code behaviors encapsulated into atomic behaviors—the complexity described via natural language without any drift of interpretation VS execution.
- *Working SW*: **the code must be a working artifact**.

---

[2] Pomodoro Technique – Wiki: https://en.wikipedia.org/wiki/Pomodoro_Technique

[3] Martin Fowler – Ubiquitous Language: https://martinfowler.com/bliki/UbiquitousLanguage.html

Graduation test – timetable

Now is the time to start:

- Prepare your scaffolding with only an empty project.
- Add the libs/packages to code in TDD/ATDD/BDD.
- Eventually and all the tooling ecosystems keep a clean code (Linter, git-hooks, CI-CD, etc.).
- Clone the empty scaffolding.
- The timebox of 2 hours starts!!
- Code within the 4 Pomodoro cycles.
- Before the 2 hours time gate, push all your code into the repo.

- Take a 15 mins break and wind down!
- Go to the graduation test scoring sheet. Tick the boxes that describe your code.
- Your starting point would be the WHITE BELT if you missed at least one.
- Suppose you checked all of them; CONGRATS! Move to the Yellow belt and repeat the evaluation of your code with the new scoring sheet. Do it until one check is unmarked.