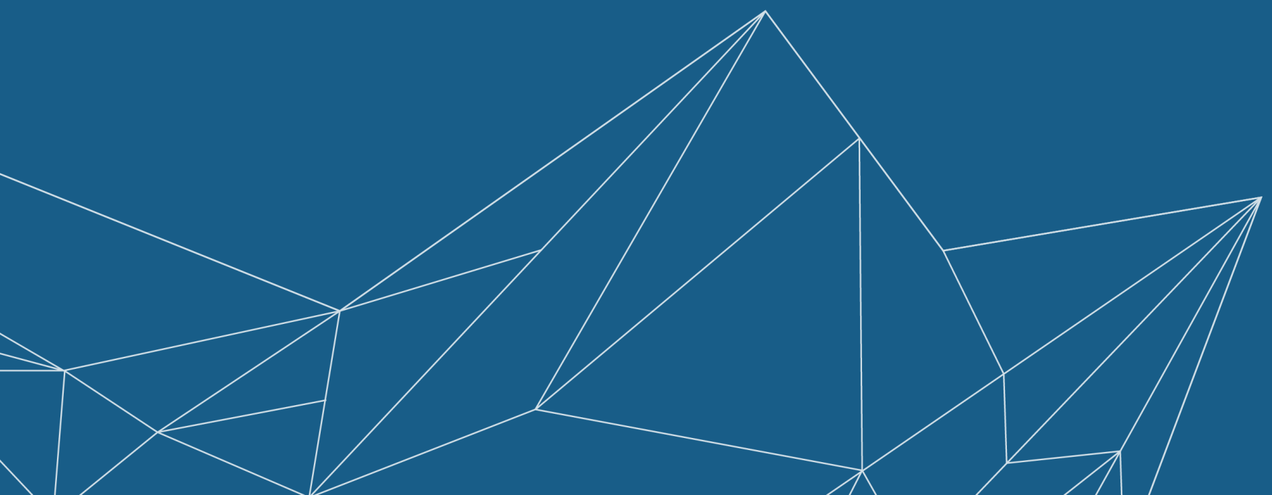
Programming in Java – Part 09 – Acceptance tests

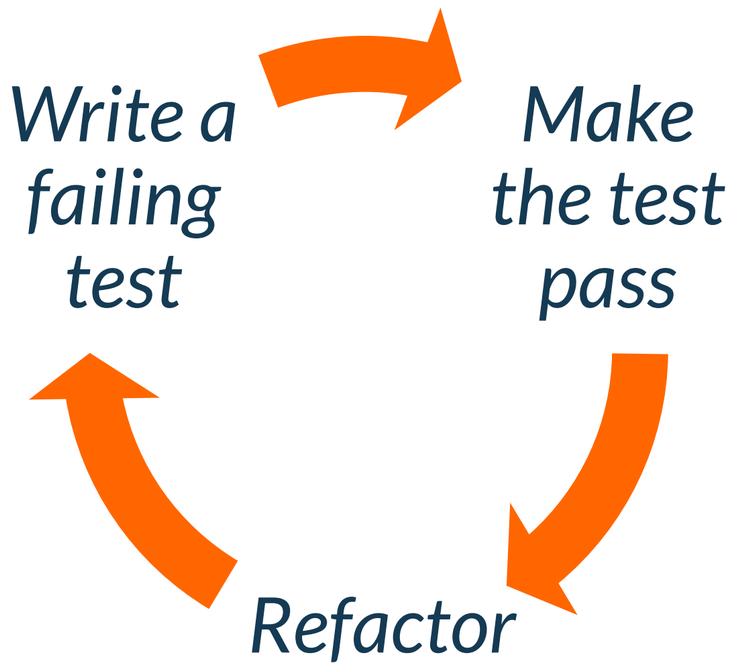
Paolo Vercesi

ESTECO SpA

Acceptance tests



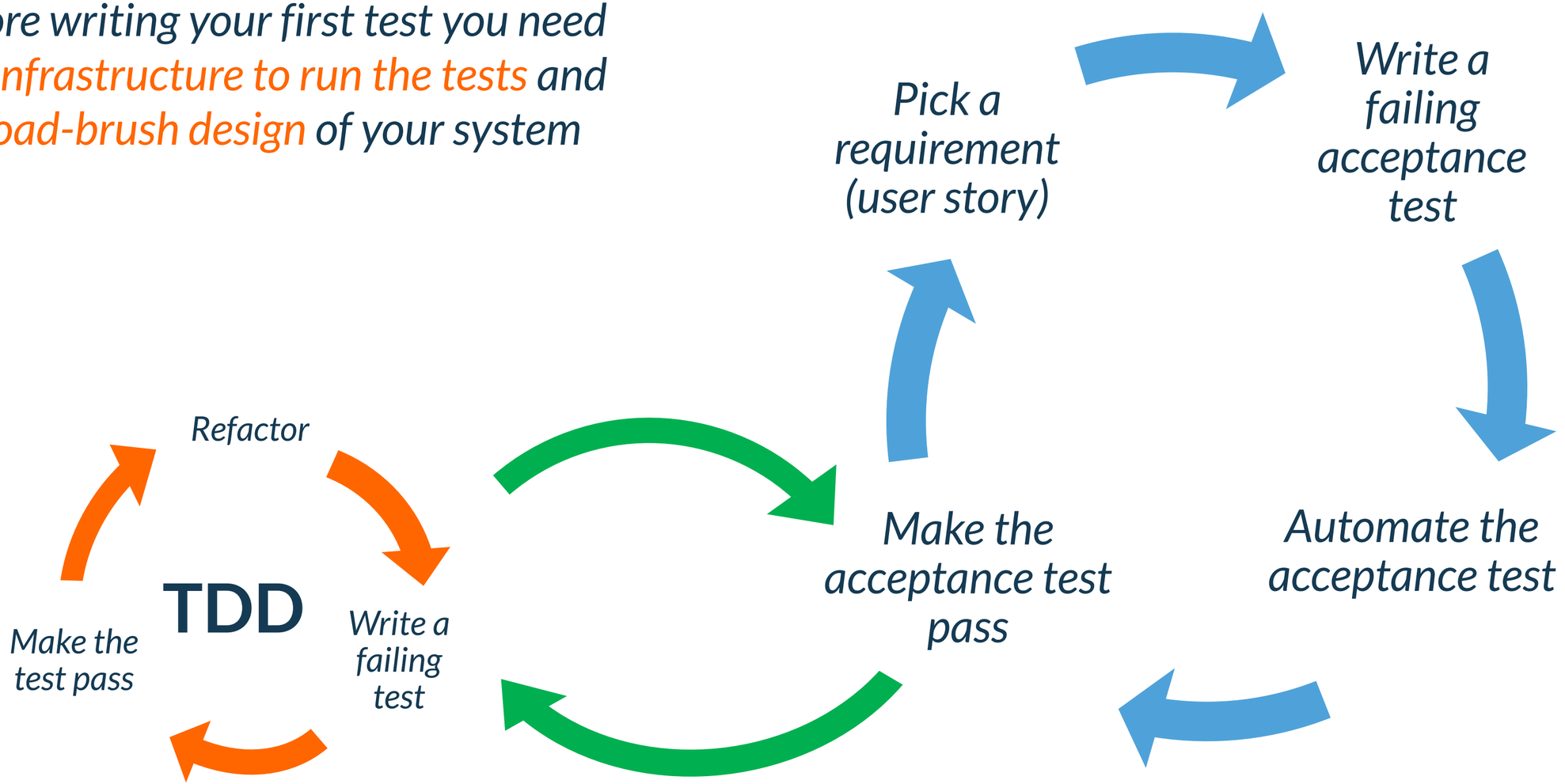
TDD Cycle



- *System grows by adding tests for the new features into an existing infrastructure*
- *What about the very first feature?*
- *You should start every new feature by writing an acceptance test*
- *An acceptance test should exercise the system end-to-end*
  - *We are making a small exception because we'll not test the GUI*

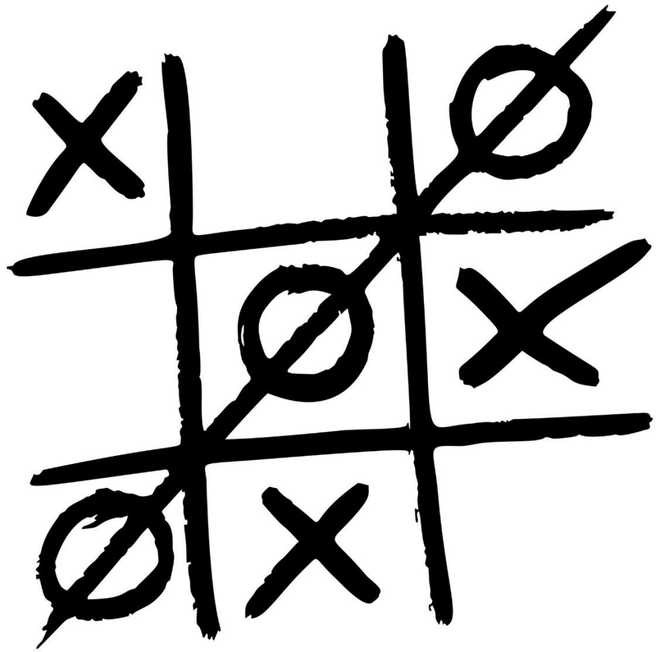
A proposed process

Before writing your first test you need the infrastructure to run the tests and a broad-brush design of your system



Problem

Develop an application to play tic-tac-toe



Where should you begin?

Let’s start with a test!

- *Which kind of test?*
- *So far, you know* **unit tests** *only, but you haven’t any class*
- *You should write an acceptance test that’s closer to the problem statement*
- *Let’s start by refining the problem statement into requirements*

Refining the problem into requirements

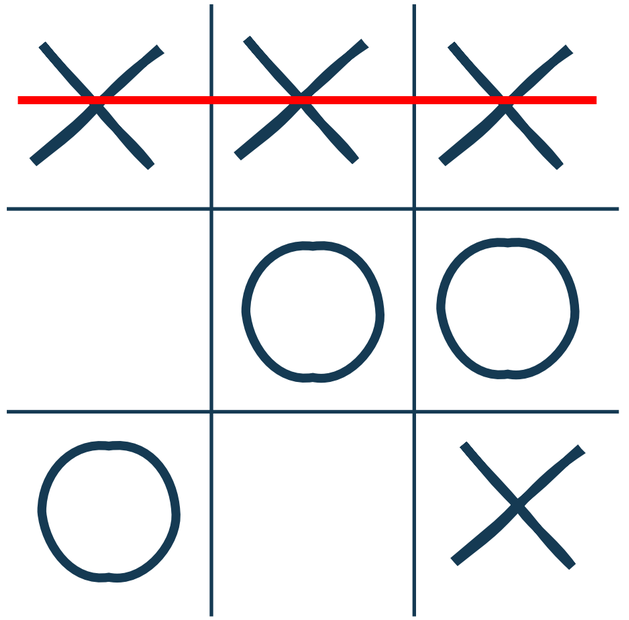
“Develop an application to play tic-tac-toe”

- *Two players (O and X) alternatively place their marks on the (3x3) board*
- *When a player places three marks in a row, the player is declared the winner of the game*
- *When the board is full and there is no winner, a draw is declared*
- *Each player places their marks by clicking on a cell of the board*
- *A player should not be allowed to click on cell on which there is already a mark*
- …

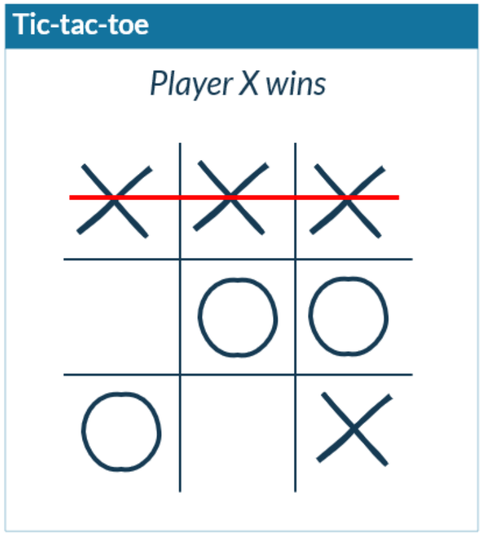
You can start with a graphic mock-up

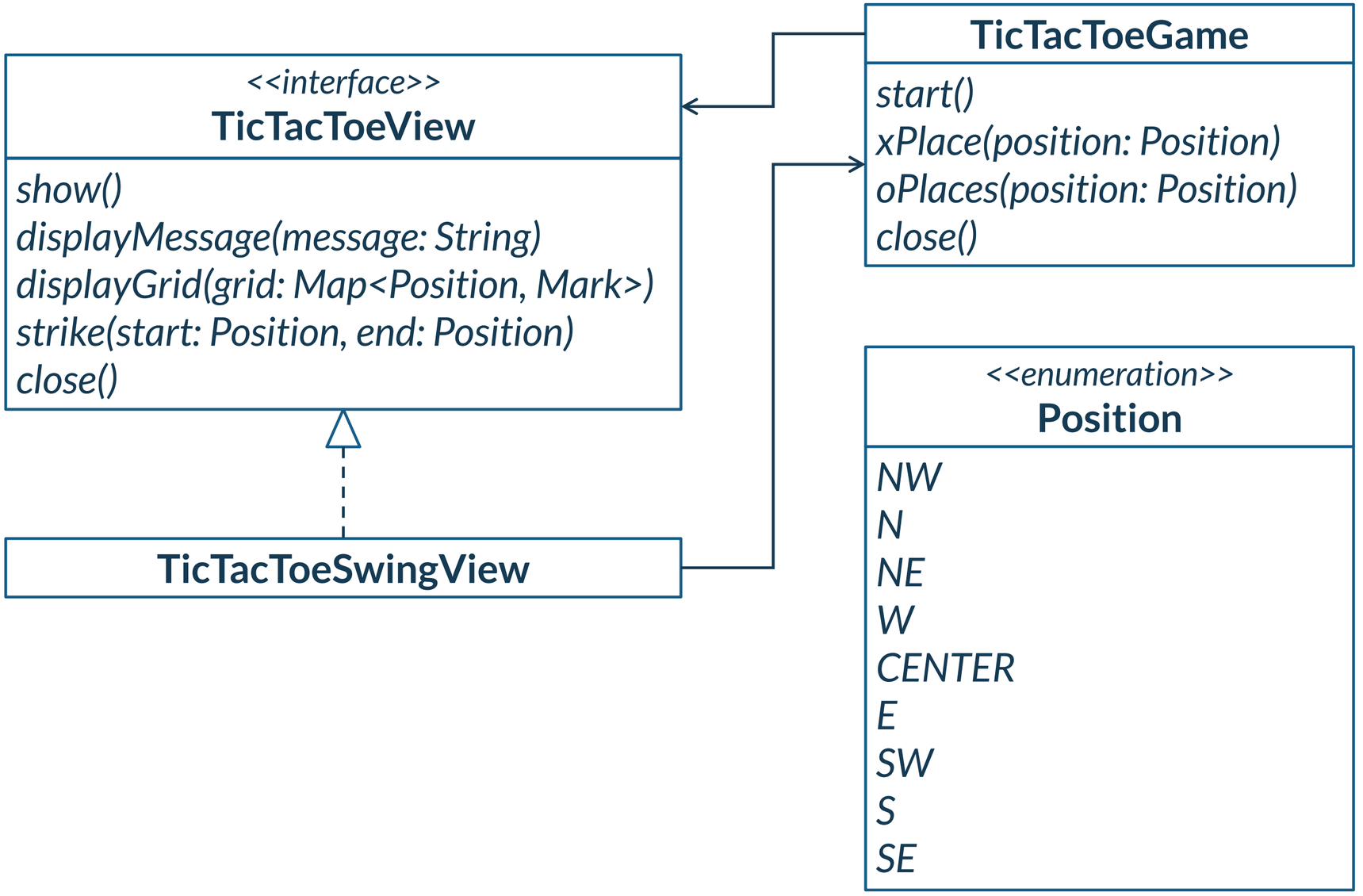
Tic-tac-toe

Player X wins



Broad-brush design

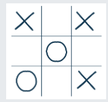
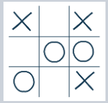
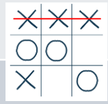




“When a player places three marks in a row, the player is declared the winner of the game”

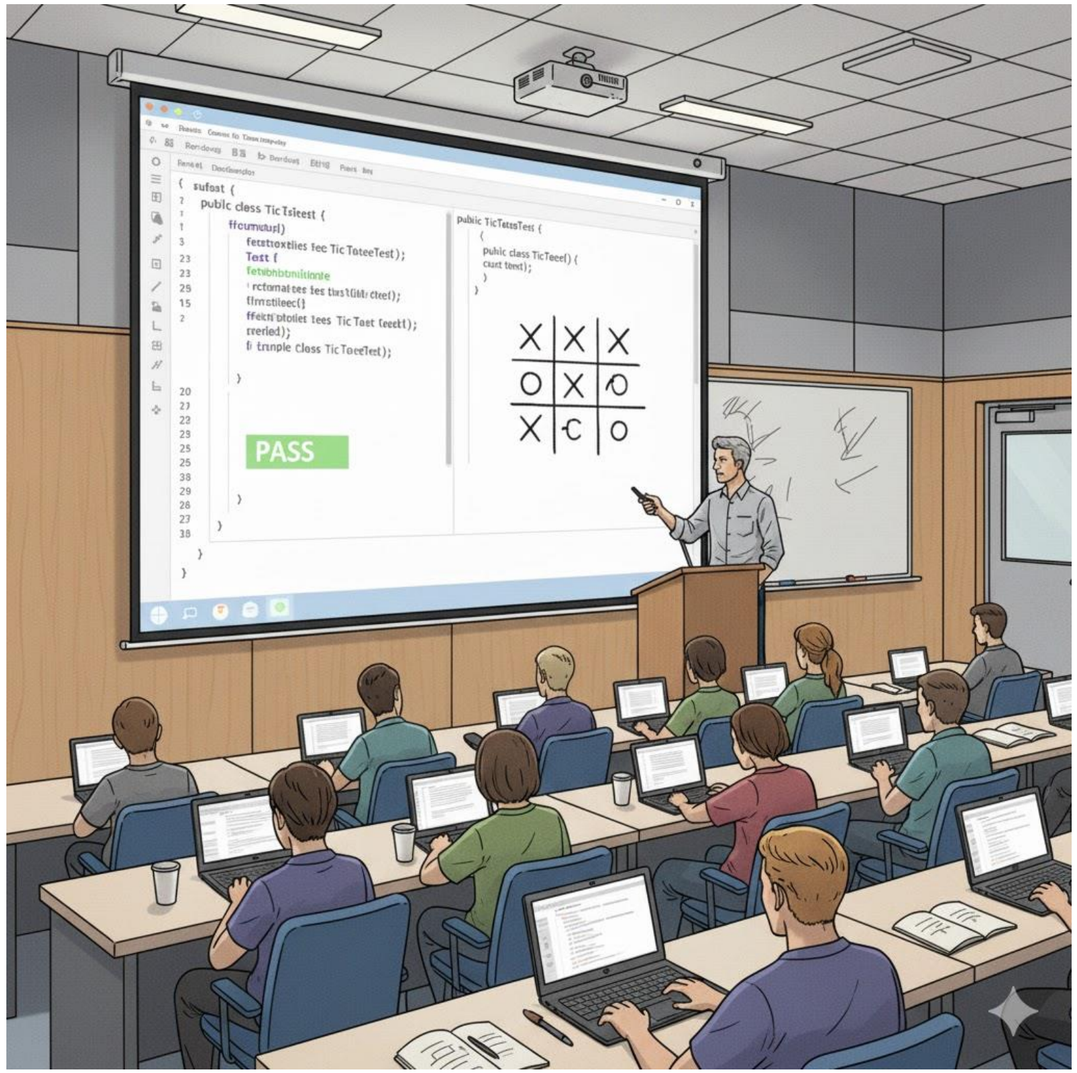
1. *game start → open window, display empty grid, display “Player X is your turn”*
2. *player X places a mark in the North-West cell → display “X” in NW cell, display “Player O is your turn” message*
3. *player O places a mark in the center cell → display “X” in NW cell, display “O” in CENTER cell, display “Player X is your turn”*
4. *player X places a mark in the South-East cell → display “X” in NW and SE, “O” in CENTER, display “Player O is your turn”*
5. *player O places a mark in the South-West cell → display “X” in NW and SE, “O” in CENTER and SW, display “Player X is your turn”*
6. *player X places a mark in the North-East cell → display “X” in NW, SE, and NE, “O” in CENTER and SW, display “Player O is your turn”*
7. *player O places a mark in the East cell → display “X” in NW, SE, and NE, “O” in CENTER , SW, and E, display “Player X is your turn”*
8. *player X places a mark in the North → display “X” in NW, SE, NE, and N, “O” in CENTER , SW, and E, strike from NW to NE, display “Player X won”*
9. *exit → close window*

“When a player places three marks in a row, the player is declared the winner of the game”

Action	Window	Grid	Message
Start	*open*	*empty*	*Player X is your turn*
X places a mark in North-West			*Player O is your turn*
O places a mark in Center			*Player X is your turn*
X places a mark in South-East			*Player O is your turn*
O places a mark in South-West			*Player X is your turn*
X places a mark in the North-East			*Player O is your turn*
O places a mark in the East			*Player X is your turn*
X places a mark in North			*Player X wins*
Exit	*close*		

An executable specification <-> acceptance test

Demo time



And now?

Acceptance test are at a higher level than unit tests, they are closer to the requirements

Unit tests *assure we are building the system in the right way*

Acceptance tests *assure we are building the right system*

Unit tests and acceptance tests are not exclusive they are complementary

Shall you write more acceptance tests or unit tests?

To know more

- *Specification by Example*
  https://less.works/less/technical-excellence/specification-by-example
  Focus on ATTD as collaborative requirements discovery approach

Self Assessment

Quiz 1: Acceptance tests

1. **According to the proposed ATDD cycle, what step immediately follows Automating the Acceptance Test?**
   A. *Make the acceptance test pass*
   B. *Refactor the code*
   C. *Pick a new requirement*
   D. *Write a failing unit test*
2. **Which statement best captures the primary goal of an Acceptance Test?**
   A. *To ensure every method in the system is implemented correctly*
   B. *To measure the system's speed and efficiency*
   C. *To assure we are building the system in the "right way"*
   D. *To assure we are building the "right system" by verifying requirements*

Quiz 1: Acceptance tests

3. **What does it imply that an Acceptance Test acts as an "executable specification"?**
   - A. *It is only written after the feature is complete*
   - B. *It exclusively tests the GUI interface*
   - C. *It provides an automated, verifiable definition of a specific requirement*
   - D. *It focuses only on minor coding details*
4. **In the broad-brush design , which component is responsible for managing the state of the game and enforcing the rules (e.g., handling player turns, checking for a winner)?**
   - A. *The Position enumeration*
   - B. *The TicTacToeView interface*
   - C. *The TicTacToeGame class*
   - D. *The acceptance test class*

Quiz 1: Acceptance tests

5. **In the broad-brush design , what is the sole responsibility of the View component (the TicTacToeView interface)?**
   A. *Checking if three marks are in a row*
   B. *Managing turns between player X and player O*
   C. *Storing the state of the marks on the grid*
   D. *Handling input/output, such as opening the window and displaying messages/grids*

Correct answers

Quiz 1:

1. *A*
2. *D*
3. *C*
4. *C*
5. *D*

Thank you!

esteco.com