Documentation for Tic-Tac-Toe Game in Dart done by Haya Alami:

Step 1: Variable Initialization

1. **Board Initialization**

```dart
List<String> board = List.filled(9, ' ');
```

   - `List<String> board = List.filled(9, ' ');`
   - Represents the Tic-Tac-Toe board as a list of 9 elements, initialized with empty spaces (`' '`).

2. **Player Turn**

```dart
bool player1Turn = true;
```

   - `bool player1Turn = true;`
   - tracking the turn :
     - `true` for Player X
     - `false` for Player O.

Step 2: Program Entry Point

1. **Main Function**

```dart
void main() {
  while (true) {
    printBoard();
    getMove();
    if (checkWin('X')) {
      print('Player X wins!');
      break;
    } else if (checkWin('O')) {
      print('Player O wins!');
      break;
    } else if (isBoardFull()) {
      print('It\'s a draw!');
      break;
    }
    player1Turn = !player1Turn;
  }
  print('Game over. Would you like to play again? (y/n)');
  if (stdin.readLineSync()?.toLowerCase() == 'y') {
    resetGame();
    main();
  }
}
```

- controlling the code sequence

2. **Game Loop**
   - Runs until a win, a draw, or the player chooses to stop.
   - Executes these steps in each iteration:
     - Print the current board (`printBoard()`).
     - Prompt the current player for a move (`getMove()`).
     - Check for a win or draw condition.
     - Toggle the turn (`player1Turn = !player1Turn`).

Step 3: Board Display

```
void printBoard() {
  print('-------------');
  for (int i = 0; i < 9; i += 3) {
    print('| ${board[i]} | ${board[i + 1]} | ${board[i + 2]} |');
    print('-------------');
  }
}
```

1. **Print Board**
   - Displays the current board state as a 3x3 grid with dividers.

Step 4: Handling Player Moves

1. **Get Move**

```
void getMove() {
  int move;
  do {
    print('Player ${player1Turn ? 'X' : 'O'}, enter a number (1-9):');
    move = int.tryParse(stdin.readLineSync() ?? '') ?? -1;
  } while (move < 1 || move > 9 || board[move - 1] != ' ');
  board[move - 1] = player1Turn ? 'X' : 'O';
}
```

   - Prompts the current player to enter a move (1-9).
   - Validates the input to ensure:
     - The input is between 1 and 9.

        - The selected position is not already occupied.
    - Marks the position with the current player's symbol (`X` or `O`).

Step 5: Win Condition

1. **Check Win**

```
bool checkWin(String player) {
  return (board[0] == player && board[1] == player && board[2] == player) ||
         (board[3] == player && board[4] == player && board[5] == player) ||
         (board[6] == player && board[7] == player && board[8] == player) ||
         (board[0] == player && board[3] == player && board[6] == player) ||
         (board[1] == player && board[4] == player && board[7] == player) ||
         (board[2] == player && board[5] == player && board[8] == player) ||
         (board[0] == player && board[4] == player && board[8] == player) ||
         (board[2] == player && board[4] == player && board[6] == player);
}
```

    - Checks if the specified player (`X` or `O`) has met any of the winning conditions:
        - Three matching symbols in a row, column, or diagonal.

Step 6: Draw Condition

1. **Is Board Full**

```
bool isBoardFull() {
  return !board.contains(' ');
}
```

    - Returns `true` if all positions on the board are occupied, meaning no further moves can be made.

Step 7: Reset Game

1. **Reset Game**

```
void resetGame() {
  board = List.filled(9, ' ');
  player1Turn = true;
}
```

- Resets the game state by:
  - Reinitializing the board with empty spaces.
  - Resetting the turn to Player X.

Step 8: Replay Prompt

1. **Replay Option**
   - After the game ends, prompts the user:

     ```
     Game over. Would you like to play again? (y/n)
     ```

   - If the user enters `y`, the game restarts by calling `main()` recursively.
   - If the user enters `n`, the program exits.

Output :

Player x wins the game :

```
-------------
| X | O | X |
-------------
| O |   | X |
-------------
| O |   |   |
-------------
Player X, enter a number (1-9):
9
Player X wins!
```

Restart the game :

```
Game over. Would you like to play again? (y/n)
y
------------
|  |  |  |
------------
|  |  |  |
------------
|  |  |  |
------------
Player X, enter a number (1-9):
```

Draw state :

```
-------------
| O | X |   |
-------------
| X | X | O |
-------------
| O | O | X |
-------------
Player X, enter a number (1-9):
3
It's a draw!
```