# 2048 Game Programming Documentation

**Overview**

The code implements a simplified version of the classic 2048 game using the Pygame library. The game is played on a 3x3 grid, where the player can move tiles in four directions (up, down, left, right) to combine the matching numbers and reach the target number 2048.

**Pygame Library:**

* pygame.init(): Initializes the Pygame library.
* pygame.display.set\_mode(): Creates the game window with a specified width and height.
* pygame.display.set\_caption(): Sets the title of the game window.

**Color Representation:**

* colors dictionary: Defines color codes for different numbers in the game grid.
* numbers dictionary: Defines color codes for the text displayed inside each grid cell.

**Game Class:**

The Game class encapsulates the logic of the 2048 game.

Methods:

* start\_game(): Initializes a new game grid.
* addnew2(mat): Adds a randomly generated number ‘2’ to an empty space in the matrix using ‘random’ module.
* reverse(mat): Returns a new matrix with elements reversed horizontally.
* transpose(mat): Returns the transpose of a matrix.
* merge(mat): Merges adjacent elements in each row of a matrix if they are equal and not equal to zero.
* compress(mat): Returns a new matrix with all non-zero elements shifted to the left.
* move\_up(grid), move\_down(grid), move\_left(grid), move\_right(grid): Move elements in the grid in the specified direction.
* get\_current\_state(mat): Checks the current state of the game board (WON, Game Not Over, Game Over).
* draw\_grid(mat): Draws the game grid on the screen.
* game\_over(): Displays a "GAME OVER" message on the screen and waits for the player to press any key
* no\_moves\_left(mat): Checks if there are no more possible moves.
* main(): Initializes the game, starts the game loop and handles user input.

**Random Module**

**random.randint(a, b):** Generates a random integer in the range **[a, b]**. Used in the **addnew2** method to select random positions for placing the number 2 in the game grid.

**Execution Flow**

1. The main() method initializes the game, starts the game loop, and draws the initial game grid.
2. Inside the game loop, user input is handled to move tiles on the grid.
3. The game state is continuously checked, and appropriate actions are taken based on the state (e.g., displaying "GAME OVER" or restarting the game).
4. The Pygame display is updated in each iteration.

**Running the Game**

The game can be executed by running the script. The main method (main()) is called if the script is run as the main module.

**Dependencies**

Pygame library must be installed (pip install pygame) 🡨 in the terminal if not installed.

**Conclusion**

The provided code offers a simplified implementation of the 2048 game, demonstrating the use of Pygame for graphics and user input handling. Further enhancements and optimizations can be made for a more polished and feature-rich game.