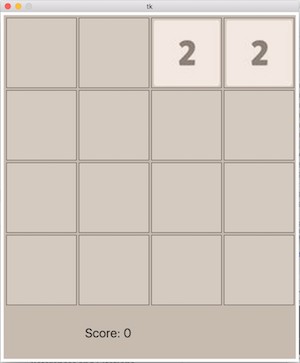
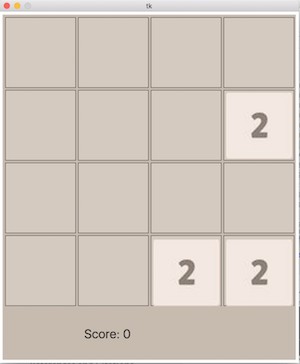
2048 Game Documentation

# Game Description

2048 is a game with a 4x4 grid with numbered tiles, to play 2048 use the arrow keys to move all the tiles in the direction of the arrow key pressed. When two tiles with the same number touch they merge into one and the new tile is their sum. The goal of the game is generate a tile with the number 2048. 2048 starts with a board with just two numbers randomly placed on it as depicted



After each move after all the tiles have been moved in the specified direction, a new number(either a 2 or a 4) is randomly placed on the board. So if we look at the board shown above, if a user were to press the down arrow key both the twos on the board would shift all the way down and a new number would now randomly appear as depicted



Also remember when two tiles with the same number collide when moving then they combine to make one tile with a value of the sum of both of the tiles. So for example using the board above if the user were to now press the left arrow key the board would now look as depicted



Note the both twos on the last row combined to make a 4 and the two on the second row also moved all the way to the left. Also note a new two was randomly placed on the board on the last row.

One more thing to note is that pressing the left key again would do nothing to the state of the board, since nothing can move to left based on the current state of the board, but you can still move right, up or down.

# Project Background

The files in this project includes an img folder, a gui.py file and a game.py file, **ensure that all these files are in the same directory in order for the game to work properly** and also ensure you are using Python 2.7.x, the code won’t work so well with Python 3. First, you will need to set up your machine to run the gui, the first step is to install the Python Library ’Pillow’ from PyPi. See below for the setup instructions based on your operating system:

## OS X or Linux

Open the terminal application and type the following command:

sudo pip install Pillow

Then you’ll be good to go!

## Windows

You will need to navigate to PyPa’s website and install the latest version of pip for windows [https://pip.pypa.io/en/](https://pip.pypa.io/en/stable/installing/) [stable/installing/](https://pip.pypa.io/en/stable/installing/). Then navigate to your command Prompt and execute the following command:

pip install Pillow

Then you’ll be good to go!

Now once you’ve finished the setup you should be able to run the game.py file and see an empty game board. You can now go on to use your Python expertise to help Shanice finish up her game.

Below contains a description and some examples on what was written.

Table 1: Documentation

|  |  |  |
| --- | --- | --- |
| Code | Description | Usage example |
| setup() | This function sets up the window frame and the game board that will be drawn for you. No GUI operations can hap- pen before this function is called. This function returns the frame ADT and the board ADT as a tuple. Through- out the code this frame or the board will be required as parameters in the functions. Wherever there is a refer- ence to the Window Frame ADT and the Game Board ADT it is referring to the two values returned from this func- tion call. Game Board has a special at- tribute called score that stores the value of the score, if updating the score up- date this value before trying to update the score on the GUI. | *>>>*import gui  *>>>*frame, board = gui.setup()  *>>>*board.score 436  *>>>* |
| start(game frame) | This function starts the game. It takes a Game Frame as it’s only parameter and then starts the game by displaying the board. All code related to setting up the board must happen before starting the game. | *>>>*import gui  *>>>*frame, board = gui.setup()  *>>>*gui.start(frame) |
| random number() | This function returns a random Num- berTile, a NumberTile will be used to represent a tile on the board and has two attributes a value and an image. To access the image or value of a number you need to store it in a variable and then use the ’.’ notation. | *>>>*import gui  *>>>*x = gui.random number()  *>>>*x.value 2 |

|  |  |  |
| --- | --- | --- |
| put(game board, key, number, row, col- umn) | This function takes 5 parameters as in- put: the first is the Game Board, the second is a unique key as a String to identify the NumberTile being placed on the board, the third is NumberTile to be placed on the board and then the fi- nal two parameters are integers row and column that specify the row and column where the tile should go on the board. The function also returns True if it suc- cessfully placed the tile on the board. The Game Board also has a special at- tribute called numbers which is a dictio- nary mapping the unique identifier to a tuple containing the row and column of that tile. This function adds to that dic- tionary mapping and maps the unique String identifier passed to it to a tuple containing the row and column the num- ber was placed at on the board. | *>>>*import gui  *>>>*frame,board = gui.setup()  *>>>*gui.put(board, ”1”,  gui.random number(), 0, 2) True  *>>>*board.numbers  *{* ”Tile number 1”: (0,2) *}*  *>>>* |
| find position(game board, key) | This function takes two parameters as input: the first is the Game Board and the second is the unique String identifier for the NumberTile you want to find the position of. | *>>>*import gui  *>>>*frame,board = gui.setup()  *>>>*gui.put(board, ”1”,  gui.random number(), 0, 2) True  *>>>*gui.find position(board, ”1”)  (0,2)  *>>>* |
| remove number(game board, key) | This function takes two parameters as input: the first is the Game Board and the second is the unique String identifier for the NumberTile you want to remove from the board. | *>>>*import gui  *>>>*frame,board = gui.setup()  *>>>*gui.remove number(board, ”id1”) |
| find number(number) | This function takes one parameter, an Integer and returns it’s NumberTile rep- resentation. The possible numbers are 2,4,8,16,32,64,128,256,512,1024 and of-  course 2048. | *>>>*import gui  *>>>*x = gui.find number(8)  *>>>*x.value 8 |

|  |  |  |
| --- | --- | --- |
| move number(game board, key, direc- tion, update grid, move by distance) | This function takes 5 parameters as in- put and moves a tile by one slot in the specified direction: the first is the Game Board, the second is a unique key as a String to identify the NumberTile being placed on the board, the third is an In- teger representing the direction to move the tile and then the final two parame- ters are two functions: the first to up- date the state of the grid, this should be a function that takes 3 parameters and the other should be a function that takes 5 parameters and actually animates the movement of a tile across the board, these will be explained in more detail later. The function returns True if it was able to successfully move the Tile, False otherwise | *>>>*import gui  *>>>*frame,board = gui.setup()  *>>>*gui.move number(board, ”1”,1, some function, some function)  False  *>>>* |
| update score(game board) | This function updates the score dis- played on the GUI at the bottom of the board. It takes one parameter which is the Game Board | *>>>*import gui  *>>>*frame, board = gui.setup()  *>>>*board.score = 1000  *>>>*gui.update score(board) |
| move tile(game board, key, horizontal, vertical) | This function updates the NumberTile on the GUI by moving it horizontally and vertically by the distance specified. The function should take 4 parameters: a Game Board, the unique identifier as a String, the distance to move the tile horizontally as an integer and the dis- tance to move the tile vertically as an integer. **Note: a negative value for the horizontal parameter will move the tile to the left, while a positive will move it to the right and a neg- ative value for the vertical param- eter will move the tile upwards, while a positive will move it down- wards.** | *>>>*import gui  *>>>*frame, board = gui.setup()  *>>>*gui.move tile(board, ”1”, -20, 0)  *>>>*  The example above would move the Tile identified by ”1” to the left by 20  pixels on the GUI |
| game over(game board, winner) | This function takes two parameters a Game Board and a boolean value. The boolean value should be True if game is over and the user reached 2048 and should be False otherwise. The function updates the GUI to display an appropri- ate message to the user. | *>>>*import gui  *>>>*frame, board = gui.setup()  *>>>*gui.game over(game board, True) |