# Word Search Game Code Documentation

## 1. Initial Setup and Imports

import tkinter as tk  
from tkinter import ttk, messagebox, simpledialog  
from PIL import Image, ImageTk, ImageSequence  
import random  
import string  
import pygame  
import os

**Notes:**

* The game uses Tkinter for the GUI
* PIL (Python Imaging Library) is used for handling images and animations
* Pygame is used for sound effects
* Other standard libraries for general functionality

## 2. Global Variables

root = None  
word\_pressed = ''  
previous = [0, 0]  
route = [0, 0]  
current\_words = []  
found\_words = []  
grid = []  
buttons = []  
# ... other globals

**Notes:**

* Tracks game state and UI elements
* Stores the game grid, selected words, and scores
* Manages game settings like difficulty and time limits

## 3. Core Game Functions

### Loading Categories

def load\_categories():  
 """Load word categories from files"""

**Notes:**

* Reads word lists from text files
* Categories include: animals, fruits, places, sports, colors
* Words are converted to uppercase for consistency

### Grid Creation and Word Placement

def place\_word(word):  
 """Place a word in the grid"""  
def can\_place\_word(word, x, y, direction):  
 """Check if word can be placed at position"""  
def fill\_word(word, x, y, direction):  
 """Fill word in grid"""

**Notes:**

* Handles random word placement in grid
* Checks for valid placement in 8 directions
* Ensures words don't overlap incorrectly

### Word Selection and Checking

def button\_press(x, y):  
 """Handle button press event"""  
def check\_word():  
 """Check if selected word is valid"""  
def colour\_word(valid):  
 """Color the selected word"""

**Notes:**

* Manages user word selection through button clicks
* Validates selected words against word list
* Updates UI colors for correct/incorrect selections

## 4. Game Modes and AI

### VS AI Mode

def computer\_turn():  
 """AI opponent's turn"""

**Notes:**

* Implements simple AI opponent
* AI randomly selects unfound words
* Updates scores and UI for AI moves

### Difficulty Settings

* Mini: 8x8 grid, 4 words, 3 attempts per word
* Normal: 10x10 grid, 6 words, 2 attempts per word
* Pro: 12x12 grid, 8 words, 1 attempt per word
* Pro Max: 15x15 grid, 10 words, 1 attempt per word

## 5. UI Components

### Welcome Screen

def create\_welcome\_screen():  
 """Create the welcome/setup screen"""

**Notes:**

* Player name input
* Game level selection
* Category selection
* VS AI mode toggle

### Game Screen

def create\_game\_screen():  
 """Create the main game screen"""

**Notes:**

* Creates main game grid
* Displays word list
* Shows timer and score
* Includes game controls

## 6. Save/Load System

### Save Game

def save\_game():  
 """Save current game state to txt file"""

**Notes:**

* Saves all game state to text file
* Includes player info, scores, grid state
* Preserves word locations and found status

### Load Game

def load\_saved\_game():  
 """Load game state from txt file"""

**Notes:**

* Restores complete game state
* Validates file format
* Rebuilds UI with saved state

## 7. Animation and Sound

### End Screen

def show\_end\_screen(won):  
 """Show the animated end screen"""  
def update\_gif(label, frames, frame\_index, delay):  
 """Update the GIF animation frame"""

**Notes:**

* Displays victory/defeat animations
* Shows final scores
* Provides options to play again or exit

### Sound Effects

* Uses pygame for sound playback
* Includes victory and defeat sounds
* Triggered on word finding and game end

## 8. Game Flow Control

### Timer System

def update\_timer():  
 """Update the game timer"""

**Notes:**

* Countdown timer based on difficulty
* Updates display every second
* Triggers game over when time runs out

### Game State Management

def game\_over(won=False):  
 """Handle game over state"""  
def reset\_game():  
 """Reset the game state"""

**Notes:**

* Handles win/lose conditions
* Resets all game variables
* Manages transitions between states

## Best Practices for Team Members

1. Always use the global keyword when modifying global variables
2. Test word placement thoroughly when modifying grid logic
3. Handle file operations with try-except blocks
4. Maintain consistent UI styling across components
5. Document any changes to game mechanics or scoring

## Common Debugging Areas

1. Word placement validation
2. Score calculation
3. Timer synchronization
4. Save/load state preservation
5. AI opponent behavior