# Word Search Game GUI Documentation

## 1. Main Window Structure

root = tk.Tk()  
root.title("Word Search Game")

**Notes:**

* The game uses a single main window (root)
* Contains three main sections: header, content area, and footer
* All other GUI elements are contained within these sections

## 2. Header Section

def create\_header():  
 header\_frame = tk.Frame(root)  
 header\_frame.pack(fill=tk.X, side=tk.TOP)  
   
 tk.Label(header\_frame, text='Word Search Game',  
 font=('Helvetica', 23, 'bold'),  
 fg='blue').pack(expand=True, fill=tk.X, pady=12)

**Notes:**

* Fixed at the top of the window
* Contains game title in blue, bold text
* Uses Helvetica font for better readability
* Spans full width of window (fill=tk.X)

## 3. Welcome Screen

def create\_welcome\_screen():  
 welcome\_frame = tk.Frame(root, bg='#f0f0f0')  
 welcome\_frame.pack(pady=56, padx=180)

### Components:

1. **Player Name Entry**

* tk.Label(welcome\_frame, text="Name")  
  name\_entry = tk.Entry(welcome\_frame)
  + Text entry field for player name
  + Required field with validation

1. **Game Level Selection**

* difficulty\_var = tk.StringVar(value="Normal")  
  ttk.Combobox(welcome\_frame, textvariable=difficulty\_var,  
   values=['Mini', 'Normal', 'Pro', 'Pro Max'])
  + Dropdown menu for difficulty selection
  + Affects grid size and game parameters

1. **Category Selection**

* category\_var = tk.StringVar(value=list(categories.keys())[0])  
  ttk.Combobox(welcome\_frame, textvariable=category\_var,  
   values=list(categories.keys()))
  + Dropdown for word category selection
  + Options: animals, fruits, places, sports, colors

1. **VS AI Toggle**

* vs\_ai\_var = tk.BooleanVar()  
  tk.Checkbutton(welcome\_frame, text="VS AI Mode",  
   variable=vs\_ai\_var)
  + Checkbox to enable/disable AI opponent

1. **Control Buttons**

* tk.Button(button\_frame, text="New Game")  
  tk.Button(button\_frame, text="Load Game")
  + Start new game or load saved game
  + Positioned at bottom of welcome screen

## 4. Main Game Screen

def create\_game\_screen():  
 frame1 = tk.Frame(root, bg="white") # Grid frame  
 frame2 = tk.Frame(root, bg='white') # Info frame  
 frame3 = tk.Frame(root) # Additional frame

### A. Grid Area (frame1)

for x in range(size):  
 for y in range(size):  
 button = tk.Button(frame1,  
 text=grid[x][y],  
 bg='#255059',  
 fg='white')

**Notes:**

* Dynamic grid of buttons based on difficulty
* Each button contains one letter
* Color coding for selected/found words:
  + Default: Dark teal (#255059)
  + Selected: Yellow
  + Found by player: Blue (#535edb)
  + Found by AI: Red (#ff9999)

### B. Information Panel (frame2)

1. **Word List**

* for word in current\_words:  
   label = tk.Label(frame2, text=word,  
   font=('Helvetica', 12, 'bold'),  
   fg='#254359', bg='#cbe5f7')
  + Displays target words
  + Found words become invisible

1. **Game Statistics**

* timer\_label = tk.Label(frame2, text="Time Left: 03:00")  
  attempts\_label = tk.Label(frame2, text="Attempts Left: 5")  
  score\_label = tk.Label(frame2, text="Score: 0")
  + Shows remaining time
  + Displays attempts left
  + Shows current score

1. **Control Buttons**

* tk.Button(frame2, text="Check Word")  
  tk.Button(frame2, text="Save Game")  
  tk.Button(frame2, text="Exit Game")
  + Controls for game actions
  + Consistently styled buttons

## 5. End Game Screen

def show\_end\_screen(won):  
 end\_window = tk.Toplevel(root)

**Components:**

1. **Animation**

* gif\_label = tk.Label(end\_window)  
  update\_gif(gif\_label, frames, frame\_index, delay)
  + Displays victory/defeat animation
  + Centered in window

1. **Score Display**

* tk.Label(scores\_frame, text=f"Final Score: {score}")
  + Shows final scores
  + Different layouts for VS AI mode

1. **Control Buttons**

* ttk.Button(buttons\_frame, text="Play Again")  
  ttk.Button(buttons\_frame, text="Exit Game")
  + Options to continue or quit

## 6. Save/Load Dialog

filename = simpledialog.askstring("Save Game",   
 "Enter filename to save:")

**Notes:**

* Modal dialog for file operations
* Input validation for filenames
* Error handling with message boxes

## 7. Message Boxes

messagebox.showinfo() # Information messages  
messagebox.showerror() # Error messages  
messagebox.askyesno() # Confirmation dialogs

**Used for:**

* Error notifications
* Success confirmations
* Exit confirmations
* Save game prompts

## 8. Styling Guidelines

1. **Colors**
   * Background: White (#ffffff)
   * Text: Dark blue (#254359)
   * Buttons: Various blues and teals
   * Highlight: Yellow for selection
2. **Fonts**
   * Main font: Helvetica
   * Size ranges: 12-23px
   * Bold for important elements
3. **Spacing**
   * Consistent padding (padx, pady)
   * Grid spacing of 2 pixels
   * Margins for visual separation

## 9. Responsive Considerations

1. **Window Sizing**

* window\_width = 400  
  window\_height = 700  
  x = (screen\_width - window\_width) // 2  
  y = (screen\_height - window\_height) // 2
  + Centered on screen
  + Adjusts based on content

1. **Grid Scaling**
   * Adapts to different difficulty levels
   * Maintains button proportions

## Best Practices for GUI Development

1. Always destroy old frames before creating new ones
2. Use consistent styling across all components
3. Handle window resizing gracefully
4. Implement proper error handling with user feedback
5. Maintain clear visual hierarchy
6. Ensure all interactive elements are clearly visible
7. Provide feedback for all user actions