typing_game.py

```
import random
import time
import Tkinter as tk

# Resources: https://www.youtube.com/watch?v=IYHJRnVOFlw
# Written in Python 2.7 (Does not work in 3+)


class TypingGame:

    def __init__(self, master):
        frame = tk.Frame(master)
        frame.pack()

        # Start button
        self.start_button = tk.Button(frame, text="Start", command=self.run_game)
        self.start_button.pack()

        # Timer label
        self.timer_label = tk.Label(frame, text="")
        self.timer_label.pack()

        # Current word label
        self.current_word_label = tk.Label(frame, text="")
        self.current_word_label.pack()

        # Entry text box
        self.entry = tk.Entry(frame)
        self.entry.bind("<Return>", self.next_word)
        self.entry.pack()
        self.entry.focus_set()

        # Resets program
        self.reset_button = tk.Button(frame, text="Reset", command=self.reset)

        # Instance variables
        self.count = 0
        self.final_list = list()
        self.entered_word = ""
        self.curr_time = 30
        self.wpm = None
        self.char_count = 0

    # Reads and returns each line from 'words_alpha.txt'
    @staticmethod
    def grab_words():
        print("Reading words_alpha.txt...")

        f = open('words_alpha.txt', "r")
        lines = f.readlines()
        f.close()

```

```
53          return lines
54
55      # Restarts program to run again
56      def reset(self):
57          self.count = 0
58          self.final_list = list()
59          self.entered_word = ""
60          self.curr_time = 30
61          self.wpm = None
62          self.char_count = 0
63
64          self.current_word_label.configure(text="")
65          self.timer_label.configure(text="")
66          self.entry.delete(0, tk.END)
67
68          self.reset_button.pack_forget()
69          self.start_button.pack()
70          self.entry.pack()
71          self.entry.focus_set()
72
73      # Chooses 50 random words from the entire list
74      @staticmethod
75      def choose_words(w_list):
76          print("Choosing words...")
77          c_words = list()
78
79          for i in range(50):
80              key = random.randint(1, len(w_list) + 1)
81              if len(w_list[key]) > 5:
82                  c_words.append(w_list[key])
83          print(c_words)
84
85          return c_words
86
87      # Removes excess characters for user-friendliness
88      @staticmethod
89      def clean_words(w_list):
90          print("Cleaning text...")
91
92          to_filter = list()
93
94          for word in w_list:
95              new_word = word[:-2]
96              to_filter.append(new_word)
97
98          return to_filter
99
100     # Parent function
101     def run_game(self):
102         print("Started...")
103
104         # Sets 'word_list' to the list of words from 'words_alpha.txt'
105         word_list = self.grab_words()
106         # Sets 'chosen_words' to the list of 50 cleaned, selected words
107         chosen_words = self.clean_words(self.choose_words(word_list))
108
```

```
        print("Ready.")

        # Debug in terminal
        print(chosen_words)
        print(len(chosen_words))

        # Transfers 'chosen_words' to instance variable: 'final_list'
        self.final_list = chosen_words
        self.update_word()
        self.timer()

        self.start_button.pack_forget()

    # Updates 'current_word_label' when needed and deletes whatever is in the entry box
    def update_word(self):
        self.current_word_label.configure(text=self.final_list[self.count])
        self.entry.delete(0, tk.END)

    def add_char(self):
        self.char_count += len(self.final_list[self.count])
        print("Char count for", self.final_list[self.count], " is ", len(self.final_list[self.count]))

    # Cycles through 'final_list' to get the next word
    def next_word(self, foo):
        self.entered_word = self.entry.get()

        # If the user spells the word correctly;
        if self.entered_word == self.final_list[self.count]:
            print("Correct spelling of:", self.final_list[self.count])
            # Add character amount to total
            self.add_char()
            # Increment count to next word
            self.count += 1
            self.update_word()
        else:
            print("Incorrect spelling of:", self.final_list[self.count])

    # Runs timer
    def timer(self):
        # Stops timer if 'curr_time' is 0 or the count has reached 50
        if self.curr_time <= 0 or self.count == 50:
            self.update_wpm()
            self.timer_label.configure(text=self.wpm)

            self.reset_button.pack()
            self.entry.pack_forget()

            print("Word Count", self.count)
            print("WPM", self.wpm)
        else:
            self.timer_label.configure(text=self.curr_time)
            self.curr_time -= 1

            # Recursively calls 'timer' function after 1 second
            self.timer_label.after(1000, self.timer)

    def update_wpm(self):
```

```
        self.wpm = str(round((self.char_count / 5) / (30 / 60.0), 2)) + " WPM"
        print("Char count", self.char_count)


root = tk.Tk()
t = TypingGame(root)
root.mainloop()
```
