c0b2108596 / ProjExD Public

Code Issues 2 Pull requests Actions Projects Wiki ...

main ...

ProjExD / ex06 / typing.py / Jump to

c0b2108596 実装完了 History

1 contributor

133 lines (109 sloc) | 4.86 KB ...

```python
import tkinter as tk
from tkinter import messagebox
import sys
import time
import threading
import random

QUESTION = ["tkinter", "geometry", "widgets", "messagebox", "configure",
            "label", "column", "rowspan", "grid", "init"]

class Application(tk.Frame):
    def __init__(self, master):
        global Hp
        super().__init__(master)
        self.pack()

        master.geometry("500x400")
        master.title("こうかとんに負けるな！")

        # 問題数インデックス
        self.index = 0

        # 正解数カウント用
        self.correct_cnt = 0

        self.create_widgets()

        # 経過時間スレッドの開始
        t = threading.Thread(target=self.timer)
        t.start()

        # Tkインスタンスに対してキーイベント処理を実装
        self.master.bind("<KeyPress>", self.type_event)

    # ウィジェットの生成と配置
    def create_widgets(self):
        global Hp
        #お題の配置
```

```
        self.q_label = tk.Label(self, text="お題：", font=("",20))
        self.q_label.grid(row=0, column=0)
        self.q_label2 = tk.Label(self, text=QUESTION[self.index], width=10, anchor="w", font=("",20))
        self.q_label2.grid(row=0, column=1)
        #解答の配置
        self.ans_label = tk.Label(self, text="解答：", font=("",20))
        self.ans_label.grid(row=1, column=0)
        self.ans_label2 = tk.Label(self, text="", width=10, anchor="w", font=("",20))
        self.ans_label2.grid(row=1, column=1)
        #HPの配置
        self.hp_label = tk.Label(self, text=f"HP：{Hp}", width=10, anchor="w", font=("", 20))
        self.hp_label.grid(row=2, column=10)
        self.result_label = tk.Label(self, text="", font=("",20))
        self.result_label.grid(row=2, column=0, columnspan=2)

        # # 時間計測用のラベル
        self.time_label = tk.Label(self, text="", font=("",20))
        self.time_label.grid(row=3, column=0, columnspan=2)

        self.flg2 = True

    # キー入力時のイベント処理
    def type_event(self, event):
        global Hp
        # 入力値がEnterの場合は答え合わせ
        if event.keysym == "Return":
            if self.q_label2["text"] == self.ans_label2["text"]:
                self.result_label.configure(text="正解！", fg="red")
                self.correct_cnt += 1
            else:
                Hp -= int(random.randint(20, 40))
                self.hp_label = tk.Label(self, text=f"HP：{Hp}", width=10, anchor="w", font=("", 20))
                self.hp_label.grid(row=2, column=10)
                self.result_label.configure(text="残念！", fg="blue")

            # 解答欄をクリア
            self.ans_label2.configure(text="")

            # 次の問題を出題
            self.index += 1

            #クリアの場合
            if self.index == len(QUESTION):
                self.flg = False
                self.q_label2.configure(text="終了！")
                messagebox.showinfo("you win!", f"あなたはこうかとんに勝ちました。\nあなたのスコアは{self.corre
                sys.exit(0)
            self.q_label2.configure(text=QUESTION[self.index])

            #失敗した時
            if Hp <= 0:
                self.flg = False
                self.q_label2.configure(text="終了！")
                messagebox.showinfo("you lose!", f"あなたはこうかとんに負けました。\nあなたのスコアは{self.corr
                sys.exit(0)

        elif event.keysym == "BackSpace":
            text = self.ans_label2["text"]
```

```
            self.ans_label2["text"] = text[:-1]

        else:
            # 入力値がEnter以外の場合は文字入力としてラベルに追記する
            self.ans_label2["text"] += event.keysym

    #経過時間の設定
    def timer(self):
        global Hp
        self.second = 0
        self.flg = True
        while self.flg:
            self.second += 1
            #10秒毎に10ダメージ受ける機能の実装
            if self.second % 10 == 0:
                Hp -= 10
                self.hp_label = tk.Label(self, text=f"HP：{Hp}", width=10, anchor="w", font=("", 20))
                self.hp_label.grid(row=2, column=10)
            self.time_label.configure(text=f"経過時間：{self.second}秒")
            time.sleep(1)


if __name__ == "__main__":
    root = tk.Tk()
    Hp = 100 #初期HPの設定
    Application(master=root)

    #キャンバス作成
    canv = tk.Canvas(root, width=300, height=400, bg="gray")
    canv.pack()

    #こうかとんの表示
    tori = tk.PhotoImage(file="fig/0.png")
    cx, cy = 150, 150
    canv.create_image(cx, cy, image=tori, tag="tori")

    root.mainloop()
```