Hangman game - python

Палач(hangman) е класическа детска игра: целта е да познаеш дадена дума като знаеш първата и последната буква и имаш няколко опита. Всички сме играли на “бесеница” като малки, дали за да научим някоя дума или просто да се забавляваме. Това е образователна игра, чрез която децата могат да се научат да мислят, но може би представена не в детски вариант, защото буквално рисуваш умиращо по ужасен начин човече. Поне някои малки деца не разбират ужасяващата реалност зад играта, а просто играят за забавление и измислят стратегии. Затова днес можем да представим нашата версия на “бесеница”, програмирана на python.

Начин на работа

Най-напред вмъкваме tkinter, което е “обвързване” на python към Tk GUI инструментариума. Това е стандартния GUI на python.

```
from tkinter import *
from tkinter import messagebox
import os
from generate_word import word
```

Както при повечето други съвременни Tk обвързвания, Tkinter се реализира като обвивка на Python около пълен Tcl интерпретатор, вграден в интерпретатора на Python . Извикванията на Tkinter се превеждат в Tcl команди, които се подават на този вграден интерпретатор, което прави възможно смесването на Python и Tcl в едно приложение.

Използваме го и за да отворим отделен прозорец където да бъде визуализирана играта:

```
window = Tk()
window.geometry(WINDOW_SIZE)
window.title('Hangman')
window.config(bg = WINDOW_BG)
#Създаваме допълнителен прозорец и му даваме
съответните стойности, които сме дефинирали в
следващото парче код(всъщност те са дефинирани преди
създаването на прозорец)
```

Още преди да отворим допълнителен прозорец, дефинираме глобално променливите:

```
score = 0
count = 0
win_count = 2
WINDOW_BG = '#e5404e'
WINDOW_SIZE = '1200x870+300+80'
FONT = ('Arial', 40)
```

Всеки бутон, буквата за която отговаря бутона и позицията му са съхранени в лист.

```
letters =
['a','b','c','d','e','f','g','h','i','j','k','l','m'
,'n','o','p','q','r','s','t','u','v','w','x','y','z'
]
buttons =
[['b1','a',80,740],['b2','b',160,740],['b3','c',240,
740],['b4','d',320,740],['b5','e',400,740],['b6','f'
,480,740],['b7','g',560,740],['b8','h',640,740],['b9
','i',720,740],['b10','j',800,740],['b11','k',880,74
0],['b12','l',960,740],['b13','m',1040,740],['b14','
n',80,800],['b15','o',160,800],['b16','p',240,800],[
'b17','q',320,800],['b18','r',400,800],['b19','s',48
0,800],['b20','t',560,800],['b21','u',640,800],['b22
','v',720,800],['b23','w',800,800],['b24','x',880,80
0],['b25','y',960,800],['b26','z',1040,800]]
#Имаме лист от букви и след това листов обект за
всички бутони отговарящи на буквите
```

След това генерираме всички етикети за думите:

```
    label = Label(window, text=" ", bg=WINDOW_BG,
font=FONT)
    label.pack(padx=40, pady=(500, 100), side=LEFT)

    label1 = Label(window, text=word[0],
bg=WINDOW_BG, font=FONT)
    label1.pack(padx=41, pady=(500, 100), side=LEFT)

    x = 21
    for var in range(1, len(word)-1):
```

```
exec('label{}=Label(window,text="_",bg=WINDOW_BG,fon
t=FONT)'.format(var))
        exec('label{}.pack(padx = {}, pady =
(500,100), side=LEFT)'.format(var, x))
        x += 1

    exec('label{} = Label(window, text = "{}", bg =
WINDOW_BG, font = FONT)'.format(
        len(word), word[-1]))
    exec('label{}.pack( padx = {},pady =
(500,100),side = LEFT)'.format(len(word), x+1))
```

На снимката по горе виждаме как биват генерирани празните полета за думата.

Генерираме снимки и етикети за графичното изобразяване на бесилото и човека. Използваме библиотеката os и взимаме директния път на app.py. Към пътя на app.py прибавяме папката и името на файловете за да се направят снимките и съответно лейбълите.

```
    path = os.getcwd()
    print(path)
    # hangman - generating imagaes and labels
    h0 = PhotoImage(file=path + "\Images\h0.png")
    h1 = PhotoImage(file=path + "\Images\h1.png")
    h2 = PhotoImage(file=path + "\Images\h2.png")
    h3 = PhotoImage(file=path + "\Images\h3.png")
    h4 = PhotoImage(file=path + "\Images\h4.png")
    h5 = PhotoImage(file=path + "\Images\h5.png")
    h6 = PhotoImage(file=path + "\Images\h6.png")

    labelh0 = Label(window, bg=WINDOW_BG, image=h0)
    labelh1 = Label(window, bg=WINDOW_BG, image=h1)
    labelh2 = Label(window, bg=WINDOW_BG, image=h2)
    labelh3 = Label(window, bg=WINDOW_BG, image=h3)
    labelh4 = Label(window, bg=WINDOW_BG, image=h4)
    labelh5 = Label(window, bg=WINDOW_BG, image=h5)
    labelh6 = Label(window, bg=WINDOW_BG, image=h6)
```

```
    labelh0.place(x=620, y=0)

    for var in letters:
        exec(f'{var}=PhotoImage(file="Images/{var}.png")')
    for var in buttons:
        exec(f'{var[0]}=Button(window,bd=0,command=lambda:
game_brain("{var[0]}","{var[1]}"),bg =
WINDOW_BG,font=FONT,image={var[1]})')
        exec('{}.place(x={},y={})'.format(var[0], var[2], var[3]))
#Алгоритъма за поставянето на снимките
```

За да се стигне до алгоритъма на играта, то той е ламбда функция,вземаща за параметри името на бутона и буквата. Функцията бива извикната когато един бутон е натиснат.

Алгоритъм на играта

```
def game_brain(button, letter):
    global count, win_count, score
    exec('{}.destroy()'.format(button))
    if letter in word[1:len(word)]:
        for i in range(1, len(word)-1):
            if word[i] == letter:
                win_count += 1
                mytext = letter
                exec('label{}.config(text=mytext)'.format(i))
        if win_count == len(word):
            score += 1
            messagebox.showinfo('GOOD JOB!', 'YOU WON!\n GOODBYE!')
            window.destroy()
        i = 1

    else:
        count += 1
        exec('labelh{}.destroy()'.format(count-1))
        exec('labelh{}.place(x={},y={})'.format(count, 620, 0))
        if count == 6:
            messagebox.showinfo('GAME OVER!', 'YOU LOST!\nGOODBYE!')
            window.destroy()
```

Начинът на работа е много прост. Когато бутонът бива натиснат, то той извиква ламбда функцията, която изтрива бутона. Ако буквата

се съдържа в думата, то функцията минава през всички букви на думата и когато стигне до някоя която съвпада, то тя променя лейбъла на съответстващата позиция на буквата в думата и “разкрива” познатата буква. Ако сме познали глобалната променлива win_count се увеличава с едно и когато достигне до броя на буквите, които трябва да познаем, играта приключва и играчът печели. Ако не сме познали, то броят на грешките ни се увеличава с 1 и когато достигне до 6 играчът губи.

Exit бутон

```
def EXIT():
    answer = messagebox.askyesno('ALERT', 'Do you want to exit the
game?')
    if answer == True:
        window.destroy()


e1 = PhotoImage(file='Images/exit.png')
ex = Button(window, bd=0, command=EXIT, bg=WINDOW_BG, font=FONT,
image=e1)
ex.place(x=1050, y=20)



window.mainloop()
```

generate_word.py

```
from data import words
import random

word = []
word_cpy = random.choice(words)

for _ in word_cpy:
```

```
    word.append(_)

word = tuple(word)


#Има файл data, в който се съдържа лист words с над
200 думи
```

Бива избрана една дума на случаен принцип от предварително въведените ни думи във файл на име data.py. Всяка от буквите на буквата бива добавена в празен tuple, за да може да не бъде променяна и да се сведе възможността за грешки до минимум.

Благодарение на pyinstaller app.py е конвертиран във .exe файл. За да бъде пусната играта, трябва да се натисне само app.exe файла.

Автори:
Димитър Желев 9г №9
Иоан Евгениев 9г №13