Memory

Als nächstes Projekt mit Python kannst du dein eigenes Memory möglichst selbstständig programmieren. Hilfestellung bekommst du dabei in erster Linie durch die folgenden Beispiele und der Recherche in Online-Tutorials. Eine kleine Übersicht guter Tutorials findest du am Ende.

Aufgabenstellung

Programmiere dein eigenes Memory mit grafischer Oberfläche. Das Spielfeld sollte aus 4 Paaren von Karten bestehen. Der Spieler soll auf eine Karte klicken können, woraufhin diese aufgedeckt wird. Sollte bereits eine Karte aufgedeckt worden sein, dann muss geprüft werden ob es ein Paar ist. Wenn ja, dann bleiben beide Karten aufgedeckt und der Spieler kann fortfahren, wenn nicht, dann werden die beiden Karten wieder umgedreht. Zu Beginn wird es nur einen Spieler ohne Punktestand geben.

Ein Grundgerüst des Programms inklusive Bilder für die Karten steht dir in Form eines ZIP-Archives bereit.

Hilfestellung zum Memory

Bilder als Buttons

Um Bilder in einem Button einzubinden, muss zunächst einmal ein Objekt der Klasse PhotoImage (in der *Tkinter*-Blibliothek enthalten) erzeugt werden:

```
bild  = PhotoImage(file='Pfad zur GIF-Datei')
```

Wichtig ist, dass es sich bei Python 2.7 um eine GIF-Datei handelt. Andere Dateiformate werden erst ab Python 3.6 unterstützt.
Anschließend kann das Bild dem Button mittels image Attribut zugewiesen werden:

```
Button(..., image=bild, ...)
```

Hier ein ganzes Beispiel:

```
fenster = Tk()
bild  = PhotoImage(file='button.gif')
button = Button(fenster, image=bild)
button.pack()
fenster.mainloop()
```

Verzögertes Ausführen einer Funktion

Angenommen du möchtest nach 5 Sekunden etwas ausführen lassen, dafür bietet das *Tk*-Objekt (also das Fenster) die *after*-Methode an:

```
fenster.after(ZEIT, FUNKTION, PARAMETER …)
```

Als erstes muss immer die Wartezeit in Millisekunden übergeben werden, anschließend der Name der Funktion die ausgeführ werden soll (also ohne Klammern und Parameter) und letztlich alle Parameter, die der Funktion übergeben werden sollen.

Möchtest du zum Beispiel deine eigene Funktion *karte_umdrehen(karte)* nach 5 Sekunden ausführen lassen, so müsste der Quellcode wie folgt lauten:

```
fenster.after(500, karte_umdrehen, karte)
```

Button soll auf Drücken eine Funktion mit Parametern aufrufen

Mit dem command-Attribut kann bei einem Button eine Methode angegeben werden, die ausgeführt wird, sobald der Button gedrückt wird. Hierbei ist es aber nicht direkt möglich, der Methode auch Parameter zu übergeben. Dafür benötigt man den Befehl *partial* aus der Bibliothek *functools*:

```
from functools import partial
```

Möchte man nun zum Beispiel beim Drücken des Buttons die Funktion karte_aufdecken mit den Parametern 2 und 3 ausführen lassen, dann lautet der Quellcode dafür:

```
mein_button = Button(..., command=partial(karte_aufdecken, 2, 3), ...)
```

Statt festen Werten wie 2 und 3 können natürlich auch Variablen verwendet werden.

Array (verschachtelte Listen) initialisieren

Manchmal möchte man zunächst einmal einen Array mit einer festen Größe initialisieren, bevor man ihn befüllt. Dies ist in Python etwas umständlicher als in anderen Programmiersprachen. Der folgende Quellcode erzeugt einen leeren Array mit vorgegebener Anzahl an Spalten und Zeilen:

```
mein_array = [['' for i in range(anzahl_spalten)] \\
                  for j in range(anzahl_zeilen)]
```

Das Ergebnis bei 3 Spalten und 2 Zeilen würde dann so aussehen:

```
[['', '', ''],
 ['', '', '']]
```

Anschließend kann zum Beispiel auf das Feld 1, 0 (zweite Zeile und erste Spalte; beim Programmieren zählt man von 0 an) ein neuer Wert hinterlegt werden:

```
mein_array[1][0] = 'a1'
```

Zufälliges Element aus einer Liste nehmen

Um aus einer Liste ein zufälliges Element zu nehmen, benötigt man die Funktion *choice* aus der Klasse *random*:

```
from random import choice
```

Anschließend wird mit *choice(meine_liste)* ein zufälliges Element aus der Liste *meine_liste* zurückgegeben. Das Element wird dabei nicht aus der Liste entfernt. Hier ein Beispiel:

```
moegliche_karten = ['a1', 'a2', 'b1', 'b2', 'c1', 'c2', 'd1', 'd2']
zufaellige_karte = choice(moegliche_karten)
```

Weitere Tutorials

Hier eine kleine Übersicht nützlicher Tutorials zu GUI Programmierung in Python:

- http://www.python-kurs.eu/python_tkinter.php
  Zeigt und erklärt die wichtigsten Komponenten und gibt Beispiele zur Einbindung.
- http://www.tutorialspoint.com/python/python_gui_programming.htm
  Listet nicht nur wichtige Komponenten auf, sondern nennt auch deren wichtigsten Attribute.
- http://www.tkdocs.com/tutorial/index.html
  Sehr umfangreiches Tutorial. Rechts in der Navigation, sollte Python als Programmiersprache im Dropdown-Menü ausgewählt werden.