34. ΓΡΑΦΙΚΟ ΠΕΡΙΒΑΛΛΟΝ TKINTER - TTKBOOTSTRAP

Επανάληψη κεφαλαίων 15 - 17

34.0.0 Λύσεις των ασκήσεων

34.0.1 Άσκηση 1

Γράψτε μια συνάρτηση η οποία μας επιστρέφει μόνο τις μεγάλες λέξεις από τη λίστα:

```
long_words=['blog', 'Treehouse', 'Python_Rules', 'hi'])
```

Μπορούμε να πούμε πως οποιαδήποτε λέξη μεγαλύτερη από 7 χαρακτήρες είναι μια μεγάλη λέξη. Γράψτε την πρώτα με τον κλασικό τρόπο και κατόπιν χρησιμοποιήστε list comprehensions:

Λύση:

```
def long_words(lst):
      words = []
      for word in lst:
            if len(word) > 5:
            words.append(word)
      return words
```

Τώρα, αναθέτουμε σε μια μεταβλητή να κρατήσει τις λέξεις μας και κάνουμε loop σε όλες τις λέξεις στη λίστα για να ελέγξουμε το μήκος τους. Αν είναι μεγαλύτερο από 7 όπως είπαμε, προσθέτουμε τη λέξη στη λίστα και τέλος, επιστρέφουμε τη λίστα. Έτσι η λίστα:

```
long_words=['blog', 'Treehouse', 'Python_Rules', 'hi'])
```

επιστρέφει:

```
['Treehouse', 'Python_Rules']
```

δηλαδή το αναμενόμενο.

Ο κώδικάς μας διαμορφώθηκε:

```
def long_words(lst):
    words = []
    for word in lst:
        if len(word) > 5:
            words.append(word)
    print(words)
list1 =['blog', 'Treehouse', 'Python_Rules', 'hi']
long_words(list1)
```

Ας ξαναγράψουμε τώρα τη συνάρτηση χρησιμοποιώντας τη σύνταξη list comprehension:

```
def long_words(lst):
return [word for word in lst]
```

Μπορούμε να χρησιμοποιήσουμε τον παραπάνω κώδικα σαν ένα ενδιάμεσο βήμα, αφού μας επιστρέφει όλη τη λίστα. Τώρα μπορούμε να βάλουμε και τη δήλωση υπό συνθήκη στο τέλος του βρόχου for):

```
def long_words(lst):
return [word for word in lst if len(word) > 5]
```

Ο κώδικάς μας τώρα είναι:

```
def long_words(lst):
    x = [word for word in lst if len(word) > 5]
    print(x)
list1 =['blog', 'Treehouse', 'Python_Rules', 'hi']
long_words(list1)
```

Δοκιμάζουμε τον κώδικά μας χρησιμοποιώντας την παρακάτω λίστα:

```
long_words(['list', 'comprehension', 'Treehouse', 'Ken'])
```

η οποία μας επιστρέφει:

```
['comprehension', 'Treehouse'].
```

34.0.2 Άσκηση 2 (class_IOstring_less12.py):

Γράψτε μια κλάση που έχει δύο μεθόδους: get_String και print_String .

Η get_String δέχεται μια συμβολοσειρά από τον χρήστη και η print_String εκτυπώνει τη συμβολοσειρά με κεφαλαία.

Λύση:

```
class IOString():
    def __init__(self):
        self.str1 = ""

    def get_String(self):
        self.str1 = input("Παρακαλούμε γράψτε κάτι: ")

    def print_String(self):
        print(self.str1.upper())

# Δημιουργία στιγμιοτύπου της IOString class χωρίς να
παρέχουμε συμβολοσειρά (ζητάμε από τον χρήστη)
str1 = IOString()
str1.get_String()
str1.print_String()
```

Παράδειγμα εξόδου:

fsdfasdf

FSDFASDF

34.0.3 Άσκηση 3 (class_rectangle_12less.py):

Δημιουργήστε μια γονική κλάση "Shape" με μια μέθοδο "area" που

επιστρέφει το εμβαδόν ενός σχήματος. Στη συνέχεια, δημιουργήστε δύο

παιδικές (child) κλάσεις "Rectangle" και "Circle" που κληρονομούν από

την κλάση "Shape" και υλοποιούν τη μέθοδο "area" με τον κατάλληλο

τρόπο για κάθε σχήμα.

```
class Shape:
    def (self):
        pass

class Rectangle(Shape):
    def __init__(self, width, height):
        self.width = width
        self.height = height
    def area(self):
        return self.width * self.height

class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius
    def area(self):
        return 3.14 * self.radius * self.radius

# Αυτό που λείπει τώρα από τον κώδικά μας είναι η δημιουργία
#στιγμιοτύπων των κλάσεών μας και ο υπολογισμός των εμβαδών

# Δημιουργία παραλληλόγραμμου και υπολογισμός εμβαδού

rectangle = Rectangle(5, 10)
rectangle_area = rectangle.area()
print("Το εμβαδόν του παραλληλόγραμμου είναι:",
rectangle_area)

# Δημιουργία κύκλου και υπολογισμός εμβαδού
circle = Circle(3)
circle_area = circle.area()
print("Το εμβαδόν του κύκλου είναι:", circle_area)
```

Η έξοδος είναι:

Το εμβαδόν του παραλληλόγραμμου είναι: 50

Το εμβαδόν του κύκλου είναι: 28.259999999999998

34.0.4 Άσκηση 4

Χρησιμοποιήστε lambda function για να ταξινομήσετε την παρακάτω λίστα (με στοιχεία 3 tuples), σύμφωνα με το δεύτερο στοιχείο: my_list = [("apple", 50), ("banana", 10), ("cherry", 30)]

```
my_list = [("apple", 50), ("banana", 10), ("cherry", 30)]

sorted_list = sorted(my_list, key=lambda x: x[1])

print(sorted_list)
```

Έξοδος

```
("banana", 10), ("apple", 50), ("cherry", 30)
```

34.0.5 Άσκηση 5

Χρησιμοποιήστε lambda function για να ταξινομήσετε την παρακάτω λίστα σύμφωνα με τον τελευταίο χαρακτήρα της κάθε συμβολοσειράς.

```
my_list = ["apple", "banana", "cherry"]

sorted_list = sorted(my_list, key=lambda x: x[-1])

print(sorted_list)
```

Έξοδος:

```
["banana", "apple", "cherry"]
```

34.1.1 Γραφικό Περιβάλλον με Tkinter

Το Tkinter προφέρεται ως tea-kay-inter. Το Tkinter είναι η διεπαφή της Python με το Tk, η οποία είναι η εργαλειοθήκη GUI (Graphical User Interface – Γραφική Διεπαφή Χρήστη) για το Tcl/Tk.

Η Tcl (προφέρεται ως tickle = γαργαλητό) είναι μια γλώσσα δέσμης ενεργειών που χρησιμοποιείται συχνά σε δοκιμές, πρωτότυπα και ανάπτυξη GUI.

Το Tk είναι μια εργαλειοθήκη γραφικών στοιχείων ανοιχτού κώδικα, πολλαπλών πλατφορμών που χρησιμοποιείται από πολλές διαφορετικές γλώσσες προγραμματισμού για τη δημιουργία προγραμμάτων GUI.

Η Python υλοποιεί το Tkinter σαν ένα module. Το Tkinter είναι ένας wrapper επεκτάσεων της C που χρησιμοποιούν βιβλιοθήκες Tcl/Tk.

Το Tkinter μας επιτρέπει να αναπτύσσουμε εφαρμογές desktop. Είναι ένα πολύ καλό εργαλείο για προγραμματισμό γραφικών διεπαφών στην Python.

Άλλες βιβλιοθήκες ή frameworks για γραφικό περιβάλλον στην Python, είναι τα:

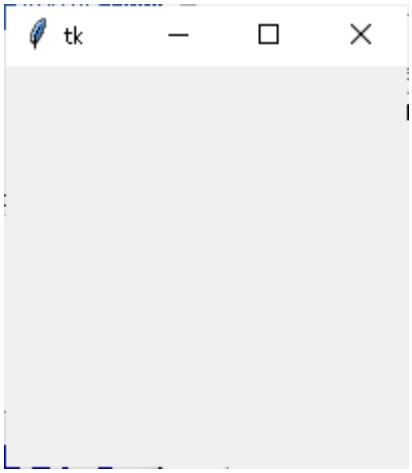
- PyQt5
- Kivy
- wxPython
- Libavg
- PySimpleGUI
- PyForms
- Wax
- PySide2
- PyGUI

34.1.2 Δημιουργία παραθύρου στο Tkinter

Με τον παρακάτω κώδικα εμφανίζουμε ένα παράθυρο στην οθόνη:

```
import tkinter as tk

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

Εάν εκτελέσουμε το πρόγραμμα, θα δούμε το ακόλουθο παράθυρο:



Πως δουλεύει.

Πρώτα, εισάγουμε το module tkinter ως tk στο πρόγραμμα:

```
import tkinter as tk
```

Δημιουργούμε ένα στιγμιότυπο της κλάσης tk.Tk που θα δημιουργήσει το παράθυρο root της εφαρμογής:

```
root = tk.Tk()
```

Κατά σύμβαση, το κύριο παράθυρο στο Tkinter ονομάζεται root. Αλλά μπορούμε να χρησιμοποιήσουμε οποιοδήποτε άλλο όνομα όπως main.

Καλούμε τη μέθοδο mainloop() του αντικειμένου του κύριου παραθύρου:

```
root.mainloop()
```

Η mainloop() διατηρεί το παράθυρο ορατό στην οθόνη. Εάν δεν καλέσουμε τη μέθοδο mainloop(), το παράθυρο θα εμφανιστεί και θα εξαφανιστεί αμέσως.

Επίσης, η μέθοδος mainloop() διατηρεί το παράθυρο να εμφανίζεται και να λειτουργεί μέχρι να το κλείσουμε.

Συνήθως, καλούμε τη μέθοδο mainloop() στο τέλος, μετά τη δημιουργία των γραφικών στοιχείων.

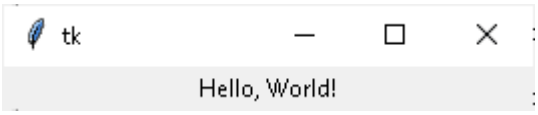
34.1.3 Εισαγωγή widget στο παράθυρο

Στο Tkinter, τα αντικείμενα ονομάζονται widgets.

Για να προσθέσουμε μια ετικέτα (label) στο παράθυρο, πληκτρολογούμε:

```
import tkinter as tk
root = tk.Tk()

# Τοποθέτηση μιας ετικέτας - label στο root window
message = tk.Label(root, text="Hello, World!")
message.pack()

# Συνεχής εμφάνιση του παραθύρου
root.mainloop()
```

Εκτελώντας τον κώδικα, βλέπουμε:



Πως δουλεύει:

Για να δημιουργήσουμε ένα γραφικό στοιχείο που ανήκει σε ένα container, χρησιμοποιούμε την ακόλουθη σύνταξη:

widget = Όνομα γραφικού στοιχείου (container, **options)

Το κοντέινερ είναι το γονικό παράθυρο ή πλαίσιο όπου θέλουμε να

τοποθετήσουμε το γραφικό στοιχείο.

Οι επιλογές (options) είναι ένα ή περισσότερα ορίσματα λέξεων-κλειδιών που καθορίζουν τις διαμορφώσεις του γραφικού στοιχείου.

Στο πρόγραμμα, τα ακόλουθα δημιουργούν ένα γραφικό στοιχείο Label που τοποθετείται στο παράθυρο root:

```
message = tk.Label(root, text="Hello, World!")
```

Και η ακόλουθη δήλωση τοποθετεί την ετικέτα στο παράθυρο root:

```
message.pack()
```

Εάν δεν καλέσουμε τη μέθοδο pack(), το Tkinter εξακολουθεί να δημιουργεί το γραφικό στοιχείο. Ωστόσο, το widget είναι αόρατο.

Κατά τη δημιουργία ενός παραθύρου, μπορούμε να επεξεργαζόμαστε και όλα τα χαρακτηριστικά του όπως είναι, το μέγεθός του, η θέση του στην οθόνη, η θέση του όσον αφορά άλλα παράθυρα στην οθόνη, η διαφάνειά του και πολλά άλλα.

Καλό θα ήταν, αν χρειαζόμαστε κάτι πιο εξειδικευμένο για τη δημιουργία κάποιου γραφικού μας, να έχουμε ανοιχτές καρτέλες με τους παρακάτω πολύ χρήσιμους πόρους για το Tkinter, τις οποίες και να συμβουλευόμαστε:

Graphical User Interfaces with Tk

tkinter — Python interface to Tcl/Tk

Tkinter 8.5 reference: a GUI for Python

34.1.4 Αλλαγή μεγέθους και τοποθεσίας του παραθύρου

Στο Tkinter, το μέγεθος και η τοποθεσία του παραθύρου καθορίζεται

από γεωμετρικά χαρακτηριστικά τα οποία δηλώνουμε ως εξής:

width x height ± x ± y

window.geometry('width x height + x + y')

- Το width είναι το πλάτος του παραθύρου σε pixels
- Το height είναι το ύψος του παραθύρου σε pixels
- Το x είναι η οριζόντια θέση του παραθύρου. Για παράδειγμα, +50 σημαίνει ότι η αριστερή άκρη του παραθύρου πρέπει να είναι 50px από την αριστερή άκρη της οθόνης και -50 σημαίνει, ότι η δεξιά άκρη του παραθύρου θα πρέπει να είναι 50 pixels από τη δεξιά άκρη της οθόνης.
- Το y είναι η κάθετη θέση του παραθύρου. Για παράδειγμα, +50 σημαίνει ότι το πάνω μέρος του παραθύρου πρέπει να είναι 50px κάτω από το πάνω μέρος της οθόνης και -50 σημαίνει, ότι η κάτω άκρη του παραθύρου θα πρέπει να είναι 50 pixels πάνω από το κάτω μέρος της οθόνης.

Για να αλλάξουμε το μέγεθος και τη θέση του παραθύρου

χρησιμοποιούμε τη μέθοδο geometry():

```
window.geometry(new_geometry)
```

Στο παρακάτω παράδειγμα, αλλάζουμε το μέγεθος του Παραθύρου σε

600 x 400 και τη θέση του παραθύρου σε 50px από επάνω και από

αριστερά της οθόνης:

```
import tkinter as tk
root = tk.Tk()
root.title('Το Πρώτο μου παράθυρο')
```

```
root.geometry('600x400+50+50')

root.mainloop()
```

Κάποιες φορές, θέλουμε να κεντράρουμε το παράθυρο στην οθόνη:

```
import tkinter as tk

root = tk.Tk()
root.title('Tkinter Window - Center')

window_width = 300
window_height = 200

# Παίρνουμε το μέγεθος της οθόνης
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
# Βρίσκουμε το κέντρο της οθόνης
center_x = int(screen_width/2 - window_width / 2)
center_y = int(screen_height/2 - window_height / 2)
# Κεντράρισμα του παραθύρου
root.geometry(f'{window_width}x{window_height}+{center_x}+{c
enter
_y}')

root.mainloop()
```

Για να εμποδίσουμε το παράθυρο να αλλάζει μέγεθος, μπορούμε να

χρησιμοποιήσουμε τη μέθοδο resizable():

```
window.resizable(width,height)
```

Έχει δύο παραμέτρους οι οποίες καθορίζουν αν το πλάτος και το ύψος

του παραθύρου μπορούν να αλλάξουν μέγεθος.

Επομένως, για παράθυρο με σταθερό μέγεθος έχουμε:

```
import tkinter as tk

root = tk.Tk()
root.title('Το Πρώτο μου Παράθυρο')
root.geometry('600x400+50+50')
root.resizable(False, False)

root.mainloop()
```

Όταν ένα παράθυρο αλλάζει μέγεθος, μπορούμε να καθορίσουμε το ελάχιστο και μέγιστο μέγεθος χρησιμοποιώντας τις μεθόδους minsize() και maxsize().

```
window.minsize(min_width, min_height)
window.maxsize(min_height, max_height)
```

34.1.5 Διαφάνεια

Το Tkinter μας επιτρέπει να καθορίζουμε τη διαφάνεια ενός παραθύρου ρυθμίζοντας το κανάλι άλφα του να κυμαίνεται από 0,0 (πλήρως διαφανές) έως 1,0 (πλήρως αδιαφανές):

```
window.attributes('-alpha',0,5)
```

Το ακόλουθο παράδειγμα απεικονίζει ένα παράθυρο με 50% διαφάνεια:

```
import tkinter as tk

root = tk.Tk()
root.title('Το πρώτο μου Παράθυρο')
root.geometry('600x400+50+50')
```

```
root.resizable (False, False)
root.attributes('-alpha', 0,5)

root.mainloop()
```

Η έξοδος είναι:

34.1.6 Σειρά στοίβαξης παραθύρων

Η σειρά στοίβαξης των παραθύρων αναφέρεται στη σειρά των παραθύρων που τοποθετούνται στην οθόνη από κάτω προς τα πάνω. Το πιο κοντινό παράθυρο βρίσκεται στο επάνω μέρος της στοίβας και επικαλύπτει το χαμηλότερο.

Για να διασφαλίσουμε ότι ένα παράθυρο βρίσκεται πάντα στην κορυφή της σειράς στοίβαξης, μπορούμε να χρησιμοποιήσουμε το χαρακτηριστικό -topmost ως εξής:

```
window.attributes('-topmost', 1)
```

Για να μετακινήσουμε ένα παράθυρο πάνω ή κάτω της στοίβας,

μπορούμε να χρησιμοποιήσουμε τις μεθόδους lift() και low():

```
window.lift()
window.lift(another_window)
```

```
window.lower()
window.lower(another_window)
```

Το παρακάτω παράδειγμα τοποθετεί το παράθυρο root πάνω από όλα τα άλλα παράθυρα. Με άλλα λόγια, το παράθυρο root είναι πάντα στην κορυφή:

```
import tkinter as tk
root = tk.Tk()
root.title('Tkinter Window Demo')
root.geometry('300x200+50+50')
root.resizable(0, 0)
root.attributes('-topmost', 1)

root.mainloop()
```

34.1.7 Αλλαγή του προεπιλεγμένου εικονιδίου

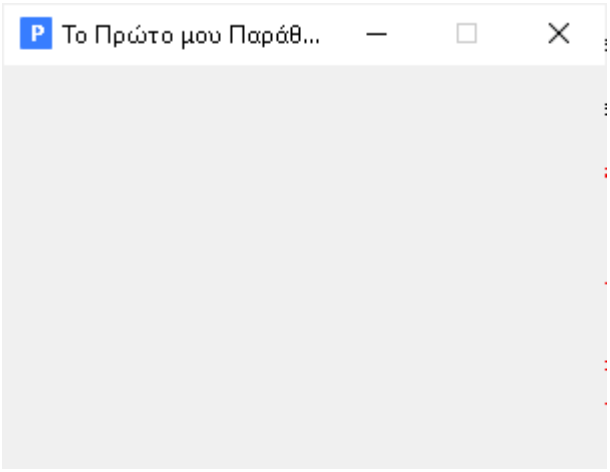
Το παράθυρο του Tkinter εμφανίζει ένα προεπιλεγμένο εικονίδιο. Για να αλλάξουμε αυτό το εικονίδιο:

1. Προετοιμάζουμε μια εικόνα σε μορφή .ico. Εάν έχουμε την εικόνα σε άλλες μορφές όπως png ή jpg, θα χρειαστεί να τη μετατρέψουμε σε μορφή .ico (μπορούμε να το κάνουμε στο διαδίκτυο).
2. Τοποθετούμε το εικονίδιο σε ένα φάκελο που είναι προσβάσιμος από το πρόγραμμα και καλούμε τη μέθοδο iconbitmap() του παραθύρου.

Ας δούμε παρακάτω:

```
import tkinter as tk
root = tk.Tk()
root.title('Το Πρώτο μου Παράθυρο')
root.geometry('300x200+50+50')
root.resizable(False, False)
root.iconbitmap('python_new.ico')

root.mainloop()
```

Η έξοδος είναι:



34.1.8 Ανακεφαλαίωση

- Χρήση της μεθόδου title() για να αλλάξουμε τον τίτλο του παραθύρου.
- Χρήση της μεθόδου geometry() για να αλλάξουμε το μέγεθος και τη θέση του παραθύρου.
- Χρήση της μεθόδου resizable() για να καθορίσουμε εάν ένα παράθυρο μπορεί να αλλάξει μέγεθος οριζόντια ή κάθετα.
- Χρήση του window.attributes('-alpha',0.5) για να ορίσουμε τη διαφάνεια του παραθύρου.
- Χρήση του window.attributes('-topmost', 1) για να κάνουμε το

παράθυρο να είναι πάντα στην κορυφή.

- Χρήση των μεθόδων lift() και low() για να μετακινήσουμε το

παράθυρο πάνω και κάτω στη σειρά στοίβαξης παραθύρων.

- Χρήση της μεθόδου iconbitmap() για να αλλάξουμε το

προεπιλεγμένο εικονίδιο του παραθύρου.

34.1.9 Tk themed widgets

Το Tkinter έχει δύο γενιές widget:

Τα παλιά κλασικά γραφικά στοιχεία tk. Παρουσιάστηκαν το 1991. Τα νεότερα θεματικά γραφικά στοιχεία ttk προστέθηκαν το 2007 με το Tk 8.5. Τα νεότερα γραφικά στοιχεία με θέμα Tk αντικαθιστούν πολλά (αλλά όχι όλα) κλασικά γραφικά στοιχεία.

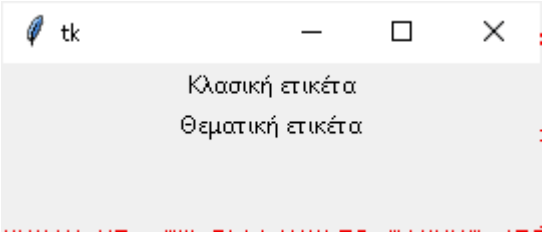
Σημειώστε ότι το ttk σημαίνει Tk theme. Επομένως, τα γραφικά στοιχεία με θέμα Tk (themed) είναι τα ίδια με τα γραφικά στοιχεία ttk.

Η ενότητα tkinter.ttk περιέχει όλα τα νέα γραφικά στοιχεία ttk. Είναι καλή πρακτική να χρησιμοποιούμε πάντα θεματικά γραφικά στοιχεία όποτε είναι διαθέσιμα.

Οι ακόλουθες δηλώσεις εισάγουν τα κλασικά και τα νέα γραφικά στοιχεία με θέμα Tk και δημιουργούν κλασικές και θεματικές ετικέτες:

```
import tkinter as tk
from tkinter import ttk

root = tk.Tk()

tk.Label(root, text='Κλασική ετικέτα').pack()
ttk.Label(root, text='Θεματική ετικέτα').pack()

root.mainloop()
```

Έξοδος:



Και οι δύο ετικέτες μοιάζουν παρόμοιες. Όμως, η εμφάνισή τους εξαρτάται από την πλατφόρμα στην οποία εκτελείται το πρόγραμμα.

34.1.10 Νέα στοιχεία ttk

Τα παρακάτω γραφικά στοιχεία ttk αντικαθιστούν τα γραφικά στοιχεία

Tkinkter με τα ίδια ονόματα:

- Button
- Checkbutton
- Entry
- Frame
- Label
- LabelFrame
- Menubutton
- PanedWindow
- Radiobutton
- Scale
- Scrollbar
- Spinbox

Νέα στοιχεία ttk:

- Combobox
- Notebook
- Progressbar
- Separator
- Sizegrip
- Treeview

Πλήρη αναφορά για τα themed widgets μπορείτε να βρείτε στην τεκμηρίωση της Python.

Συμπερασματικά:

• το Tkinter έχει κλασικά αλλά και themed widgets (ttk)

• Το module, tkinter.ttk περιέχει όλα τα ttk widgets

• Όταν είναι διαθέσιμα, είναι καλό να χρησιμοποιούμε αυτά τα widgets.

34.1.11 Εφαρμογή με GUI, Fahrenheit to Celsius

Πάμε να δημιουργήσουμε μια μικρή εφαρμογή μετατροπής βαθμών Φαρενάιτ σε Κελσίου και γραφικό περιβάλλον.

Βασικά, η εφαρμογή έχει ένα label (ετικέτα), ένα πεδίο εισαγωγής αριθμού (entry field) και ένα κουμπί (button). Όταν εισάγουμε μια θερμοκρασία σε Φαρενάιτ και κάνουμε κλικ στο κουμπί «Μετατροπή», θα μετατραπεί η τιμή στο πεδίο εισαγωγής από Φαρενάιτ σε Κελσίου.

Εάν εισαγάγουμε μια τιμή που δεν μπορεί να μετατραπεί σε αριθμό, το πρόγραμμα θα εμφανίσει ένα σφάλμα.

Για να δημιουργήσουμε αυτήν την εφαρμογή, ακολουθούμε τα παρακάτω βήματα:

1. Κάνουμε εισαγωγή όλων των απαραίτητων modules:

```
import tkinter as tk
from tkinter import ttk
from tkinter.messagebox import showerror
```

2. Δημιουργούμε το παράθυρο root και τις ρυθμίσεις του:

```
# Παράθυρο root
root = tk.Tk()
root.title('Μετατροπέας Θερμοκρασίας')
root.geometry('360x90')
```

```
root.resizable(False, False)
```

Γράφουμε τη συνάρτηση που μετατρέπει τη θερμοκρασία:

```
def fahrenheit_to_celsius(f):
# Μετατροπή Φαρενάιτ σε Κελσίου
 return (f - 32) * 5/9
```

Δημιουργούμε ένα frame το οποίο κρατάει το πεδίο εισαγωγής αριθμού:

```
frame = ttk.Frame(root)
```

Καθορίζουμε μια επιλογή που θα χρησιμοποιείται από όλα τα πεδία:

```
options = {'padx': 5, 'pady': 5}
```

Καθορίζουμε την ετικέτα (label) το πεδίο εισαγωγής (entry) και το κουμπί. Η ετικέτα θα δείξει το αποτέλεσμα μόλις πατηθεί το κουμπί «Μετατροπή»

```
# Ετικέτα θερμοκρασίας
temperature_label = ttk.Label(frame, text='Fahrenheit')
temperature_label.grid(column=0, row=0, sticky='W',
**options)
# Πεδίο εισαγωγής θερμοκρασίας
temperature_entry = ttk.Entry(frame,
textvariable=temperature)
temperature_entry.grid(column=1, row=0, **options)
temperature_entry.focus()
# Κουμπί μετατροπής
convert_button = ttk.Button(frame, text=Μετατροπή)
convert_button.grid(column=2, row=0, sticky='W', **options)
convert_button.configure(command=convert_button_clicked)
# Ετικέτα αποτελέσματος
result_label = ttk.Label(frame)
result_label.grid(row=1, columnspan=3, **options)
```

Τέλος, τοποθετούμε το frame στο παράθυρο root και τρέχουμε τη μέθοδο ***mainloop():***

```
frame.grid(padx=10, pady=10)
root.mainloop()
```

Όλος ο κώδικας του προγράμματος:

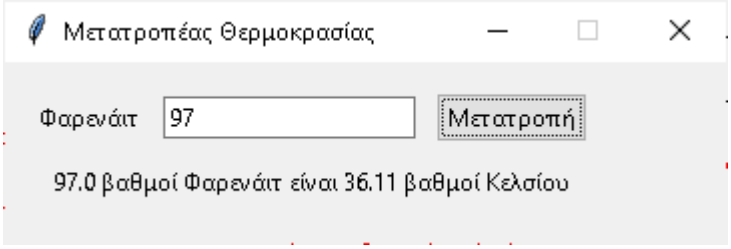
```
import tkinter as tk
from tkinter import ttk
from tkinter.messagebox import showerror

# Παράθυρο root
root = tk.Tk()
root.title('Μετατροπέας Θερμοκρασίας')
root.geometry('360x90')
root.resizable(False, False)
def fahrenheit_to_celsius(f):
    # Μετατροπή Φαρενάιτ σε Κελσίου
    return (f - 32) * 5/9

# Δημιουργία frame
frame = ttk.Frame(root)

# Ρυθμίσεις του πεδίου
options = {'padx': 5, 'pady': 5}

# Ετικέτα θερμοκρασίας
temperature_label = ttk.Label(frame, text='Φαρενάιτ')
temperature_label.grid(column=0, row=0, sticky='W',
**options)

# Πεδίο εισαγωγής αριθμού (θερμοκρασίας)
temperature = tk.StringVar()
temperature_entry = ttk.Entry(frame,
textvariable=temperature)
temperature_entry.grid(column=1, row=0, **options)
temperature_entry.focus()

# Κουμπί μετατροπής
```

```
def convert_button_clicked():
 # Διαχείριση γεγονότος εισαγωγής αριθμού και πατήματος του
κουμπιού
    try:
        f = float(temperature.get())
        c = fahrenheit_to_celsius(f)
        result = f'{f} βαθμοί Φαρενάιτ είναι {c:.2f} βαθμοί
Κελσίου'
        result_label.config(text=result)
    except ValueError as error:
        showerror(title='Error', message=error)

convert_button = ttk.Button(frame, text='Μετατροπή')
convert_button.grid(column=2, row=0, sticky='W', **options)
convert_button.configure(command=convert_button_clicked)

# Ετικέτα αποτελέσματος
result_label = ttk.Label(frame)
result_label.grid(row=1, columnspan=3, **options)

# Προσθήκη padding στο frame
frame.grid(padx=10, pady=10)

# Εκκίνηση εφαρμογής
root.mainloop()
```

Τεκμηρίωση για τις επιλογές της μεθόδου grid() – grid options

Τεκμηρίωση για τη μέθοδο button() – button options

Έξοδος:

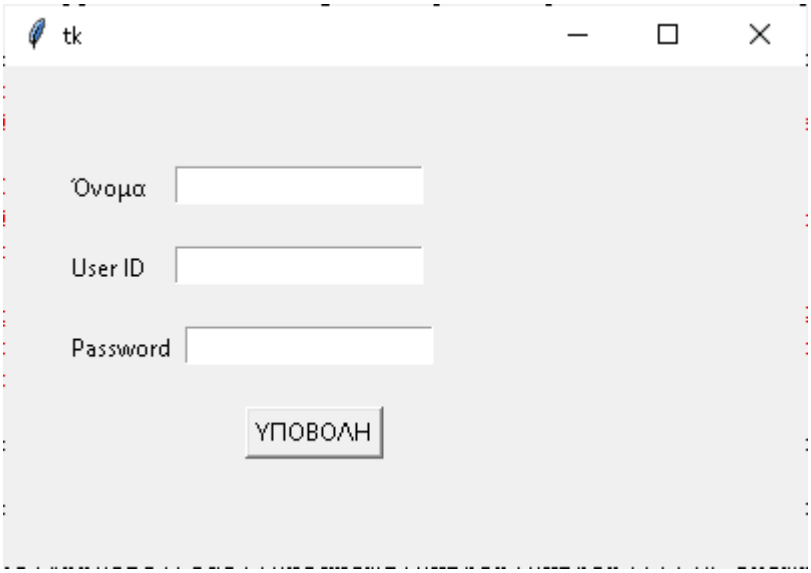


Ας δούμε ακόμα ένα παράδειγμα γραφικού περιβάλλοντος, με τη δημιουργία ενός παραθύρου το οποίο έχει τρία πεδία κι ένα κουμπί υποβολής των στοιχείων, εφόσον έχουν συμπληρωθεί:

34.1.12 Εφαρμογή με GUI, Υποβολή στοιχείων

```
'''
Γράψτε ένα προγραμματάκι το οποίο να έχει τρία πεδία
εισαγωγής:
Όνομα, User ID, και Password, καθώς κι ένα κουμπί εισαγωγής,
"ΥΠΟΒΟΛΗ".
'''

import tkinter as tk
root = tk.Tk()
root.geometry("400x250")

# Πεδίο 1
name = tk.Label(root, text = "Όνομα").place(x = 30, y = 50)

# Πεδίο 2
email = tk.Label(root, text = "User ID").place(x = 30, y =
90)

# Πεδίο 3
password =  tk.Label(root, text = "Password").place(x = 30,
y = 130)

# Κουμπί υποβολής στοιχειων
sbmitbtn = tk.Button(root, text = "ΥΠΟΒΟΛΗ",
activebackground = "green", activeforeground =
"blue").place(x = 120, y = 170)
entry1 = tk.Entry(root).place(x = 85, y = 50)
entry2 = tk.Entry(root).place(x = 85, y = 90)
entry3 = tk.Entry(root).place(x = 90, y = 130)

# Δημιουργία και επανάληψη παραθύρου
root.mainloop()
```

Έξοδος:



Με το Tk και το Ttkinter, μπορούμε να δημιουργήσουμε πλούσια γραφικά, προσθέτοντας όλα τα στοιχεία που χρειαζόμαστε, όπως checkbuttons, radio buttons, μενού κ.λ.π.

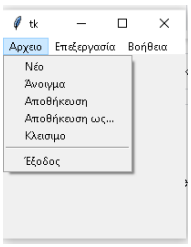
Ας δούμε ακόμα ένα παράδειγμα που αφορά τη δημιουργία μενού:

34.1.13 Δημιουργία μενού

```
from tkinter import *
def donothing():
   filewin = Toplevel(root)
   button = Button(filewin, text="Do nothing button")
   button.pack()

root = Tk()
menubar = Menu(root)
filemenu = Menu(menubar, tearoff = 0)
filemenu.add_command(label = "Νέο", command = donothing)
filemenu.add_command(label = "Άνοιγμα", command = donothing)
filemenu.add_command(label = "Αποθήκευση", command =
donothing)
filemenu.add_command(label = "Αποθήκευση ως...", command =
donothing)
```

```
filemenu.add_command(label = "Κλεισιμο", command =
donothing)

filemenu.add_separator()

filemenu.add_command(label = "Έξοδος", command = root.quit)
menubar.add_cascade(label = "Αρχειο", menu = filemenu)
editmenu = Menu(menubar, tearoff=0)
editmenu.add_command(label = "Αναιρεση", command =
donothing)

editmenu.add_separator()

editmenu.add_command(label = "Αποκοπή", command = donothing)
editmenu.add_command(label = "Αντιγραφή", command =
donothing)
editmenu.add_command(label = "Επικόλληση", command =
donothing)
editmenu.add_command(label = "Διαγραφή", command =
donothing)
editmenu.add_command(label = "Επιλογή όλων", command =
donothing)

menubar.add_cascade(label = "Επεξεργασία", menu = editmenu)
helpmenu = Menu(menubar, tearoff=0)
helpmenu.add_command(label = "Αρχειο Οδηγιών", command =
donothing)
helpmenu.add_command(label = "Περί...", command = donothing)
menubar.add_cascade(label = "Βοήθεια", menu = helpmenu)

root.config(menu = menubar)
root.mainloop()
```

Η έξοδος είναι:

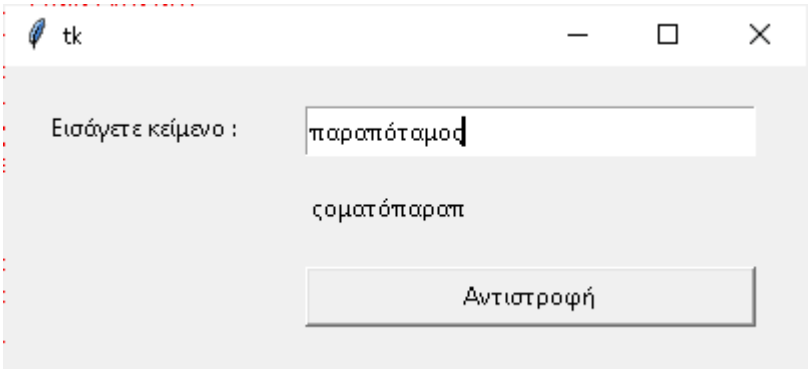


34.2.0 Ασκήσεις

1. (34.ask_1) Γράψτε ένα πρόγραμμα το οποίο να έχει ένα παράθυρο με ένα πεδίο, το οποίο δέχεται κείμενο από τον χρήστη.

Το πρόγραμμα πρέπει, με το πάτημα ενός κουμπιού, να αντιστρέφει και να παρουσιάζει το αντεστραμμένο κείμενο.

Η έξοδος, πρέπει να είναι περίπου έτσι:



2. Γράψτε ένα πρόγραμμα με γραφικό περιβάλλον, το οποίο να υπολογίζει τον Μέγιστο Κοινό Διαιρέτη (ΜΚΔ), καθώς και το Ελάχιστο Κοινό Πολλαπλάσιο (ΕΚΠ).

Θα πρέπει να δημιουργήσετε 2 πεδία που να δέχονται από έναν ακέραιο από τον χρήστη κι ένα combobox ώστε ο χρήστης να επιλέγει τη λειτουργία που χρειάζεται. Θα πρέπει το παράθυρό σας να είναι περίπου όπως το παρακάτω: