Dice Rolling Simulator

Snake and Ladders, Ludo and Checkers, are the most favourite games of all time. But, it feels terrible when you drop the plan of playing just because you couldn't find the dice. So, here is a fascinating option to give you a chance to make something cool. Let's build a Dice Rolling Simulator with basic knowledge of Python.

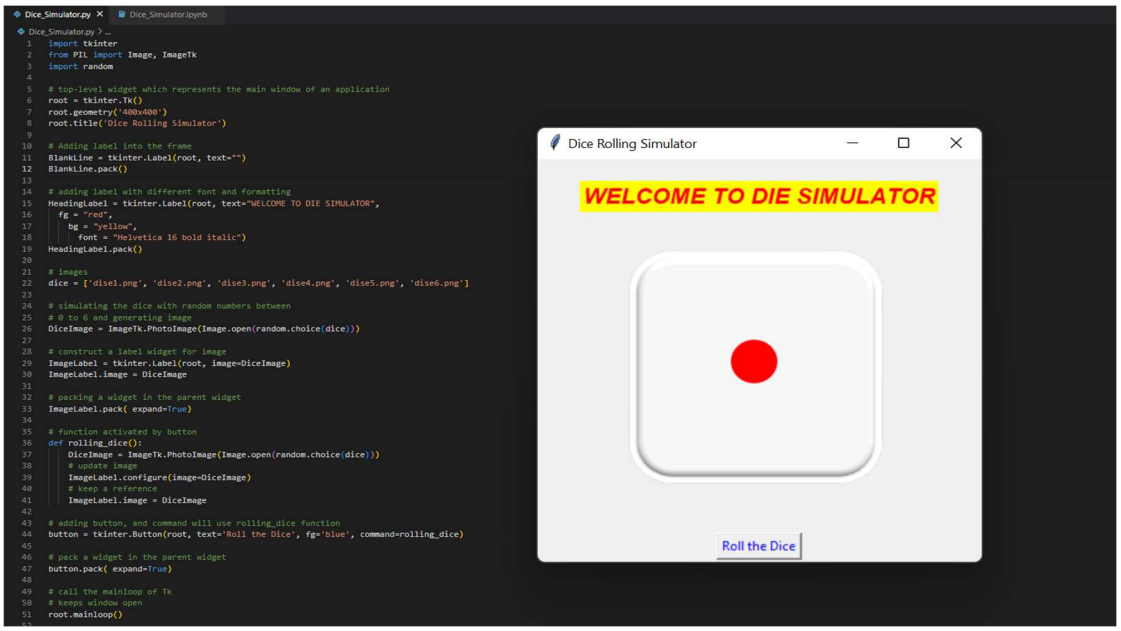
What is Tkinter?

Python offers various packages to design the GUI, i.e. the Graphical User Interface. Tkinter is the most common, fast, and easy to use Python package used to build Graphical User Interface applications. It provides a powerful Object-Oriented Interface and is easy to use. Also, you develop an application; you can use it on any platform, which reduces the need of amendments required to use an app on Windows, Mac, or Linux.

Dice Rolling Simulator in Python

We all know about dice. It's a simple cube with numbers from 1 to 6 written on its face. But what is simulation? It is making a computer model. Thus, a dice simulator is a simple computer model that can roll a dice for us.

We aim to build a dice simulator which looks like:



Build Dice Rolling Simulator

Step 1: Importing the required modules

We will import the following modules:

Tkinter: Imported to use Tkinter and make GUI applications.

Image, Imagetk: Imported from PIL, i.e. Python Imaging Library. We use it to perform operations involving images in our UI.

Random: Imported to generate random numbers.

Code:

```
import tkinter
from PIL import Image, ImageTk
import random
```

Step 2: Building a top-level widget to make the main window for our application

In this step, we will build the main window of our application, where the buttons, labels, and images will reside. We also give it a title by title() function.

Code:

```
# top-level widget which represents the main window of an application
root = tkinter.Tk()
root.geometry('400x400')
root.title('Dice Rolling Simulator')
```

Explanation:

The above code sets the title of the application window as ‘Dice Rolling Simulator’. Running the above code will generate a blank window of dice rolling simulator python project with the title on it.

Step 3: Designing the buttons

Now, just think, what we need to roll a die? Just our hands!

The below code will add a label giving a heading to our dice simulator. Also, we will add an image area, which will display the image chosen by random numbers.

Code:

```
# Adding label into the frame
BlankLine = tkinter.Label(root, text="")
BlankLine.pack()

# adding label with different font and formatting
HeadingLabel = tkinter.Label(root, text="WELCOME TO DIE SIMULATOR",
    fg = "red",
     bg = "yellow",
     font = "Helvetica 16 bold italic")
HeadingLabel.pack()

# images
dice = ['dise1.png', 'dise2.png', 'dise3.png',
     'dise4.png', 'dise5.png', 'dise6.png']
# simulating the dice with random numbers between
# 0 to 6 and generating image
DiceImage = ImageTk.PhotoImage(Image.open(random.choice(dice)))

# construct a label widget for image
ImageLabel = tkinter.Label(root, image=DiceImage)
ImageLabel.image = DiceImage

# packing a widget in the parent widget
ImageLabel.pack( expand=True)
```

Explanation:

Here, we use **pack()** to arrange our widgets in row and column form. The 'BlankLine' label is to skip a line, whereas we use 'HeadingLabel' label to give a heading.

root – the name by which we refer to the main window of the application

text – text to be displayed in the Heading Label

fg– the colour of the font used in Heading Label

bg – background colour of the Heading Label

font – used to give customised fonts to the Heading Label text

.pack() – Used to pack the widget onto the root window

Step 4: Forming a list of images to be randomly displayed

Code:

```
# images
dice = ['dise1.png', 'dise2.png', 'dise3.png', 'dise4.png', 'dise5.png', 'dise6.png']

# simulating the dice with random numbers between
# 0 to 6 and generating image
DiceImage = ImageTk.PhotoImage(Image.open(random.choice(dice)))
```

Explanation:

‘dice’ is the list of names of images kept in same folder, which are chosen randomly according to the random number generated.
‘DiceImage’ is used to store an image of dice which is chosen by randomly generated numbers.

Step 5: Constructing a label for image, adding a button and assigning functionality

Code:

```
# construct a label widget for image
ImageLabel = tkinter.Label(root, image=DiceImage)
ImageLabel.image = DiceImage

# packing a widget in the parent widget
ImageLabel.pack( expand=True)

# function activated by button
def rolling_dice():
    DiceImage = ImageTk.PhotoImage(Image.open(random.choice(dice)))
    # update image
    ImageLabel.configure(image=DiceImage)
    # keep a reference
    ImageLabel.image = DiceImage

# adding button, and command will use rolling_dice function
button = tkinter.Button(root, text='Roll the Dice', fg='blue', command=rolling_dice)

# pack a widget in the parent widget
button.pack( expand=True)
```

Explanation:

‘ImageLabel’ is to place an image in the window. The parameter expands declared as True so that even if we resize the window, image remains in the center.

Major function:

‘rolling_dice’ function is a function that is executed every time a button is clicked. This is attained through the ‘command=rolling_dice’ parameter while defining a button.

Step 6: Forming a list of images to be randomly displayed

Code:

```
# call the mainloop of Tk
# keeps window open
root.mainloop()
```

Explanation:

'root.mainloop()' is used to open the main window. It acts as the main function of our program.

Dice Rolling Simulator Python Output

Conclusion

Yay! We have successfully developed a cool application – Dice Rolling Simulator in Python. Now, you can just click on a button and get your next number. Cheers to Python and its package 'Tkinter' which supports functions and makes our work easy. Who would have thought that we can develop an application by only the 'random' function of python? As of now, we have an understanding of Python, Tkinter, and random function.