Tkinter Object-Oriented Window

Summary: in this tutorial, you'll learn how to apply object-oriented programming (https://www.pythontutorial.net/python-oop/) in Tkinter to make the code more organized.

Defining a Tkinter object-oriented window

The following simple program creates a root window (https://www.pythontutorial.net/tkinter/tkinter-window/) and displays it on the screen:

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

When the program is getting more complex, you can use an object-oriented programming (https://www.pythontutorial.net/python-oop/) approach to make the code more organized.

The following program achieves the same result as the program above, but use a `class` (https://www.pythontutorial.net/python-oop/python-class/) instead:

```
import tkinter as tk
```

```
class App(tk.Tk):
    def __init__(self):
        super().__init__()


if __name__ == "__main__":
    app = App()
    app.mainloop()
```

How it works.

- First, define an `App` class that inherits from the `tk.Tk` class. Inside the `__init__()` method, call the `__init__()` method of the `tk.Tk` class.
- Second, create a new instance of the `App` class and call the `mainloop()` method to display the root window.

Another example of an object-oriented window in Tkinter

The following class represents a window (https://www.pythontutorial.net/tkinter/tkinter-window/) that consists of a label (https://www.pythontutorial.net/tkinter/tkinter-label/) and a button (https://www.pythontutorial.net/tkinter/tkinter-button/) . When you click the button, the program displays a message box (https://www.pythontutorial.net/tkinter/tkinter-messagebox/) :

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


class App(tk.Tk):
  def __init__(self):
    super().__init__()

    # configure the root window
    self.title('My Awesome App')
    self.geometry('300x50')
```

```
        # label
        self.label = ttk.Label(self, text='Hello, Tkinter!')
        self.label.pack()

        # button
        self.button = ttk.Button(self, text='Click Me')
        self.button['command'] = self.button_clicked
        self.button.pack()

    def button_clicked(self):
        showinfo(title='Information', message='Hello, Tkinter!')

if __name__ == "__main__":
    app = App()
    app.mainloop()
```

How it works.

- First, create a label and button in the `__init__()` method of App class.
- Second, assign the `button_clicked()` method to the command option of the button. Inside the `button_clicked()` method, display a message box.
- Third, move the application bootstrapping to the `if __name__ = "main"` block.

Summary

- Use an object-oriented programming approach to make the code more organized.
- Define a class that inherits from the `tk.Tk` class. Always, call the `super().__init__()` from the parent class in the child class.