Subclassing in Tkinter

Why do we do it?

To reuse parts of a GUI, it is useful to put them in a separate class that can be used as a readymade component.

Example: an expandable checkbar (a frame containing any number of checkboxes

The text for the checkboxes is stored in an array. So, to change the number of checkboxes, nothing more is needed than to change the contents of this array.

The class is named Checkbar and it can be utilized like this in a test program:


```
def printstates():
    print (c.getstate())

root=tk.Tk()
txts=["CH0", "CH1", "All channels"]
c=Checkbar(root, txts)
c.pack()

b=tk.Button(root, text="Print", command=printstates)
b.pack()

root.mainloop()
```

The definition of the checkbar takes 3 lines:

- define the texts array for the check buttons
- create an instance of the Checkbar class
- pack it on the form

In the example a printstates function print the state of the buttons as array e.g. [0,1,1]

How do we define the class?

```
class Checkbar(tk.Frame):
    def __init__(self, parent=None, texts=[], side=tk.LEFT, anchor=tk.W):
        tk.Frame.__init__(self, parent)

        self.vars=[]

        for txt in texts:
            var = tk.IntVar()
            chk = tk.Checkbutton(self, text=txt, variable=var)
            chk.pack(side=side, anchor=anchor, expand=tk.YES)
            self.vars.append(var)
```

```
    def getstate(self):
        v = [var.get() for var in self.vars]
        return v
```

Every class definition starts with the class keyword.
In parenthesis is the base class for the new class, in this case a tk.Frame.
The new class “knows” all the functions of the class on which it is based.

The __init__ function is called when an instance of the class is created, in our example by the line
`c=Checkbar(root, txts)`

The most important parameter of the __init__ function is the array texts[], defining the check box labels.

Some other parameters are defined with default values, they can be omitted on defining the class, or they can get other values.

After initializing the Frame, an array self.vars[] is created that will contain the values of the check boxes. This is used by the self.getstates function.

The for loop creates the check boxes one after one, as many as there are labels in the text array.
For each one there is a Tkinter variable connected to the checkbox state: `var = tk.IntVar()`
This variable is appended to the self.vars array.

Variations

The Checkbar can easily be used in a vertical orientation by changing the side parameter, or by adding relief:

```
txts=["CH0", "CH1", "All channels"]
c=Checkbar(root, txts, side=tk.LEFT)
c.config(relief=tk.RIDGE, bd=3)
c.pack()
```