

Forum:[Hack Forums](#)

Thread: [Hotbar with tkinter](#)

MiguelSanchez - April 11, 2022, 8:51 am

A Tkinter hotbar for storing multiple lists of functions to easily scroll between them.

I haven't integrated keyboard hooks but I just recently did a demo so it won't take long, I'll copy/pastesome bits across.

The demo has 3 bars,
emptybar which is obviously an empty hotbar.
testbar has one item which prints "test" to the python shell.
testbar=bar("Test Bar",[item("print",lambda: print("test"))])
the last bar, Quick Launch, contains an example of the systemcommand function which simply makes an item that calls os.system.

I just quickly hacked this together over the lat hour or so, I've made beautiful versions of this idea in Racket and Emacs but I think my python skills will let me build something a lot nicer.

```
Code import tkinter as tk
from os import system
```

```
class bar:
```

```
    def __init__(self,name,items=[]):
        self.name=name
        self.items=items
```

```
class item:
```

```
    def __init__(self,name,func,col="#9A8"):
        self.name=name
        self.func=func
        self.type="NOT YET ADDED"
        self.col=col
```

```
class hotbar:
```

```
    def __init__(self,tilesize,bars=[emptybar],tiles=10,position="Bottom"):
        ##make window
        self.tiles=tiles
        self.window=tk.Tk()
        self.activebar=0
        self.bars=bars
        self.barlabel=tk.Label(text=self.bars[self.activebar].name)
        self.barlabel.pack(side=tk.RIGHT)
        self.nextbarbutton=tk.Button(command=lambda *args:self.changebar(1))
        self.nextbarbutton.pack(side=tk.RIGHT)
        self.previousbarbutton=tk.Button(command=lambda *args:self.changebar(-1))
        self.previousbarbutton.pack(side=tk.RIGHT)
        self.buttons=[tk.Button(height=int(round(tilesize*.5)),width=tilesize,command=lambda
*args:self.run(0)),tk.Button(height=int(round(tilesize*.5)),width=tilesize,command=lambda
```

```

*args:self.run(1)),
            tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(2)),tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(3)),
            tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(4)),tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(5)),
            tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(6)),tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(7)),
            tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(8)),tk.Button(height=int(round(tilesiz*.5)),width=tilesiz,command=lambda
*args:self.run(9)),
        ]
        for i in self.buttons:
            i.pack(side=tk.LEFT)
        self.drawbar()
        def mainloop(self):
            self.window.mainloop()
        def run(self,num):
            if num <len(self.bars[self.activebar].items):
                self.bars[self.activebar].items[num].func()
        def drawbar(self):
            ##Update Label
            self.barlabel["text"]=self.bars[self.activebar].name
            ##Update Buttons
            for i in range(self.tiles):
                if i <len(self.bars[self.activebar].items):

                    self.buttons[i].config(text=self.bars[self.activebar].items[i].name,bg=self.bars[self.activebar
].items[i].col)
                else:
                    self.buttons[i].config(text="UNASSIGNED",bg="#555")
        def changebar(self,direction):
            self.activebar=(self.activebar+direction)%len(self.bars)
            self.drawbar()

        def systemcommand(name,command):
            return item(name,lambda:system(command))

        emptybar=bar("Empty Bar")
        testbar=bar("Test Bar",[item("print",lambda: print("test"))])
        quicklaunch=bar("Quick Launch",[systemcommand("Shell","start cmd")])
        Main=hotbar(8,bars=[testbar,quicklaunch,emptybar])
        Once I've added hotkeys I'll do a re-release via my site http://trogramming.com which I'll share here.

```

My intention is to turn it into a macro manager, as well a bunch of "creation-wizards" I'll use for automating and streamlining development, in the future version everything will be strictly typed and everything will be chained together to form complex tasks from each mall component (for now no item has a return value, and therefore no type).

lightduty - April 12, 2022, 2:13 pm

(Apr 11, 2022 08:51 AM)MiguelSanchez Wrote: [\[CITE\]](#) A Tkinter hotbar for storing multiple lists of

functions to easily scroll between them.

I haven't integrated keyboard hooks but I just recently did a demo so it won't take long, I'll copy/pastesome bits across.

The demo has 3 bars,
emptybar which is obviously an empty hotbar.
testbar has one item which prints "test" to the python shell.
testbar=bar("Test Bar",[item("print",lambda: print("test"))])
the last bar, Quick Launch, contains an example of the systemcommand function which simply makes an item that calls os.system.

I just quickly hacked this together over the lat hour or so, I've made beautiful versions of this idea in Racket and Emacs but I think my python skills will let me build something a lot nicer.

Once I've added hotkeys I'll do a re-release via my site <http://trogramming.com> which I'll share here.

My intention is to turn it into a macro manager, as well a bunch of "creation-wizards" I'll use for automating and streamlining development, in the future version everything will be strictly typed and everything will be chained together to form complex tasks from each mall component (for now no item has a return value, and therefore no type).

Seems helpful, thanks.