# **Tkinter 组件详解（二）：Button**

### **简介**

****Button****（按钮）组件用于实现各种各样的按钮。****Button**** 组件可以包含文本或图像，你可以将一个 Python 的函数或方法与之相关联，当按钮被按下时，对应的函数或方法将被自动执行。

****Button**** 组件仅能显示单一字体的文本，但文本可以跨越多行。另外，还可以为其中的个别字符加上下划线（例如用于表示键盘快捷键）。默认情况下，tab 按键被用于在按钮间切换。

## ****何时使用 Button 组件？****

简而言之，Button 组件是用于让用户说：“干！”，通过按钮上的文字或图标让用户清楚按下去是干什么用的。Button 组件常常被用于工具栏、应用程序窗口、和表示接受或拒绝的对话框。****Checkbutton****和 ****Radiobutton****;

[Checkbutton](https://blog.csdn.net/qq_41556318/article/details/85108303" \t "https://blog.csdn.net/qq_38970783/article/details/_blank) 和 [Radiobutton](https://blog.csdn.net/qq_41556318/article/details/85108309" \t "https://blog.csdn.net/qq_38970783/article/details/_blank) 组件更适合做数据输入按钮使用。

## ****用法****

普通的按钮是非常简单易用的。你所需要做的就是指定 Button 的内容（文本、位图或者图片），并且关联当按钮被按下时应该调用的函数或方法：

2-1

import tkinter as tk

master = tk.Tk()

def callback():

    print("我被调用了！")

b = tk.Button(master, text="执行", command=callback)

b.pack()

master.mainloop()

运行结果：



>>> =============== RESTART===============

我被调用了！

如果一个按钮没有相关联的函数或方法，那么它就形同虚设。你可能在开发程序的过程中会使用到这样的按钮，在这种情况下，更好的方法是****禁用这些按钮****，从而避免导致你的测试用户产生疑惑。



如果你没有指定 \*\*Label \*\*的大小，那么 ****Label****的尺寸是正好可以容纳其内容而已。你可以使用 ****padx**** 和 ****pady**** 选项在 ****Button**** 的内容和边框间添加额外的间距。

当然你可以通过 ****height**** 和 ****width**** 选项来明确设置 ****Button**** 的大小：如果你显示的是文本，那么这两个选项是以文本单元为单位定义 \*\*Button \*\*的大小；如果你显示的是位图或者图像，那么它们以像素为单位（或者其他屏幕单元）定义 \*\*Button \*\*大小。

对于内容为文本的 ****Button**** 组件，你可以使用像素为单位指定 ****Buttton**** 的尺寸，不过这需要一些技巧。这里提供一个方法给大家参考（还有其它方法实现）：

2-2

import tkinter as tk

master = tk.Tk()

def callback():

    print("我被调用了！")

f = tk.Frame(master, height=64, width=64)

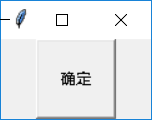
f.pack\_propagate(0)

f.pack()

b = tk.Button(f, text="确定", command=callback)

b.pack(fill="both", expand=1)

master.mainloop()



****Button**** 可以显示多行文本，你可以直接使用换行符或使用 ****wraplength**** 选项来实现。当文本换行的时候，你可以使用 ****anchor**** 和 ****justify**** 以及 ****padx**** 选项来使得文本如你所希望的显示出来：

2-3

import tkinter as tk

master = tk.Tk()

def callback():

    print("我被调用了！")

longtext = """

我明明只是一个按钮，

作为按钮并不需要太多

的文字用于告诉用户当

我被按下的时候会发生

什么事情，但我为什么

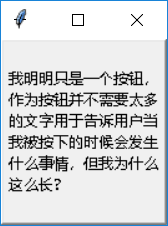
这么长？

"""

b = tk.Button(master, text=longtext, anchor="w", justify="left", padx=2, command=callback)

b.pack()

master.mainloop()

  
为了使一个普通的按钮保持被“按下”的状态，例如你希望通过某种方式实现一个工具箱（像 Photoshop 左侧的工具栏按钮，按下的时候他保持凹下去，说明你正在使用此工具），你可以简单的将 ****relief 选项的默认值 “raised” 改为 “sunken”****：

2-4

import tkinter as tk

master = tk.Tk()

def callback():

    print("我被调用了！")

longtext = """

我明明只是一个按钮，

作为按钮并不需要太多

的文字用于告诉用户当

我被按下的时候会发生

什么事情，但我为什么

这么长？

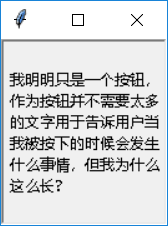
"""

b = tk.Button(master, text=longtext, anchor="w", justify="left", padx=2, command=callback)

b.pack()

b.config(relief="sunken")

master.mainloop()



你或许也希望修改背景色。不过更好的方法是使用 [Checkbutton](https://blog.csdn.net/qq_41556318/article/details/85108303" \t "https://blog.csdn.net/qq_38970783/article/details/_blank) 和 [Radiobutton](https://blog.csdn.net/qq_41556318/article/details/85108309" \t "https://blog.csdn.net/qq_38970783/article/details/_blank) 组件，并把它们的 indicatoron 选项设置为 False：

*b = tk.Checkbutton(master, image=bold, variable=var, indicatoron=False)*

在早期版本的 Tkinter 中，image 选项会覆盖 text 选项。也就是说如果你同时指定了两个选项，那么只有 image 选项会被显示。但在新的 Tkinter 中，你可以使用 compound 选项设置二者的混合模式。例如下边就是通过设置 compound="center" 使得文字位于图片的上方（重叠显示）：

2-5

import tkinter as tk

master = tk.Tk()

photo = tk.PhotoImage(file ='botton.png')

b = tk.Button(master, text="点我", font = 20, image = photo, compound = "center")

b.pack()

master.mainloop()



通过 ****“left”****，****“right”****，****“top”**** 和 ****“bottom”**** 则可以设置文字在图像的旁边显示：

2-6

import tkinter as tk

master = tk.Tk()

photo = tk.PhotoImage(file ='botton.png')

b = tk.Button(master, text="点它 ->", font = 20, image = photo, compound = "right")

b.pack()

master.mainloop()



### ****参数****

****Button(master=None, \*\*options)**** (class)

master -- 父组件

\*\*options -- 组件选项，下方表格详细列举了各个选项的具体含义和用法：

|  |  |
| --- | --- |
| ****选项**** | ****含义**** |
| activebackground | 1. 设置当 Button 处于活动状态（通过 state 选项设置状态）的背景色 2. 默认值由系统指定 |
| activeforeground | 1. 设置当 Button 处于活动状态（通过 state 选项设置状态）的前景色 2. 默认值由系统指定 |
| anchor | 1. 控制文本（或图像）在 Button 中显示的位置  2. "n", "ne", "e", "se", "s", "sw", "w", "nw", 或者 "center" 来定位（ewsn 代表东西南北，上北下南左西右东） 3. 默认值是 "center" |
| background | 1. 设置背景颜色 2. 默认值由系统指定 |
| bg | 跟 background 一样 |
| bitmap | 1. 指定显示到 Button 上的位图 2. 如果指定了 image 选项，则该选项被忽略 |
| borderwidth | 1. 指定 Button 的边框宽度 2. 默认值由系统指定，通常是 1 或 2 像素 |
| bd | 跟 borderwidth 一样 |
| compound | 1. 控制 Button 中文本和图像的混合模式 2. 默认情况下，如果有指定位图或图片，则不显示文本  3. 如果该选项设置为 "center"，文本显示在图像上（文本重叠图像） 4. 如果该选项设置为 "bottom"，"left"，"right" 或 "top"，那么图像显示在文本的旁边（如 "bottom"，则图像在文本的下方） 5. 默认值是 NONE |
| cursor | 1. 指定当鼠标在 Button 上飘过的时候的鼠标样式 2. 默认值由系统指定 |
| default | 1. 如果设置该选项（"normal"），该按钮会被绘制成默认按钮 2. Tkinter 会根据平台的具体指标来绘制（通常就是绘制一个额外的边框） 2. 默认值是 "disable" |
| disabledforeground | 1. 指定当 Button 不可用的时候前景色的颜色 2. 默认值由系统指定 |
| font | 1. 指定 Button 中文本的字体 2. 一个 Button 只能设置一种字体 3. 默认值由系统指定 |
| foreground | 1. 设置 Button 的文本和位图的颜色 2. 默认值由系统指定 |
| fg | 跟 foreground 一样 |
| height | 1. 设置 Button 的高度 2. 如果 Button 显示的是文本，那么单位是文本单元 3. 如果 Button 显示的是图像，那么单位是像素（或屏幕单元） 4. 如果设置为 0 或者干脆不设置，那么会自动根据 Button 的内容计算出高度 |
| highlightbackground | 1. 指定当 Button 没有获得焦点的时候高亮边框的颜色 2. 默认值由系统指定 |
| highlightcolor | 1. 指定当 Button 获得焦点的时候高亮边框的颜色 2. 默认值由系统指定 |
| highlightthickness | 1. 指定高亮边框的宽度 2. 默认值是 0（不带高亮边框） |
| image | 1. 指定 Button 显示的图片 2. 该值应该是 PhotoImage，BitmapImage，或者能兼容的对象 3. 该选项优先于 text 和 bitmap 选项 |
| justify | 1. 定义如何对齐多行文本  2. 使用 "left"，"right" 或 "center" 3. 注意，文本的位置取决于 anchor 选项 4. 默认值是 "center" |
| overrelief | 1. 定义当鼠标飘过时 Button 的样式 2. 如果不设置，那么总是使用 relief 选项指定的样式 |
| padx | 指定 Button 水平方向上的额外间距（内容和边框间） |
| pady | 指定 Button 垂直方向上的额外间距（内容和边框间） |
| relief | 1. 指定边框样式 2. 通常当按钮被按下时是 "sunken"，其他时候是 "raised" 3. 另外你还可以设置 "groove"、"ridge" 或 "flat" 4. 默认值是 "raised" |
| repeatdelay | 见下方 repeatinterval 选项的描述 |
| repeatinterval | 1. 通常当用户鼠标按下按钮并释放的时候系统认为是一次点击动作。如果你希望当用户持续按下按钮的时候（没有松开），根据一定的间隔多次触发按钮，那么你可以设置这个选项。 2. 当用户持续按下按钮的时候，经过 repeatdelay 时间后，每 repeatinterval 间隔就触发一次按钮事件。 3. 例如设置 repeatdelay=1000，repeatinterval=300，则当用户持续按下按钮，在 1 秒的延迟后开始每 300 毫秒触发一次按钮事件，直到用户释放鼠标。 |
| state | 1. 指定 Button 的状态 2. 默认值是 "normal" 3. 另外你还可以设置 "active" 或 "disabled" |
| takefocus | 1. 指定使用 Tab 键可以将焦点移到该 Button 组件上（这样按下空格键也相当于触发按钮事件） 2. 默认是开启的，可以将该选项设置为 False 避免焦点在此 Button 上 |
| text | 1. 指定 Button 显示的文本 2. 文本可以包含换行符 3. 如果设置了 bitmap 或 image 选项，该选项则被忽略 |
| textvariable | 1. Button 显示 Tkinter 变量（通常是一个 StringVar 变量）的内容 2. 如果变量被修改，Button 的文本会自动更新 |
| underline | 1. 跟 text 选项一起使用，用于指定哪一个字符画下划线（例如用于表示键盘快捷键）  2. 默认值是 -1 3. 例如设置为 1，则说明在 Button 的第 2 个字符处画下划线 |
| width | 1. 设置 Button 的宽度 2. 如果 Button 显示的是文本，那么单位是文本单元 3. 如果 Button 显示的是图像，那么单位是像素（或屏幕单元） 4. 如果设置为 0 或者干脆不设置，那么会自动根据 Button 的内容计算出宽度 |
| wraplength | 1. 决定 Button 的文本应该被分成多少行 2. 该选项指定每行的长度，单位是屏幕单元 3. 默认值是 0 |

## ****方法****

****flash()****

-- 刷新 Button 组件，该方法将重绘 Button 组件若干次（在 "active" 和 "normal" 状态间切换）。

****invoke()****

-- 调用 Button 中 command 选项指定的函数或方法，并返回函数的返回值。  
-- 如果 Button 的state(状态)是 "disabled"（不可用）或没有指定 command 选项，则该方法无效。