Overview

A `checkbutton` widget is like a regular button that also holds a binary value of some kind (i.e., a toggle). When pressed, a checkbutton flips the toggle and then invokes its callback. Checkbutton widgets are frequently used to allow users to turn an option on or off.

Important Information

Creating a Checkbutton

Checkbuttons are created using the `ttk.Checkbutton` class. Typically, their contents (text or textvariable) and behavior (a number of configurations that will be covered more in depth throughout this lesson) are specified at the same time.

The following example is a `CheckButton` that toggles whether a metric system or imperial system should be used for measurement:

```
# Create a variable to track which measurment system is selected
measureSystem = StringVar()
# Create a checkbutton with multiple settings
# parent represents whatever container you want to put it in. This can be
the root, or a frame.
check = ttk.Checkbutton(parent, text='Use Metric', command=metricChanged,
variable=measureSystem, onvalue='metric', offvalue='imperial')
```

Checkbuttons use many of the same options as regular buttons but add a few more. The `text`, `textvariable`, `image`, and `compound` configuration options control the display of the label (next to the checkbox itself).

Similarly, the `command` option lets you specify a command to be called every time a user toggles the checkbutton; and the `invoke` method will also execute the same command.

The `state` and `instate` methods allow you to manipulate the `disabled` state flag to enable or disable the checkbutton.

Checkbutton Value

Unlike regular buttons, checkbuttons also hold a value. We've seen how the `textvariable` option links the label of a widget to a variable. The `variable` option for checkbuttons behaves similarly, except it links a variable to the widget's current value. The variable is updated whenever the widget is toggled.

```
# Create the checkbutton
check = ttk.Checkbutton(root, text='Check box:')
# Create a variable to track the value of the checkbutton
check_var = IntVar()
# Assign the variable to the checkbutton
check['variable'] = check_var
# Get the value of check_var (the current value of the checkbutton)
value = check_var.get()
# Set the checkbutton value
check_var.set(1)  # check the box
check_var.set(0)  # uncheck the box
```

By default, checkbuttons use a value of `1` when checked and `0` when not checked. These can be changed to something else using the `onvalue` and `offvalue` options.

For example:

```
# Create the checkbutton
check = ttk.Checkbutton(root, text='Check box:')
# Normally the checkbutton onvalue is equal to 1
# However, we have changed the value of the checkbutton to 'checked' when
the checkbox is clicked
check['onvalue'] = 'checked'
# Normally the checkbutton offvalue is equal to 0
# However, we have changed the value of the checkbutton to 'unchecked'
when the checkbox is not clicked
check['offvalue'] = 'unchecked'
```

Note: When changing the `onvalue` and `offvalue` of a checkbutton, the variable that tracks the value will need to be changed to match. In the above example, instead of being an `IntVar()`, it will need to be changed to a `StringVar()`.

A checkbutton doesn't automatically set (or create) the linked variable. Therefore, your program needs to initialize it to the appropriate starting value.

What happens when the linked variable contains neither the `onvalue` or the `offvalue` (or even doesn't exist)? In that case, the checkbutton is put into a special "tristate" or indeterminate mode. The checkbox might display a single dash in this mode instead of being empty or holding a checkmark. Internally, the state flag alternate is set, which you can inspect via the `instate` method:

```
# Check the current state of the button to see if the checkbutton is in an
intermediate state
# Returns True if the button is in an alternate state
# Returns False if the button is not in an alternate state
check.instate(['alternate'])
```

Other Variable Types For Variables Attached to Widgets

So far, we've been using an instance of the `StringVar` class to attach variables to widgets. Tkinter provides other variable classes that can hold booleans, integers, or floating-point numbers. You can always use a `StringVar` (because the Tcl API that Tkinter uses is string-based) but can choose one of the others if the data stored in it fits the type. All are subclasses of the base class `Variable`.

```
# Create a string variable
s = StringVar()
# Create a boolean variable
b = BooleanVar()
# Create an integer variable
i = IntVar()
# Create a decimal variable
d = DoubleVar()
```

Note: Normally when working with decimals in python we have used `float` as a data type. In the `tkinter` library, it is instead called a `double`, but it still represents a decimal number.

We can also declare these variables with a starting value:

```
s = StringVar(value="abc")   # default value is ''
b = BooleanVar(value=True)   # default is False
i = IntVar(value=10)         # default is 0
d = DoubleVar(value=10.5)    # default is 0.0
```

Copy, Change, Challenge

Copy

```
from tkinter import *
from tkinter import ttk

root = Tk()
root.title("Checkbutton Copy Example")

# Create a variable to track the checkbutton state
check_var = IntVar(value=0)

def on_toggle():
    print("Checkbutton value:", check_var.get())

# Create the checkbutton
check = ttk.Checkbutton(
    root,
    text="Enable option",
    variable=check_var,
    command=on_toggle
)

# Add the checkbutton to the window
check.grid()

root.mainloop()
```

Change

Modify the program so that:

- The checkbutton starts **checked** when the program launches
- The text of the checkbutton is changed to **"Use metric system"**
- The value printed to the terminal clearly states whether the option is **ON** or **OFF** instead of just printing `0` or `1`

Challenge

Create a program with **two checkbuttons**:

- The first checkbutton should:
    - Use a `BooleanVar`
    - Control an option such as **"Enable sound"**
- The second checkbutton should:
    - Use a `StringVar`
    - Have custom `onvalue` and `offvalue` (for example `"enabled"` and `"disabled"`)
- When either checkbutton is toggled:
    - Print the current values of **both** variables to the terminal