CS 115, Autumn 2021

Lecture 34: GUIs

Thanks to Marty Stepp and Stuart Reges for parts of these slides

Exercise

- Create a guessing game program that:
  - Allows the user to guess a random number
  - Tells them when they are correct or if they are too high or too low
  - Allows them to restart the game

Controls

- We have discussed buttons and labels. Do we need anything else?
  - An input the user can type in

entry_box

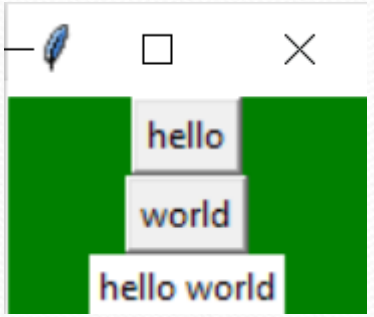
- Appears on the GUI as a white box 
  - The user can enter input into it (unlike a label)
  - Access the text the user types in an `entry_box` with:

    entry_box_variable_name`.get()`

Alignment

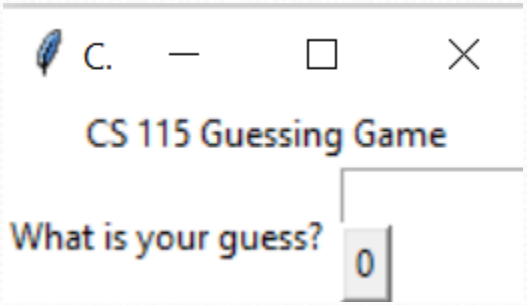
- So far all of our GUI elements have appeared in a column. How can we change their position?
  - Add a `side` parameter when we create them:

    values: `"left"`, `"right"`, `""`
    - Represents the direction the element will move to make room for the next one



Multiple Rows

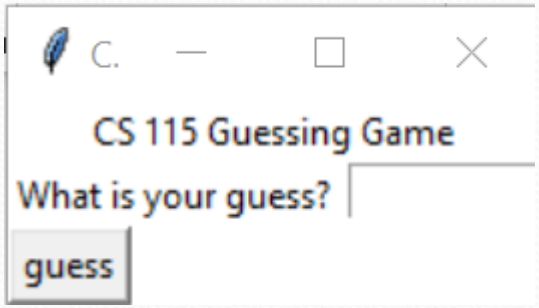
- Problem: once we move something left, it moves left of everything!
  - What if we want two separate rows?




```
def main():
    window = make_window("Guessing Game")
    add_label(window, "CS 115 Guessing Game")
    add_label(window, "What is your guess? ", side="left")
    input = add_entry_box(window)
    guess = add_button(window, "0", side="left")
    window.mainloop()
```

Frames

- If we want to create more complex layouts we can group elements together with frames
  - add to a frame just as you would a window
  - example:




```
def main():
    window = make_window("Guessing Game")
    add_label(window, "CS 115 Guessing Game")
    entry_line = add_frame(window)
    add_label(entry_line, "What is your guess? ", side="left")
    input = add_entry_box(entry_line)
    guess = add_button(window, "0", side="left")
    window.mainloop()
```

Adding Space

- Our GUI looks alright, but everything is really close together. How can we leave space?
  - Pass the following parameters to any control or frame to add space around it:

```
padding_top
padding_bottom
padding_left
padding_right
```

Adding events

- Set a button to react when it is clicked by setting its command:

  button_variable`['command'] =` **function_name**

  Make sure not to include () after your function name!
  - That will call the function right now. You want it to be called when the action happens

Example:

```
from ECGUI import *
def print_something():
    print("hello world!")
window = make_window()
hello_button = add_button(window, "hello")
hello_button['command'] = print_something
window.mainloop()
```

Parameters and events

- What happens if we write the following?

```
hello['command'] = print_something("hi")
```

  - This calls the function `print_something` passing it `"hi"` right now – it doesn't set it to be called when `hello` is clicked

  - How can we pass parameters without parentheses?
    - We can't so we need lambda notation

Lambda functions

- Lambda functions – small anonymous function that can be stored in variables
  - You can call another function from within one and have access to the variables in the function you declare it in
  - Example:

```
from ECGUI import *

def print_something(name):
    print("hello " + name.get())
def main():
    window = make_window()
    hello = add_button(window, "hello")
    name = add_entry_box(window)
    hello['command'] = lambda: print_something(name)
    window.mainloop()
```