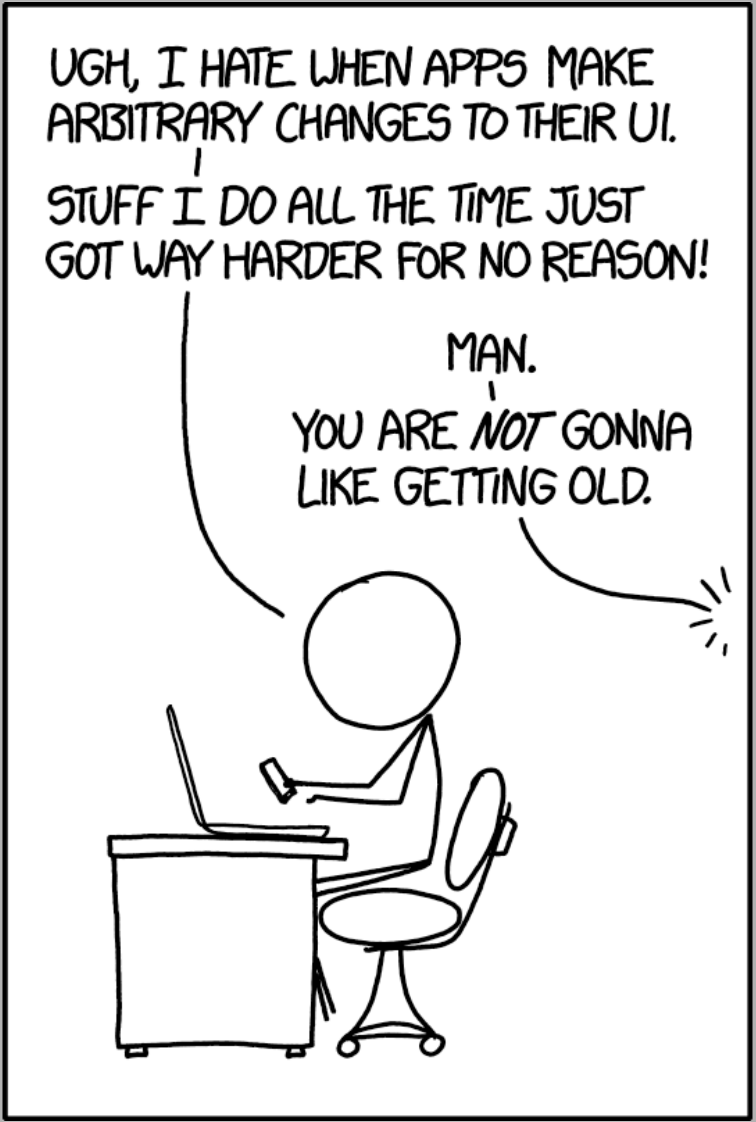
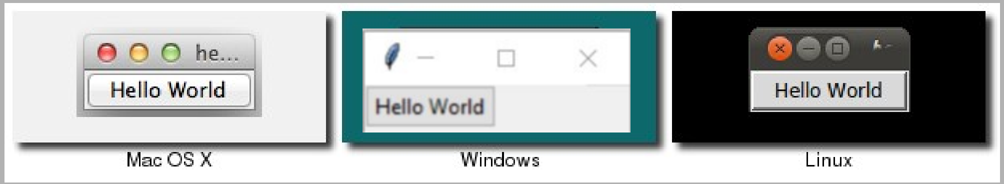
Give them a GUI with tkinter & ttk


UGH, I HATE WHEN APPS MAKE ARBITRARY CHANGES TO THEIR UI.
STUFF I DO ALL THE TIME JUST GOT WAY HARDER FOR NO REASON!
MAN.
YOU ARE *NOT* GONNA LIKE GETTING OLD.

Feb 28, 2019
DerbyPy Monthly Meetup
Kurt Strecker

tkinter and ttk

- A cross-platform UI tool kit. Users on Mac and/or Windows and/or Linux will get native looking apps with little to no customization required



- ttk (themed tool kit) extends tkinter with additional widgets and more modern UI themes (Windows 10, OSX, etc)

Why tkinter & ttk

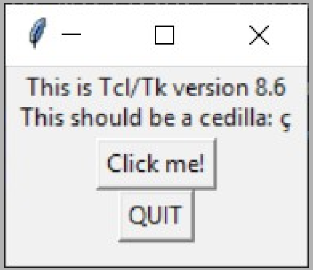
- tkinter is included in the standard library, no additional dependencies, installations, or tool kits to manage
- It’s fairly high level, making it easy to plan and design before writing any code.

First code!

- If you’re ever in doubt about the availability of tkinter on your system...

```
>> import tkinter
>> tkinter._test()
```

->


This is Tcl/Tk version 8.6
This should be a cedilla: ç

But why a GUI

- Honestly, it depends on…
  - The use case
  - The end users
  - Personal preference
- At the end of the day, it’s a powerful tool to keep in your back pocket.

tkinter, Behind the Scenes

- tkinter is python’s interface to Tcl/Tk
- Tcl (Tool Command Language), 42nd most popular programming language in 2018
- Tcl/Tk do the heavy lifting required to draw windows to the screen

```
YourApp.py → tkinter.Widget() → Tcl + C →
OS specific window system
```

Widgetdex

- Tkinter
  - Canvas
  - Frame
  - Label
  - Labelframe
  - Listbox
  - Menu
  - Message
  - Panedwindow
  - Radiobutton
  - Scrollbar
  - Spinbox
  - Dialog
  - MessageBox
  - OptionMenu
  - Popup
- ttk
  - Button
  - Checkbutton
  - Combobox
  - Entry
  - Frame
  - Label
  - Labelframe
  - Notebook
  - Panedwindow
  - Progressbar
  - Radiobutton
  - Scale
  - Scrollbar
  - Seperator
  - Sizegrip
  - Spinbox
  - treeview

Minimum Viable GUI

```
import tkinter as tk
from tkinter import ttk
```

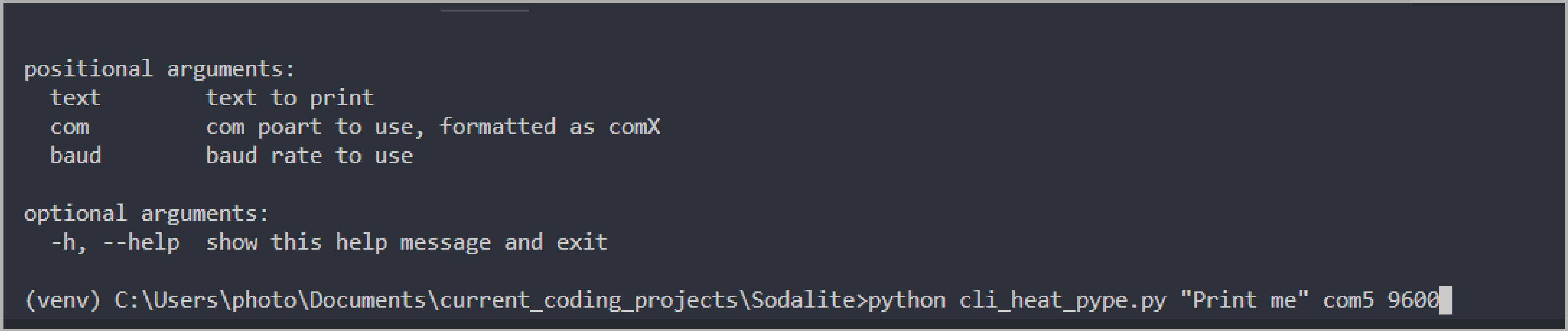
- Every tkinter app needs a root
  - `root = tk.Tk()`
- Every root should have a main frame
  - `mainFrame = ttk.Frame(root)`
- Every widget’s first argument is it’s parent
  - `label = ttk.Label(mainFrame, text=”Hello World”)`
- Button
  - `Button = ttk.Button(mainFrame, text=”Click Me”)`
- `root.mainloop()`

Geometry Managers

- tk.Widget.pack() == Flexbox
  - Great for quick and easy filling of space, auto-sizing, low specificity
  - Ideal for formatting on a single axis – rows OR columns
  - Set container as a row by passing side=”left” to first child’s .pack()
- tk.Widget.grid() == Grid
  - Easy to organize across an entire Frame
  - “Traditional” GUI layout system
- tk.Widget.place() == Explicit Absolute and Relative Placement
  - The most specific
  - Best suited for edge cases and special uses

Case Study

Replace this...

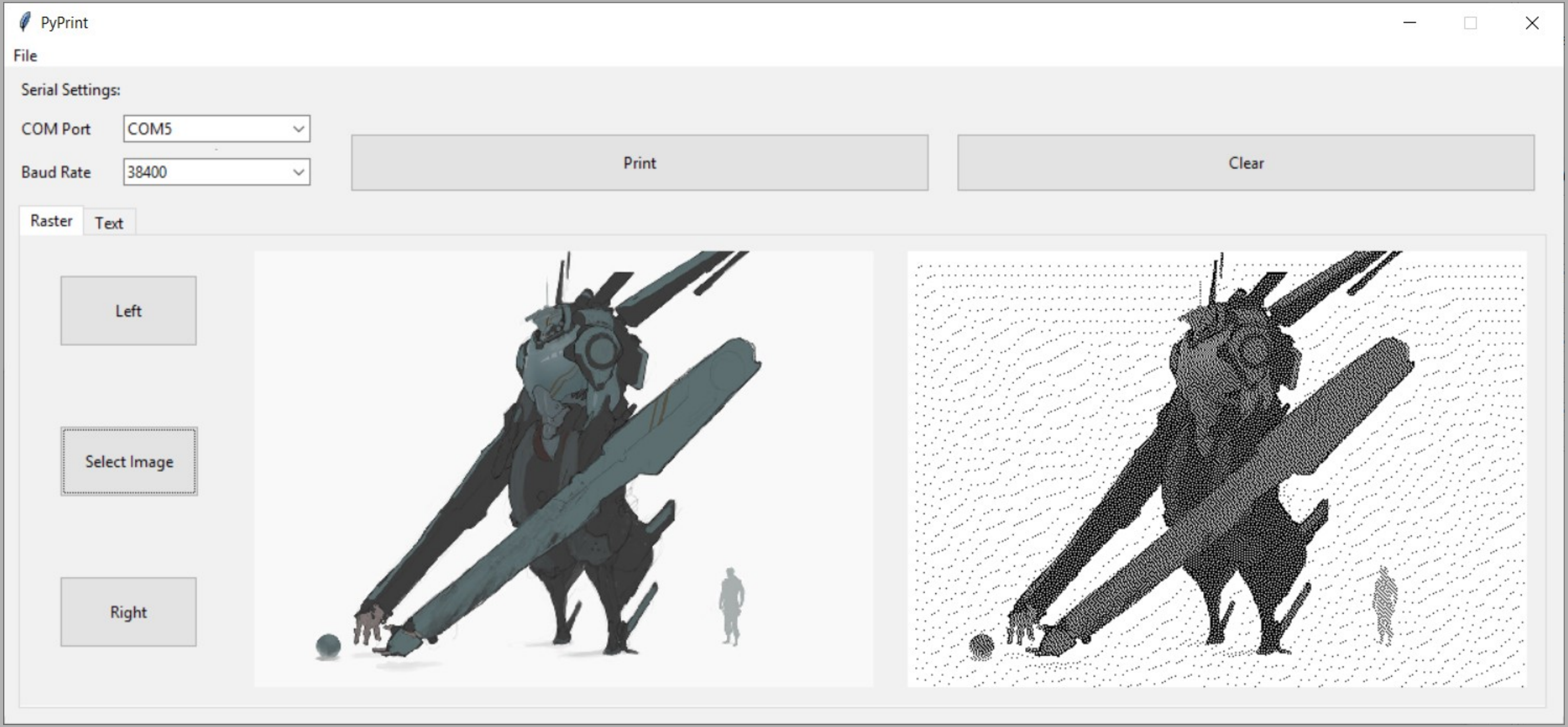


```
positional arguments:
  text        text to print
  com         com poart to use, formatted as comX
  baud        baud rate to use

optional arguments:
  -h, --help  show this help message and exit

(venv) C:\Users\photo\Documents\current_coding_projects\Sodalite>python cli_heat_pype.py "Print me" com5 9600
```

With this



PyPrint

Serial Settings:

COM Port: COM5

Baud Rate: 38400

Phase 1

- Entry
  - Returns its text via Entry.get(“1.0”, tk.END) or by binding to a TextVariable() instance
- Button
  - Command
    - Set to a function reference or a lambda
- row/columnconfigure
  - Weight determines the distribution of extra space
  - Always define a row and column with weight=1 to avoid unexpected formatting
- Text
  - Multi row text widget
  - Must use Text.get()and Text.set(), can’t bind to a TextVariable()

Tkinter variable classes

- BooleanVar, DoubleVar, IntVar, StringVar
  - Allow two-way binding via a .get() and .set() interface
  - Mind your Doubles and Ints, even if it isn’t very pythonic

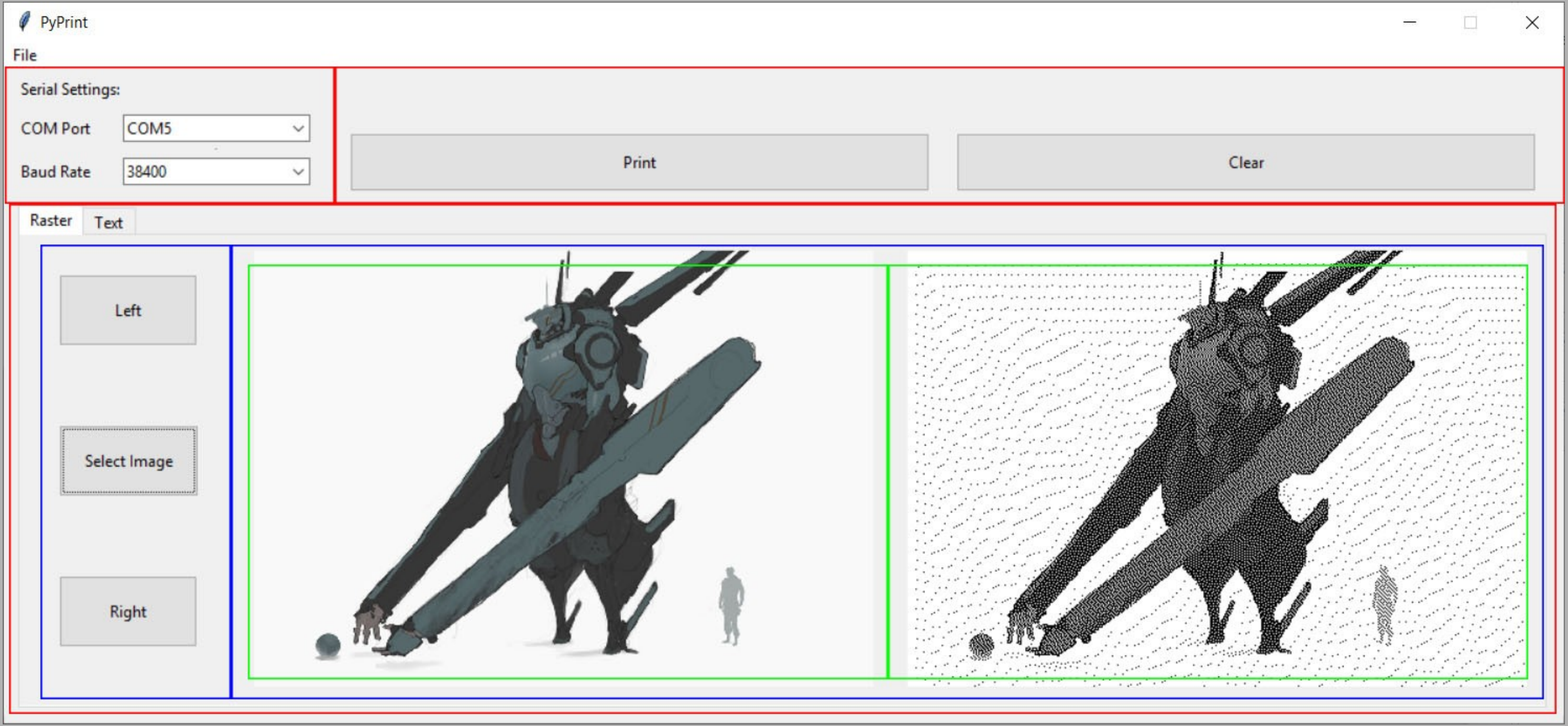
Phase 2

- Combobox
  - `values=[values]`
  - `state=[”readonly”|”disabled”]`
  - `On change binding`
- Notebook
  - Tabbed Interface
    - Each tab is its own frame
    - Labels added with Notebook.add(associated_frame, text=”Title”)

Canvas

- Extremely versatile
  - Several libraries provide compatible objects (matplotlib, Pillow)
  - Draw shapes directly with create_line(), create_rectangle, etc.
  - Each item placed on the canvas can be accessed directly via the option tag argument in their create method.

It’s Frames all the way down



PyPrint

File

Serial Settings:

COM Port: COM5

Baud Rate: 38400

Phase 3

- Get Organized
  - Roll your own widgets by subclassing them
  - Pass the parent to __init__() and then super().__init__()
  - Pass **kwargs to self.grid()

- Minimize global variable pollution
  - Think of your Frames as Components
    - Each has it’s own local variables
    - Pass function references from main_frame as needed

Polish

- Menu Bar
  - Most awkward hierarchy in all of tkinter

```
#add a blank menubar to the window
    menubar = Menu(root)
    root['menu'] = menubar

#instantiate menu items
    menu_file = Menu(menubar)
    menu_edit = Menu(menubar)

#add menu items to the bar
    menubar.add_cascade(menu=menu_file, label='File')
    menubar.add_cascade(menu=menu_edit, label='Edit')

#add cascade items to the menu items
    menu_file.add_command(label=”Quit”, command=exit)
```

Polish

- Settings Dialog
  - You can spawn a new window by calling tk.Tk() within a function
  - You can pass settings through global functions or references
  - Settings window can be closed on apply/slave by calling .destroy()

Polish

- Icons
  - `root.iconbitmap(os.path.join(‘path’,’to’,’ico file’)`
- Bindings
  - Widget events
    - `comboExample.bind("<<ComboboxSelected>>", callbackFunc)`
  - Keyboard shortcuts
    - `root.bind('<Control-q>', exit)`
    - `root.bind('<F1>', self.print_it)`

Documentation

- Best Documentation
  - Python’s own docs cover tkinter and ttk
  - https://tkdocs.com/
  - https://www.tcl.tk/

Conclusion.

Questions?

Thank you!

Email kurt@kurtstrecker.io
Twitter @KurtStrecker