Dev Seminar

GUI Review

Admin

- Please read chapter 2 in the pragmatic programmer
- We’ll have another quiz next week as well.
- This bit will be a review for some of you, but enough of you said you said you haven't covered this recently that I’ll cover this briefly

Sprint 2 retrospective

- Depending on the section this might be the time to do the retrospective

Programming paradigms

- Lets have someone summarize each:
  - Imperative
  - Object Oriented (Imperative OO)
  - Event-Driven
  - Functional
  - Logic (ok this one is obscure so we might skip it)

Programming paradigms

- Lets have someone summarize each:
  - Imperative
  - Object Oriented (Imperative OO)
  - Event-Driven
  - Functional
  - Logic (ok this one is obscure so we might skip it)
- Which one do we need for GUIs?

GUIs

- Graphical User Interfaces
- Programs with windows and user driven input
  - keyboard/mouse, touch, other?
- These programs are event driven by nature
  - No one can predict what users will do.
  - Your program just has to react

GUI Programs

- What this means
  - You will create classes
  - You will write methods
  - You will never call some of those methods
    - The system will call them for you
    - You set your program loose to run, and it reacts.

GUI Libraries

- There are many (many) GUI libraries out there
  - Java has fewer choices than most since standard library at one time included AWT, Swing, and JavaFX.
  - Python never had a lib in standard lib so there are lots of widely used libraries
  - https://github.com/vinta/awesome-python#gui-development
  - I had to pick one so I did.
    - You can pick another one if you like
    - So long as it works on my machine

GUI Demo Walkthrough

- I'll do a super simple walk-through in python with pyside the official qt for python library from the qt people https://wiki.qt.io/Qt_for_Python
- For those of you doing java and javaFX (none??) here is a link to a comp152 demo that I worked through with a comp152 class last semester ago that might give you a decent example to work from:
- https://github.com/jsantore/WindowWebData
-

Credit

- Resources that I used to build this demo:
  - https://medium.com/weekly-python/getting-started-writing-qt-6-applications-in-python-with-pyside6-389ee4c384ee
  - https://doc.qt.io/qtforpython/PySide6/QtWidgets/QListWidget.html
  - https://www.pythonguis.com/pyside6-tutorial/
- In this quick demo I'm doing everything in code, but you can absolutely use QML to do the layout of the GUI

GUI Framework key

- Key to use of any gui framework:
  - Framework provides classes/structs for appearance and events it will handle
  - You provide code for what happens when those events fire
- How does java let you provide this code?
- How does python/C/C++ let you provide this code?

GUI Framework key

- Key to use of any gui framework:
  - Framework provides classes/structs for appearance and events it will handle
  - You provide code for what happens when those events fire
- How does java let you provide this code?
  - Using interfaces that you implement
- How does python/C/C++ let you provide this code?
  - Using callback functions
- And Java?

Lets walk through a demo

- Have a look at this simple demo
- I’ll post the link in slack
- Lets review a little bit of python
- And see how to do this in qt6/pyside6
- Here is my demo if you want to follow along when we get there:
- https://github.com/jsantore/GUIDemo

QTApplication

- A QTApplication is the overall program
  - Qt6 and its python wrapper wants one, then you can run as many windows as you want while it is active.

```
qt_app = PySide6.QtWidgets.QApplication(sys.argv)
```

Do stuff and build your windows here

```
sys.exit(qt_app.exec_())
```

  - The qt_app.exec() starts the event loop
  - Exit won't be called till exec ends.

Retained Mode

- Retained Mode vs Immediate Mode GUIs
- PyQt6/Pyside6 is a retained mode GUI
  - Extend a class which holds window state.
  - QTWidget in this case.

```
class Comp490DemoWindow(QWidget):
```

  - Remember all python methods (as opposed to functions) have self as the first parameter.

Signals and Slots

- Pyside uses the terms signals and slots for event handling
  - Signals are the events
  - Slots are the places to put your event handlers.
  - For example, buttons have a clicked ‘signal’
    - And a slot of handle it. Stuff a signal hander into a slot, eg:

```
comp490_demo_button.clicked.connect(self.do_something_to_demo)
```

    - You must pass the function, not call the function do not use the ()

Beware the GC

- QT windows need to be kept in instance variables
  - Or sometimes just local variable
  - self.blah = new_window
  - Or the window never shows up.
  - That is why my
    - GuiDemo.py line 11 has the unused variable
    - And DemoWindow line 47 stores the map window into a variable
    - And the 'unused' list_item variable in line 38 of demo window.

Lets look

- Lets look through a small demo
  - https://github.com/jsantore/GuiDemo2022
  - Originally from 2 years ago when pyside6 was released, edited for changes including maps added this year.