Organization: Python Software Foundation
Sub-organization: FURY

FURY: Create new UI widget

FURY - Software Library for Scientific Visualization in Python

Contents

About Me

Brief Introduction

Hello this is Soham Biswas currently in 2nd year pursuing my Bachelor's(B.Tech) degree in Computer Science & Engineering from Institute of Engineering & Management, Kolkata.

I have been a Python developer for almost 3 years now and have worked on several major python frameworks such as Numpy, Pandas, PyQt, PyGTK, Django etc. I have built multiple personal and internship assignment projects using python and related frameworks. I have experience working in the field of Cyber Security and Data Science and would like to explore further into the field of Computer Graphics.

I am well experienced in Python, C and shell scripting. I am also well versed in other programming languages such as C++ and Java. But my programming language of choice has always been Python. I have research experience in the field of Computer Vision and UI development using Python.

I have an introductory experience with VTK, which I plan to improve soon since I have previously worked with the library while contributing and experimenting with Project FURY.

Contact Information

Name	Soham Biswas
Country	India
College	Institute of Engineering & Management, kolkata.
Degree (Pursuing)	Bachelor of Technology(Computer Science & Engineering)
Current Year	2nd Year
Expected Graduation Date	April 2022
Email	
Timezone	Indian Standard Time(IST)
Github	https://github.com/Nibba2018
LinkedIN	https://www.linkedin.com/in/soham-biswas-590784168/
Slack (FURY)	@Soham Biswas
Resume	

Code and Project Contributions

I have deployed and worked on open issues in the FURY github repo. The PRs that I have worked on are as follows:

1. #165 - Vertical Layout for LineSlider2D
   It was an **Enhancement** PR in which I enhanced LineSlider2D ui component to support vertical layout. I modified Tests and tutorials for the newly modified UI component. This PR closed Issue #108.

2. #173 - Fixing Text Overflow of ListBox2D
   It was a **Bug Fix** PR in which the Text Overflow bug was fixed. Usually texts which were longer than the width of the ListBox would overflow over the UI element and this PR fixed that problem. This PR closed Issue #15 and #166 which were indirectly related to each other.

3. #181 - Vertical Layout for LineDoubleSlider2D
   It was an **Enhancement** PR in which I enhanced LineDoubleSlider2D ui component to support vertical layout. I modified Tests and tutorials for the newly modified UI component. This PR closed issue #175. This PR was also responsible for fixing a Bug in which the handles were not aligned properly.

4. #204 - Vertical Layout for RangeSlider
   It was an **Enhancement** PR in which I enhanced the RangeSlider ui component to support vertical layout. I modified Tests and tutorials for the newly modified UI component.

5. #210 - Added contour_from_label method
   It was an **Enhancement** PR in which I implemented a method which would create contours from a labelled array and would return the resulting contours in a vtkAssembly instance. I also created unit tests and tutorials for the same. This PR closes issue #77.

The Issues that I have raised are as follows:

1. #159 - MouseWheelForwardEvent FAILED
   This Issue was related to FURY tests where the MouseWheelForward Event would always fail for an unknown BUG.
2. #160 - Saved Images are vertically Inverted
   This Bug was related to the inverted saving of Images by the ShowManager instance.
3. #199 - Loading of Inverted Icons using read_viz_icons
   This Bug was related to the inverted display of icons when icons are loaded using read_viz_icons.

Project Information

FURY

Software Library for scientific visualization in Python.

1. Project Abstract

As mentioned in the Ideas List, I would be working on building scifi-like 3D and 2D interfaces inspired from the “Guardians of the Galaxy” movie. In this project my objective would be to add more futuristic widgets and make UI elements more interactive without hampering performance.

Also I would like to develop a ComboBox UI element for the UI component of `fury.ui`, improve FileDialog capabilities and add a TAB UI component.

As *Stretch Goals* I would also like to develop Spinner, Accordion and Tree UI components.

2. Detailed Description

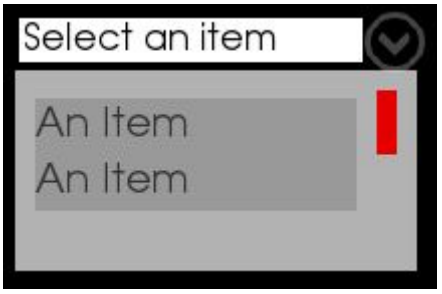
1. ComboBox UI element:

A combo box is a commonly used graphical user interface widget. Traditionally, it is a combination of a drop-down list or list box and a single-line editable textbox, allowing the user to either type a value directly or select a value from the list. The term "combo box" is sometimes used to mean "drop-down list".

The mentioned UI element can be created by the following existing UI components:

- `ui.TextBox2D` -> For the editable text field.
- `ui.Button2D` -> For opening the drop-down list.
- `ui.ListBox2D` -> For displaying available options.

A quick example of such as an arrangement can be as follows:



2. FileDialog capability improvements:

1. Horizontal Slider for File Dialog:

Currently File Dialog supports only vertical scrollbar. It would be great if a horizontal scrollbar could be introduced along with it. It will be helpful as we can display more information along with the file names such as size of the files, date modified etc.

2. Adding Buttons for additional functionality:

Adding additional buttons such as “Save”, “Open”, “Cancel” etc will help improve the functionality of the UI element. Currently, FileDialog only supports Left Clicking a particular file or folder to access it.

3. Editable Text Field for quick access:

We can provide an editable text field so that the user can directly type the file/folder name that they are looking for.

3. Tab UI component:

In interface design, a Tab is a graphical control element that allows multiple documents or panels to be contained within a single window, using tabs as a navigational widget for switching between sets of such windows.

Similar Tab like UI component can be built using the following existing UI components:

- `ui.Panel2D` -> For containing the elements within a Tab.
- `ui.TextBlock2D` -> A reference for users to change tabs.
- `ui.Button2D` -> For Closing Tabs.
- `ui.Button2D` -> For creating new Tabs.

A quick example of such an arrangement can be as follows:

3. Stretch Goals

These are the extra elements or components that I would like to work on once I am done with my main goals for the event. These stretch goals will be worked on if I have extra time left for the event after the completion of my necessary objectives.

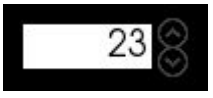
1. Spinner UI widget:

Value input control which has small up and down buttons to step through a range of values. This can be useful when we require discrete control over a particular range of value. Getting discrete values can be difficult while using line sliders or double line sliders.

Spinner UI component can be built using the following existing UI components:

- `ui.TextBox2D` -> for an editable text field to display selected values.
- `ui.Button2D` -> for increasing the value displayed in TextBox.
- `ui.Button2D` -> for decreasing the value displayed in TextBox.

A quick example of such an arrangement can be as follows:



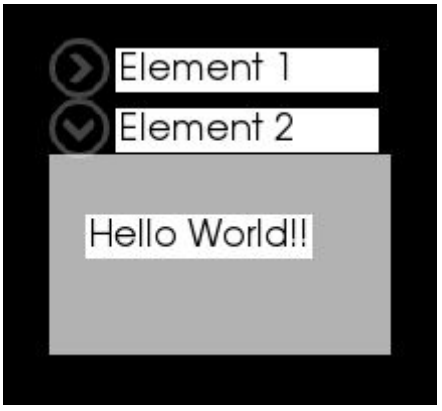
2. Accordion UI Widget:

A vertically stacked list of items, such as labels or thumbnails where each item can be "expanded" to reveal the associated content. This UI widget can be useful in categorizing various Menu elements. One such implementational example can be as such where we subdivide the menu of `viz_ui.py` example further. For e.g we can divide “Line & Ring Sliders” further into “Line Slider”, “Range Slider” and “Ring Slider”.

Accordion UI component can be built using the following existing UI components:

- `ui.TextBlock2D` -> For storing the name of the element
- `ui.Button2D` -> For acting as a state of branched or unbranched.
- `ui.Panel2D` -> For containing its children.

A quick example of such an arrangement can be as follows:



3. Tree UI Widget:

A Tree UI widget is a graphical control element that presents a hierarchical view of information. Each item (often called a branch or a node) can have a number of subitems. This is often visualized by indentation in a list.
An item can be expanded to reveal subitems, if any exist, and collapsed to hide subitems.
Tree views are often seen in file manager applications, where they allow the user to navigate the file system directories. They are also used to present hierarchical data, such as an XML document. A tree UI component is very similar to that of an Accordian. The only difference being a tree has indentation for displaying its children. Tree UI also has depth deeper than 1 whereas accordions have a depth of 1 only.

The tree UI will be quite interesting from other UI components as each instance will have a nested structure of a different instance of itself.
A tree UI component can be built using the following existing UI components:

- `ui.TextBlock2D` -> For storing the name of the element.
- `ui.Button2D` -> Arrow icon to display the state of that branch.
- `ui.TreeUI` -> TreeUI object to hold subsequent children.

Project Timeline

Period	Milestone
Community Bonding Period **May 4 to June 1**	
Week 1 **May 4 to May 11**	● Get to know the mentors and admin of the project. ● Discuss with the requirements and vision for the mentioned features.
Week 2 **May 12 to May 19**	● Discuss which of the three sub-topics should be worked on first. ● Finalize implementation details for the said sub-topics.
Week 3, 4 **May 20 to June 1**	● Identify and merge any pending pull requests which contribute towards the goals in this proposal, or otherwise essential to FURY. ● Discuss about stretch goals and decide the order of implementation.
Phase 1 **June 1 to June 29**	
Week 1 **June 1 to June 8**	● Start with the overall structural code of the ComboBox UI component by selecting the required inherits. ● Start building up the editable text-field part of the UI
Week 2 **June 9 to June 15**	● Work on Button selection and creation for displaying the drop down list for options.
Week 3 **June 16 to June 23**	● Work on ListBox component UI for displaying the options. ● Integrate all the individual components together.
Week 4 **June 23 to June 29**	● Create Unit Tests for the newly developed UI element. ● Create a tutorial or example for the same. ● Perform bug fixes.
Phase 1 Evaluations ---------- June 29 to July 3	
Phase 2 **June 29 to July 27**	
Week 1 **June 29 to July 6**	● Work on File Dialog improvements ● Add Horizontal Slider for File Dialog ● Add Buttons for functionality

Week 2 **July 7 to July 13**	● Implement an Editable text field for quick access. ● Write unit tests for the File Dialog and perform bug fixes.
Week 3 **July 14 to July 20**	● Start working on the Tab UI component. ● Work on building the panel which will hold individual tabs and close buttons together.
Week 4 **July 20 to July 27**	● Work on “add new tab” button ● Work on the content encapsulating Panel.
Phase 2 Evaluations ---------- **July 27 to July 31**	
Phase 3 **July 27 to August 24**	
Week 1 **July 27 to August 3**	● Create unit Tests for TAB UI ● Add tutorial for Tab UI ● Perform Bug fixes ● If time persists, work on Stretch Goal 1.
Week 2 **Aug 4 to Aug 10**	● Add Support for handling multiple tabs. ● Optimise the added changes for better performance. ● If extra time remaining work on Stretch Goal 2.
Week 3 **Aug 11 to Aug 17**	● Buffer period to complete any remaining tasks or to fix bugs found ● Prepare documentation for the work done during the GSoC period. ● In case extra time is remaining work on Stretch Goal 3.
Week 4 **Aug 17 to Aug 24**	● Thoroughly debug all developed components. ● Prepare demonstration videos for all introduced features. ● Prepare the final demo for all changes made.
Final Evaluations ---------- **August 24 to August 31**	

Commitments and Availability

---

1. Semester Exams during End-May or Early-June for a duration of approximately 10 days.
2. I do not plan on any personal vacations and travel otherwise.
3. My working hours would most likely be(EST) from **2:30 to 4:30** and **8:00 to 12:00** for both weekdays and weekends.
4. However, I am willing to adjust and re-plan the timings as per mentor/admin availability and any unknown requirements if any in future.
5. I will only be applying to FURY for GSoC 2020.