Processing Programming Project

Maria Ceanuri

July 2025

1 Introduction

The purpose of this project is creating a GUI library in Processing designed to support multiple screens and reusable components (widgets). This library will allow developers to:

- Create multiple "screens" and being able to switch between them
- Add and manage both static and interactive widgets
- Access widgets by reference to retrieve or modify data

2 Features

1. **Static Widgets:**
   - Label: Displays static text.
   - Image: Displays images (logos, icons, etc.).
2. **Dynamic Widgets:**
   - Button: Executes an action on click.
   - Checkbox: Stores a checked or unchecked state.
   - Checkbox List: A group of labeled checkboxes.
   - Text Input: Stores user entered text, reacts to keyboard input.
   - Checkbox with image: Stores a checked or unchecked state with an image.
   - Checkbox with image list: A group of labeled checkboxes with images.
3. **Screens:**
   - Independent sets of widgets per screen.
   - Ability to change: screen title, background color, visibility and state of widgets.

4. **Widget referencing:**
    - Each widget is assigned a unique name.
    - Widgets can be retrieved using a string key.
    - Developers can modify the characteristics of the widget (color, text) or read its state (see if checkbox is checked).

3 Architecture

Classes	**Purpose**
GUIManager	Manages the current screen and input forwarding
Screen	Holds a map of widgets and their attributes
Widget (abstract)	Base class for all elements
Label, Image	Static widgets
Button, Checkbox, CheckboxList, TextInput	
CheckboxWithImage, CheckBoxWithImageList	Dynamic widgets

4 Component Details

1. **GUIManager:**
    - addScreen(String name, Screen s).
    - setCurrentScreen(String name).
    - getCurrentScreen().
    - display() — calls current screen's handleInput().
    - handleInput()
2. **Screen:**
    - addWidget(String name, Widget w)
    - getWidget(String name)
    - setBackgroundColor(int bgColor)
    - setTitle(String title)
    - display()
    - handleInput()
3. **Widget (base):**
    - x, y, w, h
    - visible

- enabled
- text, textColor, bgColor
- display(), handleInput()

Subclasses:

Class	Notes
Label	Static text
Image	Draws image from file
Button	Calls onClick()
CheckBox	toogles checked/unchecked
CheckboxList	Array of checkboxes with labels
TextInput	Stores user-typed string
CheckboxWithImage	toogles checked/unchecked with image
CheckBoxWithImageList	Array of checkboxes with images with labels

5 Widget Standard Features

All widgets will support:

- setText(String text)
- setTextColor(color c)
- setBackgroundColor(color c)
- setPosition(float x, float y)
- setSize(float w, float h)
- setVisible(boolean v)
- setEnabled(boolean e)
- mouseInWidget (int px, int py)

6 Event Handling Model

Central event system will be managed by GUIManager.
Processing's mousePressed(), mouseReleased(), and keyPressed() will be forwarded to GUIManager, which will then pass input to the current screen, which passes it to widgets.

For example:

```
void mousePressed() {
  guiManager.handleMousePressed();
}
```

7 Data Access via References

Screen will maintain a `HashMap<String, Widget>`.

This way widgets will be added in the following manner:

```
screen.addWidget("submitButton", new Button(...));
```

And their attributes will be retrieved as follows:

```
Button b = (Button) screen.getWidget("submitButton");
b.setText("Submit");
```

8 Screen Switching

Screen switching will be controlled by GUIManager in the way described bellow:

```
guiManager.setCurrentScreen("Settings");
```

The Title and background color will be automatically updated when the new screen becomes active and the old screen's widgets will be hidden / not drawn.

9 Folder Structure

```
Processing-GUI/
|__README.md
|__src/
|  |__GUIManager.pde
|  |__Screen.pde
|  |__Widget.pde
|  |__Button.pde
|  |__Checkbox.pde
|  |__CheckboxList.pde
|  |__Label.pde
|  |__TextInput.pde
|  |__ImageWidget.pde
|  |__CheckBoxWithImage.pde
|  |__CheckBoxWithImageList.pde
|__examples/
|  |__demo_app/
|     |__demo_app.pde
|     |__assets/
|__docs/
   |__DESIGN_DOCUMENT.md
```

10 Project Timeline

Development Timeline

This section outlines the structured 9-day timeline for the development of this library.

- **Day 1**
  Create the documentation for the project, outlining design choices, architecture and schedule and set up my public repository (Processing-GUI).
- **Day 2**
  Created first UML class diagrams for the project and defined all the main classes (GUIManager, Screen, Widget, etc.). Corrected schedule and specified features to code each day. Start a first processing environment with a basic sketch.
- **Day 3**
  Implement main classes: GUIManager, Screen, and Widget. Try to add a button as a Widget subclass and onClick interface. Check functionality by creating two screens and a button that lets you switch between them when clicked.
- **Day 4**
  Add all widget subclasses: Label, Image, Button, CheckBox, CheckboxList and TextInput.
  Check that all widgets work as expected with different examples.
- **Day 5**
  Started working on the demo. Updated button class to have radio buttons. Updated check box and check box list class to allow for multiple selection. Finally created a colors enum, modified label class to have background and border as optional and added get text function in textinput class.
- **Day 6**
  Continued working on the demo and finished the first version. Also created a new class that creates check boxes with and image and another class that makes lists of the check box image class.
- **Day 7**
  Finalized the demo by correcting some issues, like button logic or text input. Also added the images used in the demo to the repository.
- **Day 8**
  Showcased the demo, start working on reflection and correct documenation according to final design choices.
- **Day 9**
  Finish reflection and update final UML diagram.