

Additional instructions on building the FLTK GUI with FLUID:

For lab 2 you will add additional buttons:

- Radio buttons: for RGBHSV channel selection
- Toggle buttons: for log and interactive selection

1. Radio Buttons

You can specify radio button in Fluid

Menu->new->buttons->round button

When a dialog appears, select label “C++”, then choose “radio” in the list after “class”, There are 3 choices. “Normal”, “Toggole” and “Radio”. Setting one radio button on will turn off other radio buttons in the same group. By default, all items inside a window belong to one and the same group. However, you may divide items into different groups. Here we choose “Radio” rather than “Toggle” since we want only one active state.

Fluid will generate the corresponding gui.h and gui.cpp code. The radio button will appear in the following source code inside gui.cpp,

```
object->type(102);// or object->type(FL_RADIO_BUTTON), since FL_RADIO_BUTTON is defined as 102
```

If you wish to set one radio button as default, in Fluid, open the button properties dialog by double clicking the button. In dialog box->“GUI” label->values->fill value by 1.

Fluid will generate the corresponding gui.h and gui.cpp code. The radio button will appear in the following source code inside gui.cpp,

```
object ->value(1);// set default value, if it is 1 then it is select when it is first time displayed
```

To set the Radio button name, when a dialog appear->“GUI”, enter “R” (for Red) in the first blank.

To set the Radio button callback name, select label “C++”->Name, fill in “R”, for example. Then put the following code into “callback”:

```
app->SetActiveTransFunction(0);
```

```
transFuncEditor->redraw();
```

Then if you set the R button, the application will call a member function (SetActiveTransFunction(0)) inside your app class (you need to write that routine and define what to do it inside that function), then redraw your transfunction window (you could also add that statement directly into your function code instead).

Now do this for all buttons and pass a different value for each via SetActiveTransFunction(), for example, for G use SetActiveTransFunction(1). To make one general callback to handle RGB & HS, our application code would be:

```
void Application::SetActiveTransFunction(int num)
```

```
{
```

```
    activeTransFunc =num;
```

```
}
```

2. Toggle Buttons

Similar to radio button.

Menu->new->buttons->round button

When a dialog appears, select label “C++”, then choose “toggle” in the list after “class”. If it is set on, it will not turn off other buttons.

If you want to set one toggle button as default, in Fluid, open the button properties dialog by double click the button. In dialog box->“GUI” label->values->fill value by 1.

Fluid will generate the corresponding gui.h and gui.cpp code. The radio button will appear in the following source code inside gui.cpp,

```
object ->value(1);// set default value, if it is 1 then it is select when it is first time displayed
```

To set the Toggle button name, when a dialog appears->“GUI”, enter “Log” in the first blank.

To set Toggle button callback name,select label “C++”->Name, fill “Log” for example. Then you put the following code into “callback”

```
app->TriggerLog();
```

Then if you set the Log button, the application will call a function `TriggerLog()` you will need to write inside your app class(you define what to do it inside that function).