Start by opening the MATLAB

Then open the file >> Motor\_spindown.mlapp

(If you want a nice quick tutorial before we begin, there is an interactive one if you click [New] at the top left and select “interactive tutorial”, you can reopen this after)

Click run in the new window that pops up to generate the interface

Chart

Description automatically generated

Change the variable to those listed above and click [Start]

Play around with is for a minute by clicking the various buttons and switches to change the setting. Observe

Next, close that window and return to the App Designer window

Chart, table

Description automatically generated

The interface on this screen is vert intuitive. You can click, drag and resize anything on this screen and even change the color and font of just about anything with the right-hand column. Another thing you can do is disable components and/or even make them invisible.

Click the text box under “Torque from Coulomb Friction”

Click on the tab > Interactivity

Uncheck Enable

Click on the [Start] button and uncheck visible in the same manor

Now click [Run] at the top

As you can see, the [Start] button is missing and you cannot interact with it or the text box at all (an aside, labels tide to components that are disabled are go opaque, as seen above the text box)

Now that we have thoroughly \*broken the GUI, let’s look at the code and see if we can fix it.

Click the box that says [Code View] that lays above the GUI face

Chart, table

Description automatically generated

The code itself can be a little daunting at first but realize that what you did by interacting with the design view is what wrote it. Also, the changes you made over there are not even editable (greyed out), so you don’t hav3 to worry about deleting anything or messing anything up.

Look at line 262 { app.StartButton.Visible = 'off'; }

Now, go back and re-visualize the [Start] button

Line 262, or rather {app.StartButton.Visible = 'off';} has disappeared because it is a default setting

Any time you change a component design from a default setting it auto-changes in the code under the appropriately labeled section at the bottom. A good way I found to code something you don’t know how to is to change it in the [Design View] and then look at the line of code added into the [Code View] especially if you want to resize or move components.

Now let’s edit some code.

I want you to copy line 219 and paste it into the StartButtonPushed callback function (line 109 is good, lines after the for loop would make you wait with the way this code is set up, so line 150 would make you wait)

Change ‘off’ to ‘on’

Now, the first time you click the start button, it will enable the text box when we run the GUI

Test it now

Chart

Description automatically generated >> Chart

Description automatically generated

Notice the text box is still opaque because even though both are affected by the design, that are two different components in the end and two lines of code are generated: Line 212 and 219

*This is getting long for an introduction, so I’m going to give you a few bullet points and let you start hacking away at it:*

**Global variables** are referred to as app.variableName in the code and are initialized in the first editable section. This code is not the best example, so I added a blurb from a different one:

Table

Description automatically generated with low confidence

Add new global variables here or in the lefthand tab ‘Properties’

Graphical user interface, text, application, email

Description automatically generated

You don’t have to do this for variable within a function that do not need to be shared with other functions

**Callbacks** are function associated with components and events. The best example of this is a button press.

Open a new blank GUI and drag and drop a button and a numeric text box

Graphical user interface, application, Word

Description automatically generated

Now go to [Code View] and right click app.Button >>Callbacks >>Add ButtonPushedFcn callback:

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

This will pop up so you can write your function to do something whenever ‘event’ (button pressed) happens

Startup and other types of event functions can also be made via the left hand column by clicking the plus sign

Graphical user interface, text

Description automatically generated

**Regular functions** can also be added via the ‘Functions’ tab that are not associated with events

***Quazi-onyourown-lesson***: In the Motor\_spindown.mlapp GUI you might have noticed that any time you clicked the button it restarts the graphic. Buttons on this GUI are interrupts, so they break the code wherever it may be. Your first mini project to add a button to a new GUI that tests your reaction time. This should include a text window and a button. Maybe an LED if you are feeling bold. Try and work on it yourself, and then reach out to me and I can send you one that I build.

Code you might need:

~ app.EditField.Value = 30; % changes the value displayed in the edit field text box

~ tic; toc % these to are a pair that count the milliseconds between tic (initialized time) and

% toc (current time). toc does not re-initialize the pair

~ app.Lamp.Color = [1 0 0]; % changes the lamp color

~ app.Lamp.Position = [705 622 51 51]; % changes the lamp position