

Showing a GUI with ScriptEase II

GUI stands for Graphical User Interface. GUIs are used in games to display hit points, show menus when paused, and more. Basically anything on the screen that is not part of the game world, but used to connect the player with it is, is part of the GUI. In this tutorial, we will use ScriptEase II to create a GUI for our Duck story in the Park scene.

We will also use some advanced story techniques to finish the story at the appropriate time.

Getting Started:

1. Back up your *.ses file as usual. It's also recommended to back up your Unity project from time to time, in case something goes wrong either in Unity or with ScriptEase II.

Creating the GUI:

2. Open your story in ScriptEase II. Select the Start story point to edit its contents.
3. Drag in a "When the GUI is updated" cause. Like the "While subject exists" cause, this cause continuously occurs. All GUI related effects need to be in this cause as this is the only cause with a GUI object attached to it. We place this cause in Start because any cause that isn't story dependant should be in the Start cause to make it easier to find them if necessary.
4. Delete the "Is Active" related description and question from the cause.
5. First we need a background for our text label. Find the "Draw a box on GUI..." effect that does not have a slot for text. Drag this into the cause.
6. We also need to draw a label using the "Draw a label on GUI with text ..." effect. Add this to the cause. Note the order of these two effects. We need to draw the box first so that the label is drawn on top of it. Once you have completed this tutorial, you can try reversing the order to see its consequences.
7. The effects first ask for a GUI object. Drag the GUI object from the Cause into their slots.
8. Let's ignore the text slot in "Draw a label" for now. The other slots ask for the x and y position on the screen of the GUI components, and their length and width.
9. We want the components to show up in the top right corner. We can subtract from the screen width to find the exact number. Since the GUI cause happens constantly, this will make sure they always shows up in the top left corner even if the screen width changes. We also don't have to worry about the type of monitor the user is playing the game on. Drag in the "Screen Width describes the width of the screen" description so

that it is above the “Draw a box...” and “Draw a label...” effects.

10. If we set the X position to the screen width, the components would appear outside of the screen. We need to subtract a number from the screen width: the width of the components. We should create another description to describe the component width, as we will be using it in two places. Add a “Number describes the number #” description, change its name to “Label Width”, and set it to 150 as in Figure 1.

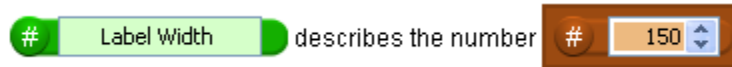


Figure 1: This description helps us since we'll use the width in two places.

11. We also want to define the height. Create another description called “Label Height describes the number 50” using the same description as the previous one. This gives enough space for the text to show up.
12. Add a description called “Difference describes # minus #” below our other descriptions but above the two effects. Turn it into “X position describes Screen Width minus Label Width” by changing its name and dragging in the description objects created above it.
13. Drag the X Position into the first number slot in the two effects. See Figure 2 if you are unsure where it goes. Note that Figure 2 is the state of the Cause when we have finished creating it.
14. We leave the Y position at 0, since the components show up at the top of the screen.
15. Drag the Label Width object into the “# wide” slots and the Label Height into “# long” for both the label and box effects.

Remembering Values:

16. We now have everything in our Effect set up except for the text itself. We could fill in the text, but then we could not change it in a later cause. Instead, we can use ScriptEase II's system of remembering values as a label.
17. ScriptEase II can remember numbers, text, questions, and even GameObjects as a label. They are then recalled by a description. We will use our current work as an example. Find the “Text describes remembered Text” description and drag it into the Cause above the “Draw a box” effect.
18. In the text field, write “Label”. The “Text” is now whatever value was assigned to “Label”. Since nothing was remembered by it, it will be blank.

19. Drag the created “Text” object into the text slot in the label effect. The cause should look like Figure 2.

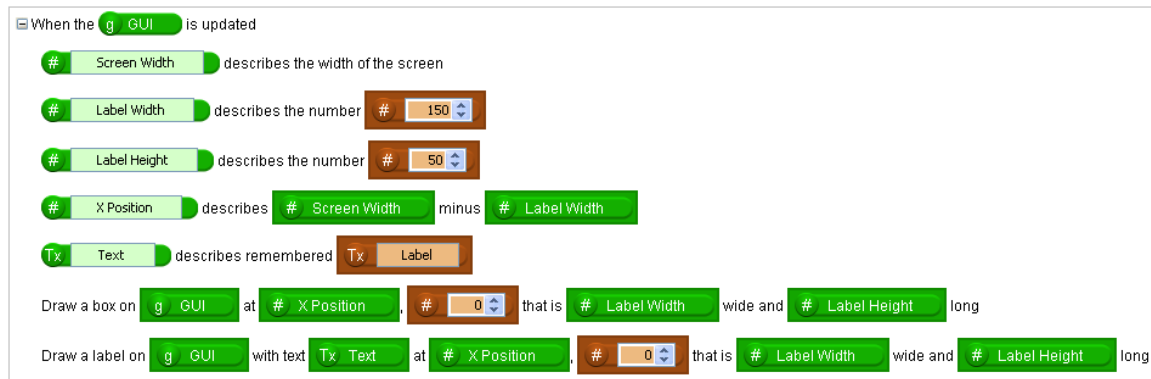


Figure 2: This will draw the text remembered as “Label”.

20. Think of GUI effects as showing their GUI component, such as a label, for about a millisecond. The “When the GUI is updated” cause constantly happens, so it will look like the GUI component is always there. But since this effect is always occurring, we can change the value that “Label” remembers to show different text in the GUI label!
21. Switch to the “Talk to Mother Duck” story point.
22. Find the “Remember [text] as [text]” effect and drag it into the Yes part of the Is Active question. Write “Find the ducks!” in the first slot and “Label” in the second as in Figure 3.

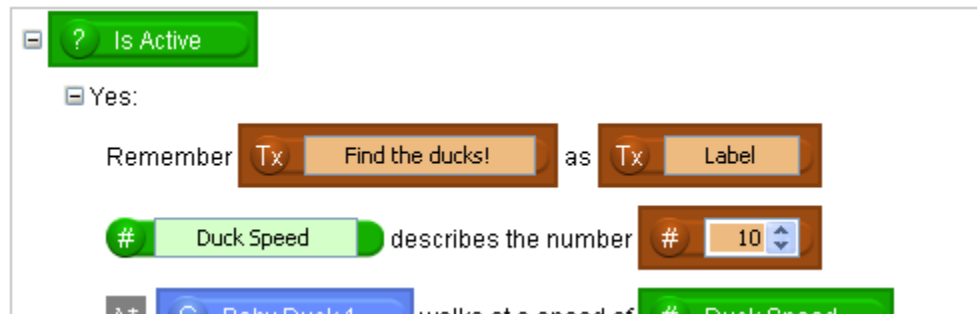


Figure 3: Changing the text that “Label” remembers.


23. Save and test your story. Notice that the label doesn’t show up until you click on the Mother Duck, when the text remembered by “Label” changes.
24. Add the “Remember [text] as [text]” effect to the four Get Duckling story points. Change the text remembered by “Label” in all of these to “We found a duckling! But there are still more.” as in Figure 4.

Tip: Alternatively, we could remember the number of ducks found, and then show it on the label using descriptions to change the number into text, and combine it with other text.



Figure 4: The text we remember when we find a duckling.

Ending the Story with Fan In:

25. The ducklings return, but the label never changes to notify the player that they have finished the quest. Add another Story Point to the “Get Duckling 1” story point. Name it “Ducklings Found.”
26. Switch to the  connection tool. Drag from “Get Duckling 2” to the new “Ducklings Found” point to connect them.
27. Repeat this for the other “Get Duckling” story points. Your graph should look like Figure 5. The vertical order of the four Get Duckling story points does not matter.

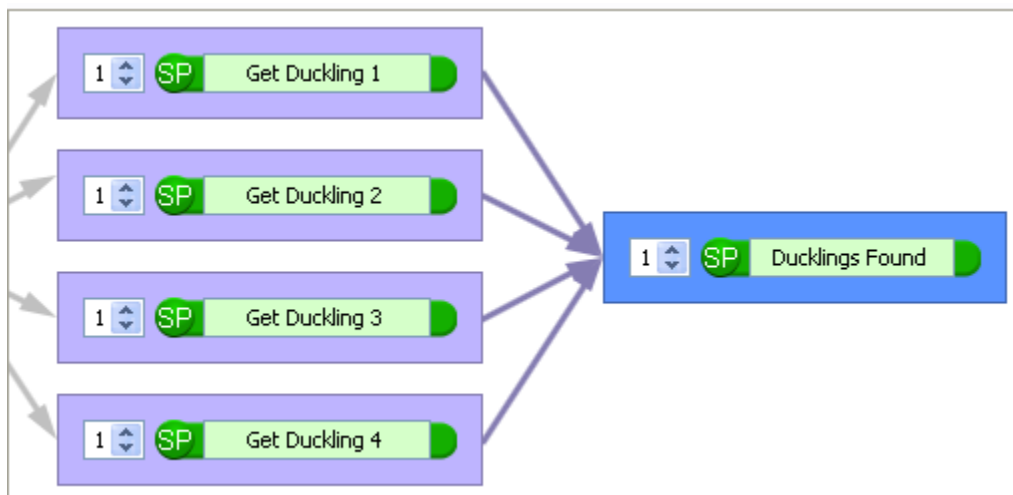


Figure 5: The graph after adding the end point.

28. Change the value of the number spinner in the “Ducklings Found” story point to 4. This means that 4 preceding story points must be succeeded before “Ducklings Found” is

activated. Since there are only 4 preceding story points, “Ducklings Found” will only be activated once all of them are succeeded; in other words, when all of the ducklings are found.

29. Select the “Ducklings Found” story point and drag in a “When subject is activated” cause. This cause takes a Story Point as its subject. Drag the “Ducklings Found” story point object into the cause.
30. We do not need the “Is Active” description and question, since they would be redundant. To make things simpler, you can delete these.
31. Drag in another “Remember [Text] as [Text]” effect and use it to change Label’s text to “The ducklings were found!”
32. We can also succeed the “Ducklings Found” story point in this cause. This will not have any effect at this point in our story, but we may need to know whether the ducklings were found in another story point. This prevents us from making a mistake in the future.
33. Your cause should look like Figure 6. Save and test your story.



Figure 6: The cause inside the “Ducklings Found” story point.

You have now finished the third tutorial. In this tutorial, you learned how to use GUI effects in Unity. You also learned how to use the remember system to remember text, numbers, and other values. The story was then completed using Fan In.

You now have the knowledge to create a basic branching story with ScriptEase II. Branches can be used to create side quests, multiple paths to the same end, or other creative story structures. We will cover another method of controlling the story in the next tutorial.