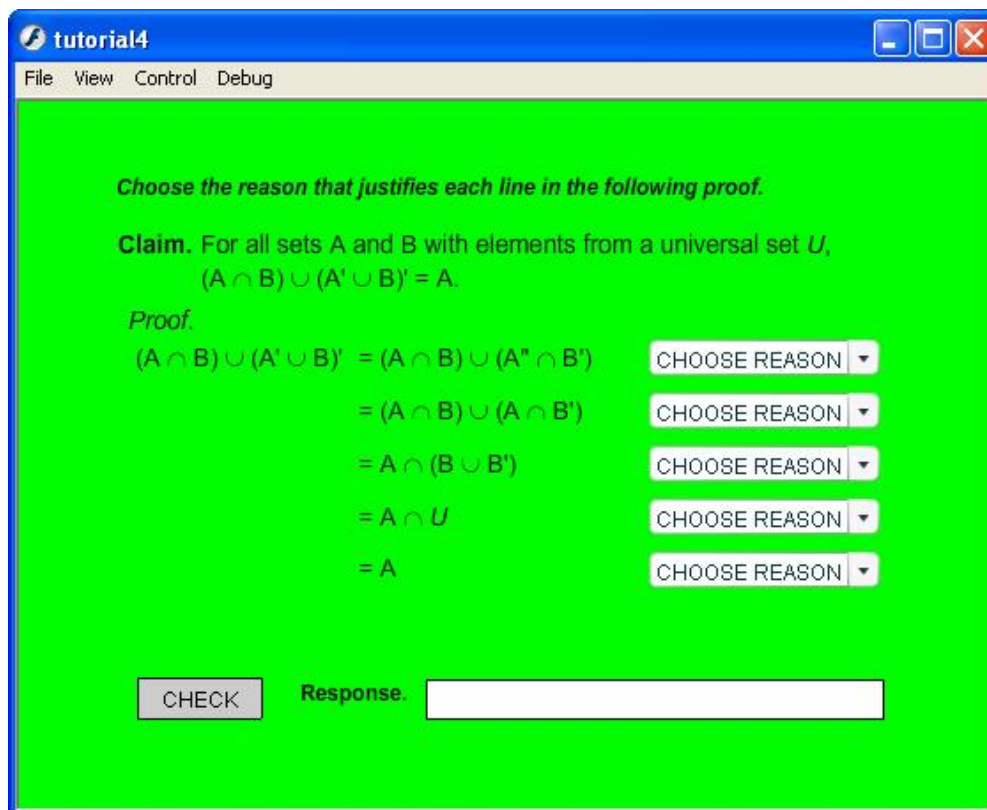


Flash basics for teaching mathematics

Tutorial 4. Using the combo box component

by Doug Ensley, Shippensburg University and
Barbara Kaskosz, University of Rhode Island

In this tutorial, we will make the Flash movie shown below. This applet shows a proof of a property of set operations, and the user must provide the explanation for the truth of each step. The program provides “right or wrong” feedback when the CHECK button is pressed.



Step 1. Add all textboxes to the stage

Start with a new project in the *Flash* authoring environment. You will need the **Tools** panel, the **Properties** panel and the **Stage** to be visible. All other panels can be closed if you wish. Just as you did in Tutorial 2, click on an empty section of the stage, and in the **Properties** panel choose a nice background color to contrast appropriately with your text. As before, save your file¹ in the tutorials folder you have created.

We will now add static textboxes to hold the instructions, the claim, and the lines of the proof. If you are unsure how to do this, see Tutorial 2, “Reading input text boxes and writing to dynamic text boxes.” If getting symbol font characters (like \cap and \cup) is new to you, we will explain one way to do this on a Microsoft Windows system:

For our example, we will make a static textbox with the string “ $A \cap B$ ”. Click on the MS Windows Start menu, and choose

¹ It is a good idea to create a separate folder for all of these tutorial exercises so they are easy to find later. Giving your project a meaningful name like SetProof will also be helpful. At the end of Tutorial 1, we talked about the files created by Flash and how this is relevant to posting your movies on a website.

All Programs > Accessories > System Tools > Character Map.

Select the Symbol font and scroll down until you have found the \cap character. Click on it and press the Select button followed by the Copy button. Return to Flash and create a static text box for a set expression. Type “A,” a space, and choose Paste from the Edit menu (or hit Ctrl+v). Then type another space followed by “B.” At this point, your text probably looks like this:

A B

Highlight the middle character and select “Symbol” font for this character only. The text will now look like $A \cap B$.

Finally, create a dynamic textbox with instance name **txtResponse** located at the bottom of the screen next to the (static textbox) label RESPONSE. You might put a simple phrase in **txtResponse** so that you can make sure that its font size and style fit comfortably within the dimensions of the box. We can simply clear the box with the first line of the script, so you can use whatever you want for an initial placeholder.

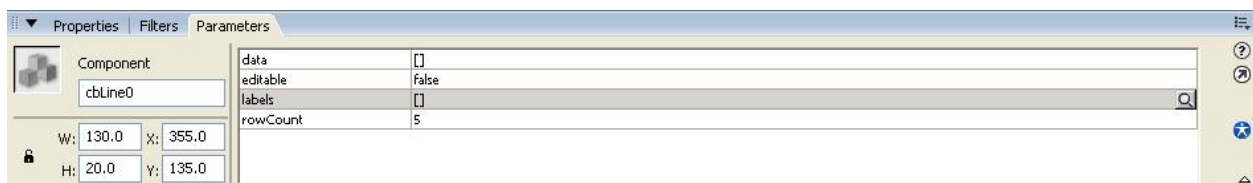
Step 2. Create the combo boxes

Each line of your proof should have a combo box (i.e., a pull-down menu) that gives the list of possible reasons for that step. We are choosing our reasons from a fixed list of “primitive” properties of set operations (like commutativity, distributivity, etc.), so the combo boxes for the lines in our proof will be identical. This means that we only have to set up one and then we can “copy and paste” to make the others.

As in Tutorial 3, choose Components from the Windows menu, and expand the list called “User Interface” if it not already expanded. Scroll down the list of **User Interface Components** until you come to the **ComboBox** item. Click on this item and drag it onto your stage in approximately the correct position. (We will line everything up exactly later.)

Now open the **Properties** panel and select the **Parameters** tab. You should first specify an instance name (I used **cbLine0**) and “nice” X- and Y- position coordinates. I used X = 355 and Y = 135, but your placement will depend upon the rest of your layout. You should also set the height and width of the combo box to be 20 and 130, respectively. The default values are not wide enough for the text that we are going to use.

If you click on the field next to the word “labels,” your **Properties** panel will look like this:



Click on the magnifying glass on the far right side of the chosen field, and a new dialog box (titled Values) will pop up to accept your input. Here we will enter the text strings that we want to appear as choices within the combo box. In our example, we list eight items in our combo box, so you should use the “+” button to create eight items (as shown on the right) and then type the following:

- | | |
|-------------------|--------------------|
| 0. CHOOSE REASON | 4. Identity |
| 1. Commutativity | 5. Negation |
| 2. Associativity | 6. Double Negation |
| 3. Distributivity | 7. DeMorgan’s Laws |



This process should be repeated with the **data** field, which contains the “value” of the combo box when a particular one of the eight items is selected. We will simply use the values 0, 1, 2, 3, 4, 5, 6, 7 for our data, so we will have to remember which number goes with which property when it is time to write our script.

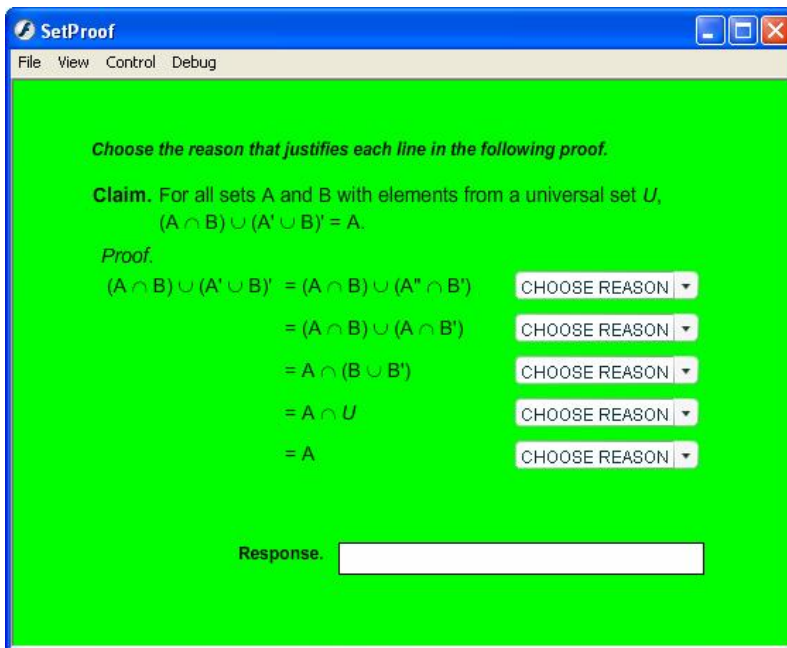
Finally, change the **rowCount** field to be 8 instead of 5 so that all eight choices are shown when the user clicks on the combo box. If you use the value 5, then only five choices at a time can be seen and a scroll bar will be used. Save and test your movie to see that your combo box actually works.

Now click once on your combo box to select it, and choose **C**opy from the **E**dit menu and the **P**aste from the **E**dit menu. In the **P**roperties panel, give this new combo box a different instance name (**cbLine1**) and appropriate x- and y-coordinates to have it line up with **cbLine0**. Repeat this process until you have created a combo box for each line of your proof, each with a unique instance name.

Save and test your movie to see that it looks like the screen shot shown at the right.

Step 3. Add a button

We now need a button that when pressed checks the answers chosen by the user. Even though we are experts at making buttons from scratch, for this application we will take a shortcut that you might find useful in the future.



With your **SetProof.fla** file remaining open, choose **O**pen from the **F**ile menu to open another Flash .fla file that includes a button that you like. (I will use tutorial2.fla since it has a button already labeled “CHECK.”) Click on this button on the **S**tage once to select it, and then choose **C**opy from the **E**dit menu. Close this .fla file without saving it, and return to your **SetProof.fla** file and click once on the **S**tage. Select **P**aste from the **E**dit menu, and a copy of your nice button will now be in this application. Be sure to give your button an appropriate instance name like **buttonCheck**.

Using this method, once you develop a personal style for buttons, you do not have to create them from scratch for each new applet.

Step 4. Add the script

The script for this movie is very simple. First click on Layer 1 of the timeline, and choose **I**nsert > **T**imeline > **L**ayer. Double click the layer labels to give the new layer the name “Scripts” and the old layer a descriptive name like “On Stage.” Click Frame 1 of the Scripts layer, and open the **A**ctions panel in preparation of writing some code. Here is the full script for this movie. Once again, you do not need to type comments (following the //). If you do not wish to type *any* of this, you can open the file **F**ullScript.txt, and copy and paste the script into your movie at this time.

```
// Clear the Response box when the program is started.
txtResponse.text = "";
```

```
// Set the correct answer to the problem by giving the reasons for the steps
```

```

// of the proof (in order) using the following numbers:
// 1 = Commutativity           2 = Associativity           3 = Distributivity
// 4 = Identity               5 = Negation              6 = Double Negation
// 7 = DeMorgan's Laws

var correctAnswer:Array = [7, 6, 3, 5, 4];    // Reflects that line 1 is true by
                                              // DeMorgan's Laws (7), line 2 is
                                              // true by Double Negation (6), etc.

butCheck.onRelease = function():Void {
    // Initially we say that the answer is correct until proven otherwise.
    var isCorrect:Boolean = true;

    if (cbLine0.getValue() != correctAnswer[0]) { // If first line has wrong
        isCorrect = false;                       // reason, then overall
    }                                             // answer is wrong.

    if (cbLine1.getValue() != correctAnswer[1]) { // Same idea repeated for
        isCorrect = false;                       // each of the five lines.
    }

    if (cbLine2.getValue() != correctAnswer[2]) {
    }

    if (cbLine3.getValue() != correctAnswer[3]) {
    }

    if (cbLine4.getValue() != correctAnswer[4]) {
    }

    // If we have not found a mistake, then the overall answer is correct.
    // Otherwise, we will say the answer is wrong.
    if (isCorrect) {
        txtResponse.text = "Correct proof! Good work!!";
    }
    else {
        txtResponse.text = "At least one step is incorrect.";
    }
}

```

Save and test your movie, and you will see a full working version of the applet we described at the beginning of this tutorial.

Additional resources

To see the source code for the applet we made for this tutorial, open the file **tutorial4 fla** in Flash.