



ADDING DIFFERENT RESPONSES TO DRAG AND DROP GAMES IN FLASH CS3

This tutorial is designed as a follow-up resource for students who have worked through the **Make a Drag and Drop Game with Targets in CS3** tutorial.

In this tutorial, you will learn how, by adding small changes to this game by either adding items or script, we can change the nature of the game and provide the player with different types of feedback. The different parts of this tutorial cover:

- providing feedback through a dynamic text box
- providing feedback with a sound
- providing feedback with a movieclip
- providing a final response when all objects are in the correct place
- removing the snap to target feature of the game

It is recommended that you work through this tutorial in order. Whilst you may only want to use all of the responses above in games you create, the sections build on each other in complexity so you will learn more if you work through them in order.

Getting started

For this tutorial, we are going to continue to build on our `shapematch.fla` file we worked on in the **Make a Drag and Drop Game with Targets in CS3** tutorial. A copy of this file is available for download from the Learning Object Design page of the Learn Flash section of the Flash Classroom.

1. Open up your copy of **shapematch.fla** or the copy you have downloaded from the Flash Classroom site.
2. Review the important features of this file:
 - the shapes layer that contains shapes you converted to movieclip symbols with instance names e.g. `triangle_mc`
 - the targets layer that contains the silhouette targets of the shapes. These were converted to movieclip symbols and given instance names e.g. `targettriangle_mc`.
 - the actions layer that contains one keyframe with a small `a` in it. This frame contains the script that makes the game work. Click on this frame and press F9 to see the script we entered to create the interactivity for the game.
3. Test the game again by selecting **Control > Test Movie**. You should be able to drag each shape to the correct location and have the shape lock into place. Alternatively, if you drag it to the wrong target, it will snap back into its original place. If your game isn't working - download my version to use or work through the **Drag and Drop with Targets** tutorial again.



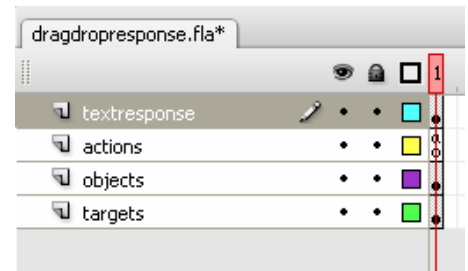


Providing Feedback with a Dynamic Text Box

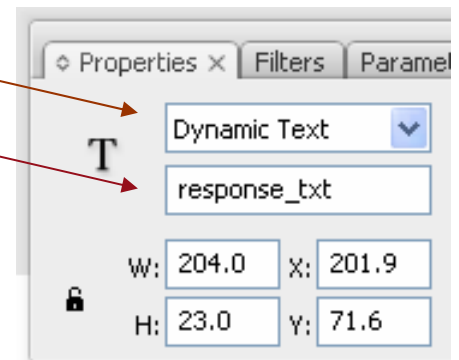
In this section, we are going to explore how we can set up a text box that will give a different message when the user gets a shape in the correct or incorrect position. To do this, we will set up a dynamic text box. In this context, think of dynamic as meaning having the ability to change. In Flash, dynamic text boxes store the value of variables. The value of variables can be determined by the script the developer of the Flash file has entered or a value that the user has entered into an input text box. In our case, the value will be determined by the script we enter.

1. Add a new layer to your game by clicking the **Insert Layer** button. Name this layer **textresponse**.

2. Click on the first keyframe in the textresponse layer and **draw a text box** on the stage using the Text tool.



3. In the **Properties** panel at the bottom of the screen, change the text type to **Dynamic Text** and give your textbox the instance name **response_txt**.



4. Choose the **font** and **font size** you want for the text that will display in the box. Make sure the box is long enough to fit in three - four words and position it in the place you want it to appear on the stage. Note that the game will work best if the text box isn't in the area the objects will be dragged across.

We are now going to add three simple lines of actionscript to our existing script to make the response text box work when the game is played. The first line of script is within the first function.

5. Select the first keyframe on the **actions layer** and **press F9** to open your **actions** panel.
6. Within the first function shown below, we are going to add one simple line of script. This script tells Flash to make the text box empty when the user starts dragging one of the shapes. Enter this line of text in the line after the `objectoriginalY` line.

```
function pickupObject(event:MouseEvent):void {
    event.target.startDrag(true);
    event.target.parent.addChild(event.target);
    objectoriginalX = event.target.x;
    objectoriginalY = event.target.y;
    response_txt.text = " ";
}
```

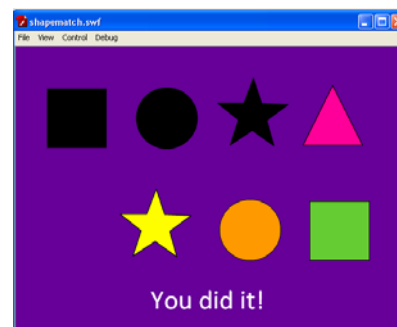




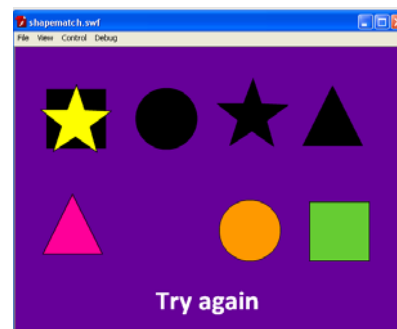
FLASH CLASSROOM - ADDING DIFFERENT RESPONSES TO DRAG AND DROP GAMES IN CS3

7. We are now going to add the final two lines of script to the second function in your actionscript. This function is shown below. I have used a larger size font to highlight the two lines you need to add. Note that you can change the text that will appear (e.g. You did it! / Try again) to anything you like. Ensure your text box is big enough to fit your text though.

```
function dropObject(event:MouseEvent):void {
    event.target.stopDrag();
    var matchingTargetName:String = "target" + event.target.name;
    var matchingTarget:DisplayObject = getChildByName(matchingTargetName);
    if (event.target.dropTarget != null && event.target.dropTarget.parent == matchingTarget){
        event.target.removeEventListener(MouseEvent.MOUSE_DOWN, pickupObject);
        event.target.removeEventListener(MouseEvent.MOUSE_UP, dropObject);
        event.target.buttonMode = false;
        event.target.x = matchingTarget.x;
        event.target.y = matchingTarget.y;
        response_txt.text = "You did it!";
    } else {
        event.target.x = objectoriginalX;
        event.target.y = objectoriginalY;
        response_txt.text = "Try again";
    }
}
```



8. Your game should now be ready to test. Select **Control > Test Movie** and try it out. If the text doesn't display when your object is placed in the correct or incorrect position, double check the script and the instance name you used to ensure they match.



9. **Save your game as shapematchresponse fla.**

You may also want to publish your game so that it can be played on any computer. Select File > Publish settings and choose your file formats to do this.

It is a good idea to keep a copy of this file so that you can refer to it and copy the script when you want to make a similar game in the future.





Providing Feedback with Sound

In this section, we will explore how we can get Flash to provide feedback by playing a different sound when the user gets the shape in a correct or incorrect position. We will do this by loading two external sound files in **mp3 format** and adding some additional lines of script into our existing actionscript.

1. Your first task will be to **find two mp3 files** that you wish to use for your game. One sound should be a positive sound to indicate the piece is correct. The other sound should be a negative sound to signify the piece has been placed in the wrong position. If you don't have sound files that you can use, you can find and download some free ones from Flashkit - www.flashkit.com/soundfx/. This is one of the best sites on the web to find free sound loops and effects.
2. Once you have found two mp3 files, **save** these into the **same folder** as your shapematchresponse.fla file. Rename them positive.mp3 and negative.mp3. These will be the names that we refer to in the script we add in the next step.
3. We are now going to add the two blocks of script that we will need to include in order to load each sound into Flash and set it up as a sound object.

This script should be typed at the top of the script you have already into the actions panel.

```
var req:URLRequest = new URLRequest("positive.mp3");  
var positive:Sound = new Sound();  
positive.load(req);
```

```
var req2:URLRequest = new URLRequest("negative.mp3");  
var negative:Sound = new Sound();  
negative.load(req2);
```

4. Our final step to get the sounds to play is to add two more lines of script within the if statement. These two lines are highlighted below.

```
if (event.target.dropTarget != null && event.target.dropTarget.parent ==matchingTarget){  
    event.target.removeEventListener(MouseEvent.MOUSE_DOWN, pickupObject);  
    event.target.removeEventListener(MouseEvent.MOUSE_UP, dropObject);  
    event.target.buttonMode = false;  
    event.target.x = matchingTarget.x;  
    event.target.y = matchingTarget.y;  
    response_txt.text = "You did it!";  
    → positive.play();  
} else {  
    event.target.x = objectoriginalX;  
    event.target.y = objectoriginalY;  
    response_txt.text = "Try again";  
    → negative.play();  
}  
}
```





5. Your game should now be ready to test to see if the sounds work. Select **Control > Test Movie** and try it out. If you can't hear the sounds, make sure the sound is up on your computer. If it is, then check through your script to check that it matches mine.
6. **Save your game** as **shapematchwithsnd.fla**.

You may also want to publish your game so that it can be played on any computer. Select File > Publish settings and choose your file formats to do this.

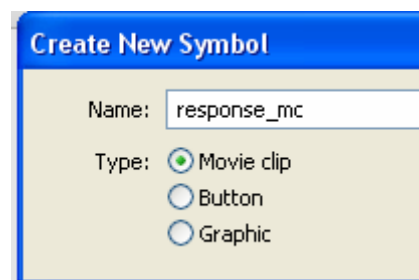
Providing Feedback with a Movie Clip Symbol

There will be times when you want to provide a response that is more interesting. You might want the user to be shown a different image or even an animation if they get the object in the right or wrong location. The easiest way to do this is to set up a movie clip symbol with multiple frames. You can then add script that tells Flash to go to certain frames depending on the response that is required.

This section of the tutorial will show you how to set up a simple movie clip with three frames. The first frame will be empty and will be what the user sees to begin with. The second will contain a smiley face that the user will see if they drag the shape into the correct location. The third will be a sad face that the user will see if they drag the shape into the incorrect location.

1. Select **Insert > New Symbol** to set up a new symbol in your game.

Name your symbol **response_mc** and select the **Movie clip** type. Click OK to begin creating the movie clip. Note that you will see a new timeline. This is the timeline for the movie clip. To go back to the main stage at anytime, you can select the Scene 1 link at the bottom of the timeline.



2. Select the first keyframe on Layer 1 in the movie clip symbol and press **F9** to open up the Actions panel. As we want Flash to stay on this empty frame until the user drags a shape to the correct or incorrect location, we will need to **add a stop action**. Add the following line of script into your panel:

```
stop();
```

3. Click on the **second frame** in the response_mc timeline and press F6 to make it a **keyframe**. In this keyframe, draw what you want the user to see if they drag a shape into the correct location.
4. Now click on the third frame in the response_mc timeline and press F6 to make a third keyframe. In this keyframe, draw what you want the user to see if they drag a shape into an incorrect location.



In my example, I have used smileies (as shown here).





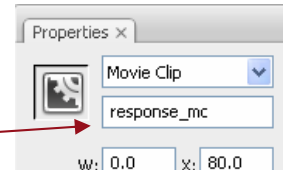
FLASH CLASSROOM - ADDING DIFFERENT RESPONSES TO DRAG AND DROP GAMES IN CS3

5. You have now set up the response_mc movieclip with three keyframes. It's time to go back to the main timeline to add an instance name. To do this, Select the **Scene 1** link underneath the timeline.



6. We are now going to place the response_mc movie clip you have just created onto the stage. To do this, click **Ctrl + L** to open the library. Select the response_mc movie clip from the library and **drag it** onto the stage. Note that you will only see a small white circle at this point as the first frame of our movie clip is empty. When we test the movie, the movie clip will not be visible until a shape has been dropped in a correct or incorrect location.

7. It's time to give our movie clip an instance name so that we can refer to it in our script. To do this, click on the response_mc on the stage and then in the Properties panel, enter the instance name **response_mc**.



8. We're now ready to add a couple of lines of script to make the response work. To do so, click **F9** to open the **Actions** panel.
9. **Add the three highlighted** lines of script shown below to your own script in the same location.

```
function pickupObject(event:MouseEvent):void {
    event.target.startDrag(true);
    event.target.parent.addChild(event.target);
    objectoriginalX = event.target.x;
    objectoriginalY = event.target.y;
    response_txt.text = "";
    response_mc.gotoAndStop(1);
}
```

This line tells Flash to display frame 1 of the response_mc when a shape is picked up by the player.

```
function dropObject(event:MouseEvent):void {
    event.target.stopDrag();
    var matchingTargetName:String = "target" + event.target.name;
    var matchingTarget:DisplayObject = getChildByName(matchingTargetName);
    if (event.target.dropTarget != null && event.target.dropTarget.parent == matchingTarget){
        event.target.removeEventListener(MouseEvent.MOUSE_DOWN, pickupObject);
        event.target.removeEventListener(MouseEvent.MOUSE_UP, dropObject);
        event.target.buttonMode = false;
        event.target.x = matchingTarget.x;
        event.target.y = matchingTarget.y;
        response_txt.text = "You did it!";
        positive.play();
        response_mc.gotoAndStop(2);
    } else {
        event.target.x = objectoriginalX;
        event.target.y = objectoriginalY;
        response_txt.text = "Try again";
        negative.play();
        response_mc.gotoAndStop(3);
    }
}
```

This line tells Flash to move to frame 2 of the response_mc when a player drops a shape in the right place.

This line tells Flash to move to frame 3 of the response_mc when a player drops a shape in the wrong place.





10. If you've followed the steps correctly, your game should now work and you should see the different frames in the response_mc movieclip you made when you drag the shapes into the correct and incorrect positions. If not, work through these steps again to see where you went wrong.

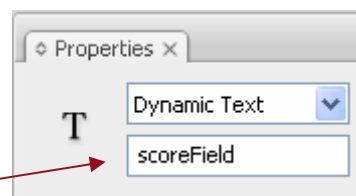
Test your movie by selecting **Control > Test Movie** and then save your file as shapematchmcresponse fla.

Providing a final response when all shapes are in the correct place

One of the things you'll probably want to do is provide some sort of feedback or response to the player when they have placed all the pieces in the correct place. The easiest way to do this is to add a variable that will track the number of pieces that have been put in the correct place. You can then set up some script that will provide a response once the variable reaches the total number of pieces.

Let's have a look at how this would work in the context of a game where we have are going to provide the user with a text based response if all pieces are put in place. Note that you could use any of the other response types we have covered when you make your own game.

1. To begin with, we'll set up a dynamic text box on the stage that can display the number of matched pieces. Select the **Text Tool** and draw a text box on the stage. In the properties menu, select **Dynamic Text** from the drop down menu and enter **scoreField** in the instance name cell.



2. We are now going to add the script that will set up a variable to track the number of matches and display it in the text box we just set up and named scoreField. To do this, press **F9** to open the **Actions Panel**.

To begin with we'll declare the variable score and give it an initial value of 0. To do this, add the following line of script in the first line of your panel.

```
var score:Number = 0;
```

3. We'll now add the following script in the two lines **before the } else { line** in our dropObject function. Note that these two lines need to be in this order. If you put the score++; after the other line it won't work.

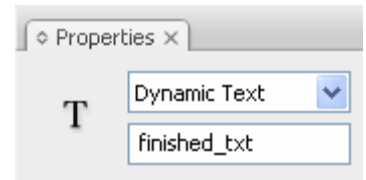
```
score++;  
scoreField.text = String(score);
```

4. Test your game by selecting **Control > Test Movie**. When you match the pieces you should see the dynamic text box called scoreField display the value of the variable you set up called score. If not, double check the script you have entered.





5. It's now time to add another dynamic text box to our stage. This will be the text box that will display a message to the player to tell them they have matched all the pieces. Draw a **text box** on the stage using the Text Tool. Select the **Dynamic Text** option and give your text box the instance name **finished_txt**.



6. We are now going to go back to our **Actions** panel (press **F9**) to add a second if statement to our dropObject function. This statement will simply tell Flash to check if all the pieces have been matched when any of the shapes are dropped and to display a message in the finished_txt text box if the answer is yes.

Add the if statement that is shown in the three lines of script highlighted below. Note that the other lines of script have just been included to help guide you as to the spot to place these lines.

```
    } else {  
        event.target.x = objectoriginalX;  
        event.target.y = objectoriginalY;  
        response_txt.text = "Try again";  
        negative.play();  
        response_mc.gotoAndStop(3);  
    }  
    if(score == 4){  
        finished_txt.text = "Well done!";  
    }  
}
```

You can of course change well done to a message of your choice. Note also that if you wanted one of the other types of responses rather than text, you could simply exchange the second line in the if statement. For example, if you had set up a movie clip symbol with multiple frames and given it an instance name such as gameover_mc, you could then add these three lines of script to make Flash display the second frame in that movie clip.

```
if(score == 4){  
    gameover_mc.gotoAndStop(1);  
}
```

Note also that if you had more than four pieces in your game, you would have to change the number four to the number of pieces you have. Whilst I have only used four pieces to make this simple game, there is no reason why you couldn't use the script in these tutorials to create a game with one hundred pieces. You'd just need the patience to build it.

7. Test your game by selecting **Control > Test Movie**. Your score should be increasing each time a piece is placed and when all pieces are matched, you should see the text 'Well done' appear in the dynamic text box. If so, save your game. If not double check the instance name and the script you have added.





Removing the snap to feature of the game

When you test the game, you will see that the shapes lock or snap in to the targets when you drop them in the correct location. This is great for this simple game as it prevents the player from increasing the score by dropping the same shape on a target multiple times. However, there will most likely be a time when you want to create a game that doesn't snap or lock the objects being dragged in to place. Similarly, you may want to remove the snap back to the original location feature of the game.

Rather than creating new script from scratch, you can simply use your existing script and delete some of it to remove these features.

Follow the steps below to do this.

1. Save your existing game file as **shapematchnosnap.fla**. Note that at this stage, you should have multiple .fla files in your collection. These are now resources you can refer to when you want to make similar games.
2. In this new version of your game, we are going to start by **deleting** the following two lines of script.

```
event.target.removeEventListener(MouseEvent.MOUSE_DOWN, pickupObject);  
event.target.removeEventListener(MouseEvent.MOUSE_UP, dropObject);
```

3. **Test your game** to see the change by pressing **Control > Test Movie**. Note that once an object has been dropped on the target, you can then pick it up and move it again. You will probably notice though the object will continue to return to the x and y coordinates of the target as the game has identified that this was the place you picked the object up.
4. Let's remove two more lines to remove this annoying problem. **Delete** the following two lines of script from the else part of the statement.

```
event.target.x = objectoriginalX;  
event.target.y = objectoriginalY;
```

5. **Test your game** to see the change by pressing **Control > Test Movie**. Note that once an object has been dropped on the target, you can now pick it up and drop it into any location. Note also that you can drop objects onto their targets multiple times and increase the score each time. This means that the user could reach the score of four without having matched all the pieces. A tutorial showing a way of avoiding this problem will be added to the Flash Classroom site in the future.

Congratulations

You have now mastered how to add a range of different types of responses to drag and drop games. To consolidate your skills, you should now try to build another game where you add only the responses you want. If you are stuck for ideas, check out some of the games teachers and students have created in our Flash Classroom gallery.

