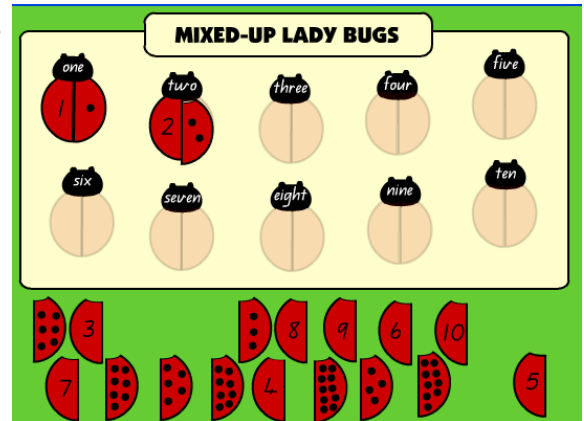




MAKE A DRAG AND DROP GAME WITH TARGETS

In this tutorial, you will learn how to create a drag and drop style game in Flash where the object being dragged will return to its original location if not dropped in the correct place.

An example of this style of game is the 'Mixed-Up Lady Bugs' puzzle shown here. This game is designed to help consolidate students understanding of the numbers 1-10.



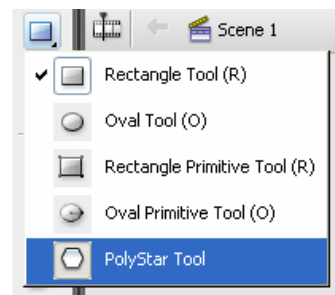
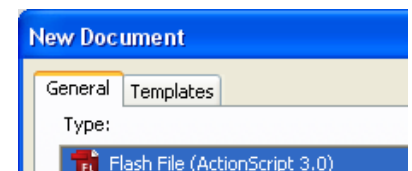
To complete the game, the user must drag each of the parts of the ladybug onto the ladybug with the corresponding number. If the numbers match, the piece will stay on the ladybug. If not, the piece will return to its original place at the bottom of the screen. You can play this game in the Flash Classroom Gallery.

LET'S MAKE A SHAPE MATCHING GAME

The game above is quite time-consuming to make due to the large numbers of pieces. For this reason, this tutorial will focus on making a simple shape matching game where the user has to drag a shape into its matching silhouette. Once you have mastered this tutorial, you will have the skills to create more intricate games such as the Ladybugs example above.

STEPS

1. To begin, open a new Flash file (Actionscript 3.0) in Flash CS3 and save it as **shapematch.fla**.
2. Select a background colour for your game by clicking on the black arrow tool selected and in the **Properties Panel** changing the **Background** colour.
3. In the **Colors** menu on the left hand side of the screen, change both the Stroke Colour and Fill Colour to Black.
4. On the stage, use your **draw and shape** tools to draw four different shapes. Note that Flash CS3 has a fly out menu that is on the Rectangle Tool Button. To access the other shapes in this menu, click and drag your mouse over this tool. You can then select the polystar tool to create polygons or stars.



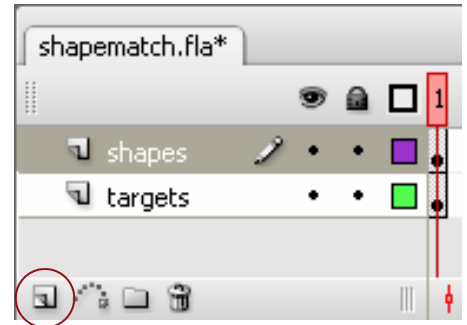


FLASH CLASSROOM - MAKE A DRAG AND DROP GAME WITH TARGETS IN FLASH CS3

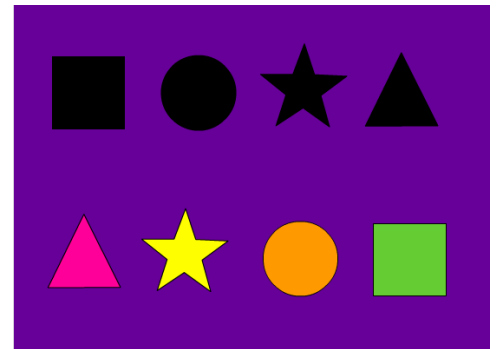
5. Move your four shapes to the top half of your stage. Name the layer that they are on **targets** by double-clicking on Layer 1.

6. Create a new layer above the targets layer by clicking on the **add new layer button** underneath the timeline. (This button is circled here)

Name this layer **shapes**. Your layers should look the same as those shown on the right.



7. Copy each shape and paste a copy of it on the bottom half of the stage on the new **Shape Layer**. Make sure that the shapes are mixed up, so that they are not all under their silhouette. Ensure the silhouettes and shapes are on their right layers. If not, when you drag your coloured shape onto the silhouette, you may find it hides behind it.



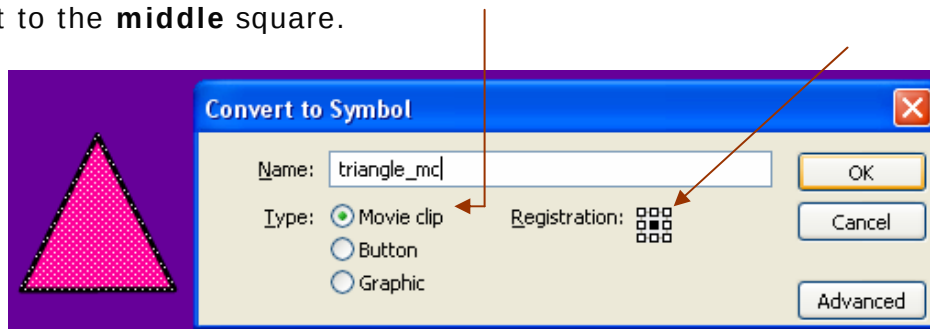
8. Change the colour of the bottom shapes with the **Fill Tool**.

At this stage of the tutorial, your stage should look very similar to the above image.

PREPARING THE SHAPES AND TARGETS

We are now going to convert each of the shapes and targets to movie clip symbols and give each of them an instance name. This is important because in a later step we will add actionscript that will refer to these names. It is a good idea in Flash to get into the habit of using naming conventions. These can be added onto the end of the name you give to each object. For example, in this game I am converting the shapes to movieclip symbols. I will call them **triangle_mc**, **star_mc**, **circle_mc** and **square_mc**. The **_mc** indicates that this object is a movie clip. **For this game, it is essential that you follow the naming conventions I use. If you don't your game won't work.**

9. To begin with we are going to convert each of the **coloured** shapes to movie clip symbols. To do this, select each shape individually and select **Modify > Convert to Symbol**. In the Convert to Symbol box, name your shape and select the **Movie Clip** option. Ensure the **Registration** is set to the **middle square**.





FLASH CLASSROOM - MAKE A DRAG AND DROP GAME WITH TARGETS IN FLASH CS3

In this game, the black shapes or silhouettes are the targets. These shapes need to be converted to movie symbols too so that through actionscript we can make them targets where the other shapes can be dropped.

When doing this we need to ensure we give them a different name from the other shapes. The name should make the symbol clearly recognisable as a target.

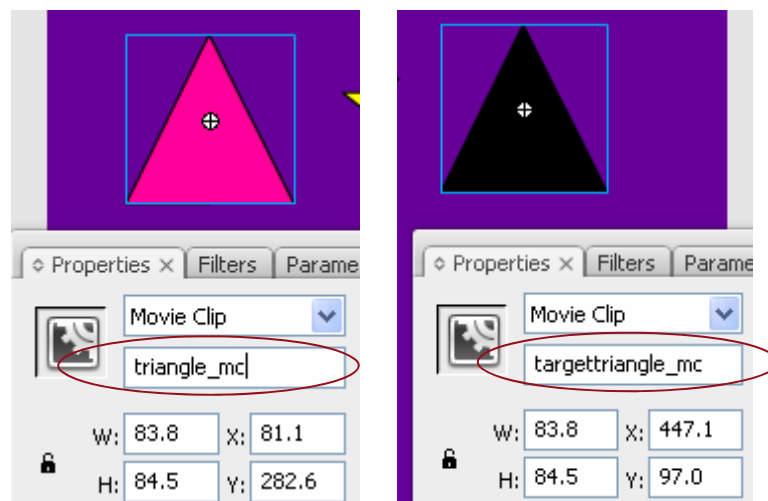
Each target in your name needs to be given a name that begins with the word **target** and ends with **_mc**. The shape name should be in the middle. For example:

I named the shape to be dragged **triangle_mc**

I named the target that the object is to be dropped on **targettriangle_mc**

OK—Let's Keep Going.

10. Select each of the black shapes one by one and convert them to movieclip symbols by selecting **Modify>Convert to Symbol**, giving them a name (e.g. **triangletarget_mc**) and selecting the **Movie Clip** behaviour. Remember to select the middle square in the **Registration** option.
11. We will be referring to these movie clip symbols as targets in the actionscript that we write. Therefore, we will need to enter instance names for each target shape. To do this, select the shape and target on the stage and in the **Properties** panel, enter an **instance name** in the cell circled below. Note that I given each object the same name I gave them when converting them to symbols. e.g. **triangletarget_mc**, **triangle_mc** etc. In this game, your instance names should match the names of your movieclips.



Once you have converted every shape to movie clip symbols and given the black/silhouette shapes instance names, you are ready to add the actionscript that will make your game work. This script is on the next page.

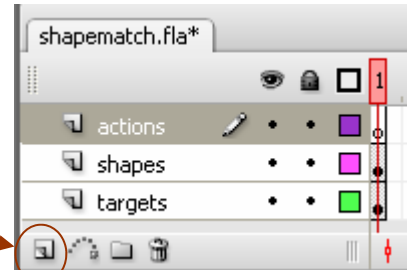


ADDING THE ACTIONSCRIPT

We are now going to add some actionscript to the game to make it interactive. Flash CS3 uses **Actionscript 3.0** and the script we will use to create the interactivity for this game is significantly different to the script we used in previous versions of Flash. A main difference is that in Flash CS3, you can only add script to keyframes in the timelines. The good news is that the new way of scripting is much easier as you can place all of your script in the one spot. This makes it much easier to check if you have any errors.

Let's add the actionscript

12. Add a new layer to your timeline by clicking on the **Insert Layer** button. Name this layer **actions**. Your timeline should look like this.
13. Click on the first keyframe of the **Actions** layer and press **F9** to open the Actions Panel. Note that you can open and close the Actions panel by clicking F9.
14. In the Actions Panel, **add the following script**. Ensure that you use the correct type of brackets and remember to use capitals where shown.



```
var objectoriginalX:Number;
var objectoriginalY:Number;

triangle_mc.buttonMode = true;
triangle_mc.addEventListener(MouseEvent.CLICK, pickupObject);
triangle_mc.addEventListener(MouseEvent.CLICK, dropObject);

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

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.CLICK, pickupObject);
        event.target.removeEventListener(MouseEvent.CLICK, dropObject);
        event.target.buttonMode = false;
        event.target.x = matchingTarget.x;
        event.target.y = matchingTarget.y;
    } else {
        event.target.x = objectoriginalX;
        event.target.y = objectoriginalY;
    }
}
```



FLASH CLASSROOM - MAKE A DRAG AND DROP GAME WITH TARGETS IN FLASH CS3

15. Double check that you have entered your script correctly. Remember that it needs to be exactly the same in order to work. One missed semi colon or wrong type of bracket can prevent your game from working.

EDITING THE ACTIONSCRIPT

In the script you just typed in, you will see three lines of script that refer to the instance names of one of the movieclip symbols (triangle_mc). This script is shown here.

```
triangle_mc.buttonMode = true;  
triangle_mc.addEventListener(MouseEvent.CLICK, pickupObject);  
triangle_mc.addEventListener(MouseEvent.CLICK, dropObject);
```

To make our game work, we need to add these three lines for each of the shapes in our game that we want to be able to be dragged. We then will need to change the instance names in each of the three lines to match a shape.

16. In your **Actions** panel, select the three lines shown above and **make copies** of it under this script so that you have one lot of three lines for each shape.
17. Edit the **instance name** in each of the copies until you have three lines referring to each shape in your game.

Once you have edited this part of your script, it should look like this. The only difference should be the names of your shapes if you chose different shapes to the ones I used.

```
3  
4 triangle_mc.buttonMode = true;  
5 triangle_mc.addEventListener(MouseEvent.CLICK, pickupObject);  
6 triangle_mc.addEventListener(MouseEvent.CLICK, dropObject);  
7  
8 star_mc.buttonMode = true;  
9 star_mc.addEventListener(MouseEvent.CLICK, pickupObject);  
10 star_mc.addEventListener(MouseEvent.CLICK, dropObject);  
11  
12 circle_mc.buttonMode = true;  
13 circle_mc.addEventListener(MouseEvent.CLICK, pickupObject);  
14 circle_mc.addEventListener(MouseEvent.CLICK, dropObject);  
15  
16 square_mc.buttonMode = true;  
17 square_mc.addEventListener(MouseEvent.CLICK, pickupObject);  
18 square_mc.addEventListener(MouseEvent.CLICK, dropObject);  
19
```

18. Providing you have entered your script and edited it correctly, your game should now work. To test your game, select **Control > Test Movie** (or Ctrl + Enter).

If your game works you are ready to save your game. If not, check your script against mine and also check to ensure you have given an instance name to each shape and target.





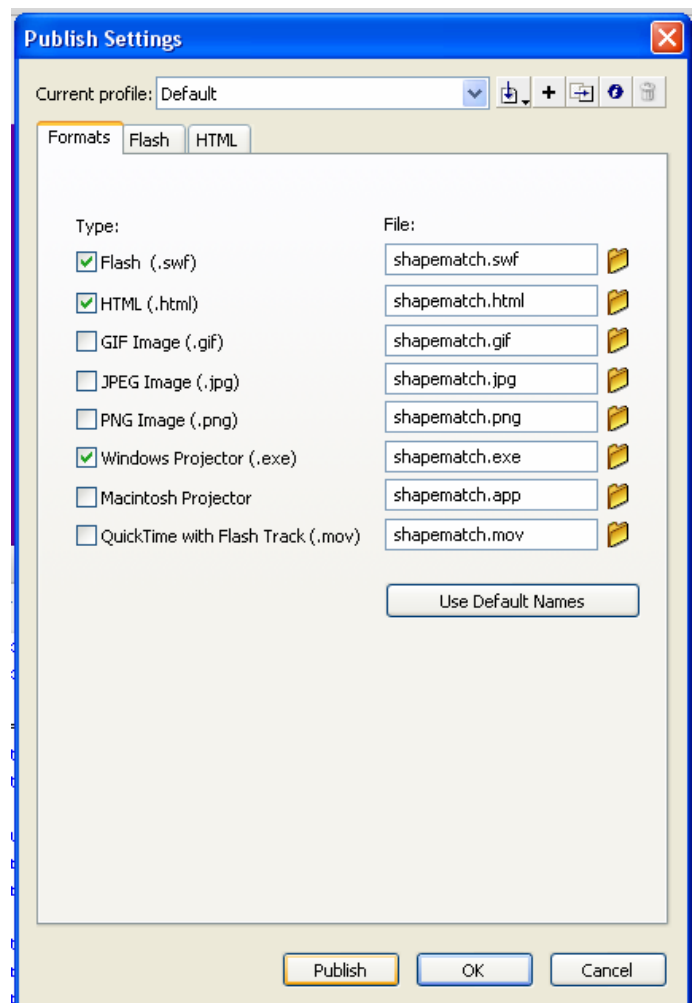
PUBLISH & SHARE YOUR GAME

19. Save your work by selecting **File > Save**.

20. Turn your flash file into a game that can be played on any computer by publishing it in different file formats. To do this select **File > Publish Settings**. The box shown to the right will appear.

21. Tick the file formats you want and click on the **Publish** button. These files will be saved in the same location you saved your original file.

If you want your game to be a standalone file that can be played on Windows or Macintosh machines - ensure you check the Windows Projector (.exe) and Macintosh Projector (.exe) format options.



If you've got to here successfully, well done! If not, ask your teacher to help you troubleshoot. If after this, you're still stuck - send your file through to us via email at the Flash Classroom.