



MAKE A DRAG AND DROP GAME

In this tutorial, you will learn how to make a simple drag and drop game in Flash CS3 using Actionscript 3 (AS3). To create this game you will make a series of movie clips and then add some actionscript which will enable each movie clip to be able to be dragged and dropped.

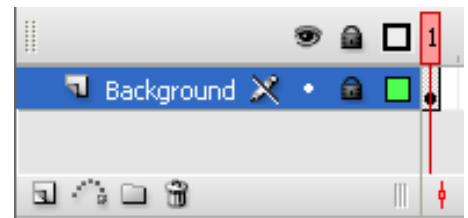
The example I will use for this tutorial is a dress-up game that I created for the Gidgits project. A screenshot of the game is shown below.

In this game, users can choose different clothing items and drag them on or off the Gidgit girl. Note that the clothing is on a layer above the girl image so that it always appears in front.



SETTING UP YOUR FILE

1. Open Flash CS3 and select the Flash File (Actionscript 3.0) type.
2. On layer 1, add any elements you want in the background of your game. In my example, the text 'Gidgits Style', the girl and the white rounded rectangle shape are located on this layer. These are all of the items on the stage that will not be able to dragged.
3. Name the layer 'Background' by double-clicking on the text **Layer 1** and typing in the word '**Background**'.
4. Lock the layer by clicking on the dot on the layer that is underneath the lock icon.
5. Add a second layer by clicking on the Insert Layer button at the bottom of the timeline. Double click on the text Layer 2 and rename your layer **Objects**.



On this layer you will draw or insert all of the objects that you want to be able to drag in the game. Each of these objects will need to be put through the process outlined on page 2 of this tutorial.

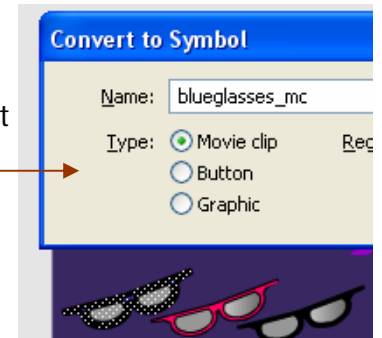




CREATING YOUR MOVIE CLIP SYMBOLS

To make the objects in your game draggable, we need to convert each drawing or imported object into a movie clip symbol. Follow these three steps to convert each object and to give each object an instance name.

6. Select an object that the user will be able to drag in the game. Once selected, convert this object to a movie clip symbol by choosing **Modify > Convert to Symbol** (or F8).



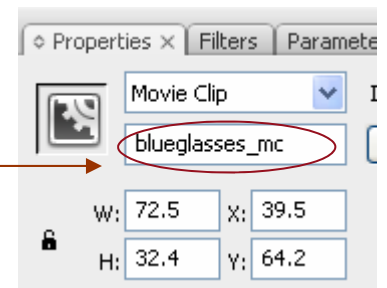
7. We are now going to give the object a name. In Flash developers usually give objects a name that ends with a naming convention. If it is a movie clip symbol, it will end with `_mc`. If it is a button symbol, the name ends with `_btn`. As all of the objects you want to be dragged are going to be movie clip symbols, the names you should give your symbols should end with `_mc`.

In my example above, I have called my new symbol `blueglasses_mc`.

Name your symbol and select the **Movie clip** type option. Click **OK**.

8. Now that you have converted your object into a symbol, we need to give your object an instance name. This is because when we add script later, the script will refer to the **instance name** of the object.

To add the instance name for your new symbol, click on the symbol you just converted and in the **Properties** panel, type in the name of your symbol in the **instance name cell**.



Repeat steps 6 - 8 for each object in your game that the user will be able to drag.

ADDING ACTIONSCRIPT TO MAKE THE GAME WORK

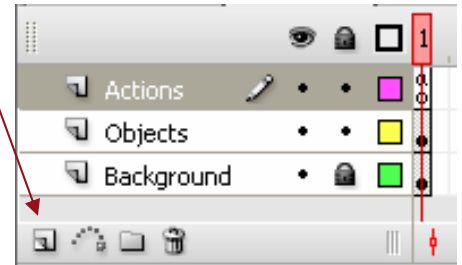
Flash CS3 is different to previous versions of Flash as it has moved to a new, more sophisticated scripting language called Actionscript 3.0. One key difference with Actionscript 3.0 is that it can't be attached to movie clips or buttons. All of the script is placed on the main timeline or on the timelines within symbols. Whilst this is a new way of thinking for those that have done Flash scripting in the past, it has many benefits - the main being that in most Flash files all your script is in the one place. This saves time and makes it much easier to troubleshoot.

In the steps on the following page, you'll set up an Actions layer and add and edit the script you'll need to make your game work.



9. Add a third layer to the timeline by clicking on the **Insert Layer** button. Rename this layer **Actions**.

Your timeline should now look like the one pictured here. Note that there is a small letter a above the dot in the first keyframe of the Actions layer. This indicates that script has been added to that frame. You won't see an a in your frame yet, but you will after you add some script in the next couple of steps.



10. 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.
11. In the Actions Panel, **add the following script**. Pay particular attention to ensure you get the right types of brackets in the right spots and that your script is exactly the same as that shown below. Flash is case sensitive which means that you need to use uppercase letters where uppercase letters are shown and lowercase letters where they are shown. Similarly, if you have used capitals in your instance name, you will need to ensure you use them when you type the name in your script.

When you add the script below, **replace the blueglasses_mc** in the first two lines to the instance name of one of your movie clips.

```
blueglasses_mc.addEventListener(MouseEvent.CLICK, pickupObject);
blueglasses_mc.addEventListener(MouseEvent.CLICK, dropObject);
```

```
function pickupObject(event:MouseEvent):void {
    event.target.startDrag(true);
}
function dropObject(event:MouseEvent):void {
    event.target.stopDrag();
}
}
```

The script works by telling the Flash player to watch or listen for the users press of the mouse button. The first line says to Flash that if the user presses the mouse button down whilst over the movieclip symbol, to run the function pickupObject. This sounds tricky, but it basically tells Flash to jump to the script below in lines 4,5 and 6 and to run through it. The script in line 4 tells the Flash player to start dragging the movieclip. The second line says to Flash that if the user lifts their finger off the mouse button when it is on the movieclip, to run the function that is named dropObject. This tells Flash to run through lines 7,8 and 9 of the script. These lines tell Flash to drop the object that the user has been dragging.

Actionscript can be a bit tricky to understand at first so don't be worried if you can't get your head around what's happening. For now, just follow each step carefully to make your game work.





12. If you've entered the script correctly, the object that you named where I had `blueglasses_mc` should be able to be dragged and dropped. Test that your object is able to be dragged and dropped by clicking on **Control > Test Movie**.
13. If your object could be picked up and dragged and dropped, your script has been entered correctly. If so, you are up to the final stage. This involves copying the first two lines of script and pasting a copy of it for each symbol that will be able to be dragged and dropped. You will need to replace the instance name in each two lines to show the name of each object.

The script below shows how I have copied the first two lines and then just changed the instance name to `pinkglasses_mc`.

```
blueglasses_mc.addEventListener(MouseEvent.CLICK, pickupObject);
blueglasses_mc.addEventListener(MouseEvent.CLICK, dropObject);

pinkglasses_mc.addEventListener(MouseEvent.CLICK, pickupObject);
pinkglasses_mc.addEventListener(MouseEvent.CLICK, dropObject);
```

Once you've added the two lines for each object, all objects in your game should be able to be dragged and dropped. If not, check through your script to ensure it matches mine and is error free.

Don't forget to save your work and to prepare it to share using the Publish Settings option in the File menu. To make it a stand-alone game, select the Windows (.exe) or Macintosh (.hqx) projector options in the Publish Settings menu and click publish.

Extending your game

The Flash Classroom site contains a number of other tutorials covering different types of drag and drop games. These include a tutorial which snaps the object back into the original location if it is moved to an incorrect spot and another that provides the user with a response if they move the object to a correct location.

These games are slightly more sophisticated and provide the user with more feedback. They are very useful for a range of educational games including sequencing activities, matching games, cloze activities and more.

