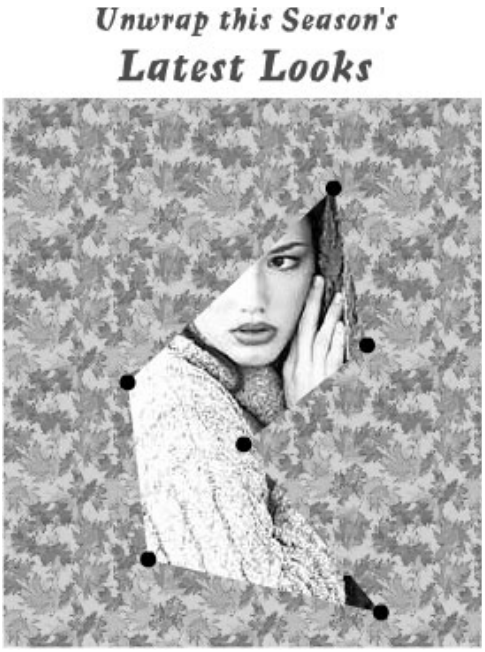

Figure 7.91 Combining draggable movie clips (the vertices), drawing methods (the lines connecting the vertices), and dynamic masking (uncovering the image "behind" the wrapping paper).

Figure 7.92 A movie clip containing a bitmap cityscape will be the masked movie clip.

Using Dynamic Masks

You can turn any movie clip into a mask and specify the movie clip to be masked with the method `setMask()`. To use the method, you define the movie clip you want to be masked as the target path before the method and then define the movie clip you want to act as a mask as its parameter: `masked.setMask(mask)`. Because you can control all the properties of movie clips, you can make your mask move or grow and shrink in response to viewer interaction. You can even combine the `setMask()` method with the movie-clip drawing methods to create masks that change shape (**Figure 7.91**).

An effective combination is to assign `startDrag()` and `stopDrag()` methods to the movie clip mask and create draggable masks. By adding `startDrag()` to an onPress handler and `stopDrag()` to an onRelease handler, your viewer can control the position of the movie clip mask.

To set a movie clip as a mask:

1. Create a movie clip, place an instance on the Stage, and name it in the Property Inspector (**Figure 7.92**).

   This movie clip will be masked.

continues on next page

2. Create another movie clip, place an instance on the Stage, and name it in the Property Inspector (**Figure 7.93**).
   This movie clip will act as a mask.
3. Select the first frame of the root Timeline, and open the Actions panel.
4. Choose Objects > Movie > Movie Clip > Methods > setMask.
5. In the Object field of the Parameters pane, enter the target path to the movie clip you want to be masked.
6. In the Parameters field, enter the name of the target path to the movie clip you want to use as a mask (**Figure 7.94**).
7. Test your movie.
   The opaque shapes of the mask movie clip reveal the masked movie clip (**Figure 7.95**).

Figure 7.93 A movie clip of vertical shapes will be the mask movie clip.

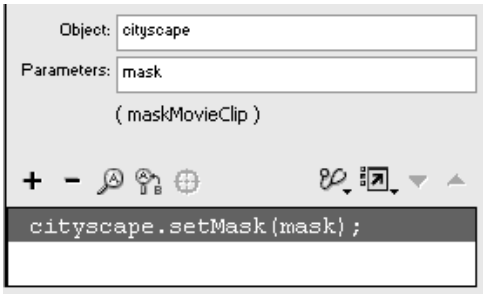

Figure 7.94 The mask movie clip will mask the cityscape movie clip.

Figure 7.95 The result of the `setMask()` method.

✔ Tips

- You can specify the root Timeline as the movie clip to be masked, and all the graphics on the main Timeline (including movie clips on the Stage) will be masked. To do so, enter `_root` in the Object field of the Parameters pane.
- Holes in the opaque shapes of your mask movie clip are not recognized and do not affect the mask (**Figure 7.96**).
- To undo a `setMask()` method, use the `null` keyword for its parameter as follows: `masked.setMask(null)`.

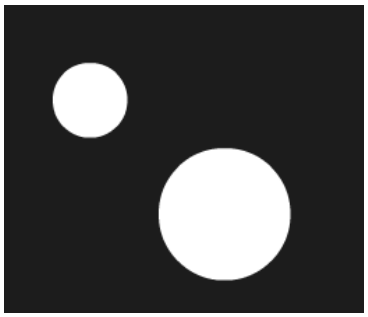

Figure 7.96 If this shape were to be used in a movie clip for the `setMask()` method, Flash would recognize only a square for the mask.

Figure 7.97 You'll create a draggable mask to uncover this movie clip to make the viewer look for the monkey in the bush.

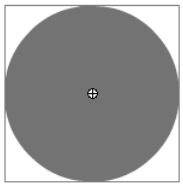

Figure 7.98 A simple circle will be the mask.

To create a draggable mask:

1. Create a movie clip, put an instance of it on the Stage, and name it in the Property Inspector.

   This movie clip will be masked (**Figure 7.97**).

2. Create another movie clip and put an instance of it on the Stage, and name it in the Property Inspector.

   This movie clip will act as a mask (**Figure 7.98**).

3. Select the first frame of the root Timeline and open the Actions panel.

4. Choose Objects > Movie > Movie Clip > Methods > setMask.

5. In the Object field, enter the target path of the movie clip to be masked.

6. In the Parameters field, enter the target path of the mask movie clip (**Figure 7.99**).

continues on next page

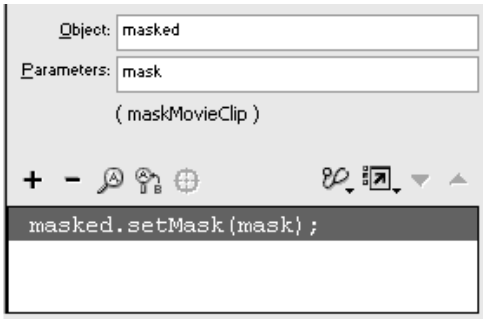

Figure 7.99 Create the mask of the circle over the masked image (the monkey in the bush).

7. Choose Objects > Movie > Button > Events > onPress.
8. In the Object field, enter the target path of the mask movie clip (**Figure 7.100**).
9. Choose Actions > Movie Clip Control > startDrag.
10. In the Target field, enter the target path of your mask movie clip and check the Expression box (**Figure 7.101**).
11. Select the closing brace of the onPress event handler, and choose Objects > Movie > Button > Events > onRelease.
12. In the Object field, enter the target path of the mask movie clip (**Figure 7.102**).
13. Choose Actions > Movie Clip Control > stopDrag.
14. Test your movie.

    The movie clip acts as a mask while the onPress and onRelease handlers provide the drag-and-drop interactivity (**Figure 7.103**).

Figure 7.100 The `onPress` event handler is created for the movie clip called mask.

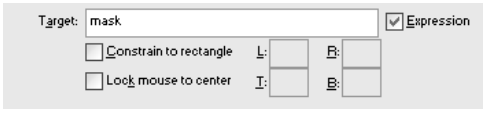

Figure 7.101 The Parameters pane for the `startDrag` action.

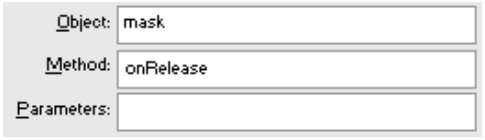

Figure 7.102 The `onRelease` event handler is created for the movie clip called mask.

```
masked.setMask(mask);
mask.onPress = function() {
 startDrag(mask);
};
mask.onRelease = function() {
 stopDrag();
};
```

Figure 7.103 The final ActionScript code (top) assigns interactivity to the movie clip that makes it act as a mask and enables it to be dragged.