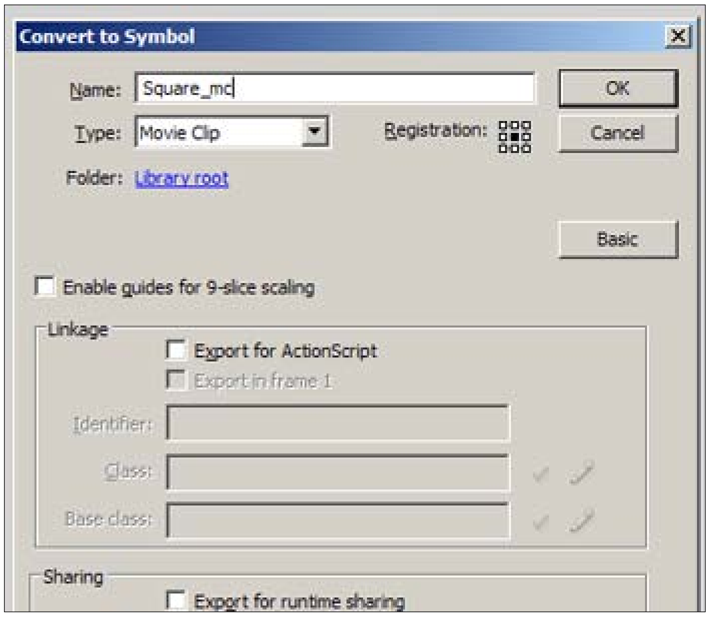
CGT 353: Principles of Interactive and Dynamic Media
Symbols, Instances, and Libraries

Types and Purpose of Different Symbols:

- Three types: **1) graphic, 2) button, and 3) movie clip**
- Each basically differ in their behavior, which can change at any time in the Library…..


Convert to Symbol

Name: Square_mc

Type: Movie Clip

Registration:

Folder: Library root

OK

Cancel

Basic

☐ Enable guides for 9-slice scaling

Linkage

☐ Export for ActionScript

☐ Export in frame 1

Identifier:

Class:

Base class:

Sharing

☐ Export for runtime sharing

Editing Symbols:

- Double click a symbol to access its timeline and make changes to it.
- Can also use the symbol dropdown menu.
- <u>Making changes in the symbol timeline will affect all instances</u>.

Registration of Symbols:

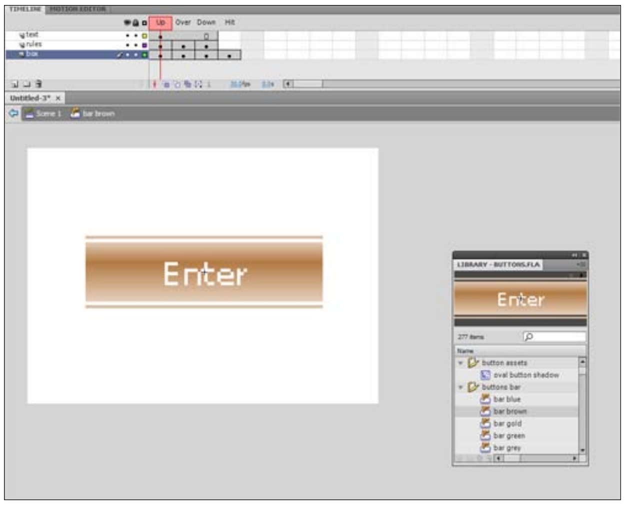
- The **registration point** of the main movie is the upper-left hand corner.
- You must physically move the content within a symbol to chance the registration point, which will affect all instances of the symbol…
- **Note:** <u>The RP is different from the **origin** of a clip.</u>

Graphic Symbols:

- Used primarily for static components….
- Graphic symbol timelines are <u>synced to the main timeline.</u>
- When the <u>main timeline stops, the graphic timeline stops.</u>
- Should not be used for clips that you need to loop.

- **Note:** <u>Graphic symbols should be used sparingly. Movie clips are usually preferable.</u>

Button Symbols:



- A symbol that behaves like a push-button.

- Contains special frame for up, over, down, and hit states.

- **Warning:** Do NOT use dynamic or input text fields in your buttons.

Movie Clip Symbols:

- <u>Movie clip timelines keep playing regardless of the main timeline.</u>

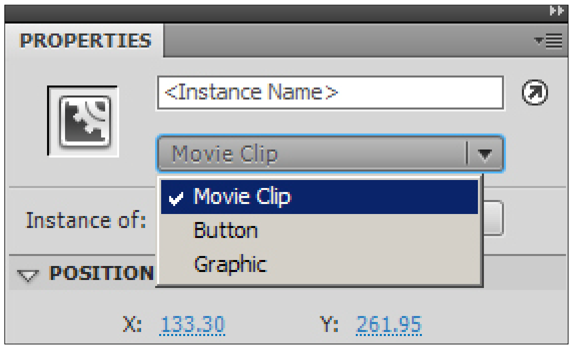
- <u>Can include all other times of symbols within them.</u>

- Should use when you want a clip to keep playing.

- Need to either **a) include a stop() action inside it** or **b) directly tell the individual instance to stop via ActionScript**

Redefining Symbols:

- Can change a symbol's behavior in the property panel, effectively changing the type of symbol.



- This <u>will not change the behavior of the symbol in the library</u>, which can be problematic.

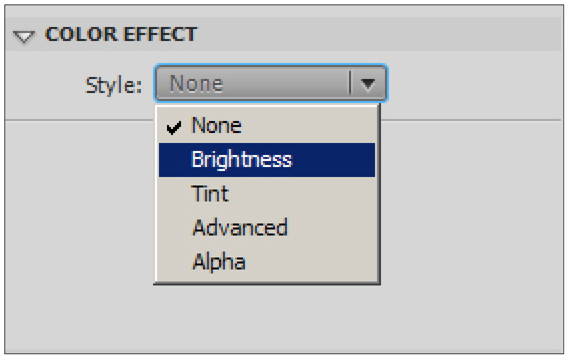
- Can also break apart a symbol instance, which will not affect the main library symbol or any other instances.

Swapping Symbols:

- Can swap different symbols while still retaining the properties of a particular instance.

The Color of Symbols:

- Allows you to apply a particular effect to a symbol, including:
    - Brightness
    - Tint
    - Alpha
    - Advanced

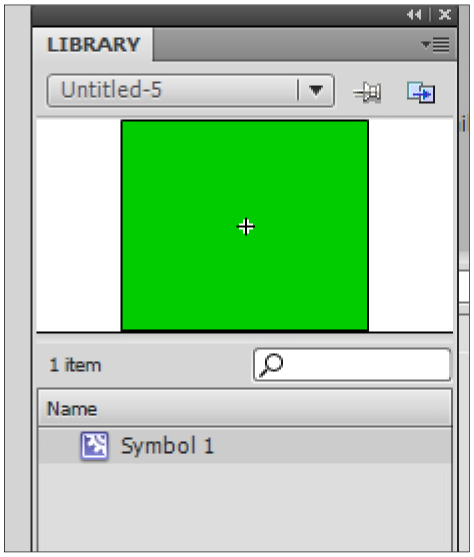


IMPORTANT: Compound or Nested Symbols: Precedence Rules

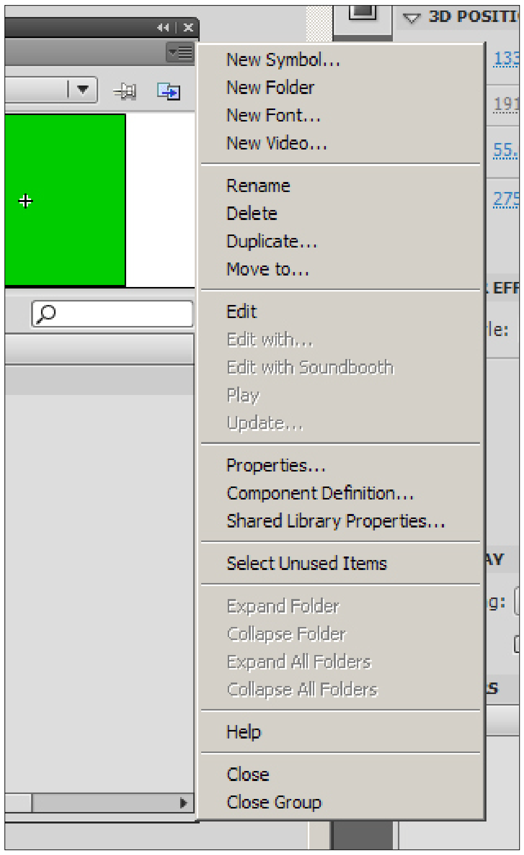
- Can recombine symbols to achieve different functionalities.
- <u>Putting a button into a graphic symbol disables the button.</u>
- <u>Putting a movie clip into a graphic will allow the animation to play, but any buttons or sounds within the movie will be disabled.</u>
- <u>You can put a graphic or movie clip into any of the button states.</u>
- <u>You can put a graphic or button into a movie clip.</u>

Libraries:

- Libraries <u>store symbols and assign symbols their basic behaviors</u>.
- Every Flash file that has symbols can be a library.
- All you have to do is open a particular fla file as a library, then drag and drop symbols either onto the stage or into the current files library.
- Remember that a <u>symbols **registration point** is dependant on where you place it on the stage.</u>
- Also remember that the symbols in the library control the basic parameters of all instances of that symbol.



The Library Panel:



- Use the buttons on the right of the panel to toggle its appearance.
- **File menu** has a number of functions that you can perform.
- Remember that <u>once you have deleted a symbol, there is no way to undo it.</u>
- Deleting a symbol <u>will cause all instances of the symbol to disappear.</u>
- Check the **usage count** before deleting a symbol.
- Can now view **multiple libraries** in a single panel.
- <u>All library items are NOT exported with .swf export.</u>

Common Libraries:

- Useful for creating basic movies....
- Used to have a number of graphics and movie clips in them....

Shared Runtime Libraries:

- By setting the **linkage properties** of items in the library, you can share those assets with other swfs
- Can use movie clips, fonts, etc…
- The advantage of doing this is to minimize the amount of file size in “primary” swf’s.

Procedure:

1. Create your fla that will contain the majority of your assets

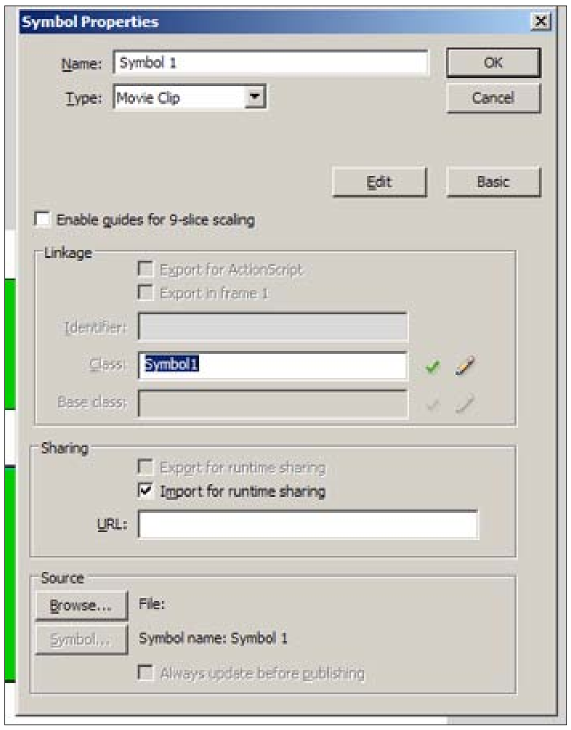
2. Right click each symbol in the library and choose “Linkage”

3. For Linkage, select “Import For Runtime Sharing” to link to the asset in the source document.

4. Enter an identifier for the symbol, bitmap, or sound that is identical to the identifier used for the symbol in the source document. Do not include spaces.

5. Enter the URL where the SWF source file containing the shared asset is posted, and click OK.

6. In the destination document, do one of the following:



Name: Symbol 1
Type: Movie Clip
[ ] Enable guides for 9-slice scaling

Linkage
[ ] Export for ActionScript
[ ] Export in frame 1
Identifier:
Class: Symbol1
Base class:

Sharing
[ ] Export for runtime sharing
[✓] Import for runtime sharing
URL:

Source
File:
Symbol name: Symbol 1
[ ] Always update before publishing

7. Select File > Open.

8. Select File > Import > Open External Library.

9. Select the source document and click Open.

10. Drag the shared asset from the source document Library panel into the Library panel or onto the Stage in the destination document.

Assigning Linkage Identifier to a Library Item:

1. Select the font item in the Library panel.
2. Do one of the following:
   - Select Linkage from the Library Panel menu.
   - Right-click (Windows) or Control-click (Macintosh) the font symbol name in the Library panel, and select Linkage.
3. Under Linkage, select Export for Runtime Sharing.
4. In the Identifier text field, enter a string to identify the font item.
5. In the URL text field, enter the URL of the SWF file that contains the font item.