CONTENTS

Authoring from Lingo

Lingo can generate a movie's two most important components—cast members and a finished Score. Each of these by itself lets you automate authoring, but you can use them together to generate entire movies.

Building cast members from Lingo

Typically, you use the Director interface to:

- Select a location in the Cast window
- Fill the location by importing content or creating content in one of Director's editing windows
- Set cast member properties in the appropriate Properties dialog box

Lingo can automate creating cast members by performing many of these same tasks. Automating a process makes it faster and more reliable.

You need to do several things to create cast members from Lingo:

- Use the new() function or importFileInto command to create the cast member, assign it a type, and if you want, to specify its location in a Cast window.
- Use the importFileInto function or set the fileName property to assign content to a cast member.
- Finish setting up the new cast member by setting its properties as needed.

Assigning content to a cast member

The new() function gives a new cast member a location in a Cast window and determines the cast member's type—nothing more.

Assign content to a cast member by importing a file into the cast member or linking the cast member to an external file.

To import a file into a cast member:

Use the importFileInto command. However, not all import options available in the Director interface are available with the importFileInto command. For example, the importFileInto command can't import a bitmap's palette.

The following handler uses importFileInto to import a sound file into the cast member Sound, which already exists. The variable newFile already has a sound file already assigned to it.

```
on newSound newFile
    importFileInto(member "Sound", newFile)
end
```

If the cast member doesn't exist yet, create the cast member and import content into it by embedding the new() function as the first argument for importFileInto.

In the following sample handler, the first argument of the importFileInto command creates a new bitmap cast member for content to be imported:

```
on newBitmap newFile
    importFileInto(new(#bitmap), newFile)
end
```

In this handler, the newFile argument contains the file to be imported.

Linking to an external file

To link a cast member to an external file, set the cast member's fileName property to the file name of the external file.

This handler creates a new digital video cast member and then links it to an external digital video file:

```
on newVideo newName, newFile
    set newMember = new(#digitalVideo)
    set the name of newMember = newName
    set the fileName of newMember to newFile
end
```

To use this handler to create a new digital video cast member named Parade and link it to the external file Parade.mov, issue the calling statement:

```
newVideo "Parade", "Parade.mov
```

This handler creates a sound cast member from a sound file:

```
on newSound newFile
    set newMember = new(#sound, member ¬
    3 of castLib "Content")
    set the name of newMember to newFile
    set the fileName of newMember = newFile
end
```

The new() function in this handler creates a new cast member. Using #sound as the argument makes the cast member a sound cast member.

The on newSound handler specifies the cast member's content by setting the fileName of member instead of using the importFileInto command. This makes the sound cast member a linked external sound instead of an internal sound.

This handler creates a script cast member:

```
on newScript
    set newMember = new(#script, member ¬
    2 of castLib "Content")
    set the name of newMember to "Script"
    set the scriptType of newMember to #score
    set the scriptText of newMember to the text ¬
    of member "Script Content"
end
```

The Lingo that the script contains is the text in the field cast member Script Content, which was already in the cast.

Setting cast member properties

A cast member is a collection of its properties. The specific properties depend on the cast member's type.

To set up a cast member entirely from Lingo, specify the cast member's properties.

You can automate setting up and editing cast members using the = operator or set command to set the cast member's properties.

Generating Score from Lingo

Just as Lingo can automate the work you'd otherwise do manually to create cast members, it can also automate setting up the Score.

Lingo can perform the same tasks that you perform manually in the Score—such as selecting frames, specifying what's in each channel, and animating sprites over a series of frames—by creating a new frame and then specifying each channel's content. It repeats this, frame by frame, until the entire sequence of frames is set up.

Lingo can add, edit, or delete frames. The number of frames you generate and what you set for each one varies, but you generally do the following each time you generate Score from Lingo:

1. Use the beginRecording keyword to tell Director that a Score generation session is starting.
2. Specify what goes in the current frame's channels.
3. Use the updateFrame command to enter the changes for the frame.
4. Advance to the next frame and specify its content; delete frames if appropriate.

   Repeat loops are useful for creating a series of frames. Examples later in this section show how repeat loops can help manage generating a series of frames.
5. Continue adding or deleting frames and specifying them as needed until the series of frames is complete.
6. Use the endRecording keyword to let Director know that the Score generation session is finished.

For example, the following handler makes the Score editable and then generates 100 frames in which a sprite moves across the Stage. Each cycle of the repeat loop moves the sprite one pixel to the right, and uses the updateFrame command to enter the changes and insert a new frame:

```
on animBall
    beginRecording
    repeat with counter = 0 to 100
        sprite(4).member = castLib(1).member(1)
        sprite(4).locV = 140
        sprite(4).locH = counter
        updateFrame
    end repeat
    endRecording
end animBall
```

This handler assigns a set of sprites to frames 10 through 20:

```
on addSprites
    beginRecording
    set the frameLabel = "Start"
    repeat with counter = 10 to 20
        sprite(1).member = member(1)
        sprite(3).member = member(10)
        sprite(4).member = member(11)
        sprite(5).member = member(12)

        repeat with i = 3 to 5
            sprite(i).locV = 200
            sprite(i).locH = 200
        end repeat
        if counter = 20 then member(7).frameScript ¬
        = castLib("Scripts").member(7)
        updateFrame
    end repeat
    endRecording
end
```

This handler does several things to generate Score:

- The term beginRecording tells Lingo that a Score recording session is beginning.
- The second line sets a marker named Start.
- The first repeat loop generates frames one by one and specifies the content for each channel. The nested repeat loop sets the vertical and horizontal location of sprites 3 through 5.
- The statement that starts with if counter = 20 checks whether the current frame is frame 20, and if it is, makes cast member 7 in the cast Scripts the frame script for frame 20.
- The updateFrame command enters the new settings and advances to the next frame.
- The term endRecording tells Lingo that the Score recording is done.

Specifying a frame's content

Lingo can specify each channel's content during a Score recording session. The following table lists Lingo that can be set for each channel:

Channel	Lingo
Marker	the frameLabel
Tempo	the frameTempo
Palette	the framePalette
Transition	the frameTransition
Sound channel 1	the frameSound1
Sound channel 2	the frameSound2
Script	the frameScript
Sprite channels	Sprite properties such as member, loc, locH, locV, moveable, and scriptNum

When the frame's content is complete, use the updateFrame command to enter the new content.

The updateFrame command enters changes to the current frame and then advances to the new frame. When Lingo is in the new frame, you can set it up as you did in the previous frame.

Note: The duplicateFrame command copies everything in the current frame. If the current frame contains something you don't want to copy, such as a frame script, consider using a clearFrame command to jump to a new frame. Also consider including Lingo that clears any cells that might have been unintentionally copied.

Adding and deleting frames

Several commands are available to add or delete frames. The following table lists these commands and their result:

Command	Result
clearFrame	Deletes everything in the current frame, but remains in the frame.
deleteFrame	Deletes the current frame. The next frame then becomes the current frame.
duplicateFrame	Makes a copy of the current frame and inserts the copy in the next frame. The new frame then becomes the current frame. This command does the same as the insertFrame command.
insertFrame	Makes a copy of the current frame and inserts the copy in the next frame. The new frame then becomes the current frame. This is the same as the duplicateFrame command.
updateFrame	Enters the changes made to the current frame and then advances to the next frame.

Keeping the Stage from changing

You can keep the Stage display constant during a Score recording session by setting the updateLock movie property to TRUE before Lingo updates the Score. If the updateLock is FALSE, the Stage updates to show a new frame each time the frame is entered by the updateFrame command. If the updateLock is TRUE, the Stage doesn't change as new frames are generated.

Changing movie content

Lingo can change a movie's content by switching the casts that the movie uses. This allows you to reuse the same Score in a variety of ways. For example, you can publish a new edition of a multimedia journal by distributing the updated external cast files to your readers or by replacing the cast of a movie available online. You can make a multilingual movie by creating casts with text and fields in different languages and using the cast for the language that a user chooses.

To switch content while a movie plays, use one of the following techniques:

- Change the cast number assigned to the sprite. Lingo treats a sprite's cast member number (memberNum) and its cast number (castLibNum) as separate properties. Changing one doesn't change the other.

  Because the cast member number remains the same, the sprite takes the cast member with the corresponding cast member number in the new cast.

  For example, suppose that cast member 1 of cast number 1 is assigned to sprite 1. The following statement assigns cast number 2 to sprite 1:

  ```
  sprite(1).castLibNum = 2
  ```

  When using this approach, make sure that similar cast members have the same cast member number so that they appear in the correct sprites.

- Change the file assigned to an external cast. For example, if a movie has an external cast named Daily News, this statement makes the cast file Tuesday.cst the content for Daily News:

  ```
  castLib("Daily News").filename = "Tuesday.cst"
  ```

  This statement assumes that both casts are in the same folder as the movie or projector.

Generating entire movies

Together, cast creation and Score recording can generate entire movies. By making copies of the movies you construct and saving them in separate files, you can use the original movie as a template that produces many movies with each one customized according to the values you send it. For example, you can generate a set of multimedia invitations to a party with each invitation having a personalized name and instructions for what to bring.

Including final components

Make sure the movie includes the movie scripts that don't have a specific assignment in the Score. In particular, be sure to add movie scripts that contain any `on prepareMovie`, `on startMovie`, `on stopMovie`, and primary event handlers.

Generating a separate movie file

When the movie is complete, you can save it in a separate file by using the `saveMovie` command and including a file name as the second parameter.

For example, this statement saves the current movie as a separate file named Jane Invitation:

```
saveMovie "Jane Invitation"
```

You probably don't want the Lingo template that you used to generate the cast and Score to be in the final version that you distribute. Delete these scripts from the movie if appropriate.

INDEX

N

P

R

S

U