MicroWorlds™ 2.0 User's Guide

Macintosh Version

Acknowledgments

Software Concept
Dr. Seymour Papert, MIT
Brian Silverman

Design Group
Mario Bergeron
Paula Bontá
Billo Diallo
Brian Silverman
Alain Tougas
René Yelle

Software Engineers
Billo Diallo
Mario Bergeron
Paula Bontá
Brian Silverman

Programmer/Tester
René Yelle

Project Manager
Alain Tougas

Writers
Alain Tougas
Sharnee Chait

Editing
Eric Brown
Geni Dresher

Package Design
DRK, Inc.
Publicité Stavro

Graphics and Layout
Julien Perron
Chantal McMillan
Le Groupe Flexidée

MicroWorlds Version 2.0 is based on MicroWorlds Project Builder. We would like to thank everyone who was involved in the design and testing of the original version of MicroWorlds, including LCSI staff members, Canadian and American teachers and students who tested it, and special contributors.

Legal deposit, 1st semester 1996

ISBN 2-89371-460-9
Printed 1-96 (REV. 3-98)

Contents

Preface

The *User's Guide* describes the techniques you need to get the most out of MicroWorlds. The MicroWorlds Vocabulary can be accessed through the Help menu. Choose ? from the Help menu, then click on an icon or object on the page to get quick information. The Vocabulary item in the Help menu contains a description of each primitive, in alphabetical order, and an example of its use.

This book has four sections:

Section 1, Basic Techniques describes MicroWorlds objects and concepts. Each menu item and each item in the Tool Palette, the Drawing Center, and the Shapes Center is explained.

Section 2, Logo Programming explains the grammar and syntax of the Logo language on which MicroWorlds is based.

Section 3, Advanced Techniques shows programming techniques unique to the MicroWorlds Logo environment. These techniques can help you build more complex projects.

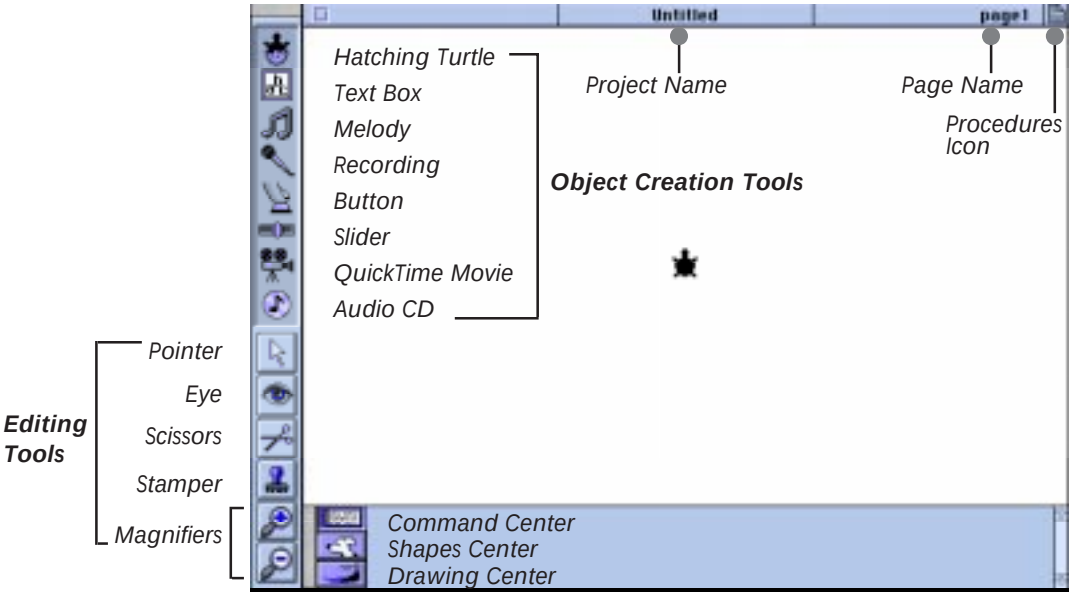
Section 4, Appendices lists the primitives by function, and lists every key combination.

Section 1
Basic Techniques

Starting Up and Saving

Starting With a New, Empty Project

To start up MicroWorlds, double-click on the MicroWorlds icon in the Finder. MicroWorlds opens to a new, empty project.



New projects are automatically named Untitled. The first page of a new project is called Page1. You can change your page names if you want, but you don't have to. You must name your project before you can save it.

The Centers

MicroWorlds has three “Centers”: the Command Center, the Drawing Center, and the Shapes Center.

In the Command Center, you type commands.

In the Shapes Center, you select the turtle’s shape, edit shapes, or create your own shapes.

The Drawing Center contains drawing tools for drawing graphics. You can also program each color.

The Drawing Center

Use the Drawing Center tools to draw on the background, set the turtle’s pen size and pen color, and program colors. Click on the Drawing icon to open the Drawing Center. To close the Drawing Center, click on one of the other Center icons.

The drawing tools are on the left, followed by the undo tool, the pen size selection, and the colors. There are 140 shades to choose from, plus black and white. Use the scroll bar to see lighter and darker shades of the 14 colors.

The Drawing Tools

Pencil

Draws in the selected color and pen size. To draw horizontal or vertical lines, hold down the **SHIFT** key as you draw.

Line tool

Draws a straight line in the selected color and pen size. To draw horizontal or vertical lines, hold down the **SHIFT** key as you draw.

Rectangle

Draws a rectangle in the selected color and pen size. To draw a square, hold down the **SHIFT** key as you draw.

Solid rectangle

Draws a solid rectangle in the selected color. To draw a square, hold down the **SHIFT** key as you draw.

Eraser

Erases graphics as you drag the eraser. Objects such as turtles, text boxes, and buttons are not part of the background graphics, so they are not erased. Double-clicking on the eraser clears all graphics on the page. Use the undo tool to restore anything you accidentally erase. To erase horizontal or vertical lines, hold down the **SHIFT** key as you erase.

Paint can

Fills an enclosed area with the selected color.

Spray can

Sprays speckles of the selected color and pen size.

Oval

Draws an oval in the selected color and pen size. To draw a circle, hold down the **SHIFT** key as you draw.

Solid oval

Draws a solid oval in the selected color. To draw a circle, hold down the **SHIFT** key as you draw.

Selection tool

Selects a rectangular area of graphics. Objects such as turtles, text boxes, and buttons are not part of the background graphics, so they are not selected. After your selection is made, you can:

- copy or cut the graphics
- drag one of the "handles" at the corners to expand or shrink the selection
- move the selection by clicking inside the selected area and dragging. Press the **OPTION** key while dragging to move a copy of the selection.
- double-click in the selection to open the Fat Bits window for precise editing. See Fat Bits Window further in this section.

Undo

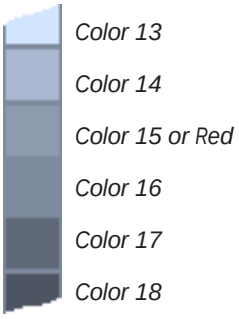
Undoes the last drawing tool's action. (The undo menu item, or **APPLE -Z**, has the same effect.)

Pen size

Determines the pen size for the empty rectangle, the empty oval, the pencil, the line tool, the turtle's pen, and the spray can.

Colors

Besides black and white, MicroWorlds has 14 colors, each with ten shades. Use the scroll bar at the right of the Drawing Center to see the lighter and the darker shades of the 14 colors.



Each color has a number. The colors are numbered in tens: for example, the shades of red are numbered from 10 to 19; 15 is the "mid-tone" red that you see when you open a new project. The blues are numbered from 100 to 109, with 105 the mid-tone blue. The mid-tone colors also have names: gray, red, orange, brown, etc.

Black is color number 9, and white is number 0.

To find the name or number of a color, choose ? from the Help menu and click on the color.

Choose ? from the Help menu. *Click on a color.* *Choose ? again to leave Help.*

See **setc** in Section 4, *MicroWorlds Vocabulary*.

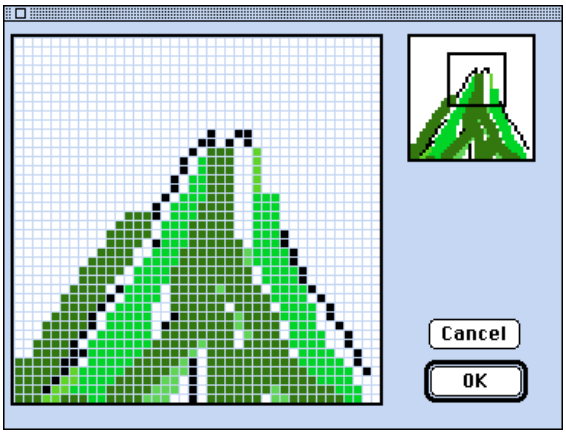
Fat Bits Window

You can do detailed editing of background graphics with the Fat Bits window. This window zooms into the graphic so you can see and change each screen dot, or pixel, one at a time. To enter the Fat Bits window, use the selection tool to select a region on the background, then double-click inside the region:

Select the selection tool. *Drag a region.* *Double-click inside.*

This opens a window in which you can draw one pixel at a time.

The right side of the Fat Bits window shows the graphic in its normal size. Drag the view selector (small square) around to view or edit the rest of your selection.



Click OK or Cancel to return to normal view.

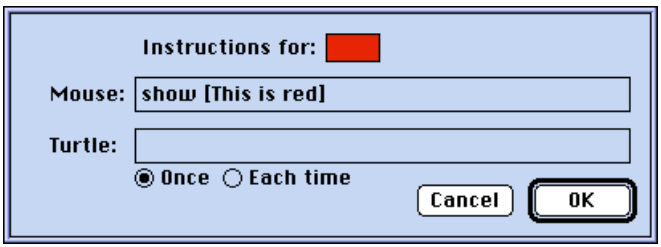
Programming Colors

You can program each color to react to mouse clicks or turtles. You can program the color red, for example, so that whenever the mouse clicks on anything drawn on the page in any shade of red, an instruction runs. Or, you can program the color so that whenever a turtle crosses over anything drawn in red, an instruction runs.

Mouse Detection

Double-click on any color in the Drawing Center to open its dialog box. (If the color is already selected, click once.) The color red is used in this example.

Type a command in the Mouse instruction and click OK.

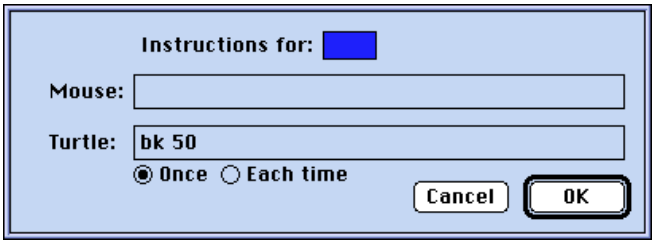


Draw a red circle on the page (any shade of red).

Open the Command Center and click on the red drawing. The words “This is red” will be displayed. You can program different colors with different instructions. Colors can move the turtle, play tunes, go to a different page, and so on.

Turtle Detection

Double-click on another color in the Drawing Center. For this example, use blue. Type the following in the Turtle instruction:



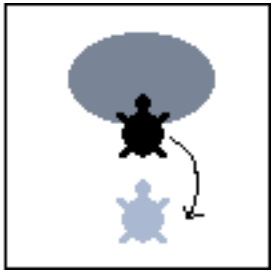
Now draw something blue, just ahead of the turtle. Open the Command Center and type:

```
repeat 50 [fd 1]
```

Let the turtle run over the blue area you drew. It should bounce back when the color blue “senses” the turtle.

Draw something blue in front of the turtle.

*Open the Command Center and type **repeat 50 [fd 1].***



See it bounce.

You can tell which colors have been programmed by looking at the Drawing Center. Programmed colors are identified by a dot.

Note: Color detection works for graphics drawn on the page, not objects. You cannot detect the color of turtles, buttons, sliders, icons, or text boxes.

The Shapes Center

In the Shapes Center you can select a shape for the turtle, or change or create your own shapes. Click on the Shapes icon to open the Shapes Center. To close the Shapes Center, click on one of the other Center icons.

The Shapes Center has one special shape — the turtle shape — and many other shapes for you to choose from. Some of the shapes are empty. You can change or completely replace any shape except the turtle.

Changing the Turtle's Shape

To change the turtle's shape, click on a shape in the Shapes Center, then click on the turtle. To see more shapes, use the scroll bar.

The original turtle shape is the only one that turns to show where the turtle is heading, and the only one that changes color to show the pen color. For this reason it's sometimes easier to use the turtle shape while you're preparing your program, then switch to the final shapes when your program is complete. To set the turtle back to the turtle shape, click on the turtle shape in the Shapes Center, then click on the turtle on the page.

See **setsh** in *Vocabulary* on the Help menu.

Shape Names and Numbers

Shapes have names and numbers. To find out a shape's name or number, choose **?** from the Help menu and click on the shape in the Shapes Center. The shape's name and number will be displayed. The name also appears in the Shape Editor.

Choose ? from the Help menu.

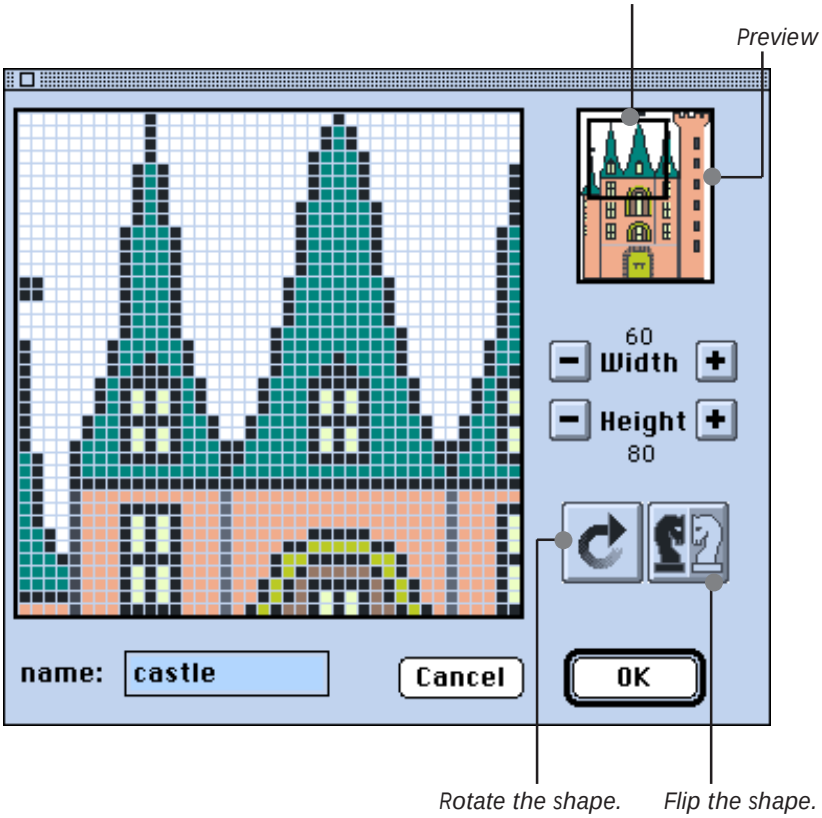
Click on a shape.

Choose ? again to exit Help.

Editing Shapes

To modify a shape or create your own, open the Shapes Center and double-click on the shape of your choice. (If the shape is already selected, click once.) The Shape Editor opens.

When a shape is larger than the editor (as shown below), drag the view selector (small square) to view or edit the rest of the shape.



Use the Drawing Center tools to change color or draw dots, lines, circles, or rectangles while the Shape Editor is open.

Double-click on the eraser to clear the whole shape. (If you do this by mistake, click on undo to get the shape back.)

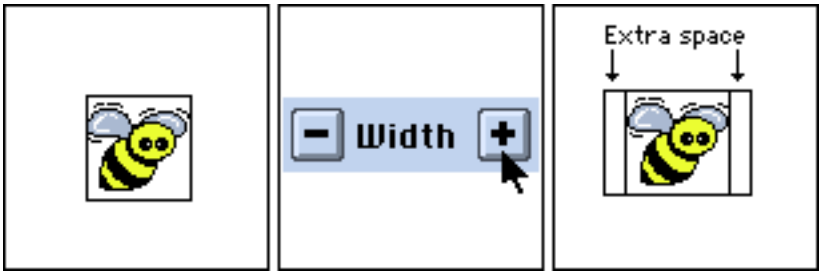
Note that the shape has a name — you can change it if you want. Use a one-word name, with no spaces. See **setsh** in *Vocabulary* on the Help menu.

To create a new shape, double-click on an empty shape. Empty shapes have blank names, so name your shape after you have finished.

Changing the Size of a Shape

The size of a shape is the number of dots (called pixels) it is made of. Blank shapes and most of the pre-defined shapes are 40 x 40 pixels, which is the smallest size for a shape. You can make shapes that are much larger, but larger shapes use up a lot of computer memory.

Use  and  to make a shape larger or smaller. The shapes grow or shrink by 5 pixels at a time. Watch the shape grow or shrink in the preview (in the top, right corner) as you go, and watch the pixel size change in the height and width display.



Original size, as seen in preview. *Make it wider.* *Five pixels added on each side.*

Note: Don't confuse the shape size with the turtle size. The shape size is the number of pixels used to draw the shape. The turtle size is how big or small it looks on the page. See *Changing the Turtle's Size* later in this section.

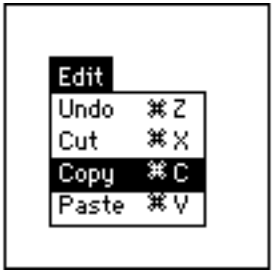
See **setsize** in *Vocabulary* on the Help menu.

Copying Shapes

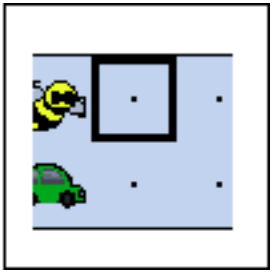
You may want to change one of the pre-defined shapes but keep a copy of the original. This is useful when making an animated shape (two shapes that are slight modifications of each other) or making two copies of the same shape, each with a different heading (like a cat facing left and one facing right).

To copy a shape:

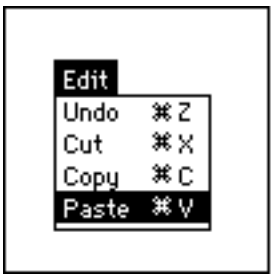
Select a shape.



Choose Copy from the Edit menu.



Select an empty shape.



Choose Paste from the Edit menu.

There's the copy. Double-click to edit the copy.

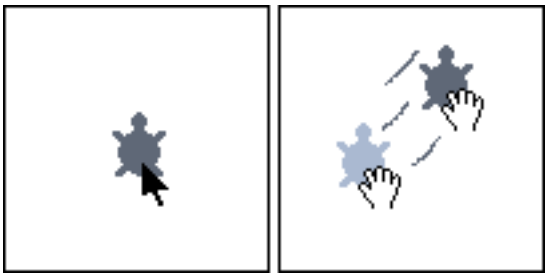
Note that the new shape doesn't have a name. Open the Shape Editor and give it a name that is different from any other shape's name.

Turtles

In MicroWorlds, turtles are everywhere. You use them to draw, decorate, act like push buttons, and play animated scenes. Turtles have many properties: each turtle has a name, a position, a heading, a pen size, a pen color, a shape, and even an instruction when you click on it.

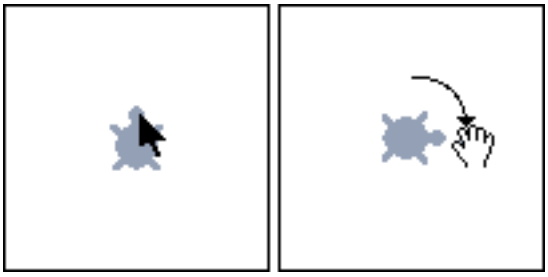
Moving and Turning the Turtle

You can move the turtle by dragging it with the mouse.



With the pointer, drag the turtle.

You can change the turtle's heading by dragging its head.



Point to its head. *Make it spin.*

You can only turn the turtle by dragging its head if the turtle has the original "turtle" shape.

When there are two turtles, you can bring one to the front by pressing the **OPTION** key while clicking on it. This feature is particularly useful when animating turtles in different shapes.

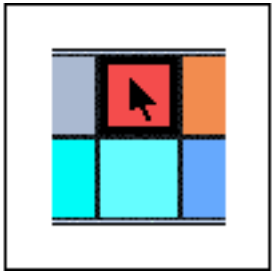
See **setpos**, **seth**, **fd**, **lt**, and **rt**, in *Vocabulary* on the Help menu.

Pen Color and Pen Size

When you hatch a new turtle, its pen is up. The **pd** command puts the turtle's pen down.

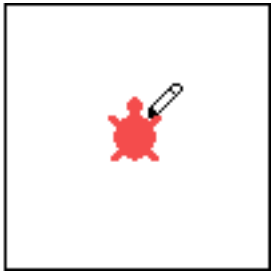
You can change a turtle's color or pen size using the Drawing Center.

Select the pencil.



Choose a color.

Select a pen size.



Click on a turtle.

Type this in the Command Center:

```
pd
fd 100
```

The turtle will draw a line in the color and pen size that you have selected.

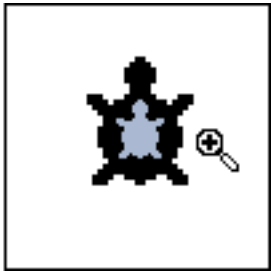
This only works if the turtle has the original "turtle" shape. If the turtle has another shape, either change its shape back to the turtle, or use commands to change the pen color and size.

See **setc** and **setpensize** in *Vocabulary* on the Help menu.

Changing the Turtle's Size

You can expand or shrink the turtle using the magnifiers in the Tool Palette.

Select the magnifier.



Click on the turtle.

See **setsize** in *Vocabulary* on the Help menu.

New Turtles and Turtle Names

You can use lots of turtles — just hatch them!

Select the hatching turtle and click anywhere on the page. A new turtle appears. Do this a second time and there will be two new turtles on the page.

The turtles are numbered as they appear: t1, t2, and so on. Choose **?** from the Help menu and click on a turtle to find its name.

Choose ? from the Help menu. *Click on a turtle.* *Choose ? again to exit Help.*

If you have several turtles and want them to do different things, you have to address them individually. When you talk to someone, you might say, “George, do this.” Notice the comma after the name George. In MicroWorlds you also use a comma after a turtle’s name. Hatch a second turtle and try the following commands in the Command Center:

`t1,`	Talk to the turtle t1. Don’t forget the comma.
`setsh "house`	Set its shape to a house.
`t2,`	Talk to the turtle t2.
`fd 100`	T2 moves forward.

When you have many turtles on a page, there is always one that is listening to your commands. That turtle is:

- the last turtle that you’ve created
- or the last turtle on which you’ve clicked
- or the last turtle that you’ve addressed using its name followed by a comma

See **newturtle** and **talkto** in *Vocabulary* on the Help menu.

Stamping a Turtle

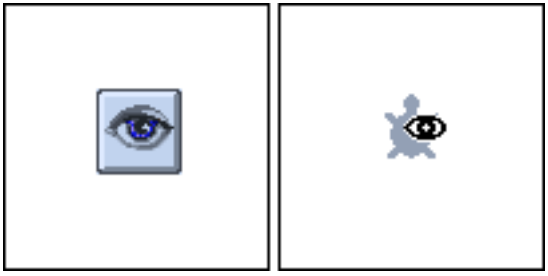
Stamping a turtle copies its image onto the background graphics. To stamp a turtle, select the stamper and click on a turtle. You should hear the stamper sound. You may think that nothing happened, but if you drag the turtle away, you'll see that the "live" turtle moves, and the graphic image stays behind.

See **stamp** in *Vocabulary* on the Help menu.

Clickable Turtles

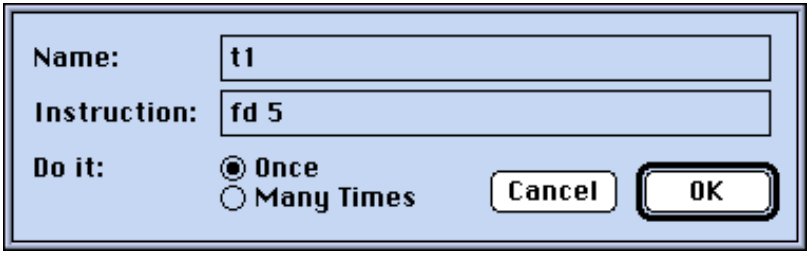
You can make a turtle do something special whenever you click on it. For example, a turtle can start moving when you click on it. These special instructions must be typed in the turtle's dialog box. Each turtle has its own dialog box.

To open a turtle's dialog box:

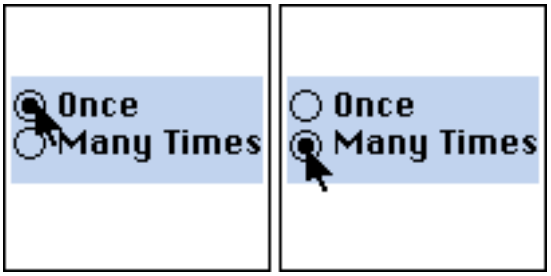


Select the eye tool. *Click on the turtle.*

Type this instruction:



Do you want the instruction to run once or should it run again and again?



Choose Once to run once.

Choose Many Times to run repeatedly.

Click on the turtle to see what happens. If you've set the instruction to Many Times it keeps going. Click on the turtle again to stop the action.

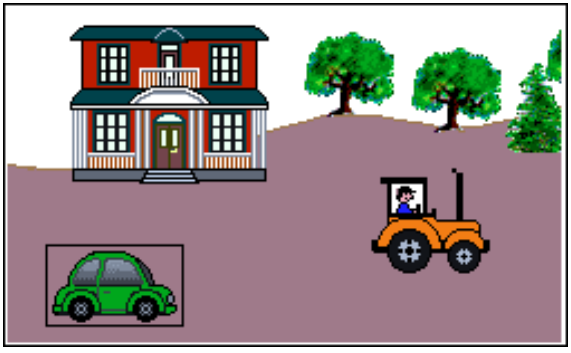
If you are having trouble "catching" the turtle, you can also choose Stopall from the Edit menu or press **APPLE - . (period)** to stop.

See **clickon** and **clickoff** in *Vocabulary* on the Help menu.

Finding Programmed Turtles

It can be hard to remember which turtles have been programmed to react to a mouse click, and which are just plain turtles.

To tell which turtles are programmed and which aren't, select the eye tool. The turtles programmed to react to a mouse click will show a frame around them (like the car below). See *Modifying Existing Objects*, later in this section.



Animation Techniques

There are three types of MicroWorlds animation:

- moving (such as when a car drives across the screen)
- changing shapes (such as when a flag waves)
- both moving and changing shapes (such as when a dog runs)

The easiest method, moving, is shown in *Clickable Turtles*, just above. The following describes the third type of animation, moving and changing shapes.

Using Setshape

Look in the Shapes Center. Some of the shapes come in sets of two or three. By switching shapes, you can make a bird or bee fly, a boy walk, or a dog run.

Here's how to make the bird fly. Hatch a new turtle or use a turtle already on the page.

- Point the turtle toward the top, left corner of the page.
- Open the turtle's dialog box (use the eye tool).
- As the instruction, type: **setsh "bird1 fd 2 setsh "bird2 fd 2.**
- Click on Many Times to run the instruction again and again.
- Click OK to close the dialog box.
- Click on the turtle, and see how it changes to a flying bird.
- Click on the bird-turtle again to make it stop.

You can use this technique for any of the sets of shapes. Check the names of the shapes in the Shapes Center. If there are three shapes in the set (as in the walking boy), add another **setsh** and **fd** command. If the instructions are too long, write them into a procedure on the Procedures page. See *The Procedures Page*.

Using Setshape With a List

You have seen that **setshape** is used with the name of a shape (as in **setshape "bird1**). **Setshape** can also be used with a list of shape names. In that case, the turtle will "cycle" through the list of shapes each time it runs a **fd**, **bk**, or **glide** command. Try these commands in the Command Center:

```
setsh [bird1 bird2]
repeat 20 [fd 2]
repeat 20 [fd 10]
glide 100 5
```

In other words, you can use a single **setsh** command to set things up (in a **setup** procedure, for example) and not worry about switching shapes anymore. The next **forward** or **back** command, issued from the Command Center, a button, or a clickable turtle, will take care of the animation.

To stop the shape switching, give the turtle a single shape: **setsh "bird1**.

This can also be done from the Shapes Center:

- Hold down the **SHIFT** key.
- Select a shape.
- Click on the turtle.
- With the **SHIFT** key down, select another shape.
- Click on the turtle.
- Do this for as many shapes as you want.

Then try commands like these:

```
repeat 20 [fd 10]
```

To stop shape switching, select one shape (don't hold down the **SHIFT** key this time) and click on the turtle.

Removing Turtles

If you have too many turtles on the page, select the scissors and click on the unwanted turtles.

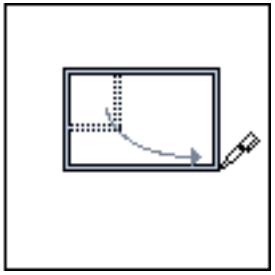
Here's an important tip: turtles are best when you want shapes that move or objects that you click on. They're also useful for parts of a picture that you might want to change later. But if you want to draw a forest, for example, it's better to stamp a turtle 25 times than to create 25 turtles. 25 turtles will use too much of the computer's memory, and limit other aspects of your project. See *Stamping a Turtle*, above.

Text Boxes

Text boxes are not part of the graphic image. You can move them around, change the text in them, and shrink or expand them. You can have many text boxes on the same page.

Creating a Text Box

Select the text box tool.

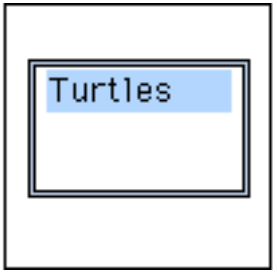


Drag to expand the text box.

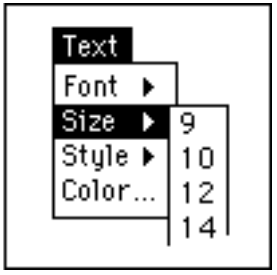
Typing and Formatting Text

To type text in a text box, click in the text box. When the text cursor flashes, type the text.

Text can be set to a different font or size. To change the size:



Select the text.

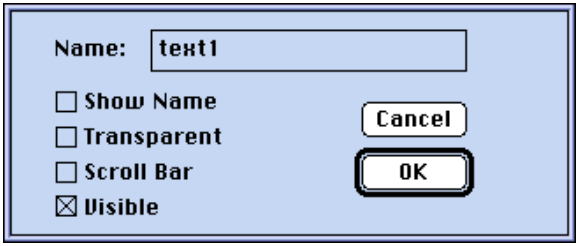


Choose a size from the Text menu.

You can change the text font, size, style, and color, as long as you keep the text selected and make the changes one at a time.

Transparent Text Box, Name Tag, and Scroll Bar

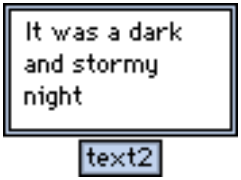
You can make a text box transparent so background graphics show through. To make the text box transparent, select the eye tool and click on the text box. Its dialog box appears:



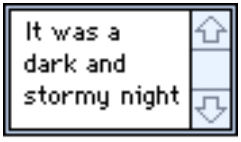
Check Transparent. Note that you can't type or change the text in a transparent text box. You can move it by dragging it, but to change anything else you have to open its dialog box and uncheck Transparent. See the **transparent** and **opaque** commands in *Vocabulary* on the Help menu.

Note: Transparent text boxes use a lot of space. Avoid using this feature on a text box that contains a lot of text.

Check Show Name to make the box's name tag visible.



Check Scroll Bar to add scroll bars to the box. Use scroll bars if the box will contain so much text that you'll need to scroll through it.



Uncheck Visible to make the text box invisible. You will need the eye tool or the **showtext** command to make it visible again. See *Eye Tool* in this section and **showtext** and **hidetext** in *Vocabulary* on the Help menu.

Stamping Text

In some situations (such as labeling a map) it's better to stamp the text rather than leaving text boxes all over the page. That way, you won't accidentally move a text box to a different location. Stamped text becomes part of the background graphics. Follow these steps to stamp text:

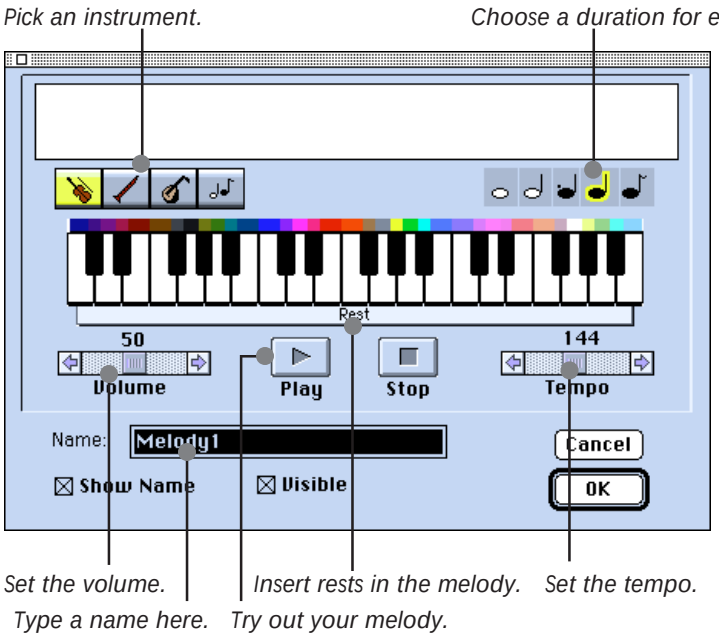
- Create a text box and type in the text you want.
- Use the items in the Text menu to format your text.
- Open the text box's dialog box and make it transparent.
- Select the stamper, and click on your transparent text box.

You won't see a difference because the transparent text box is sitting on top of the stamped text. Drag the text box away, and you'll see the transparent text box move, while the graphic text stays behind. If you don't need the text box, use the scissors and cut it.

See **stamptext** in *Vocabulary* on the Help menu and *Object Management* in this section for more information about editing, selecting, moving, copying, and deleting text boxes.

Melodies

To create a new melody, select the melody tool and click anywhere on the page. The Melody Editor opens.



Click on the piano keyboard to create a melody. The "score" of the melody appears at the top. You can select notes in the score, and delete them, copy them, cut them, or change their duration. You can click anywhere in the list of notes and add new notes by typing them in.

A melody can use only one instrument at a time, but different melodies can use different instruments. The maximum number of notes in a melody is 78.

Important: Use a one-word name, so you can use the name as a command.

When you leave the Melody Editor, you'll see a Melody icon on the page. Click on the icon to play the tune, click on it again to stop it or let it end by itself. Two melodies can play at the same time.

Creating a melody also creates a MicroWorlds command that plays the melody. For example, if you create a melody named "mytune" you can use **mytune** as a command in a button, a procedure, or in the Command Center to play the melody.

A melody can play while the turtles move, for example:

```
t1, forever [fd 1] mytune
```

See *Object Management* for more information about editing, selecting, moving, copying, and deleting melodies.

Recording

If there is a microphone connected to your computer, you can record your voice, then play it back with a command. You can record your voice directly into your project, or create a sound file.

Before recording, you have to decide whether your recording will stay inside your project or whether it should be saved as a separate sound file. Each method has an advantage and a disadvantage:

- Recording to project is simpler. You record your voice, name the recording, and click OK. The recording becomes part of your project and will always be there when your project is on the screen. The disadvantage is that recordings take a lot of workspace. This limits other aspects of your project, like the maximum number of pages you can have.

- Recording to disk adds an extra step to the process. You have to create a file to store the recording. The next time you open your project, you have to make sure that the sound file hasn't been moved or renamed. The advantages are that sound files don't take up your project's workspace, and sound files can be longer than recordings in the project.

To record your voice, click on the record tool and click on the page. The Record dialog box will appear:

Record to Project

- Select the record tool and click on the page. The Record dialog box opens.
- Leave the setting to "Record to Project."
- Click on the Record button to start recording. You can record for up to 20 seconds.
- Click Stop to stop recording.
- Click Play to listen to your recording.
- If you are not satisfied with the recording, click on the Record button again.
- Give a name to your recording.

Important: Use a one-word name, so you can use the name as a command.

No sound? If you can't hear your recording, check the Sound Control Panel (or Sound and Display) and check that the input sound device is set to Microphone, and not to Internal CD.

When you leave the Record dialog box, you'll see a Recording icon on the page. Click on the icon to play the recording, click on it again to stop it or let it play until it ends.

Creating a recording also creates a MicroWorlds command that plays the recording. For example, if you create a recording named MyVoice you can use **myvoice** as a command in a button, a procedure, or in the Command Center to play the sound.

Recordings can be played back from any page in a project. For example, if Page1 of a project contains the icon for a recording named Boo, the **boo** command plays this sound from any page in your project.

You can play a sound while moving the turtle, for example:

```
t1, forever [fd 1] myvoice
```

Record to File

To record to a file on the hard disk:

- Select the record tool and click on the page. The Record dialog box opens.
- Click on Record to File.
- The Save dialog box opens for this sound file. First find a location — it'll be easier to find the file later if you save it in the Sounds and Movies folder, in the MicroWorlds folder.
- Type a name and click OK to create the sound file (the file is currently empty but it will be "filled in" as you record).
- After the sound file is open, you'll be back at the Record dialog box. Click on the Record button to start recording. The duration of your recording is limited by the space available on your hard disk. Be careful: 10 seconds of sound file takes 40 K of disk space.
- Click Stop to stop recording.
- Click Play to listen to your recording.
- If you are not satisfied with the recording, click on the Record button again.
- Name the recording.

When you record to a file, the sound file is not loaded in your MicroWorlds project, the file remains on your hard disk. MicroWorlds has created two things:

- an icon that will remain permanently in your project
- a link to the actual sound file. MicroWorlds will remember where the sound file is on your hard disk, so the next time

you try to play the recording, MicroWorlds will know where to look for your sound file.

The next time you try to play this recording, MicroWorlds will look in three places to find the sound file:

- the Sound and Movies folder inside the MicroWorlds folder
- the folder where your project is saved
- the folder where the sound was originally created

If the file is not in any of these locations, the sound will not play.

If you move a MicroWorlds project to a different computer, and the project contains a link to a sound file, be sure to move the sound file as well.

Sound Quality

When you record a sound, the sound is recorded in "good" quality: the quality that takes the least amount of memory. To improve the sound quality, use the **set** command:

```
set "sounds "quality "better
```

sets the sound quality to a medium setting, taking up more memory.

```
set "sounds "quality "best
```

sets the sound quality to the highest setting. This setting only allows you 3 seconds of recording time.

```
set "sounds "quality "good
```

sets the sound quality back to the default setting.

Note that the setting is not saved with the project, so you will have to reset the sound quality each time you open a new project if you use a higher setting.

Deleting a Recording

Sound recordings that are kept in projects take up a large amount of workspace. To save space, delete any recordings you don't need. Since recordings are available from every page in a project, don't copy the recording's icon onto every page. If you duplicate a page, use the eye tool to see if the new page contains any recording icons, then use the scissors to delete them.

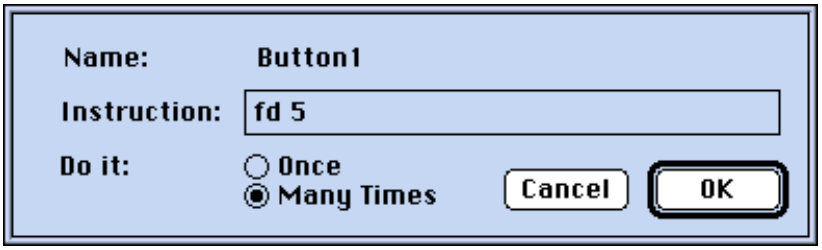
See *Object Management* for more information about selecting, moving, and copying recordings.

Buttons

Buttons run instructions when you click on them. A button can be set to run its instruction once (Once) or repeatedly (Many Times). While the instructions are running, you can do other things. For example, you can type some words in a text box, type a command in the Command Center, or click other buttons while one button's instructions are running.

Creating a Button

Select the button tool and click on the page. The button's dialog box opens. Type **fd 5** as the instruction and click Many Times:



Click on the button to try it. If it's set to Many Times, click on the button again to stop. You can also choose Stopall from the Edit menu or press **APPLE - . (period)** to stop.

If a button's instruction is very long, put the instruction in a procedure, and use the name of the procedure instead. For example, this button and this procedure go together (see *The Procedures Page* later in this section):

```
to wave
setsh "usa1
wait 3
setsh "usa2
wait 3
end
```

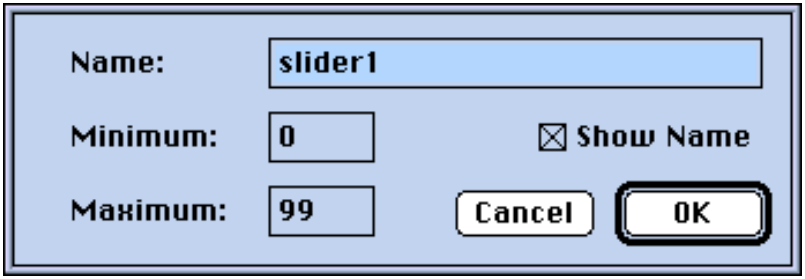
See *Object Management* for information about editing, selecting, moving, copying, resizing, and deleting buttons.

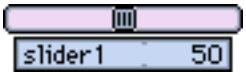
Sliders

A slider reports a number. The number, in turn, is used as the input to a command. For example, you can use a slider to change the speed of a jeep moving across the screen.

Creating a Slider

Select the slider tool and click on the page. The slider's dialog box opens:





The slider's name reports the slider's current value. For example, if you create a slider with the default values shown above, you can type these commands:

```
show slider1
50                  This number depends on the slider's value.
fd slider1          The slider's value is used as the input for fd.
setc slider1        The slider's value is used to set the color.
```

Important: Use a one-word name, so you can use the slider's name in an instruction.

See *Sliders as Variables* in Section 2.

See *Object Management* for information about editing, selecting, moving, copying, and deleting sliders.

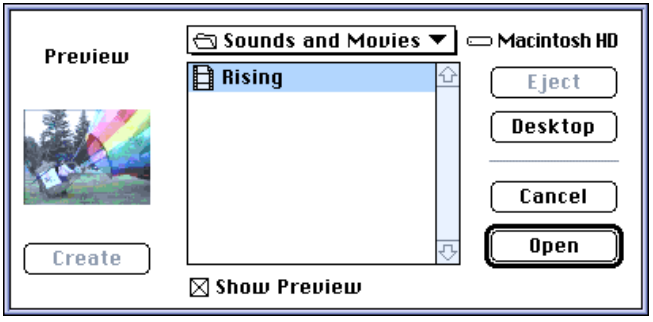
Movies

MicroWorlds projects can include movie clips (QuickTime files). The movies are not saved in your project, they are stored in separate files on your hard disk. Your movie will only play if MicroWorlds can find the QuickTime file on your hard disk.

Importing a Movie

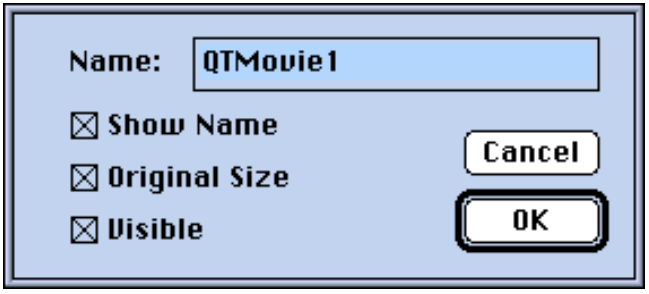
To import a movie, select the QuickTime Movie tool and click on the page. Alternately, you can choose Import Movie/Sound from the File menu.

The Import Movie/Sound dialog box opens. Locate and open the folder where the movie files are stored.



Only movies and sound files in the QuickTime format are displayed. Choose a movie or a sound file and click Open.

A second dialog box opens. This is used to name the movie, make its name tag visible or not, lock the movie at its original size, and make the movie visible or not (if a movie is made invisible, it will only show when you actually play it). The name of the movie doesn't have to match the name of the QuickTime file.



Important: Use a one-word name, so you can use the name as a command.

Click OK and the movie appears on your page.

Click on the movie to start it, click on it again to stop it. Clicking on it again will make the movie continue from where it stopped.

You can also use the movie name as a command in the Command Center, in a button, a procedure, and so on.

When you import a movie into a project, MicroWorlds creates two things:

- a "poster" that remains on the page. The poster is the first image of the movie.
- a link to a QuickTime file on disk. MicroWorlds will remember where the QuickTime file is on your hard disk, so the next time you try to play the movie, MicroWorlds will know where to look.

The next time you try to play this movie, MicroWorlds will look in two places to find the QuickTime file:

- the folder where your project is
- the folder where the movie was when it was originally imported

If the file is not in any of these locations, the movie will not play.

If you bring a MicroWorlds project to a different computer and the project contains a link to a movie, be sure to bring the QuickTime file as well.

Changing a Movie's Size

Movies look best at their original size. Also, they take more space and run more slowly when expanded. If you choose to change the size of the movie, you may have to "unlock" its size.

- Select the eye tool and click on the movie poster.
- In the movie's dialog box, uncheck Original Size.
- Then select the movie and drag one of its handles to shrink or expand it.

If the movie has been stretched to a larger size, and the picture appears grainy, check Original Size in the movie's dialog box. Use the **resetquicktime** command to reset a movie or quicktime sound back to the beginning of its track.

Stamping an Image From a Movie

Stamping a frame from a movie is a bit different from stamping other objects. Select the stamper and click on the movie. The movie can be playing or not. If you want to stamp a particular frame, stop the movie at the right moment, then stamp it. To see the stamped image, drag the Movie icon away. What is left behind is a graphic image that you can edit with the drawing tools.

If, after stamping an image, you don't need the movie anymore, use the scissors to delete it.

See *Object Management* for information about selecting, moving, copying, resizing, and deleting movies.

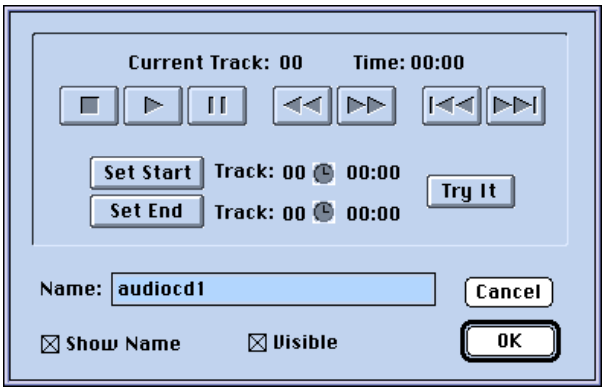
Audio CD's

You can add audio CD tracks, or sections of tracks, to your projects. Creating an audio CD clip doesn't load the music into your project. It creates a MicroWorlds object that plays a portion of the audio CD, for example, from 10:00 seconds to 20:00 seconds from track #4.

Creating an Audio CD Clip

To create an audio CD clip, first put an audio CD in the CD ROM drive. Don't use a regular CD ROM disk unless you are certain it contains audio tracks. Then, select the audio CD tool and click on the page. This opens the audio CD dialog box.

The top portion of the dialog box is like the control panel of a CD player. The controls are: Stop, Play, Pause; Rewind and Fast Forward; Skip to Previous Track, Skip to Next Track. Browse the CD until you find the sound you want.



- When you find the beginning of the sound clip you want, click on Set Start. You can click on Pause to hold the spot, and then click on Set Start, if you find that's easier.

- Click on Play, listen for the end of the sound clip, then click on Set End.
- Click on Stop to stop the CD, then click on Try It to hear your sound clip. Click OK if it is fine, otherwise redo the steps above.

When you've clicked OK, a Sound Clip icon appears on the page. Click on it to start the sound clip, click again to stop it or let it end by itself.

No sound? If you can't hear any sound, check the Sound Control Panel (or Sound and Display) and check that the input sound device is set to Internal CD, and not to Microphone.

Most of the time, when music is used in a multimedia presentation, it plays while something else happens. In cases like that, the audio CD clip will be launched by a command, not by clicking the icon. If you don't want to see the Sound Clip icon, make it invisible.

Important: Be sure to use a one-word name for your audio CD clip so you can use the name as a command. For example, if you create a sound named MyClip you can use **myclip** in a button, a procedure, or in the Command Center to play the sound.

You can play an audio CD clip while turtles move, for example:

```
t1, forever [fd 1] myclip
```

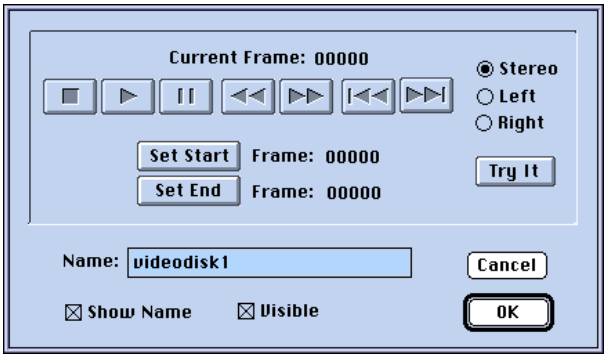
See *Object Management* for information about editing, selecting, moving, copying, and deleting an audio CD clip.

Videodisks

First make sure that you have a videodisk player connected to the computer's modem port, the videodisk player is turned on, and there is a videodisk in the unit.

To create a videodisk movie clip, select New Videodisk... from the Gadgets menu. This opens the videodisk dialog box.

The top portion of the dialog box is like a videodisk player's control panel. The controls are: Stop, Play, Pause; Rewind and Fast Forward; Skip to Previous Track, Skip to Next Track. Browse the videodisk until you find the video clip you want.



- When you find the beginning of the section you want, click on Set Start. You can click on Pause to hold the spot, and then click on Set Start, if you find that's easier.

- Click on Play, watch for the end of the video clip, then click on Set End.
- Click on Stop to stop the videodisk, then click on Try It to view the clip. Click OK if it is fine, otherwise redo the steps above.

When you've clicked OK a Video Clip icon appears on the page. Click on it to start the video clip, click again to stop it or let it end by itself.

Important: Be sure to use a one-word name for your videodisk clip so you can use the name as a command. For example, if you create a video named MyVideo you can use **myvideo** as a command in a button, a procedure, or in the Command Center to play the video clip.

See *Object Management* for information about editing, selecting, moving, copying, and deleting video clips.

Object Management

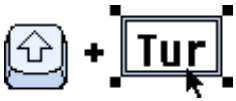
This section describes the Editing tools (the lower half of the Tool Palette).

Note: Frozen objects cannot be deleted, stamped, moved, resized, etc. See **freeze** and **unfreeze** in *Vocabulary* on the Help menu.

Selecting Objects

There are three ways to select an object: **SHIFT**-clicking on the object, dragging around the object, and using Select All from the Edit menu. In all cases, use the pointer.

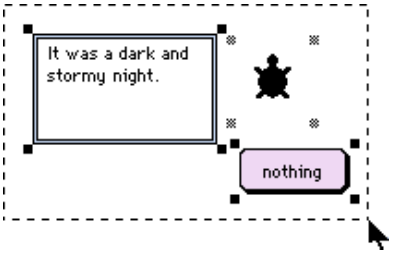
Selecting by SHIFT-Clicking



Hold down the **SHIFT** key and click on any text box, turtle, button, slider, melody or movie icon on the page. While the **SHIFT** key is down, you can continue to click on objects. Everything you click on will be added to the selection. Objects that are selected have "handles" at their corners.

Selecting by Dragging

Using the pointer, drag a rectangle around one or more objects. When you release the mouse button, you will see “handles” around the selected objects.



Selecting All Objects (or All Text)

To select all objects at once, click on the page and choose Select All from the Edit menu. The keyboard shortcut is **APPLE -A**. If you click inside a text box, in the Command Center, or on the Procedures page, **APPLE -A** will select all the text.

Moving Objects

To move objects, use the pointer. Click on the object and, while holding the mouse button down, drag the object.

The only exception to this rule is the text box, which is dragged from its frame.

Alternately, you can select the text box first, and drag it from any point inside the box.

Copying and Pasting

Copying objects is a quick way of making many identical objects. If you need many identical turtles, for example, first create one and set its characteristics (size, color, instruction, shape), then copy it. You can copy an object, then change pages, then paste the object. It will be pasted, along with all its characteristics, onto the new page.

Objects

Follow these steps to copy an object:

- Select the object using the pointer (see technique above).
- Choose Copy from the Edit menu.
- Choose Paste immediately to get a copy on the same page, or
- Go to another page and choose Paste.

Graphics

The pointer doesn't select graphics. Use the selection tool in the Drawing Center instead. Be sure to click once on the page before pasting graphics. If a text area is active (a text box or the Command Center), graphics will not be pasted.

Copying Text and Copying a Text Box

You can copy any text into or from text boxes, the Command Center, and the Procedures page.

To copy the text from a text box, select the text (not the text box, the text *in* the box). Then choose Copy from the Edit menu.

To copy a text box, select it, then choose Copy from the Edit menu. See *Selecting Objects* and *Copying and Pasting Objects* above.

Selected text. *Selected text box.*

About Pasting

Before pasting, click in the area where the pasted object will go. For example, buttons, graphics, or text boxes can't be pasted into the Command Center or on the Procedures page. Click on the page before trying to paste these objects.

Similarly, text can only be pasted where text normally appears. Before pasting text, click in the Command Center, on the Procedures page, or inside a text box.

Unique Names for Copied Objects

Each object on a page has a unique name. When you paste a copy of an object, MicroWorlds assigns it a name of its own. For example, if you copy a slider named Slider1 and paste it on the same page, the copy will be named Slider2. If you paste it on a page with no sliders, the copy will have the same name as the original.

Recordings are the only exception to this rule. The name of a recording must be unique across a whole project.

In general, after copying and pasting an object, you should open its dialog box and check the name. If the name that's automatically provided doesn't suit your project, give the object a new name that you prefer.

Hint: If you want to copy most of the objects from one page to a new page, it's simpler to duplicate the page and then delete the objects you don't need. See *Duplicate Page* in *Menus* below.

Modifying Existing Objects

The eye tool has two functions. It can be used to show you how many objects you have created, and it can be used to open the dialog box of any object on a page.

When you select the eye tool, all the objects on the page become temporarily visible, and turtles that are programmed are framed. Use this technique to make an inventory.

In order to delete or change an object, select the eye tool and click on any object.

To modify an invisible object, you must first make it visible. Clicking on the eye tool is not enough, it only makes them visible temporarily. While the eye tool has made an object visible, click on it to open its dialog box. If you want to delete the object, check Visible, then click OK. Now the scissors will delete the object.

Stamping Objects

The stamper can be used to stamp the turtle's shape, the text in transparent text boxes, and images from QuickTime movies. You can save on memory by stamping turtle shapes, text, and movie images rather than leaving them as objects. For example:

- To draw a forest, stamp a tree shape 25 times, instead of using 25 turtles with the tree shape. Only use real turtles if you want to program them, or to animate them, or if you might change your mind about their location or shape.
- To draw a map with many text labels, use only one text box to stamp the labels, instead of creating many text boxes. Remember to delete the text box after you stamp or you won't be saving as much memory as you think.
- If you need only one frame from a movie, stamp it, then delete the movie.

About Real Turtles and Stamped Turtles

It can be hard to remember which turtles on the page are "live" turtles and which ones are stamped images. To find out, select all the objects on the page. The ones with the "handles" around them are real turtles (like the two cars and one of the trees below). The ones without handles are stamped images (the house and the rest of the trees below).

To select everything, click on the page and choose Select All from the Edit menu (**APPLE -A**).

Deleting Objects

There are many ways to delete the objects you have created. The easiest method is to select the scissors and click on the unwanted objects.

If you can't seem to be able to delete a turtle, it might be a stamped image. The scissors can't delete graphics. Frozen objects also can't be deleted. See **freeze** in *Vocabulary* on the Help menu.

Deleting Invisible Objects

Some objects, like recorded sounds and QuickTime movies, take a lot of computer memory. If you are working on a project over many days, you may have forgotten that you have such objects. So if you get a message that says MicroWorlds is running out of memory, go to each page and click on the eye tool to see any invisible objects.

To delete an invisible object, you must first make it visible. Clicking on the eye tool is not enough, it only makes them visible temporarily. While the eye tool has made an object visible, click on it to open its dialog box. Check Visible, then click OK. Now the scissors will delete the object.

Making Objects Larger and Smaller

Some objects can be made larger and smaller using the magnifiers, some by dragging the handles, and some with MicroWorlds commands.

The magnifiers change the size of turtles, buttons, and text boxes.

Selecting and dragging changes the size of buttons, text boxes, and movies (uncheck Original Size first, in the movie icon's dialog box).

There are commands to change the size of turtles (**setsize**), text boxes (**set**), text (**setfontsize**), buttons (**set**), and movies (**set**). See *Vocabulary* on the Help menu.

The Procedures Page

The Procedures page is where you write and edit procedures (programs). Each project has its own Procedures page. Projects which are provided for you have procedures that are already defined.

Writing and Editing Procedures

To open the Procedures page:

*Click here to open and close the Procedures page, or choose Procedures from the Pages menu, or press APPLE - **F**.*

A procedure is a list of instructions with a name. Once you've created a procedure, you just type its name to run all the instructions it contains.

A procedure has three parts:

`to go`	the title line (with the name of the procedure)
`t1, fd 20`	instructions
`t2, fd 10`	more instructions
`end`	the end line

All procedures start with **to** and the name of the procedure. You choose the name. Always use one word for the name.

Things to Note About Procedures

The title line of a procedure (**to**, the name, and its inputs if any) must be on a line by itself. You must press **RETURN** at the end of the line. See *Variables*, in Section 2, for an example of a procedure with an input.

Lines of instructions within the procedure are also separated by **RETURN**.

The last line of a procedure is the word **end** on a line by itself.

Always press **RETURN** after the word **end**.

To Test Out a Procedure

Go back to your page and type the procedure's name (for example, **go**) in the Command Center followed by **RETURN**.

or

Create a button with the name of the procedure.

A Shortcut to Making Small Changes

To save typing, you can copy the text of a procedure, paste it, then edit the copy. For example, the **wave** procedure can be copied and edited to define a new procedure named **fly**.

Say the **wave** procedure is already written:

```
to wave
repeat 20 [setsh "usa1
     wait 3
     setsh "usa2
     wait 3]
end
```

- Select the **wave** procedure by dragging over each line.
- Choose Copy from the Edit menu.
- Place the cursor at the end of the Procedures page, and choose Paste from the Edit menu.
- Edit the copied procedure:

```
to fly
repeat 20 [setsh "bird1
     wait 3
     setsh "bird2
     wait 3]
end
```

Printing

Before printing, make sure the printer is connected, turned on, and selected in the Chooser.

Printing a Page

To print the contents of the current page, choose Print Page from the File menu. The page will be printed just the way it looks on the screen — except, of course, that if your printer is black and white, you can't print the colors!

If a text box contains more text than is visible, only the text you see on the screen will be printed.

If the Procedures page is on screen, the text will be printed in "draft quality."

Printing All the Pages in Your Project

Choose Print Project from the File menu. Every page in the project will be printed, in the order shown in the Pages menu. The Procedures page will not be printed.

Printing the Contents of a Text Box

To print the entire contents of a text box, including text that may not be visible, click in the text box to make it current and type **printtext** in the Command Center. If you have several text boxes on the pages, you can make sure you print the correct one by typing its name followed by a comma:

```
text2,
printtext
```

The text will print, using the full width of the page, in draft quality.

Menus

Some of these features are described in more detail elsewhere in this section.

File

New Project

Opens a new project. If there is an open project already on the screen, you'll be asked if you want to save it first.

Open Project

Opens a dialog box where you choose which project to open. If there is an open project already on the screen, you'll be asked if you want to save it first.

Close Project

Closes the current project, first asking if you want to save it.

Import

Import offers four choices:

- Import Movie/Sound imports QuickTime movies or QuickTime sound files.
- Import Text imports the contents of a text file into the current text box, the Command Center, or the Procedures page, whichever contains the flashing cursor.
- Import Picture imports a PICT file and places it on the page's background graphics.
- Import Project allows you to add procedures, pages, or shapes from a different project.

See *Importing and Exporting* in Section 3, *Advanced Techniques.*

Save Project

Saves any changes you've made.

Save Project As

Opens a dialog box where you choose a new location and/or name for your project.

Page Setup

Opens a dialog box where you choose the paper size and other printer details before you print.

Print Page

Opens a dialog box where you choose to print your page or pages. They will print exactly as they appear on the screen. If the Procedures page is showing, all the text will be printed. Choose Page Setup before printing.

Print Project

Opens a dialog box where you choose to print all the pages in your project, exactly as they appear on the screen. Choose Page Setup before printing.

See *Printing*, in this section.

Quit

Quits MicroWorlds and returns to the Finder.

Edit

Undo

Undoes the last text editing operation, the last Drawing action, or the last deletion of page elements (buttons, for example). Programmed actions can't be undone.

Cut

Cuts the selected text, graphics, turtle shape, or page element (buttons, for example) and puts it in the Clipboard.

Copy

Copies the selected text, graphics, turtle shape, or page element and puts it in the Clipboard.

Paste

Pastes the contents of the Clipboard where the cursor is currently flashing. Paste only works if the contents of the Clipboard are of the right type for the current location (for example, you can't paste graphics if the cursor is currently flashing in the Command Center).

Select All

Selects all the objects on the page, including those that are not visible. If the cursor is inside a text box, the Command Center or the Procedures page, all the text is selected.

Clear

Clears whatever is selected. If the cursor is in a text box, the selected text is cleared. If objects (buttons, sliders, turtles, etc.) are selected, they are cleared. If a shape in the Shapes Center is selected, it's cleared.

Find/Change

Opens the Find/Change dialog box, where you can search for, and change, a word or a number of words in the current location (text box, Command Center, or Procedures page).

Cancel

Selectively stops some processes while leaving others running. Choosing Cancel opens a menu where you choose which process to cancel. Processes can be launched using **launch**, **forever**, and **when**. See **cancel** in *Vocabulary* on the Help menu. Some processes, like clickable turtles, don't show in the Cancel list.

Stopall

Stops all running processes, including repeating processes launched by buttons and clicked turtles. If your programs seem to suddenly run more slowly than usual, you may have left processes running on other pages. Choose Stopall to stop everything and then restart the processes you really need.

Text

You can choose the font, style, color, etc. for any text in text boxes, the Command Center, or the Procedures page.

See *Text Boxes* in this section.

Font, Size, Style

Opens a menu showing the text fonts, sizes, and styles you can choose from.

Color

Opens a dialog box where you can choose a text color.

Pages

New Page

Opens a new page, which becomes part of the project. All the pages in a project are listed in the Pages menu.

To remove a page, first check the Pages menu for the list of page names. Then go to the Command Center and type:

`remove "page1` Use your page name. Press **RETURN**.

Caution: Deleting a page is permanent. Once you remove a page, you can only recover it if the project was previously saved. If you've accidentally deleted a page from a saved project, choose Open Project from the File menu. Click Don't Save when the Save Changes Before Closing box comes up. Then reopen the saved version of the project.

Name Page

Opens a dialog box where you name the current page. Pages are given names by default (Page1, Page2, and so on) but you can choose your own names. Be sure to use a single word as a page name because page names can be used as commands to go from page to page.

For example, you can create a multi-page project with four pages, called Contents, Intro, Demo, and Exit. You can have buttons or clickable turtles on the Contents page to jump to the page of your choice. Simply use a page name as the instruction of a button or a clickable turtle.

Duplicate Page

Creates a new page which is identical to the current page. All the current page's graphics and objects are copied.

In a multi-page project, it may happen that a second page has a lot in common with the first one you did (the background and the turtles for example). In that case, you may simply duplicate the first page and make some changes on the second page.

When you choose Duplicate Page from the Pages menu, a new identical page is created. Check the page name on the title bar of the page, or the list of pages in the Pages menu, to see that a new page has been added to your project.

Recordings and melodies don't have to be duplicated. If you duplicate a page that has a recording or a melody, delete it from the second page to save some memory.

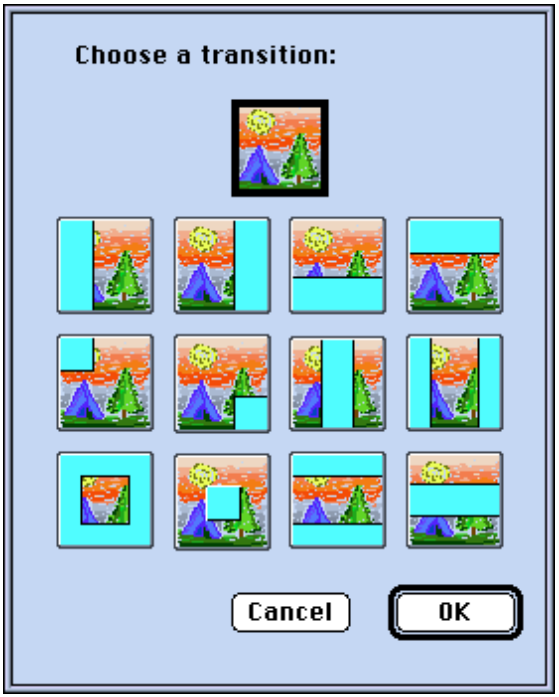
Procedures

Opens the Procedures page.

Transitions

Sets a visual effect that happens when the current page opens.

A visual effect can smooth the transition when you change pages. The transition you choose will affect only the page that is open. For each page you can choose the kind of visual effect that works best for the text or graphics on that page.



Select the transition of your choice and click OK. To test the transition, go to a different page (go to the Procedures page if you don't have another page) and come back. The transition effect will only be visible if you have graphics or objects on your page.

To remove the transition effects, pick the top selection.

Page1

All pages in the current project are listed at the end of the Pages menu. Selecting a page displays it. If a project contains a page named Page1, it will open with Page1 showing.

Gadgets

Tool Palette

When this is checked (√), the Tool Palette will be displayed.

Tool Sounds

When this is checked (√), some Drawing Center and Shapes Center tools will make sounds. Uncheck this selection for quieter operation.

Command Center

When checked (√), the Command Center will be displayed.

Presentation Mode

Presentation Mode is for demonstrating completed projects. In Presentation Mode:

- the Command Center and the Tool Palette are hidden
- the project's title bar and MicroWorlds menus are hidden
- the project is centered on the screen and the background is filled in

Before using Presentation Mode, make sure that your program is completely tested and working as it should. You won't have a Command Center, so you won't be able to type commands, see error messages, or use **show** to display text. Your program will have to start with a button, turtle click, or some other object.

To get out of the Presentation Mode, glide the mouse up to the area where the menu bar is supposed to be and press the mouse button.

New Videodisk

Opens the Videodisk dialog box so you can create a Videodisk video clip.

Help

Vocabulary

Displays a window that lists every primitive in MicroWorlds. Choose a command or a reporter name and click on Help to see its definition.

Last Message

Displays a window with a description of the last message that was printed in the Command Center, and some solutions for preventing the situation that caused the message.

?

Changes the mouse pointer to a question mark. Click the **?** on tool icons, objects on the page (turtles, buttons, sliders, text boxes), shapes, and colors to get Help information on those icons or objects. Clicking again or moving the mouse pointer makes the Help information disappear. Choose **?** again to exit Help. See also *Creating Your Own Help Balloons*, in Section 3.

Section 2
Logo Programming

MicroWorlds is based on the Logo language. Logo is a computer language, but like a spoken language it has a vocabulary, and rules for using the vocabulary. Where a spoken language has sentences, Logo has instructions. Where a spoken language's rules for how to put words into sentences can be complicated, Logo's rules for building instructions are much simpler.

In MicroWorlds, every word you type into the Command Center is interpreted as a request to do something. Some words are built in — **fd** and **show** for example. Words that are built into MicroWorlds are called "primitives." You can't edit or remove primitives.

When you write procedures, you're adding new words to the MicroWorlds vocabulary. Names like **start** and **go** could be the names of procedures. Procedures can be edited or erased, and they only stay in memory while the project you made them in is open, but apart from that they work like MicroWorlds primitives.

Objects like buttons, sliders, turtles, melodies, and movies have names. These names are also words in the MicroWorlds vocabulary. MySong and WinnerTune could be the names of melodies. Text1 and Speed could be the names of a text box and a slider. These words can all be typed in the Command Center and used in instructions.

Normally, you use no punctuation in MicroWorlds instructions. There are times when you use punctuation characters, usually to tell MicroWorlds that a word should *not* be treated as an instruction, but instead as something else.

Words and Lists

A word beginning with a quotation mark is treated literally, *not* as a procedure name. For example:

```
show "friend
friend          Displays the word friend in the Command Center.
setsh "bird1    Sets the turtle to the shape named bird1.
```

In the last example, if you omit the quotation mark (e.g., **setsh bird1**) MicroWorlds will try to *run* **bird1**. If **bird1** is not the name of a procedure, a melody, a slider, a text box, or a page name, MicroWorlds will display an error message:

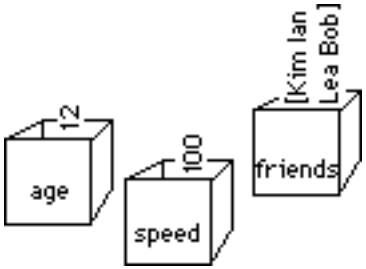
```
I don't know how to bird1
```

A list is a group of words. You can even have a list of lists. A list is always enclosed in square brackets. For example:

```
show [Kim Ian Lea Bob]
Kim Ian Lea Bob  Displays these words in the Command Center.
question [How are you today?]  Displays these words in
                                the question dialog box.
```

In general, spaces are used to separate the components of a list. A special case is a kind of word called a "long word," which can contain spaces. See *Object Names as Commands*.

Variables



You can think of a variable as a container with a name on the outside and a value (a word or a list) inside. When you create a variable, you create the container and at the same time you put the value inside. There are three types of variables: local, project, and global variables.

Local Variables

You create a local variable when you place an input on the title line of a procedure. A local variable holds its value only at the time MicroWorlds runs the procedure.

Here is a simple example:

```
to square :size
repeat 4 [fd :size rt 90]
end
```

The word size can be any word, but the same word must be used throughout the procedure. A colon must precede each occurrence of the word. A colon means "I don't want the *word* size, I want the value inside the *container* named size."

If MicroWorlds displays this message:

```
has no value in square
```

the word that seems to be missing at the beginning of the error message is in fact a space. The cause of the problem is a space left between the colon and the word used as a variable name in your procedure (e.g., **fd : size)**. Remove the space and try the procedure again.

Local variables can also be created with the primitives **let** and **local**.

Project Variables

Project variables are related to a specific project. They are cleared when you go to a different project, but they are saved with your project. The next time you open your project, the variables will have retained their value.

You create project variables with the command **createprojectvar**:

```
createprojectvar "points
```

This adds two words to MicroWorlds vocabulary: **points** and **setpoints**.

From now on, in this project, you can type instructions like:

`setpoints 10`	Set the value of points to 10.
`show points`	Show the current value of points.
`10`	
`setpoints points + 5`	Add 5 to the value of points.
`show points`	Show the current value.
`15`	

To find the names of all project variables in a project, type:

`show projectvars`	
`points`	The names of the project variables in the current project.

To remove a project variable, type:

`remove "points`	Use the name of your project variable.

Global Variables

You create a global variable with the commands **make** or **name**. A global variable holds its value even when you change projects. Global variables only lose their value when you deliberately erase them, or when you quit MicroWorlds. Global variables are not saved with a project. The next time you open your project, they will have to be defined again. A good way to do this is to put the **make** commands into a procedure.

```
make "friends [Kim Ian Lea Bob]
```

When you need the value of a variable, place a colon in front of its name. Note the difference between the quote and the colon:

`show "friends`	Show the word friends.
`friends`	
`show :friends`	Show the value of the variable friends.
`Kim Ian Lea Bob`	

Global variables are useful when the value of a variable must be kept in the computer's memory even when going from project to project (such as in a game where you want to keep score as you go from one project to another).

Text Boxes as Variables

A text box can be used as a container, just like a variable. The difference is that you can see the text box and you can set its contents just by typing in the text or by using special commands.

For example, create a text box, name it Friends, and type in some names (see *Text Boxes*, in Section 1). The word **friends** can then be used to report the contents of the text box.


```
show friends
Kim Lea Ian Bob
```

You can use the **cleartext**, **print**, and **insert** commands to change the contents of the text box. In addition, the word **set** followed by the name of the text box (**setfriends** in this example) can be used to replace its contents. Note that **insert** and **print** add text to the text box without removing any text, but set-text-box-name (**setfriends**) completely replaces the current text.

```
setfriends "Lucy
setfriends "Sam
```

There is one more difference between using **make** to create a variable and using a text box as described above. The variable created by **make** above is a list with four words in it. The name of the text box (**friends**) reports a long word containing 15 characters (4 names of three letters each, plus three spaces). See *Long Words* further down in this section.

In general, spaces separate words. The list **[Kim Ian Lea Bob]**, for example, contains four words. Sometimes, MicroWorlds recognizes a series of words as a long word containing one or many spaces. The difference between words and long words affects how you name objects and how you manipulate the contents of text boxes.

Contents of Text Boxes

The name of a text box reports its contents.

In this example, the text box is named Text1. The word Text1 is a reporter that reports all the characters in the text box, including the spaces, as one long word.

In computer jargon, long words are *character strings*. In fact, every word is a character string but the expression is used here to emphasize that what seems to be a set of individual words is, in fact, a set of individual characters, some of which are spaces.

```
show Text1
It was a dark and stormy night...
```

If you count everything, there are 33 characters in the text box:

```
show count text1
33
```

Since this is one long word, the first element of a word is a character:

```
show first text1
I
```

If you want to do something with the contents of a text box under program control, it is usually more convenient to report the text as a list of words, rather than as a long word containing characters. The **parse** reporter turns a long word containing spaces into a list containing words.

```
show parse text1
It was a dark and stormy night...
```

It looks the same, but if you count the items, you will see the difference:

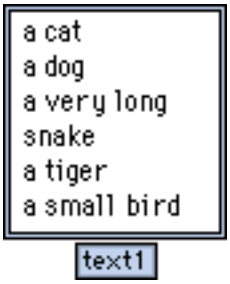
```
show count parse text1
7
```

Text1 reports a word with 33 characters, but **parse text1** reports a list with 7 words.

```
show first parse text1
It
```

The first word of this list is **It**.

There are a few special primitives that consider the contents of a text box to be a set of lines rather than a set of characters. The next example is based on the following text box, named Text1.



It looks like Text1 contains six lines. But the third and fourth lines, “a very long snake,” are in fact a single line that didn’t fit within the width of the box. If you make the text box wider, you will see that it contains only five lines.

MicroWorlds understands only *logical* lines: lines that stop when you press **RETURN**.

Textcount reports the number of lines in a text box. It takes the name of a text box as input:

```
show textcount "text1
5
```

Textitem reports one line from a text box.

```
show textitem 3 "text1
a very long snake
```

Textpick reports one line from the text box, chosen at random.

```
show textpick "text1
a dog
show textpick "text1
a small bird
```

Note that **textitem** and **textpick** report a long word, not a list.

Sliders as Variables

A slider can be used as a kind of variable, one that only reports numbers. The name of a slider reports its current value. Using a slider as a variable has two advantages: you can usually see its value, and you can use the mouse to change the value.

For example, create a slider and name it Speed. Set the minimum to 0 and the maximum to 10 (see *Sliders*, in Section 1). The word **speed** can then be used to report the current value of the slider.

```
forever [fd speed]
```

Note that there is no colon before the word **speed**. It's the name of a slider, not a variable. Now change the value of the slider.

You can change the value of the slider with a command. The word **set** followed by the slider's name (**setspeed** in this example) sets its value. The new value has to be within the slider's minimum and the maximum settings.

You can also change the minimum and maximum settings by giving a list as input. **Setspeed [10 100 20]** would change the minimum to 10, the maximum to 100, and the value to 20.

Talking to Turtles and Text Boxes

Each turtle and each text box has a name. You can see their names using the **?** from the Help menu, or by opening their dialog boxes (see *Turtles* in Section 1).

There are four ways to control which turtle or text box will be affected by your instructions.

1. Type the name of the turtle or text box, followed by a comma, as in **t1,** or **text1,** .
2. Type **talkto "t1** or **talkto "text1**, using the name of the turtle or text box you wish to address.
3. Click on the turtle or in the text box you wish to address.
4. Create a new turtle or a new text box. This one automatically becomes the "current" one.

For the following examples, assume there are two turtles and two text boxes. You can only talk to one text box at a time, but you can talk to many turtles at once.

```
t1,
fd 50
t2,
setsh 8
talkto [t1 t2]
bk 50
text1,
print "hello
text2, print "there
text1, ct
```

Click in the box named Text2, then click in the Command Center to type the next instruction.

```
print "there
```

Whenever you have more than one turtle or text box on a page, you can avoid confusion by starting your procedures with a **talkto** or a "comma instruction."

For example, if you have two text boxes and a button that runs **cleartext**, clicking on the button will clear the last text box that you used. You could easily clear the wrong text box by mistake. Instead of having a **cleartext** button, write a **cleanup1** procedure and make a **cleanup1** button:

```
to cleanup1
text1,
cleartext
end
```

This button will only clear the text box named Text1.

Object Names as Commands

The names of many objects are also commands. The names of recordings, audio CD clips, melodies, QuickTime movies, and videodisk clips are commands that bring the objects into action. The names of text boxes and sliders report their value. You can type the names in the Command Center, or use them in procedures, buttons, etc.

Spaces in Object Names

Avoid using spaces in object names. If you choose one-word names with no spaces, you can use the names as commands.

For example, if you name a melody Mytune, you can type the word **mytune** in the Command Center to play the melody. If you had named the melody My Tune, typing **my tune** in the Command Center would result in the message:

```
my tune
I don't know how to my
```

MicroWorlds may know the object "**my tune**," but it doesn't know a command named **my** or a command named **tune**.

If you have a particular reason for an object name that contains spaces, it is possible to have one: enclose the long word within vertical bars (|). Vertical bars tell MicroWorlds that everything inside is a single word.

```
|my tune|
```

Also be careful, when you're typing in dialog boxes, not to add spaces at the end of object names. A melody named "Jingle " (note the space at the end), wouldn't play if you typed the word **jingle** as an instruction:

```
jingle
I don't know how to jingle
```

Objects on Other Pages

On any page there is one turtle that is "current," meaning that it will follow your instructions. There is also one text box that is current, meaning that text will be printed in it when you use the **print** or **insert** command.

If there is no text box on a page, an instruction like **print "hello** will cause the message:

```
No text box found for print
```

You can control which turtle or text box is current by addressing them by name with a comma instruction, such as:

```
t2,
```

or

```
text3,
```

On any page, no two turtles can have the same name, and no two text boxes can have the same name, and so on for all objects. On *different* pages, though, two text boxes will often have the same name. If you leave the names that are assigned automatically, several pages in a project can have text boxes named text1 and text2.

You can print or insert text in a text box on a different page, if:

- you address it with a comma instruction, and
- there is no text box on the current page with that name

So if there is no text box named Text3 on the page that is open, but there is such a box on another page, you can print or insert text in it, use its name to report its contents, and use the set-text-box-name command (**settext3**) to replace its contents.

Be careful: If the same text box name is used on two (or more) other pages, you'll cause an error message. For example, you could have a Page1 and Page2 both containing a text box named Text1. If you were on another page, with no text box named Text1, any instructions such as **text1, show text1**, or **settext1 "hello** would cause the message:

```
There is more than one text1
```

The reason is that MicroWorlds can't decide which of the two text boxes you're addressing.

This also applies to turtles and using sliders as variables.

Arithmetic

You can use MicroWorlds for many kinds of calculations, but you must keep in mind that there are rules that control the results. If you don't pay attention to the rules, you may not get the results you're after.

Rule 1: Leave a space on each side of the arithmetic symbol:

```
show 100/2
I don't know how to 100/2
```

MicroWorlds thought that 100/2 was one word. It didn't "see" the / sign. With spaces around the / sign, MicroWorlds will understand what to do:

```
show 100 / 2
50
```

Rule 2: Tell MicroWorlds what to do with the answer. If you don't, MicroWorlds will display a message:

```
100 / 2
I don't know what to do with 50
```

MicroWorlds has calculated an answer, but it doesn't know what to do with it. You can **show** the answer, use it for a **print** instruction or a **forward** instruction, or whatever you choose.

```
show 100 / 2
50
print 100 / 2
forward 100 / 2
```

Rule 3: Multiplication and division are always calculated before addition and subtraction:

```
show 3 + 2 * 4
11
```

2 * 4 was calculated first. The result of the multiplication (8) was then added to 3 (total 11).

If you want to force MicroWorlds to do the calculation in a different order, you have to use parentheses (). What's inside parentheses is always calculated first.

```
show (3 + 2) * 4
20
```

Rule 4: Arithmetic is always calculated before other commands:

For example, in an instruction like **show random 6 + 1**, the **6 + 1** is calculated first, and **random** is calculated next. This is a common source of program "bugs," since the intention is usually to have MicroWorlds report a random number from 0 to 5, and then add 1 to it. This simulates the roll of a die, where the random numbers vary from 1 to 6.

What actually happens is that 1 + 6 is added first, so MicroWorlds calculates **random 7**, and therefore produces random numbers from 0 through 6.

You can make MicroWorlds calculate things the way you want, by using parentheses: **show (random 6) + 1** forces **random 6** to be calculated first, and the addition later, which will result in a "proper" roll of the die.

Commands and Reporters

The MicroWorlds vocabulary can be divided into two groups: commands and reporters.

Commands *do* something:

Forward tells MicroWorlds to move the turtle some amount.

Setsh tells MicroWorlds to set the shape of the turtle.

Reporters *provide things* for commands to use. **Pos**, for example, reports a pair of numbers (a turtle's x-y coordinates). But **pos**, like every reporter, doesn't tell MicroWorlds what to do with the numbers. You have to use a reporter with a command. If you don't, MicroWorlds will tell you:

```
pos
I don't know what to do with [0 0]
```

Use **pos** with a command that tells MicroWorlds what to do with the thing that's been reported:

```
show pos
0 0
```

Formatting Your Procedures

There are two things you can do to make your procedures easier to understand. You can add comments and you can format your procedures.

Comments

On the Procedures page, MicroWorlds treats any text between **to** and **end** as part of a procedure.

The title line (starting with **to**) and the **end** line must each be on a line of their own. The title line should contain only the word **to**, the name of the procedure, and the name of the variables (the inputs).

Any text that is *not* between **to** and **end** is not part of any procedure. You can use this space to type comments.

```
This procedure starts an animation and plays a
melody:

to trick
launch [animate]
mytune
end

Mytune is the name of a melody and animate is a
subprocedure.

to animate
repeat 100 [setsh 1 wait 5 setsh 3 wait 5]
end
```

You can also put comments inside procedures by inserting a semicolon. Any text between a semicolon and a carriage return is ignored.

Indenting Instructions

Long instructions which take up more than one line are sometimes difficult to read. It is also more difficult to ensure that all square brackets are in matched pairs:

```
to next
question [Hall or cave... Where do you want to
go?]
carefully [getpage answer stop][announce [There is
no such place.]]
next
end
```

To make long instructions easy to read, split them over separate lines. Insert spaces to line up portions of a long instruction.

```
to next
question
 [Hall or cave... Where do you want to go?]
carefully
 [getpage answer stop]
 [announce [There is no such place.]]
next
end
```

Section 3
Advanced Techniques

This section describes programming techniques, most of which are unique to MicroWorlds Logo. With these techniques you can build more advanced projects.

Page Order

A project can contain many pages. The order in which these pages will appear when you open the project depends on two factors:

1. If the project contains a page named Page1, this page will appear first.
2. If there is no page named Page1, the first page to appear will be the page that was showing when the project was saved.

Therefore, if you make a project with a special opening page, name the page Page1. You can also write a **startup** procedure that displays the page of your choice. See *Startup Procedure*, below.

Startup Procedure

Startup is a special name for a procedure that runs the moment a project is opened. Procedures are defined on the Procedures page. A **startup** procedure can't have an input.

This **startup** example opens a page named Welcome, then starts an animation.

```
to startup
welcome
turtletype [Welcome to my World!]
snapshot
t1, repeat 20 [fd 10]
restore
end
```

This second **startup** example creates a global variable, sets a slider named Speed to 0, and clears a text box named Comments.

```
to startup
make "points 0
setspeed 0
comments, ct
end
```

Question and Answer

Question and **answer** are primitives for opening a dialog box to ask a question, then using the answer that was typed in. After a question is asked and answered, the **answer** primitive reports what was typed by the user. **Answer** holds the answer and can report it repeatedly until you use **question** again.

Type:

```
question [How old are you?]
```

Type your answer in the dialog box. After you click OK, the answer can be used repeatedly. In the Command Center, type:

```
show answer
11                  The number depends on what you typed.
fd answer           The turtle goes forward 11 (or the number
                    you typed) steps.
if answer > 10 [show [That's old!]]
That's old!
```

What **answer** reports is a word. It may contain spaces, but it is still only one word (see *Long Words* in Section 2).

It is common to ask the user to respond yes or no to a specific question. For example, if the question is

```
question [Should we continue?]
```

the user may type Y, or Yes, or, by mistake, put a space before or after the word yes.

In this case, it is better to use **member?**, rather than **=** to check the contents of **answer**. The following instruction will only work if the answer is yes, with no space before or after the word.

```
if answer = "yes [do.this]
```

The next instruction will work as long as the answer contains a **y** (upper or lower case):

```
if member? "y answer [do.this]
```

Sometimes you may need a specific type of answer. In the example below, if you type an answer that is not a number, the instruction starting with **if answer > 10** will produce an error message because > (greater than) only works with numbers. In the following example, the user has typed eleven instead of 11 in the question dialog box:

```
if answer > 10 [show [That's old]]
> doesn't like eleven as input
```

This kind of problem could stop an interactive program. To prevent that, you can check whether the answer is the right kind of answer, and if not ask the user to try again.

If you need a numerical answer, use a procedure like this one:

```
to insist :list
question :list
if number? answer [stop]
insist :list
end
```

If the answer is not a number, the question will be asked again.

Usually, you want to avoid cases where the user doesn't answer the question and just clicks OK. That kind of "empty" answer could also stop your program. If you really need an answer from the user, use a procedure like this one instead of a plain **question** instruction:

```
to insist :list
question :list
if empty? answer [insist :list]
end
```

If you need a one-word answer, use a procedure like this one. It checks to see if the answer contains a space (ASCII character 32), and if so the question will be asked again:

```
to insist :list
question :list
if not member? char 32 answer [stop]
insist :list
end
```

The location where the dialog box appears, and its size can be set using the **set** command.

Processes and Synchronization

You've probably noticed that in MicroWorlds you can do many things at the same time. For example, you can click on a button and then on a turtle; you can play a song while a bird flies. This seems perfectly natural because this is how things happen in real life. In a programming environment, this is called parallel processing. Each time you use a button, or program a turtle to run an instruction by clicking on it, an independent process is launched. You can also launch processes from procedures or the Command Center by using the **launch** and **forever** primitives.

Buttons and Turtles

When you click on a button to run an instruction, a process starts up. You can then click on another button or type an instruction in the Command Center. When you click on a turtle that's been programmed to run an instruction, you also are starting an independent process. Clicking on Audio CD icons or Melody icons also start independent processes.

In the buttons' and turtles' dialog boxes, you can set an instruction to run Once or Many Times. Many Times just repeats the instruction continuously. Usually, you set a procedure or a long instruction to run Once. A short instruction is usually set to Many Times.

Command Center

Typing words in the Command Center tells MicroWorlds to run them as an instruction. Usually, MicroWorlds "reads" an instruction and then runs it *before* "reading" the next instruction. When MicroWorlds is busy running the current instruction, a large dot appears at the end of the instruction in the Command Center. This dot tells you that MicroWorlds will not accept other instructions until the current instruction is finished.

Try these instructions in the Command Center (you'll need two turtles on the page):

```
t1, repeat 100 [fd 1]
t2, repeat 100 [fd 1]
```

Press **RETURN**.

You can click in the Command Center to stop the first instruction. The dot disappears, and then you can type in the second instruction.

There are two primitives which allow you to run instructions as independent processes (which is how they're run from buttons). When you type an instruction beginning with **launch** or **forever**, the dot in the Command Center disappears immediately, and you can run another instruction right away. This is because **launch** and **forever** start independent processes.

Launch

Launch takes an instruction list as input. It runs the instructions as if you typed them alone on a line. The difference is that the instructions are run as independent processes, and MicroWorlds does not wait until they're finished before going on to the next instruction. Try the same two instructions as above, but this time add **launch** at the beginning of the line:

```
t1, launch [repeat 100 [fd 1]]
t2, launch [repeat 100 [fd 1]]
```

Now the dot appears only for the moment when MicroWorlds is "reading" the instruction, then it disappears so you can type in another instruction.

In short, you can use **launch** to start an instruction that takes a long time to run, and once it's running you can start something else.

Forever

Forever repeats an instruction continuously while still allowing MicroWorlds to run another instruction. Here is an example:

```
forever [fd 1]
forever [rt 1]
```

Press **RETURN**.

Using **fd 1** as a button's instruction and setting it to Many Times is similar to typing **forever [fd 1]** in the Command Center. You can stop the button's instruction by clicking on the button again, and you can stop a **forever** instruction by selecting the instruction in the Cancel menu (in the Edit menu).

Warning: Never use **forever** in a button or turtle's instruction that is set to Many Times. **Forever** will keep launching processes until MicroWorlds displays a message "Too many processes." If this happens, choose Stopall from the Edit menu.

When

When starts an independent process. The inputs to this command are two instruction lists. The first list is a "condition" that must report **true** or **false**. Whenever this instruction reports **true**, the second instruction list is run. In the following example, the **when** command sets a process that checks if the y coordinate of the current turtle is greater than 50. If so, it moves that turtle back 20 steps.

```
when [ycor > 50] [bk 20]
seth 0
repeat 1000 [fd 1]
```

Note that you only run a **when** command once. If you run a **when** command many times, you will, in fact, launch many processes. Choose Stopall or Cancel from the Edit menu or press **APPLE - .** to stop the **when** process.

Synchronization

In the following procedure, the animation and the music will start at the same time but will probably not end at the same time.

```
to trick
launch [animate]
mytune
end
```

One way to synchronize two processes is to run the first instruction **forever**, then start the second one, and cancel the first one when the second one is finished. For this to work, both

trick and **animate** have to be redefined:

```
to trick
forever [animate]          Run animate repeatedly.
launch [mytune]            Play the melody while the animation runs.
waituntil [done? [mytune]] Wait here until the melody's
                           finished.
cancel [animate]           ... then stop animate.
end

to animate
setsh "usa1 wait 5 setsh "usa2 wait 5
end
```

Animate changes the turtle to each of two shapes. The **forever** instruction (in the **trick** procedure) runs **animate** repeatedly so the shapes keep changing.

The **launch [mytune]** process stops when the melody is over. But since it is a **launch** instruction, MicroWorlds immediately goes to the next instruction, **waituntil. Waituntil** waits until **done? [mytune]** reports **true**. Then, MicroWorlds goes to the next instruction to cancel the animation.

Make sure the instruction given as input to **cancel** is the same as the input to **forever**.

Carefully

Use **carefully** to run an instruction that may fail to work. If the instruction fails, the procedure still won't stop.

Here's an example of a problem that's solved with **carefully**: in an adventure game, a question dialog box appears and asks the player where he wants to go. The player's answer is used as input to **getpage**.

```
to next
question
     [Hall or cave... Where do you want to go?]
getpage answer
end
```

If the player makes a spelling mistake or types a word that is not one of the choices, the procedure will stop and MicroWorlds will display an error message. In this example, the player added an extra "s" to cave:

```
caves
getpage doesn't like caves as input
```

The instruction in **next** that is likely to fail is **getpage answer**. Use **carefully** on it.

Carefully has two inputs. Both are lists: the first is the instruction that may fail, in this case **[getpage answer]**. The second is the instruction to run if the first one fails. In this case, the user will be given a message, then the procedure will run again.

```
to next
question
     [Hall or cave... Where do you want to go?]
carefully [getpage answer stop]
   [announce [There is no such place.]]
next
end
```

Creating and Modifying Objects Under Program Control

Many of the operations that can be performed with the Tool Palette and the dialog boxes — object creating, object editing — have corresponding primitives.

With the primitives that create objects, remove objects, or change them under program control, you can create projects that interactively modify themselves.

An adventure game is an example of such a project. Depending on the player's interaction, new buttons can appear, turtles' behavior may change, or buttons may get "clicked" under program control.

Newturtle hatches a new turtle. The input is the turtle name. Pick a name that doesn't already exist. When a turtle is first hatched, it is invisible. You can set its shape, position and heading before making it visible. Consider the following example:

```
newturtle "runner setsh "girl1 setpos [10 100] st
```

Setinstruction gives a turtle an instruction. Use **launch** or **forever** in the instruction, to correspond to the Once and Many Times setting.

Clickon starts the turtle's instruction, just like clicking on the turtle. **Clickoff** stops the turtle's instruction.

```
newturtle "runner
st
setinstruction [launch [fd 50]]
clickon
clickoff
setinstruction [forever [fd 1]]
clickon
clickoff
remove "runner
```

Variables can be given to turtles using the **turtlesown** command:

`turtlesown "speed`	Gives each turtle a variable named speed.
`turtlesown "animal`	Gives each turtle a variable named animal.

The input to **turtlesown** becomes a variable that's linked to each individual turtle. Initially, this variable is empty. You assign a value to a turtle's variable with **set** and the variable name:

`t1, setspeed 12`	Sets turtle t1's speed variable to 12.
`setanimal "horse`	Sets turtle t1's animal variable to the word horse.
`t2, setspeed 10`	Sets turtle t2's speed variable to 10.

The name of the variable itself reports the value of that variable for the current turtle:

```
t1, show speed
12
t2, repeat 10 [fd speed]
```

Use the **get** command to report the list of variables and values for the current turtle:

`show get "t1 "own`	Shows the **turtlesown** variables for turtle t1.
`speed 12 animal horse`	
`remove "speed`	Removes the speed variable from all turtles.

Other **set** commands can be used to control the turtle state. The following examples list all the possibilities. The corresponding **get** commands report the turtle state. Portions in italics should be replaced by values of your choice.

`set` *`"t1`* `"rule` *`[launch [fd 1]]`*	Equivalent to **setinstruction**.
`set` *`"t1`* `"on? "true`	Equivalent to **clickon**.
`set` *`"t1`* `"on? "false`	Equivalent to **clickoff**.
`get` *`"t1`* `"visible?`	Reports **true** or **false**.
`get` *`"t1`* `"rule`	Reports its instruction list.
`get` *`"t1`* `"on?`	Reports its state.
`get` *`"t1`* `"own`	Reports the current property values for this turtle.

Text Boxes

Newtext creates a new text box. The first input is the text box name. Pick a name that doesn't already exist. The second input is the position (the xy coordinates of the top, left corner), and the third input is the size of the text box. Use **transparent** and **opaque** to set a text box's visibility.

The position, size and name, and the contents of an existing text box can be set individually.

Note that the text box's position and size coordinates must fit within the page boundaries.

Other **set** commands can be used to control a text box state. The following examples list all the possibilities. The corresponding **get** commands report the text box state. Portions in italics should be replaced by values of your choice.

```
newtext "text1 [90 90] [200 50]
set "text1 "text [Hi there]      Equivalent to settext1 [Hi there].
set "text1 "visible? "true       Equivalent to showtext.
set "text1 "visible? "false      Equivalent to hidetext.
set "text1 "pos [0 0]            Sets the position of the text box.
set "text1 "size [100 100]       Sets the size of the text box.
set "text1 "transparent? "true   Equivalent to transparent.
set "text1 "transparent? "false  Equivalent to opaque.
set "text1 "showname? "true      Displays the name tag.
set "text1 "showname? "false     Hides the name tag.
get "text1 "visible?             Reports its current state.
get "text1 "pos                  Reports its current state.
get "text1 "size                 Reports its current size.
get "text1 "transparent?         Reports its current state.
get "text1 "showname?            Reports its current state.
get "text1 "text                 Equivalent to text1 as a
                                 reporter.
```

Buttons

Newbutton creates a new button. The first input is the button's name. Pick a name that doesn't exist. The second input is the position (the xy coordinates of the top, left corner), and the third input is the button's instruction. Buttons created with **newbutton** are set to Once.

A button's position, size, state, and instruction can be set individually.

Use **launch** or **forever** in the instruction, to correspond to the Once and Many Times setting.

Every button has an "on?" state which is either **true** or **false**. The on? state is **true** if the button is "clicked" and the instruction is being run. When the action is finished, the state returns to **false**.

Other **set** commands can be used to control the button state.

The following examples list all the possibilities. The corresponding **get** commands report the button state. Portions in italics should be replaced by values of your choice.

```
newbutton "button1 [90 90] [fd 50]
set "button1 "pos [0 0]          Sets the position of the button.
set "button1 "size [100 100]     Sets the size of the button.
set "button1 "rule [launch [fd 1]]    Sets its instruction.
set "button1 "rule [forever [fd 1]]   Sets its instruction.
set "button1 "on? "true          Turns the button on.
get "button1 "pos                Reports its current position.
get "button1 "size               Reports its current size.
get "button1 "rule               Reports its current instruction.
get "button1 "on?                Reports its current state.
```

Sliders

Newslider creates a new slider. The first input is the slider's name. Pick a name that doesn't exist. The second input is the slider's position (the xy coordinates of the center), and the third input is a list with three numbers: the minimum, maximum, and current value of the slider.

Other **set** commands can be used to control the slider state. The following examples list all the possibilities. The corresponding **get** commands report the slider state. Portions in italics should be replaced by values of your choice.

```
newslider "speed [90 90] [0 50 25]
set "speed "pos [0 0]            Sets the position of the slider.
set "speed "showname? "true      Makes the name tag visible.
set "speed "limits [0 100]       Sets the limits.
set "speed "value 50             Equivalent to setspeed.
get "speed "pos                  Reports its current position.
get "speed "showname?            Reports its current state.
get "speed "limits               Reports its current limits.
get "speed "value                Equivalent to speed as a
                                 reporter.
```

Recordings

You cannot create a recording under program control, but you can change many parameters. The following examples list all the possibilities. Portions in italics should be replaced by values of your choice.

set *"record1* "pos *[50 50]*	Sets the position of the Record icon.
set *"record1* "visible? *"false*	Makes the icon invisible.
set *"record1* "showname? *"false*	Makes the name tag invisible.
set *"record1* "on? *"true*	Starts playing the recording.
get *"record1* "pos	Reports its current position.
get *"record1* "visible?	Reports its current state.
get *"record1* "showname?	Reports its current state.
get *"record1* "on?	Reports its current state.

Melodies

You cannot create a melody under program control, but you can change many parameters. The following examples list all the possiblities. Portions in italics should be replaced by values of your choice.

set *"melody1* "pos *[50 50]*	Sets the position of the Melody icon.
set *"melody1* "visible? *"false*	Makes the icon invisible.
set *"melody1* "showname? *"false*	Makes the name tag invisible.
set *"melody1* "on? *"true*	Starts playing the melody.
get *"melody1* "instrument	Reports the current instrument.
get *"melody1* "volume	Reports the current volume.
get *"melody1* "tempo	Reports the current tempo.
get *"melody1* "pos	Reports its current position.
get *"melody1* "visible?	Reports its current state.
get *"melody1* "showname?	Reports its current state.
get *"melody1* "on?	Reports its current state.

Movies

You cannot create a QuickTime movie under program control, but you can change many parameters. The following examples list all the possibilities. Portions in italics should be replaced by values of your choice.

set *"qtmovie1* "pos *[50 50]*	Sets the position of the movie.
set *"qtmovie1* "visible? *"false*	Makes the movie invisible.
set *"qtmovie1* "showname? *"false*	Makes the name tag invisible.
set *"qtmovie1* "on? *"true*	Starts playing the movie.
get *"qtmovie1* "pos	Reports its current position.
get *"qtmovie1* "visible?	Reports its current state.
get *"qtmovie1* "showname?	Reports its current state.
get *"qtmovie1* "on?	Reports its current state.

Audio CD Clips

You cannot create an audio CD clip under program control, but you can change many parameters. The following examples list all the possibilities. Portions in italics should be replaced by values of your choice.

set *"audiocd* "pos *[50 50]*	Sets the position of the movie.
set *"audiocd* "visible? *"false*	Makes the movie invisible.
set *"audiocd* "showname? *"false*	Makes the name tag invisible.
set *"audiocd* "on? *"true*	Starts playing the movie.
get *"audiocd* "pos	Reports its current position.
get *"audiocd* "visible?	Reports its current state.
get *"audiocd* "showname?	Reports its current state.

Videodisk Clips

You cannot create a videodisk clip under program control, but you can change many parameters. The following examples list

all the possibilities. Portions in italics should be replaced by values of your choice.

set ***"videodisk1*** **"pos** ***[50 50]***	Sets the position of the movie.
set ***"videodisk1*** **"visible?** ***"false***	Makes the movie invisible.
set ***"videodisk1*** **"showname?** ***"false***	Makes the name tag invisible.
set ***"videodisk1*** **"on?** ***"true***	Starts playing the movie.
get ***"videodisk1*** **"pos**	Reports its current position.
get ***"videodisk1*** **"visible?**	Reports its current state.
get ***"videodisk1*** **"showname?**	Reports its current state.

Colors

Every color can have an instruction that runs when the turtle passes over that color, as well as an instruction that runs when the mouse clicks on the color.

Use **set** to set these instructions under program control. The mouseclick is the instruction to run if a mouse clicks on the color. The turtlerule is the instruction to run if a turtle stops on the color. The "turtlemode" Once and Each Time settings only affect the turtlerule.

Setting parameters for a given color affects all the colors in that color set. For example, setting the color red (number 15) sets all colors numbered 10 to 19.

set ***"red*** **"turtlerule** ***[bk 50]***	Sets the instruction for turtle detection.
set ***"red*** **"turtlemode** ***"once***	Sets the mode.
set ***"red*** **"turtlemode** ***"eachtime***	Sets the mode.
set ***"red*** **"mouseclick** ***[fd 50]***	Sets the instruction for mouse detection.

Pages

There are no **set** instructions for pages. **Get** instructions report a list of the names of the page elements of a given type on the specified page. As well as the examples below, the last input can be any of the following words: melodies, records, videodisks, qtmovies, qtsounds, audiocds, colordemons.

`show get "page2 "turtles`	Reports the names of the turtles on this page.
`show get "page2 "texts`	Reports the names of the text boxes on this page.
`show get "page2 "buttons`	Reports the names of the buttons on this page.

Creating Your Own Help Balloons

You can create your own Help balloons, to help people use your projects! You can only change the balloons for objects on the page, not the balloons for the Drawing Center, Shapes Center, Tool Palette, or the editors. Here are the steps:

1. Decide which Help balloon you want to change.
2. Type your Help text for that balloon, in a text box. For example:

3. Select the text and then choose Copy from the Edit menu.
4. Choose **?** and open the Help balloon.

5. Press **APPLE -V** to paste the text.

To paste graphics into the Help balloon, follow the same steps. Instead of copying and pasting text, you'll select, copy and paste the background graphics. To make a Help balloon with both text and graphics:

1. Type the text into a text box.
2. Make the text box transparent, and move it near the graphics you want.

3. Pick the stamper. Stamp the text on the background graphics.
4. Use the selection tool from the Drawing Center to select the graphics and stamped text.
5. As shown above, copy the selection, use **?** to open the balloon, then paste.

Protecting Your Page From Changes

The first time you save a MicroWorlds project, you create a copy of your project on disk. Every time you re-save your project under the same name, you save it over the copy that was there before.

Perhaps you don't want to change the project on disk. Perhaps it's complete and you want to protect it from accidental changes. Perhaps you want to try out some new changes to a project but want to save the previous version "just in case." There are ways to protect pages.

Using Save As

Immediately after opening a project, choose Save Project As from the File menu, and use a different name to save the project. Once the project is saved under the new name, your changes won't affect the original.

From the Finder

In the Finder, you can protect your projects by checking Locked or Stationery in the file's Get Info window.

1. Quit MicroWorlds and save your project.
2. In the Finder, click on your project's icon.
3. Choose Get Info from the File menu.
4. Check Locked or Stationery.
5. Close the Get Info window.

The next time you open this project, MicroWorlds won't let you save your changes. You can use Save Project As and change the project name, or decide not to save changes, but either way, the original project will remain unchanged.

Freezing Objects and Graphics

Whether your project is locked or not, you may want to "freeze" some or all of the buttons, turtles, text boxes, sliders, and other objects. Frozen objects can't be moved, changed in size, or removed using the mouse.

For example, if you freeze a text box, you can't move it, expand it, or cut it using the mouse. But you can type text, delete text, or change the contents using **ct**, **print**, **insert**, or **settext1** (set-text-box-name).

If a turtle is frozen, you cannot use the Shapes Center tools to change it, but you can open its dialog box or use commands such as **stamp**, **forward**, and **setsh**.

Freeze takes a word or a list as input. You can freeze objects one by one, a list of objects, or an entire page.

```
freeze "text1
freeze [text1 t1 slider1 mytune]
freeze "page1
```

Unfreeze "thaws" objects so they can be moved and changed. If you need to find an object's name so you can use it with **unfreeze**, open its dialog box or choose **?** from the Help menu.

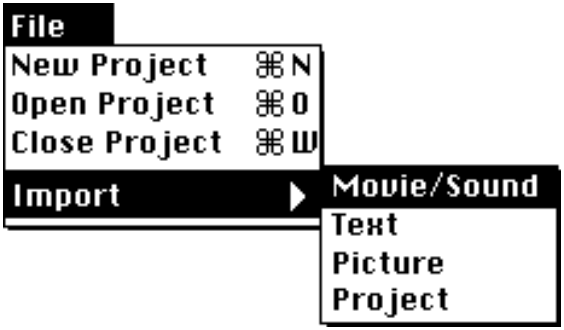
```
unfreeze "text1
unfreeze [text1 t1 slider1]
unfreeze "page1
```

Get can be used to report the names of all the objects on a given page. Use your page name instead of Page1. The second input names the kind of object: texts, sliders, buttons, or turtles:

```
show get "page1 "turtles
[t1 t2 t3]
show get "page1 "texts
[text1 text2 mytext info]
unfreeze "info
```

Freezebg freezes the background of the current page. A frozen background cannot be erased, but you can still add to it. If you draw over a frozen background, you can erase your new drawings, but not those that were present when you typed **freezebg**. The command **unfreezebg** sets the background back to normal.

Importing and Exporting



Using commands or the Import menu, you can import clip art, scanned images, pictures created in other applications, text, and pages from other MicroWorlds projects. You can also export pictures or text created in MicroWorlds to other applications, using the **savepict** and **savetext** commands.

Importing Pictures

You can load a PICT graphic created in another application, onto the background of a MicroWorlds page.

1. Create a graphic using a drawing application.
2. Save it in PICT format.
3. Then, start MicroWorlds and choose Import Picture from the File menu.

Alternately, you can use **loadpict**. See **loadpict** in *Vocabulary*.

Pictures imported using Import Picture in the menu, remain selected so you can move or adjust the size of the picture. Pictures loaded using **loadpict** are dropped onto the background, unselected, so you can make a "slide show" presentation by using several **loadpict** instructions.

A third method for importing graphics consists of using the Macintosh Clipboard. Copy some graphics from any application, start MicroWorlds, click on the page (you can only paste graphics on the page) and choose Paste from the Edit menu.

Exporting Pictures

You can save the background of a MicroWorlds page in PICT format using the **savepict** command. PICT format is compatible with many drawing applications. Just open a page with a background you want to save and type in the Command Center:

```
savepict "mypict
```
Use a file name of your choice.

You may want to save the file into a specific directory. To do this, choose Save Project As from the File menu. Find the folder where you want to save the picture. Do not save your current project! Just click Cancel when you find the directory. **Savepict** will save the file into this folder.

Importing Text

You can load a text file created in another application into MicroWorlds.

1. In a word processing program, save a file in "Text Only" format.
2. In MicroWorlds, choose Import Text from the File menu.

The text will be loaded into the current text location. This could be a text box, the Command Center, or the Procedures page.

Alternately, you can use **loadtext**. See **loadtext** in *Vocabulary*.

Exporting Text

You can export the text in text boxes using the **savetext** command. **Savetext** exports the contents of the current text box (the last one created, used, or talked to). If the Procedures page is showing, **savetext** saves the text in the Procedures page.

```
savetext "mytext
```
Use a file name of your choice.

Importing Movies

See *Movies* in Section 1.

Importing Sounds

To import a sound, choose Import Movie/Sound from the File menu. The Import Sound dialog box opens. Locate and open the folder where the sound files are stored. Sound files in the AIFF and Quicktime formats are displayed. Choose a sound and click Open.

The sound icon will appear on the screen.

Click on the icon to hear the sound, click on it again to stop it or let it end by itself. Importing a sound also creates a MicroWorlds command that plays that sound. For example, if you imported a sound named "tornado" you can use **tornado** as a command in a button, a procedure, or in the Command Center to play the music.

Sounds can be played back from any page in a project. For example, if page1 contains the icon for a sound named "Boo," the **boo** command plays this sound from any page in your project.

You can play a sound while moving the turtle, for example:

```
t1, forever [fd 1] tornado
```

You cannot play a sound at the same time as a melody or a movie.

If you want to access the sound's properties, click on the Sound icon with the eye tool. The dialog box will appear so you can change the sound name, make its icon visible or not, and display its name tag.

The sound file is not loaded into your project; the file remains on your hard disk. MicroWorlds has created two things:

- An icon that will remain permanently in your project.
- A link to the actual sound file. MicroWorlds will remember where the sound file is on your hard disk, so the next time you try to play the sound, MicroWorlds will know where to look for your sound file.

The next time you try to play this sound, MicroWorlds will look in two places to find the sound file:

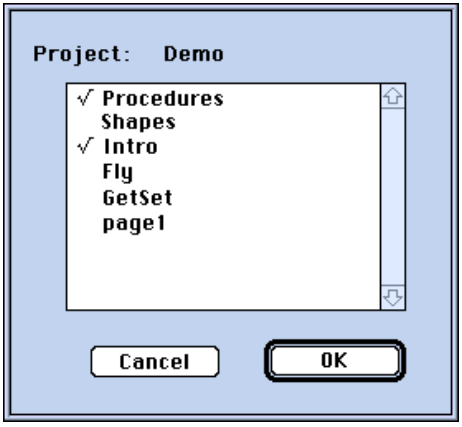
- the directory where your project is saved
- the directory from where the sound was originally imported.

If the file is not in any of these locations, the sound will not play. If you move a MicroWorlds project to a different computer, and the project contains a link to a sound file, be sure to move the sound file as well.

Importing Projects

Pages, procedures, and shapes can be imported from other projects into the current project. Choose Import Project from the File menu. This brings up the Open dialog box, where you select the project you want to import from.

Select a project and click on Open. The Import Project dialog box will appear:



Check Procedures to import the procedures; check Shapes to import all the shapes from the project. Check any pages that you want to add to your project.

Imported procedures are added to the procedures on the Procedures page. After importing, open the Procedures page and make sure that you don't have duplicate procedure names. If two procedures have the same name, only the new one will work.

Imported shapes do not replace all of the shapes in the Shape Center. Only the imported shapes which you have edited replace the corresponding shape numbers in the Shape Center.

If imported pages have the same page name as pages already in your project, the imported one will be renamed. The new names will be the word Page with the first available number (Page1, Page2, Page3...).

256 Colors vs. Thousands of Colors

All Macintosh color computers are capable of operating in 256 colors. Some Macintosh computers are capable of higher resolution modes. MicroWorlds can operate in 256 colors and "thousands" of colors. The setting is made in the Monitors Control Panel.

The color mode of a MicroWorlds project depends on the computer's Monitors setting when the project was created. You can check the color mode of a project by clicking on the rectangle above the Undo button in the Drawing Center.

There are two advantages to the "thousands" color mode:

- Imported graphics, especially photographs and QuickTime movies, look better in that mode.
- You can use decimal numbers to set the turtle's pen color. For example:

```
setc 10
repeat 100 [setc color + 0.1 fd 100 bk 100 setx
xcor + 1]
```

Projects using thousands of colors take a lot more space, both disk space and memory (RAM).

Switching Color Modes

If you have some projects that use 256 colors, and others that use thousands of colors, you should keep these situations in mind:

- If a project was created in thousands of colors mode, and is opened in 256 color mode, the project remains in thousands of colors, but only 256 color mode is displayed. Imported

pictures and graphics using non-integer colors won't look as good, but nothing in the project will be lost. The graphics will look fine when opened again in thousands of colors mode.

- If a project was created in 256 color mode, and is opened in thousands of colors mode, the project is limited to 256 colors.

MicroWorlds Player

MicroWorlds V2.01 Player allows you to run your MicroWorlds projects on computers that don't have MicroWorlds. Licensed use of the MicroWorlds 2.0 Player application is subject to the terms set out in the MicroWorlds 2.0 Player License Agreement. (See Read Me First on the MicroWorlds application disk.)

MicroWorlds Player is especially convenient for viewing projects at home. Just copy it to your home computer's hard disk, and then you can bring projects home to share with your family.

To start up the Player program, double-click on its icon. Use Open from the File menu to open your project. Another way to start up is to drag a project onto the Player icon.

In MicroWorlds Player, there is no Command Center, Tool Palette or menus except for the File, Edit, and Help menus. It is highly recommended that you try out your project with the Player before sharing it with your friends, so you can be sure the project runs properly.

Player mode does not run the following primitives:

procedures
printtext
setfooter
newprojectsize
saveproject
show (automatically changes to **announce** in Player mode)
shownames
cc
merge

Therefore, do not include these primitives in any program that you intend to run with the Player.

Use buttons to run your programs as well as change pages in the project since there is no Pages menu.

If you would like to view the project in Presentation mode, make a **startup** procedure running the **presentationmode** command.

Importing and Exporting MicroWorlds Projects

The most important feature of MicroWorlds 2.03 for Macintosh is the ability to import MW 2.0 Windows projects for use on the Macintosh, and export MW 2.0 Macintosh projects for use on Windows computers. Two MicroWorlds primitives are used: **import** and **export** (see MicroWorlds Help Vocabulary). These primitives are only available on the Macintosh version of MicroWorlds, so MicroWorlds project conversion must be done with a Macintosh computer.

We recommend increasing the preferred size for MicroWorlds to 10000K when importing and exporting. Choose Get Info from the File menu when the MicroWorlds icon is selected in the Finder to change the preferred size. Some MicroWorlds Windows projects may have to be simplified or split into smaller projects to work within the Macintosh environment.

MicroWorlds 2.03 can import Windows projects and export projects to Windows almost seamlessly. All MicroWorlds objects such as turtles, shapes, procedures, text boxes, and buttons are converted automatically. After importing a Windows project, you can run it almost effortlessly on the Macintosh. But there is an important exception. Although the MicroWorlds project can be imported and exported to and from Windows, the media resources that are external to MicroWorlds are not automatically converted with the project. Media resources include sounds, midi music files (on Windows), and movies.

MicroWorlds 2.03 provides media resources that correspond almost exactly to those included with MicroWorlds 2.0 Windows. If a project includes sounds/midis/movies from these MicroWorlds resources, you just need to find the

corresponding media resource (in the Macintosh MicroWorlds CD ROM or the Windows MicroWorlds CD ROM). You can directly copy these files into your project folder.

Here is a table of the media resource formats comparison between Macintosh and Windows:

	Macintosh	Windows
sounds	AIFF	.wav
midis	quicktime sounds	.mid
movies	quicktime movies	.avi

Note: If you are importing many projects from Windows, we recommend using **merge** to change the project size in MicroWorlds Windows to the same size as MicroWorlds Macintosh and limiting the number of pages in a project to less than five.

Importing MicroWorlds 2.0 for Windows projects

To import a Windows project into MicroWorlds 2.03:

1. Create a folder in which to place the original Windows project and media resources. Copy the Windows files into this folder from an external disk or using a network utility.

Note: If your project does not contain media resources external to the project, skip to Step 3.

2. If the media resources are taken from the media resources included with MicroWorlds, simply copy the Macintosh media resources from the MicroWorlds Macintosh CD ROM. If there are sounds or midis that are new, then use a sound utility. A good source for sound utilities on the Internet is: www.wavenet.com. If you have system 7.6 or higher, you can also use MoviePlayer to convert sounds and midis.

When converting sounds in a sound utility, keep the same name of the sound, but delete the extension, so your project

will run without changing the sound names. For example, if you are converting BEEP.WAV to an AIFF format, name the sound BEEP.

3. Use Save as from the file menu to access the folder you have just created directly.

4. Type in the Command Center:

```
import "myproj.mw2
```
Use the name of your project.

Note: If the Windows project name is more than 8 characters, the name will be truncated with the special character ~. Type the project name exactly as it appears on the Finder.

You can also import a Windows project using the option Import Text from the menu File.

5. The project appears on the screen as Untitled. Name the project using Save as from the File menu.

6. Use the eye tool to check which objects are in the project. .WAV and .MID files appear as QuickTime sound icons in the project. An .AVI file appears as a QuickTime movie poster. If you have already converted the media files, or copied the needed Macintosh media resources into the project folder, the project should be working. See below, Problems/Differences between MicroWorlds Macintosh and Windows, for further adjustments you may have to make in the project.

Note: There is another way to import a MicroWorlds Windows project. You can use Import Text from the File menu instead of typing import in the Command Center. The Windows project file is recognized by MicroWorlds as a text file. This may be confusing since it is not evident that a Windows project file is a text file. In addition, Import Project is an item in the File menu. Import Project only works with Macintosh projects, and is used to merge projects.

Exporting MicroWorlds 2.03 Macintosh projects to Windows

To export a Macintosh project to Windows:

1. Create a folder and place the original Macintosh project and its media resources in this folder.

2. Open the project that you want to convert, then type in the Command Center:

```
export "myproj
```
Use the name of your project.

3. The file appears as a PC file on the Finder, with the extension .mw2 added. Use an external disk or a network utility to copy this file to your Windows computer.

As in importing, you can either copy the MicroWorlds media resources for Windows, or use a utility to convert the media resources. Once you open the project with MicroWorlds 2.0 for Windows, AIFF and QuickTime sound files appear in the Windows project as .WAV icons, and QuickTime movies will appear as .AVI icons.

Note: A sound that you recorded on the Macintosh which was part of the project will become an external AIFF sound when the project is exported.

Problems/Difference between MicroWorlds Macintosh and Windows

1. Sounds and Midis
In Windows, a sound effect can be heard while midi music is playing. On the Macintosh, both sounds and midis are quicktime sounds, and only play on one channel. This means you will only hear one or the other.

On the Macintosh, there are system and MicroWorlds internal sounds. This means there is a slight difference between the Macintosh and Windows media resources. For example, there is no sound called Oops in the Macintosh media resources because it already is a MicroWorlds internal sound. In addition, recorded sounds in the Macintosh are considered internal sounds while all sounds in Windows are external to the project.

2. Laserdisk
There is no laserdisk object in MicroWorlds Windows. Thus, once a Macintosh project containing a laserdisk object has been exported, the laserdisk object is no longer in the project.

3. Colors and Detection
There are two color modes in MicroWorlds Macintosh projects, 256 colors and thousands of colors. MicroWorlds Windows projects can be created only in thousands of colors.

The colors that are programmed may change when a Windows project has been imported on the Macintosh or vice-versa. This means that although you think that you have programmed color 70 on Windows, it may become 80 on the Macintosh.

4. Text Count
If you use text primitives in your programs, you may encounter problems between Macintosh and Windows, because characters are counted differently. On Windows, a line of text has a carriage return and a line feed following it, so textcount reports the number of characters in the line plus 2. On the Macintosh, there is only a carriage return following the line. In other words, you may have to adjust your programs manipulating text.

5. Long Names
If you use an external disk to transfer projects and media files between Windows and Macintosh, you will notice that long names on Windows are truncated to 8 characters on the Macintosh, often with the special character ~. Use the name exactly as it appears on the Finder when using import.

6. Objects
Since projects are much bigger in Windows, objects such as text boxes may be partially out of the screen area when they are imported. Use the eye tool to find the objects. If you cannot grab them with the mouse, use the set command. For example:

`set "text1 "pos [0 0]` puts Text1 back in the screen area.

When using the command get to check which sounds are available on a project page, MicroWorlds Macintosh uses a different word than in Windows. You must use the word qtsounds. See get in the Help vocabulary.

7. Melodies
Melodies created with the Melody Editor on Windows have a diverse selection of instruments. The Macintosh version is more limited. When a melody is converted, you may want to adjust the instruments.

In addition, the timing of melodies on Windows is twice as fast as on the Macintosh, so you may have to make timing adjustments once a melody is converted.

8. Transitions
Transitions between pages are different on the Macintosh and Windows. You may have to adjust the transition after importing or exporting.

9. Text Font
The fonts available on the Macintosh and Windows are different. Many fonts in the Macintosh are not available in Windows and vice versa. You may have to make changes in fonts after importing or exporting.

10. Text and Turtles
In projects where turtles and transparent text appear, text is under a turtle on Windows, whereas it is over the turtle on the Macintosh. You may have to adjust the position of the text to compensate.

11. Movies
Movies can be resized on the Macintosh but not on Windows. If a movie that has been resized is exported to Windows, it will appear with a larger frame than the actual movie. Just delete the movie, and import the Windows format movie from the MicroWorlds media resources instead. Or go back the the Macintosh, set the size of the movie to its original size, and then re-export the project.

12. Announce and **Question**
If you have changed the position of the **announce** and **question** alert boxes, the position will not be the same if it is exported to Windows. The position [0 0] is the top left corner on the Macintosh, while in Windows it corresponds to turtle coordinates, i.e., the center of the screen.

13. Help balloons
MicroWorlds 2.0 Windows does not have the help balloon feature. If you have customized the help balloons in a project, this feature will not be available when the project is exported.

Section 4
Appendices

Appendix 1
MicroWorlds Primitives

Graphics

back (bk)
bg
cg
clean
color
colorunder
distance
fill
forward (fd)
freezebg
glide
heading
home
ht
left (lt)
newturtle
pd
pe
pensize
pos
pu
restore
right (rt)
setbg
setc
seth
setpensize
setpos
setshape (setsh)
setsize
setx
sety
shape
size
snaparea
snapshape
snapshot
st
stamp
towards
turtletype
unfreezebg
who
xcor
ycor

Objects

ask
freeze
get
newbutton
newslider
newtext
remove
resetquicktime
resetvideo
set
talkto (tto)
turtlesown
unfreeze

Text Editing

bottom
cb
cd
cf
cleartext (ct)
clipboard
copy
cu
cut
delete
eol
eot?
fontsize
found?
hidetext
insert
opaque
paste
print (pr)
search
select
selected
setfont
setfontsize
setstyle
settc
show
showtext
sol
stamptext
tc
textcount
textitem
textpick
textwho
top
transparent
unselect

Words and Lists

ascii
butfirst (bf)
butlast (bl)
char
count
empty?
equal?
first
fput
identical?
item
last
list
list?
lput
member?
number?
parse
pick
sentence (se)
word
word?

Screen Management

cc
getpage
getproject
merge
namepage (np)
newpage
newprojectsize
pagelist
presentationmode
printtext
procedures
projectsize
setfooter

Disk Access

directories
erfile
export
files
import
loadpict
loadtext
loadshape
pictlist
placepict
prefix
projectlist
savehtml
savepict
saveproject
saveshape
savetext
setprefix
textlist

Flow of Control and Logic

and
cancel
carefully
clickoff
clickon
dolist
done?
dotimes
errormessage
everyone
forever
if
ifelse
launch
listen
not
or
output (op)
repeat
run
setinstruction
stop
stopall
stopme
touching?
waituntil
when

Workspace

recycle
space

Assigning

clearname
clearnames
createprojectvar
let
local
make
name
names
name?
projectvars
shownames
thing

Math

+
-
*
/
=
>
<
abs
arctan
cos
difference
exp
greater?
int
less?
ln

log
minus
pi
power
product
quotient
random
remainder
rerandom
round
sin
sqrt
sum
tan

Input

announce
answer
key?
mousepos
question
readchar

Time

resett
timer
wait

Sound

note
rest
setinstrument
soundlist

Special

the name of a melody
the name of a sound
the name of a page
the name of a text box
set combined with the name of a text box
the name of a slider
set combined with the name of a slider

Appendix 2
Special Key Combinations

Key Function

APPLE - . Stops all processes.

APPLE - A Selects all the objects on the page, or all the text if the cursor is flashing in a text box, the Command Center, or the Procedures page.

APPLE - C Copies the selection.

APPLE - F Toggles between the Procedures page and the current page.

APPLE - H Turns **?** Help on or off.

APPLE - L Opens a line.

APPLE - N Opens a new project. Opens the Save Changes Before Closing box if changes have been made to the current project.

APPLE - O Opens a project file. Opens the Save Changes Before Closing box if changes have been made to the current project.

APPLE - P Prints the current page. If the current page is the Procedures page, all the procedures are printed.

APPLE - R Runs the selected text as a Logo instruction.

APPLE - Q Quits the MicroWorlds application. Opens the Save Changes Before Closing box if changes have been made to the current project.

APPLE - S Saves the project. Opens the Save Project As dialog box if the project hasn't been named.

APPLE - V Pastes the Clipboard.

APPLE - W Closes the project. Opens the Save Changes Before Closing box if changes have been made to the current project.

APPLE - X Cuts the selection.

APPLE - Z Undoes the last action.

Index

U

V

W