

MAX



Getting Started

Table of Contents

Introduction: <i>What is Max?</i>	3
How To Use The Manuals ; Manual Conventions; ;Other Resources for Max Users	
Setup: <i>Setting Up MIDI</i>	6
System Requirements, Installation, and Authorization; Connecting MIDI Equipment; MIDI Setup Dialog; QuickTime Musical Instruments (Mac OS 9)	
Overview: <i>The Max Application</i>	9
Introduction; Programming With Objects; Normal Objects and User Interface Objects; Ar- guments; Messages; Numbers; Message Order; Windows; Max Window; Patcher Window; New Object List; Text Window; Table Window; Timeline Window; Max Documents; Max Preferences; MIDI; Processing MIDI; Help	
Menus: <i>Explanation of commands</i>	28
File Menu; Edit Menu; View Menu; Object Menu; Font Menu; Options Menu; Trace Menu; Window Menu; Extras Menu; Help Menu; Contextual Menus in the Patcher Window; Ob- ject Contextual Menu; Patch Cord Contextual Menu; Blank Space Contextual Menu; Object Quick Reference Menu	
Objects: <i>Creating a new object in the patcher window</i>	48
Object Palette; Object Box; New Object List; Max Object List	
Externals: <i>Extending the Capabilities of Max</i>	53
Two Kinds of Max Objects; Keep External Objects Where They Can Be Found; Object Map- pings; External Objects and Collectives; Errors When Loading External Objects; External Objects from Third Parties	
Shortcuts	57
Locked Patcher Window; Unlocked Patcher Window; Contextual Menus; Selecting and Moving Objects; Patch Cord Shortcuts; Creating Objects; New Object List; send, receive, and value; Table Editing Window; Any Window; Inspectors	
MIDI: <i>MIDI overview and specification</i>	61
MIDI Messages; MIDI Objects; Raw MIDI; MIDI Messages; Control Changes	
Core MIDI: <i>Using Max with Core MIDI on OSX</i>	64
About Core MIDI; Using the MIDI Setup window with Core MIDI; Default Devices for MIDI Objects; Easy Default Setup; Virtual Input and Output Devices	
OMS: <i>Using Max with OMS</i>	67
About OMS; Using the MIDI Setup Dialog in OMS; Default Devices for MIDI Objects; OMS and Other Applications; Inter-Application Communication (IAC); OMS Names; OMS Timing	
Ports: <i>How MIDI ports are specified</i>	73
Max's MIDI Objects ; Specifying MIDI Ports ; The port Message ; Specifying Ports	

Introduction

What is Max?

“Max is a graphical music programming environment for people who have hit the limits of the usual sequencer and voicing programs for MIDI equipment.”

—Miller Puckette, Max reference manual, 1988

Max was conceived in 1986 as a project for producing interactive music at IRCAM (Institut de Recherche et de Coördination Acoustique/Musique) in Paris. The original author was Miller Puckette. Max became a commercial product from Opcode Systems in 1991 with further development by Puckette and David Zicarelli. Cycling '74 became the publisher of Max in 2000.

Max lets you control your equipment in any way you want. You can create applications for composing music, improvising music, providing accompaniment as you play, sending commands to synthesizers, modifying synthesizer patches—anything you can imagine doing with a computer and MIDI. Because Max turns all control information into a simple stream of numbers, you can “patch” anything to anything else.

Max provides you with a high level, graphical programming language that takes full advantage of the Macintosh's graphic capabilities and icon-based user interface. Programs are “written” using graphical objects rather than text. This reduces the need to learn a lot of arcane commands and syntax, and it provides a clear and intuitive way to write programs simply by connecting objects to each other.

Max takes care of all the low level programming tasks for you. It will trigger events at any arbitrary time in the future, interface to MIDI and other communication protocols, and perform useful logical operations.

Applications made with Max run in real time. Because of its speed, Max enables you to write programs that generate music instantly based on what you play, or that modify your performance as you play.

Max is based on the C programming language. Max provides a simple yet versatile, high level, graphical language which is itself written in C, but will be easy to use for those familiar with almost any other programming language, or even for those who have never programmed before. For those who are fluent in the C language, however, Max can be combined with C code that you write. So, if there's something you need to do that Max can't do—and Max can do a lot—you can write your own Max objects in C.

How To Use The Manuals

The Max User's Manual consists of three volumes: **Getting Started, Tutorials and Topics**, and the **Max Reference Manual**.

Introduction

This volume, **Getting Started**, includes:

Setup — Read this section first.

Tells you what equipment and System software you will need to use Max successfully, how to connect your MIDI devices, how to install the Max application and how to get technical support.

Overview — Read this section next if you have never used Max before.

Contains a description of the Max application, introduces basic terms that will be used throughout the manuals, and tells you how to use online help.

Elements of Max

Contains information on menu commands, keyboard shortcuts, and Max objects.

Max and MIDI

Contains information on the MIDI software protocol, receiving and transmitting MIDI, using Max with CoreMIDI (Mac OS X) or OMS (Mac OS 9).

The second volume, **Tutorials and Topics**, provides a step-by-step course on how to program with Max and a collection of discussions of certain topics unique to programming with Max. This volume includes:

Max Tutorial — Read this section next if you have never used Max before.

Takes you through a step-by-step course on how to program with Max. Each chapter in the Tutorial is accompanied by an instructive program written in Max, found in the Max Tutorial folder.

Max Topics

Contains discussions on issues of programming—data structures, loops, encapsulation, debugging, graphics, making standalone applications, etc.—and explains specifically how those issues are handled in Max.

The third volume, **Max Reference Manual**, should be referred to as needed for detailed information on specific Max objects.

Max Objects

Contains precise technical information on the workings of each of the built-in and external objects supplied with Max, organized in alphabetical order.

Introduction

Max Object Thesaurus

Consists of a reverse index of Max objects, alphabetized by keyword rather than by object name. Use this Thesaurus when you want to know what object(s) are appropriate for the task you are trying to accomplish, then look up those objects by name in the *Objects* section.

An additional document, **Writing External Objects for Max**, explains how to create your own external objects for Max using the C programming language. The document can be found on your Max installation disks, in a folder called *C Developer Materials*.

Most users of Max will not need to read the **Writing External Objects for Max** document. However, if you find you want to add additional functionality to Max, and you know how to program in C, the document contains complete instructions on how to do so, and sample source code is available in the *Max SDK* folder. The instructions assume a thorough working knowledge of the C programming language and the Macintosh Toolbox.

If you are new to Max, we suggest you begin by reading the *Setup and Overview* sections of this manual, then trying a few of the Tutorials. You can also learn by looking at the help files in the *max-help* folder, and by browsing the Max Object Thesaurus and some of the expository chapters in the **Tutorials and Topics** Manual.

Manual Conventions

The central building block of Max is the object. Names of objects are always displayed in bold type, like this.

Messages (pieces of information that are passed to and from objects) are displayed in plain type, like this.

In the “See Also” sections, anything in regular type is a reference to a section of this manual, the **Tutorials and Topics** manual, or the **Max Reference Manual**.

Other Resources for Max Users

The help files found in the *max-help* folder provide interactive examples of the use of each Max object.

The Cycling '74 web site provides the latest updates to our software as well as an extensive list of frequently asked questions and other support information.

Cycling '74 runs an internet mailing list where you can ask questions about programming, exchange ideas, and find out about new objects and examples other users are sharing. For information on joining the discussion, as well as a guide to third-party Max-related resources, visit <http://www.cycling74.com/community/>

Finally, if you're having trouble with the operation of Max, send e-mail to support@cycling74.com, and we'll try to help you. We'd like to encourage you to submit questions of a more conceptual nature (“how do I...?”) to the mailing list, so that the entire community can provide input and benefit from the discussion.

Setup

Setting Up MIDI

System Requirements, Installation, and Authorization

See the file **Read Me Before Installing** for the minimum system requirements, installation instructions, and authorization procedure.

Connecting MIDI Equipment

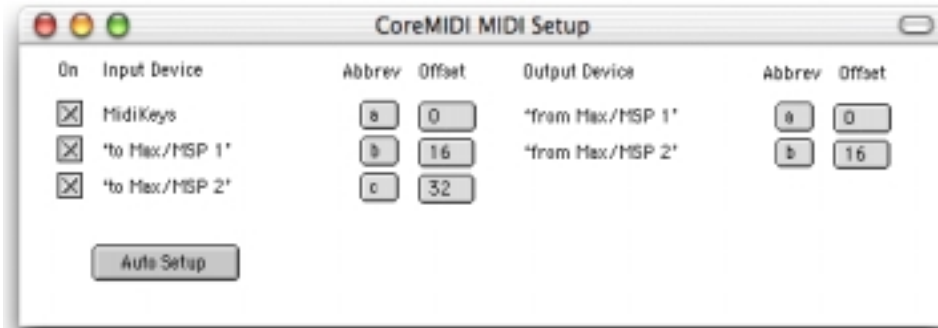
Note: This section covers the use of external MIDI gear with your computer. If you are going to send MIDI to other applications, refer to those applications' setup instructions. If you are planning to use only audio generated by MSP with Max, or don't have any MIDI gear, you don't need to read this section.

1. Make sure your MIDI interface is connected to the appropriate serial or USB port on your computer.
2. Connect the MIDI Out of the MIDI controller(s) to the MIDI In of the interface to send incoming MIDI messages to Max.
3. Connect the MIDI Out of the interface to the MIDI In of the sound source(s) to send output MIDI messages from Max.
4. If you are running Max 4.1 under Mac OS 9 and have not already done so, run the OMS Installer on the Max CD-ROM. If you downloaded Max and need this installer, it can be found on the Cycling '74 web site. Mac OS X comes equipped with a built-in MIDI system called CoreMIDI. You can use the *Audio MIDI Setup* application found in /Applications/Utilities to specify the ports, interfaces and MIDI devices in your MIDI setup.

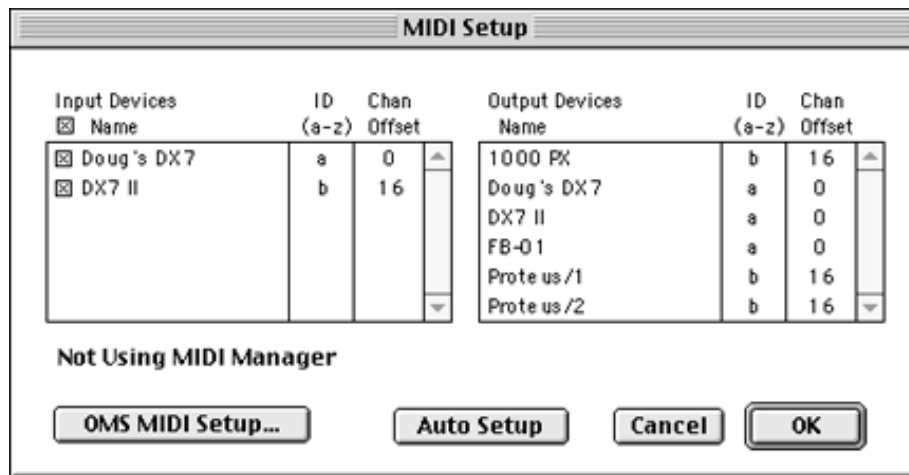
Note: OMS is not compatible with Mac OS X. CoreMIDI is not available in Mac OS 9.

MIDI Setup Dialog

If you choose **Midi Setup** from the File menu), you will be presented with a window showing you the Input and Output devices available:



On Mac OS 9, you specify what ports and interfaces you'll use with MIDI with the OMS Setup application installed by the OMS installer. If you are using OMS for the first time, double-click on **OMS Setup**, a program for telling OMS about your MIDI equipment. Refer to the OMS Manual installed by the OMS installer for details on using this application. After you've finished setting up OMS, run Max, and you should see the following MIDI setup dialog box (if you don't see it, choose **Midi Setup** from the File menu).



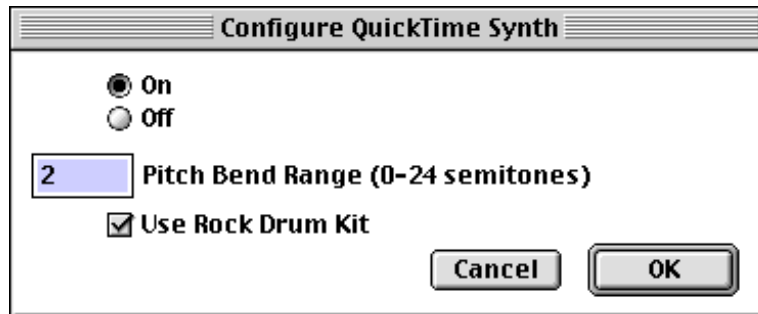
Regardless of what operating system you're using, Max's MIDI Setup dialog box is used to assign abbreviations and MIDI Channel Offsets to each of your Input and Output Devices in your MIDI Setup. The *abbreviation* is for your convenience. You may wish to specify a specific device in creating a Max MIDI object, and it's much faster to type the letter b than *Kurzweil 1000PX*. The Channel Offset is used to distinguish MIDI devices by their MIDI channel. For example, a Channel Offset of 16 sets a MIDI Output Device to receive MIDI on Channel 1 when you specify channel 17 in Max. Setting the Channel Offset for a MIDI Input Device lets you determine the source of a MIDI message. For example, if the message is marked as being on MIDI Channel 17, you know it comes from MIDI channel 1 of a device with a Channel Offset of 16.

It is not essential that you configure this information before using Max for the first time, but MIDI input and output may not be predictable unless you do so. The CoreMIDI and OMS chapters of this manual provide complete information about using this dialog box.

For now, you may wish to click *Auto Setup*, which will assign unique abbreviations and MIDI Channel Offsets to all devices in your system that are connected to different MIDI cables. Devices at the top of the list will be assigned an abbreviation and a Channel Offset of 0. You should set the Channel Offset of your most commonly used Input and Output Device to 0. This will let you address it with the most common range of MIDI Channels, 1-16. If Auto Setup did not do this, you may wish to change the assignments.

QuickTime Musical Instruments (Mac OS 9)

If you wish to use Max to send MIDI to QuickTime Musical Instruments on OS 9, note that its OMS MIDI driver is turned off initially. In order to turn it on, you need to run the OMS Setup application, double click on the QuickTime Musical Instruments icon and click the On radio button in the dialog box to enable the MIDI driver.



Overview

The Max Application

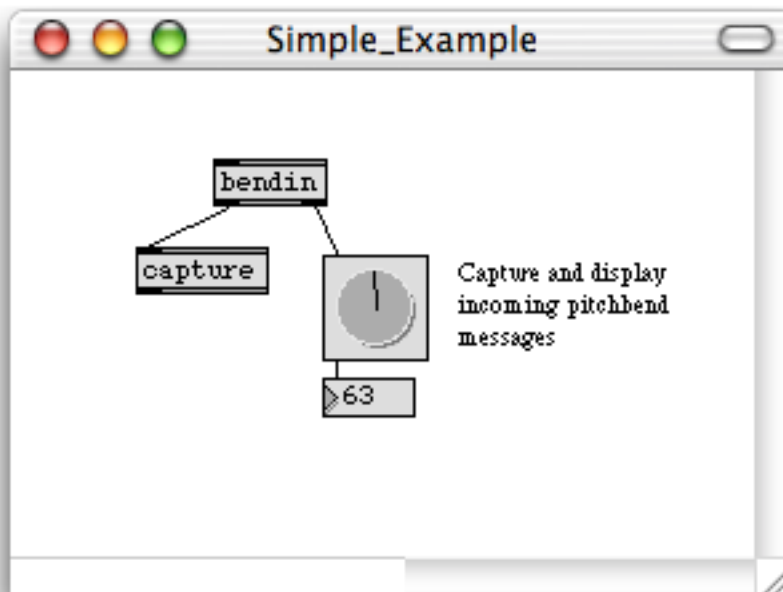
Introduction

This chapter briefly describes the Max application and introduces basic terms that will be used throughout the manuals.

If you have relatively little experience with computer programming or MIDI, you might not immediately understand everything you read in this chapter. Never fear. The Tutorial chapters in the Max **Tutorials and Topics** manual will cover topics in much greater detail, starting from square one. Max is a complex application and we don't expect everyone to "get" everything immediately. This overview will simply familiarize you with the application and its basic elements.

Programming With Objects

When you work with Max, you'll be developing applications *graphically* using objects, which will appear on your screen as boxes that contain either text or icons. You connect objects together to create a working program.

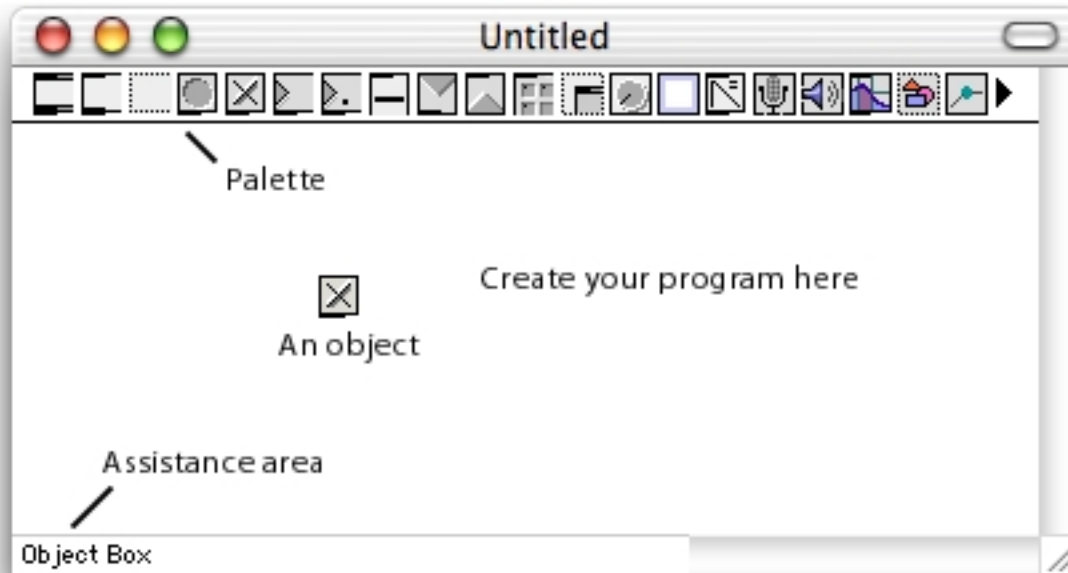


A simple program written graphically in Max

Max has about 200 different objects (and MSP adds about 200 more for audio processing), each of which performs one or more specific tasks. In addition, you can write your own program in Max, save it, and then use that program as an object inside other programs that you write.

Overview

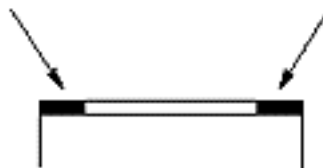
You create a Max program in a *Patcher* window, which works much like a drawing program. You select an object from a palette, then click where you want the object to go.



Objects are different types of boxes, like this:



Objects have *inlets* at the top, used to receive information from other objects...



...and *outlets* at the bottom, used to send information to other objects.



You connect two objects together with a *patch cord*. Always drag from the outlet of one object to the inlet of the other. When dragging a patch cord, you can be anywhere inside the object box and the inlet will “expand” to indicate that you can release the mouse button to make the connection.



Inlet expanding to accept a patch cord

Normal Objects and User Interface Objects

The most common type of object is the *object box*, which has two lines at its top and bottom.



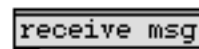
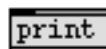
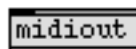
The action performed by an object box depends on the name you type into it. The name is like a verb, describing what the object will do. The name will either be a single word, such as **makenote**, or a symbol such as + (add) or > (is greater than).

Typically, objects send information out their outlets in response to information they receive in their inlets. Each inlet and outlet of an object has a different purpose. These are described in detail in the *Objects* section of the **Max Reference Manual**. A brief description of the use of each inlet and outlet is available in the *Assistance* area of the Patcher window, as shown in the next example. Just move the mouse over the inlet, and descriptive text appears in the Assistance area.



Getting Assistance for the right inlet of the delay object

A few objects lack either inlets or outlets because they receive or send information from somewhere outside of the patch that contains them. For example, the **midout** object has no outlets because it transmits values directly to MIDI.



Other objects are *user interface* objects. They look like buttons, dials, sliders, keyboards, etc., and respond not only to messages received from other objects, but also to clicking and dragging with the mouse. They let you control the program by using the mouse, and also provide a means of dis-

playing numbers and other messages onscreen in a variety of ways. The name of each object is printed in the assistance area when you click on its icon in the palette.



There are some objects, such as **comment**, that do nothing. The comment object lets you put notes to yourself reminding you what you have done, or notes to others telling them how the program works. You can also use comment objects to label user interface items to tell other people how to use your program.

When you've finished making your program, you can save it as a Max document, otherwise known as a *patch*. If you give your patch a single-word filename (that doesn't begin with a number), you can then use it as an object within another Max patch just by typing the filename into an object box.

For a more detailed explanation of Objects and related issues, you can refer to the chapters titled *Objects* and *Externals* in this manual and the chapter titled *Encapsulation* in the **Tutorials and Topics** manual.

Arguments

Often additional words or numbers are included in an object box after the object name to provide initial information to the object or to specify some of its characteristics. The additional words are known as *arguments* to the object.

When you create a metronome object **metro**, for example, you may type in a number argument after the message name to specify how many milliseconds the metronome should wait between ticks.



A few objects have *obligatory* arguments, while others have *optional* arguments. If you forget to type an obligatory argument into an object box, Max will print an error message in the Max window advising you of that fact, and will refuse to create the object. If you choose not to type in an optional argument, Max provides a default value for that argument. The type of arguments that can be given to each object—and their default values—are detailed in the *Objects* section of the **Max Reference Manual**.

Messages

What actually gets sent through the patch cords? A *message* is the information passed from one object to another. A message can be a number, a *list* of numbers separated by spaces, a word (referred to in Max as a *symbol*), or any arbitrary combination of words and numbers. The contents of a message determine its *type*.

The types of Max messages are:

- int When a message consists of nothing but an integer number (such as the message 127), it is understood by Max to be a message of type int. A message could also say int 127.
- float When a message consists of nothing but a number containing a decimal point (such as the messages 3.97 or 3. or .97), it is understood to be a message of type float. The decimal point lets Max know it is a float. As with ints, it is possible—but not necessary—for a message to say float 3.97.
- list A list of two or more numbers separated by spaces (such as 60 79 1.02 4) is of the type list. The message need not (but may) begin with the word list. Max recognizes a list whenever it sees a message consisting of a number followed by anything. In fact, a list can contain words as well as numbers (as in the message 1 start 768), so long as it *begins* with a number.
- bang The message bang is a special message meaning “do whatever it is you do.” For example, when the **random** object receives the message bang, it sends a randomly chosen number out its outlet.
- symbol A symbol is a word or other non-numeric collection of characters. Many symbols are meaningful commands when received by certain objects. The exact symbols understood by each object are listed in the *Objects* section of the **Max Reference Manual**. For example, the **seq** object, a sequencer for recording MIDI performances, responds to messages received in its inlet such as start, stop, record, delay, and print. All of those messages would be meaningless to an object like * (multiply), which expects only numbers (or bang) in its inlets.
- any message A message can, in fact, consist of *any* combination of words and numbers. Some objects can handle any type of message. Most objects, however, expect to receive one of the above types of message, and will not understand other messages.

When an object receives a message it doesn't understand, it will either ignore the message entirely or will print an error report. In many cases, Max even stops you from making such mistakes while you are editing a program. As you connect an outlet of one object to the inlet of another object, Max does its best to analyze what message will be travelling through the patch cord. Max won't let you connect an outlet to an inlet that won't understand the message. If an inlet is not highlighted when you try to connect a patch cord to it, Max will not make the connection.

Numbers

Max distinguishes integer numbers from decimal numbers (with a fractional part). Integer numbers in Max are a data type called *int*, and decimal numbers are a data type called *float*.

When a number is converted from float to int by Max, the fractional part of the float is chopped off (*truncated*) rather than rounded up. Thus, when the number 4.1 is converted to an int, it becomes 4. Likewise, the number 4.999999 becomes 4 when converted to an int.

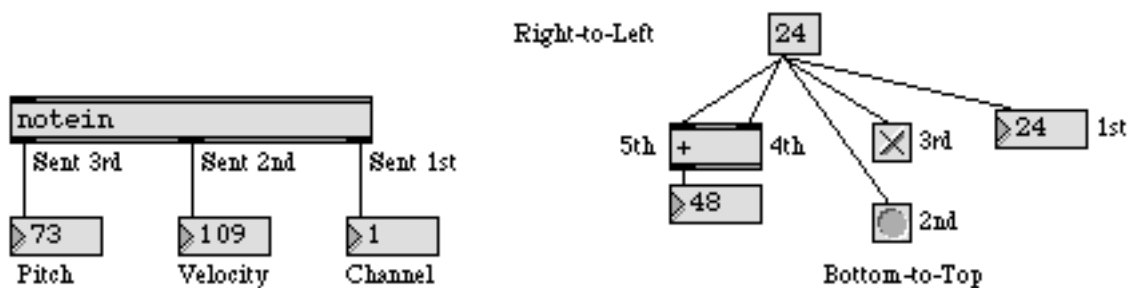
When you write programs that deal with MIDI information, you'll generally use ints. When dealing with audio information, you'll probably use more floats. MIDI data is always expressed in terms of ints in Max. Time is usually expressed in terms of milliseconds in Max, which can be either int or float.

You can type in any int as a hexadecimal number. Hexadecimal is a base-16 number system often used in MIDI specifications and other computer-related documentation. Precede a hexadecimal number with 0x (zero plus a small x), as in 0xF0 (same as the decimal number 240).

Message Order

An object often has more than one outlet, and usually sends messages out all outlets "at the same time." Actually, nothing *really* happens at the same time in Max. Things can happen so fast as to seem simultaneous, but it's important to know how Max orders messages.

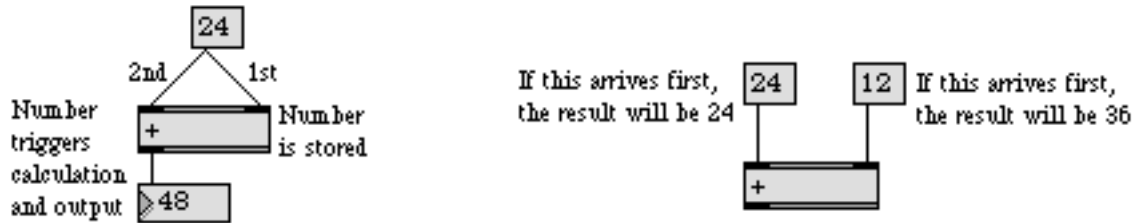
When an object sends several messages "at once", out different outlets, the order is actually *right-to-left*. The rightmost outlet sends its message out first, then the outlet just to the left of that, and so on until the leftmost outlet. This is true of virtually *every* object in Max.



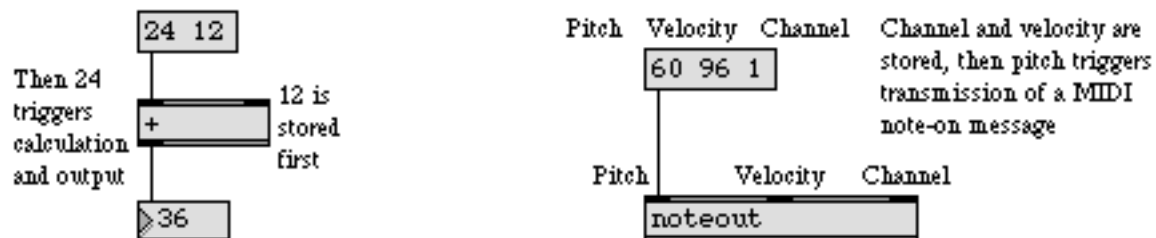
The example on the left illustrates the right-to-left order of message output. In the example on the right, a single outlet has many patch cords connected to it, all leading to different inlets of other objects. In this case, the order that messages are sent is determined by the *right-to-left* screen position of the receiving objects. If two receiving objects are perfectly aligned vertically (neither is further right than the other), the order is *bottom-to-top*; the object that is lower on the screen receives the message first. If a single outlet is connected to two inlets of the same receiving object, the rightmost inlet will receive the message first.

When an object has more than one inlet, it *expects* to receive messages in its inlets in right-to-left order. Generally, an object will store all the messages it receives *until* it receives a message in its leftmost inlet, then it will perform some operation and send messages out its outlets. The vast majority of objects are "triggered" by a message in their leftmost inlet. There are some exceptions to this rule, but the left inlet is the triggering inlet for almost all objects. Notice that this makes sense

because objects send their messages right-to-left, so the last message an object receives will usually be the one that triggers output.

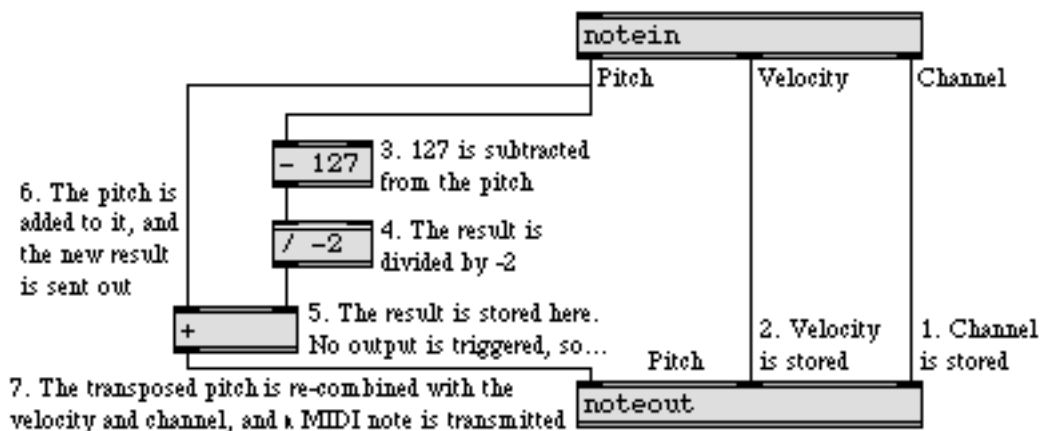


Some objects can receive a list of numbers in the left inlet, and the result will be the same as if the numbers had been received in different inlets in right-to-left order.



Finally, suppose that a message is sent several places from the same outlet, but that message triggers the receiving objects to send *their* messages. What is the order in that case? The answer is that the message will trigger the receiving object, and that object's output will be sent next. The messages will continue triggering other objects until a message is sent that triggers no other action, then Max goes back to the original sending object and sends its next message.

In the example below, note that steps 3, 4, and 5 are all performed (up to the point where no further message is triggered) before the message is sent to the left inlet of the + object.

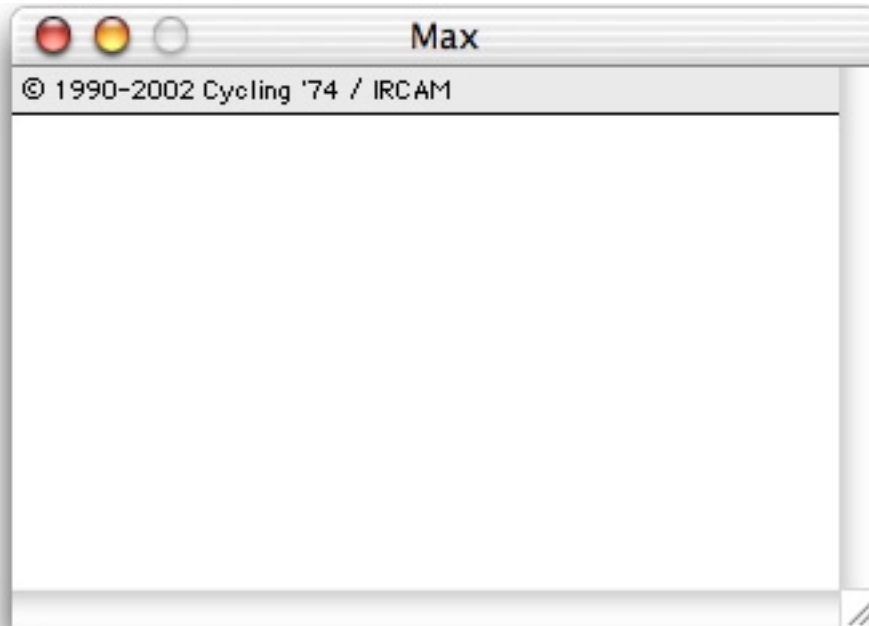


Windows

Max has six main window types: Patcher, Text, Table, Timeline, the Max window, and the New Object List. You can open as many of the first four types of window as you like, but there is only one New Object List and one Max window. Other objects such as env and movie open their own windows.

Max Window

When you open the Max application, you'll see the Max window first. This is the place Max prints out messages to the user.



After you launch Max, you may see a few messages from some external objects that were loaded during the startup process.

Any error messages or warnings generated while you are editing or running a program will appear in the Max window. As you're debugging the programs you write, you can have Max print out exactly what is flowing through the patch cords at any given moment, to see if your patch is working properly. For more information on printing in the Max window, see the *Debugging* chapter in the **Tutorials and Topics** Manual.

On Mac OS 9, the top part of the Max Window displays the amount of free memory you have. If this number dips below 1000K, you might want to consider using the Get Info window in the Finder to give Max a larger memory partition.


Patcher Window

Once the Max application has loaded, you begin programming by opening a new Patcher window (by choosing Patcher from the **New** submenu of the File menu) or by opening an already existing patch (with the **Open...** command in the File menu).

Programs are “written” in Max by placing objects in a Patcher window and connecting them with patch cords to make patches. Once you have created the patch, you can run it and ensure that it actually does what you want it to do. You can save the patch as a Max document, to run or edit again later.

A patcher window is either locked or unlocked. When it is unlocked, you can edit the patcher by moving objects around, creating new ones, and connecting objects together. When it is locked, you are operating the patcher by clicking on objects (such as sliders) that do things.

The lock/unlock state of the window is indicated by the presence of the patcher palette at the top of the window. If you see the palette, then the window is unlocked. On Mac OS 9, there is also a lock icon in the title bar of the window you can click to toggle between locked and unlocked.

Locked:  or unlocked: 

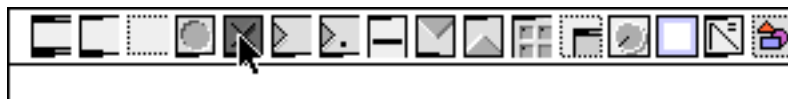
On OS X, there is a transparent rectangular pill on the right side of the window that can be used for toggling between locked and unlocked state.



There are several other ways you can lock or unlock a patcher.

- Click on the lock icon in the title bar of the Patcher window.
- Choose **Edit** from the View menu, or type `z -E`.
- Command-click on the “white space” in the Patcher window.

When you open a new Patcher window, it will already be unlocked. To place an object in the window, click on the desired icon in the palette.



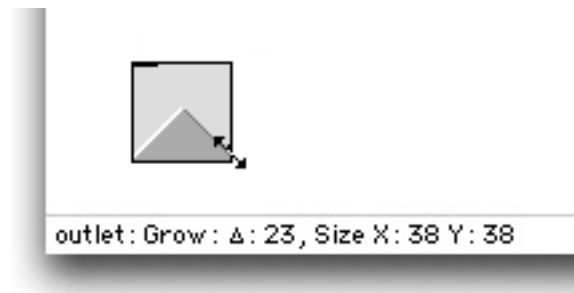
The cursor will become the same as the palette icon.



Then click in the Patcher window where you want to place the object.



You can move one or more objects to a new location by selecting and dragging them with the mouse. You can select objects for editing operations such as **Cut**, **Copy**, and **Duplicate**, or if an object box contains text you can change its font or font size by using the Font menu. You can also resize an object by clicking on its grow bar (the tiny rectangle in the lower-right corner of the object). The cursor will change to a two-way or four-way arrow when you are above the grow bar (which is sometimes invisible). As you resize the object, the assistance window will show you the change in the size of the object, and its current dimensions, in pixels.

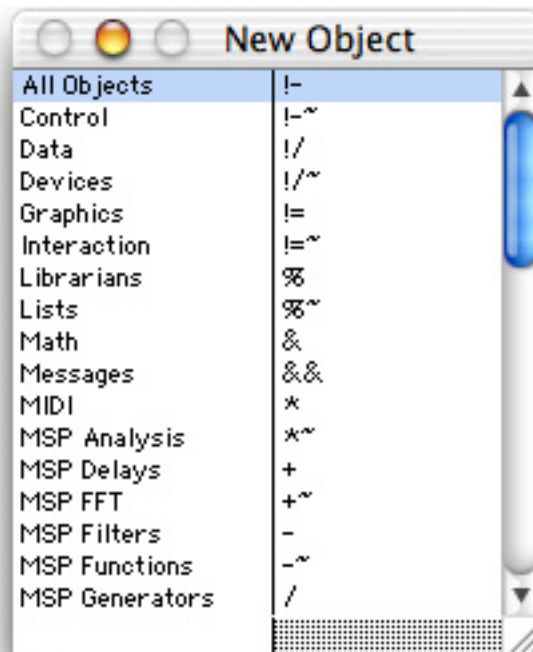


At any time you can lock the Patcher window and use your patch. When the window is locked, the object palette disappears, and clicking on certain objects now operates them as parts of the user interface. When you have finished editing your patch, and have tested it to see that it runs properly, save it by choosing **Save As...** from the File menu. The next time you open that Max document, it will open locked and ready to use.

New Object List

When you add an object box (the leftmost icon in the palette) to a Patcher, Max opens the New Object List window (if **New Object List** is checked in the Options menu). The window is divided

into two columns. The left column lists categories of objects, and the right column lists the names of objects within the selected category.



You can scroll through the list of objects and select one with the mouse, or you can use the Macintosh keyboard to type in the first few letters of an object name as you would in a standard “open file” dialog. (This can be a useful shortcut for typing the entire name of an object, especially when the current category is All Objects.)

When the object name you want is selected, double-click on it, or press Return, Enter, or Space, and the name will be entered into the object box. If you want to open a help file for an object, option-double-click on its name in the list. If you don’t want any of the names shown in the New Object List, press the Delete (Backspace) key, and the window will go away without entering any text into your box.

Text Window

In addition to Patcher windows, you can open simple Text windows for taking notes or for editing the data inside certain objects. Create a new Text window by choosing **Text** from the **New** sub-menu of the File menu.

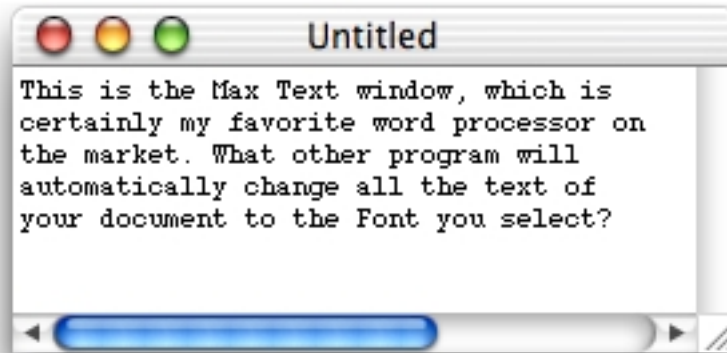
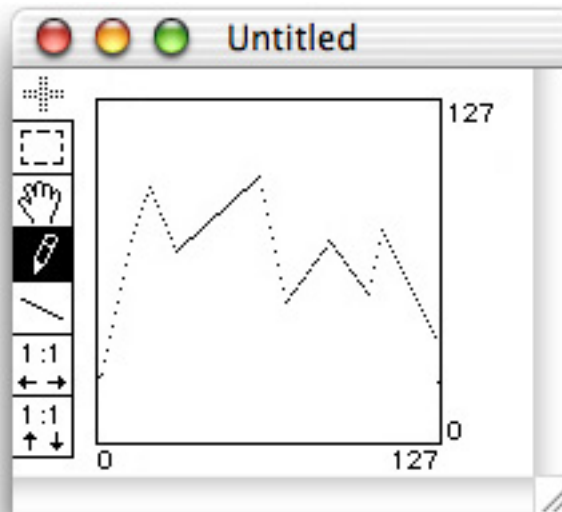


Table Window

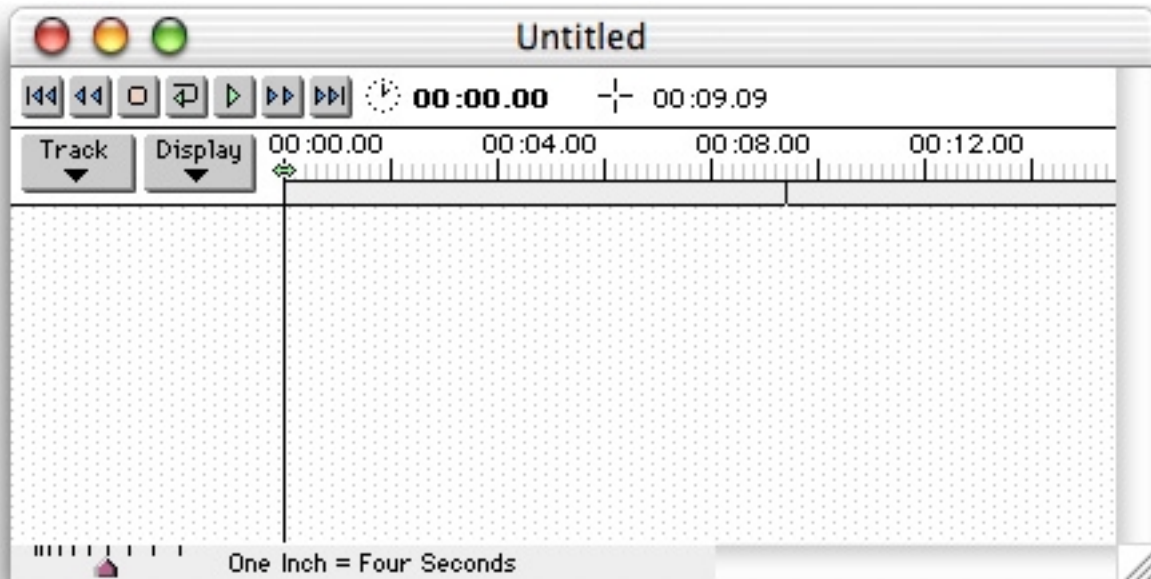
The Table window is used to view and edit an array of numbers. The numbers displayed in such a window may be contained in a corresponding **table** object that you will place in a Patcher window to access the data.



Timeline Window

A Timeline is a graphic arrangement of Max messages, to be sent to specific objects in specific patches at specific times. The timeline can be played using the controls in the Timeline window, or it can be played from within a patch that contains a **timeline** object. Create a new Timeline window

by choosing **Timeline** from the **New** submenu of the File menu, or by typing timeline into an object box.



For more information, see the *Timeline* chapter in the **Tutorials and Topics** Manual.

Max Documents

Whenever Max tries to find a document automatically, it looks first in the folder that contains the patch that's being loaded, then it looks in the folders specified in the File Preferences window (accessed by choosing **File Preferences...** from the Options menu.), then it looks in the folder containing the Max application. This searching order is known as the *file search path*.

There are many kinds of Max documents. The Max application is accompanied by:

- The *max-startup* folder containing external objects (objects that are not contained within the application itself) which Max loads into memory automatically when it starts up.
- The *externals* folder containing additional external objects.
- The *max-help* folder containing example patches demonstrating how each object works.
- The *Max Tutorial* folder containing example patches to accompany the *Max Tutorial* sections of the **Tutorials and Topics** Manual. (These patches are not essential to the Max application.)
- The *patches* folder containing folders which hold Max editors and preference files (*editors*), the Inspectors used to set the attributes for some Max and MSP objects (*inspectors*), graphic images used for some UI objects (*picts*), and useful utility and guide patches (*extras*).
- The *tiAction* folder for containing timeline “action” patches.

Note: If you're using Max/MSP, there are some additional folders created for audio-specific purposes. Refer to the MSP documentation for information on these folders.

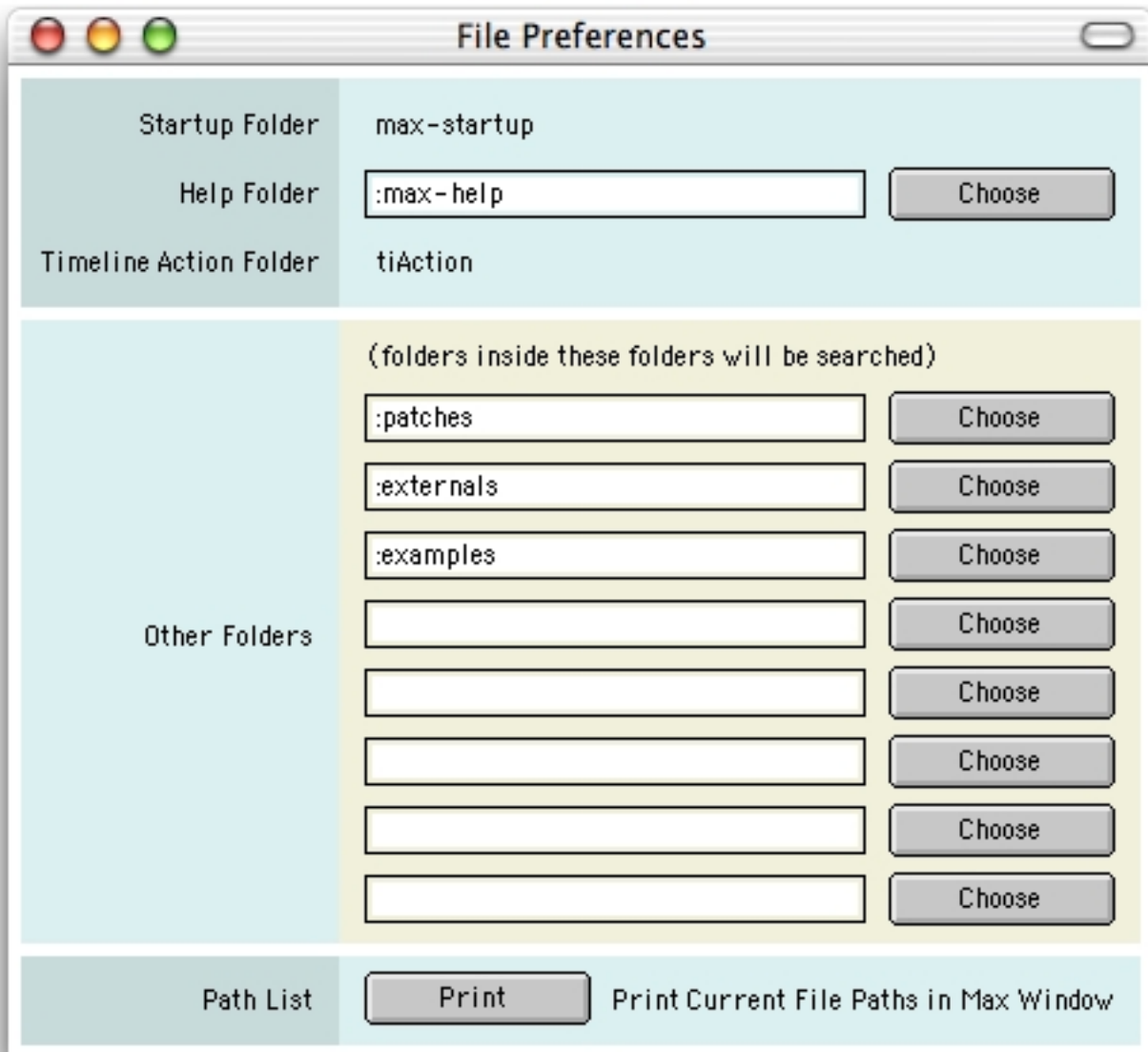
We hope you'll be using Max to create a variety of documents yourself:

- Patches you construct in the Patcher window. These are the documents you will create most often; they are working programs you have written in Max.
- Binary or text files containing data for such objects as **coll**, **funbuff**, **mtr**, **preset**, and **table**.
- MIDI files saved by the **seq** or **detonate** objects.
- Script files describing envelope functions.
- Timelines and timeline action patches.

In order to keep track of all these documents, Max lets you specify the names of folders that it should use when it looks for various files. To name these folders, choose **File Preferences...** from the Options menu.

The **File Preferences** window displays the name of the important folders used by the program. The startup folder, from which external objects are automatically loaded when the application is opened is called *max-startup*. The folder for help files is called *max-help*. The timeline action folder, where files used in timelines are stored is called *tiAction*. You can specify six other folders (and their subfolders) for Max to examine when loading a file.

This is the File Preferences window:



If a directory name starts with a colon, it is a subdirectory of the folder that contains the Max application. Otherwise, the name must start with the name of the volume (disk drive) which contains the folder.

By default, Max will look for patches, external objects, tables, and other files in folders called *:externals*, *:patches* and *:examples*. You can specify additional folders by adding their names in slots below these names.

Max Preferences

Max remembers many of your working preferences by storing settings each time you quit the application. The first time you use Max, a file called Max 4 Preferences is created. On OS 9, this file is in the Preferences folder inside the System Folder. On OS X, this file is in the Preferences folder

inside the Library folder of your home directory. This is where Max stores settings you have chosen for in the **MIDI Setup** and **File Preferences** windows, as well as the current settings of the Options menu.

In addition, you can set a default font and font size for the Max window and subsequent new Patcher windows by using the Font menu while the Max window is the active window. Those font characteristics will be stored in the preferences file.


MIDI

Max has many built-in objects for transmitting and receiving MIDI data to and from the ports available in your MIDI system. Objects that transmit MIDI messages have no outlets, since they send MIDI data out from the Max application. Similarly, objects that receive MIDI messages from the outside world get their input from MIDI devices, rather than from other Max objects.

The **midiiin** and **midiiout** objects receive and transmit raw MIDI data one byte at a time, without analyzing the MIDI messages at all. These objects can then be connected to objects that record the MIDI data, process it, or play it back.

You will probably end up using the specialized MIDI objects the most. They filter the raw MIDI data coming into Max, and output only the vital information. For example, the **notein** object looks only for MIDI note-on messages, and when it receives what it is looking for, it outputs the key number, the velocity, and the channel number. Similarly, the **bendin** object looks only for incoming pitch bend messages, and sends out the amount of pitch bend and the channel number. Other specialized MIDI input objects are **ctlin** for control change data, **polyin** for polyphonic key pressure data, **touchin** for aftertouch data, **pgmin** for program change data, **rtin** for real time messages, and **sysexin** for system exclusive messages.

Each of the specialized MIDI objects that receives channel voice messages has a transmitting counterpart, which receives numbers from other Max objects and uses those numbers as data for transmitting specific MIDI channel voice messages. The specialized MIDI transmitting objects are: **noteout**, **polyout**, **ctlout**, **pgmout**, **touchout**, and **bendout**.

On OS 9, the Patcher window has a button —  — in its title bar that serves to disable MIDI (and audio). When the button shows a MIDI plug icon, the MIDI objects are enabled and function normally. When the button shows an X, all MIDI receiving and transmitting objects in that window are disabled. Clicking on the button with the mouse toggles it back and forth between Enabled and Disabled.

Processing MIDI

Once MIDI data has been received by the MIDI objects, it can be modified and manipulated by other objects in Max. Max has many objects designed specifically to handle MIDI data.

The **midiparse** object receives numbers from **midiiin**, analyzes the type of MIDI message, and routes MIDI channel voice data out specific outlets. The **seq** object is a simple MIDI sequencer that records a sequence of data received from **midiiin** and can play it back later, even changing the speed. A multi-track sequencing object, **mtr**, can record many separate tracks of data, then play them back either simultaneously or individually. The **detonate** object is a complex multi-track

sequencer with a graphic editing window and score-following capability. To format numbers into well-formed MIDI messages, there is a **midiformat** object which prepares MIDI messages, then sends them to **midiout** to be transmitted out from Max.

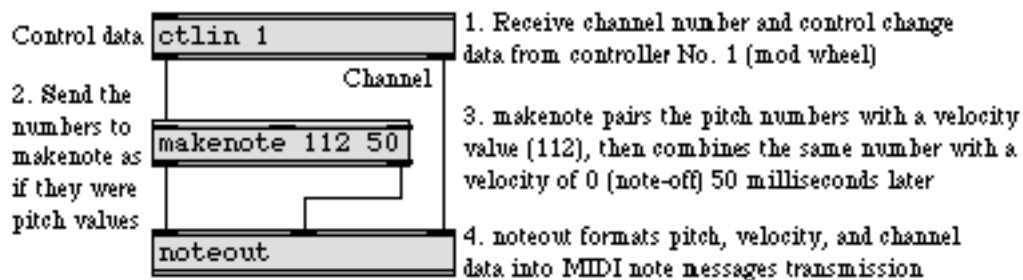
Borax analyzes note-on data received from **notein**, and sends out information regarding the duration of notes, the time between note-ons, how many notes are being held at any given moment, how many notes have been played, etc. Both **Borax** and the **poly** object are capable of assigning voice numbers to the notes that pass through them, and **poly** can steal voices and provide note-off messages to turn off notes that are being held.

The **stripnote** object filters out note-off messages (note-on messages with a velocity of 0) received from **notein**, passing on only note-on messages with a positive velocity. The **makenote** object is like the reverse of **stripnote**. It adds note-offs after note-on messages generated within Max, then sends both the note-ons and the note-offs to **noteout** for transmission. To help avoid stuck notes, the objects **bag**, **flush**, and **midiflush** keep track of the note-ons they have received, and can be used to turn off any notes for which they have not yet received note-offs.

Max usually uses only 7-bit values (numbers from 0 to 127) for pitch bend data, and uses note-on messages with a velocity of 0 to express a note-off. Max is capable, however, of recognizing and transmitting 14-bit precision pitch bend values and MIDI note-off messages with release velocity using **xbendin**, **xbendout**, **xnotein**, and **xnoteout**.

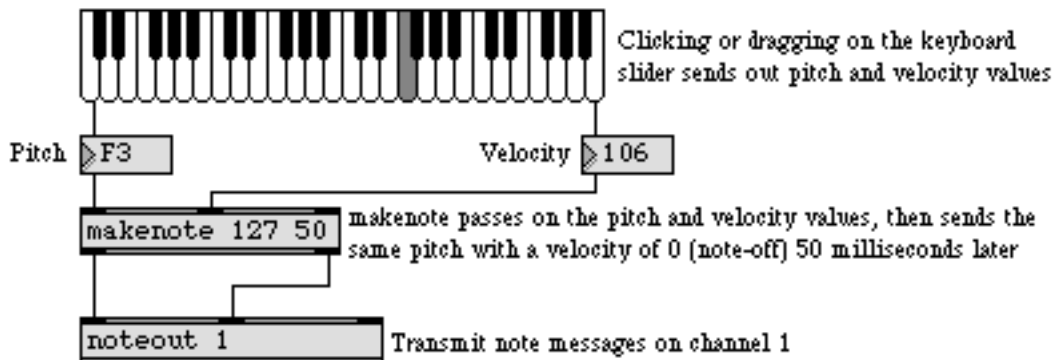
The above objects are specifically designed to handle MIDI data, but since all MIDI values are just expressed as ints in Max, they can be processed in any way imaginable. The **pipe** object, for instance, delays all numbers passing through it by a certain amount of time. Or the **+** object can be used to add some number to a MIDI pitch value, thus transposing the pitch. The other sections in the manuals will show you a few of the many ways to process numbers in Max.

MIDI data can be reassigned to have other meanings. For example, data from the mod wheel of the MIDI keyboard could be received with **ctlin**, sent to **makenote** (to combine the number with a velocity value and add a note-off after it), then transmitted by **noteout**, as shown below. This would let you play *notes* with the mod wheel.



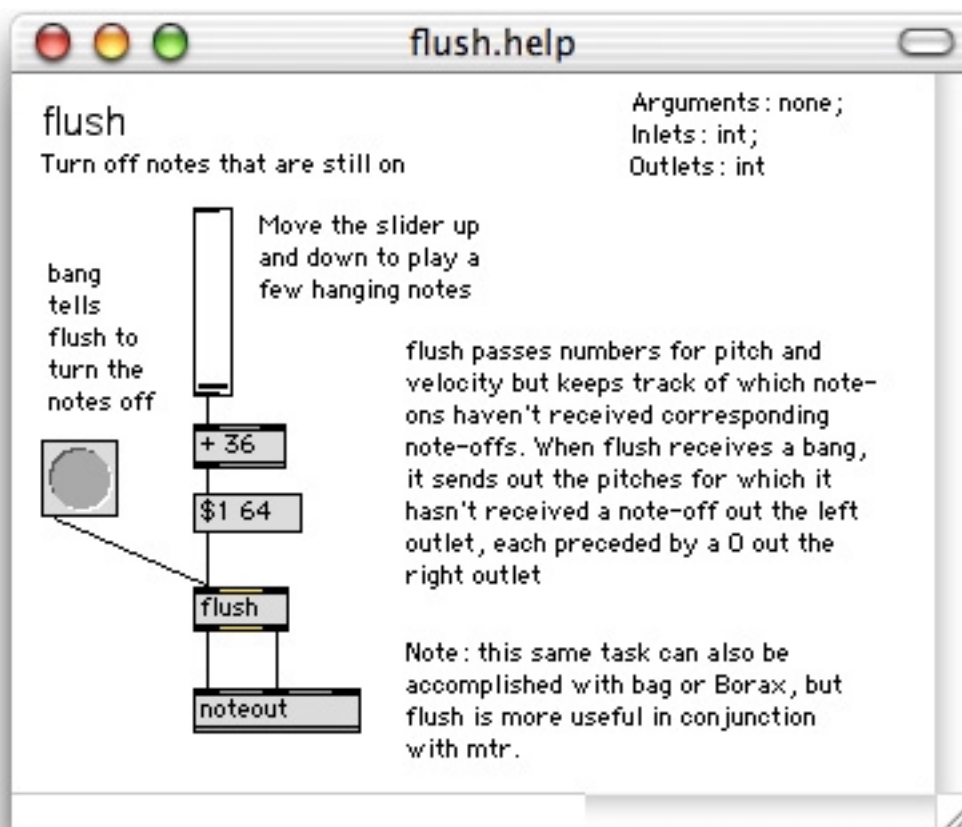
Numbers can also be generated within Max, either automatically with timing objects such as **metro**, **tempo**, and **clocker**, or interactively onscreen with user interface objects such as **slider**, **dial**,

and **number box**. The numbers generated in Max can be sent to MIDI objects and transmitted out to play music.



Help

You can see Help files for built-in and external Max objects at any time. Option-clicking on an object in an unlocked Patcher opens the Help file for that object. You can also option-click on any object's box in a locked Patcher to open a help file for that object if **Help from Locked Patcher** in the Options menu is checked. Help files are actual operating patches that let you see an object "in action." The following example shows a typical Help file.



In an unlocked Patcher, you can also open the Help file for an object by selecting the object and then choosing **Help** from the Help menu. You can open Help files from the New Object List by holding down the Option key while you select the object's name from the list.

If you still have questions about an object, look up the object's complete description in the Objects section of the **Max Reference Manual**. For help with other Max topics, consult the other chapters of that manual, or look up your topic in the **Max Object Thesaurus** or the Index.

Menus

Explanation of commands

File Menu

New... This set of pulldown menus lets you create new Max windows. The submenu options are described below:

Patcher	Create a new Patcher window.
Table	Create a new Table window, to make a table by drawing in values.
Text	Create a new Text window, to enter values in a text file, or simply to use as a note pad. Often values can be entered most efficiently by typing them in as text. For example, a table file can be created just by typing the word table , followed by a space-separated list of values. Once such a text file has been saved, it can be read by table objects in a patch. A funbuff file can be created by typing the word funbuff , followed by a space-separated list of alternating x and y values. Objects for which you might want to type in data in a Text window include coll , env , funbuff , lib , mtr , seq , table , and Text .
Timeline	Create a new Timeline window, for creating a graphic score of Max messages.

Open... Open an existing Max document. Max automatically determines the proper type of window in which to display the document—Patcher, Table, Text, or Timeline.

Open as Text Force Max to open an existing document in a Text window for editing. Any Max document or plain text file can be opened with this command.

Close Close the active (foreground) window. If changes have been made to the document since it was last saved, Max will ask you if you want to save those changes.

Save Save the active document. The values stored in **number box** objects and other storage locations are not saved with a Patcher document. Max saves the values that are stored in data objects such as **table**, **coll**, or **funbuff** within a Patcher, unless the *Save With Patcher* option is set for those objects.

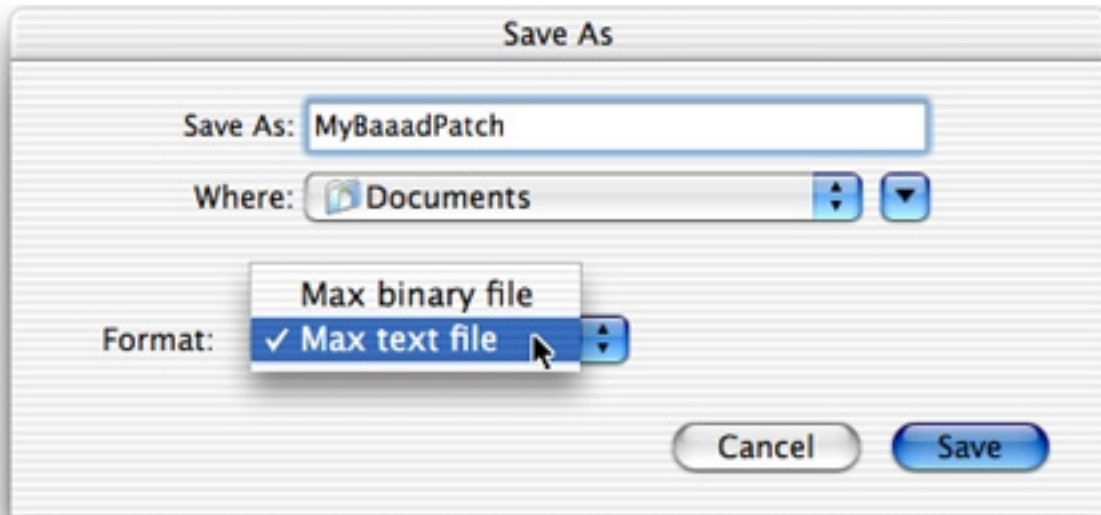
When you save an Untitled document, Max automatically uses the **Save As...** command, so that you can rename the document.

When saving a patch for the first time, it is possible to protect it from future modification, by holding down the Option key while clicking on the *Save* button in the dialog box. This renders the file uneditable when it is opened in the future.

(Before you close the file, however, you can still edit it and save it again without this edit protection.)

Save As... Save the active document in a separate file with a different name. The document is saved under the new name, and the document with the old name is closed unchanged.

By default, most files are saved in Max Binary format, which loads most quickly and takes up the least disk space. You can also save a file in Text format. A pop-up menu in the standard save file dialog box allows you to choose.

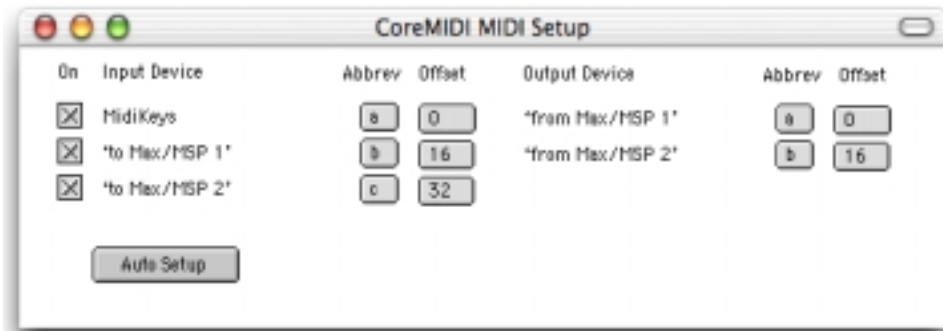


Save As Collective... Save the active document, along with all other files necessary for it to function properly, together as a single collective file. For more information on the collective save dialog, see the Collectives topic.

Midi Setup... Configure Max's MIDI connection to the outside world. The MIDI Setup dialog box is presented to you the first time you use Max. From then on, Max remembers your setup in the *Max Preferences* file in the Preferences folder.

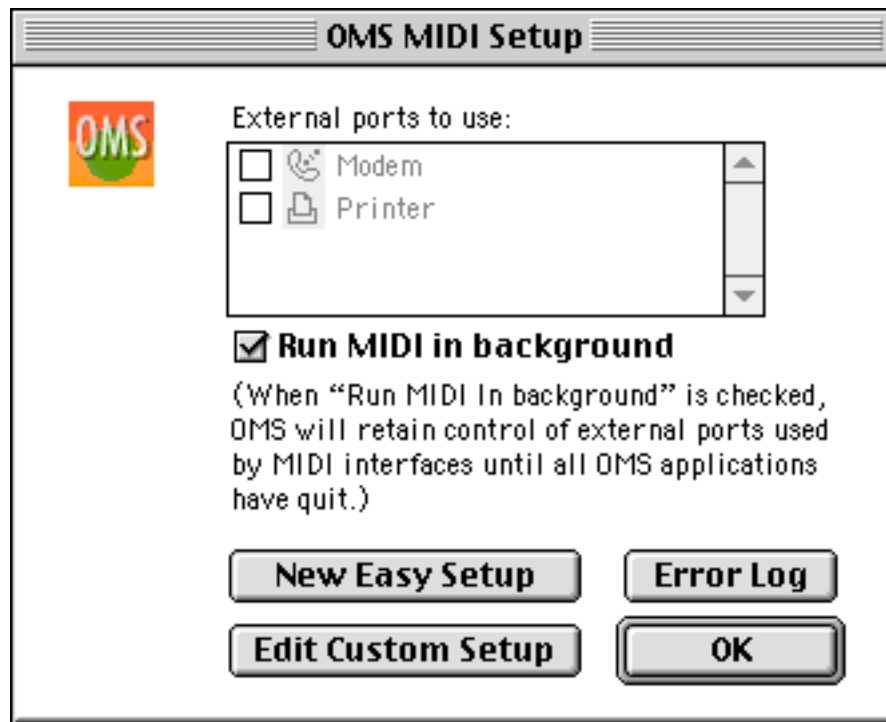
You should assign your various MIDI devices to specific ports using the Audio MIDI Setup application (Mac OS X) or OMS Setup application (Mac OS 9). The MIDI Setup dialog in Max then allows you to give each port an abbreviation (a letter a-z), and assign a range of MIDI channels to each input and output port. In the following example, port a indicates the modem port and port b indicates the printer port. Port b has been assigned a *channel offset* of 16, which means that

your Max patch can transmit MIDI to those devices either by transmitting explicitly out port b, or by transmitting on channel numbers in the range 17-32.



If you click on the *Auto Setup* button, Max will automatically assign IDs and channel offsets in a reasonable configuration.

When using OS 9, clicking on the *OMS MIDI Setup...* button brings up a dialog box that lets you select which serial ports to use for MIDI, and lets you specify how OMS should interact with other programs.



Page Setup... Specify paper size, paper orientation, and printing options before printing. If the patch you wish to print exceeds the bounds of the paper, you can reduce the image by specifying a percentage of reduction.

-
- Print . . . Print the contents of the active Patcher or Text window.
 - Install . . . Install Max external objects. When the Max application is opened, it installs external objects in its *Startup Directory*. Other external objects can be installed later with **Install. . .** However, if an external object is in Max's search path, you can just type its name into an object box to install it.
 - Quit Quit Max. If you have any unsaved windows, you'll be asked if you wish to save changes to the windows.

Edit Menu

- Undo Undo the most recent editing change in an unlocked Patcher or Text window. Editing changes that can be undone in a Patcher window are **Undo, Cut, Copy, Clear, Paste Replace, Fix Width, Align, Hide on Lock, Show on Lock, Bring to Front, Send to Back, Ignore Click, Respond to Click, Include in Background, Exclude from Background**, changing an object's font and size, and moving an object.
- Cut In a unlocked Patcher window, table editing window, or Timeline window, **Cut** removes the selection and copies it to the clipboard

When **Cut** is chosen when the Max window is active, the text contents of the Max window is erased and text is copied to the clipboard.
- Copy In a unlocked Patcher window, table editing window, or Timeline window, **Copy** copies the selection to the clipboard without deleting it.

When **Copy** is chosen when the Max window is active, the text contents of the Max window is copied to the clipboard.
- Paste In a unlocked Patcher window, table editing window, or Timeline window, paste the contents of the clipboard into the active window. When you move objects from one Patcher window to another, Max will paste the objects into the new window in the same position they had in the old window.
- Clear In a unlocked Patcher window, table editing window, or Timeline window, **Clear** deletes the selection without copying it to the clipboard. Pressing the Delete key has the same effect in most Max windows.
- Duplicate In an unlocked Patcher or Timeline window, **Duplicate** makes a copy of the selection. The duplicated objects are placed just below and to the right of the original objects, and are automatically highlighted so that they can be dragged to the desired location. After the duplicate objects have been dragged, Max takes note of their position relative to the original objects. If you then reduplicate the objects, without deselecting them, the new duplicates will be placed in the same position relative to the selection.

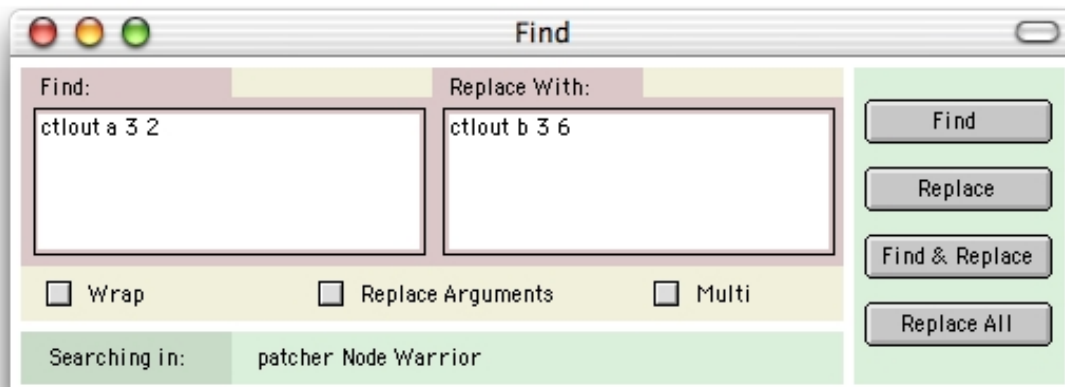
In an unlocked Patcher window, you can also duplicate quickly by option-dragging on an object, which creates a copy of the object and lets you drag it away from the original.

Select All In an unlocked Patcher window, table editing window, Text window, or Timeline window, this command selects all objects, values, or text that are not hidden (i.e., It does not select foreground boxes if the foreground is hidden. It does not select background boxes if the background is hidden. It does not select patch cords if connections are hidden). In an unlocked Patcher window, **Select All** works differently from selecting all the objects by dragging around them with the mouse, which selects the objects but not the patch cords. The distinction is important if you want to hide all the selected objects and patch cords by choosing **Hide On Lock** from the Object menu. You can select everything with the **Select All** command, then exclude individual items from the selection by holding down the Shift key and clicking on them.

Paste Picture Paste a graphic image into a Patcher window from the clipboard. Graphics can be made functional by placing the transparent button object, **ubutton**, over the image. If you have copied objects in a Patcher window to the clipboard, **Paste Picture** pastes an image of the objects, rather than the objects themselves. **Paste Picture** creates an object called **vpicture** in the Patcher window. **vpicture** has no messages, arguments, or output. For more flexible picture handling, save the picture as a file and load it into the **fpic** object.

Paste Replace Replace the currently selected objects in an unlocked Patcher with the object on the clipboard. This command is only enabled if you have copied a single item to the clipboard. It can be very useful for updating the settings of a group of objects.

Find... Open a window for finding a string of characters in the active window. **Find...** searches for a certain word or words and optionally replaces them with other words. This is especially useful in either Text windows or Patcher windows that contain many instances of the same word or words. In Patcher windows, Find operates on object boxes, message boxes, and comments.



The Find dialog has two main text entry fields, one for the text you would like to search for, and one for the text you would like to replace it with. The checkboxes operate as follows:

If the *Wrap* option is checked, Max returns to the beginning of the Text or Patcher window after it reaches the end, and searches until it is back where it started.

When the *Replace Arguments* option is checked, and the search finds an object, message, or comment box that begins with the text in the Find text field, the Replace string replaces the entire contents of the text object. When *Replace Arguments* is not checked, the replace string will replace only the number of arguments in the replace string itself, leaving the rest unchanged. The *Replace Arguments* option has no effect in a Text window.

When the *Multi* option is checked, Max searches through all open Patcher windows after exhausting the search in the Patcher window where you started your search.

The *Find* button performs a search on the text in the Find field, but merely selects the text found, if any.

The *Replace* button replaces the selected text with the text in the Replace text field.

The *Find & Replace* button performs a search on the text in the Find field, and, if found, replaces it with the text in the Replace field.

The *Replace All* button replaces all instances of the text in the Find field with the text in the Replace field.

Find Next	Repeat the most recent search performed with Find... , finding the next instance of the search string.
Replace	Replace the selected text with the replace string from the Find... dialog box
Replace and Find	Replace the selected text with the text in the Replace field of the Find window, then perform another search for the text in the Find field of the Find window.
Replace All	Replace all instances of the text in the Find field of the Find window with the text in the Replace field of the Find window.
Resume	It is possible to get Max working so hard and fast that it doesn't have time to respond to your commands. For example, if you have a number of timing objects such as metro sending out messages as fast as they can, or if you have created a loop with little or no delay between messages, Max may be too busy to pay attention to you. Holding down the command key and typing a period (<i>z</i> - <i>.</i>) stops Max's scheduler, giving you time to turn off some of the overloading processes. Choose Resume to restart Max's scheduler.

If you have a bug in your program that causes a Stack Overflow error, such as connecting the outlet of an object in such a way that it indirectly sends input back into its own triggering inlet, Max will stop its scheduler and notify you of the error. After you have fixed the bug, choose **Resume** to restart Max's scheduler

View Menu

- Edit** When **Edit** is checked, the active Patcher window can be edited. When **Edit** is unchecked, the active Patcher window is *locked*, and its user interface can be operated. Locking and unlocking the active window by clicking on the Lock button in the window's title bar has the same effect as choosing **Edit** from the View menu. Yet another way to lock and unlock the active window is to hold down the Command key (on the Mac OS) or Control key (on other systems) while clicking in white space within the window.
- Hide Object Palette** Hide the palette of objects at the top of an unlocked patcher window. When the object palette is hidden, this menu item changes to Show Object Palette, which when chosen, makes the object palette reappear.
- Hide Connections** Hide all patch cords in an unlocked patcher window. **Hide Connections** also unselects any selected patch cords so that they can not be deleted while connections are hidden. When all patch cords are hidden, this menu item changes to Show Connections, which when chosen, makes all patch cords reappear. Since any patch cords that are not marked as hidden reappear when you lock the patcher, the purpose of this command is to move patch cords out of the way for precise editing or positioning of objects.
- Hide Foreground** Hide objects that belong to the foreground layer in a Patcher window. All objects in a Patcher are by default in the foreground layer unless they are included in the background by choosing **Include in Background** from the Object menu. This command hides only objects, not patch cords. It is intended to make it easier to edit or position objects in the background layer. When foreground objects are hidden, this menu command changes to Show Foreground, which when chosen, makes the foreground objects reappear. Note that if hidden, foreground objects will not reappear when the Patcher window is locked.
- Hide Background** Hide objects that belong to the background layer in a Patcher window. All objects in a Patcher are by default in the foreground layer unless they are included in the background by choosing **Include in Background** from the Object menu. This command hides only objects, not patch cords. It is intended to make it easier to edit or position objects in the foreground layer. When background objects are hidden, this menu command changes to Show Background, which when chosen, makes the background objects reappear. Note that if hidden, background objects will not reappear when the Patcher window is locked
- Lock Background** Make it impossible to select or edit objects belonging to the background layer in an unlocked Patcher window. This is useful for editing over an image in the back-

ground while viewing the image. After choosing **Lock Background**, the menu item is checked. Choosing **Lock Background** when the item is checked unlocks the background layer and makes the check mark disappear from the menu item.

Set Origin This item is enabled if you have scrolled a Patcher window so that its original top-left corner is no longer in the same location (it might be hidden or somewhere in the middle of the screen). Choosing **Set Origin** in an unlocked Patcher window defines the current top-left corner as the one that will appear when the patcher window is re-opened after being saved.

Restore Origins This item is enabled if an origin has been defined with **Set Origin** and you have scrolled the Patcher window away from it. **Restore Origin** scrolls the Patcher window to make the defined origin the top-left corner.

Object Menu

Fix Width Adjust the width of an object box, a **comment**, or a **message** box to accommodate all the text it contains on a single line. When **Auto Fix Width** is checked in the Options menu, Max will automatically adjust the width of object and **message** boxes each time they are edited.

Align If two or more objects are select, this command adjusts the position of all selected objects so that they are perfectly aligned vertically or horizontally. All selected objects are aligned in relation to the leftmost object in the selection.

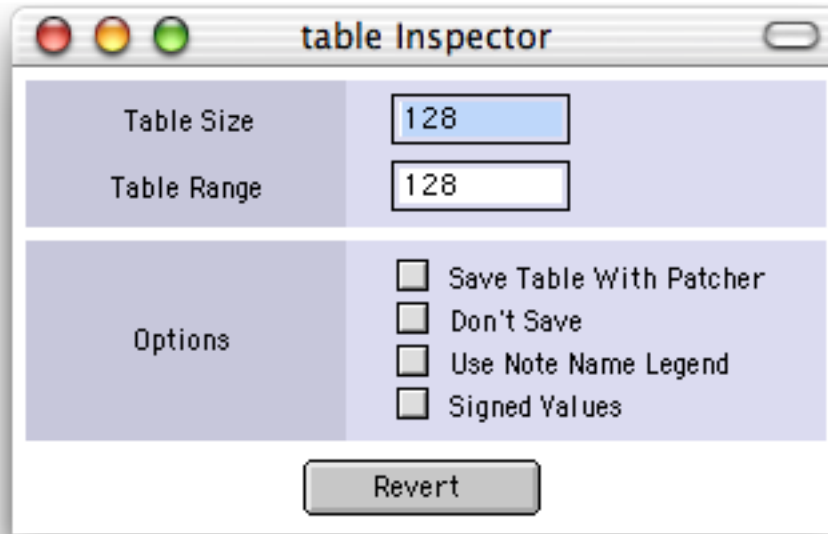
The Align feature makes an educated guess as to whether to align the selected objects vertically or horizontally. If all objects in the selection are below the top edge of the leftmost object, but not completely to the right of the right edge of the leftmost object, the objects are aligned vertically. Otherwise, they are aligned horizontally.

If a patch cord is selected, Align squares off the patch cord, attempting to avoid crossing over boxes in doing so. In certain cases, the task of avoiding boxes is too difficult and **Align** gives up without changing the patch cord's shape.

Get Info... Opens an Inspector window that lets you change special options for an object. Almost all of the user interface objects, as well as **table** objects and several other objects, have certain characteristics (such as size and range) which can be set with **Get Info....** The actual contents of the Inspector depend on the object.

Opens an Inspector window that lets you change an object's properties. Almost all of the user interface objects, as well as **table** objects, have certain characteristics

(such as size and range) which can be set with **Get Info...** The actual contents of the Inspector window will depend on the object.



Inspector window for a table

Choosing **Get Info...** in an unlocked Patcher window prints a list in the Max window of all subpatches and external objects used in that patch. You can use this to locate the external object or subpatch files.

- Color This submenu features 16 preset colors that can be applied to a number of user interface objects and patch cords. You can change the colors by choosing **Edit Colors...** in the Options menu.
- Name... Assigns a variable name for an object in a patcher window that can be used in a script. This name is not the same as a name for a **table** or **coll** object; it is local to a Patcher window and can only be used with script messages to the **thispatcher** object. For more information, see the Scripting tutorials in the **Tutorials and Topics** manual.
- Hide on Lock Make the selected objects and patch cords invisible when the Patcher window is locked, which reduces the cluttered appearance in a patch. Note that user interface objects that are hidden will not respond to mouse clicks.

Select MIDI Channel



Some objects and cords hidden

Select MIDI Channel



Nothing hidden

Show on Lock	Set the selected objects and patch cords to be visible when the Patcher window is locked, undoing the action of Hide On Lock .
Bring to Front	Bring the selected objects to the front of their current layer within the Patcher window. If a user interface object such as a dial or a ubutton is partially obscured by another object (or a picture) on top of it, Bring to Front puts it in the foreground and allows it to respond normally to mouse clicks. When a new object is created, it is automatically made the frontmost object
Send to Back	Send the selected objects to the back of their current layer of the Patcher window, behind all other objects. If an object sent to the back was covering any user interface objects, they will now respond normally to mouse clicks.
Ignore Click	Cause the selected user interface object(s) not to respond to mouse clicks when the Patcher is locked. This attribute stays with any copies subsequently made of the object with the Cut , Copy , or Duplicate commands in the File menu. If an object is set to Ignore Click , and there is another user interface object behind that one, the background object will receive the mouse click.
Respond to Click	Undo the effect of a prior Ignore Click command, restoring the selected user interface object to its normal way of responding to mouse clicks.
Include in Background	Places the selected objects in the background layer. Objects in the background layer appear behind objects in the foreground layer, even those on which the Send to Back command has been used. In addition, the background layer can be locked so that foreground objects can be easily edited on top of background objects.
Remove from Background	Removes the selected objects from the background layer

Font Menu

Each Patcher or Text window, and each object that displays text in a Patcher window, can be assigned a specific font and font size.

To change the font of an object, select it, then choose the desired font characteristics from the Font menu. When no objects are selected, choosing a font or font size will set the default font characteristics for new objects created in the active window. In a locked Patcher window, you can make a textedit object active and change its font with this menu.

Choosing a font or font size when the Max window is active changes the font used in the Max window and also sets the default font characteristics for any new Patcher or Text window created from then on, and stores those settings in the *Max Preferences* file.

9-36 Specify the font size. Font sizes available on your system for the currently chosen font will be shown in outline.

(Fonts) The fonts shown in the menu depend on the fonts available on your system.

Options Menu

Settings for all the commands in the Options menu are stored in the *Max Preferences* file, and are recalled each time the Max application is opened.

Overdrive When **Overdrive** is enabled, Max gives priority to timing and MIDI processing over screen drawing and user interface tasks such as responding to mouse clicks. This will often yield more accurate event timing, but may preempt screen drawing tasks. (Note: The **Overdrive** command and the **Enable** command in the Trace menu are mutually exclusive.)

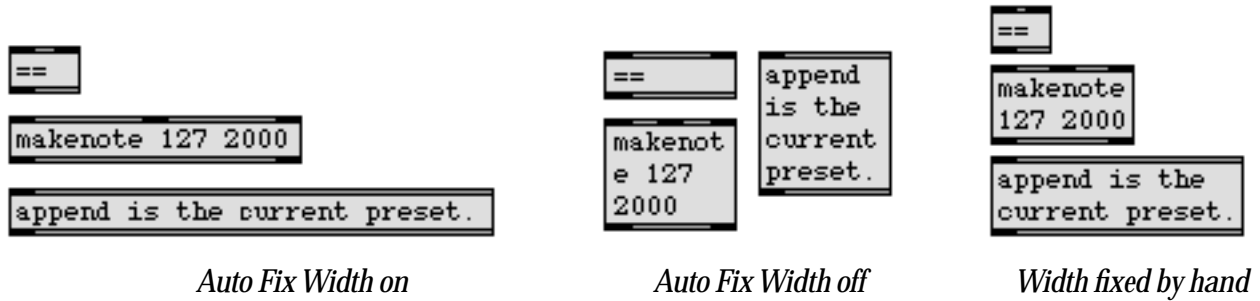
All Windows Active Allow any window to respond to the mouse, without first being brought to the front. Normally (when **All Windows Active** is not enabled), only objects in the frontmost window can respond to mouse clicks, and clicking on a window in the background brings it to the foreground. With **All Windows Active** checked, objects in a background window will respond to the mouse, but you must click in the title bar of a background window (or choose its name from the Windows menu) to bring it to the foreground.

Regardless of the setting of **All Windows Active**, you can only type numbers into a **number box** if it is in the foreground window. Keystrokes on the computer keyboard will be sent out of the outlets of **key** and **keyup** objects in any loaded patch or subpatch, regardless of the setting of the **All Windows Active** option.

New Object List When **New Object List** is enabled, Max shows a complete list of available objects whenever you create a new object box. To override seeing the New Object List window temporarily, hold down the Option key when placing a new object box in the Patcher window. See the *Objects* chapter of this manual for details about the New Object List window.

Auto Fix Width When **Auto Fix Width** is enabled, the width of an object or message box is automatically adjusted each time you edit the text inside it. The box is changed so that all the text fits on a single line. You can do this to any selected text box manually by choosing **Fix Width** from the Object menu. Some examples of **Fix Width**

are shown below.



Segmented Patch Cords

When **Segmented Patch Cords** is *not* checked, you can draw a patch cord between two objects simply by dragging with the mouse from the outlet of one object to the inlet of another object.

The procedure for drawing patch cords differs slightly when **Segmented Patch Cords** is checked. First you click on the outlet of one object, then click at each of the points where you want the patch cord to bend, then click on the inlet of the other object. The shape of a segmented patch cord can be adjusted afterward by dragging on one of the segments with the mouse.



If you make a mistake while drawing a segmented patch cord, you can get rid of the most recent segment by Option-clicking with the mouse. Or you can get rid of the whole patch cord by Command-clicking anywhere.

Although segmented patch cords redraw a little more slowly when selected or deselected, they function exactly like straight patch cords, and do not take up additional memory when the patch is loaded.

To make a segmented patch cord when **Segmented Patch Cords** is not checked, hold the shift key down when clicking on an outlet. Similarly, to make a straight patch cord when **Segmented Patch Cords** is checked, hold the shift key down when clicking on an outlet.

Assistance

When **Assistance** is enabled, the Patcher window gives you a running commentary about the function of each object's inlets and outlets when the mouse moves

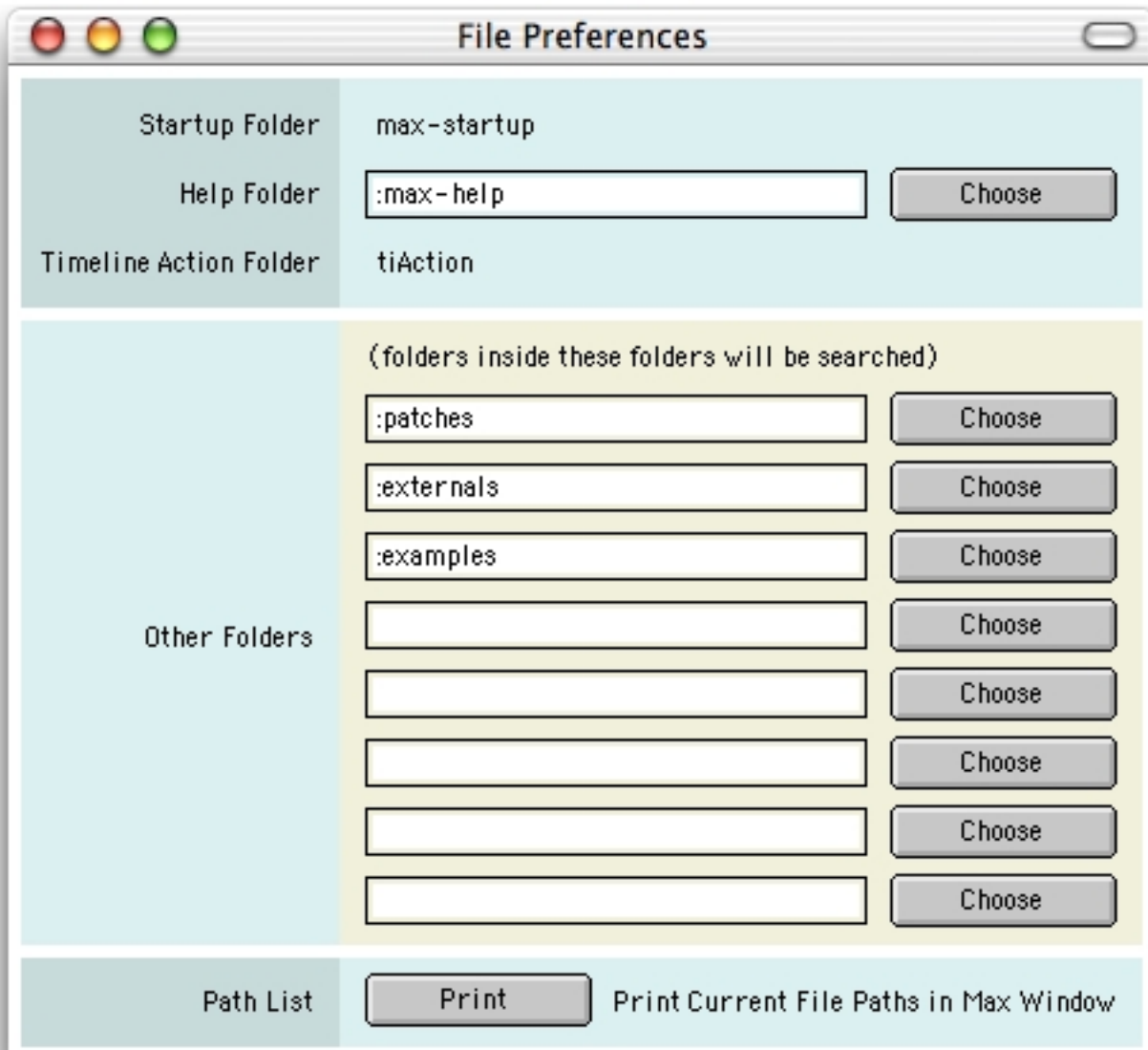
over an inlet or outlet. The following example shows **Assistance** for the right inlet of the **accum** object.



When you edit a patch to be used as an object inside other patches, you can type in your own Assistance messages for your object's inlets and outlets. Select an **inlet** or **outlet** object in your patch, and choose **Get Info...** from the Object menu. You will be presented with a dialog box which lets you type in a description of that **inlet** or **outlet**. When you use your object inside another patch, the Assistance messages will appear when the mouse is placed on the object box's inlets or outlets.

- Float Display Correction The way a computer stores floating-point numbers means that not all numbers can be represented. This means that you will often see values such as 2.3 displayed as 2.29999. When **Float Display Correction** is checked, Max rounds numbers and truncates them at what it guesses is the least significant digit.
- Max Window at Startup Show the Max window when the Max application is opened. Unchecking this option keeps it hidden until you explicitly open it by choosing **Max** from the Windows menu.
- Enhanced File Preview When checked, this option asks QuickTime to preview all files in the standard open file dialog, even those for which preview information does not exist. This may mean significant delays when creating previews for large graphics files. Uncheck this option if you are working with many large graphics files that do not contain preview information
- DSP Status... If MSP is installed, opens the DSP status window for changing Max/MSP audio settings. See the Audio Input and Output chapter of the MSP **Getting Started** manual for more information.

File Preferences Opens a window for setting the Max *file search path*: the folders that will be searched when Max looks for documents by name.



When the Max application is opened, it loads all external objects and patches contained in the *Startup* folder.

The *Help Folder* is where online help files for each object are stored. When you option-click on an object, Max searches the help folder for a file with that name and the .help suffix.

The eight slots listed for Other Folders comprise the root levels of the Max search path. You can either type the names of the folders you want to include, or use the Choose button to use a folder selection dialog to specify it.

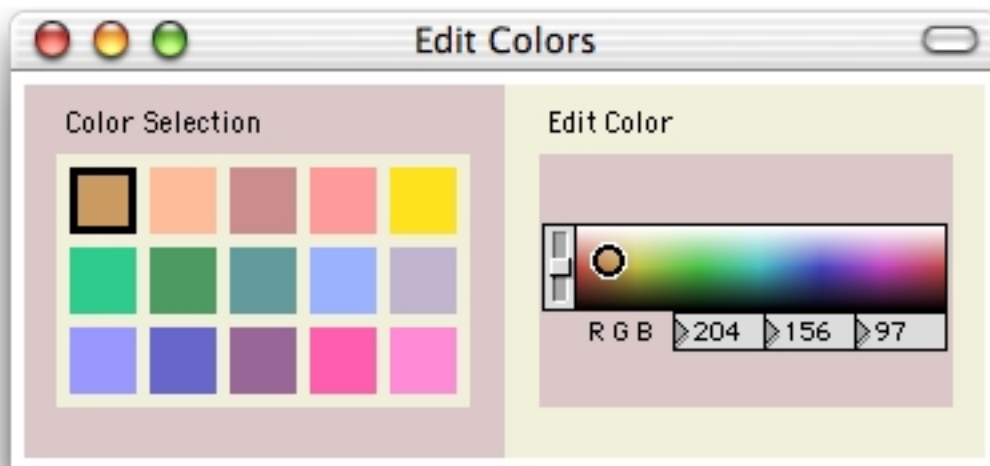
If a colon precedes the folder name listed in the File Preferences window, it is a *subfolder* of the folder in which the Max application resides. To specify a folder elsewhere, you must type in the full path name of the folder, beginning with the name of the hard disk that contains it. Use additional colons to specify folders within folders. For example, a subfolder called *InProgress* in the *MyPatches* folder shown above would be specified as `:MyPatches:InProgress`. If a name does not begin with a colon, it must be the name of a volume (a disk or disk partition), and any folders in the volume are separated by colons. For example, if you want Max to look in a folder called *New Compositions* inside the folder called *MIDI files* on the disk named *Macintosh HD*, specify it as `Macintosh HD:MIDI files:New Compositions`.

If there are any spaces in the name when you type it in, you need to enclose the entire name in single smart quotes (option-right-bracket and shift-option-right-bracket on US keyboards).

Max's *file search path* is the list of folders that will be searched when Max looks for documents. The file search path is:

1. the folder that contains the most recently loaded patch (including a patch that is in the process of being loaded),
2. the folders specified in the **File Preferences** dialog,
3. the folder that contains the Max application

Colors... You can apply color to a number of Max objects and patch cords using the Color submenu of the Object menu. The **Colors...** command opens a window that edits the palette of colors used in this menu.



To edit a color, select one of the fifteen colors on the palette. You will see the selected color displayed on the right using the color swatch, and you will see an

RGB color listing for the selected color. Change the selected color by either moving the swatch cursor around in the color space or by typing the RGB value for the color you want. Object colors in all open patchers are updated when you close the window.

Trace Menu

Enable/Disable **Enable** Trace mode, which allows you to step through every single message sent in a patch, is used for debugging purposes. If Trace mode is currently enabled, choosing this command will disable it. Choosing the **Enable** command automatically disables **Overdrive** if it is currently checked in the Options menu, before enabling the Trace mode for debugging.

Once Trace is enabled, the next message sent out an outlet will cause the patch cord through which it's travelling to blink, and information about the message will be printed in the Max window. You can then use other commands in the Trace menu to continue the debugging process. When you **Disable** a Trace in progress, Max finishes tracing the current messages, then reverts to normal operating mode.

Step Send the message that's in the blinking patch cord into its destination inlet, and advance to the resulting next message. The next message will be reported in the Max window, and the patch cord through which it is travelling will blink.

Continue Advance in normal operating mode until the next breakpoint is encountered. (See the **Set Breakpoint** command, below.)

Abort Immediately stop the execution of the patch.

Auto Step Step through the messages in the patch at a steady, moderate pace until a) the Trace is completed, b) a breakpoint is encountered, or c) a **Disable**, **Abort**, or **Auto Step** command is chosen from the Trace menu.

Set Breakpoint With the Patcher window unlocked and a patch cord selected, **Set Breakpoint** will put a stopping point at that patch cord, at which a Trace will stop when it is operating on a **Continue** or **Auto Step** command.

Clear Breakpoint With the Patcher window unlocked and a patch cord selected, **Clear Breakpoint** will remove from that patch cord any breakpoint previously set with the **Set Breakpoint** command.

Clear All Remove all breakpoints from all loaded patches.

Breakpoints

Window Menu

- Hide Subwindows Double-clicking on a **table** or **patcher** object, or on an object that is actually a subpatch, opens a *subwindow* to display the contents of that object. The **Hide Subwindows** command closes any subwindows that may be open.
- Send Window to Back Send the currently active window to the background, behind all other open windows.
- Max Bring the Max window to the front, or show it if it has been hidden.
- Show Floating Inspector Opens a floating window that changes its contents to the Inspector for the currently selected object in the active unlocked Patcher window. Since Inspector windows only operate on one object at a time, the Floating Inspector window will hide its Inspector if you select multiple objects. This feature requires Mac OS 8.6 or later. If Max is running on an earlier system, this menu item will not be present in the Windows menu.
- (other) The names of all other open windows are listed in the Windows menu. Choosing a window name brings that window to the front, making it the active window. A diamond (◊) to the left of a window's name indicates that it has been modified since it was opened, and the changes have not yet been saved.

Choosing a window's name from the Window menu while holding down the option key moves the window onto the main screen and resizes it to fit within the bounds of the main screen. You can use this if you open a file whose window is partially or completely off the screen.

Extras Menu

The Extras menu contains the names of patches that you can use for reference or utility purposes. You can add your own patches to this folder—choose **Adding Extras...** from the Extras menu for instructions. Note that patches opened using the Extras menu are brought to the front by choosing their name from the Extras menu. They are not listed in the Windows menu.

- Adding Extras... Gives you instructions for creating your own patches to the Extras menu.
- Tips Gives you a quick list of shortcuts you can use while working in a Patcher window.

In the Max/MSP installation, the following items are included in the Extras menu:

- Audiotester Lets you send a test signal out a desired channel.
- Meterin Displays an input signal level meter.
- Meterout Displays an output signal level meter.
- Quickrecord Lets you record your output to an audio file in the Max application folder.

Help Menu

Help... Choosing **Help...** when an object in an unlocked Patcher window is selected opens a help file describing that object. You can get help on any object at any time by holding down the Option key and clicking on its box (or by Option-double-clicking on its name in the New Object List window). You can also option-click on any object's box in a locked Patcher to open a help file for that object if **Help from Locked Patchers** in the Options menu is checked.

Contextual Menus in the Patcher Window

When you hold the control-key down and click inside a Patcher window, you'll get a pop-up menu that lists different items depending on what you click on. There are three basic menus: one for clicking on objects, one for clicking on patch cords, and one for clicking on blank space. In addition, control-clicking on the object box and message box icons in the Patcher window's palette display the Recent Object and Recent Message submenus described below for the Blank Space Contextual Menu.

Object Contextual Menu

This menu contains items from the Edit, Object, and Help menus. In addition, some objects, such as **bpatcher**, add items to the bottom of this menu. See the manual page from the **Max Reference Manual** or the **MSP Reference Manual** for the object you selected for a description of the items it adds to this contextual menu.

Patch Cord Contextual Menu

This menu contains items from the Object menu.

Blank Space Contextual Menu

This menu contains items from the Object menu as well as a number of items specific to this context described below.

- Object Palette This submenu displays all the icons for user interface objects in the Patcher window's object palette. Choosing one creates a new object of the type you specify.
- New Object This submenu comprises a number of submenus that create a new object box and place some text in it. The internals submenu lists all of the objects built into the Max kernel. The externals submenu shows you all of the files that can be opened by Max in your search path. The other items in the menu are files contained in the *Max* folder.

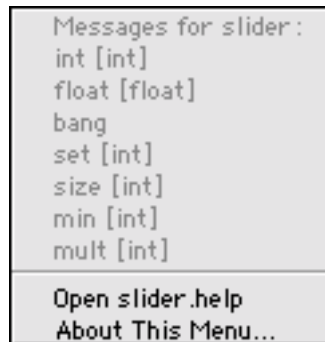
Note: a number of the files listed are not appropriate to type into an object box. However, Max does not analyze the contents of files before deciding whether to include the file's name in the New Object menu.

Note: Using the Object Palette and New Object submenu allows you to create objects in a patcher while the Object Palette is hidden (choose **Hide Object Palette** from the View menu to do this). You may find you like this method of creating objects better than using the Object Palette.

- Recent Object This menu contains the text of the most recently created object boxes, in case you would like to create another copy of one of the boxes.
- Recent Message This menu contains the first part of the text of the most recently created message boxes, in case you would like to create another copy of one of the boxes.
- Set Default Color This submenu shows the Color submenu, allowing you to choose a color other than black for newly created patch cords and objects.

Object Quick Reference Menu

Holding down the option and control keys and clicking on an object produces a pop-up menu that lists all of the messages you can send to the object along with some information about the arguments to those messages.



The message name is listed first, followed by the argument types (if any) in square brackets. If you see [int] after a message name, it means the message expects a single number as an argument. If you see [int, int, int] after a message name, it means the message expects three numbers as arguments. If you see [variable] after a message name, it means the argument can vary, and Max does not have enough information to determine the exact argument syntax required.

Beneath the message listing, there are two additional items. One opens the object's help file, and the other opens a file called boxquickref.help that explains the Object Quick Reference menu.

See Also

<code>menu</code>	Pop-up menu, to display text and send messages
<code>menubar</code>	Put up a custom menu bar
Objects	Creating a new object
Collectives	Grouping files to create a single application

Objects

Creating a new object in the patcher window

Object Palette

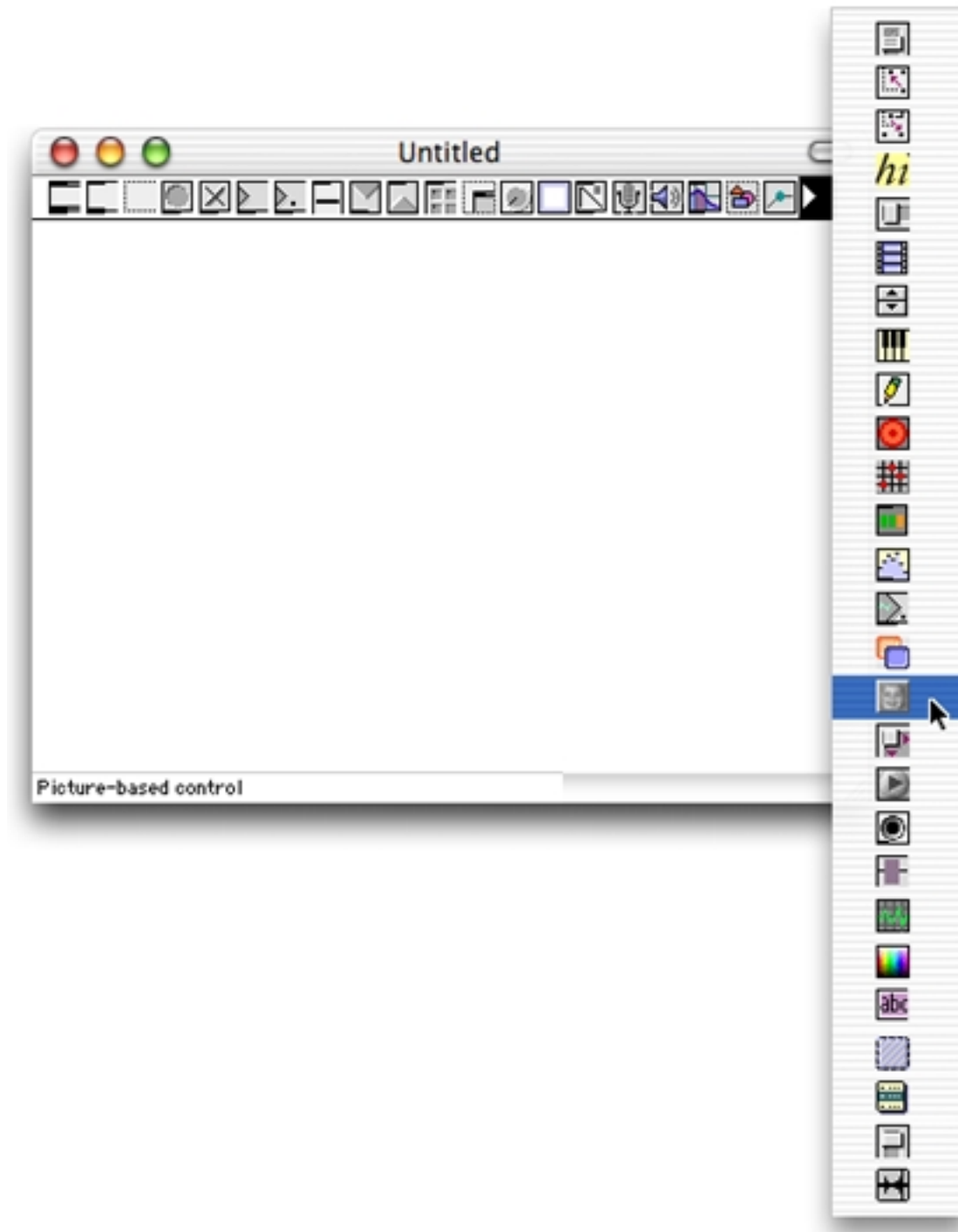
When the Patcher window is unlocked, the *object palette* appears at the top of the window. To create a new object in the Patcher window, simply click on the desired icon in the object palette (the cursor will change to that icon), then click where you want to place the object.

When the Patcher window is narrower than the width of the icons in the palette, the far right palette item in the window becomes an arrow. Holding down the mouse button on the arrow causes the rest of the object palette to be displayed as a pop-up menu. Choose the desired object from the

Objects

*Creating a new object
in the patcher window*

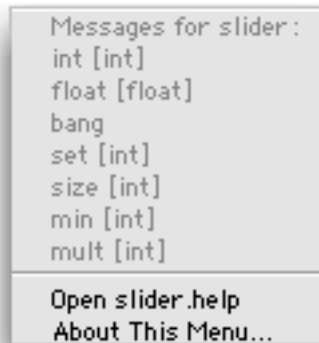
menu. Note that as you move the mouse over different objects in the palette, they are described in the Assistance area of the Patcher window.



If you choose the wrong object by mistake, you can cancel by clicking on the blank area at the far left of the palette or pressing the Delete (Backspace) key.

Object Box

A quick reference list of the messages understood by a given object can be seen by holding down the Control and Option keys while clicking on the object in a Patcher window: object.



Control-Option-Click to get a quick list of messages understood by an object

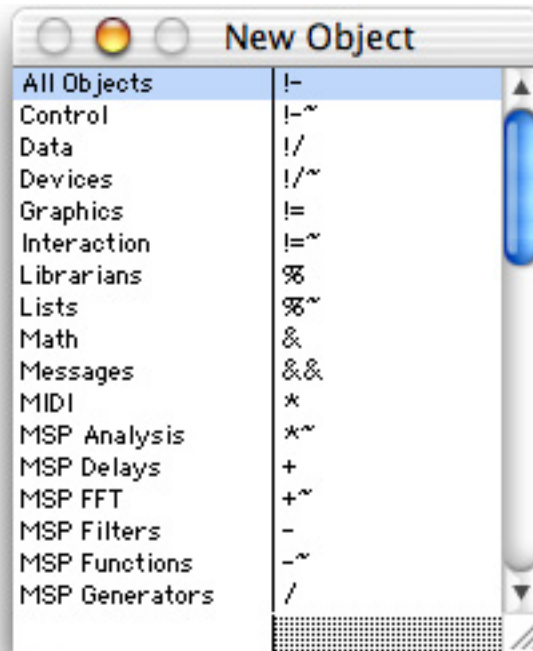
If a message requires additional arguments, the type of argument is listed after it in parentheses.

Some of the messages are shown using a terminology internal to Max. For example, in1 refers to an int in the first inlet to the right of the leftmost inlet. Furthermore, some messages shown in the quick reference message list are messages that Max sends to the object internally, *not* messages that you should send to the object in your patch. If a message appears in the quick reference message list but does not appear in the documentation, it is probably not worth worrying about.

New Object List

Because there are so many names that may be typed into an object box, you might need some help remembering all the possible choices. When **New Object List** is checked in the Options menu,

Max provides you with a complete list of all available object names each time you place a new object box in the Patcher window.



New Object List window

The New Object List window has two columns. The left column lists categories of objects, and the right section lists the names of objects in the selected category. The *All Objects* category lists all object names known to Max.

To find an object name, you can scroll through the list, type the first few letters of the desired object name, or use the up and down arrow keys on the Macintosh keyboard.

To choose an object name from the list and place it in the object box, double-click on a name, or press Return, Enter, or Space to choose the highlighted name. When an object name has been chosen, the New Object List disappears. The blinking insertion point remains inside the object box, allowing you to type in any arguments the object may require. Finally, click anywhere outside of the object box, or press the Enter key, and the new object will be created.

When you are in the New Object List window, you can get help on the selected object name by holding down the option key while you double-click, press Return, Enter, or Space.

If you want the New Object List to disappear to type in an object name yourself, press the Delete (Backspace) key, or click anywhere outside the New Object List. The blinking insertion point will be placed back inside the object box, so that you can type in the name. To create a new object without even seeing the New Object List, hold down the Option key as you place the object box in the Patcher window, or uncheck **New Object List** in the Options menu.

Max Object List

The *Max Object List* is a text file in the same folder as the Max application. By editing the Max Object List you can customize the names in the New Object List, adding your own names which you might type often. The objects you add can be external Max objects, or they can be your own patches you have created and saved as Max documents. You can even create your own categories of objects or patches.

The Max Text window is well suited for editing the Max Object List file, or you can use Apple's text editing programs TeachText or SimpleText. You can also use a word processor such as Microsoft Word, provided you save the document in plain TEXT format.

The proper format for the text in the Max Object List is as follows: on each line, type the name of the object category, a tab stop, and the name of the object. For example:

MIDI	xbendin
MIDI	xbendout
Arith/Logic/Bitwise	minimum
Arith/Logic/Bitwise	maximum
Notes	buildSeq
Dave's faves	comptroller

Note that buildSeq and comptroller are not the names an existing Max objects. They could be patches that you have written and want to use as objects. And Dave's faves is not the name of a standard Max category, but one added by the user (presumably a friend of Dave).

After editing the Max Object List file, you need to restart Max before it will change the appearance of the New Object List window.

There are no restrictions on the category name, but the object name will not create a valid object after being entered into an object box if it contains a comma or semicolon. You can add arguments after names of objects if you want them entered to an object box.

Adding categories can be useful to provide you with the names of objects you access frequently. To see all the categories, use the grow box at the bottom-right corner of the New Object List window to make the window tall enough to see all of your categories. The size of the window is saved as a preference for the next time you create an object box, and the expanded window size will be also used the next time you launch Max.

Externals

Extending the Capabilities of Max

Two Kinds of Max Objects

In various places in this manual, you may see references to *built-in* and *external* objects. The differences are minor but important. Built-in objects are part of the Max application.

External objects are separate files that contain C code that can be “linked into” the Max environment in such a way that it is impossible to tell that the code was not part of Max in the first place. That’s why we rarely mention whether any particular object is a built-in or an external in this manual.

In addition, there are two kinds of Max objects:

- *normal* objects—Max objects that appear in a Patcher window as an object box that contains its name or a character (e.g., **Borax**, **&&**)
- *user interface* objects—Max objects that appear as icons in the Patcher window palette and have some distinctive graphical appearance in a Patcher.

External user interface objects should be placed in your max-startup folder so their icons are available in the palette from the time you begin working. Normal external objects can be placed in another folder in Max’s search path and will be loaded as they are needed.

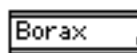
Normal external objects can provide access to non-MIDI hardware devices (such as the **cd** object that controls the Apple CD-ROM drive.), and they may also implement capabilities not found in the basic set of Max objects (such as the **coll** object that allows the storage and retrieval of arbitrary list structures indexed by numbers or symbols). External user interface objects, such as the Keyboard Slider **kslider**, are important for constructing custom interfaces within Patcher windows.

Keep External Objects Where They Can Be Found

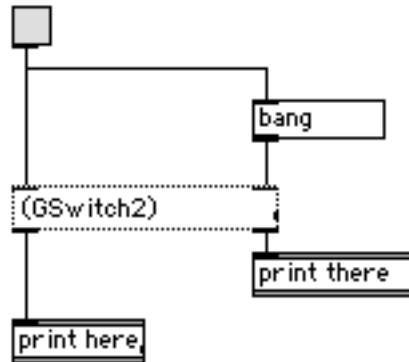
Max needs to be able to find external object files when loading a document. When an external object cannot be located, you’ll see the following error in the Max window:

• error: objectname: no such object

In the patcher you loaded, you’ll see an object box with gray lines at the top and bottom (i.e., with no inlets or outlets) with the name of the object in it if it’s not a user interface object.



If you load a patch that includes a user interface object that cannot be found, you'll see a gray-out-line box that contains the name of the object in parentheses.



If you load a patch that contains references to missing external user interface objects, the wiring of the patch may get messed up when the patch is opened. Trying to load a patch full of missing user interface objects that also contains embedded **preset** objects could result in a large stream of unhappy error messages in the Max window.

In order for Max to find external objects automatically, they must be located in a folder specified in the File Preferences window. To add folders, choose **File Preferences...** from the Options menu.

You can manually load external objects into the Max environment by double-clicking on them or choosing **Install...** from the File menu and using the standard Open File dialog to locate the external object file you want to install.

If you are in the process of developing an external object, you cannot use **Install...** to add a “revised” copy of an external object into the environment—Max will continue to use the first object definition until you restart the application.

If you expect an external object to be loaded automatically only when it is referenced (by loading a Max document that uses it, or typing its name into an object box), the name of the external object file must be consistent with the name of the object, even though the name of the object is derived from the name of a resource inside the file.

Object Mappings

Some Max/MSP objects have names that contain characters that are reserved or have other meanings for certain OS file systems. Starting with version 4.2 of Max, these objects have file names that differ from the object names that you would type into an object box—for example, the external filename of the `! /` object is actually `rdiv`. When you type `! /` into an object box, Max uses a text file called `max-objectmappings.txt` that is loaded when you start Max to associate what you've typed with the name of the external object. The `max-objectmappings.txt` contains a complete listing of file mappings. This and other object-mapping files can be found in `/Library/Application Support/Cycling '74/init`.

External Objects and Collectives

When you choose the menu item **Save As Collective...** from the File menu to create a collective file, the file you produce will contain the patch and all subpatcher files and external objects it uses.

In certain cases, not all external objects will be included and you may see “no such object” errors when you load your collective. This means that the standard search path containing your externals hasn’t been specified—for example, a patch that uses other patcher files that are loaded dynamically with the **pcontrol** object will need to have those files included explicitly in your collective script using the **patcher** keyword. For more information on making collectives, see the Collectives topic in the Max Tutorials and Topics manual.

Errors When Loading External Objects

When there is an error loading an external object, Max will act the same way as if the object could not be found. The name of the object will appear in an object box with gray lines at the top and bottom (for a normal object), or within a gray rectangle with the name in parentheses (for a user interface object).

Some external objects may require libraries from Apple or third-party developers—MSP external objects require a library called **MaxAudioLib** built into the Max/MSP and Max/MSP Runtime applications. If you see an error message in the Max window such as

- error: can't fragload objectname: missing LibraryName (function name)

you’ll need to either obtain or enable the specified shared library in order to use the external object(s). Doing this isn’t always a straightforward procedure, since some objects may require a library called **AppearanceLib** which is built into the OS and may not be available for some earlier OS versions—updating your System to a newer version may be the only solution.

Another error occurs when you try to use an object that was created for an older version of Max and has not been updated to work on the PowerPC. In this case, you’ll see the following error in the Max window:

- error: objectname: not available for PowerPC

The only solution is for you to attempt to obtain a new version of the object from the developer, or use an older version of Max that has a 68K version. Versions 3.6.2 and earlier of Max come with 68K versions, however, none of the external objects included with Max 4 or later work with these versions.

External Objects from Third Parties

In addition to the objects that come with Max itself, there are hundreds of external Max objects that are available from third-party developers. Some are included in commercial packages, but most are free and available for you to include in non-commercial applications. You’ll find many interesting collections of objects that perform everything from random number generation to

video gesture recognition. We maintain a web page devoted to listing sources for external objects at

<http://www.cycling74.com/community/>.

Note: Before releasing a collective or application built with Max that uses a third-party external object, you should check to make sure that you are permitted to distribute it. If there is no license or use information accompanying the object, we strongly recommend that you check with the author of the object. Max objects released under the GPL (GNU Public License) cannot be included in commercial software.

If you experience problems or crashes when using a third-party external object, you should contact the author of the object. While Cycling '74 makes an effort to assist developers in creating external objects, we do not provide support for them.

Locked Patcher Window

- If **Help from Locked Patchers** in the Options menu is checked, option-clicking on any object's box opens a help file for that object.
- Command-clicking in any white space unlocks the Patcher window (if it's editable).
- Option-clicking on a window's close box closes all windows except the Max window.
- Option-clicking on the title bar of a subpatch window pops up a menu that allows you to bring to the front any parent windows of the subpatcher. If the subpatcher is an "edit-only" Patcher window (produced by double-clicking on a Patcher object which was read from a Max document), the top item in the list opens the Max document for editing.
- Typing command-period (.) stops the Max scheduler, allowing you to recover from a runaway process which might be taking up too much CPU time to stop by normal methods. After you have remedied the situation which caused the process to get out of hand, choose **Resume** from the Edit menu to restart the scheduler.

Holding down both the Command and Shift while a patch is loading prevents **loadbang** objects in that patch from sending any output.

Unlocked Patcher Window

Contextual Menus

Control-clicking in a Patcher window brings up a menu of useful editing commands. For more information about the Patcher window contextual menus, see the Menus topic.

Control-Option-clicking on any object displays a menu of all the messages you can send to an object. For more information on this menu, see the Menus topic.

Selecting and Moving Objects

- Option-clicking on any object's box opens a help file for that object.
- Shift-clicking on an object box reverses the selected state of the object without changing the selected state of other objects.
- Shift-dragging an object box helps constrain the dragging of the object in the horizontal or vertical dimension.
- Option-clicking on one or more objects and dragging will duplicate the object(s).

- Hold down the shift key when clicking to place an object. Except for the object box, the cursor will remain the same, so you can make multiple copies of the same object.
- The arrow keys move the selected boxes by 1 pixel in any direction.
- Option-dragging a rectangle around a set of objects and patch cords selects both the patch cords and the objects.
- To drag a text object which has been selected for editing, move the cursor to the top or bottom edge of the box, then drag, as shown below. This is a handy way to move an object after duplicating it.



Click at the top or bottom edge of a selected text box to drag it

- Command-clicking on any user interface object, such as a **slider** or **number box**, operates the object as if the Patcher window were locked.
- Command double-clicking edits objects such as **patcher**, **table**, and **coll** that open when you double-click them in a locked Patcher window.
- Command-clicking in any white space locks or unlocks the Patcher window.
- With no objects selected, holding down the Option key and choosing **Send to Back** from the Object menu sends all **comment** objects to the background so they will not appear in front of other objects when the Patcher is locked. This may be helpful for updating files from previous versions of Max in which **comment** objects appear in front of other objects. (This shortcut does not work with the command key equivalent Option-z -B.)
- After typing into an object or message box, option-clicking outside the box prevents Auto Fix Width from changing the box's size. Option-clicking outside a comment box invokes Auto Fix Width on the comment, where it is normally disabled.

Patch Cord Shortcuts

- If Segmented Patch Cords is not checked in the Options menu, shift-clicking on an outlet of an object uses **Segmented Patch Cords** mode, in which subsequent clicks define the “corners” of the patch cord.
- Shift-clicking on an inlet of an object when making a connection lets you make multiple connections from a single outlet; another patch cord will be created immediately for the next connection.
- If **Segmented Patch Cords** is checked in the Options menu, shift-clicking uses the normal mode of dragging from outlet to inlet to make a straight patch cord.

-
- Hold down the shift key if you are making a segmented patch cord and want to make a corner over an object—the normal auto-connection feature will be disabled.
 - Command-clicking while making a segmented patch cord gets rid of the patch cord and cancels the operation.
 - Option-clicking while making a segmented patch cord gets rid of the last segment.

Creating Objects

- Option-clicking after selecting the Object Box tool in the palette places the object box without showing the New Object List window.
- Pressing the Delete (Backspace) key after selecting a tool from the palette cancels the operation and returns to the normal cursor.
- Clicking on the white area at the far left of the palette also cancels the selected palette tool.

New Object List

- Option-clicking on an empty object box opens the New Object List window.
- Delete (Backspace) hides the New Object List window.
- The Space bar enters the text of the selected item in the New Object List into the object box and adds a space afterwards for typing in any arguments.
- Return or Enter enters the text of the selected item in the New Object List into the object box.
- The Up and Down Arrow Keys scroll the selected item up and down.
- Tab switches which column of items is affected by typing.
- Holding the option key down while double-clicking or typing Return, Space, or Enter opens a help file on the selected item.

send, receive, and value

- Double-clicking on a **send**, **receive**, or **value** object brings to the front another window which contains a corresponding **send**, **receive**, or **value** object (i.e., one which uses the same name). Double-clicking on the **send**, **receive**, or **value** object in the newly selected window selects another window which contains a corresponding object. Continued double-clicking visits all such windows in a cyclic fashion.

Table Editing Window

- Command clicking with the *Pencil* tool magnifies the area around where you clicked. If you are at 8:1 x 8:1 zoom, command-clicking returns to 1:1 x 1:1 magnification, or the minimum allowed zoom above 1:1.

Any Window

- Command-clicking in any window while **All Windows Active** is enabled brings that window to the front (if the window is not already in front).

Option-clicking on the close box of a window closes all windows except the Max window.

Inspectors

- The Cut, Copy, and Paste keyboard shortcuts (Cmd-X, Cmd-C, and Cmd-V) work in the text fields of any object's Inspector window.

MIDI Messages

A MIDI message is composed of a *status* byte, which identifies the message type, followed in most cases by one or more *data bytes*. The most significant bit of data bytes is always 0, to distinguish them from status bytes, whose most significant bit is 1. Except for System Exclusive messages, the status byte specifies exactly how many data bytes will follow it. System Exclusive messages are used for synthesizer patch dumps and parameter changes.

MIDI Objects

Max MIDI objects extract the essential data from incoming messages, so you don't need to know the details of the structure of MIDI messages. These objects are listed in the table below describing each type of MIDI message.

Raw MIDI

If you wish to deal with receiving and transmitting entire MIDI messages yourself, you can use the `midii` and `midio` objects. The `midiparse` and `midiformat` objects filter and format raw MIDI, and `sxformat` can help in formatting system exclusive commands for transmission by `midio`.

To help you in managing MIDI data in Max, we have provided two reference charts here. The first chart identifies the different types of MIDI messages and shows their format. The second chart identifies controller numbers (the second byte of a MIDI control change message) that have been assigned a specific function.

MIDI Messages

Channel Messages						
Channel Messages use the lower 4 bits of the status byte to indicate the MIDI channel of the message. 0 is MIDI channel 1, and 15 is MIDI channel 16						
Function	Objects	Status Byte			2nd Byte (0-127)	3rd Byte (0-127)
		Decimal	Hex	Binary		
Note Off	xnotein, xnoteout	128-143	80-8F	1000xxxx	Key Number	Release Velocity
Note On	notein, noteout	144-159	90-9F	1001xxxx	Key Number	Velocity
Poly Pressure	polyin, polyout	160-175	A0-AF	1010xxxx	Key Number	Aftertouch
Control Change	ctlin, ctlout	176-191	B0-BF	1011xxxx	Controller Number	Controller Data

Channel Messages						
Channel Messages use the lower 4 bits of the status byte to indicate the MIDI channel of the message. 0 is MIDI channel 1, and 15 is MIDI channel 16						
Function	Objects	Status Byte			2nd Byte (0-127)	3rd Byte (0-127)
		Decimal	Hex	Binary		
Program Change	pgmin, pgmout	192-207	C0-CF	1100xxxx	Program Number	
Aftertouch	touchin, touchout	208-223	D0-DF	1101xxxx	Aftertouch Value	
Pitch Bend	bendin, bendout	224-239	E0-EF	1110xxxx	Bend (LSB)	Bend (MSB)
System Messages						
System Exclusive	sysexin, miidiout	240	F0	11110000	Mfr. ID Number	Arbitrary
Song Pos Ptr	miidiin, miidiout	242	F2	11110010	Position (LSB)	Position (MSB)
Song Select	miidiin, miidiout	243	F3	11110011	Song Number	
Tune Request	miidiin, miidiout	246	F6	11110110		
End of Sys Ex	sysexin, miidiout	247	F7	11110111		
Clock	rtin, miidiout	248	F8	11111000		
Start	rtin, miidiout	250	FA	11111010		
Continue	rtin, miidiout	251	FB	11111011		
Stop	rtin, miidiout	252	FC	11111100		
Active Sensing	miidiin, miidiout	254	FE	11111110		
System Reset	miidiin, miidiout	255	FF	11111111		

Control Changes

On/Off Switch Controllers	64-95	
Sustain Pedal	64	127 and 0
Function	Ctl Num	Values
Continuous Controllers (MSB)	0-31	
Modulation Wheel	1	0-127
Breath Controller	2	0-127
Foot Controller	4	0-127
Portamento Time	5	0-127
Data Entry	6	0-127
Main Volume	7	0-127
Balance	8	0-127
Pan	10	0-127
Expression	11	0-127
Extra precision for the above (LSB)	32-63	
On/Off Switch Controllers	64-95	
Sustain Pedal	64	127 and 0
Portamento On/Off	65	127 and 0
Sostenuto Pedal	66	127 and 0
Soft Pedal	67	127 and 0
Other	96-121	
Data Entry Yes (+1)	96	127
Data Entry No (-1)	97	127
Channel Mode Messages	122-127	
Local Control On/Off	122	127 and 0
All Notes Off	123	0
Omni Mode Off	124	0
Omni Mode On	125	0
Mono On	126	0-16
Poly On	127	0

Core MIDI

Using Max with Core MIDI on OS X

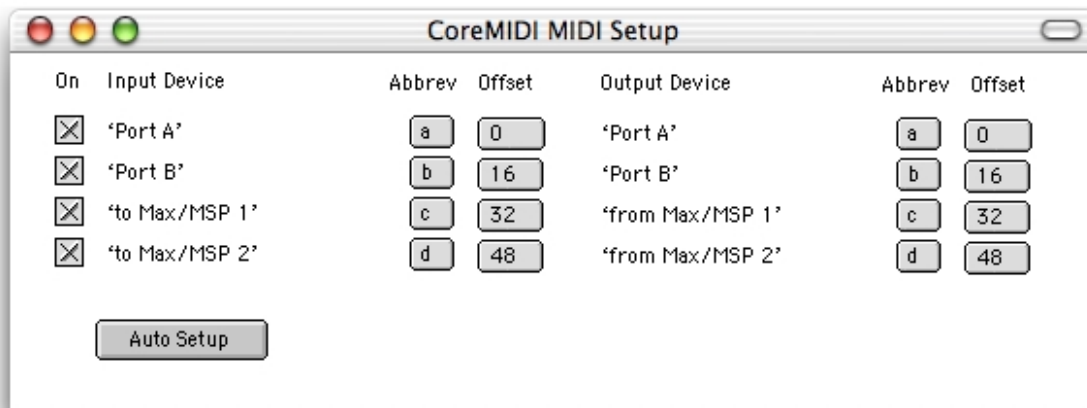
About Core MIDI

Apple's Core MIDI system is used to communicate with and to manage MIDI interfaces in OS X. It is similar to and replaces OMS (Opcode MIDI System), which was used with previous versions of the Macintosh operating system. Like OMS, Core MIDI allows you to address MIDI devices by name, in addition to port abbreviations a-z.

Core MIDI is already installed and always available on OS X. The OS X application Audio MIDI Setup, located in the Utilities folder inside the Applications folder, is used to set up and describe your MIDI setup to Core MIDI.

Using the MIDI Setup window with Core MIDI

The Max application specifies multiple MIDI ports by letter (a-z), or by using MIDI channel numbers above 16. This requires a certain amount of translation between Core MIDI device names and the Max lettering scheme. You can use the MIDI Setup window to perform this translation. To see the MIDI Setup window, choose **MIDI Setup...** from the File menu.



The MIDI Setup window lists the Core MIDI device names and lets you associate letter abbreviations and ranges of MIDI channels with them.

To assign an abbreviation and MIDI channel range to a device, select the desired values from the pop-up menus. The abbreviation allows you to type the letter instead of the device name as an argument to MIDI objects such as **notein** or **noteout**. The channel offset is added to the MIDI channel of the incoming data to distinguish its device for MIDI objects set to receive from multiple devices or sent to multiple devices. For example, if a device is given a channel offset of 32, a note-on message sent to Max will be output on channel 33 of a **notein** object. Similarly, if you send

33 to the MIDI channel inlet of a **noteout** object, a subsequent int sent to the left inlet will produce a note message on MIDI channel 1 for the device whose channel offset is 32.

Default Devices for MIDI Objects

If you create a MIDI output object without specifying a device name, the object transmits MIDI to the first device in the list of output devices in the MIDI Setup window.

A MIDI input object that is not assigned to a specific port (i.e., one that does not have a Core MIDI device name or abbreviation as an argument) will merge all input devices. In that case, the only way for a Max patch to determine which device is actually the source of input to these objects is to compare the incoming MIDI channel number to the MIDI channel offset specified for each device. Since you may wish to treat all MIDI input identically regardless of its source, this may be useful.

The **midiiin** object is an exception to this behavior—it receives data only from the first device in the input device list if no device is specified in an argument. If multiple devices share the same letter abbreviation, Max will use the first one in the list at the time a MIDI input or output object is created with that abbreviation as an argument. Changing the abbreviations of devices has no effect on pre-existing objects, although it will have an effect on the meaning of subsequent port messages sent to MIDI output objects with abbreviations as argument.

Easy Default Setup

To set all devices' abbreviations and channel offsets, click the *Auto Setup* button. Max will create a standard set of abbreviations and channel offsets for both input and output devices. The first device in each list will be given an abbreviation of a and a channel offset of 0, the second device will be given an abbreviation of b and a channel of 16, and so on.

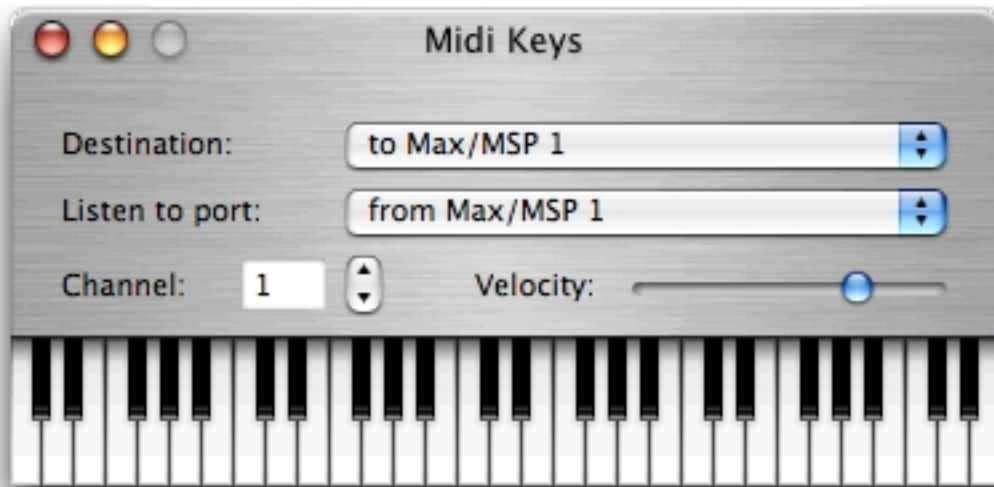
Virtual Input and Output Devices

Max has two input and output devices you can use to send MIDI to other programs running your computer, or to send MIDI to Max from other programs. The output devices are *virtual sources*.

They are labeled on the MIDI setup window as “from Max/MSP 1” and “from Max/MSP 2”. They are sources of MIDI for other programs, and can be selected as MIDI inputs in another application to establish a MIDI connection from Max.

The input devices labeled “to Max/MSP 1” and “to Max/MSP 2” are known as *virtual destinations*, because you can choose them as MIDI output devices in other applications.

Here is an example of an application—MIDIKeys—used to send MIDI to Max. The Destination is listed as to Max/MSP 1, so when you click on the keyboard, MIDI will be sent to Max's virtual destination.



Similarly, the Listen to port: pop-up menu shows the Max virtual source from Max/MSP 1. If MIDI notes were sent from Max to the from Max/MSP 1 device, they would be displayed on the Midi Keys keyboard.

See Also

Ports	How MIDI ports are specified
setclock	Control the clock speed of timing objects remotely
timein	Report time from external time code source
timeout	Send current time to other applications

About OMS

OMS provides an expanded connection between Max and MIDI devices. Without OMS, Max can only address MIDI devices directly connected to the two serial ports. OMS provides a device-independent way for Max to send and receive MIDI from any device that has an OMS driver, including multi-port MIDI interfaces, software synthesizers and hardware devices such as the Digidesign SampleCell.

Devices are addressed by name in OMS, not port letter or MIDI channel number. For example, if you have a DX7 connected to your modem port, you could have an OMS “output port” called DX7 and an OMS “input port” called DX7. OMS can also be used in conjunction with the MIDI Manager, in which case the Apple MIDI Driver is replaced with the OMS MIDI Manager Driver.

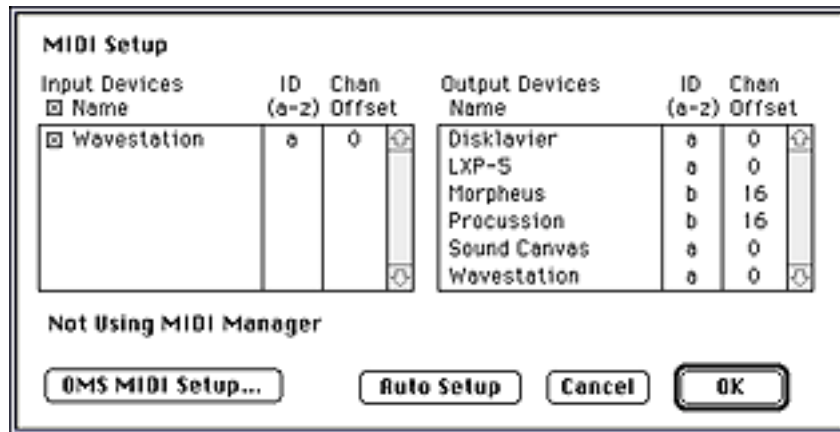
OMS consists of the *Open Music System Extension*, the *OMS Folder* containing device drivers and setup files, and the *OMS Setup* application. You install OMS on your hard disk by double-clicking on the *OMS Installer*. The OMS Setup application allows you to name the devices connected to the physical ports of your computer. Once you’ve done this, all OMS-compatible applications will refer to ports and devices by these names.

If OMS has been installed, Max will automatically use the OMS MIDI drivers instead of its own drivers. A setting shared by all OMS-compatible applications determines whether the MIDI Manager is being used or not. This setting can be changed in the OMS Setup application, or by clicking on the *OMS MIDI Setup...* button in the MIDI Setup dialog that appears when Max is running under OMS.

Using the MIDI Setup Dialog in OMS

In Max you refer to multiple MIDI ports with letters a-z or with MIDI channel numbers above 16, so a certain amount of translation is necessary between OMS device names and the Max lettering scheme. You match OMSdevice names with letter abbreviations and ranges of MIDI channels in

the MIDI Setup dialog. This dialog will appear the first time you run Max under OMS, and can also be accessed by choosing **Midi Setup...** from the File menu.



To assign an abbreviation and MIDI channel range to a device, drag on the Numericals next to each device name. The Channel Offset sets the number added to a MIDI channel for identifying an input device or specifying an output device. For example, if an output device has a Channel Offset of 32, sending the number 33 into a channel inlet of a MIDI output object will specify MIDI channel 1 on this device.

Note that output devices that are connected to the same *physical* interface will share the same abbreviation and channel offset.

Default Devices for MIDI Objects

If you create a MIDI output object without specifying a device name, the object transmits MIDI to the first device in the list of output devices in the MIDI Setup dialog (at the time the MIDI output object was created). However, a MIDI input object that is not assigned to a specific port (i.e., one that does not have an OMS device name or abbreviation as an argument) will *merge* all input devices. In that case, the only way for a Max patch to distinguish which device is actually the source of input to these objects is to compare the incoming MIDI channel number to the MIDI channel offset specified for each device. That's not necessarily a bad thing since you may wish to treat all MIDI input identically, no matter where it comes from. An exception is the `midin` object, which receives data only from the first device in the input device list if no device is specified in an argument.

If multiple devices share the same letter abbreviation, Max will use the first one in the list at the time a MIDI input or output object is created with that abbreviation as an argument. Changing the abbreviations of devices has no effect on already existing objects, although it will have an effect on the meaning of subsequent port messages sent to MIDI output objects with abbreviations as arguments.

In the MIDI Setup dialog, you can click the *Auto Setup* button to have Max arrange a standard set of abbreviations and channel offsets. Typically, this assigns the first letters of the alphabet and lower channel offsets to the first device names listed.

OMS and Other Applications

Clicking the *OMS MIDI Setup...* button opens the standard OMS MIDI Setup dialog, which allows you to select which serial ports are used for OMS and the *Compatibility Mode*, one of three possibilities:

- Use MIDI Manager
- Only OMS Applications (allows OMS applications to send and receive MIDI in the background)
- Allow non-OMS applications (relinquishes the serial port when OMS applications are in the background)

If you are using Apple's MIDI Manager, six additional input and output devices will be shown. These devices are named *MIDI Mgr Input 1* to *MIDI Mgr Input 6* and *MIDI Mgr Output 1* to *MIDI Mgr Output 6*, and correspond to Max's MIDI Manager *virtual port* connections. When using OMS, these MIDI Manager ports are available in addition to the direct connections with your MIDI interface or other devices provided through OMS. When using OMS, you will use the MIDI Manager ports for sending MIDI data between Max and other Macintosh applications.

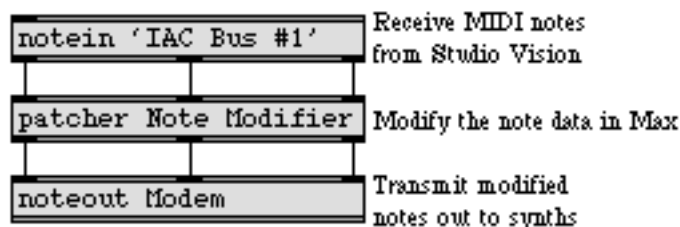
If you use the OMS MIDI Setup dialog to stop using the MIDI Manager, the Max MIDI Manager input and output ports will disappear from the list of device names in the main MIDI Setup dialog box.

For communicating between Max and other OMS-aware applications, we recommend using the IAC Driver available for OMS.

Inter-Application Communication (IAC)

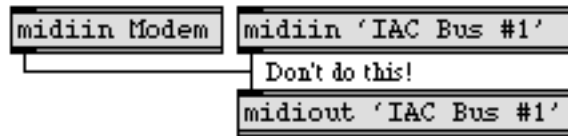
Max can transmit MIDI data to, and receive from, other OMS-aware MIDI applications via IAC busses (virtual ports) defined in the *OMS Setup* application. (See the documentation for the OMS Setup application for details.) Once an IAC bus has been named in the current OMS setup, that name can be used as a port argument in Max MIDI objects.

MIDI data transmitted to an IAC bus will go to any other application that is set to receive from that bus. Likewise, a MIDI input object in Max that you designate to receive from an IAC bus will get MIDI from any other application that is transmitting to that bus.



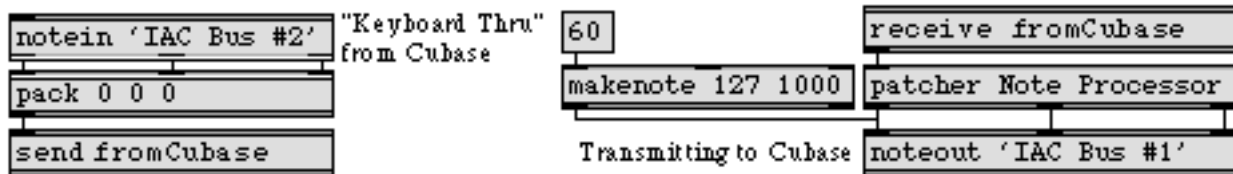
Using Max to modify the output of a sequencer

When communicating with other applications via an IAC bus, it is important to take care not to create feedback loops of MIDI data. For example, the following patch would create a continuously flowing loop of data as soon as any MIDI message is received from the modem port.



This patch will create a feedback loop of MIDI data

The flaw in the preceding example may be obvious, but similar feedback loops can occur in less obvious ways with IAC, depending on what other applications are doing. For example, if the **Keyboard Thru** option is checked in the Vision application, and its output device is set to IAC Bus #2, then the following patch would cause feedback, even though it may appear to be correct within Max.

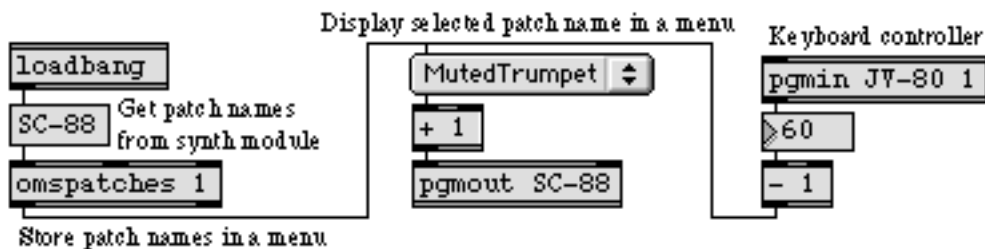


MIDI data will feed back if Vision receives on bus 1 and throughputs to bus 2

If proper care is taken to avoid such feedback loops of MIDI data, OMS MIDI applications can collaborate successfully with Max. You can use Max to process the output of other applications, to transmit MIDI messages to other applications (to be recorded in a sequencer application, for example), or to trigger processes in other applications (such as starting sequences).

OMS Names

OMS permits you to refer to virtual MIDI ports by the names of the connected MIDI devices as they appear in the current OMS setup. You can also obtain a list of patch names from devices in the current OMS setup with the `omspatches` object, which can in turn place those names in a `menu` object. The menu of patch names can then be used for selecting and displaying synth patches by name rather than by number.



Refer to ports by device name; use `omspatches` to get all patch names for a device

The patch names are actually obtained from the OMS Name Manager, which keeps track of the patch names relevant to the current OMS setup. The patch names are generally provided to OMS

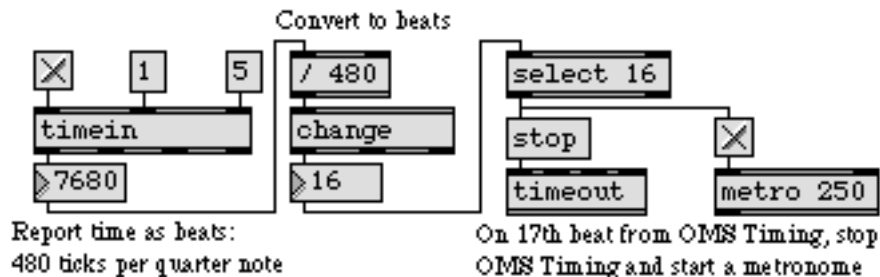
by a Name Manager compatible patch librarian such as Opcode's Galaxy application. A particular Galaxy bundle can be subscribed to by another Name Manager compatible application such as Opcode's sequencer Vision, thus informing OMS of the current patches for each device.

In a similar manner, you can obtain names of the continuous controllers for a particular device with the `omscontrollers` object, and you can obtain names of specific notes for a device (on a drum patch, for example) with `omsnotes`.

OMS Timing

The OMS Time Manager provides a clock source useful for slaving multiple applications to a single clock. Any application that supports OMS Timing can synchronize to it, and can also start or stop its clock. In Max, the `timein` object reports the current time, as obtained from OMS Timing, allowing processes in Max to be synchronized with events in another application. Max can start and stop OMS Timing itself with a start or stop message to a `timeout` object.

For example, you could begin playing a score in Opcode's Vision application. Doing so starts the clock of OMS Timing. Meanwhile, Max—running in the background—could get the elapsed time of that score from OMS Timing, reporting it out the outlet of `timein`. The elapsed time from OMS Timing could thus be used to trigger events in Max.

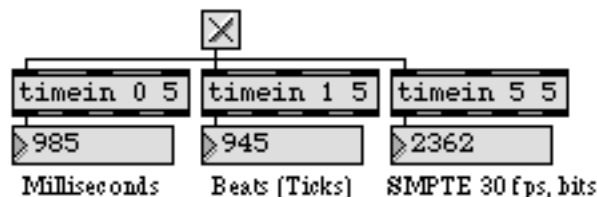


Get timing information from OMS, and use it to synchronize events with other applications

The `timein` object reports timing information from OMS Timing, and `timeout` can control OMS Timing. The `setclock` object, with the `ext` argument, will synchronize with OMS Timing, and other timing objects such as `clocker` and `metro` can in turn use that `setclock` object as the source of their timing, with a `clock` message.

The `timein`, `timeout`, and `setclock` objects can communicate with OMS Timing using various time code formats: *milliseconds*, *beats* (actually fractions of a beat called *ticks*—normally 480 ticks per quarter note), and *SMPTE code* at any of a number of different frame rates. The format used will determine the time units referred to by the objects. For example, if `timein` is reporting the time from OMS in milliseconds, one second will be represented by the number 1000. If it is reporting

time in beats, and the beat tempo is 120 MM, one second will be represented as 960. If it is reporting time at 30 fps in bits (80ths of a frame), one second will be represented as 2400, and so on.



Approx. one second, represented with three different time units obtained from OMS Timing

It is worth noting that, because OMS Timing only reports its time to Max periodically—every few milliseconds—and those reports will generally not be in perfect synchrony with the time interval at which `timein` reports the time, there will be some slight discrepancies between the mathematically correct numbers used as examples in the preceding paragraph and the numbers you will actually get from `timein`. Still, the numbers will generally be accurate enough to permit you to synchronize Max successfully with other applications that support OMS Timing.

When OMS is not installed, and the Apple MIDI driver or another application is providing timing information to Max via MIDI Manager, `timein`, `timeout`, and `setclock` synchronize with that timing information. If both OMS and MIDI Manager are present, Max uses OMS timing rather than timing from MIDI Manager.

See Also

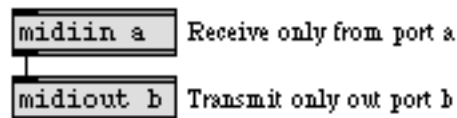
Ports	How MIDI ports are specified
<code>omscontrollers</code>	Report controller names for OMS devices
<code>omsnotes</code>	Report note names for OMS devices
<code>omspatches</code>	Report patch names for OMS devices
<code>setclock</code>	Control the clock speed of timing objects remotely
<code>timein</code>	Report time from external time code source
<code>timeout</code>	Send current time to other applications

Ports

How MIDI ports are specified

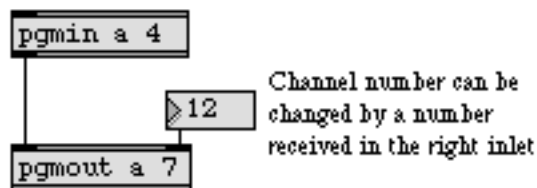
Max's MIDI Objects

Each object in Max that transmits or receives MIDI data can be set to communicate with a specific port by typing in a letter argument after the name of the object.

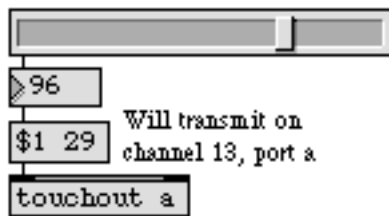


Any letter from a to z can be used, provided that letter has been assigned to a device (or a virtual port) in the MIDI Setup dialog box. Refer to the CoreMIDI section (or the OMS section if you are using Mac OS 9) for information about assigning abbreviations and MIDI channel ranges.

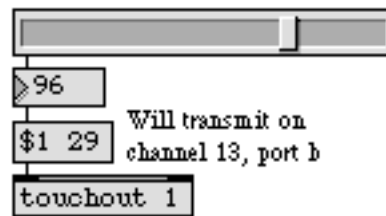
For MIDI objects that receive and transmit specific channel voice messages, such as **notein** and **bendout**, a letter argument followed by a number argument indicates a port and a specific MIDI channel on which to receive or transmit MIDI messages. For transmitting objects (except **midio**), channel numbers are received in the rightmost inlet to change the channel.



A number alone can be used in place of a letter and number combination for such objects. Numbers 1-16 specify a channel number on port a, numbers 17-32 specify channels 1-16 on port b, etc. Numbers greater than 16 can be received in the right inlet, to specify both port and channel. When a letter argument is present, however, the port is fixed, and channel numbers greater than 16 received in the right inlet are *wrapped around* to stay within the 1-16 range.



Letter argument transmits to only one port.



Otherwise, number specifies both port and channel.

The ports can also be assigned an arbitrary range of sixteen MIDI channels. So, for example, port b could be assigned channels 1-16 and port a could be assigned channels 17-32.

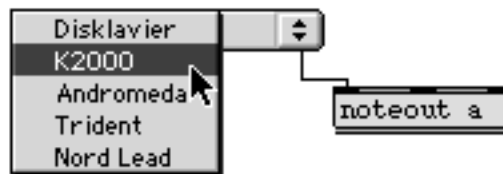
Specifying MIDI Ports

MIDI ports may be specified three ways: by device name, letter abbreviation (a to z), or MIDI channel number. Device names are used as arguments to MIDI objects in place of letters. Here are some examples. Note the use of “smart” single quotes—obtained by typing option-] and option-shift-]—around names of devices that contain spaces.

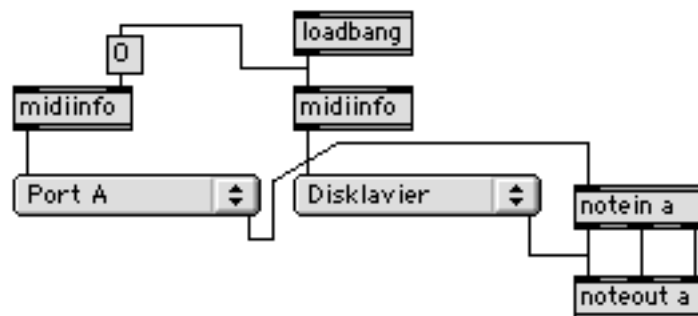


The port Message

All MIDI objects accept a message that changes the port they use to transmit or receive MIDI data. The port message takes one argument, which is the letter or (when using OMS or CoreMIDI) device name associated with the port. The word port can be omitted, which is convenient for making a pop-up menu with port names or abbreviations, as shown below.



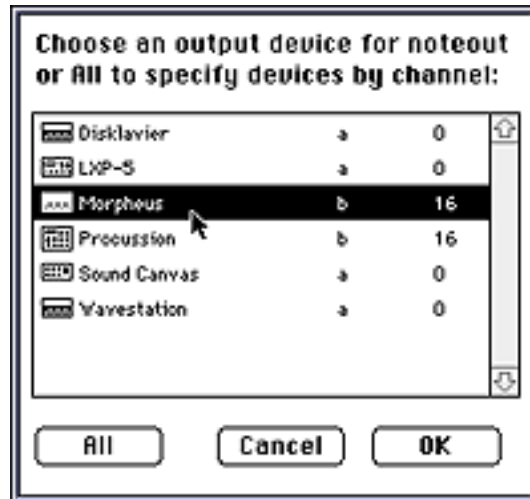
The pop-up menu can be created automatically by using the `midinfo` object.



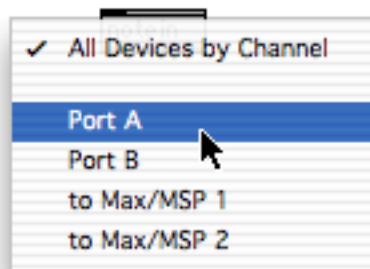
After receiving the port message, the MIDI output object acts as if the port had been specified as an argument—all MIDI channels above 16 wrap around to 1-16, and data is always sent on the specified port.

Specifying Ports

Yet another way to set the port of a MIDI object is to double-click on the object when the Patcher window is locked. On Mac OS 9 you will be shown a dialog box listing all the devices that are appropriate to assign to that object. You can select one device and click *OK*, or you can click *All* if you want to be able to identify or specify different devices by means of channel number alone.



On Mac OS X, instead of a dialog, you are presented with a menu containing available MIDI ports. You also have the option to choose *All Devices by Channel* if you want to be able to identify or specify different devices by means of channel number alone.



See Also

Core MIDI
OMS

Using Max with Core MIDI on Mac OS X
Using Max with OMS on Mac OS 9

A

- Abort 43
- active window 44
- Adding Extras... 44
- Align 35
- All Windows Active 60
- Allow non-OMS applications (OS 9) 69
- Any Window
 - shortcuts 60
- any window, clicking in 38
- argument 12
 - for specifying a MIDI channel 73
 - for specifying a port 73
- Assistance 11, 39
- Audiotester 44
- Auto Setup (Core MIDI OS X) 65
- Auto Setup (OMS OS 9) 68
- Auto Step 43

B

- background object in a Patcher 37
- background operation of Max 71
- bang 13
- bendin 24
- bent patch cords 39
- Blank Space Contextual Menu 45
- Borax 25
- bottom-to-top 14
- Bring to Front 37

C

- C programming language 3
- channel mode message
 - reference chart 62
- Clear All Breakpoints 43
- Clear Breakpoint 43
- clear the selection 31
- close the active window 28
- collective 29
- Color 36
- color of an object 36
- Color palette 42
- command-period 57
- comment 12
- configuring MIDI 6
- connecting MIDI equipment 6

- Contextual Menus 45
- Continue 43
- control change
 - reference chart 62
- controller numbers 62
- copy the selection 31
- Core MIDI 64
 - default ports for MIDI objects 65
 - device abbreviations 64
 - device names 64
 - virtual destinations 65
 - virtual sources 65
 - with other applications 65
- creating a new object 48
- cut the selection 31

D

- data byte
 - of a MIDI message 61
- decimal number 13
- dialog
 - Find... 33
 - Midi Setup... 29
- Disable Trace 43
- documents 12
- DSP Status 40
- duplicate the selection 31

E

- Edit 34
- Edit menu 31
- Enable Trace 43
- Enhanced File Preview 40
- external object 53
 - install 31, 52
- external objects
 - automatic loading 54
- Extras menu 44

F

- feedback loops of MIDI data (OS 9) 70
- File Preferences... 54
- File Preferences... 21
- file search path 21, 41
- filtering MIDI messages 24
- Find Next 33

Index

Find... 32
Fix Width 35
float 13
Float Display Correction 40
font characteristics 18, 24
Font menu 37
foreground object in a Patcher 37
fraction 13

G
Get Info... 35
grow bar 18

H
Help 27
Help Menu 45
Help... 45
hexadecimal number 14
Hide Background 34
Hide Connections 34
Hide Foreground 34
Hide Object Palette 34
Hide Subwindows 44

I
IAC (Inter-Application Communication) 69
Ignore Click 37
import a graphic 32
inlet 10
Inspector
 Edit menu commands 60
 Get Info... 35, 36
Inspectors 60
Install... 54
installing external objects 31, 52
int 13
Inter-Application Communication (IAC) 69
invisible objects and patch cords 36
IRCAM 3

L
list 13
locating offscreen windows 44
Lock Background 34
locking and unlocking a Patcher window 34

M
Max documents 12
Max Object List 52
Max Preferences 23, 38
Max window 16, 44
Max Window at Startup 40
message 12
 order of messages 14
 quick reference 50
 tracing 43
Meterin 44
Meterout 44
MIDI
 connecting MIDI equipment 6
 reference chart 61
 transmitting and receiving 24
MIDI channel
 specifying 73
MIDI Enable/Disable 24
MIDI handling in OS X 64
MIDI interface
 connection 6
MIDI message 61
MIDI objects 24, 61, 73
Midi Setup 6
MIDI Setup Dialog 7
MIDI Setup window (OS X) 64
MIDI Setup...
 when using OMS 29
Midi Setup... 29
 When Using Core MIDI 64
 When Using OMS 68
midiin 24
midiout 24
midiparse 24
Miller Puckette 3
millisecond 14

N
name of an object 11
Name... 36
New Object 45
new object 48
New Object List 18, 50, 59
 adding names to 52
notein 24

number 13

O

object box 11

Object Contextual Menu 45

Object Mappings 54

Object menu 35

object message list 50

Object Palette 45

Object Quick Reference Menu 46

objects

aligning 35

creating 48

Max Object List 52

MIDI 61, 73

name of 11

New Object List 50

width of 35, 38

offscreen windows 44

OMS

default ports for MIDI objects 68

device abbreviations 67

device names 67

IAC Driver 69

installing 67

OMS (OS 9)

with other applications 69

OMS MIDI Manager Driver 67

OMS MIDI Setup... 30

OMS MIDI Setup... 67

OMS Setup application 29, 67

OMS Setup application (OS 9) 69

OMS Time Manager 71

OMS Time Manager (OS 9) 71

OMS Timing 71

inaccuracies 72

Only OMS Applications (OS 9 MIDI setup)

69

Open as Text 28

Open Music System Extension 67

Open... 28

Options menu 38

order of messages 14

outlet 10

Overdrive 38

P

Page Setup... 30

Paste 31

Paste Picture 32

Paste Replace 32

patch 17

patch cord 11

segmented or straight 39

Patch Cord Contextual Menu 45

Patcher window 16, 28

creating 28

locking and unlocking 34

shortcuts 57

port 73

specifying in Max MIDI objects 73, 74

when not using MIDI Manager 73

Preferences... 41

Print... 31

protocol of MIDI messages 61

Q

quick reference message list 50

Quickrecord 44

QuickTime Musical Instruments (Mac OS 9)

8

quit Max 31

R

real time 3

receive

double-clicking on 59

receiving MIDI from other programs (OS X)

65

Recent Message 46

Recent Object 46

Replace 33

Replace All 33

Replace and Find 33

Respond to Click 37

Restore Origins 35

Resume 33

right-to-left 14

S

save a file as text 29

save a Max document 28

Index

Save As... 29
save file as collective 29
segmented patch cords
 cancelling 59
select all objects in the active window 32
send
 double-clicking on 59
Send to Back 37, 44
sending MIDI to other programs (OS X) 65
serial port 6
Set Breakpoint 43
Set Default Color 46
Set Origin 35
setclock 71
Show on Lock 37
simultaneous messages 14
size
 of objects, grow bar 18
stack overflow 34
Startup Directory 31
Step 43
subwindow 44
symbol 13

T

table
 entering values as text 28
Table window 28
 shortcuts 59
Text window 28
ticks 71
Time Manager (OS 9 OMS) 71
time units 71
timein 71
timeline 20
timeline editor window 28
timeout 71
timing from OMS 71
timing inaccuracies from OMS 72
Tips 44
title bar 24, 34
Trace
 Enable/Disable 43
Trace menu 43
triggering 14
truncation 13

U

undo the most recent editing change 31
Unlocked Patcher Window
 shortcuts 57
Use MIDI Manager (OS 9 MIDI setup) 69
user interface object 11

V

value
 double-clicking on 59
View menu 34
virtual destinations (OS X) 65
virtual sources (OS X) 65

W

width of an object 35, 38
Windows menu 44