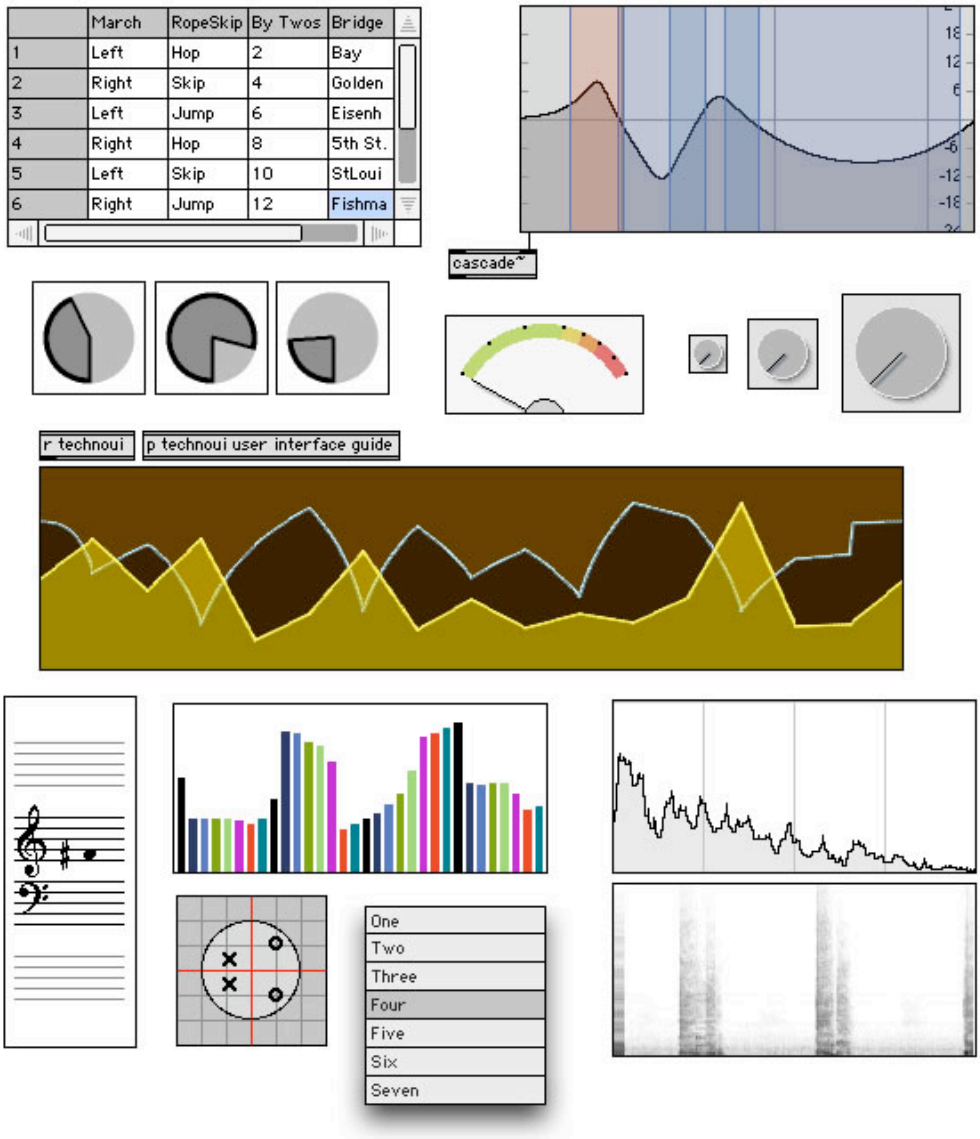
What's New in Max/MSP 4.5



Overview of New Features

New DSP functions

- Mastering quality single and multi-band dynamics objects using OctiMax technology for compression, expansion, limiting, and noise gating. Also comes with a complete dynamics tutorial to help the novice learn more about the subtle details of dynamics processing.

- Rewire slave support. Now you can run MaxMSP from inside your favorite rewire host application.

- High quality anti-aliased oscillators with **rect~**, **saw~** and **tri~** that provide a warmer, more "analogue" sound.

- Spectral domain pitch shifting with **gizmo~**, time domain frequency shifting with **freqshift~**, fft bin shifting with **fbinshift~**, hilbert transform with **hilbert~**

- Signal rate sequencing with **techno~**

- **poly~** aware envelopes for easier development of polyphonic synthesizers

User Interface Objects

- All in one spectral display with **spectroscope~**

- Edit multiple filters simultaneously using one **filtergraph**, or view and edit filter coeffecients on the z-plane with **zplane~**

- Imageburgers allow the use of images in place of objects/subpatchers as visual mnemonics

- High quality image interpolation for **fpic**, **pictctrl**, and imageburger image scaling

- Javascript driven User Interface elements, featuring:

    - 2D and 3D rendering using OpenGL
    - antialasing support
    - alpha blending support
    - library of prebuilt templates (sliders, knobs, buttons, panels, and more)
    - pattr compatible

Patcher Attributes

Patcher attributes (**pattr**) provide a more effective, efficient, and integrated means to handle state management tasks than previously possible in Max.

- assign attributes to your patcher file
- manage these attributes in presets which support patcher and preset hierarchy
- interpolate between presets in a variety of ways
- preset files are stored in human editable XML

Programming Language Support

Extending the functions of Max with text based code has never been easier, now with support for Javascript, Java, and Mach-O support for C development. Not only does this make programming easier, but non-programmer users have access to an even wider range of possibilities in Max. Several Javascript and Java examples/objects are included that non-programmers will find useful in every day Maxing.

Java

- crossplatform programming with integrated editor and compilation
- access to everything the Java API provides, including rich networking and database support, file I/O, graphics, arbitrary precision arithmetic, and more.
- automatically maps Java object methods to max object messages
- script patchers with java code
- access **buffer~** objects from within java

Javascript

- crossplatform scripting with minimal programming experience necessary
- build objects that perform complicated tasks with only a few lines of code
- script patchers with Javascript code
- build interactive user interface objects with **jsui**
- many useful examples including physical simulation, file I/O, UI widgets, list and string manipulation, dynamic arithmetic expressions

C

- Mach-O support for Macintosh. Permits external development using Project Builder and XCode

New documentation in 4.5

Max Documents

Six new tutorials are provided with version 4.5. These include:

Tutorial 48: Basic JavaScript

> A brief introduction to using JavaScript inside Max. It covers basic use of the **js** object, editing and error-correction of the code, variable scope issues and JavaScript event handling.

Tutorial 49: Scripting and Custom Methods in JavaScript

> Describes patcher scripting in JavaScript (as an alternative to the "messages to Max" system in 4.3), and the creation of custom methods for event handling.

Tutorial 50: Tasks, Arguments and Global Objects in JavaScript

> Scheduling (using Tasks), js object arguments and Global object creation are covered in depth. This tutorial provides examples that extend the **js** object's functionality for large-scale patcher development.

Tutorial 51: Designing User Interfaces in JavaScript

> The **jsui** object is examined, with a focus on creating objects that interact with the user. Using the OpenGL ("sketch") object to create compelling interface objects provides an alternative to C-code programming.

Tutorial 52: Patcher Storage (using **pattr**)

> A brief introduction to using **pattr** and **pattrhub** for complex patcher preset storage. This tutorial shows the use of the pattr system to solve most of the difficulties found in creating multi-level preset systems.

Tutorial 53: More Patcher Storage

> Advanced information is provided for the use of **pattrstorage** and **autopattr**, providing external file storage, and automatic binding of controls.

These tutorials are located in the Max 4.5 Topics and Tutorials guide (called Max45TutorialsAndTopics.pdf in the documentation distribution).

In addition to the new tutorials, there is a document entitled "JavaScript in Max" that provides programming details for the **js** and **jsui** objects. This document provides in-depth development information on the JavaScript development environment, as well as the many built-in JavaScript objects that interface with Max.

An extensive set of documentation and numerous tutorials are provided for developing Max objects with the Java programming language (using the **mxj** Max-to-Java bridge object). This material can be found in the "java-doc" folder of the Max application, and contains information on setting up your system to write, compile and execute Java code in Max. The file "mxj_getting_started" (in both text and RTF format) provides information on using the Javadocs-based documentation and the example classes.

MSP Documents

The "topics and tutorials" section has been removed from the MSP Reference manual, and is now a separate document. This includes updated information on ReWire handling and the various audio drivers.

An additional document, entitled "The OctiMax Compression Tutorial" (OctiMaxCompressionTutorial.pdf in the documentation distribution) is a compression primer, written by Peter Elsea, for use in conjunction with the **omx.*** objects. If you are not familiar with compression techniques, this document will help you get started using these high-quality compression and limiting objects.

Performance enhancements and tuning features

One of our goals for the Max/MSP 4.5 upgrade was to address system performance, particularly on Mac OS X. We identified three major areas for improvement:

- Threading performance when using Overdrive
- MIDI timing accuracy
- Screen refresh timing

For threading performance, we have optimized our interaction with the operating system and eliminated any significant difference in performance with Overdrive on or off. Jitter users in particular should see substantial improvements in frame rates with Overdrive turned on. For MIDI timing accuracy, we use MIDI time-stamping facilities available on both platforms to maintain the timing integrity of MIDI data as it flows through Max. The new **sync~** object produces an audio signal that can be driven from MIDI clock and is extremely accurate.

Specifically on Mac OS X, the screen refresh issue related to the fact that, unlike on Windows and OS 9, the rate at which the screen updates is largely independent of the rate at which the screen actually changes. A new refresh feature updates the screen whenever it changes, up to a specified frame rate. If you have a Mac with an older video card you may experience an overall degradation of visual performance with the refresh feature on. It can be turned off in the PerformanceOptions patch described below.

We've exposed a number of the advanced performance settings in an easy-to-use patch called PerformanceOptions found in the Extras menu. You can use this patch to adjust the way Max/MSP behaves to suit your specific needs. The default settings are intended to serve a broad variety of applications, but if you notice a specific performance issue, you may be able to address it using this patch. Included are variables that specify how often and how "busily" the Max scheduler runs, how the scheduler reacts to losing time, and so forth. Each performance parameter is described in sub-windows of the patch.

New objects in 4.5

The following objects are new for Max version 4.5:

atodb	Convert linear amplitude to a dB value.
atoi	Convert ASCII characters to integers.
autopattr	Expose multiple objects in a patcher to the **pattr** system.
bgcolor	Set a patcher's background color. Previously included in Jitter.
bline	An event-driven multi-segment line object. Previously included in Jitter.
cpuclock	Make precise "real-world" time measurements.
dbtoa	Convert a dB value to linear amplitude.
deferlow	Defer the execution of a message (always).
filewatch	Notify Max of a file change.
fontlist	List the system fonts. Previously included in Jitter.
function	A graphical breakpoint function editor. Previously included in MSP.
hi	Human Interface (game controller) device input.
itoa	Convert integers to ASCII characters.
jit.cellblock	Two-dimensional storage and viewing. Previously included in Jitter.
js	Execute Javascript code.
jsui	A Javascript user interface and OpenGL graphics environment.
linedrive	Scale numbers exponentially. Previously included in MSP.
listfunnel	Index elements of a list and output them individually.
loadmess	Send a message when a patch is loaded.

mtof	Convert a MIDI note number to frequency. Previously included in MSP.
mxj	Execute Java code in Max.
nslider	Display and output numbers from an onscreen notation display.
pak	Output a combined list when any element changes. Previously included in Jitter.
patcherargs	Get parent patcher arguments.
pattr	Patcher-specific, named data wrapper.
pattrhub	Access all of the **pattr** objects in a patcher.
pattrstorage	Save and recall presets of **pattr** data.
qlim	Queue-based message passing control. Previously included in Jitter.
qlist	A collection of messages to send remotely.
qmetro	A queue-based metronome. Previously included in Jitter.
quickthresh	Fast chord detection.
regexp	Use PERL-style regular expressions to process input.
router	A **matrixctrl**-compatible Max message router. Previously included in Jitter.
suckah	Get pixel color at a display coordinate. Previously included in Jitter.
ubumenu	A non-interrupting pop-up menu. Previously included in Jitter.

The following list includes new MSP objects in version 4.5:

adsr~	An ADSR envelope with thispoly~ muting features.
atodb~	Linear amplitude to dB conversion at signal rate.

cascade~	A cascaded series of biquad filters, useful in conjunction with the **filtergraph~** implementation.
dbtoa~	dB to amplitude conversion at signal rate.
fbinshift~	A frequency domain bin shifter for use with **pfft~**.
filtercoeff~	Converts filter frequency, amplitude and resonance to biquad coefficients at signal rate.
freqshift~	A time-domain frequency shifter.
ftom~	Converts frequency to MIDI note numbers at signal rate.
gizmo~	A frequency domain frequency shifter for use with pfft~
hilbert~	A Hilbert transform for signals.
hostcontrol~	ReWire host control formatting and transmission.
hostphasor~	Retrieve a synchronization signal from a ReWire host.
hostsync~	Retrieve transport control information from a ReWire host.
levelmeter~	A VU-style meter with ballistics and range control.
mtof~	Convert MIDI note numbers to frequency at signal rate.
omx.4band~	A 4-band multiband compressor/limiter.
omx.5band~	A 5-band multiband compressor/limiter.
omx.comp~	A compressor/limiter with a dual-band option.
omx.peaklim~	An audio peak limiter.
rect~	A pulse oscillator with anti-aliasing.
saw~	A sawtooth oscillator with anti-aliasing.
spectroscope~	User-interface display of spectrogram or sonogram information.
sync~	Synchronization from tap tempo, MIDI beat clock or audio click tracks.
techno~	A signal-driven step sequencer.

tri~	A triangle oscillator with anti-aliasing.
zplane~	Display filter poles and zeros on the Z-plane.

Enhancements have been added to a number of existing objects. A few highlights include:

filtergraph~	Now supports multiple filter settings, with the output properly formatted to control the cascade~ object.
in, out, in~, out~	Descriptive text can be added (using the inspector) which will be displayed as assistance for the corresponding outlet when used within a **poly~** instance.

Although it is not an "object", an important addition is the new ReWire output driver, which provides output to any host that accepts a ReWire connection.

Changes to the MIDI system

New drivers

There are a number of new MIDI features in Max 4.5 that, while representing a big improvement over previous versions, may require you to change patches that make assumptions about your MIDI setup. We now allow multiple active MIDI driver objects - in previous versions, only *coremidi* on the Mac and *midi_mme* on Windows were available. We've introduced two new ones:

- *midi_adrewire,* which sends and receives MIDI from a ReWire host application when Max/MSP is a ReWire client
- *augraph (Mac) / midi_dm (Windows),* which address DLS synthesizers built into each platform's operating system.

In addition, using the *coremidi* driver on the Macintosh, MIDI ports may have different names than they did in 4.3. We are now using a standard technique to display the MIDI device name provided by the user in the Audio MIDI Setup application. This means that the order of available MIDI ports may have changed. We suggest that you review the MIDI Setup dialog in Max/MSP 4.5 after launching the application for the first time, and verify the operation of MIDI objects (such as notein and noteout) that might use specific port abbreviations, offsets, or names which may have changed due to the new features.

DLS Synthesizer Access

Both the Macintosh and Windows XP provide a DLS (Downloadable Soundfont) synthesizer for MIDI playback. The ability to drive DLS synths directly from Max via MIDI is a welcome feature for learning the program without needing to use external MIDI gear.

By default, a single augraph (on Mac OS X) or midi_dm (on Windows) port is created. However, you can create additional MIDI synthesizer ports and assign new DLS sound bank files to each one. Addressing the DLS synthesizers currently requires the use of the message box technique where you send messages to named objects by typing a semicolon followed by the message text into a message box, then click the message box. Here are the details:

Creating a Port:

;#SM createoutport <portname> <drivername>

where *drivername* is midi_dm on Windows and augraph on the Mac. *portname* is the name you assign to the port. For example:

; #SM createoutport myOtherSynth midi_dm
; #SM createoutport myOtherSynth augraph

Deleting a Port:

;#SM deleteoutport <portname> <drivername>

where *drivername* is midi_dm on Windows and augraph on the Mac. *portname* is the name of your choice. For example:

; #SM deleteoutport myOtherSynth midi_dm
; #SM createoutport myOtherSynth augraph

Loading a DLS Bank (type 1 or 2):

;#SM driver loadbank <filename> <portname>

where *filename* is the name of an existing DLS bank file, and *portname* is the name of the port that will use this bank. If portname is omitted, all DLS ports will use the bank. On Mac OS X, the folder /Library/Audio/Sounds/Banks is added to the search path when looking for a DLS bank file.

Loading the Default GM Bank:

;#SM driver loadbank 0 <portname>

Turning Reverb On and Off:

;#SM driver reverb 1/0 <portname>

By default reverb is off in both augraph and midi_dm.

Setting the MIDI Output Latency (midi_dm only)

;#SM driver latency <time> <portname>

where *time* is a value in milliseconds and *portname* is the port that is set to this value. For example, the following message would set the latency to 10 milliseconds:

;#SM driver latency 10 portname

Adding Virtual MIDI Ports (Macintosh only)

By default, Max creates two "virtual" MIDI ports for both input and output. If you wish to add additional virtual ports to your MIDI system, you can use the same message box technique described above to send the createoutport and createinport messages.

;#SM createoutport <portname> <drivername>
;#SM createinport <portname> <drivername>

where *portname* is the name you assign to the port, and *drivername* is the driver to be used (currently, only CoreMIDI is supported). For example, the following two messages:

;#SM createoutport myvirtualport CoreMIDI
;#SM createinport myvirtualport CoreMIDI

Would create a virtual MIDI input and output port, each named "myvirtualport". These virtual ports are not saved as part of the Max/MSP setup, so they will have to be recreated each time you restart Max.

Cycling ’74 Folder Management

Working with New and Old Versions

Max/MSP stores its external objects and other support files in a folder that we call the Cycling '74 folder, located at the following locations:

/Library/Application Support/Cycling '74 on Mac OS X

\Program Files\Common Files\Cycling '74 on Windows XP

The Cycling '74 folder is a system-wide common resource; it is intended to be shared by all versions of Max/MSP, Pluggo and Max-based applications (such as radiaL on Mac OS X). The external objects in the Cycling '74 folder for Max/MSP 4.5 are, in general, not compatible with previous versions of the application. This means that you cannot launch a previous version of Max/MSP after having installed 4.5 and expect it to work correctly. Similarly, the 4.5 application will not work properly with the set of 4.3 externals in the old Cycling '74 folder.

If you need to switch back and forth between versions 4.3 and 4.5 for some reason, you'll need to make a copy of your Cycling '74 folder **before** installing version 4.5. To revert, rename the Cycling '74 folder something else (like Cycling '74 4.5) and rename the 4.3 folder to be Cycling '74.

If you expect to use version 4.3 extensively, or if you will be using 4.3-based applications (like radiaL), you can take advantage of version 4.5’s new resource search method. The search strategy used for the Cycling ’74 folder is:

1. If a folder named “Cycling '74” is found in the Max Application folder, use that one.
2. If a folder named “Cycling '74” is found in the Common or /Library/Application Support folder, use that one.
3. Otherwise, the C74 folder is set to be the same as the Application folder. This effectively is the same thing as adding “./” to the search path.

Note that this will only work for Max 4.5. However, it will allow the following:

1. Install Max 4.3.
2. Rename Max 4.3 Cycling '74 folder.
3. Install Max 4.5.
4. Move Cycling '74 folder into MaxMSP 4.5 Application folder.
5. Rename Max 4.3 Cycling '74 folder back to Cycling '74.

This will allow both versions of Max to operate without any modification of the max search path.

Issues with other Cycling '74 software

Radial

Radial will not operate with a standard Max/MSP installation. If you want to use radial on the same computer as Max/MSP 4.5, you will need to use the techniques described in the "Working with New and Old Versions" section of this document.

Pluggo and Mode

Pluggo 3.1 and Mode 1.0 are not compatible with Max/MSP 4.5. Installing Max/MSP will make both packages unusable both in Max/MSP and other hosts. We have the new Pluggo version 3.2 and Mode 1.1 that are compatible with Max/MSP 4.5 available on our web site for downloading.

Jitter

A public beta version of Jitter is available for download that will operate with Max/MSP 4.5. Current Jitter users can download and test using this version.