Welcome to Modalys!

With Modalys you will enjoy creating ear-blowing virtual music instruments based on physical models.

Operating Sytem and Computer Requirements

Modalys works for :

- Mac OS 10.7 and later (including High Sierra)
- Windows 7 or later ("Modalys for Max" environment only)
- Intel Mac or PC with a minimum of 2 Gigs of RAM.

Environment

Modalys is primarily a framework that can be used under various environments:

- As **ModaLisp**, a stand-alone application powered by LISP language.
- As **Modalys for Max**, a set of Max/MSP objects intended for realtime use. Compatible with Max 5, Max 6, Max 7, in 32 or 64bit context.
- As an **OpenMusic** library.
- As a **MatLab** set of 64bit objects for scientific applications.
- With **Max for Live** stand-alone instruments.

Installation

- Download the latest version of Modalys for Mac (.dmg archive).
- Launch the installation package (« Standard » installation is recommended.)
- If you already have an /Applications/Modalys folder, don't worry, it will be renamed and backed-up.
- If you want to use Modalys with Max/MSP, OpenMusic or MatLab, you need to manually copy the right components to the dedicated place. Components are found in /Applications/Modalys/Components.
- The Documentation can be accessed either from ModaLisp Help menu or directly with the help.html file in /Applications/Modalys.

Registration

If this is the first use of Modalys, or if you are upgrading from version 3.2 or older, you need to authorize Modalys with the activation code sent by the Ircam Forum server. Launch the **Modalys Authorization Tool** from the Modalys application folder.

However, this step is no longer necessary for use in Max/MSP!

On the next pages are found the cumulative release notes.

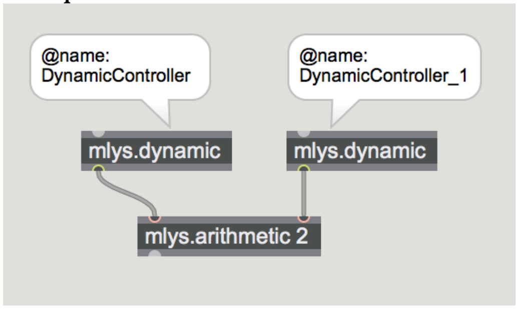
Release notes for Modalys 3.5.0 (rc1)

General

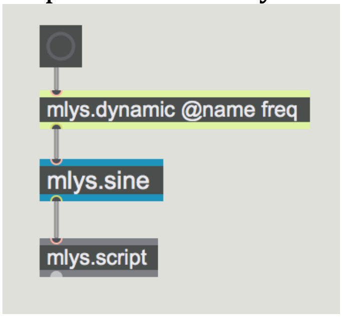
- Mac OS High Sierra compatibility checked.
- Version 3.4.5 → 3.5.0

Modalys for Max

- More help patches available (still in progress…)
- `modalys~`:
  - new "clear" message to completely reinitialize the Modalys engine.
  - `getinfo` now returns correct float values if necessary (instead of just symbols)
- Windows: `mlys.expression` now available!
- Mac: an undesired « debug » CPU-consuming modalys~ object was found in 3.4.4 :-o
- Bug fixed:
  - Crash with `getinfo value` message on expression controller.
  - Crash with expression controller with `getinfo` or `setinfo` instruction.
- Improved management of **variable names**: a new algorithm has been implemented to avoid mlys variable name duplicate. So from now on, if you create to similar objects without specifying the @name parameter, the actual name of each object is unique:



- Improved `mlys.sine` controller (sine wave): now the **frequency** can be controlled with some input controller dynamically:

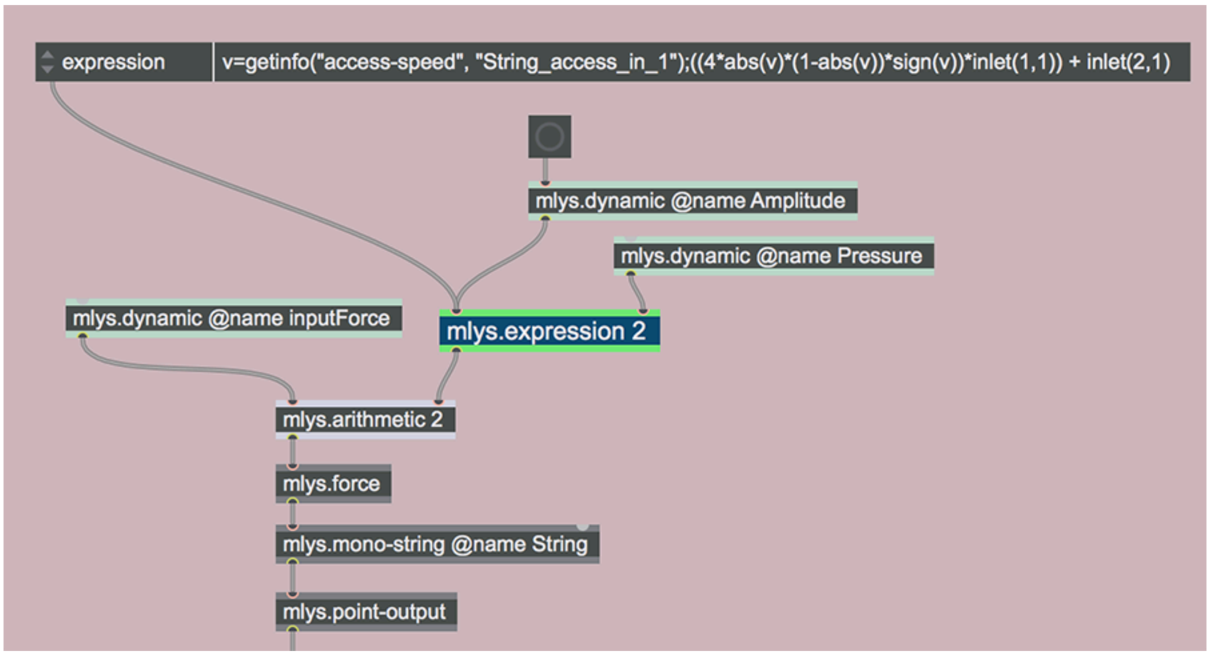


- Improved `mlys.dynamic`:
  - You no longer need to specify the dimension of the default value, which is 0 (multidimension will be implemented later)
  - Default value did not work and was always 0.
- Improved `mlys.point-ouput` when a controller is set as inlet.

- `mlys.access-speed`, `mlys.access-force` and `mlys.access-position`: more specific inlet and outlet tooltips.
- Improvement to the `mlys.expression`.
    - New `update` attribute: -1 = automatic, 0=every sample, otherwise a value in second. Default is 0.01 (=10 ms).

  Let us remind what is is about: the expression controller is a powerful tool – yet seldom used – that allows to create little code snippets (called "expressions") that can be evaluated at each sample. It was created years ago by Nicolas Ellis, but the use in Max MSP had remained very limited so far. With the help of the Hans Peter Stubbe, we have reviewed and improved this precious tool quite a lot.

  Here is a situation where the expression can be used:

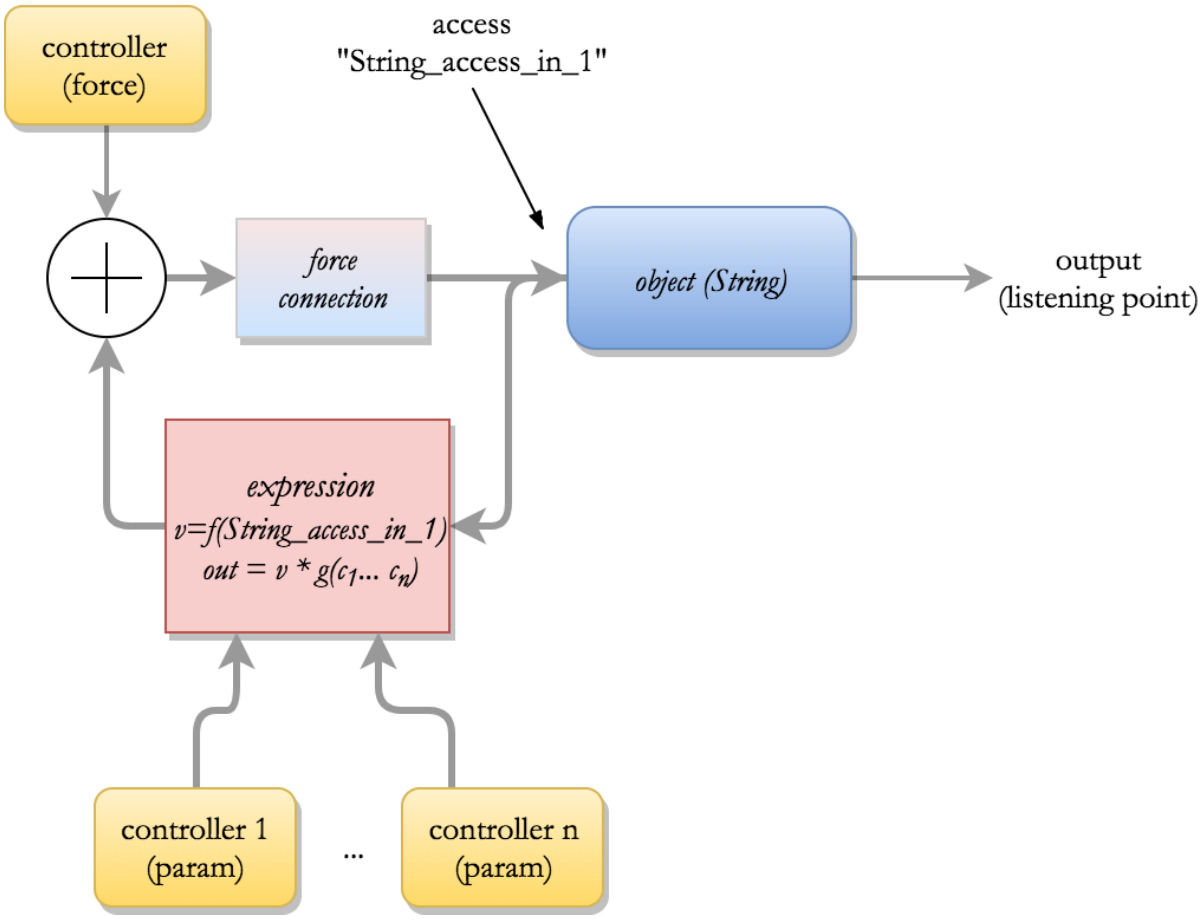

expression
mlys.dynamic @name Amplitude
mlys.dynamic @name Pressure
mlys.dynamic @name inputForce
mlys.expression 2
mlys.arithmetic 2
mlys.force
mlys.mono-string @name String
mlys.point-output

```
v=getinfo("access-speed", "String_access_in_1");((4*abs(v)*(1-abs(v))*sign(v))*inlet(1,1)) + inlet(2,1)
```

  Description of the example. A force is injected into a string, through a dynamic controller ("inputForce"). The expression controller listen to the speed at the access "String_access_in_1" where the force is injected. Depending on 2 additional controllers ("Amplitude" and "Pressure"), the expression controller adds a certain value to the injected force, creating a feeedback loop.

  That kind of loop situation is generally not handled very gracefully by Max MSP, this is why we created two special types of "getinfo", to get the speed or the force in real time at a certain access.

  Here is the more readable diagram:



Properly set, the expression controller can help create for instance *non-linear* modal objects. And it can be used in potentially any situation where some feedback is involved.
Please open the mlys.expression help patch for more details.

Release notes for Modalys 3.4.4

General

The installer is now cleanly signed ! ;-)

ModaLisp

- ModaLisp is now fully compatible with Mac OS Sierra and runs in native 64bit mode.
- The progress bar has been updated to Cocoa (to comply with the new 64bit mode).

Modalys for Max

- Better handling of path
- Fixed : rare crash with Max 7.3.

- Fixed : (long-standing) crash with expression controller when running Max in 32bit mode.
- A new package info item has been added making Modalys appear nicely in Package Manager.
- The 'vocalys' instrument has been updated and added to the Modalys for max Instrument Series (Mac only).

Release notes for Modalys 3.4.3

ModaLisp

Surprisingly, *get-info* did not work on non-dynamic controllers (envelope, constant etc.)

Envelope controllers could be slightly ≠ 0 when expected to be really null!

Open Music

A nasty and obscure library conflict when using the latest version of Open Music (64bit) forced us to updgrade and recompile some of Modalys' key modules.

Examples have been updated, too.

Modalys for Max

getinfo message returned a non-standard mix of value and symbols. Now a consistent list is returned.

Windows: some specific configuration incompatibilities have been fixed.

Release notes for Modalys 3.4.2

Mac

Compatibility with Mac OS X 10.11 « El Capitan »

Windows

For the first time, Modalys now offers Windows support in Max/MSP environment! This works in 32 or 64bit context. No forum authorization code is necessary for the Mlys objects to work.

Modalys for Max

Performance

A significant performance boost is expected with this version. ;-)

Protection

The Forum protection scheme has been removed fromt the Modalys for Max objects, so now you can install them from a machine to another with no protection hassle. Also, the Modalys engine is now embedded in modalys~ object, so you're no longer dependent on the Modalys Framework (which is still required for ModaLisp, OpenMusic or Matlab)

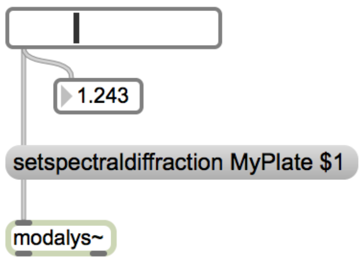
Pseudo-physical parameters

Version 3.4.1 introduced the cent pitch adjustment as the first « pseudo-physical » parameter in Modalys. Unlike « real » physical parameters, these pseudo-physical parameters have a direct action on the object spectrum, i.e. on the modal frequencies and loss, without having to recalculate everything each time. So they're lightning fast!

Pseudo-physical parameters are also available for hybrids (melt- or mix-).

3.4.2 introduces a new pseudo-physical parameter:

- New message **setspectraldiffraction**:

  The spectral diffraction contracts or dilates the spectrum of an object:
  If $\{f_i\}$ are the object's mode frequencies, with f_0 the fundamental, the effect of a spectral diffraction ρ is:

  $$f_i' = (f_i - f_0)\cdot \mathrm{len}(\boldsymbol{\rho}(e-1)+1) + f_0$$

  with f_i' the resulting mode frequency.

  So $\rho = 1 \Rightarrow$ no diffraction, $\rho = 0 \Rightarrow$ spectrum completely folded onto f_0.

  This can be used to explore object's timbre alteration, for example to add some slight inharmonicity.

- Message **setcentspitchbend** is no longer limited to +/- 2400 cents.

New Modalys for Max Instruments Series

As part of the long term project of completey renewing the Max-based instrumentarium, we are releasing in 3.4.2 new relatively sophisticated instruments. In this release you'll find:

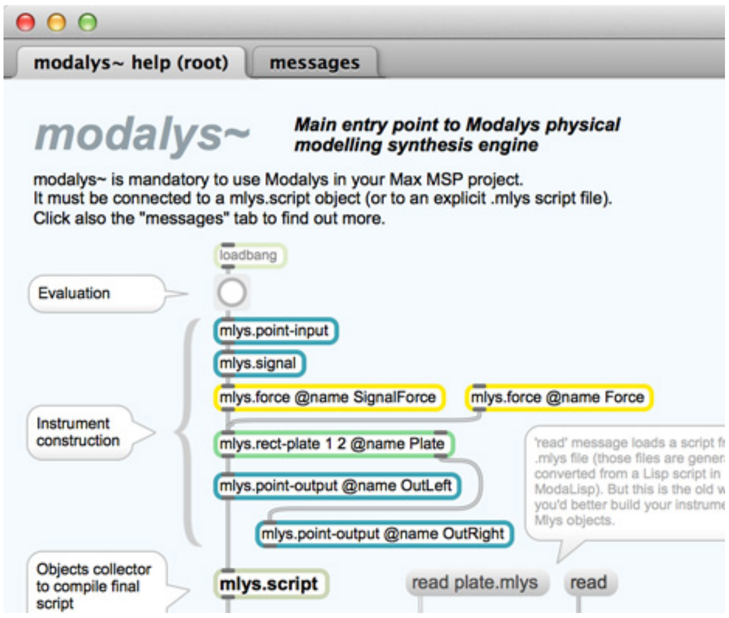
- Snare drum
- Bass drum
- Simple cymbal
- Hi-hat cymbals
- Eolian harp
- Plate filter

An **Modalys for Max Instrument Series** item from the « Extras » menu provides with a simple access to the corresponding patches.

Instruments can be found in the /instruments folder within the Mac package. We hope that they will be inspiring to you!

Help Files

The mlys and modalys~ help files are almost finished. In this release you can see the progress we've made and how they can help you to get the most of Modalys in Max.


modalys~ help (root)
messages

modalys~ *Main entry point to Modalys physical modelling synthesis engine*

modalys~ is mandatory to use Modalys in your Max MSP project.
It must be connected to a mlys.script object (or to an explicit .mlys script file).
Click also the "messages" tab to find out more.

'read' message loads a script fr .mlys file (those files are gener converted from a Lisp script in ModaLisp). But this is the old w you'd better build your instrume Mlys objects.

loadbang
Evaluation
mlys.point-input
mlys.signal
mlys.force @name SignalForce
mlys.force @name Force
Instrument construction
mlys.rect-plate 1 2 @name Plate
mlys.point-output @name OutLeft
mlys.point-output @name OutRight
Objects collector to compile final script
mlys.script
read plate.mlys
read

Bugs

- The Envelope controller did not work well in Lisp context.
- « Save as Script » was broken in ModaLisp.
- Old nasty bugs potentially affecting finit elements computing.
- The Listener window could no longer be re-opened if closed once.
- Matlab objects were broken since a few builds.

Documentation

The documentation has been updated to reflect the current changes and additions.

Release notes for Modalys 3.4.1

General

- The Felt connection had been broken recently (thanks, Mr Pagliei!)
- An important (and long-standing I think…) source of crash fixed.
- Important performance regression fixed comparing with b7.
- Lots of internal changes and cleanups for Windows compilation compliancy.
- Removed behind-the-scene expression controllers instances.
- Inconsistencies in version # fixed.

Max/MSP

Bug fix

- Crash fixed with mlys.read-from-file (bad default access types)
- Crash in mlys.labium fixed.
- Crash using *version* message.

Help files

MLYS objects (modalys~ and mlys.*) are now fully documented. This had been a long-standing request and, after the release of the general Modalys documentation, this represents an important step in making Modalys easier to use.

Instrument examples

In the same direction, we have entirely updated and redesigned the instrument examples. All examples are written using MLYS.

New messages

- **freeze/unfreeze** messages can now be send to an object ; a « frozen » object no longer interacts or vibrates, and consequently don't spend any CPU. The syntax is: *(un)freeze <Modalys object name>*.
- **setcentspitchbend** is the message to set the new pitchbend control that is now available in any object. The syntax is *setcentspitchbend <Modalys object name> <value>*. The value are in cents[1], so -300 will bend the pitch down by a minor third and +50 up by a quarter tone for instance, default is 0 (normal pitch). The value is limited to +/- 2 octaves (i.e. +/- 2400 cents)
Warning! This was implemented differently in 3.4.1.b6!

Matlab

set-info and *get-info* function were missing from last distribution.

Release notes for Modalys 3.4.0

64bit

Modalys is now fully 64bit throughout, with 32bit backward compatibilty. That means that for applications that are natively 64bit (such as Max 6.1 or MatLab), Modalys runs natively in 64bit, and for 32bit apps (such as Max 5 / 6.0, ModaLisp or Open Music), it still runs in 32bit mode.

Max/MSP

Objects

One single set of Max/MSP objects is available, compiled with 6.1.4 SDK and fully compatible with:

- Max 5 and 6.0 (running in 32bit)
- Max 6.1, running in 32 or 64bit, depending on how Max is launched.

Furthermore, the objects set is compatible with 6.1 "packages".

Warnings

A rather cryptic warning has been taken care of ("*the number of oulets blah blah...*"). This was a long-standing issue.

[1] http://en.wikipedia.org/wiki/Cent_(music)

MatLab

- Objects are now natively 64bit. The legacy 32bit set of objects has been discontinued as MatLab is natively 64bit since version 2010b.
- Several significant bugs have been fixed.

Strike connection

The strike connection was weak as molasses in the latest public release (3.3.1). Should be back to normal now!

Lisp environment

- ModaLisp has been upgraded to LispWorks 6.1 framework.
- Lisp precision had been set to *single-float* by default for ages… This is now set to **double-float** for more accurate and consistent results.
- The mouse wheel or trackpad is now functional in the editor.

License

- The license engine has been updated for the Forum 2013 system.
- Support for IrcaMax 2

Release notes for Modalys 3.3.1 (rc3)

Framework

Some work has been done for 64bit compliancy but this is not quite ready for prime-time, so we stick to 32bit at the moment.

That said, any version is clearly tagged 32bit or 64bit from now on, just to avoid confusion in the future when we have both options.

Performances

Better Max performances and stability comparing with 3.3.0.

Documentation

Habemus Modalysis documenta !

That was a promise made at 2012 Forum and here it is! We are proud to release this new, up-to-date documentation for Modalys.

The doc written by Morrison and Waxman around 1996 was obviously getting more and more obsolete. Nonetheless, we have kept as much spirit and content of this great legacy work, and **Richard Dudas** has worked really hard to make sense of all this, adding and rewriting sections entirely, and always making sure that everything would work as stated.

For now the help files focus on the ModaLisp interface. But this is just a starting point: release after release, the documentation will expand with tutorials, examples, and not only in the Lisp environement but also for Max (MLYS and modalys~) and Open Music.

We can I find the documentation?

Choose whatever format is suitable for you:

html version

If you work with ModaLisp, open any file and a Help menu will show up with 2 options:

- **Open Modalys Documentation**
- Go to Modalys Discussion Group (on forumnet.ircam.fr)

If you have an internet connection, you will be automatically directed to the most recent version of the documentation hosted at Ircam's server.

If you don't, the local files (i.e. on your Hard Drive) will be used instead.

If you want to access the doc files locally, just double-click on the **help.html** file found in /Applications/Modalys folder.

The latest documentation can also be found online here :

http://support.ircam.fr/docs/Modalys/current/co/publication-web.html

Note that the **search** function is currently available from the online version only.

pdf version

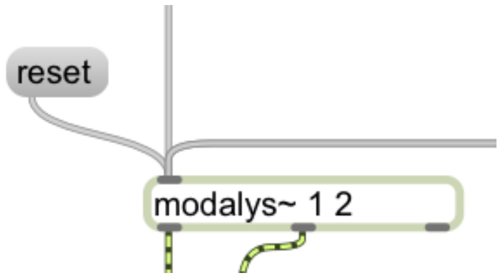
In addition, we have added a PDF version of the same doc, which is more easily searchable.

New connection

Following the work of composer Per Bloland, we now have an induction connection, to simulate crazy things such as electromagnet interacting with piano strings! See the new corresponding lisp example for further details.

modalys~ for Max/MSP

A new **reset** message has been added. Simply send this message to the modalys~ object in order to reset all objects from the current Modalys instance.



Be aware that it exclusively resets object speeds and positions. It *does not* reset access values, nor object and connection parameters.

Miscelaneous bug fixes (ModaLisp)

- (view 'mode) was broken
- Closing a window using the menu close command or with the keyboard shoftcut (cmnd+W) crashed the editor.

Release notes for Modalys 3.3.0.1 (rc1)

Authorization

Authorization Tool could not be launched under Mac OS 10.5.

MLYS for Max/MSP

Bug fixes

- **mlys.felt** object (felt connection) was broken.
- Certain connection Max objects were wrongly located in the *modalys/object/* folder. They have been moved to */connections/*.
- **bi-string** and **bi-two-mass** objects now have the correct number of elements in **access-in-type** parameter (trans0 or trans1).

Installer

Growl application has been removed from the installer. It is no longer required since authorization processed changed in 3.3.0.

Release notes for Modalys 3.3.0 (rc1)

Authorization

Modalys 3.3 implements the new authorization scheme (Nov. 2012). Be sure to own a valid activation code before installing this version. Old activation codes will not work with version 3.3 and later.

After installing Modalys, launch the little application **Modalys Authorization Tool** found in /Applications/Modalys folder.

Versioning

Considering the number of new features (already implement or upcoming), the version has been upgraded from 3.2.1 to 3.3.0. That will be the official version for the 2012 Ircam Forum.

Stability & Bug Fixes

- Crash recently reported involving a « point_output » function…).
- *(view 'mode …)* LISP command was broken in recent release.
- Crash when merging a great deal of finite element nodes.
- Long-standing *POINT-OUTPUT* error in OpenMusic environment.

MLYS (Max /MSP environment)

Directory

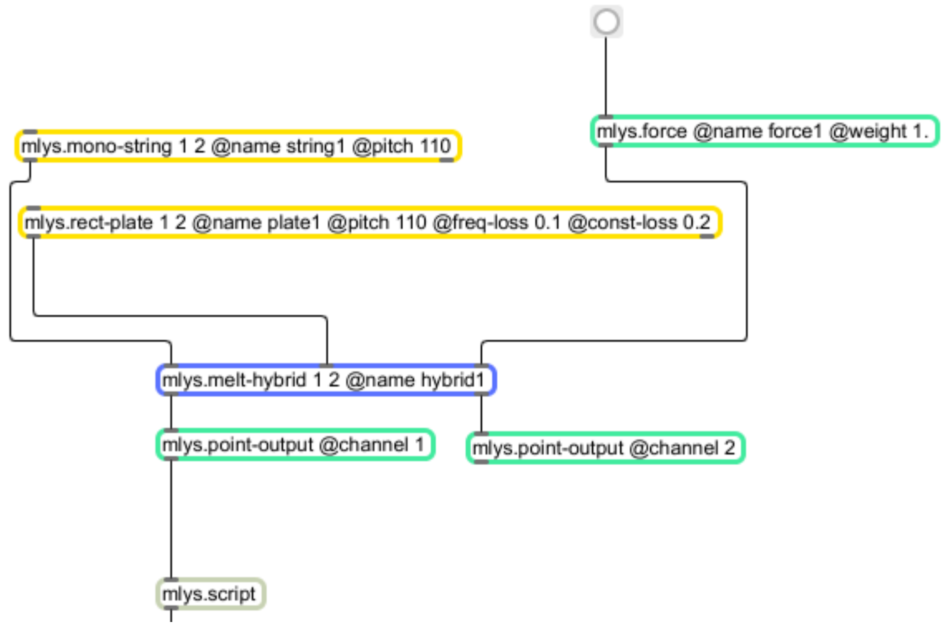
Modalys objects for Max have been reorganized in consistent subfolders. Please trash your previous **Cycling '74/…/modalys~/** or **mlys/** folder and copy-paste the the **modalys/** folder from /Applications/Modalys/Components/Max.

Objects

- **Hybrid** objects were an important missing part of **MLYS** in Max/MSP. The whole family of hybrids are now being added :

- **mlys.melt-hybrid**
- **mlys.mix-hybrid**
- **mlys.tri-hybrid** (see next bullet point)

Setting up hybrids in **MLYS** is pretty straightforward, as this example shows :

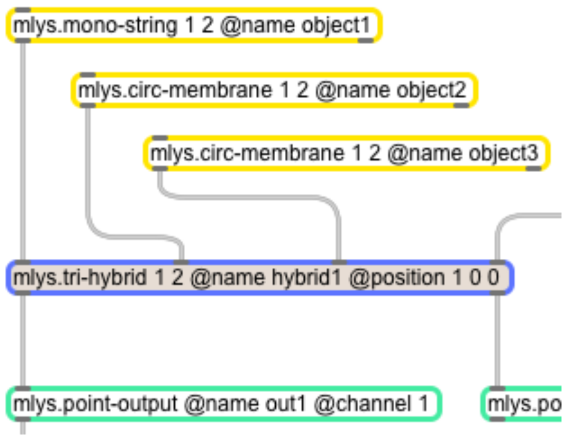

mlys.mono-string 1 2 @name string1 @pitch 110
mlys.force @name force1 @weight 1.
mlys.rect-plate 1 2 @name plate1 @pitch 110 @freq-loss 0.1 @const-loss 0.2
mlys.melt-hybrid 1 2 @name hybrid1
mlys.point-output @channel 1
mlys.point-output @channel 2
mlys.script

Some further details :

- Accesses are automatically created under the hood.
- The first 2 inlets of a (mix- or melt-) hybrid are for object 1 and 2 input.
- The attribute **@position** rules the distance from object 1 and 2 and shall be controlled using messages : hybrid1@position $1 0.01

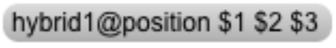
See the *examples/Max/MLYS/***hybrid_example.maxpat** file to see an MLYS hybrid in action.

- **Tri-hybrids** are now implemented, which makes the hybrid family complete! Tri-hybrids in MLYS work pretty much like melt- or mix-hybrids, except that we have 3 object inputs and that the position attribute has 3 values. Here is an example:


mlys.mono-string 1 2 @name object1
mlys.circ-membrane 1 2 @name object2
mlys.circ-membrane 1 2 @name object3
mlys.tri-hybrid 1 2 @name hybrid1 @position 1 0 0
mlys.point-output @name out1 @channel 1
mlys.po

Like for 2-hybrids:

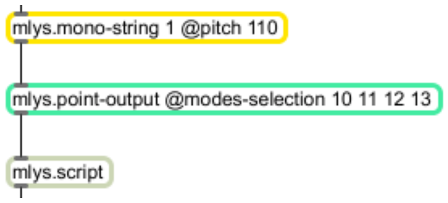
- Accesses are automatically created under the hood.

- The first 3 inlets of a 3-hybrid are for the objects input.
- The attribute **@position** rules the 3 objects's interpolation and shall be controlled using messages with 3 values : 

hybrid1@position $1 $2 $3

Open the *examples/Max/MLYS/***tri-hybrid_example.maxpat** file to see an MLYS tri-hybrid in action.

- **mlys-point-output** object :
  - A new **gain** dynamic controller has been added.
  - Output level was extremely low with mode selection.
  - Modes to be listened to can now be speficied using a list, using the new **modes-selection** attribute:



```
mlys.mono-string 1 @pitch 110
mlys.point-output @modes-selection 10 11 12 13
mlys.script
```

Note that this attribute, once chosen, cannot be changed dynamically at this time. No attribute means « all modes », that is, the normal output.

See the *examples/Max/MLYS/***modes_selection_example.maxpat** file to see an MLYS output mode selection in action.

- **New tooltips**. In edition mode, some message shows up while hovering around inlets. They previously didn't say much (« I am inlet 0 ») and they now depend on the context much more gracefully.
- **MLYS controllers**

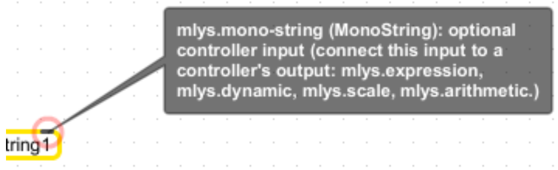
In version 3.3, we are opening up the world of Modalys controllers to MLYS. When you double-click the **mlys.script** object, you can already see that many controllers are created behind the hood, and most of them are hidden from the visual Max paradigm. The only controllers that were exposed so far were mlys.access-force, mlys.access-speed, mlys.access-position, and, to some extent, modalys.expression.

You now have the option of making certain controllers **explicit** for a finer control over objects. Also combination and connection are more and more flexible.

This is a giant step for building complex instruments with MLYS.

How does all this work ?

Objects such as strings, tubes, etc. now have a new default input at the right top of the box, called the **optional controller input**:



mlys.mono-string (MonoString): optional controller input (connect this input to a controller's output: mlys.expression, mlys.dynamic, mlys.scale, mlys.arithmetic.)

  - **mlys.dynamic**
    Dynamic controllers are entry point of the Modalys controller chain. A dynamic controller is basically a value (or a set of values, i.e. a vector) that

can be controlled dynamically via messages. It must be connected to other controllers or to objects.
    - **mlys.scale**
      This controller has just one dimension and maps an value range onto another. For instance, you can map [0…360] onto [-3.14…+3.14].
    - **mlys.arithmetic**
      This controller has 1 or 2 inputs and makes simple operations: + - * / sqrt ^ inv max < <= >= >, and ouputs the result.
    - **mlys.spread**
      This controller spread the (single) input value onto a vector of identical values of a given dimension.
    - **mlys.expression**
      Expression controllers are powerful but remain somehow mysterious to most users. We want to make them easier to understand and use and we will add some more documentation and examples soon.

  Open the *examples/Max/MLYS/***controllers_example.maxpat** file to see several situations with controllers, and how they can be combined.
- Bug fixes :
    - Pitch set to zero resulted in a click. Zero pitch is now simply ignored.
    - Long-standing warning « number of inlets for this box (0) differs from that of the script (1) » for **modalys~** with no parameter has been fixed.

Finite elements (LISP environment)

- Materials & Domains: the function **make-material**, used for finite elements-based objects, has been fully implemented in LISP and now allows multiple material « domains ».
  Basically, starting with a mesh of finite elements, a material domain declares some material properties (such as density, thickness, young modulus, etc.) and it can be defined for a certain set of elements only. Thus, this allows to build objects made of composite materials.
  One can imagine for instance a bowl going almost continuously from glass to stain, or a wine glass getting thinner and thinner towards the rim.
  Here is a piece of sample Lisp code to see how it works :

```
; Start with a mesh :
(setq my-mesh (make-mesh ...))

; Define a number of material domains and create your material.
; ( (make-material) is equivalent to (make-material 1) )
(setq number-of-domains 2)
(setq my-material (make-material number-of-domains))

; Set some mixed physical properties to your domains
; (the parameter after 'my-material' is the domain index, so here it can be 0 or 1)
(set-info 'physical 'density  my-material 0  7800)  ;setting a high density to the first domain
(set-info 'physical 'density  my-material 1  2100)  ;setting a low density to the second domain
(set-info 'physical ...)
; An additional dimensional parameter can be specified for certain types of physical parameters
; (Young, Poisson, ShearCoefficient, RotationalInertia)
; using this syntax: (set-info 'physical 'Poisson <material> <domain> <dimension> <value>)
(set-info 'physical 'poisson  my-material 0 1 7800)
; the 1st argument (after my-material) is the domain, the 2nd is the dimension (0=x, 1=y, 2=z)
(set-info 'physical 'poisson  my-material 0 2 5400)
; so different values can be set for a same parameteter depending on the dimension!

; To unable the friction constant on a certain domain and dimension you may write (infinity):
(set-info 'physical 'shear-coefficient  my-material 0 0 (infinity))

; The following physical parameters can be used:
;       young                 poisson               length                mass
;       density               thickness             tension
;       shear-coefficient (for beams)               rotational-inertia (beams)
;       section (beams)       torsion-constant (beams)
;       air-density (tubes)   air-elasticity (tubes)
```

Release notes for Modalys 3.2.0 (rc1)

Framework

There have been major changes in the way the « context » is handled. A Modalys context is a world (i.e. memory space) where objects are created and interact. Several contextes can coexist within the same thread and they are completely independent from eachother. All this is now handled much more gracefully.

Performances

There were serious performance drawbacks in the previous release (3.1.3), especially using Max. Things are back to normal in this release.

Max for Live environment

The above framework changes (context) were originally done to solve nasty bugs (crashes) when using multiple Modalys instruments in Max for Live. The situation should be way cleaner now!

LISP environment

- Some examples have been updated ; Bach.lisp example has been added.

- Some issues in the component files have been fixed.

Max environment

poly~ with Modalys had been broken in recent beta builds (3.2.0.b1/b2).

MatLab environment

Better, safer error management.

Installer

Until now, Modalys installed « Growl » (a third-party notification system) and now you can unselect/select it from the custom install.

Release notes for Modalys 3.1.3

OpenMusic

- Missing **produce-samples** variable upon Modalys object invocation.
- Object of type 'Reson' could not be created because of wrong number of arguments.
- Unicity for generated name file in **synthetize** function has been improved (now based on « tick » count).
- **save-script** LISP function resulted in an error if the file chooser was cancelled.
- Modalys data for OpenMusic have been cleaned up and simplified. Examples can now be found within the **tutorials** folder inside the Modalys library.

LISP environment

- The **Lisp** menu has been rebaptized **Script**.
- A new **Export Last Synthesis as AIFF** item has been added to Script menu (superseeds the previous experimental File/Export as AIFF).
- A new **Play Last Synthesis** item has been added to the Script menu (keyboard shortcut : ⌘ + P).
- **Evaluate** item has been renamed **Run Script** (Script menu)

Max/MSP environment

Objects have been updated to 5.1.7 SDK (5.1.6 was used before).

Release notes for Modalys 3.1.2

Reorganization of packaging

- Modalys now has a **standard installer** with several options :
    - Install Framework
    - Install main application folder
    - Install MAX/MSP components
    - Install OpenMusic components
    - Install MatLab components

All options are set by defaults.
During installation, if no Modalys license can be found on the machine, the registration dialog shows up (if then no license is entered, the installation completes anyway, and you're notified through « Growl »)

- The **Modalys folder** is simply called Modalys within Applications.
- If a previous Modalys folder is found, don't worrry, it won't be erased and will be renamed with the current date as a suffix (note that the Modalys framework, however, will be replaced).
- Several obsolete components such as « Framework Selector » have been dropped.
- Files within the Modalys folder have been reorganized so it looks less messy :
    - **Components** folder : includes files for different environment such as MAX, OpenMusic or MatLab.
    - **Examples** folder : includes the legacy examples (ex1, etc.). Examples will be dramatically enriched and updated in a future issue.
    - Legacy **mos** folder has merged with the Components Folder.
    - **Help** folder : with the legacy help files (the Help system is being entirely redesigned – patience…)
    - **ModaLisp** is the Lisp environment for Modalys (it was named « Modalys (LispWorks) » in the past, and the icone has been udpated, too)
    - **modalys** command line executable has been moved out to /usr/bin/ ; so, for expert users, it can now be invoked from anywhere directly.
    - **Medit** the 3D mesh viewer provided by INRIA and used by Modalys, has been relocated to Modalys.framework.

Code cleaning throughout the Modalys project

Nothing apparent to the you here, but yes, some work has been done to make the code clearer and cleaner.

Drop of PPC support

Amongst other benefits, this makes binary files smaller.

Strike connection

A strike connection can now be initialized with identical positions without Modalys complaining.

Exporting to file

When exporting to a file (as a modal object or a sound file), errors (bad path, etc.) are now treated as simple warnings , so they don't stop the sample generation anymore.

LISP environment

- The default font size is bigger (12 vs. 10)
- A new **View** menu has been added with two items : **Increase/Decrease Font Size** with keyboard shorcuts Cmnd « + » / « - »

- A new **Export as mlys** item has been added to the File menu. I believe this is a most natural way to produce .mlys files for MAX!