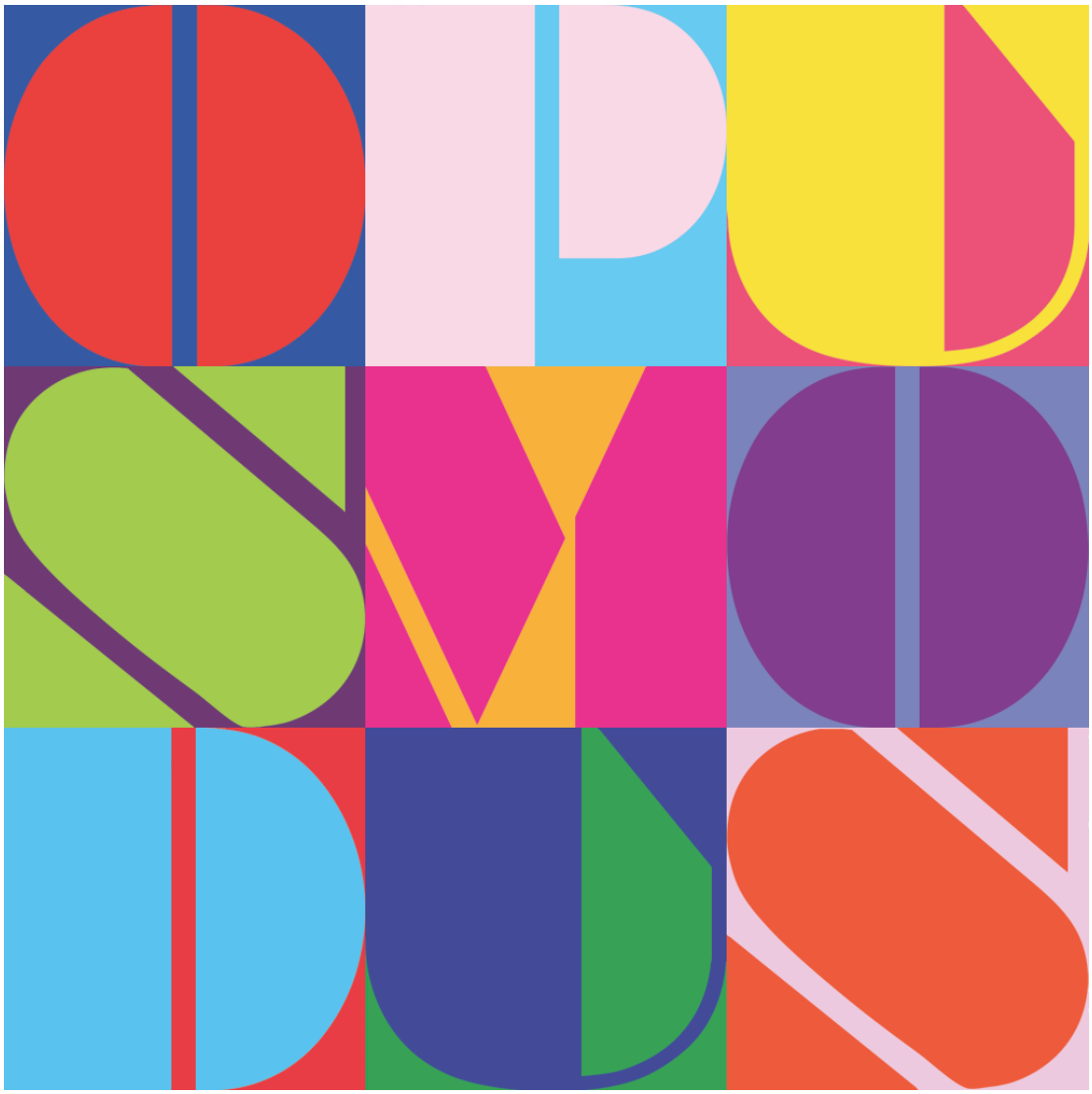
Quick Guide

Introduction to Opusmodus

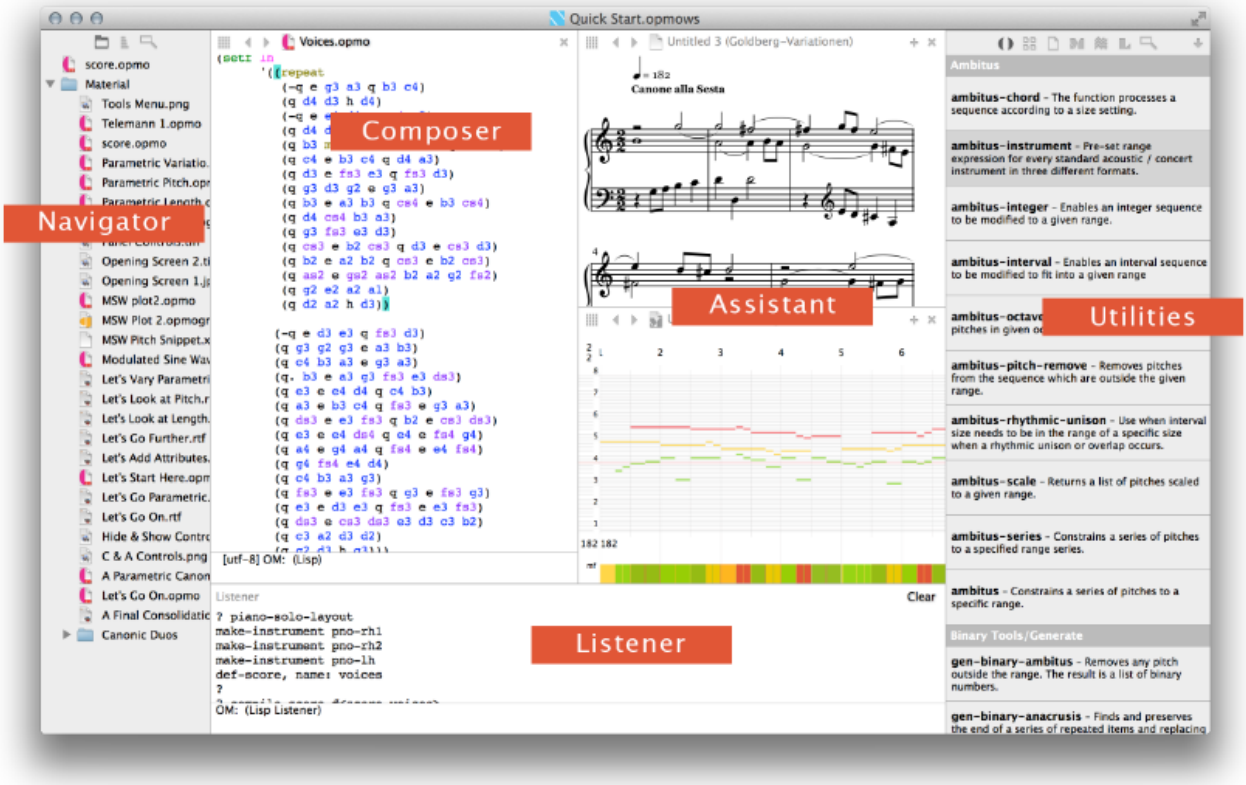


Contents

Workspace

Navigator, Composer, Listener, Assistant and Utilities

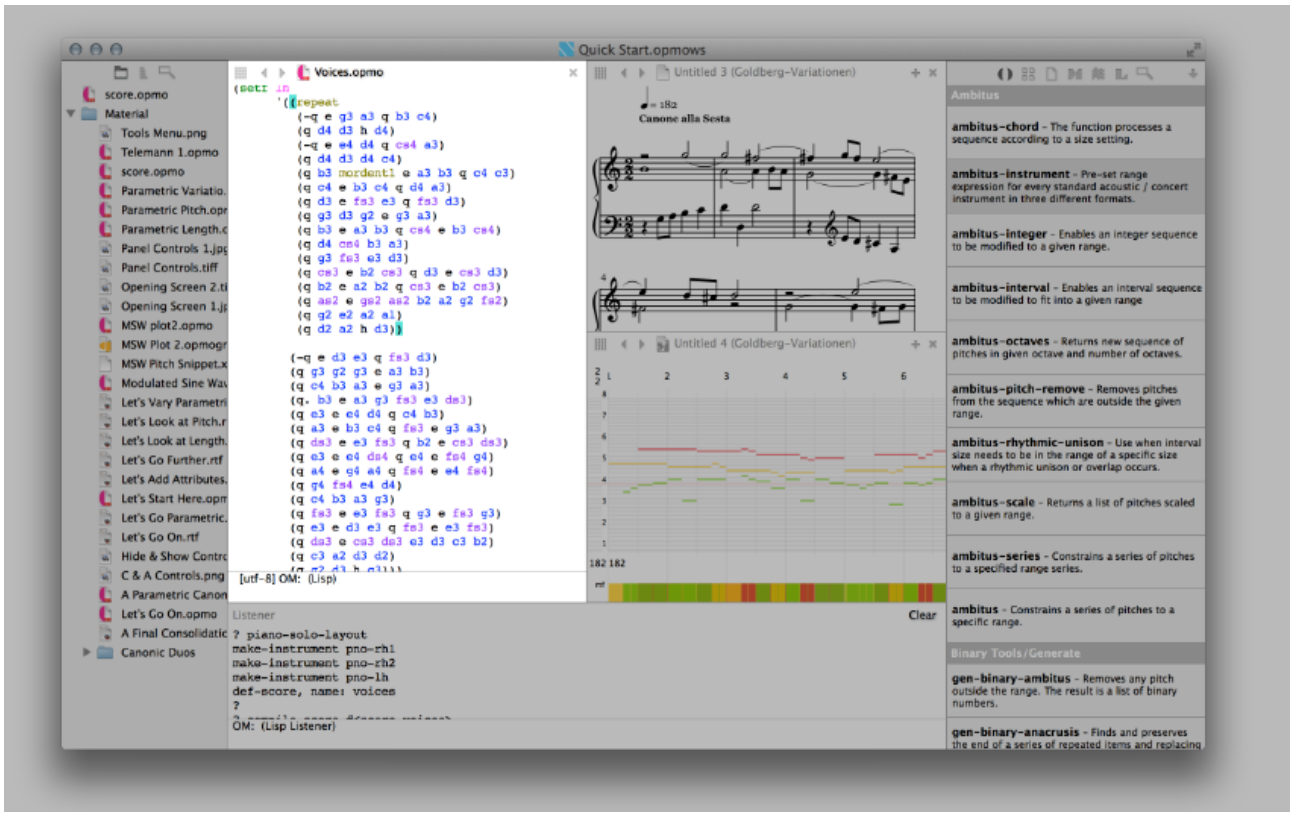
The Opusmodus workspace consists of five integrated panels, each dedicated to a specific purpose. The images below display each of the panels that make up this unique music composition software. Together the panels make up a most exciting and flexible workspace for musical creativity.



How the composer will use the Opusmodus interface with its many features and possibilities will always be a matter of experiment and personal choice. One of the objectives around the design of Opusmodus is to respond to the many and various approaches composers have to make in particular projects and circumstances.

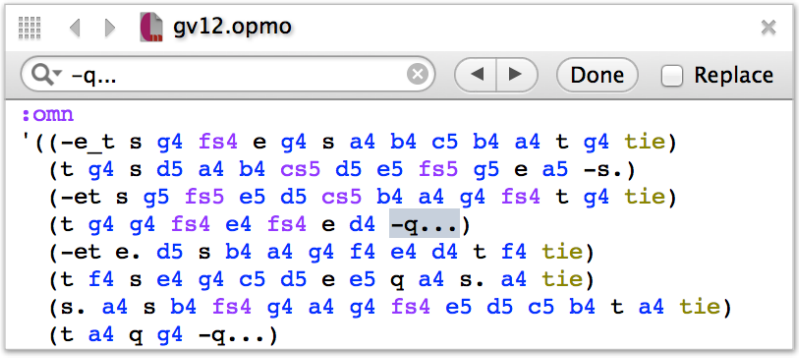
Composer

At the heart of Opusmodus is the Composer panel. This is where the script for a new piece of music comes together. It starts as a blank page, but composers quickly learn to fill it with expressions that can be seen as a score, listened to, shown as notation or visualised graphically. The Composer is a script-editor; it is an active space, with the Listener constantly monitoring its activity.



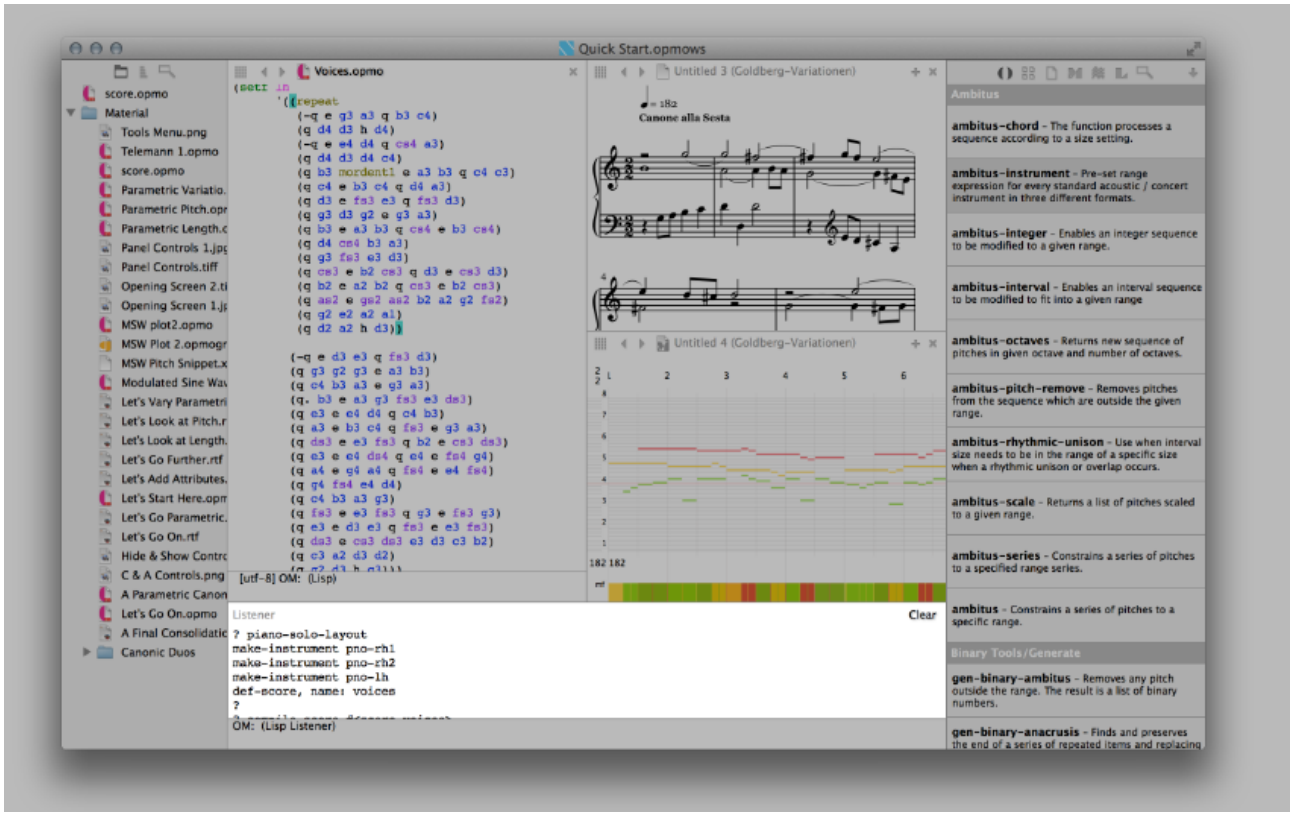
Find and replace bar

Find quickly any text or symbol in your score file.

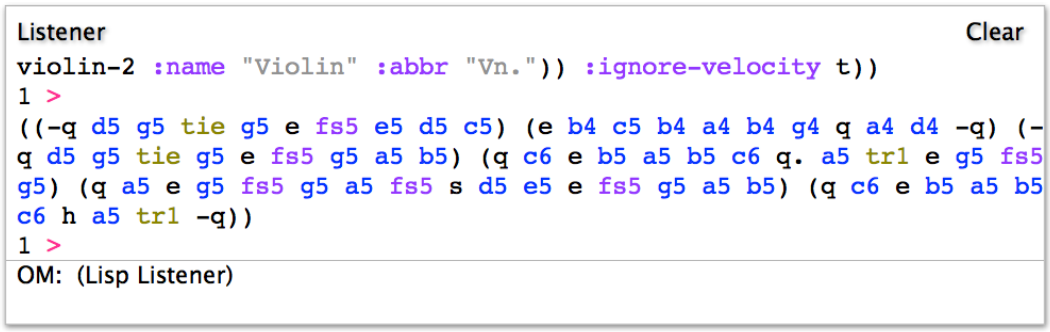


Listener

A Listener panel is provided to let you evaluate Lisp expressions and OMN forms. This tool is invaluable as a method of testing your score and for reading the results of evaluated expressions. Anything that's entered as data into the Composer shows up in the Listener as a trace of the output every expression creates.

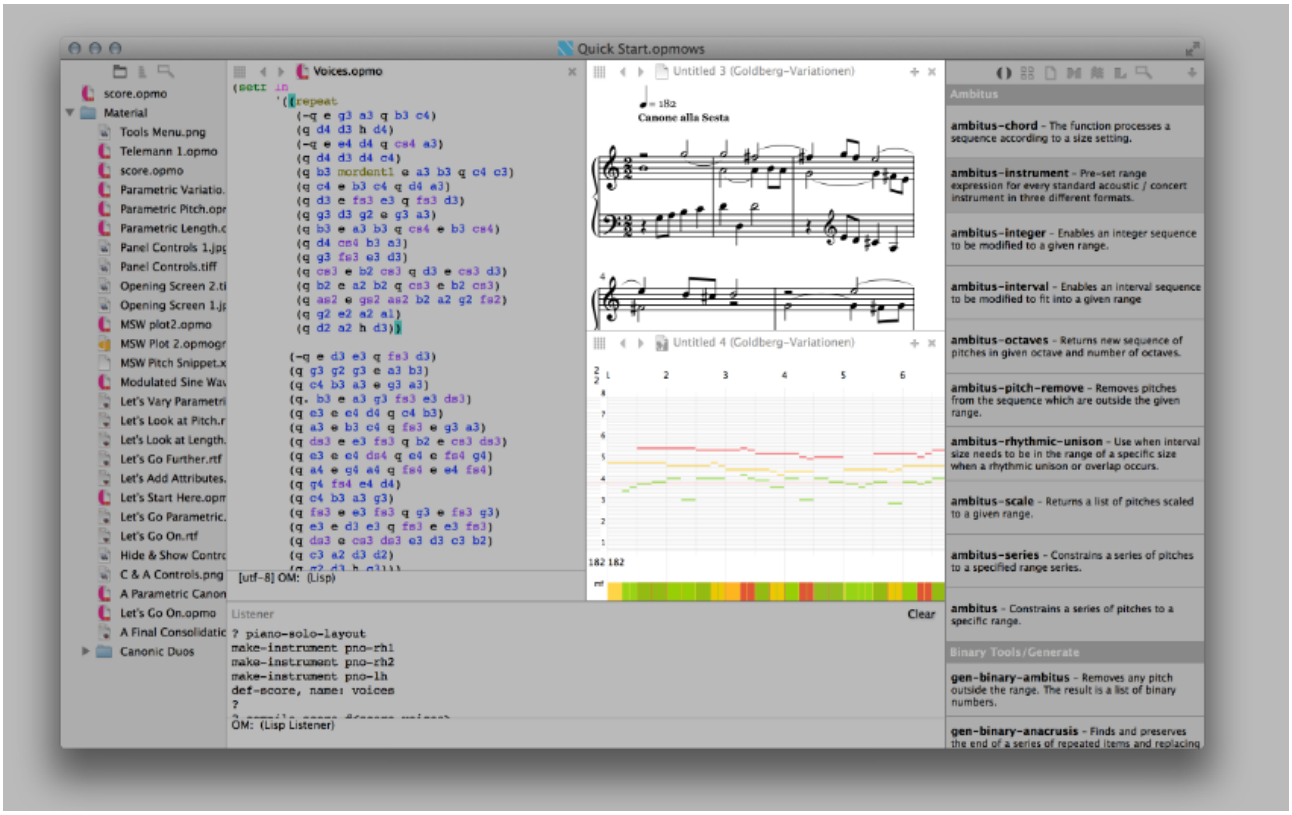


The main role for the Listener is a kind of super script-checker. If the script isn't correct a helpful error message shows up in the panel.



Assistant

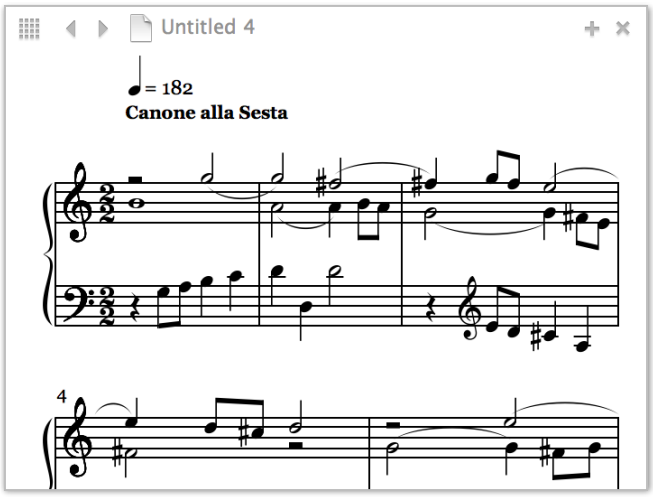
Composing on screen needs more than a single workspace. There are often multiple scripts, texts, audio references, even visual media that play a part in making a new piece of music happen. So Opusmodus calls up the Assistant. This is a panel that can be divided and sub-divided to contain and display pretty much any media. The Assistant is such a flexible space that it can, at a single keystroke command, stand on its own and take over the entire screen!



Imagine working on a multi-movement piece; a composer might want to refer to earlier score scripts, examine and edit a library file, compare a number of System Functions side by side, audition a midifile performance, and look at a score in PDF. This is where the Assistant, 'the composer's assistant' becomes invaluable. It's even possible to multitask: audition with the Midi Player while adding new sub panes and expand the viewing area, and then open up an Internet link.

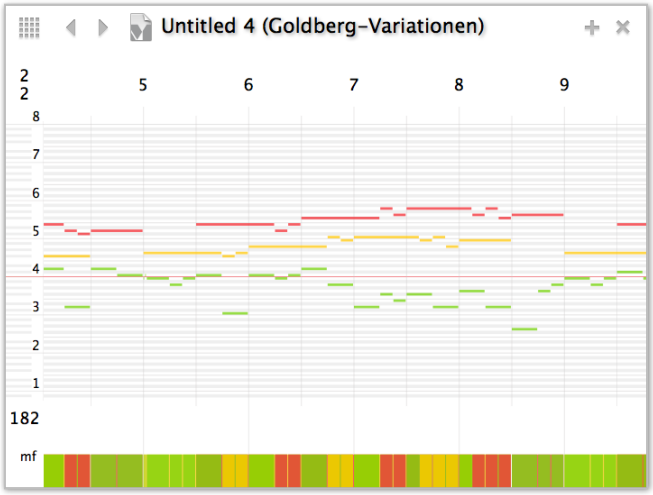
Notation (MusicXML)

That Opusmodus has adopted MusicXML as the de facto standard for displaying notated scores should be no surprise. This is inextricably bound up with the development of the distinctive Opusmodus Notation script (OMN). It gives the composer the means to design into the very composition of a score a host of musical details that have until now been impossible to bring together in a single line of script.



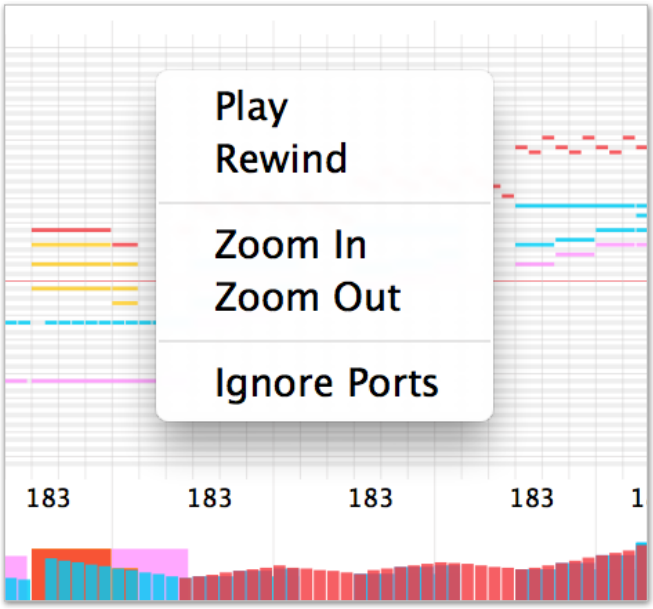
MIDI Player

The Midi Player provides an instant visual guide to the play of pitch, rhythm, duration and velocity (dynamics). Sounding out a composer's script is accompanied by its graphic representation and immediate playback in this Midi Player window. A pitch event's intersection with a bar and beats grid is uniquely colour-coded and matched in a display



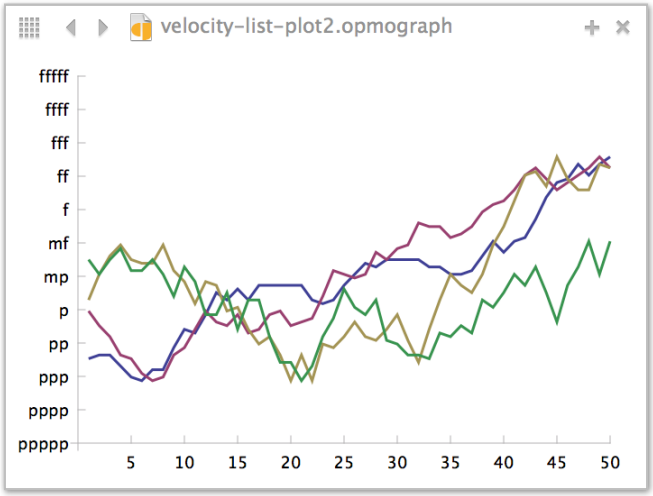
of velocity below the bar and beat grid. **Play** and **Rewind** controls are activated from a contextual menu or from keystroke commands: **Spacebar** for **Play** and **Return** for **Rewind**.

With **Zoom In** and **Zoom Out** one or any number of bars can be viewed in sharp detail. In the top left corner see the time signature display. Where changing signatures are frequent this display changes as the first bar on the left of the window appears. Tempo changes are displayed in a similar way at the bottom of the grid. The instruction **Ignore Ports** allows the composer to choose between the on-board GM sample-player or use personal outboard or inboard sound sources.



Graphs (Plot)

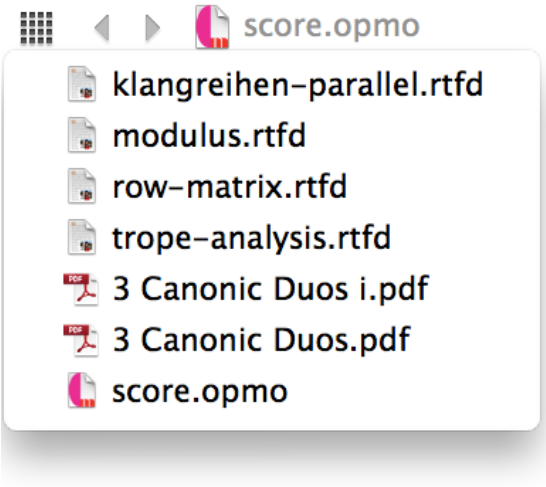
Making 2-D visualisations of musical parameters offer a new way of conceptualisation. Opusmodus graphical tools can plot pitch, rhythms, duration, dynamics and orchestration



and there's a host of different display paradigms available. The composer can now view the interaction of multiple streams of parametric data, a perfect way to take in complex algorithmically-generated material. Composers often use such visualisations in the early stages of a project before precise pitches or rhythms are decided upon.

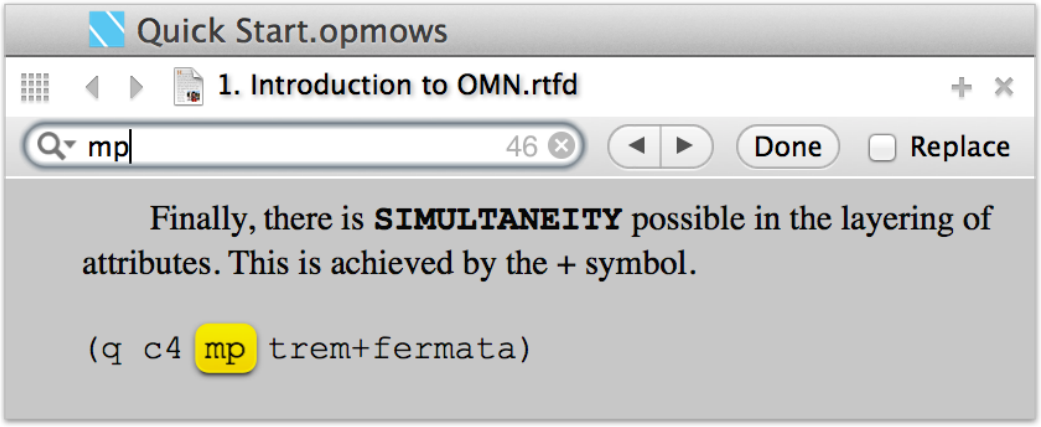
Open Files

In the course of a composing session you may have many files open in both the Composer and Assistant panels. You can see what you have open by clicking the grid icon in the top left of each panel. Use the arrows to browse and display any previously opened material in the Composer or Assistant panels.



Find & Replace

The Find bar is a part of every Assistant panel for text format documents.

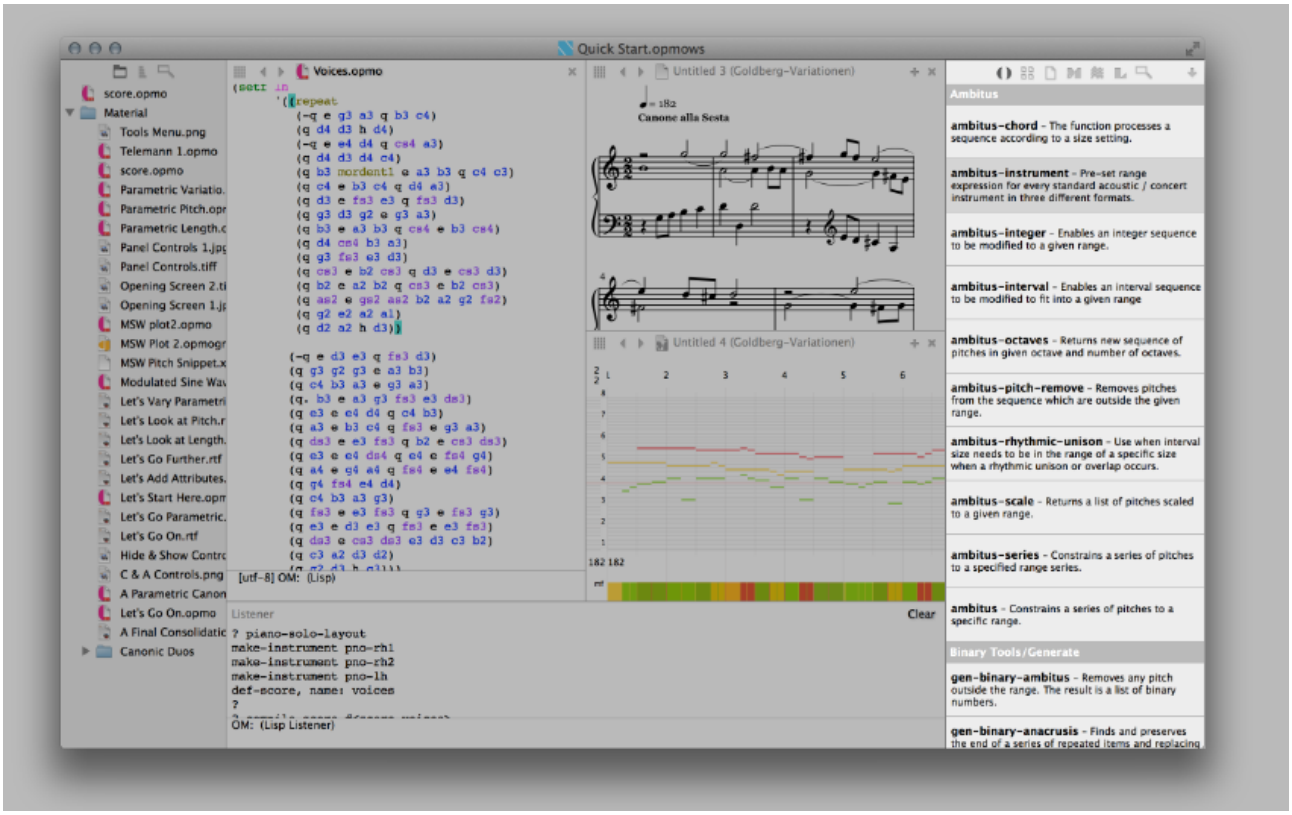


Closing and Duplicating Assistant panel

To close an active Assistant panel you click on the last top-right cross.
To duplicate a panel you click on the first top-right cross.

Utilities

In such a powerful environment as Opusmodus there are just so many things that are not just useful but necessary. The advantage of a digital workspace for a composer is that it can bring together in a single location many, different, and essential things we need to make effective music.

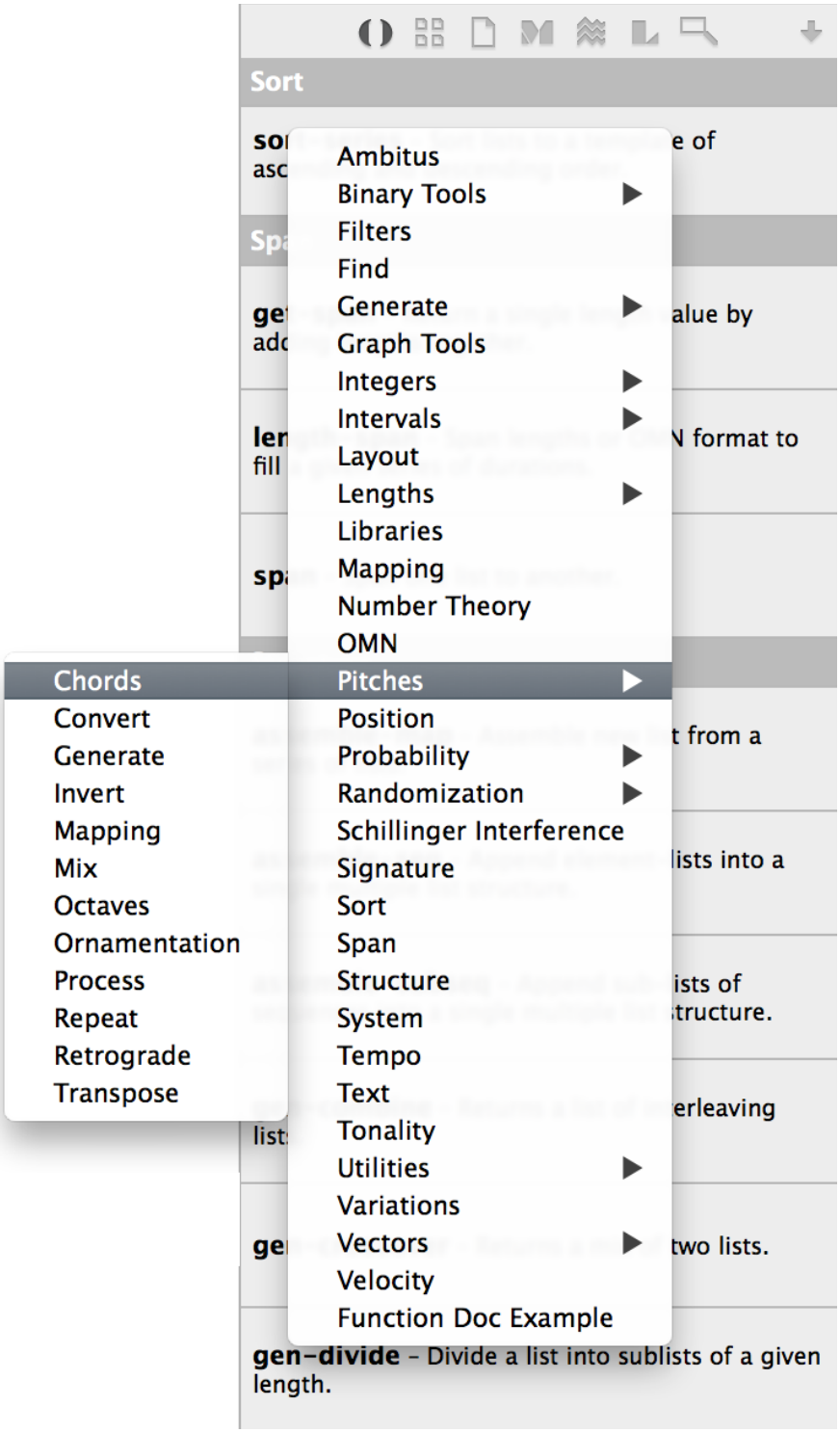


Composition in Opusmodus is supported by hundreds of specialised functions for manipulating musical data. The 30 introductory compositions in the Tutorial Guide - Stages introduce a number of these. The Utilities browser and how to use the function documentation will enable you to realise your own musical ideas.

System Functions

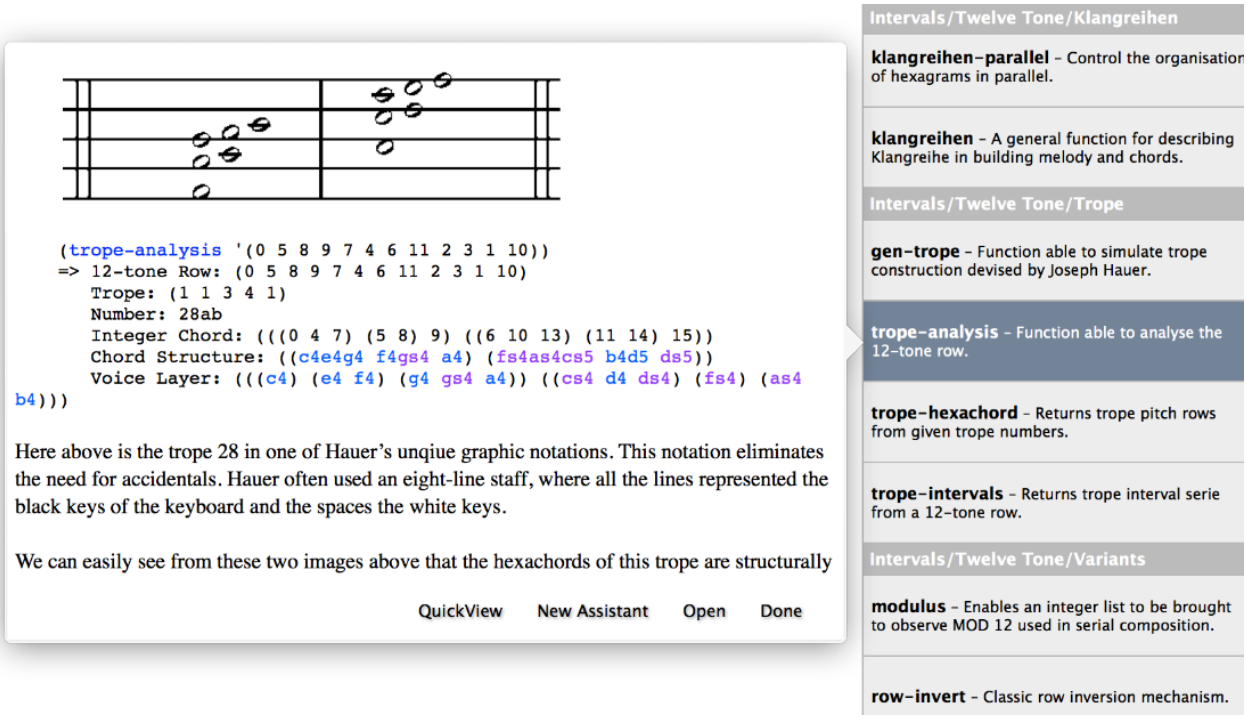
The first icon on the left brings up all the 'help' guidance about the System Functions that form the vocabulary of the scripting language of Opusmodus. To find our way around the many hundreds of words in this dictionary of functions there is a contextual menu: to find, learn about and see / hear examples of what might be useful. By scrolling up and down the list, you will notice that it is organised into groups of functions of similar types. You

can see the hierarchy of this organisation, and quickly locate the types of function you require via a contextual menu accessed by right-clicking on the Utilities panel.

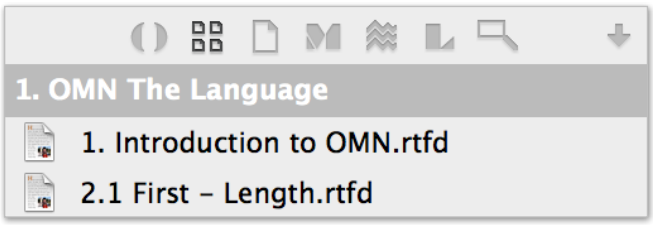


Popover preview

Clicking on any item in the Utilities panel creates a popover preview with a number of further options, depending on the type of file you have selected.

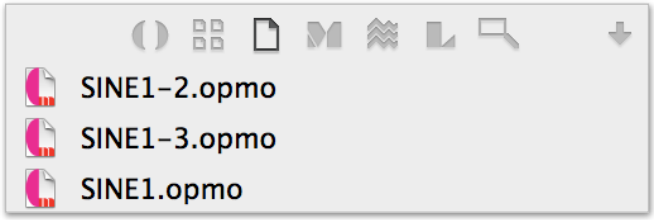


Documentation



Learning to 'speak the language' of Opusmodus doesn't just come from its dictionary of functions. The Documents section of Utilities has specially-written examples of 'HowTo' use these functions in musical situations. Next, discover an extensive section focusing on a range of musical instances from *Ornaments* to *Repeats and Endings* found in Opusmodus Notation (OMN). Finally, there's a reference collection of Opusmodus score scripts and PDFs of notated scores by professional composers working in different styles and contexts.

Scores

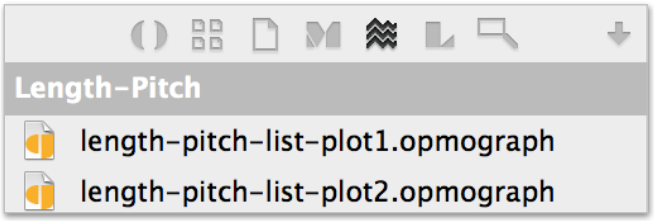


Welcome to your personal Opusmodus score library. Here, anything that belongs to a particular project in scripted code can be saved and archived. Many composers find that producing short sections of score is often the most efficient way of working, so it's really important to have such a utility. Files and folders can be moved, removed, renamed and opened in different workspaces. Notice that the pop-out feature makes it possible to view the score 'on the fly'.

MIDI Files

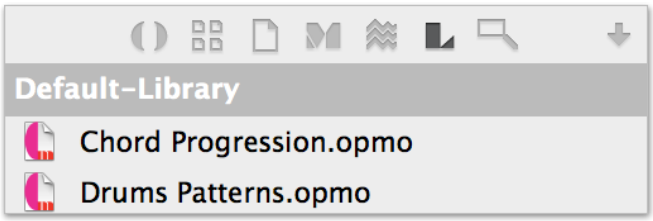
Here is a more personal library space, this time for midifiles. When a script is evaluated it is able to play in the Midi Player or be viewed in Notation without a midifile being saved. A score script can be concluded with an instruction to compile-score, and that means to midifile. The score is now transferable to other software or loaded into the Live Coding Instrument. In some situations saving to a midifile may be quite unnecessary as a MusicXML file is all that's needed to take the script to a dedicated scorewriter.

Graphs (Plot)



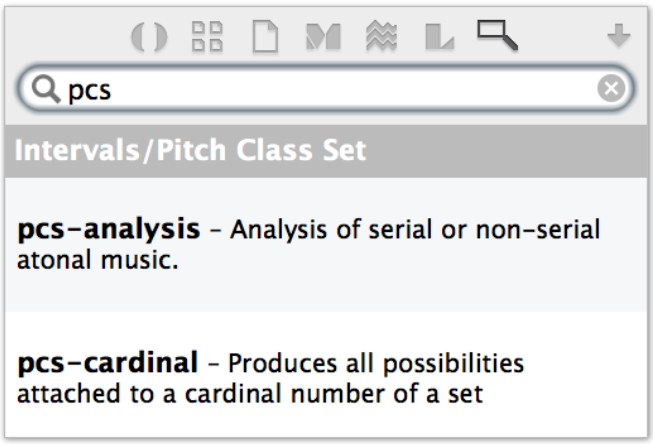
This is an archive of graph material that you have generated with Opusmodus. For many composers to produce graphical representations of musical parameters may prove to be the most significant development found in Opusmodus. These 2-D visualisations in a host of different formats and colours are seen to open up a whole new way of conceptualising and understanding the interaction of parametric data in multiple streams, be it pitch, rhythm, dynamics or structure. It's the perfect way to take in complex algorithmic data. Graphical representations are held in 'plot' files in this dedicated location and can be linked to a composer's working project.

Libraries



Composers working in serious sound design and digital media production for film, TV and web applications often have an arsenal of sound sources on-board and out-board. Opusmodus has a great range of default sound sets – and it's easy to set up your own. Just imagine mapping the output of natural algorithms like white noise to midi controllers. User utilities include 'active' libraries that store retrievable data such as Slonimsky scales and patterns, poetic metrics, and Euclidian rhythmics. This data can be brought into scripts in random or template-controlled forms.

Find



A fast and effective search tool supports the large number of System Functions that make Opusmodus the exciting environment for composing. In script-based composing it is so necessary to reduce the cognitive load – the ability to remember our knowledge so we can understand and work in partnership with software rather than battling against it! This is where robust search tools are essential – no one can remember every detail of several hundred functions when some of these are often rich in variable parameters.

How to Read the Function Documentation

The complexity of the function documentation varies depending on the complexity of the function. Some will be very straightforward to grasp, others might need a little more time. Knowing how to interpret the documentation will help you make sense of even the most complex-looking functions.

Clicking the Open button on the popover opens the function documentation in an Assistant panel. As a demonstration we'll select the **`PITCH-FRAGMENT`** function documentation.

```
pitch-fragment (times range pitch &key (transpose nil)
                (span :pitch) (flat nil) (section nil)
                (seed nil))
```

Immediately we can see that this function **PITCH-FRAGMENT** has a large number of possible arguments and values associated with it. Fortunately, most of these are optional. By looking at the first section of the documentation, the function definition, we can discern which arguments are mandatory and which are optional.

In the above example, the first three arguments (times, range and pitch) are required to successfully run the function. The additional optional elements of the function are defined as a series of 'keywords', following the &key marker. You will notice that each of these has a default value associated with it – usually nil – which can be overridden if desired.

What each of the arguments and keywords actually means is detailed below the function definition in the **Arguments and Values** section of the document:

Arguments and Values:

times	an integer or list of integers.
range	an integer list.
pitch	a list or list of pitches.
transpose	a list of integers (transposition values).
section	a list of integers (sublists to process).
seed	NIL or an integer. The default is NIL.
flat	NIL or T. If true, the OMN single type lists are flatten. The default is NIL.
span	:length, :pitch or :velocity. The default is :pitch.

The expected data types are shown for each argument, along with an explanation of the types of data that can be used with the optional keywords. Note that a number of the functions also have an OMN section. This indicates extended functionalities for processing lists in the OMN notation format.

Each documentation file also includes a detailed description of the function, along with numerous examples of varying complexity illustrating more advanced applications of the function.

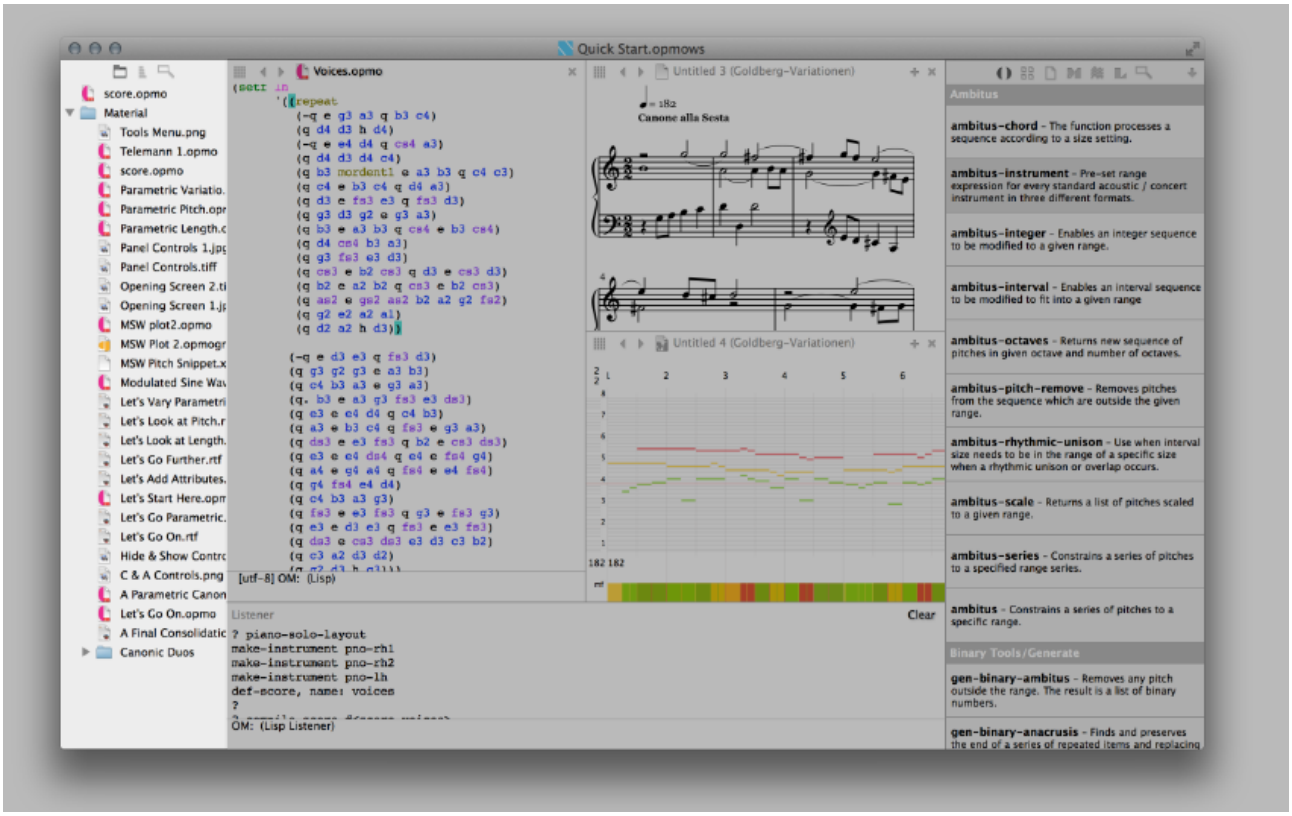
Evaluating and Auditioning Documentation

As with material in the Composer panel, anything in an Assistant panel can also be evaluated and auditioned as an aid to understanding how the functions work. By selecting an expression and using the **Evaluate Expression (⌘E)** or **Audition OMN (⌘1)** you will be able to see and hear the output of the function:

```
(pitch-fragment 4 '(3) '(c4 d4 e4 f4 g4 a4)
                :transpose '(1 -1 2 -2))
=> (gb4 ab4 bb4 e4 gb4 ab4 g4 a4 b4 eb4 f4 g4)
```

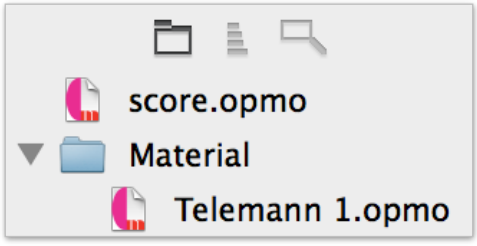
Navigator

Writing score scripts in Opusmodus begins in a similar way to composing on paper. It's equally messy. But in a virtual workspace we can keep everything in place that connects with first thoughts and experiments, no matter what the format. The three sections of the Navigator help make that journey towards music possible.



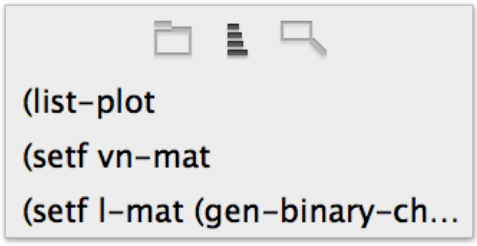
Finder

The Navigator opens a script into the Composer and/or as many other documents in the Assistant panel.



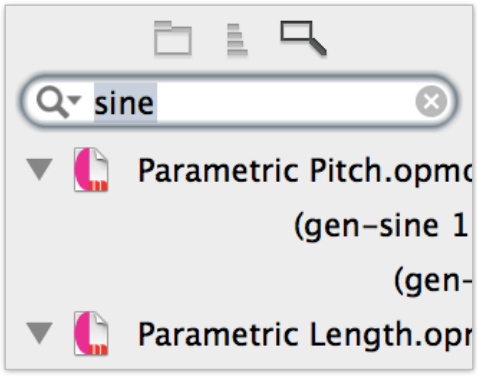
Definition

The Definition icon activates the display linked to the score script current in the Composer and enables a search for a particular definition of an expression. When it's identified the mechanism places a green pair of cursors highlighting each end of the expression. If it is a long score this feature can be an invaluable 'navigation' aid!



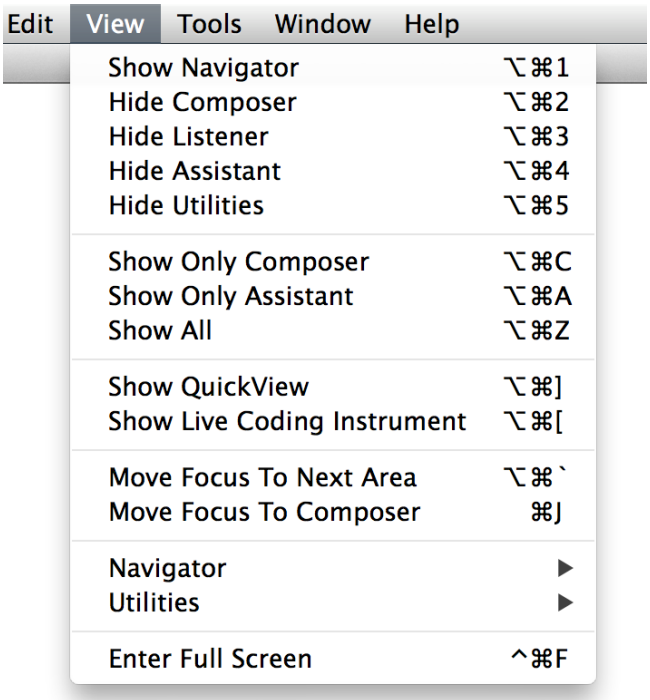
Find & Replace

Further help to navigation is found with the Search Tool. It can and does search inside every file and folder placed in the Navigator. This is wonderful for searching out instances of use (and reuse) of particular System Functions. In the illustration below there is a search for a word "sine". The pointer selects one instance and immediately the file in which it belongs appears in the Composer.

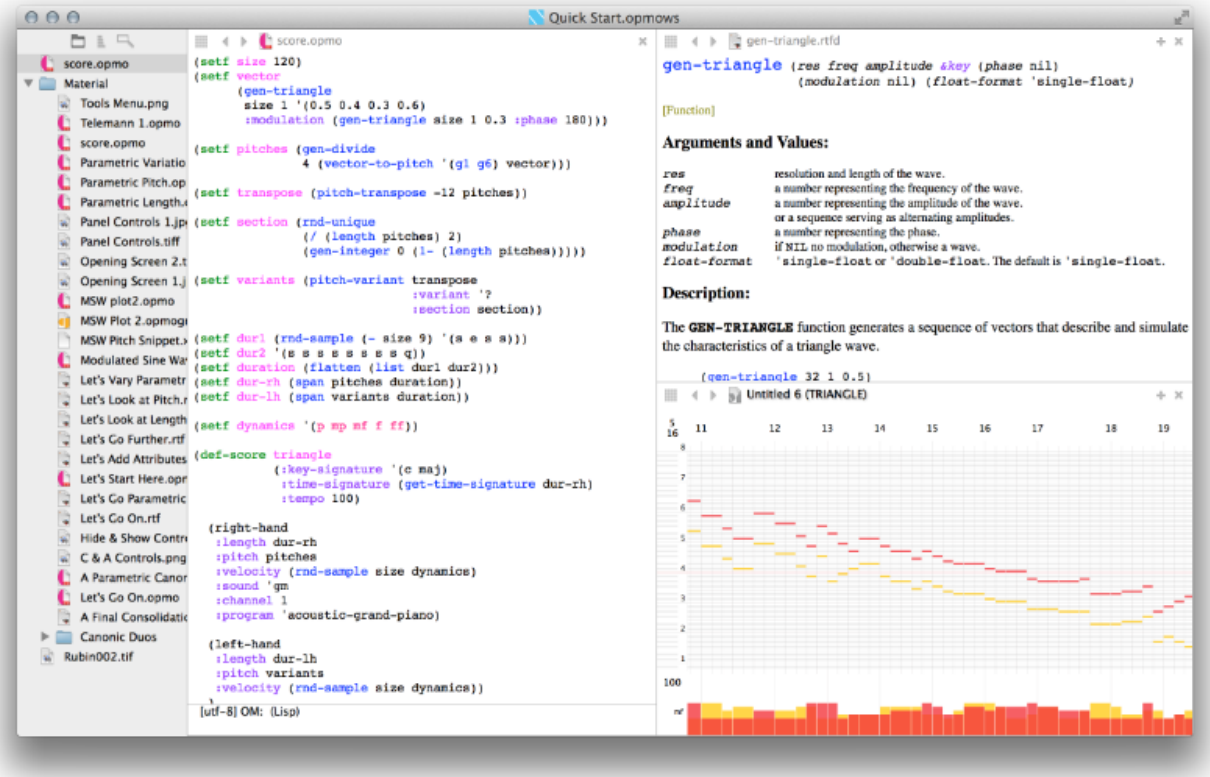


View

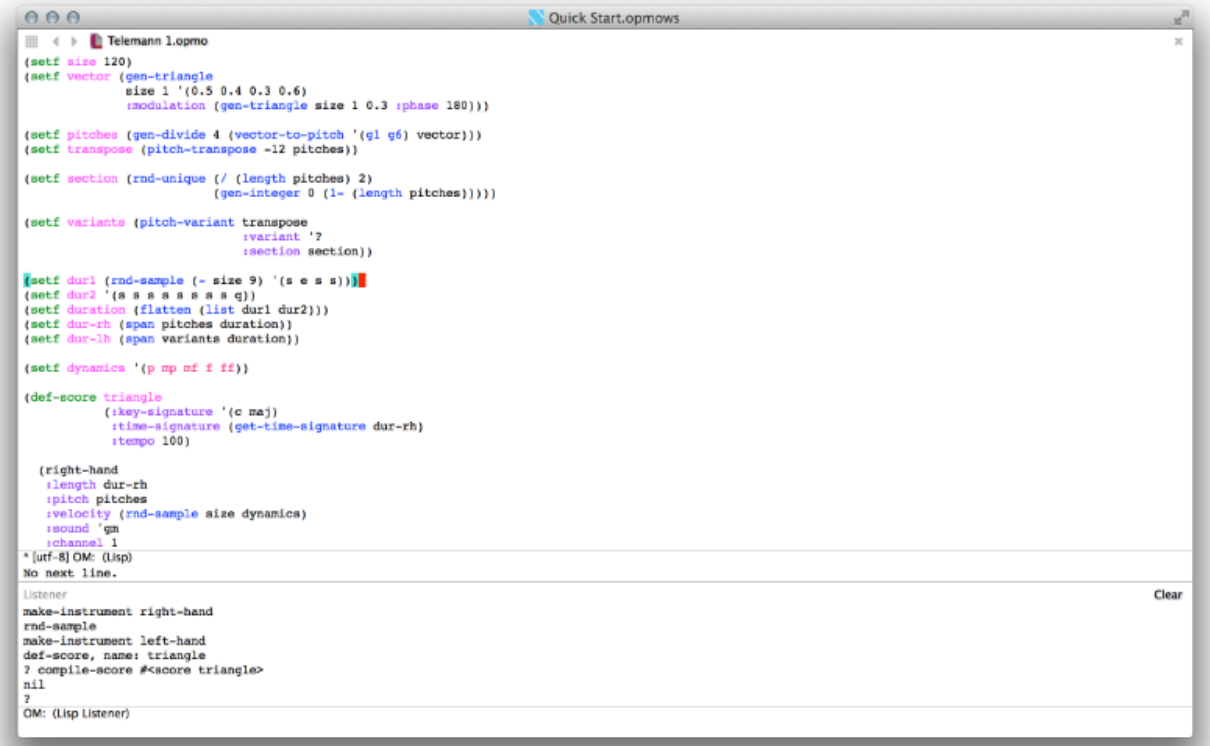
At any time you can tailor the display to your needs by hiding the panels. You can show and hide any of the panels by using the View menu at the top of the screen.



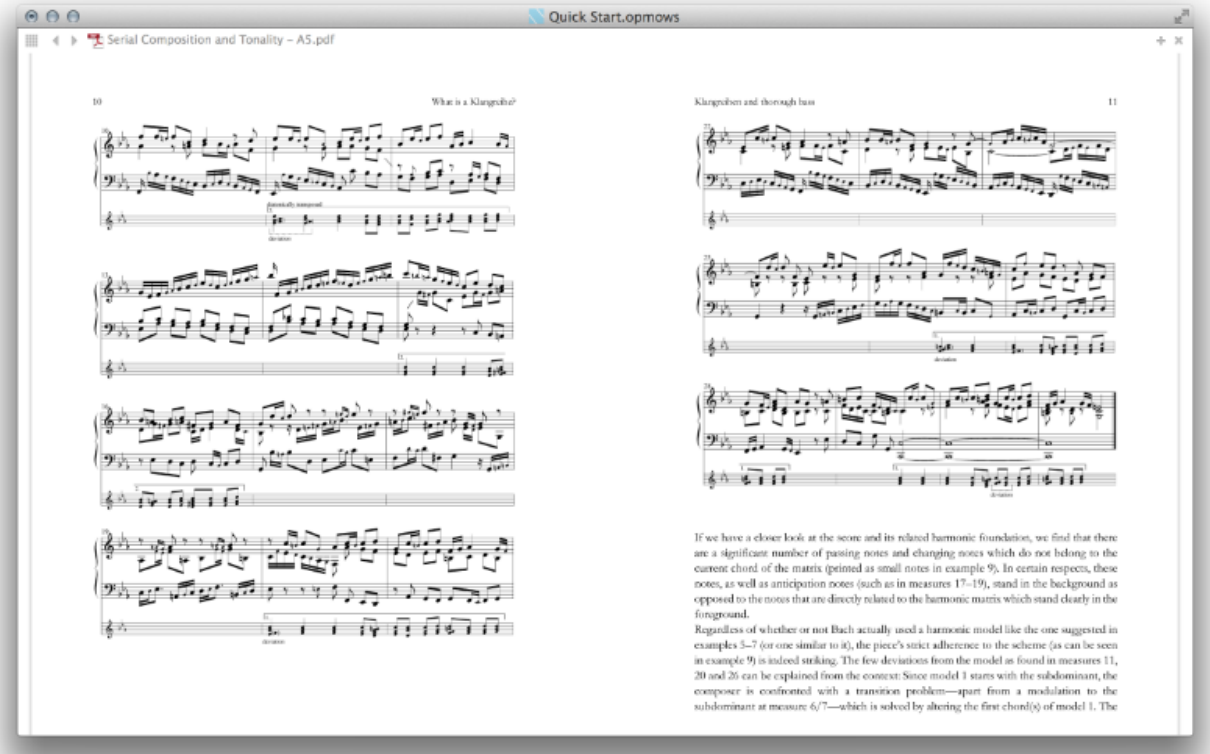
Here, the Listener and Utilities panels have been hidden:



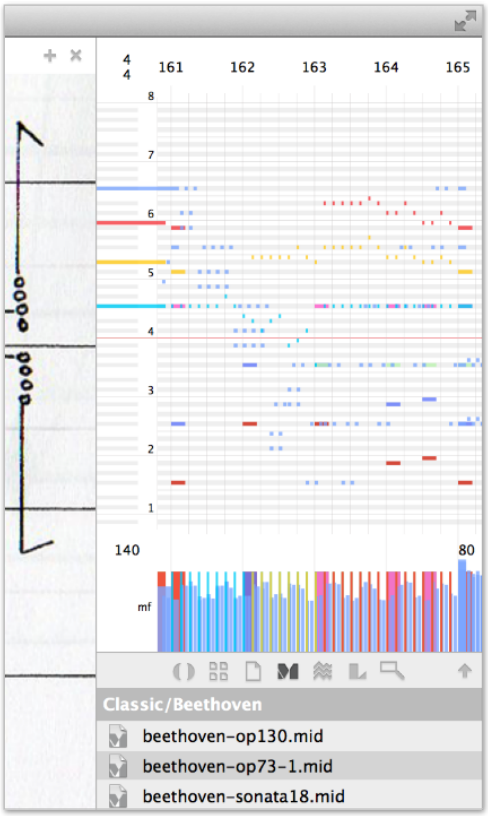
Here, the Composer and Listener panels are shown only:

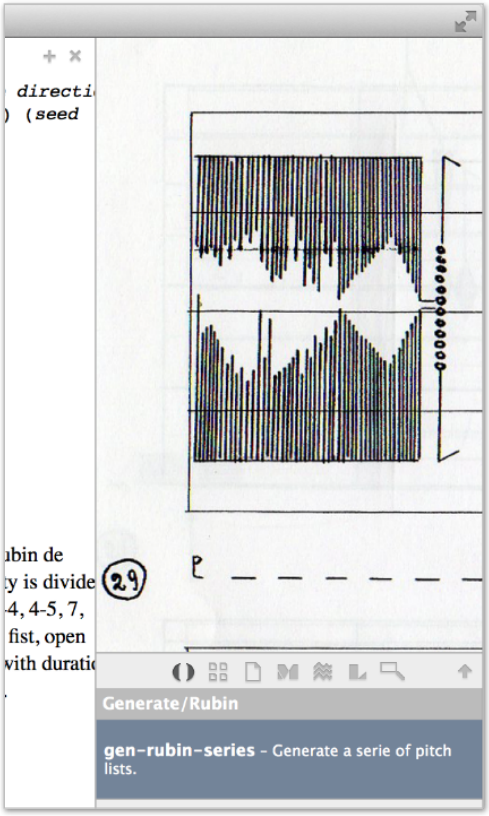


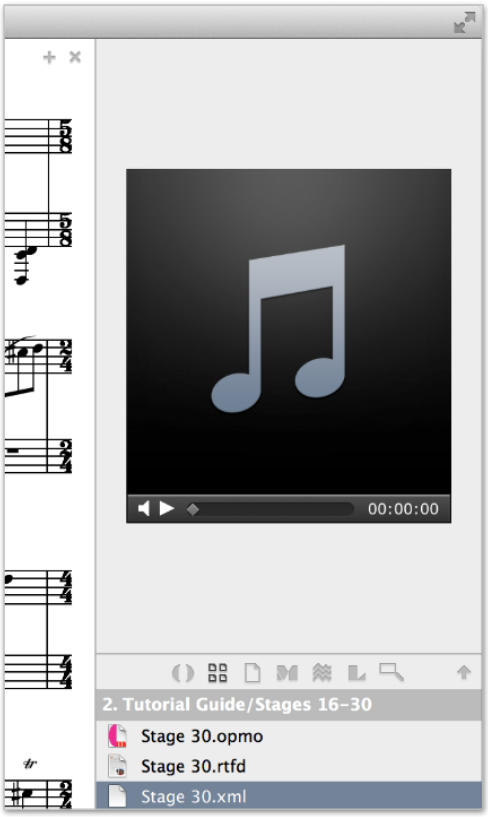
Here the Assistant panel is in the ‘presentation’ mode display:

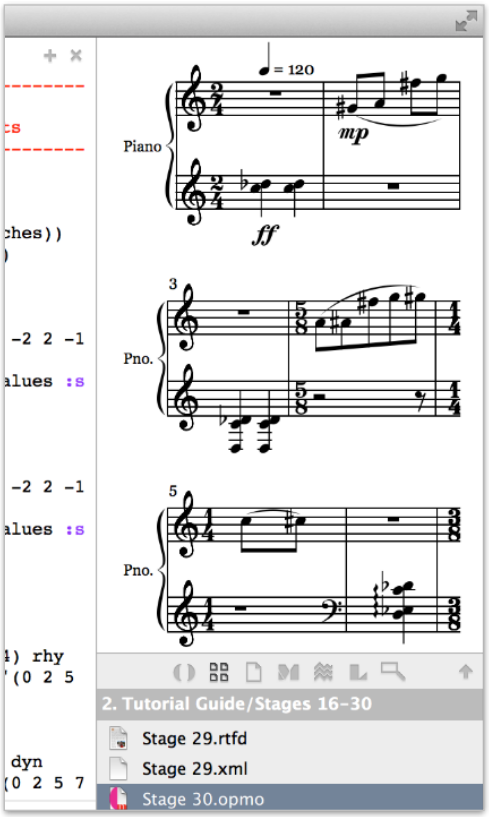


QuickView



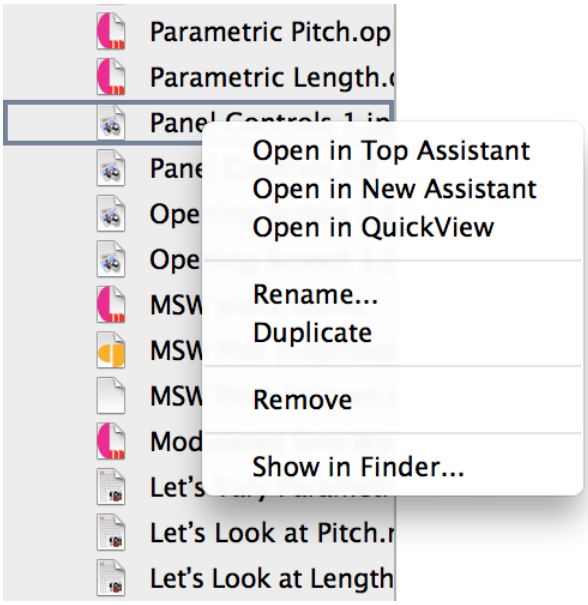




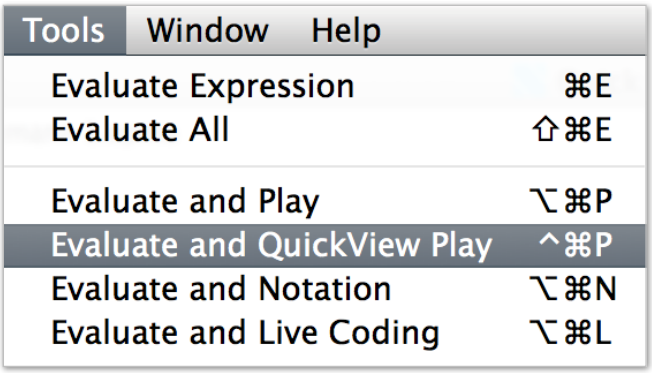


The QuickView area is a flexible multi-application space occupying the top half of the Utilities panel. It can show the Midi Player, the Live Coding Instrument (LCI), PDF, text or image files. Opening the Midi Player in the Assistant can break up the pattern of workflow in a session, so there's a contextual menu alternative to take the Midi Player into the additional space of QuickView. When working on a complex orchestral score composers may find themselves needing to script in both Composer and Assistant panels. By leaving the Assistant free and to hear and see results quickly the use of QuickView area can relieve the log jam!

Via a contextual menu accessed by right-clicking on the Navigator panel you can open any file with the command **Open in QuickView**.



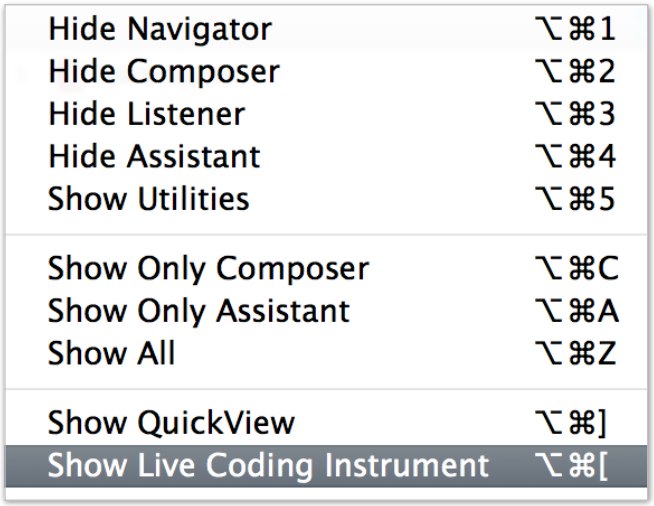
A score can be defined and played in the QuickView panel via the **Tools > Evaluate and QuickView Play** menu.



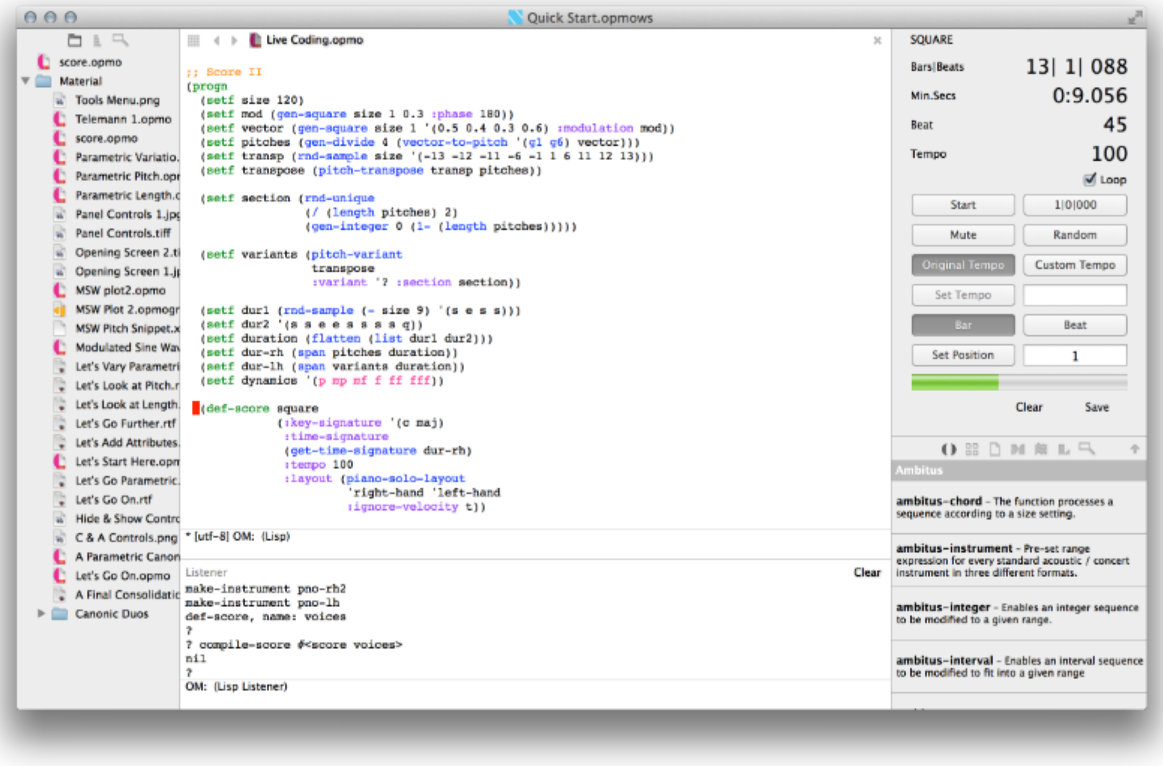
The popover window can open a file in the QuickView panel via a click on the **QuickView** button.

Live Coding Instrument

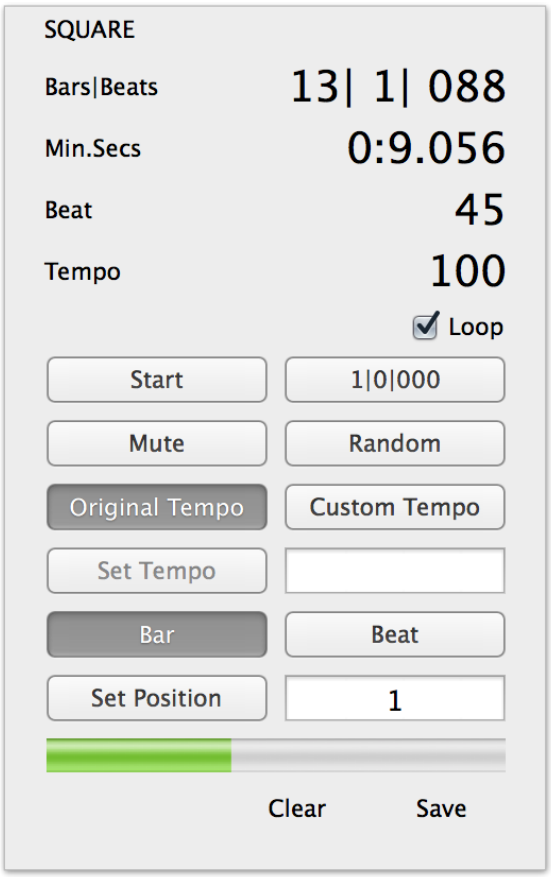
To show LCI in QuickView panel can be done via the **View > Show Live Coding Instrument** menu.



The Live Coding Instrument in the QuickView panel (top right) can explore in realtime the further potential of your own script or midifile recording. Performing with LCI allows two modes of interaction: with the scripted code itself, and with the buttons and slider of the LCI interface. Live Coding is sometimes called 'on-the-fly programming' or 'just in time scripting'. It is a scripting practice centred on the use of improvised interactive programming.



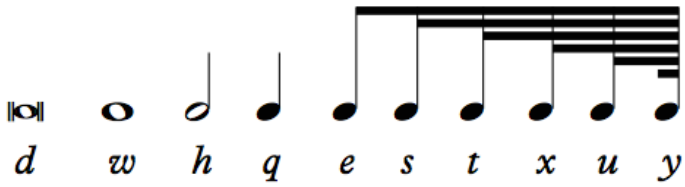
The LCI gives the composer an intuitive control panel and the possibility of working in true live coding style directly with the script. In practice composers who use the Live Coding Instrument often begin with a script, make a change, then 'playing' that change from the buttons of the LCI control panel.



OMN (Opusmodus Notation)

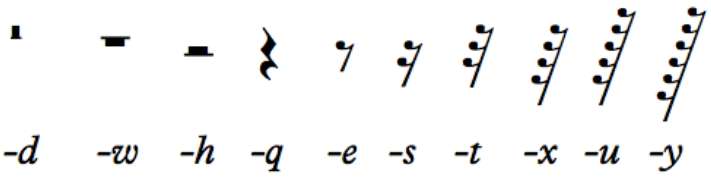
While many lists might consist of only one type of data, such as musical pitches, Opusmodus Notation (OMN) allows lists to contain fully realised fragments of music. This means that musical material becomes significantly easier to manipulate, read and edit, once the essential syntax of OMN has been understood.

Note-Length



Here are the standard values of note-lengths. The most commonly-used are represented in the OMN language by the first letter of their American arithmetic name, so **`w`** is a whole-note, **`h`** is a half-note, **`q`** is a quarter note and so on.

Rest-Length



The prefix of a (minus) sign, so **`-w`** is a whole-note rest, **`-h`** is a half-note rest, **`-q`** is a quarter note rest. To assist with multiple rests `(-3)` will produce 3 bars of whole-note rests.

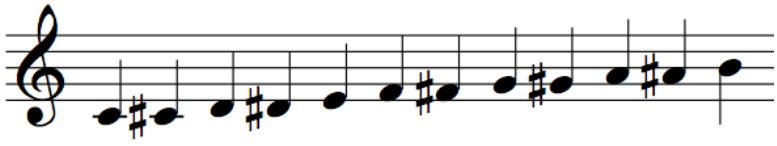
Here is a list of three quarter-notes. The list has to begin with a `'(a quote)` and be enclosed by parentheses `( )`.

```
'(q q q)
```

Pitch

In OMN a pitch is written as a text symbol that combines a note's lower-case letter name with its octave number. OMN uses the convention that `c4` is 'middle c' , so numbered because of the note's position as the fourth C key on a standard 88-key piano keyboard. In fact the composer can go a little beyond the standard keyboard range because OMN takes in the MIDI range of 0-127 pitches.

```
'(c4 cs4 d4 ds4 e4 f4 fs4 g4 gs4 a4 as4 b4)
```



Writing OMN

A simple list of pitches, assigned to the variable melody:

```
(setf melody '(c4 gs4 as4 g4 fs4 e4 ds4 c4))
```

This list is already potentially OMN, and we can freely add in other musical parameters without needing to resort to any other operation to convert the list to OMN.

Adding Lengths

In OMN, note lengths are indicated by letters:

```
t = 1/32   s = 1/16   e = 1/8
q = 1/4    h = 1/2    w = 1
```

Any of these values can also be dotted, or double dotted, for example: `e..` In OMN the duration is always the first piece of musical data we write.

We can add some rhythm to our basic melody by adding OMN lengths:

```
(setf melody-omn '(q c4 e gs4 as4 q g4 e fs4 q e4 e ds4 h c4))
```

Note that we only need to enter a duration when there is a change in rhythm. In the above example, `(e gs4 as4)` is the same as writing `(e gs4 e as4)`.

Adding Rests

The length values can also function as rests by prefixing them with a minus sign, for example:

```
(setf melody-omn
      '(q c4 e gs4 as4 -q g4 e fs4 -q e4 e ds4 -q h c4))
```

Note that the pitches following a rest will be played using the same length as the preceding rest. Therefore `(e gs4 as4 -q g4)` is the same as writing `(e gs4 e as4 -q q g4)`.

Adding Dynamics

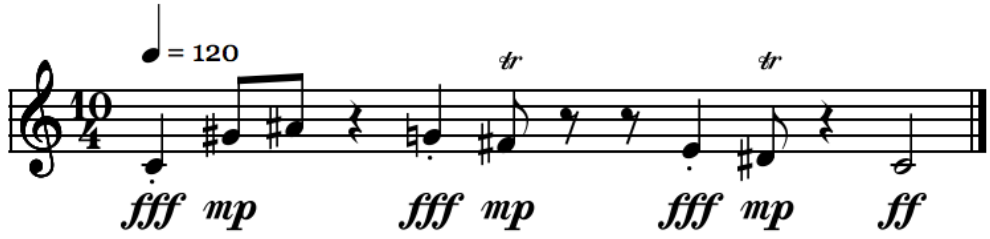
Dynamics in the range `ppppp` to `fffff` can be integrated into OMN notation. A dynamic always comes after the pitch:

```
(setf melody-omn
      '(q c4 ff e gs4 mp as4 -q g4 ff e fs4 mf
        -q e4 ff e ds4 mp -q h c4 ff))
```

Adding Articulations

Finally, many articulations can be added to OMN notation. Articulations are not only heard in Audition, but are also accurately notated when displayed in MusicXML. An OMN articulation usually comes last, following any pitch or dynamic marking. Here are some articulations `stacc` (staccato) and `tr2` (trill) added to the OMN fragment, which has also been re-formatted to ease reading:

```
(setf melody-omn
      '(q c4 fff stacc e gs4 mp as4
        -q g4 fff stacc e fs4 mp tr2
        -q e4 fff stacc e ds4 mp tr2
        -q h c4 ff))
```



Disassembling and Making OMN

OMN can easily be disassembled into separate lists of pitches, lengths, dynamics and articulations so that lists can be processed separately or combined with new materials.

Using the **DISASSEMBLE-OMN**, you can view every parameter in the listener. These parameters can be pasted back into the composer for further editing:

```
(disassemble-omn melody-omn)
=> (:length (1/4 1/8 1/8 -1/4 1/4 1/8 -1/4 1/4 1/8 -1/4 1/2)
    :pitch (c4 gs4 as4 g4 fs4 e4 ds4 c4)
    :velocity (fff mp mp fff mp fff mp ff)
    :articulation (stacc - - stacc tr2 stacc tr2 -))
```

Alternatively you can use the OMN function to extract the parameters to variables, useful if working from, for example, an imported MIDI fragment or from freely composed OMN:

```
(setf length-ex (omn :length melody-omn))
(setf pitch-ex (omn :pitch melody-omn))
(setf velocity-ex (omn :velocity melody-omn))
(setf articulation-ex (omn :articulation melody-omn))
```

Parametric lists can also be easily recombined using the **MAKE-OMN** function. If, for example, we wanted to reverse the order of the lengths in our original OMN fragment and then create a new OMN list, we could do the following:

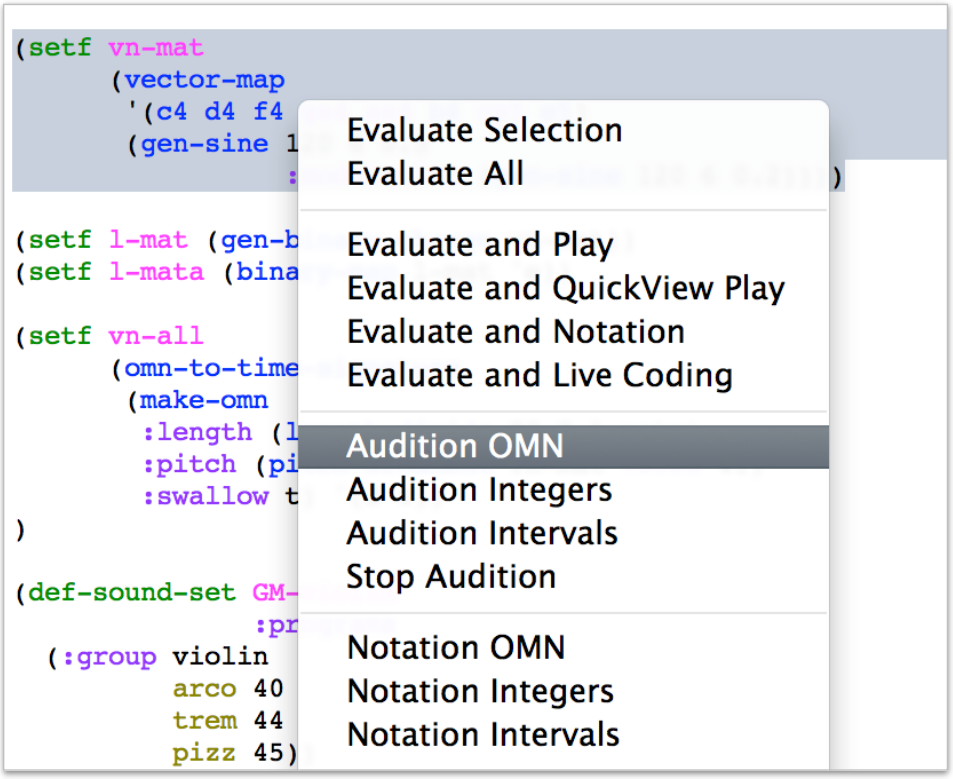
```
(setf length-ex2 (gen-retrograde length-ex))
```

This reverses the order of the original lengths using **GEN-RETROGRADE**, and then recombines the new lengths with the original material using **MAKE-OMN**.

```
(setf melody-omn2
      (make-omn
       :length length-ex2
       :pitch pitch-ex
       :velocity velocity-ex
       :articulation articulation-ex))
=> (h c4 fff stacc -q e gs4 mp q as4 - e g4 fff stacc
    q fs4 mp tr2 - e e4 fff stacc ds4 mp tr2 q c4 ff)
```

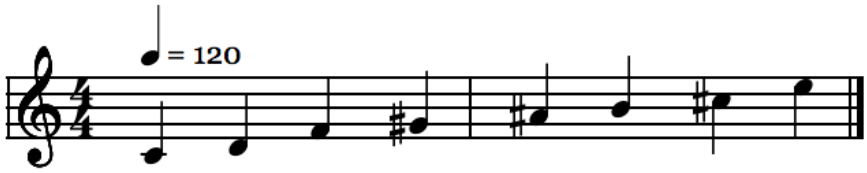
Snippet

There are two types of snippets. One is the Audition snippet which will play any selected expression, list or fragment of your material, the second is the notation snippet. To audition an expression for example, you highlight it (select) and then via contextual menu by right-clicking on your mouse you select the command **Audition OMN (⌘1)**.



The same process is used to show a notation snippet, but this time we select **Notation OMN (⌘2)**. The snippet is designed to deal with three different inputs, OMN, integers and intervals.

```
(vector-map
 '(c4 d4 f4 gs4 as4 b4 cs5 e5)
 (gen-sine 120 6 0.5
  :modulation (gen-sine 120 6 0.2)))
```



Output

Opusmodus can export your compositions to a number of formats including MIDI data, MusicXML notation and Midi to Score. However, to export your music it is necessary to first set up a score definition.

For any evaluation of data within the Composer, Assistant or QuickView panels it is essential for the chosen panel to be 'active'. This can be done with a mouse click inside the chosen panel or by using the menu items **Move Focus to Next Area** or **Move Focus to Composer**.

Score Definition Template

A score definition is created using **`DEF-SCORE`**. It is here that you can bring together all your composed materials and assign them to MIDI tracks and instrumentation. A basic score definition might look like this:

```
(def-score Score-Name
           (:title "Your Title"
            :composer "Your Name"
            :copyright "Copyright Information"
            :key-signature '(c maj)
            :time-signature '(5 8)
            :tempo '(q 120))

  (instrument
   :omn melody-omn
   :channel 1
   :sound 'gm
   :program 'acoustic-grand-piano))
```

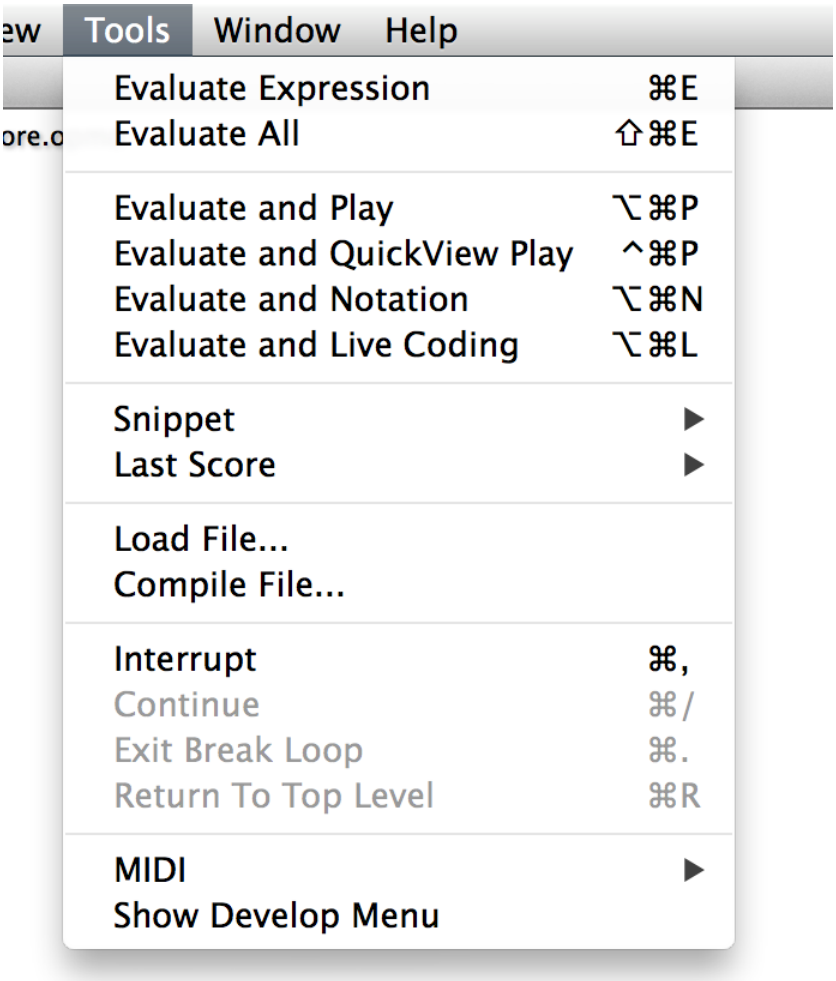
In the above example, *Score-Name* is a variable and can be anything you choose. You could even have multiple def-scores in the same file for different arrangements of your piece.

Following the variable name is the score header, which contains the high-level information about the score such as the tempo, key signatures and time signatures. Note that these can also be lists of values allowing multiple changes in tempo, time and key signatures.

The remaining sections of **`DEF-SCORE`** comprise of definitions for each instrument in the piece. In the above example all the musical data has been turned into OMN. However, you can still use `:pitch`, `:length` and so on to provide discrete lists for each track.

MIDI Playback

Once a score definition has been created, it can be played via the **Tools > Last Score > Play** or with **Evaluate and Play** menu item.

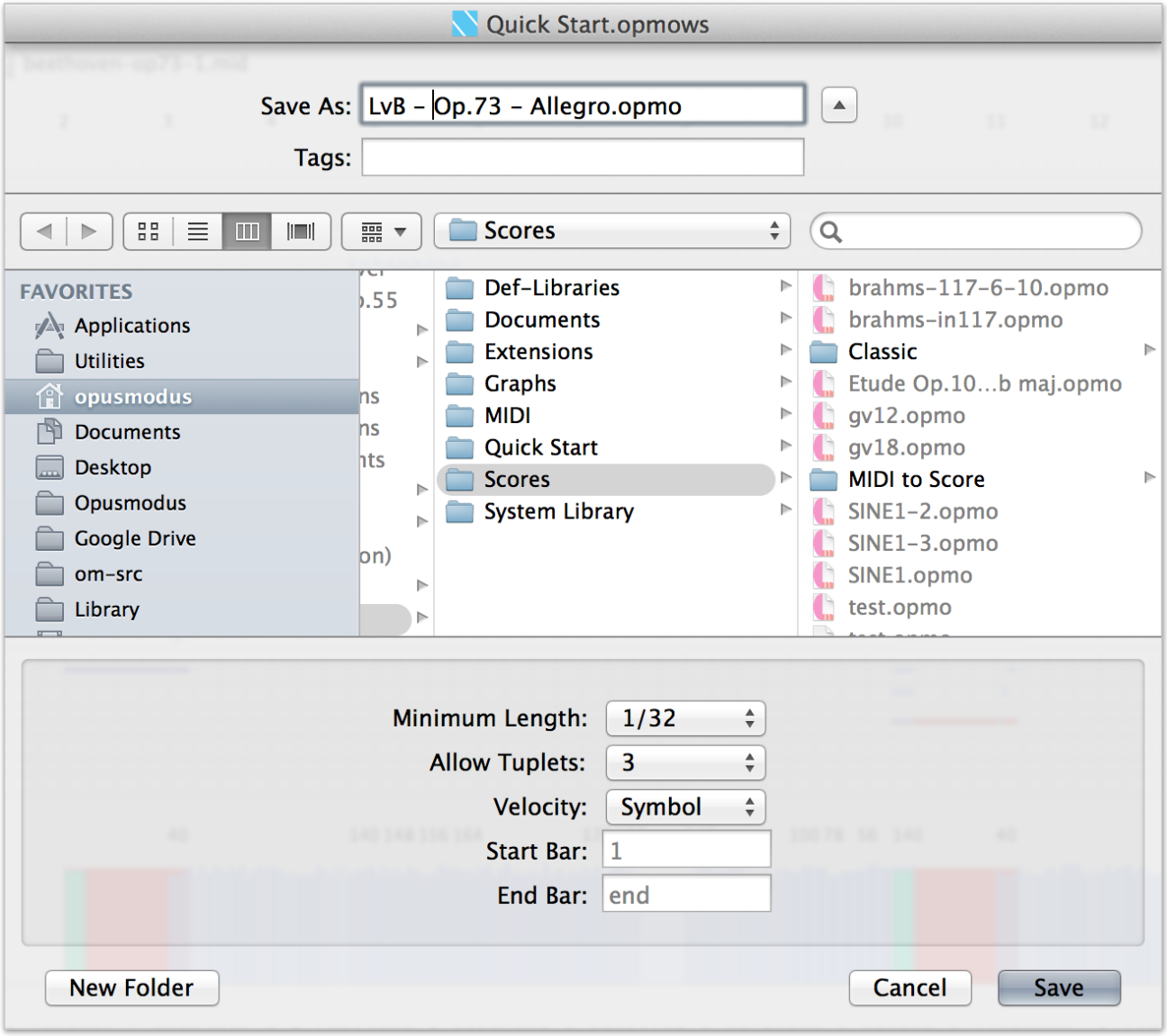


You can also preview your composition notated in standard notation within Opusmodus via the **Tools > Evaluate and Notation** menu item. This will open up a new Assistant showing the preview of your score.

Last Score

You can export a MIDI file via the **File > Export > Last Score to Midi…** menu item. In this instance the last evaluated **`DEF-SCORE`** section will be compiled into a MIDI file. All MIDI files will be saved into Opusmodus' MIDI folder. Specifying an additional folder name will create a new folder and output the score in that location. The same goes for transferring your score-file to a MusicXML file. **File > Export > Last Score to MusicXML…**

MIDI to Score

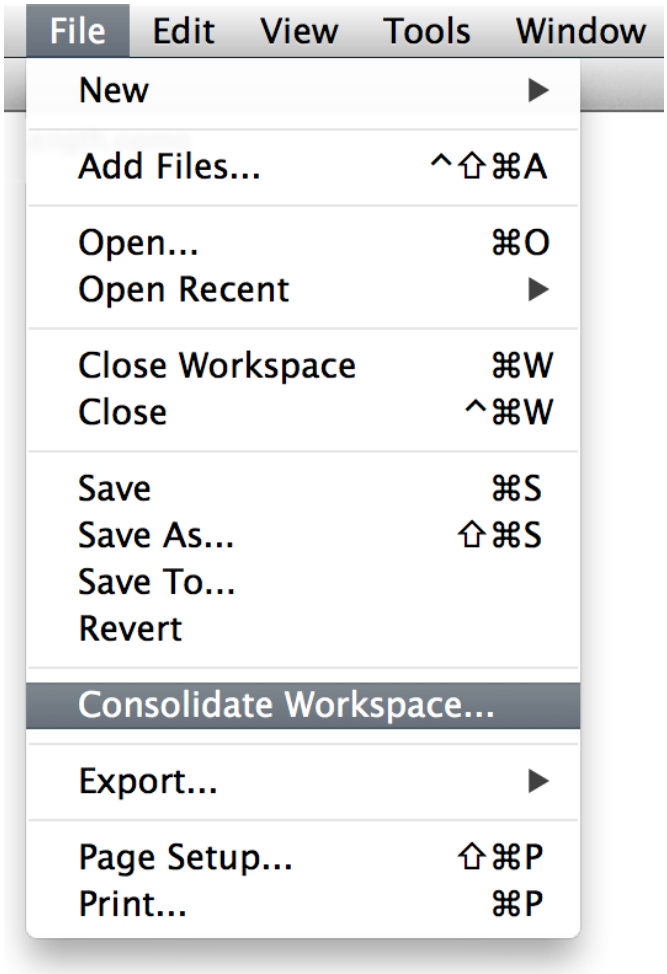


A new score doesn't have to begin from working with Opusmodus. A composer's improvisation can be recorded and captured to midifile; in making music for film, TV or web production composers often source a style or atmosphere from one of the many midifile libraries on-line. Opusmodus provides a unique Export feature in its file menu. This allows a midifle to be converted into an .opmo file format with every detail faithfully transcribed.

The Export dialogue box for **Midi to Score...** shows quantisation, elimination of dynamics and the invaluable 'section capture' feature as in-built additions. Now you can take that fragment from a Vivaldi concerto and extend it with Opusmodus functionality.

Consolidate Workspace

Finally, the **Consolidate Workspace** command is able to create a copy with a dedicated folder of all the files in a particular workspace (project).

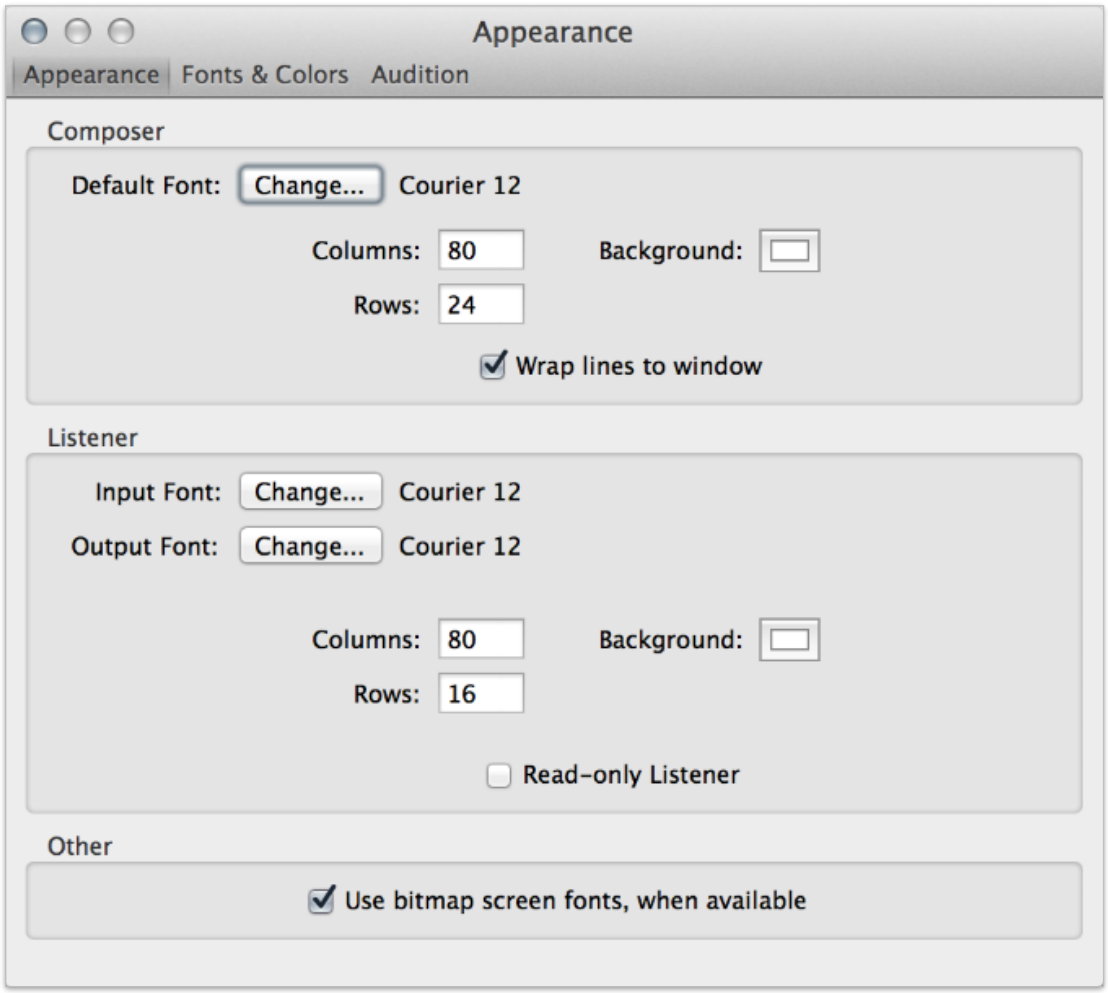


Preferences

In Opusmodus Preferences there are three window sub-panels in which two are for Composer and Listener setup, and the third for Audition snippet setups.

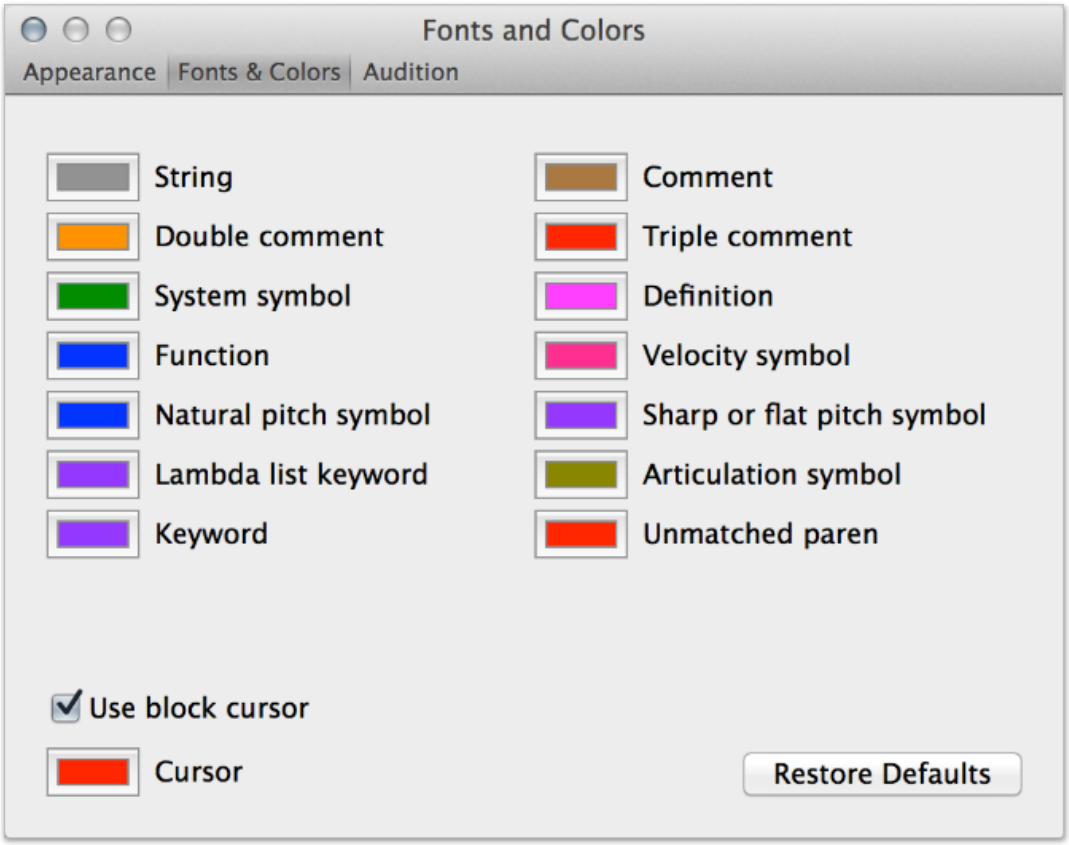
Appearance

Here you can set the Composer and Listener default font and font size.



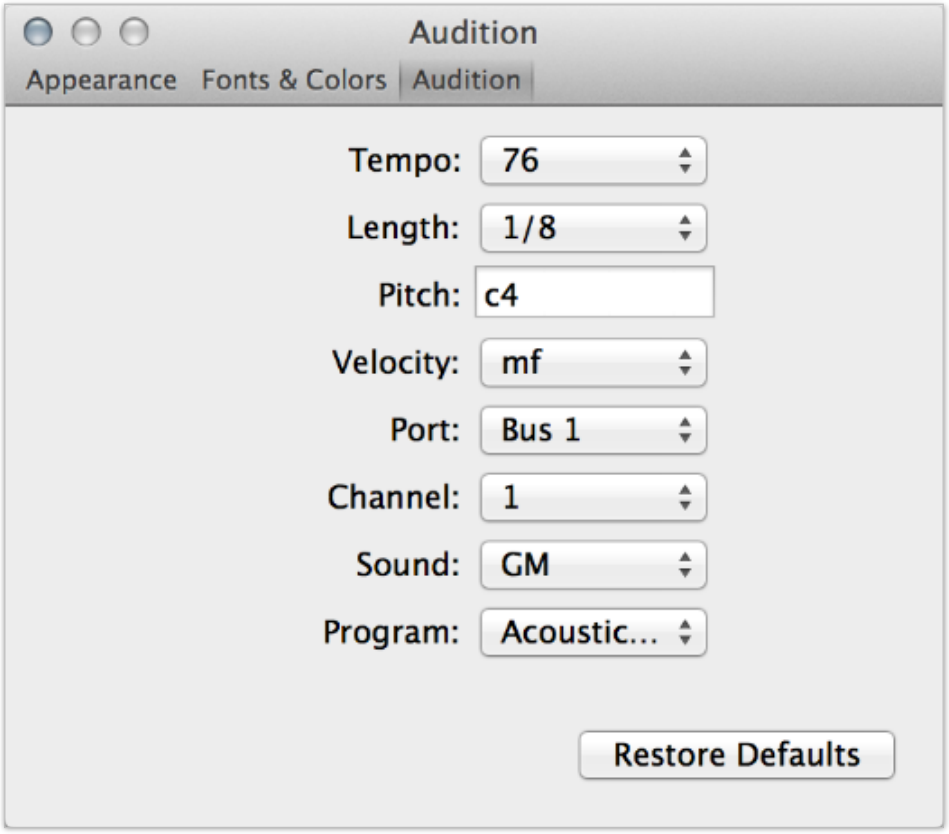
Fonts & Colours

The colourisation is very valuable preference. Here you can assign your own choice of colour to the Definition, Function, Articulation etc… for better reading and writing, particularly in those scores written in OMN.



Audition

Here we set the default conditions for Audition snippet.



Written by Nigel Morgan, Janusz Podrazik and Phil Legard