



Nigel Morgan
Janusz Podrazik

Quick Guide

Introduction to Opusmodus

Contents

Graphical User Interface	1
Workspace	2
Navigator, Composer, Listener, Assistant und Utilities	2
Composer	3
Find and Replace	4
Listener	5
Assistant	6
Closing and Duplicating Assistant panel	7
Open Files	7
Notation (MusicXML)	8
MIDI Player	9
Graphs (Plot)	10
Utilities	11
System Library	12
Popover Preview	13
Documentation	14
Scores	15
Media (Midi Files)	16
Graphs (Plot)	17
Libraries	18
Search	19
How To Read The Function Documentation	20
Evaluating And Auditioning Documentation	21
Navigator	22
Finder	23
Definition	24
Find	25
View	26
QuickView	29
Live Coding Instrument	32
Snippet	34
Output	35
Score Definition Template	35
Midi Playback	36
Last Score	36
Midi To Score	36
Consolidate Workspace	37
Preferences	38
Appearance	38
Fonts & Colours	39
Audition	40
Notation	41

Graphical User Interface

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.

The screenshot shows the Opusmodus interface with four main panels:

- Navigator (Left):** Shows a file tree with various Opusmodus files (e.g., OMN Notation, Scores, MIDI) and system files (e.g., Welcome to Opusmodus.pdf).
- Composer (Top Center):** Displays a code editor for a file named "2. Full score.opmo". The code contains musical notation and algorithmic definitions. A yellow diagram to its right illustrates the workflow: an "Idea" leads to "OMN", which then leads to "algorithms", resulting in "composition and analysis" which outputs to "score MusicXML", "MIDI OSC", "graph", and "DAW".
- Listener (Bottom Left):** Shows a Lisp Listener window with the following code:

```
(ch :omn :zh
     :channel 1
     :sound 'gm
     :program 0)
(1h :omn :lh)
```
- Utilities (Right):** A sidebar titled "Ambitus/Generation" containing various functions and their descriptions, such as "ambitus-series", "ambitus-instrument", "ambitus-processing", etc.

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.

The screenshot shows the Opusmodus interface with two main panels:

- Composer Panel (Left):** Displays a hierarchical file tree on the left and the actual script code on the right. The script includes various musical expressions such as chords, intervals, and dynamics, along with playback instructions like `audition-musicxml-last-score` and `#process play loop(39) [Reset] #x30200A3D8FD`.
- Listener Panel (Right):** Shows a musical score preview at the top and a detailed score view below. The score includes multiple staves (Soprano, Alto, Tenor, Bass) with various musical notes, rests, and dynamic markings. The score is labeled "Ohne Titel 44 (PREVIEW)".

Fine and Replace

Find quickly any text or symbol in your score file.

The screenshot shows a text editor window titled "Choralis.opmo". In the search bar at the top, the text "g4b4d5" is entered. Below the search bar, the file content is displayed. The search term "g4b4d5" is highlighted in several places within the code. The code itself is a Common Lisp-like script for generating musical scores, defining chords, chorals, and parts (Soprano, Tenor, Bass). It includes comments in German explaining the search function's behavior.

```

;; Platzieren Sie den Cursor an das Ende jedes Ausdrucks
;; und drücken Sie *E, um Wiedergabe und Notation aufzurufen.
;; Die Wiedergabe kann jederzeit durch Drücken der *ESC-Taste
;; gestoppt werden.
;;
;; Beispiel I
;;
(progn
  (setf chords
    '((w c4e4g4) (h g4b4d5) (h c4e4g4) (-h) (q e4g4b4 q)
      (q a4c5e5 mp leg q leg) (q g4b4d5 leg) (q f4a4c5) (w d4f4a4)
      (w a4c5e5 mf) (-q) (q d4f4a4 q q) (q a4c5e5) (h c4e4g4) (h d4f4a4)
      (q d4fs4a4 leg q leg q) (h g4b4d5 h) (-q) (q bb4d5f5 q q)
      (w eb4g4bb4) (-q) (q c4eb4g4 q) (q g4bb4d5) (h c4eb4g4) (h bb4d5f5)
      (w f4a4c5 h) (-q) (q d4fs4a4) (q g4bb4d5 q q q) (h d4fs4a4 h)
      (-q) (q g4bb4d5) (h c4eb4g4) (h g4bb4d5) (h d4fs4a4 h) (h. g4bb4d5)
      (-q) (w g4bb4d5) (h d4f4a4 h) (-q) (q a4c5e5) (q a4c5e5) (q a4c5e5)
      (h c4eb4g4) (h c4eb4g4) (h g4bb4d5) (q g4bb4d5) (q g4bb4d5)
      (h d4f4a4) (h d4f4a4) (-q) (q d4f4a4 mf) (q a4c5e5 f) (q d4f4a4 p)
      (h. b4d5fs5 p) (q d4f4a4 mp) (q f4a4c5 q q q) (w e4g4b4) (h a4c5e5)
      (-h) (h d4fs4a4) (q d4fs4a4 q) (h g4b4d5) (h g4b4d5) (h e4g4b4)
      (q e4g4b4) (q e4g4b4) (h. c4e4g4) (q c4e4g4) (h f4a4c5)
      (q f4a4c5) (q bb4d5f5) (h. g4b4d5) (q c4e4g4) (h a4c5e5)
      (h d4f4a4) (q g4b4d5) (q d4fs4a4) (w g4b4d5 p) (w g4b4d5 pp)))
  (choralis chords
    :rotation 1
    :index 'v
    :interval 8
    :path '?'
    :seed 826083
    :methods
    '(:soprano ((t5 6) (t-4 (31 32) (3 1)) (t-12 32 2))
      :tenor ((t4 (31 32) (3 1)))
      :bass ((t-12 (1 2 3)) (t-12 4 (1 2)))))
  (ps 'gm
    :satb (list v1 v2 v3 v4)
    :flexible-clef nil
    :tempo 60)
)

```

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 screenshot shows the Opusmodus interface with the Listener panel open. The Listener panel displays a large amount of LISP code for a piece titled "Triangle.opmo". The code includes various OMN forms and LISP functions for generating musical patterns. To the right of the Listener panel is a musical score titled "Ohne Titel 45 (Triangle)" with two staves of musical notation. Below the Listener panel is a graph showing a fluctuating waveform, likely a sound visualization or a plot of a signal. The interface also shows a file browser on the left and a list of MIDI files on the right.

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.

The screenshot shows the Listener panel with a command-line interface. The text area contains a series of numerical values and a question mark, followed by the text "OM: (Lisp Listener)". There is a "Löschen" (Delete) button in the top right corner.

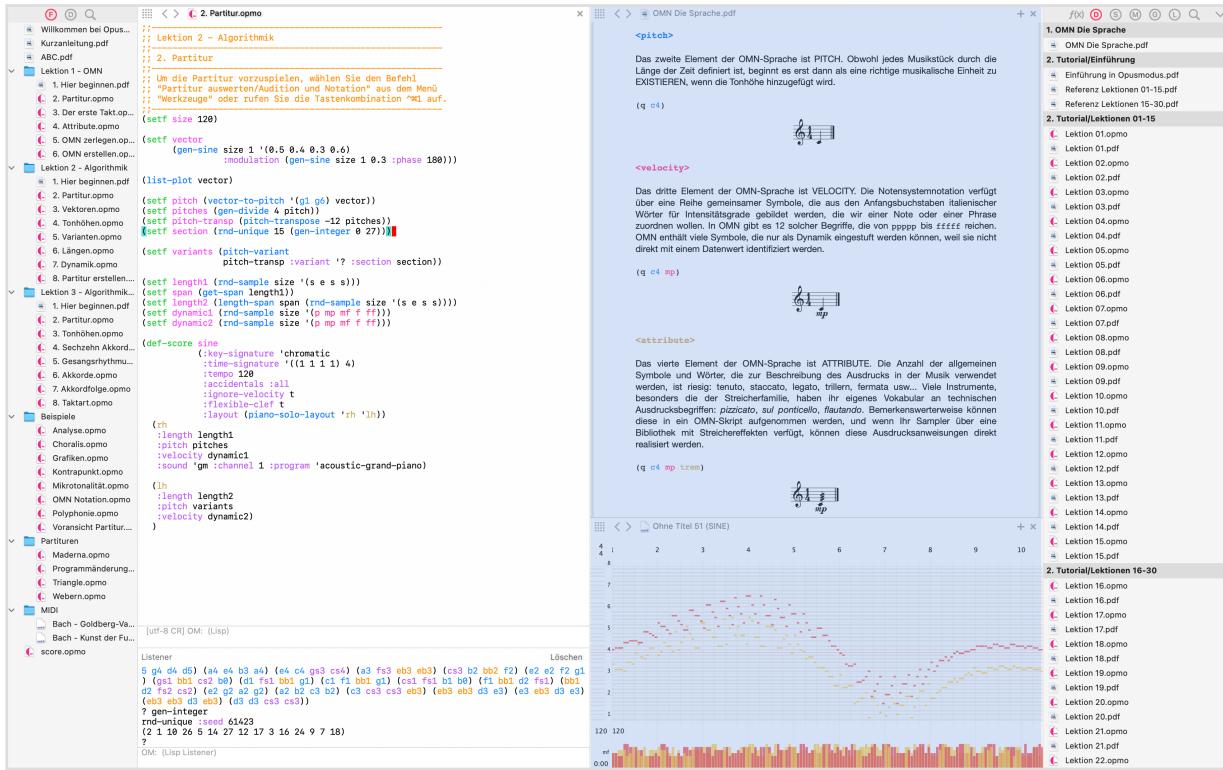
```

Listener
.06116519 -0.12727247 -0.2644759 -0.068613485 -0.14223957 -0.29455185 -0.07616799 -0.15
741992 -0.3250558 -0.08382993 -0.17281608 -0.3559931 -0.09160066 -0.1884305 -0.3873686
-0.09948132 -0.19573422 -0.38081247 -0.09252665 -0.17967544 -0.3485447 -0.084422044 -0.
1633905 -0.31582308 -0.07620349 -0.14687686 -0.2826421 -0.06786966 -0.13013177 -0.24899
636 -0.059419204 -0.11315246 -0.21488047 -0.050850775 -0.09593635 -0.18028899 -0.042162
966 -0.07848048 -0.14521638 -0.03335438 -0.060782146 -0.109656714 -0.0244236 -0.0428384
8 -0.07360478 -0.015369224 -0.024646712 -0.03705435 -0.0061897994 -0.0062038423)
?
OM: (Lisp Listener)

```

Assistant

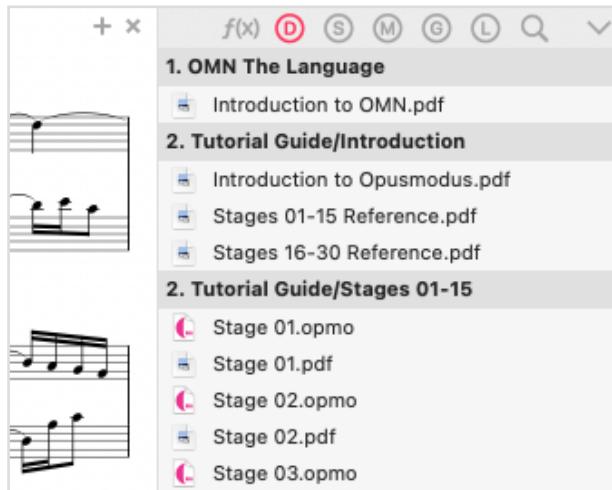
Composing on screen needs more than a single workspace. There are often multiple scripts, texts, audio references, even visual media that play a part in making a new piece of music happen. So Opusmodus calls up the **Assistant**. This is a panel that can be divided and subdivided to contain and display 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 midi file 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.

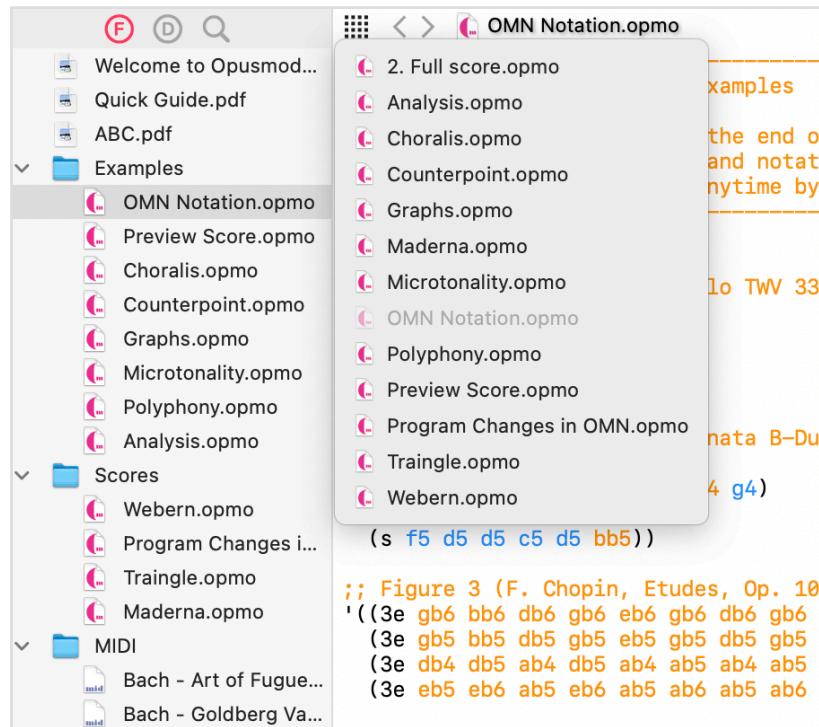
Closing and Duplicating Assistant panel

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.



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.



Notation (MusicXML)

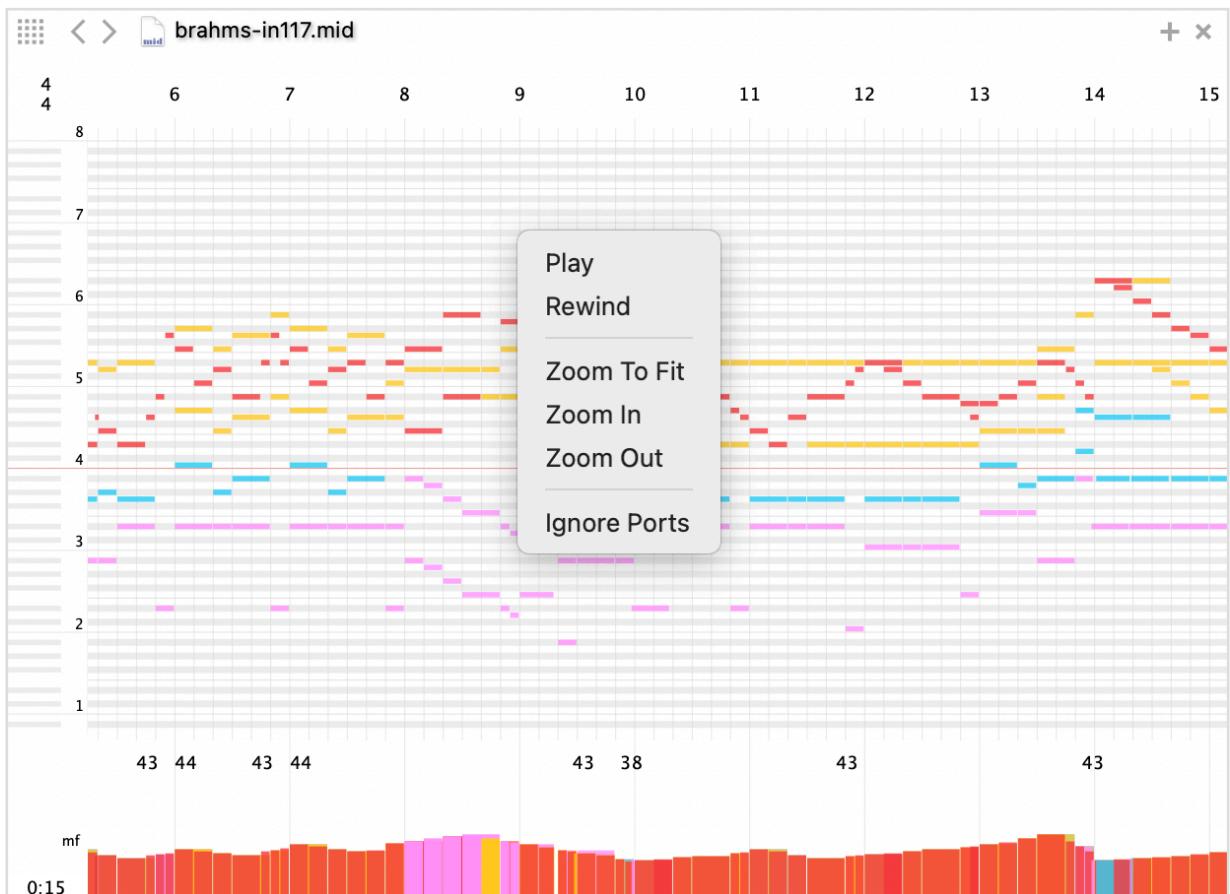
That Opusmodus has adopted **MusicXML** as the de facto standard for displaying notated scores should be no surprise. This is inextricably bound up with the development of the distinctive Opusmodus Notation script **OMN**. It gives the composer the means to design into the very composition of a score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

The image displays three staves of musical notation for string instruments. The first staff (Violin I) starts at measure 26 with a tempo of 60. It includes dynamic markings such as "arco norm.", "ppp", and "rit.". The second staff (Violin II) continues from measure 26. The third staff (Viola/Cello) begins at measure 26. Measures 27 and 28 show various dynamics including "pppp", "sff", "mf", and "ffff". Measures 28 and 29 conclude with dynamics "mf", "ff", and "pp". The notation uses a mix of standard musical symbols and the unique OMN script for specific instructions.

Opusmodus

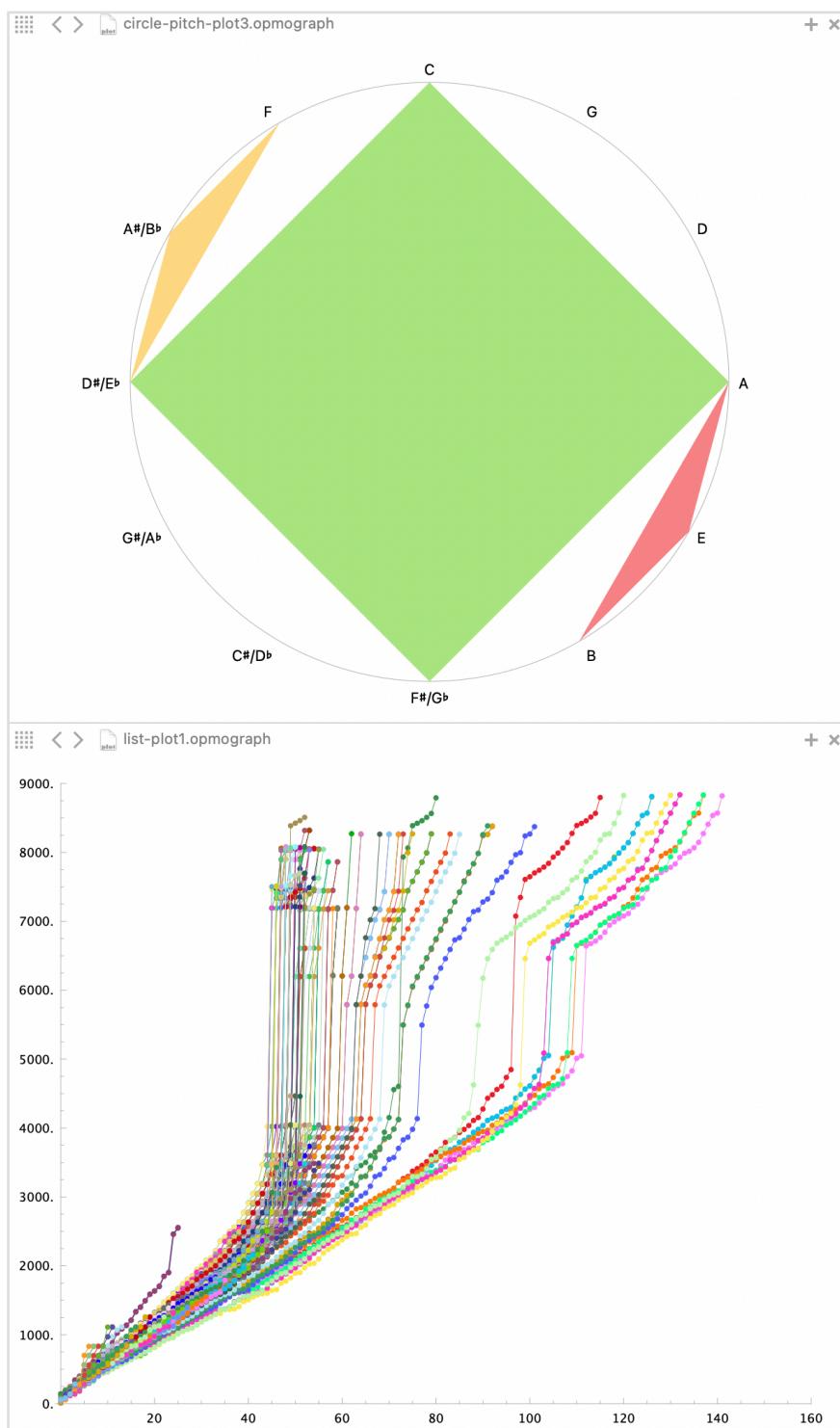
MIDI-Player

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.



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.

The screenshot shows the Opusmodus interface with the Utilities browser open on the left and a musical score on the right.

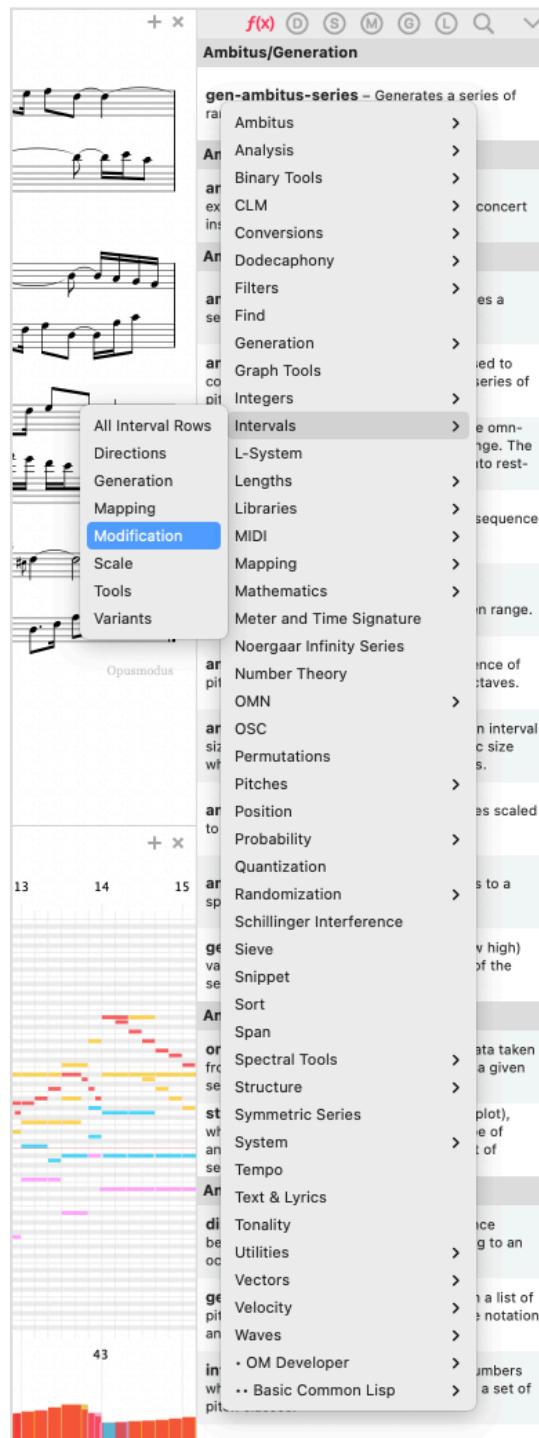
Utilities Browser:

- Mikrotonalität.opmo**: A script for generating microtonal patterns. It includes sections for `vn2` and `vlo`, defining various note heads and their corresponding pitch and duration values.
- Ohne Titel 59 (Luigi Nono, Fragmente-Stille, An Dictima (fragment 26/27))**: A musical score for string quartet (Violin I, Violin II, Viola, Cello) and Bassoon. It shows complex rhythmic patterns and dynamic markings like `ppp` and `fff`.
- Graph Tools** section:
 - `length-list-plot` – Generate a graph from the data provided in sequence of lengths.
 - `length-pitch-list-plot` – Generate a graph from the data provided in two sequences, lengths and pitches.
 - `list-plot` – Generate a graph from the data provided in sequence of real numbers.
 - `omn-list-plot` – Generate a graph from the data provided in sequence of OMN form.
 - `pitch-list-plot` – Generate a graph from the data provided in sequence of pitches.
 - `velocity-list-plot` – Generate a graph from the data provided in sequence of velocities.
- Documents** section:
 - Lektionen 01-15.pdf
 - Lektion 10.opmo
 - Lektion 10.pdf
 - Lektion 11.opmo
 - Lektion 11.pdf
 - Lektion 15.opmo
 - Lektion 24.opmo
 - Grafik.opmo
 - Kontroller 2.opmo
 - Grand Layout.opmo
 - Start & Ende.opmo
 - Sortieren.opmo
 - Fantaisie Pour Boris.opmo
 - Strings Trio.opmo
 - Add Sine.opmo
 - Add Triangle.opmo
 - Fields.opmo
 - Frictions for Two Pianos.opmo
 - Mod Triangle.opmo
 - Triangle.opmo
 - Vector to Pitch.opmo
- Graphs** section:
 - length-pitch-list-plot.opmograph
 - length-pitch-list-plot2.opmograph
 - list-plot1.opmograph
 - list-plot2.opmograph
 - list-plot3.opmograph
 - omn-list-plot.opmograph
 - pitch-list-plot1.opmograph
 - pitch-list-plot2.opmograph
 - velocity-list-plot.opmograph

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 Library

The first icon on the left brings up all the “help” guidance about the **System Library** 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.

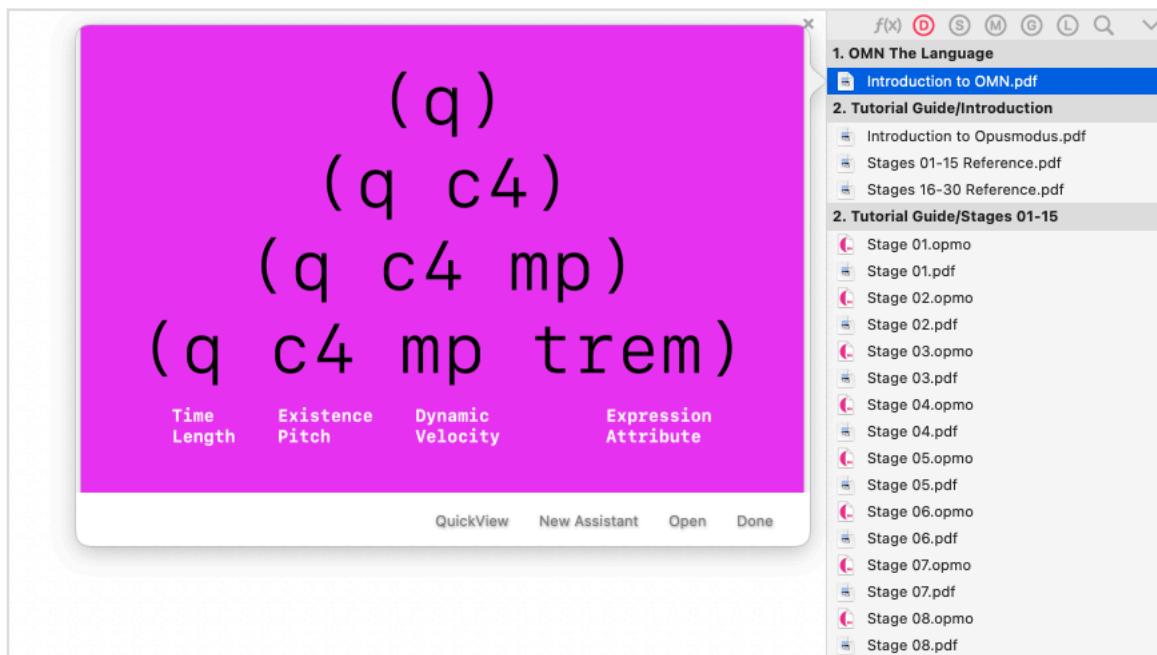
The screenshot shows the Opusmodus interface with a project titled "brahms-in117.mid". A popover window is open, displaying information about the "Tonality" section of the Utilities panel. The popover has a title bar with "Description:" and a close button. The main content area contains text and code snippets related to harmonic paths and tonality mapping. On the right side of the popover, there is a vertical list of utility functions with their descriptions:

- Tonalities** – Access a collection of defined tonalities.
- create-tonality** – Define a new tonality by name.
- expand-tonality** – Retrieve pitch, interval or integer values for a given tonality/scale/chord in any variant.
- get-harmonic-path** – Returns a pitch sequence made up of a sequence or any number of voices.
- harmonic-path** – Map a list of pitches to a harmonic path. (This function is highlighted with a blue background).
- harmonic-progression** – Returns a chord progressions from a given scale.
- harmonics** – Returns a number of harmonics for a given pitch.
- tonality-map** – Map a list of pitches to a tonality.
- tonality-series** – Returns a series of tonalities.
- Twelve-Tone/Analysis**
- get-form-set** – Returns twelve-tone row of a given form.

At the bottom of the popover, there are buttons for "QuickView", "New Assistant", "Open", and "Done".

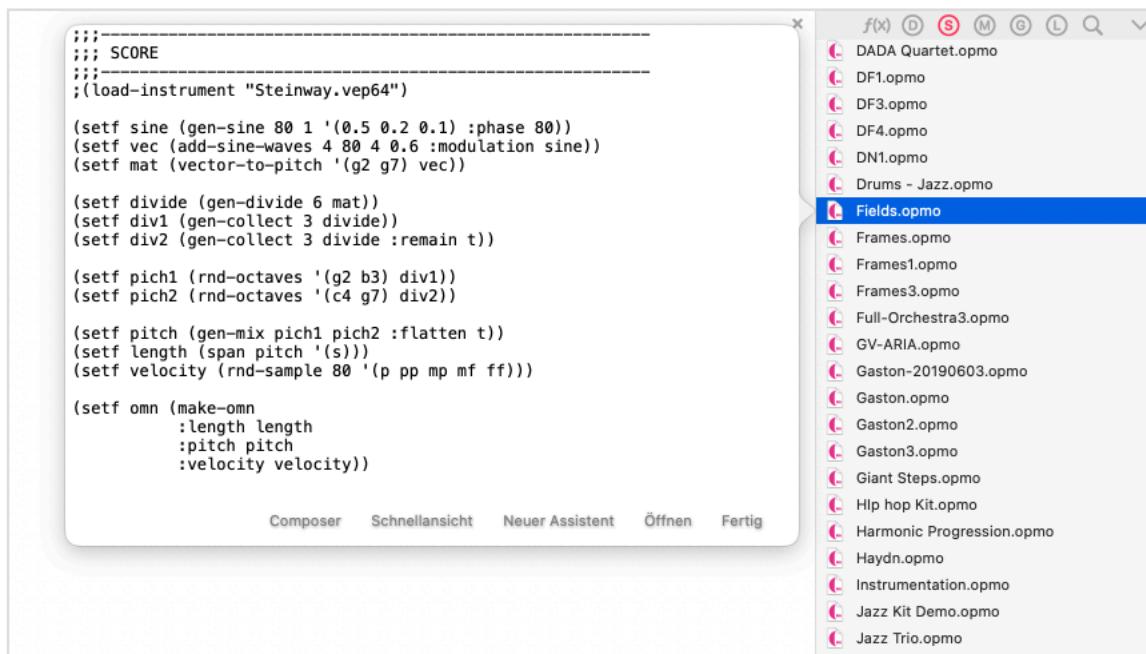
Documentation

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 **How-to** 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.



Score

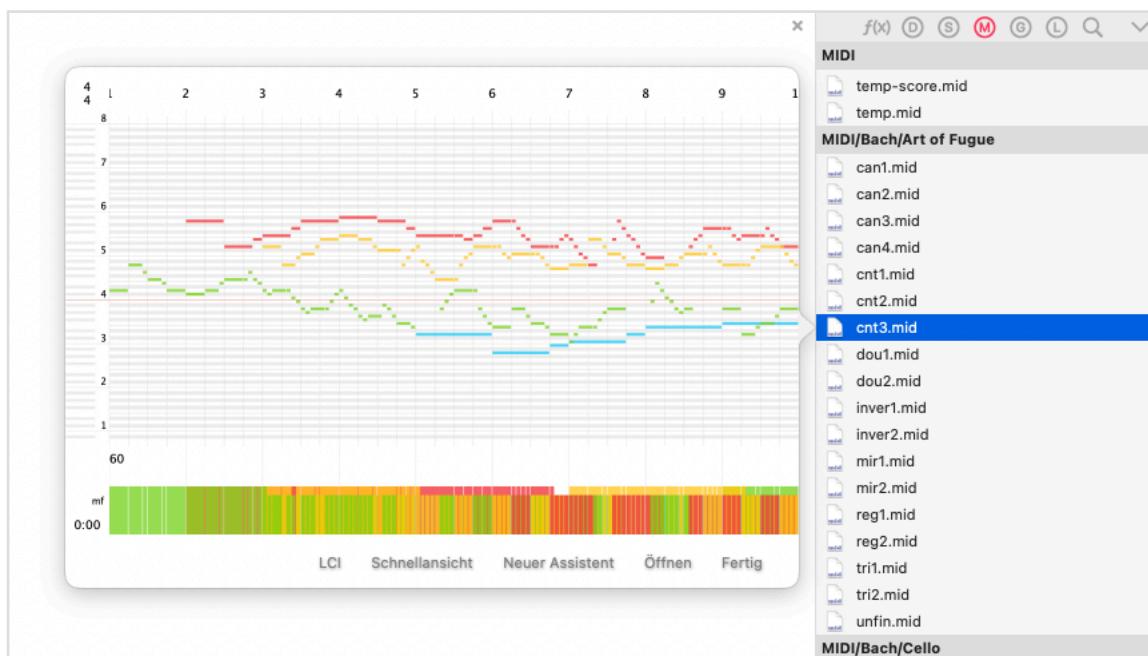
Welcome to your personal Opusmodus score library. Here, anything that belongs to a particular project in scripted code can be saved and archived. Many composers find that producing short sections of score is often the most efficient way of working, so it's really important to have such a utility. Files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



Notice that the popover feature makes it possible to view the score “on the fly”.

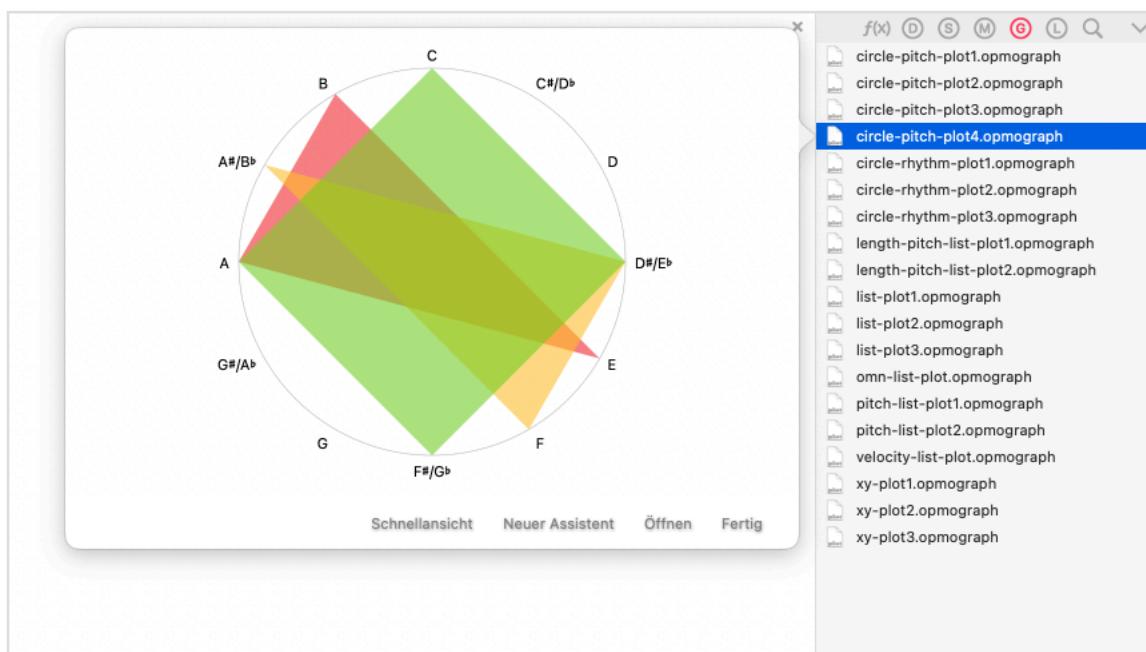
Media (MIDI Files)

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.



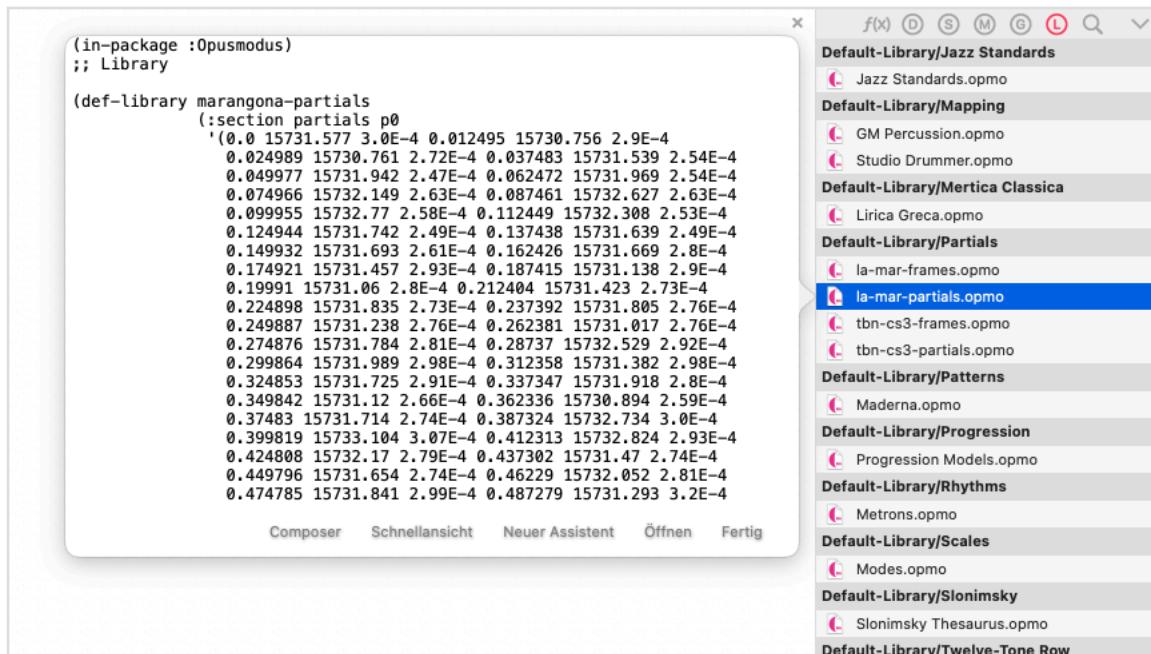
Graphs (Plot)

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

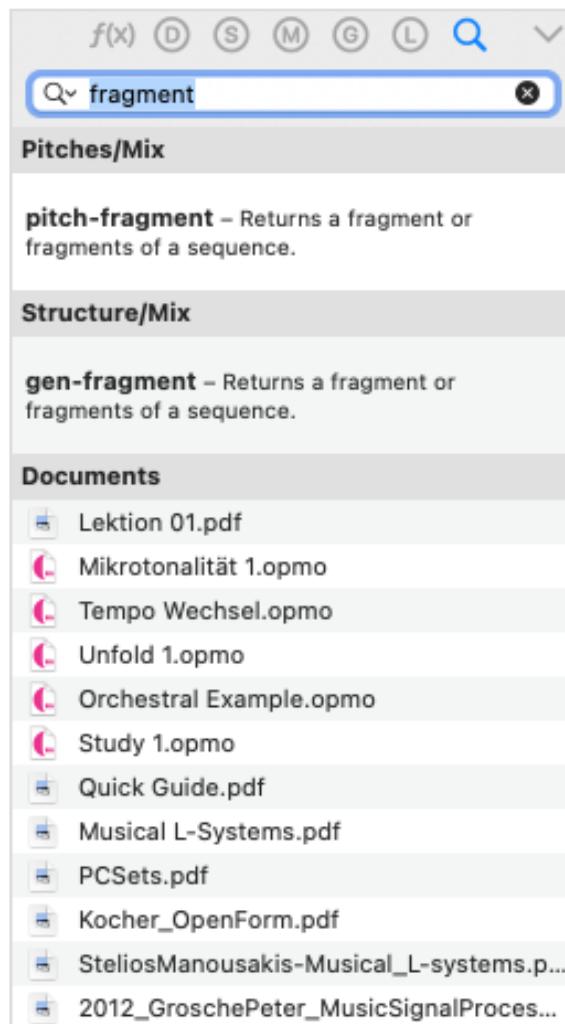
Composers working in serious sound design and digital media production for film, TV and web applications often have an arsenal of sound sources on-board and out-board. Opusmodus has a great range of default sound sets – and it's easy to set up your own. Just imagine mapping the output of natural algorithms 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 rhythms. This data can be brought into scripts in random or template-controlled forms.



Search

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. 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.

```
pitch-fragment count size pitch &key transpose flatten
    section exclude seed span flat
```

[Function]

Arguments and Values:

<i>count</i>	an integer or list of integers (count of fragments).
<i>size</i>	an integer list (size of fragments).
<i>pitch</i>	a list or list of pitches.
<i>transpose</i>	a list of integers (transposition values).
<i>flatten</i>	T or NIL . The default is NIL . (internal process).
<i>section</i>	an integer or list of integers. Selected list or lists to process. The default is NIL .
<i>exclude</i>	an integer or list of integers. Excluded list or lists from process. The default is NIL .
<i>seed</i>	NIL or an integer. The default is NIL .
OMN:	
<i>flat</i>	NIL or T . If true , the OMN single type lists are flatten. The default is NIL .
<i>span</i>	:length , :pitch or :velocity . The default is :pitch .

Description:

This function returns a fragment of a sequence from an existing list or lists.

```
(pitch-fragment 4 3 '(c4 d4 e4 f4 g4 a4) :seed 235)
=> (d4 e4 f4 e4 f4 g4 c4 d4 e4 d4 e4 f4)
```

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.

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 **Snippet → Audition and Notation** (⌘1) you will be able to see and hear the output of the function:



The screenshot shows a software interface with a code editor at the top containing Lisp code. Below it is a "Listener" window with a list of evaluated expressions and their results. A red box highlights the first line of the code editor.

```
(setf soprano (integer-to-pitch (rnd-number 18 7 14 :seed 20)))
(setf alto (integer-to-pitch (rnd-number 18 0 5 :seed 19)))
(setf tenor (integer-to-pitch (rnd-number 18 -7 -3 :seed 21)))
(setf bass (integer-to-pitch (rnd-number 18 -9 -16 :seed 23)))
```

* [utf-8 CR] OM: (Lisp)

Listener	Löschen
? rnd-number :seed 23 integer-to-pitch (c3 cs3 cs3 gs2 cs3 c3 eb3 eb3 b2 c3 a2 c3 d3 d3 eb3 cs3 a2 gs2) ? rnd-number :seed 20 integer-to-pitch (gs4 bb4 a4 a4 d5 a4 bb4 g4 g4 b4 bb4 cs5 bb4 gs4 gs4 g4 a4 cs5) ?	
OM: (Lisp Listener)	

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.

OMN Notation.opmo

```
;; Figur 8 (J. S. Bach, Goldberg-Variationen,  
Variatio 18 a 1 Clav., 1743)  
((<= h q g tie) (h, g8) (h, f5) (e g f5 h e5 tie)  
(e d e d5 c5 q d8 tie) (q d5 -h))  
;  
;; Figur 9 (M. Reger, Aus meinem Tagebuch, I, 3, 1984/12)  
((q d4#d4b0d5 f< leg c4#e4a4c5 > leg bb3d4g4b4 < leg  
e 4#e4a4c5 < leg d4#f4b4d5 ffs > leg  
f4#d4b4d5 f< leg e4#g4b4b4 > leg ff 4#f4a4d5 mf)  
(q d3s3d3d3 p leg bb3e4g4 < leg b3d4g4 < leg g3c3s4 leg  
d3s3d3 pp))
```

;; Figur 10 (K. Stockhausen, Mantra für 2 Pianisten, 1979)
((e a3 g stacc a3 stacc a3 tie
e a3 q,, b3 t f h, g4 mp (acc e e4 p f4 e4 d4) w e4))

;; Figur 11 (B. Smotra, Aus meinem Leben,
Streichquartett e-Moll, 1876)
((e bo-1 (w bo sf leg)
(q e4 marc stacc esp - -> f6 stacc)
(w q sf leg) (b3 stacc mrc - -> d6 stacc)
(h e4 sf marc leg q g3 stacc mrc - -> a3 stacc)
(h e4 sf marc leg q g3 stacc mrc - -> a3 stacc)
(Leg q g3 marc e e3 -> f53 ten)))

;; Figur 12 (Opusmodus, Algorithmic, 2014)
((s cs8 p g6 mp mf -) (s cs8 f e g3 ff s p)
(s cs8 ff e g3 ff s p)
(s g3 mp e e5 sf mrf) (s g6 ff e e5 ff s p)
(s g3 mp ff e cs8 f) (s g6 ff e cs8 mp ff s mf)
(e g3 f e5 ff g6 p) (e g6 mp -s cs8 mf)
(e g3 f e5 ff g6 p) (s cs8 p e g6 mp s mf)))

;; Figur 13 (L. v. Beethoven, op. 59, No.1, Finale)
(((leg q d3 e e3 s d3 e3))
(((leg q d3 e e3 s d3 e3)))
(((leg q a3 e e4 mp s bbs >))
(((leg q a3 e e3 g3 s a3)))
(((taac e F3 d3 c3 a2)))
(((taac e d3 f3 e3 a3)))

;; Figur 14 (S. Prokofiev, Peter and the Wolf, op.67)
((#2bb2bd2 s marc
((#2bb2bd2 s marc ff#3a3 db#3b2 eb#3a3 e d3g#b3s)
db#3b2 g3b3d4 ten tb3c3c4 ten g3b3d4 ten g3b3s ten
(acc e c3) H f3c4 marc))

Listener
p
nil
? audition-musicxml-omn-snippet
audition-musicxml-omn-snippet
audition-musicxml-omn-snippet
audition-musicxml-omn-snippet

OMN (Lisp Listener)

Öffnen Löschen

OMN Notation.opmo

;; Figur 8 (J. S. Bach, Goldberg-Variationen,
Variatio 18 a 1 Clav., 1743)
((<= h q g tie) (h, g8) (h, f5) (e g f5 h e5 tie)
(e d e d5 c5 q d8 tie) (q d5 -h))

;; Figur 9 (M. Reger, Aus meinem Tagebuch, I, 3, 1984/12)
((q d4#d4b0d5 f< leg c4#e4a4c5 > leg bb3d4g4b4 < leg
e 4#e4a4c5 < leg d4#f4b4d5 ffs > leg
f4#d4b4d5 f< leg e4#g4b4b4 > leg ff 4#f4a4d5 mf)
(q d3s3d3d3 p leg bb3e4g4 < leg b3d4g4 < leg g3c3s4 leg
d3s3d3 pp))

;; Figur 10 (K. Stockhausen, Mantra für 2 Pianisten, 1979)
((e a3 g stacc a3 stacc a3 tie
e a3 q,, b3 t f h, g4 mp (acc e e4 p f4 e4 d4) w e4))

;; Figur 11 (B. Smotra, Aus meinem Leben,
Streichquartett e-Moll, 1876)
((e bo-1 (w bo sf leg)
(q e4 marc stacc esp - -> f6 stacc)
(w q sf leg) (b3 stacc mrc - -> d6 stacc)
(h e4 sf marc leg q g3 stacc mrc - -> a3 stacc)
(h e4 sf marc leg q g3 stacc mrc - -> a3 stacc)
(Leg q g3 marc e e3 -> f53 ten)))

;; Figur 12 (Opusmodus, Algorithmic, 2014)
((s cs8 p g6 mp mf -) (s cs8 f e g3 ff s p)
(s cs8 ff e g3 ff s p)
(s g3 mp e e5 sf mrf) (s g6 ff e e5 ff s p)
(s g3 mp ff e cs8 f) (s g6 ff e cs8 mp ff s mf)
(e g3 f e5 ff g6 p) (e g6 mp -s cs8 mf)
(e g3 f e5 ff g6 p) (s cs8 p e g6 mp s mf)))

;; Figur 13 (L. v. Beethoven, op. 59, No.1, Finale)
(((leg q d3 e e3 s d3 e3))
(((leg q d3 e e3 s d3 e3)))
(((leg q a3 e e4 mp s bbs >))
(((leg q a3 e e3 g3 s a3)))
(((taac e F3 d3 c3 a2)))
(((taac e d3 f3 e3 a3)))

;; Figur 14 (S. Prokofiev, Peter and the Wolf, op.67)
((#2bb2bd2 s marc
((#2bb2bd2 s marc ff#3a3 db#3b2 eb#3a3 e d3g#b3s)
db#3b2 g3b3d4 ten tb3c3c4 ten g3b3d4 ten g3b3s ten
(acc e c3) H f3c4 marc))

Listener
p
nil
? audition-musicxml-omn-snippet
audition-musicxml-omn-snippet
audition-musicxml-omn-snippet
audition-musicxml-omn-snippet

OMN (Lisp Listener)

Öffnen Löschen

OMN Die Sprache.pdf

OMN Die Sprache.pdf

2. Tutorial/Einführung

Einführung in Opusmodus.pdf

Referenz Lektionen 01-15.pdf

Referenz Lektionen 15-30.pdf

2. Tutorial/Lektionen 01-15

Lektion 01.opromo

Lektion 01.pdf

Lektion 02.opromo

Lektion 02.pdf

Lektion 03.opromo

Lektion 03.pdf

Lektion 04.opromo

Lektion 04.pdf

Lektion 05.opromo

Lektion 05.pdf

Lektion 06.opromo

Lektion 06.pdf

Lektion 07.opromo

Lektion 07.pdf

Lektion 08.opromo

Lektion 08.pdf

Lektion 09.opromo

Lektion 09.pdf

Lektion 10.opromo

Lektion 10.pdf

Lektion 11.opromo

Lektion 11.pdf

Lektion 12.opromo

Lektion 12.pdf

Lektion 13.opromo

Lektion 13.pdf

Lektion 14.opromo

Lektion 14.pdf

Lektion 15.opromo

Lektion 15.pdf

2. Tutorial/Lektionen 16-30

Lektion 16.opromo

Lektion 16.pdf

Lektion 17.opromo

Lektion 17.pdf

Lektion 18.opromo

Lektion 18.pdf

Lektion 19.opromo

Lektion 19.pdf

Lektion 20.opromo

Lektion 20.pdf

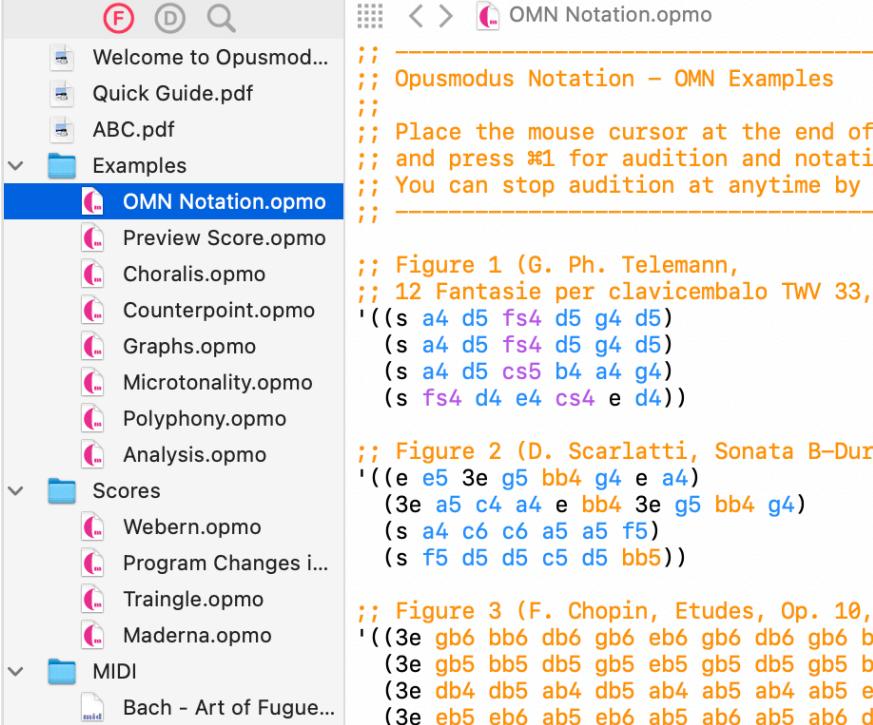
Lektion 21.opromo

Lektion 21.pdf

Lektion 22.opromo

Finder

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



The screenshot shows the Opusmodus Navigator interface. On the left is a file tree:

- Welcome to Opusmod...
- Quick Guide.pdf
- ABC.pdf
- Examples
 - OMN Notation.opmo** (selected)
 - Preview Score.opmo
 - Choralis.opmo
 - Counterpoint.opmo
 - Graphs.opmo
 - Microtonality.opmo
 - Polyphony.opmo
 - Analysis.opmo
- Scores
 - Webern.opmo
 - Program Changes i...
 - Traingle.opmo
 - Maderna.opmo
- MIDI
 - Bach - Art of Fugue...

The right pane displays the content of the selected file, **OMN Notation.opmo**:

```

;; -----
;; Opusmodus Notation - OMN Examples
;;
;; Place the mouse cursor at the end of
;; and press #1 for audition and notation
;; You can stop audition at anytime by
;;
;; Figure 1 (G. Ph. Telemann,
;; 12 Fantasie per clavicembalo TWV 33,
'((s a4 d5 fs4 d5 g4 d5)
 (s a4 d5 fs4 d5 g4 d5)
 (s a4 d5 cs5 b4 a4 g4)
 (s fs4 d4 e4 cs4 e d4))

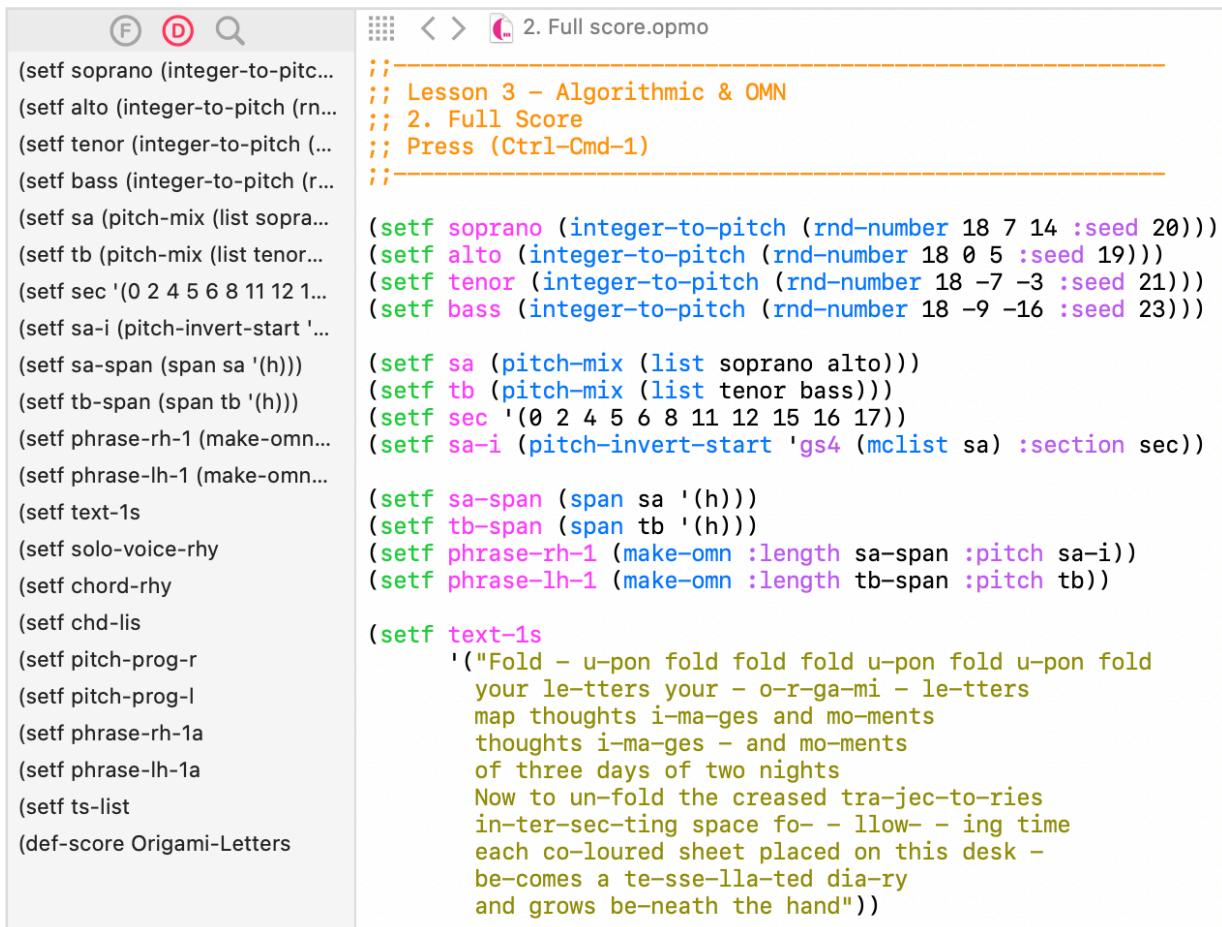
;; Figure 2 (D. Scarlatti, Sonata B-Dur
'((e e5 3e g5 bb4 g4 e a4)
 (3e a5 c4 a4 e bb4 3e g5 bb4 g4)
 (s a4 c6 c6 a5 a5 f5)
 (s f5 d5 d5 c5 d5 bb5))

;; Figure 3 (F. Chopin, Etudes, Op. 10,
'((3e gb6 bb6 db6 gb6 eb6 gb6 db6 gb6 b
 (3e gb5 bb5 db5 gb5 eb5 gb5 db5 gb5 b
 (3e db4 db5 ab4 db5 ab4 ab5 ab4 ab5 e
 (3e eb5 eb6 ab5 eb6 ab5 ab6 ab5 ab6 d

```

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!



```

(setf soprano (integer-to-pitch...))
(setf alto (integer-to-pitch (rn...
(setf tenor (integer-to-pitch (...))
(setf bass (integer-to-pitch (r...
(setf sa (pitch-mix (list sopra...
(setf tb (pitch-mix (list tenor...
(setf sec '(0 2 4 5 6 8 11 12 1...
(setf sa-i (pitch-invert-start '...
(setf sa-span (span sa '(h)))
(setf tb-span (span tb '(h)))
(setf phrase-rh-1 (make-omn...
(setf phrase-lh-1 (make-omn...
(setf text-1s
(setf solo-voice-rhy
(setf chord-rhy
(setf chd-lis
(setf pitch-prog-r
(setf pitch-prog-l
(setf phrase-rh-1a
(setf phrase-lh-1a
(setf ts-list
(def-score Origami-Letters

;;-----
;; Lesson 3 - Algorithmic & OMN
;; 2. Full Score
;; Press (Ctrl-Cmd-1)
;;-----

(setf soprano (integer-to-pitch (rnd-number 18 7 14 :seed 20)))
(setf alto (integer-to-pitch (rnd-number 18 0 5 :seed 19)))
(setf tenor (integer-to-pitch (rnd-number 18 -7 -3 :seed 21)))
(setf bass (integer-to-pitch (rnd-number 18 -9 -16 :seed 23)))

(setf sa (pitch-mix (list soprano alto)))
(setf tb (pitch-mix (list tenor bass)))
(setf sec '(0 2 4 5 6 8 11 12 15 16 17))
(setf sa-i (pitch-invert-start 'gs4 (mcclist sa) :section sec))

(setf sa-span (span sa '(h)))
(setf tb-span (span tb '(h)))
(setf phrase-rh-1 (make-omn :length sa-span :pitch sa-i))
(setf phrase-lh-1 (make-omn :length tb-span :pitch tb))

(setf text-1s
'("Fold - u-pon fold fold u-pon fold u-pon fold
your le-tters your - o-r-ga-mi - le-tters
map thoughts i-ma-ges and mo-ments
thoughts i-ma-ges - and mo-ments
of three days of two nights
Now to un-fold the creased tra-jec-to-ries
in-ter-sec-ting space fo- - llow- - ing time
each co-coloured sheet placed on this desk -
be-comes a te-sse-lla-ted dia-ry
and grows be-neath the hand"))

```

Find

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 Library**. In the illustration below there is a search for a word tenon-thy. The pointer selects one instance and immediately the file in which it belongs appears in the **Composer**.

```

;; Lesson 3 – Algorithmic & OMN
;; 6. Chords
;;
;; With the rhythmic sketch of the vocal part in place
;; the chordal rhythm can be added. Now we can see / hear
;; the rhythmic outline of voice and accompaniment.
;; Go to TOOLS Menu and choose Evaluate Score.

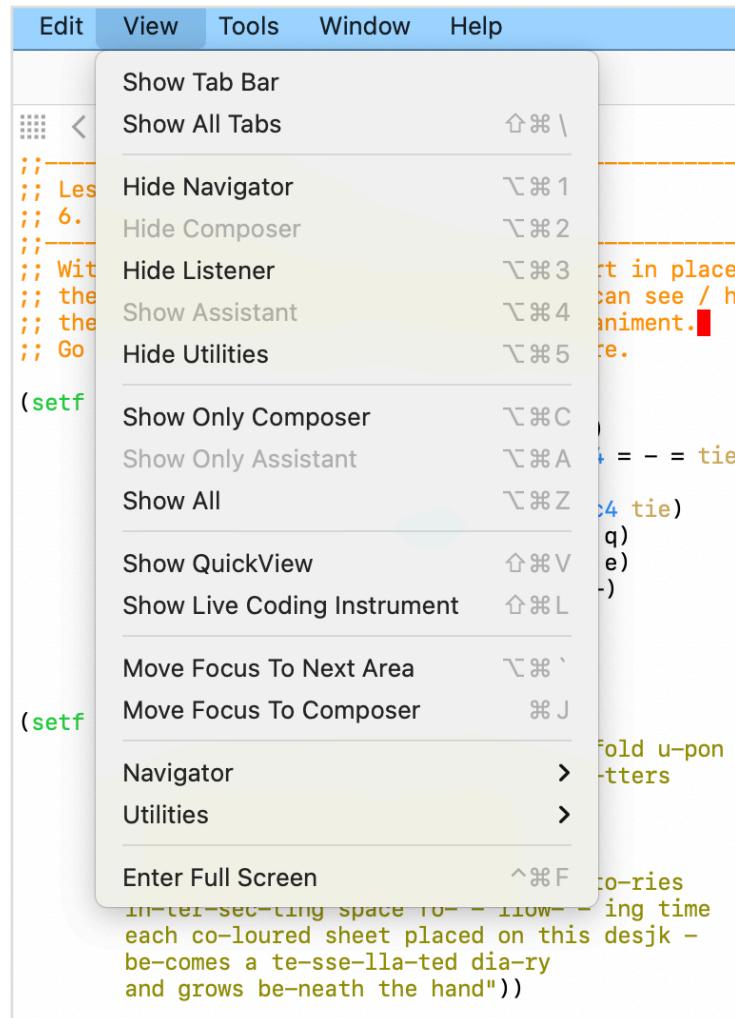
(setf tenor-rhy
  '((-q h c4 tie) (e c4 = = q) (-q c4 -)
    (e c4 = = = =) (q c4 - e =) (e c4 = - = tie)
    (e c4 = = = tie) (q c4 e = - =)
    (q c4 s = = e =) (-e q. c4 s = e c4 tie)
    (e c4 = = = - =) (h c4 q) (-e c4 q. q)
    (-q -h) (-e c4 = =) (e c4 = = s = e)
    (s c4 = = = q) (-e q c4 = e) (q c4 -)
    (-e c4 = = q) (e c4 = = q tie)
    (s c4 = = = = = =) (e c4 = - =)
    (q c4 = = =) (q c4 - -)))]

(setf text-1s
  '("Fold – u-pon fold fold fold u-pon fold u-pon fold
  your le-tters your – o-r-ga-mi – le-tters
  map thoughts i-ma-ges and mo-ments
  thoughts – i-ma-ges – and mo-ments
  of three days of two nights
  Now to un-fold the creased tra-jec-to-ries
  in-ter-sec-ting space fo- – llow- – ing time
  each co-loured sheet placed on this desjk –
  be-comes a te-sse-lla-ted dia-ry
  and grows be-neath the hand"))

```

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:

The screenshot shows the Opusmodus interface with the following components visible:

- Left Panel (Composer):** Displays a file tree and the source code for '2. Partitur.opmo'. The code includes various musical definitions and patterns, such as 'Lektion 1 - OMN' and 'Lektion 2 - Algorithmik'.
- Right Panel (Score):** Shows two staves of musical notation for 'Goldberg Variationen'. The top staff is labeled 'al tempo di Giga' and the bottom staff is labeled 'Opusmodus'.

Here, the **Composer** and **Listener** panels are shown only:

The screenshot shows the Opusmodus interface with the following components visible:

- Left Panel (Composer):** Displays the source code for 'Maderna.opmo', which includes definitions for 'dictum-b', 'dictum-c', 'globals', and 'assemble-voices'.
- Right Panel (Listener):** Shows a piano-roll style visualization of musical events over time, with tracks for piano solo, bassoon, and maracas.

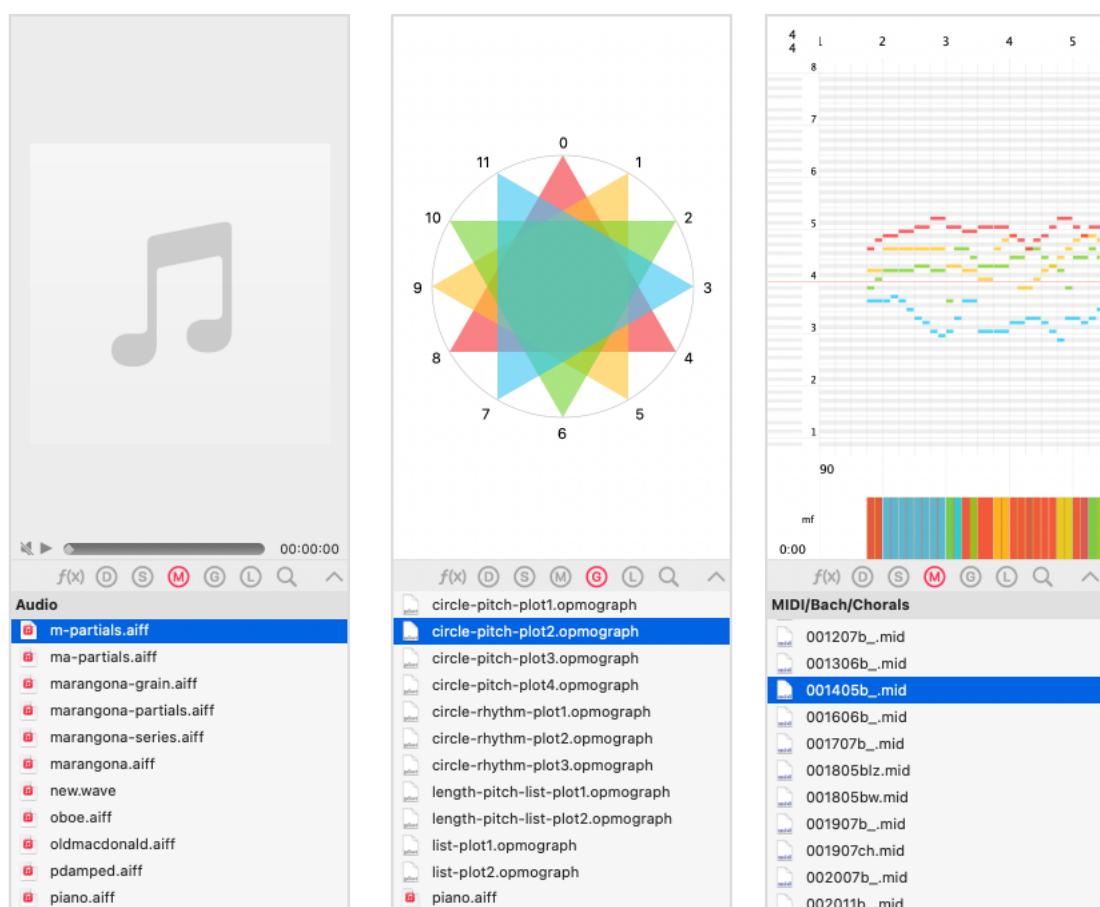
Here the **Assistant** panel is in the “presentation” mode display:

J.S. Bach
The Art of the Fugue
BWV 1080
Contrapunctus I

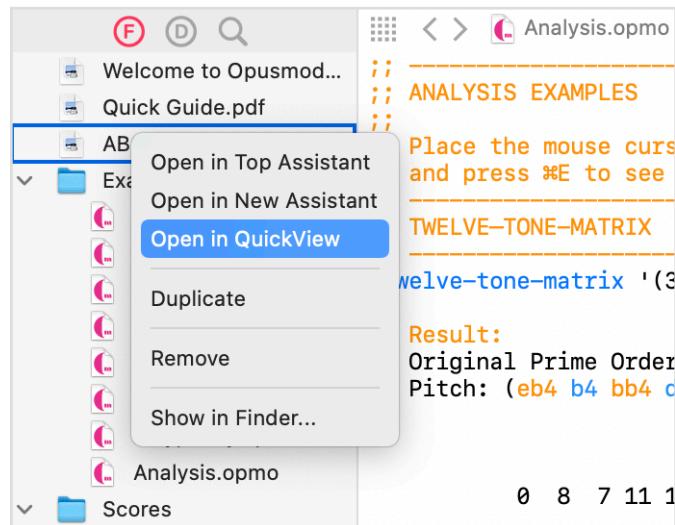
The image shows two staves of musical notation. The top staff begins with a treble clef, a key signature of one sharp (F#), and a 13/8 time signature. The bottom staff begins with a bass clef, a key signature of one sharp (F#), and a 13/8 time signature. Both staves feature complex rhythmic patterns and harmonic changes.

QuickView

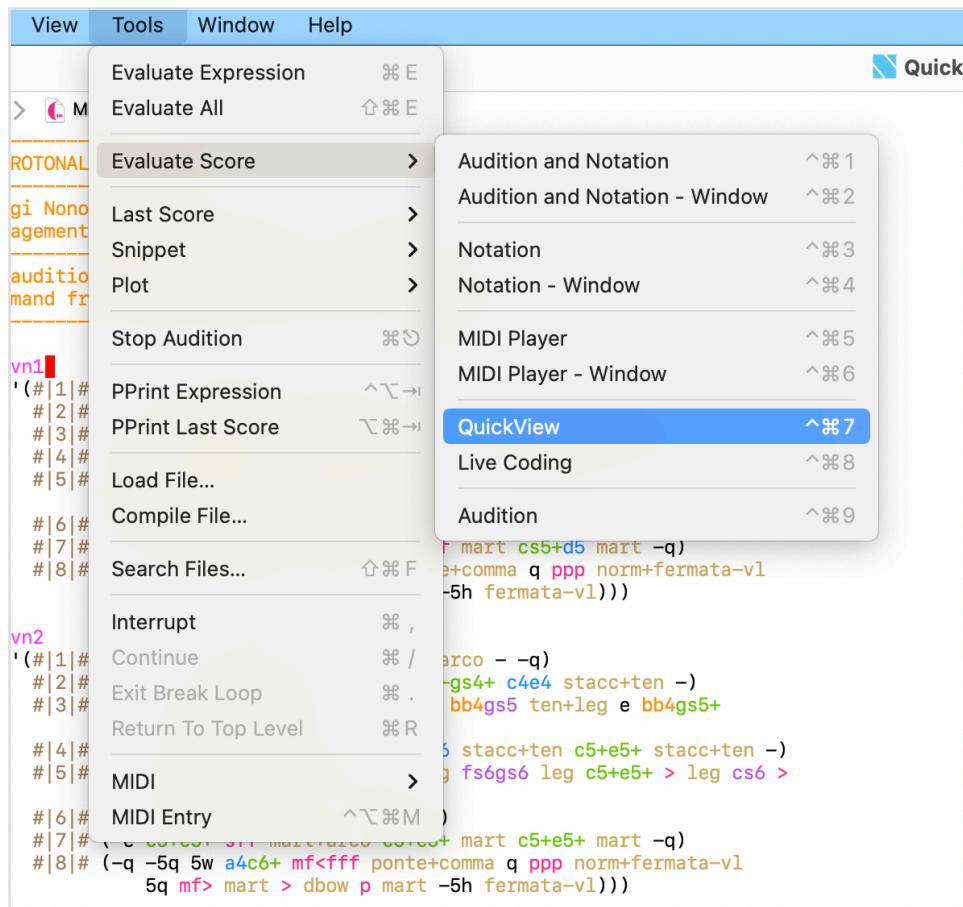
The **QuickView** area is a flexible multi-application space occupying the top half of the Utilities panel. It can show the **Midi Player**, the **Live Coding Instrument** (LCI), PDF, text or image files. Opening the **Midi Player** in the **Assistant** can break up the pattern of workflow in a session, 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!



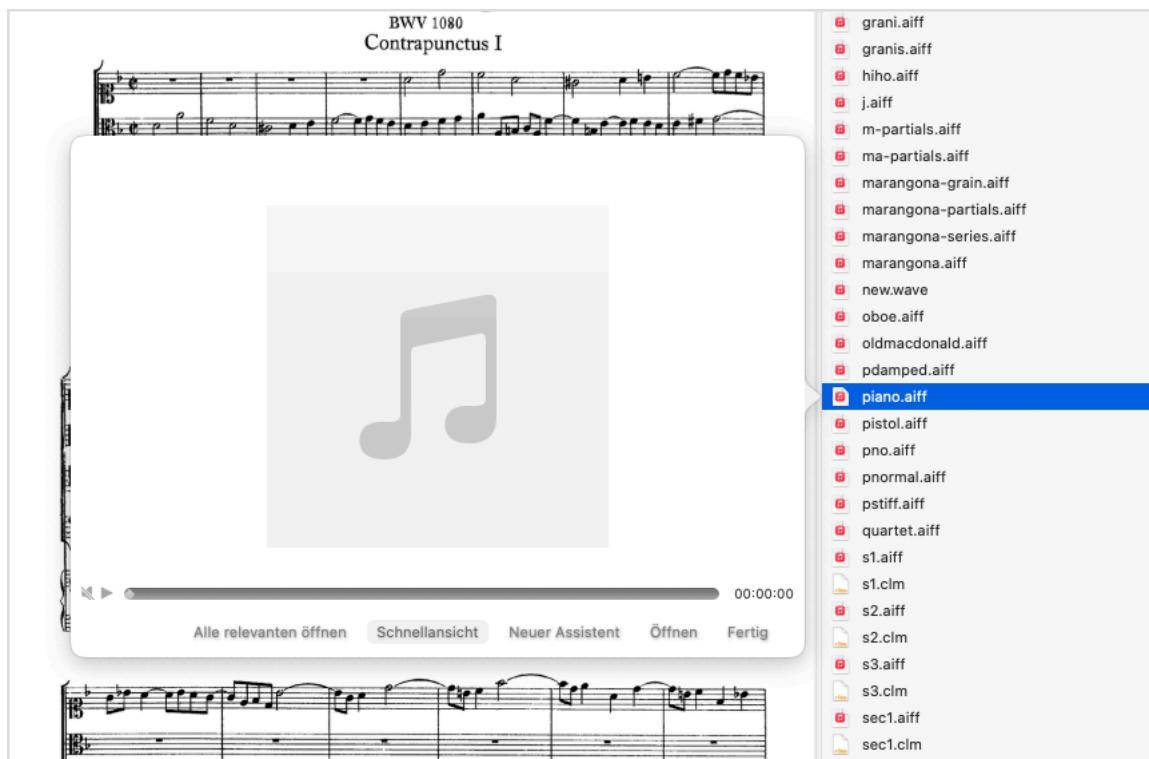
Via a contextual menu accessed by right-clicking on the **Navigator** panel you can open any file with the command **Open in QuickView**.



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

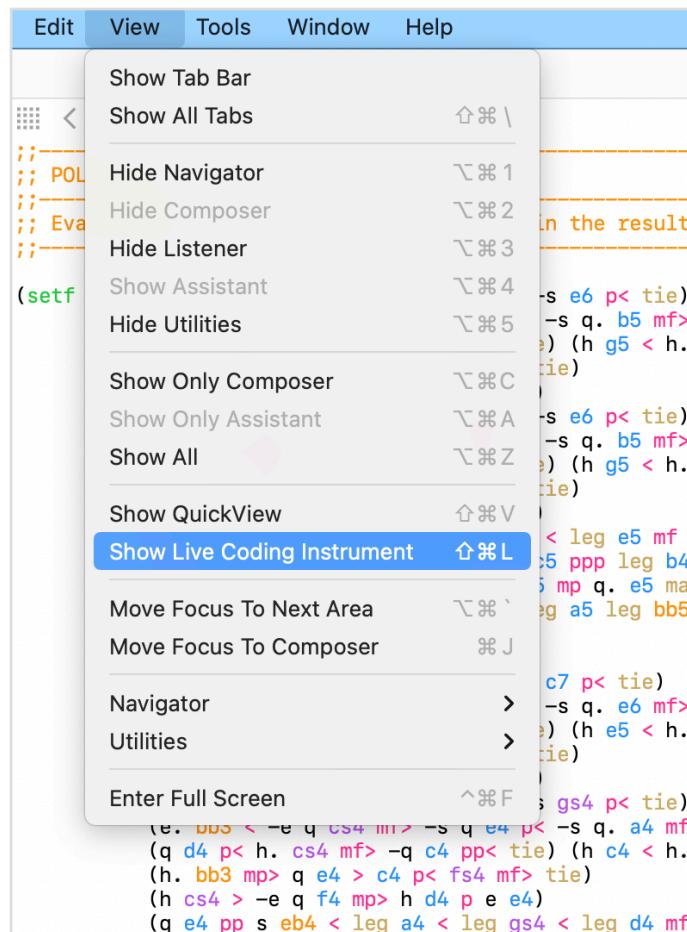


The popover window can open a file in the **QuickView** panel via a click on the **QuickView** button.



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.

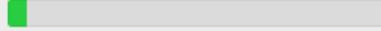
The **LCI** gives the composer an intuitive control panel and the possibility of working in true live coding style directly with the script. In practice composers who use the **Live Coding Instrument** often begin with a script, make a change, then “playing” that change from the buttons of the **LCI**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

Bruno Maderna, Serenata Per un

Bars Beats	4 3 117
Min.Sec	0:17.439
Beat	14
Tempo	42

Loop

Start	1 0 000
Mute	Random
Original Tempo	Custom Tempo
Set Tempo	
Bar	Beat
Set Position	1



f(x)       

Ambitus/Generate

gen-ambitus-series – Generates a series of range lists derived from two vectors.

Ambitus/Instrument List

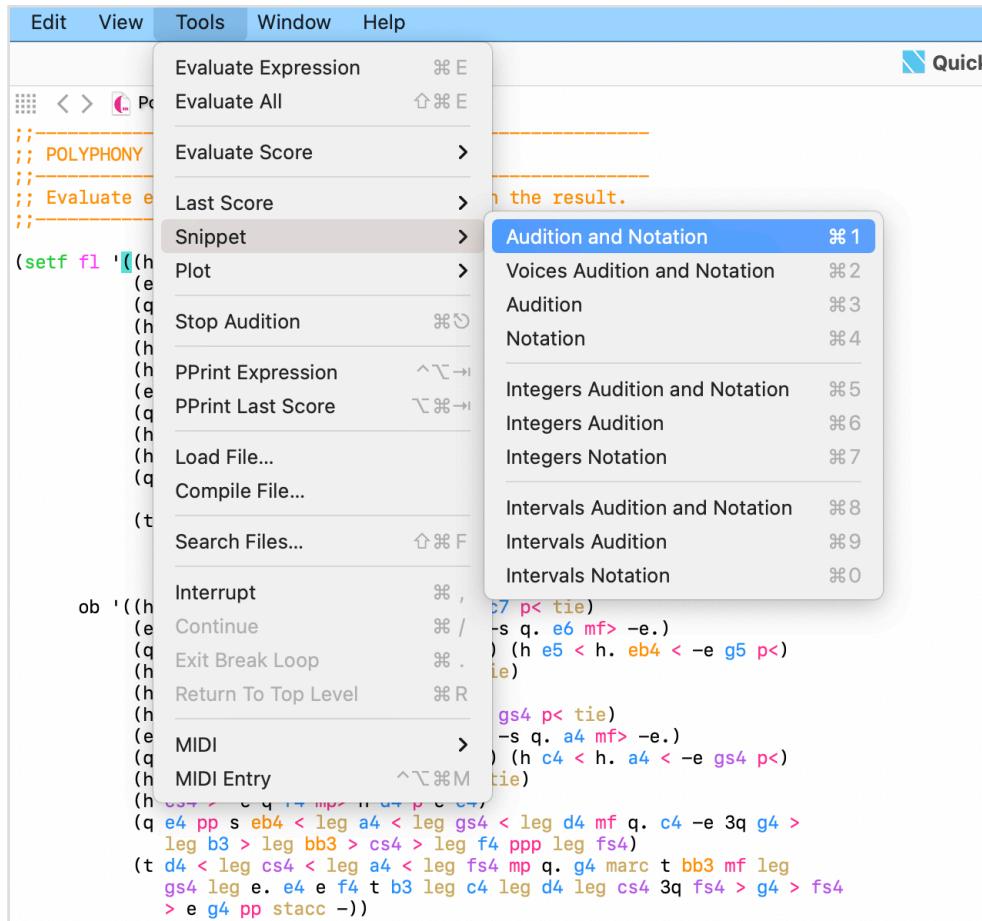
ambitus-instrument – Pre-set range expression for every standard acoustic / concert instrument in three different formats.

Ambitus/Process

ambitus-chord – The function processes a sequence according to a size setting.

Snippet

There are four types of snippets. One is the Audition snippet which will play any selected expression, list or fragment of your material, the second is the Voices 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 **Snippet → Audition and Notation** (⌘1).



The same process is used to show a notation snippet. The snippet is designed to deal with three different inputs, OMN, integers and intervals.

```
'((-e -s (leg eb5 pp< ponte+con-sord 3e a6 < eb5 < a6 < leg))
  ((leg 3e eb5 mp> a6 > eb5 >) -e -e)
  (-e -3e - (leg+stacc e4 ppp< < < p)))
```

Opusmodus

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 "Titel"
   :composer "Komponist"
   :copyright "copyright"
   :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-SCORE** 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

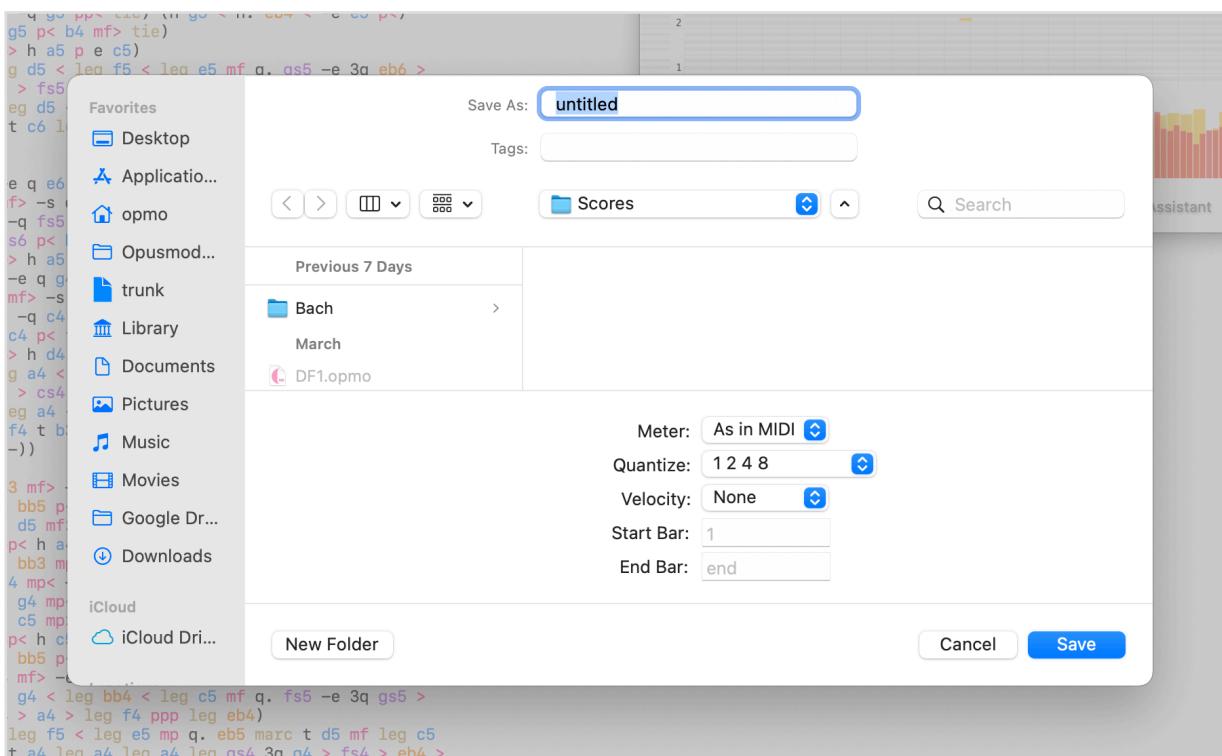
Once a score definition has been created, it can be played via the **Tools → Evaluate Score → MIDI Player** menu item. You can also preview your composition notated in standard notation within Opusmodus via the **Tools → Evaluate Score → Audition and Notation** menu item. This will open up a new **Assistant** 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

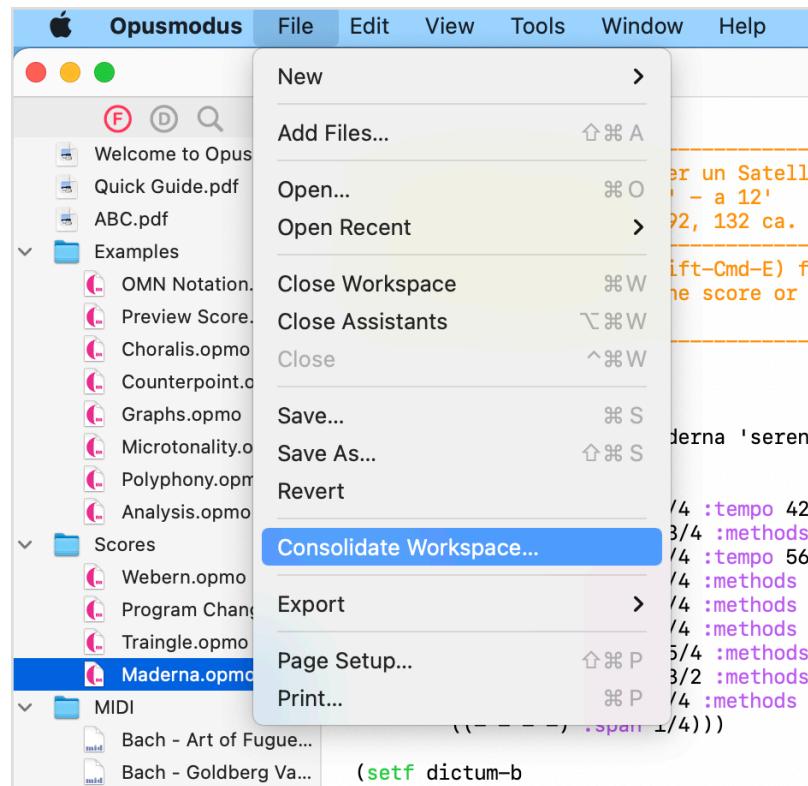
A new score doesn't have to begin from working with Opusmodus. A composer's improvisation can be recorded and captured to midi file; in making music for film, TV or web production composers often source a style or atmosphere from one of the many midi file libraries on-line. Opusmodus provides a unique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

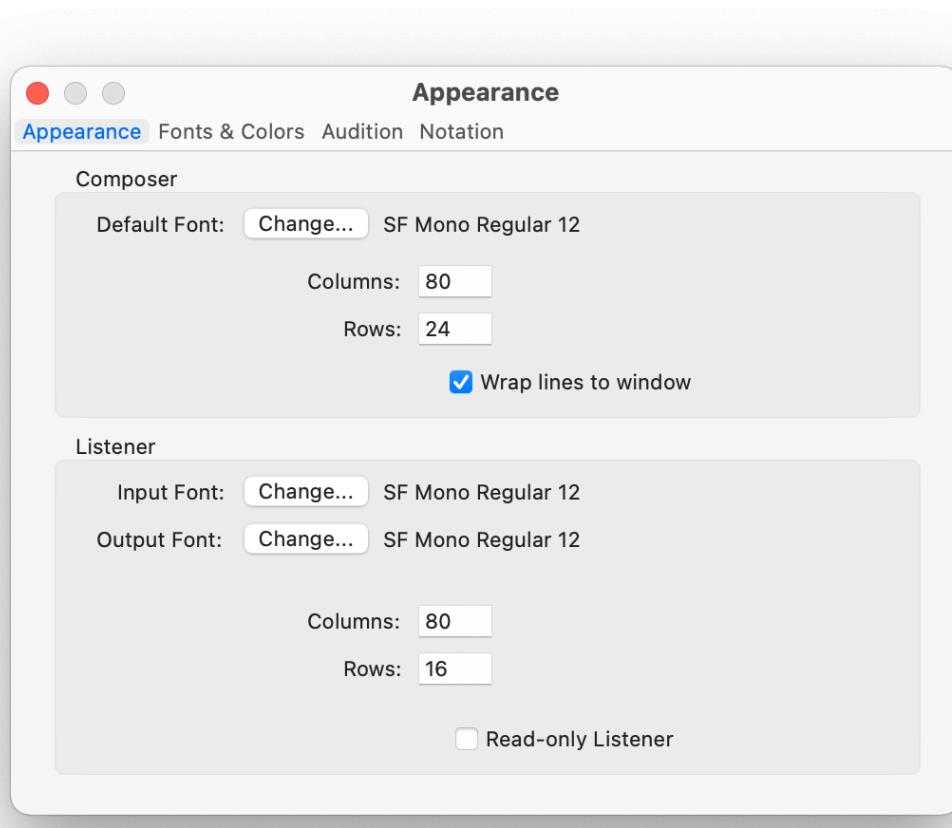


Preferences

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.

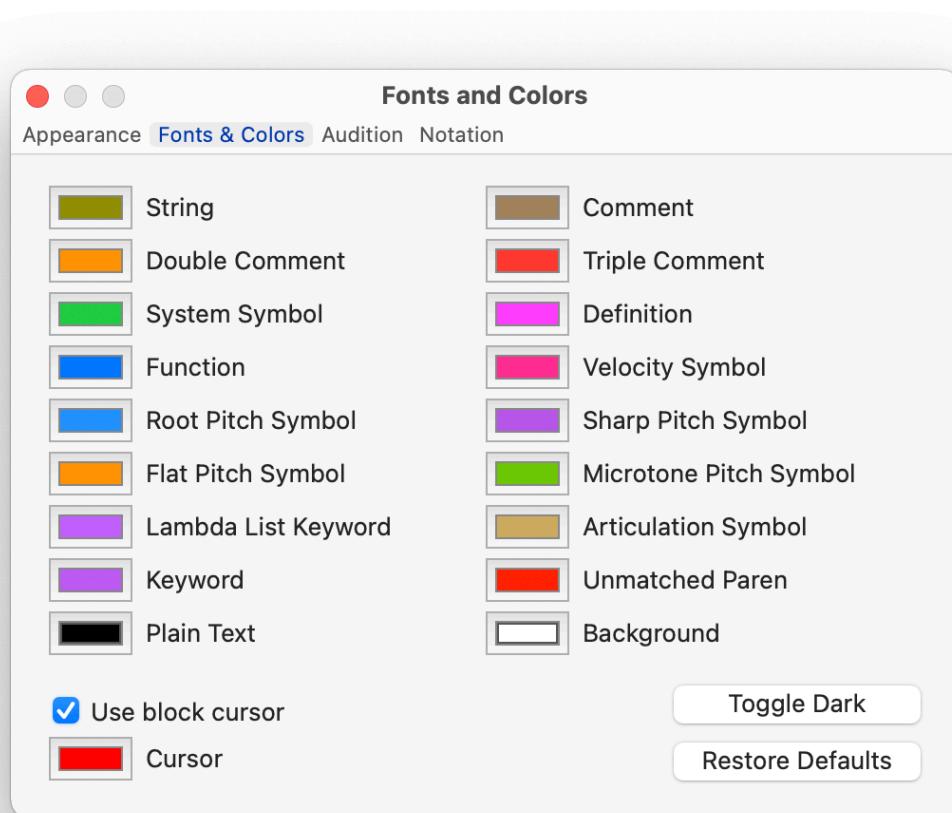
Appearance

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



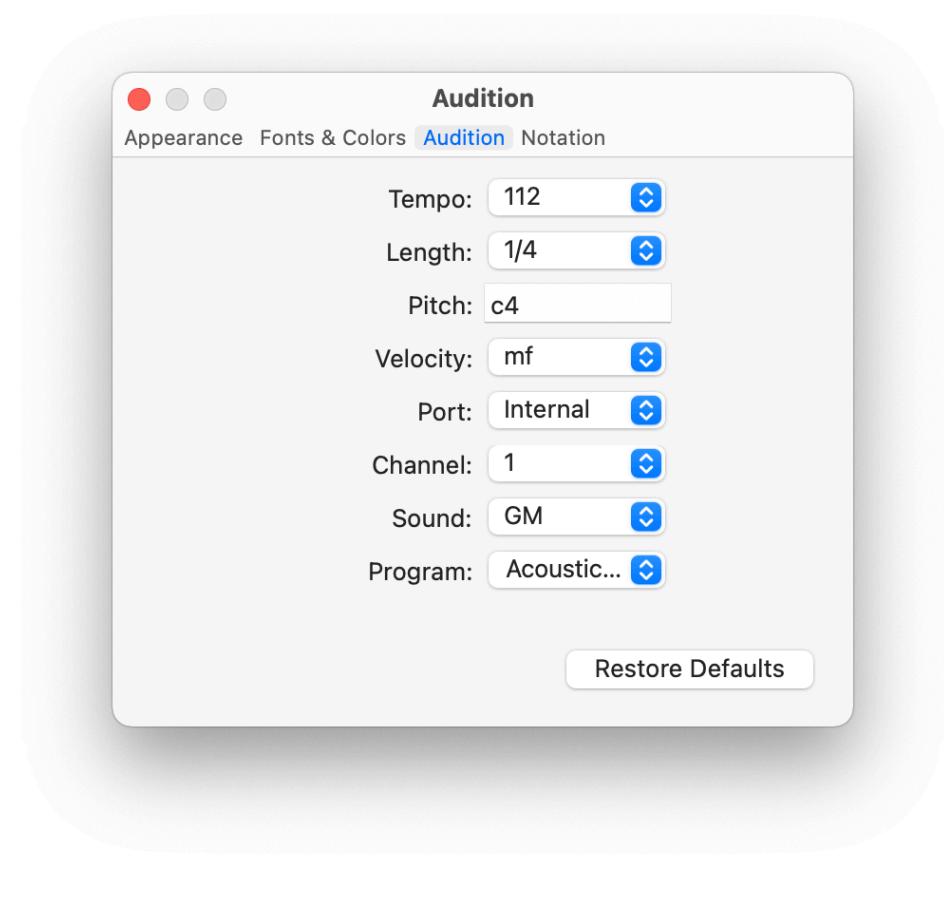
Fonts & Colours

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.



Notation

Notation scale and Bar Number Display.

