Technical Documentation: Philippe Manoury, Das Wohlpräparierte Klavier

This document describes the technical requirements for presenting a live performance of Philippe Manoury's Das Wohlpräparierte Klavier, also known as the Third Piano Sonata, as of March 2026. You may obtain this document, the score, and supporting software from Editions Durand, Paris, France. The software, including this documentation, is also available as open source free software, hosted on msp.ucsd.edu.

Das Wohlpräparierte Klavier was first presented in 2025. The piece itself dates from 2021 and was premiered by Daniel Barenboim using a realization by Gilbert Nouno, based in turn on a realization by Thomas Goepfer of an installation, also by Manoury, named Scala. Both Scala and DWPK were realized in Max. Nouno's patch ran in Max 8.0, Intel architecture only, and relied on specific versions of Java and several external libraries. The realization described here retains the electronic "score", in Antescofo, written by the composer, with minor updates. The Pure Data patches described here imitate the behavior of Nouno's Max patch in order to preserve the Antescofo score in its original form.

Copyright information

This realization (including this documentation) is provided as open source software under BSD license. For convenience, the downloadable software distribution includes two other packages under separate ownership: Antescofo by Arshia Cont and Jean-Louis Giavitto, and vstplugin~ by Christof Ressi; these two are packaged with their separate copyright notices.

Dependencies

The realization runs in Pure Data (Pd) version 0.55-2 or later, and uses Antescofo (2025 version) or, when available, its replacement TsimTsoum. Both Pd and TsimTsoum are open source and should be readily adaptable to future architectures and OSes. In addition there are the following dependencies:

- Synful. This is a synthesis package by Eric Lindemann, used to make string sounds, primarily pizzicati. This is freely available and is planned to be open source once the compilation procedure can be made more straightforward. This currently exists as a VST3 plug-in, but might in the future be available as a Pd external object instead.
- vstplugin~, by Christof Ressi, a VST adapter for Pd, used to load the current (VST3) implementation of Synful into Pd. If the provided copy does not work (either because of a non-Macintosh OS or because of anti-malware protection) vstplugin~ can be downloaded from within Pd using the "find externals" item in the help menu. If it is downloaded, it is wise

to move the downloaded object into the "extern" directory, replacing the existing one, so that the entire piece is as self-contained as possible.

- smerdyakov, a Markov chain simulator, included with C source as part of this realization. A MacOS (Apple silicon) object file is included, although you may need to circumvent Apple's anti-malware protection to use it. Alternatively it should be straightforward to recompile using the included makefile. Its only dependency, besides standard "C" libraries, is Pd itself.

Concert setup

The diagram below shows the audio setup used. (As shown the mixer and audio interface are assumed to be separate but alternatively the mixer can be interfaced directly with the computer. Also, the computer interface is assumed here to be USB, but other interface technologies may be used instead.)

It may or may not be desirable to amplify the piano, depending on room size. The loudness of the electronic sound should be comparable to that of the piano, and the combination should be at a level appropriate for chamber music. There are nonetheless some low-frequency electronic sounds that will benefit if there are sub-woofers - but there are no dedicated subwoofer outputs provided by the patch, so the audio system should then derive its sub-woofer sends from the patch's six main outputs.

Software installation and layout

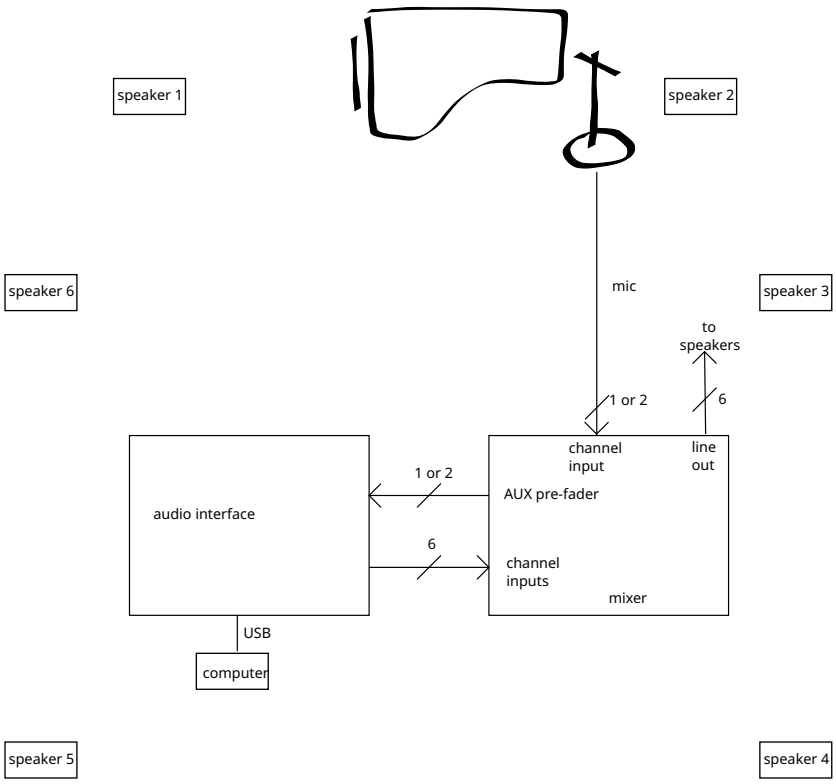
There are only three file packages that should be referenced to run this piece: the Pd installation (which should not contain any extensions beyond the base distribution), the files included in this realization (possibly recompiling or otherwise updating the "externs" antescofo, vstplugin~, and/or smerdyakov), and the Synful plug-in.

The Synful plug-in currently runs in MacOS or Windows, not linux, and should be installed in one or another standard location for VSTs. In addition, the 8 or 9 "presets" for synful, which are distributed in the subdirectory "patch/lib/synful-presets/", should be installed in the Synful preset directory. On MacOS this is "~/Library/Audio/Presets/Synful/SynfulOrchestra".

Everything else needed should be arranged as in this distribution. The file layout is as shown (directories in parentheses aren't needed in performance but are included for reference):

```
samples                 soundfiles used in sampler and for tests
extern                  all Pd "externs"
extern/antescofo~           antescofo extern - later replace with TsimTsoum
extern/smerdyakov           random walk generator, including source
extern/vstplugin~           VST3 host, used to load Synful into Pd
patch                   Pd patches, antescofo script, etc., mostly text format
patch/lib                   sub-patches used by main Pd patch
```

Technical diagram for Philippe Manoury's Das Wohlpräparierte Klavier , for piano and electronics



The mixer and computer should be placed on a table at front of house..

Figure 1: tech diagram for performance

```
patch/lib/walks                 random walks used by smerdyakov
patch/lib/zrev~                 reverberator used for spatialization
patch/lib/synful-presets        (synful presets to be copied to system)
patch/lib/spatrev~              (alternative reverberator)
patch/prep                  (scripts and notes used to prepare patch)
patch/doc                   (this documentation)
patch/.git                  (git repository showing history of patch)
```

Startup

The patch runs at 48 kHz sample rate and should be run without any external "path" settings so that it only loads files included with the distribution. (Pd is often set up to search a "user extensions" directory; this should be turned off using its preferences dialog.) Pd's audio delay should be set to the smallest value that doesn't result in buffer underflows when the piece is run, typically 10 to 15 msec. The "buffer size" may usually be left at the default 64 samples. The "use callbacks" toggle is usually turned off.

A file, patch/override/locale.txt, may be present, and if so, it can be used to choose audio channels for input and output. By default the patch only takes audio input from a single channel, but using the "locale.txt" file one can arrange for it to sum two audio input channels in case this simplifies the audio setup. If there is no locale.txt file in the patch/lib/override directory, the patch/lib/locale.txt file is loaded instead and you will get the default configuration (input channel 1 and output channels 1-6).

At startup the patch loads a number of audio samples and the Synful plug-in, a process that should take a few seconds. The Antescofo script is loaded anew whenever the piece is started or restarted. Both when loading the patch and when starting it it is wise to check the Pd printout window for errors.

The patch presents a front panel in two parts. The left-hand part (shown below) contains controls normally used to run the patch in a rehearsal or performance. The right-hand side contains the implementation and control panels for the roughly 200 parameters that are sent from the antescofo script to the patch. These can be modified by hand to study or demonstrate the patch but need not be watched or monitored during performance.

Checking audio and setting levels

The "piano-monitor" control, if nonzero, causes the piano input signal to be monitored (as a mono signal) through outputs 1 and 2. A value of 50 corresponds to unit gain (100 is +24 db). The "test tone" controls can be used to generate either a sinusoid or filtered noise at the input of the patch.

Three gain controls are provided, for the input level, the output level, and the reverb level. Each of these controls correspond to unit gain at midpoint (50) and +24 dB at maximum (100). The reverb level is intended to be set appropriately

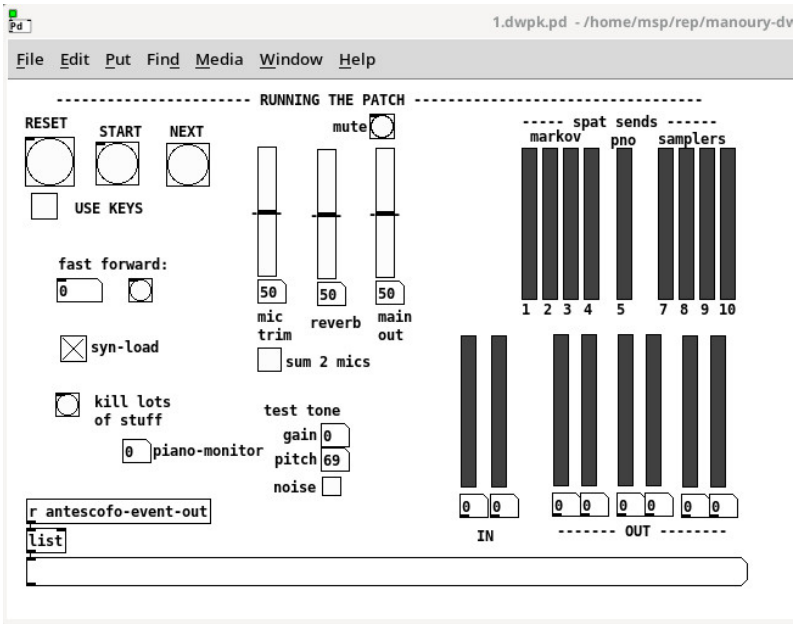


Figure 2: patch screen shot

for the hall acoustics and left alone, whereas the input and output levels may have to be ridden live. (The input level can be raised or lowered to adjust the balance between whatever processing is done to the live piano input and the rest of the mix. This becomes important at one point in the piece where a frequency shifter should be loud enough to cover the live sound of the piano).

The signal paths in the patch are summarized below:

Running the piece

If all goes according to plan, only the "start" and "next" controls and the input and output levels need to be managed in performance. (Moreover, the input and output levels may be controlled from the audio mixer in which case the patch operator need only use the "start" and "next controls).

If the "use keys" toggle (beneath the "reset", etc., buttons) is turned on, the right-arrow key can be used in place of the "next" button. This is a more ergonomic choice, but remember to turn this off if you want to edit the patch.

To perform the piece, each of about 100 cues have to be hit using the "next" button or right arrow in sync with the pianist. In some spots the cue should be hit simultaneously at the moment the pianist plays a note or chord. The cues are all notated in the score and are mostly numbered (1-94) but in some spots there are fractional cues (61.0, 61.1, ...). The current cue number is shown,

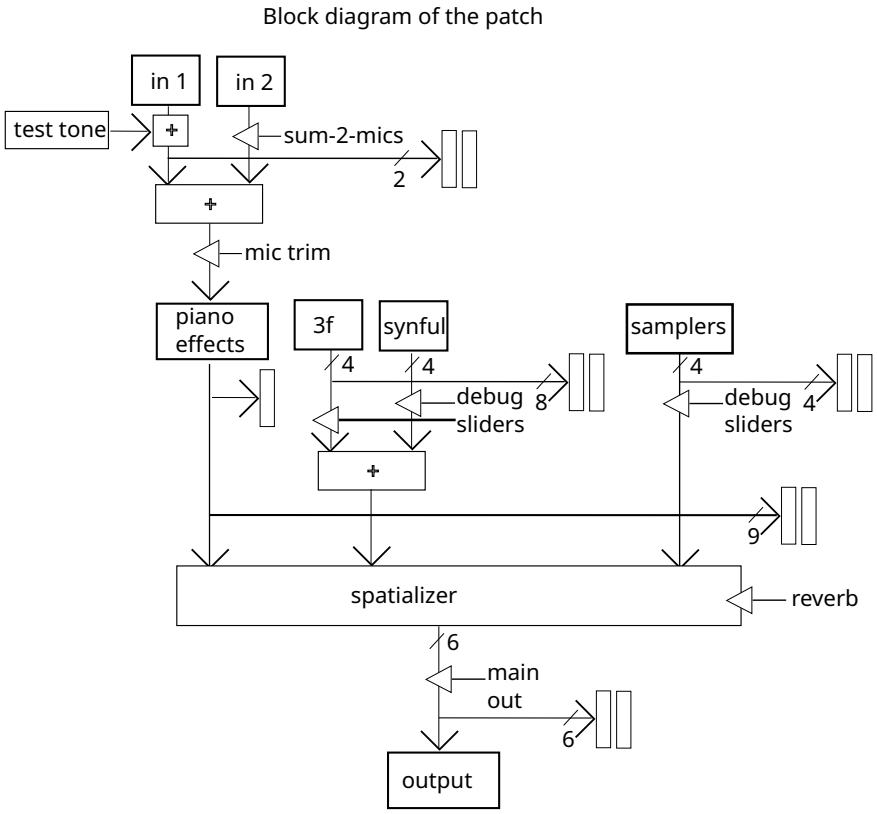


Figure 3: patch signal paths

usually with an accompanying comment, at the bottom of the patch window.

During rehearsal it is often desirable to fast-forward from the beginning of the piece to a specified cue. This is done with the "jump" control. The cue number is entered in the number box, and then the "jump" button causes the patch to reset, restart, and fast-forward to the cue. This is not foolproof, since many of the cues contain delayed parameter changes that are skipped when the next cue overtakes them (not to mention messages to stop loops and "whenever" actions that might be previously started by the antescofo script). As a hack, a button is provided that kills the most common of these, but there will be plenty of error printout. Usually things will be nearly enough correct to function as a rehearsal point.