

Requiem for Beauty

for soprano, ensemble and electronics

documentation for electronics

Shu-Yu Lin

Table of Contents

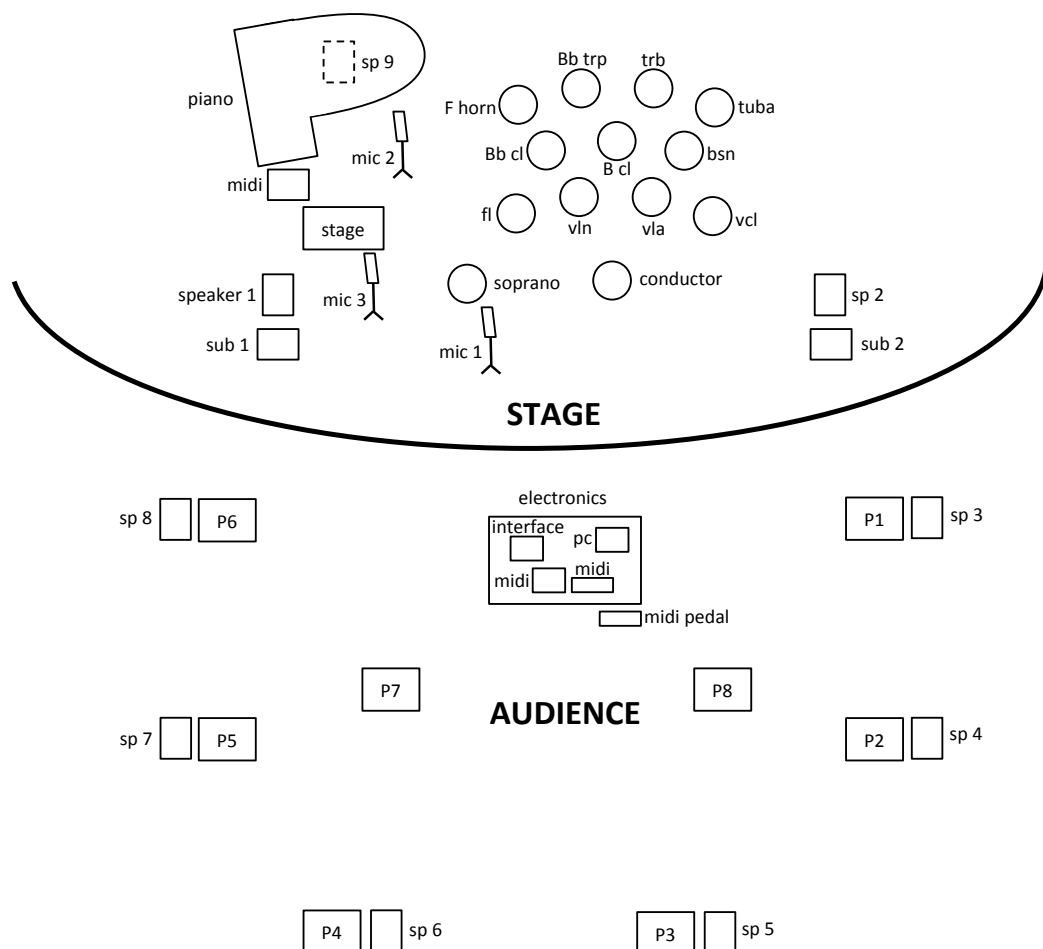
Technical specification	3
Stage plan	3
Performance instruction	4
Hardware setup	10
SuperCollider code instruction	16
Speaker test	28

Technical specification

- A midi controller with at least 7 turn knobs (eg. Akai LPD 8)
- A push button controller with at least 77 buttons and 8 sliders (eg. Akai APC mini)
- A foot midi controller with at least 4 pedals (eg. Actition 4 button midi foot pedal)
- A midi controller with at least 3 push buttons (eg. Akai MPK mini)
- 9 mid-range speakers
- 2 subwoofers (chain to each L and R mid-range speakers on stage)
- An audio interface with at least 9 outs and 3 ins
- 2 dynamic microphones (eg. Shure SM58)
- A condenser microphone (eg. Rode NT 5)
- A laptop computer
- SuperCollider programming environment

Stage plan

P1, P2 etc. refers to soprano's walking positions; stage refers to actor's stage position.



Performance Instruction

The electronics performer operates the SuperCollider script and midi controllers to generate sound at particular times that indicate on the score.

The Akai APC mini and Akai LPD 8 are suggested to be placed on a table with the laptop computer as most of the electronics operations are controlled using these devices.

For the purpose of engaging the performers in the electronics operations, the soprano and the pianist may be given controllers to operate certain electronics processes. The Actition foot pedal is ideally operated by the soprano on stage. If it is not possible, the electronics performer can operate on Akai APC mini as some of the buttons corresponds to the processing of the foot pedal. Or, the electronics performer may operate the foot pedal. The pianist triggers electronics processing on Akai MPK mini controller. Likewise, electronics performer can operate using APC if it is necessary.

Explanatory notes for score

△	Press button 1 on controller to trigger electronics; '2' refers to press button 2 etc.
△ off	Press button 1 again to turn off trigger; '2 off' refers to press button 2 again etc.
K1	Turn knob 1 on controller accordingly to modify computer-generated sound; K2 refers to knob 2 etc.
K4 back	Turn knob 4 back to 'min' on controller in order to get ready for next performance instruction
b1	Step on button 1 on foot controller to trigger electronics; 'b2' refers to step on button 2 etc.
b1 off	Step again on button 1 on foot controller to turn off trigger; 'b2 off' refers to stepping again on button 2 etc.
①	Press button 1 on controller to trigger electronics. Number '2' refers to button 2 etc.
① off	Press button 1 again to turn off trigger. Number '2 off' refers to pressing button 2 again etc.

Midi controllers

- Akai APC mini

Electronics performers are instructed to press a certain button on the push button controller to begin a particular electronics processing. As indicated in the explanatory notes above, triangle shapes enclose button numbers.

Buttons b1-b4 on the right (vertically arranged), corresponds to the ActioN foot pedal buttons 1-4. Buttons 1, 2 and 30 corresponds to the Akai MPK mini buttons 1, 2 and 3.

Each slider on the APC mini corresponds to the numbers above. For example, the slider 1 (counting from the left) controls the volume of the processing of the buttons 1, 9, 17, 25, 33, 41, 49, 57, 65 and slider 2 controls the volume of the processing of the buttons 2, 10, 18, 26, 34, 42, 50, 58, 66 etc. Slider 8, which labeled 'foot', controls also the volume of the processing of the foot pedal.



- ActioN foot pedal for soprano

The soprano is instructed to step on a specific button on the foot pedal according to the score. As indicated in the explanatory note, square shapes enclose the button numbers.



- Akai MPK mini

The pianist is instructed to press a specific button on the Akai MPK mini controller according to the score. As indicated in the explanatory note, the circle shapes enclose the button numbers.



- Akai LPD 8

The electronics performer is instructed to turn a specific knob at a particular time according to the score to operate the electronics processing. K1 refers knob 1, K2 refers to knob 2 etc.



Electronics numbering

For reference, the brief explanation of instructions of each button number is provided.

Movement	Electronics trigger number	Brief description
I	1 (also piano button trigger 1)	Soprano microphone signal project under piano through speaker 9 (mic 1 to channel 8)
	2 (also piano button trigger 2)	Piano mic signal project through speaker 5, 6 (mic 2 to ch 4, 5)
	3	Pan soprano signal (mic 1 signal) using knob 4 (K4)
	4	Granularize soprano pre-recorded reading of section 13 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
	5	Frequency modulate soprano pre-recorded reading of section 13 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7) using knob 1 (K1)
	6	Granularize soprano pre-recorded reading of sec 1 of text through speakers 3, 4, 7, 8 (ch 2, 3, 6, 7)
IV	7	Granularize soprano pre-recorded reading of section 10 through speakers 1, 2 (ch 0, 1); control frequency using knob 1 (K1), control grain duration using knob 2 (K2), control density using knob 5 (K5)
II	8	Record soprano (record from mic 1; REC 1)
	9	Granularize soprano pre-recorded reading of sec 12 of text through speakers 1, 2 (ch 0, 1); control the density of grain using knob 5 (K5), control frequency using knob 1 (K1)
	10	Record soprano (record from mic 1; REC 2)
	11	Amplify soprano through speakers 3, 4, 7, 8 (ch 2, 3, 6, 7)
	12	frequency modulate using knob 8 on soprano signal through speakers 5, 6 (mic 1 to ch 4, 5)
	13	Amplify soprano through speakers 1, 2 (mic 1 to ch 0, 1)
III	14	Granularize soprano pre-recorded reading of sec 11 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7); control frequency using knob 1 (K1), control grain duration using knob 2 (K2), control density using knob 5 (K5)
	15	Granularize REC 2 through speakers 1, 2 (ch 0, 1)
	16	Granularize REC 2 through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
IV	17	Granularize soprano pre-recorded reading of sec 10 of text through speaker 8 (ch 7); control frequency using knob 1 (K1), control grain duration using knob 2 (K2); control density using knob 5 (K5)
VI	18	Granularize soprano pre-recorded reading of sec 8 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7); control frequency using knob 1 (K1), control grain duration using knob 2 (K2), control density using knob 5 (K5), control panning using knob 6 (K6, clockwise panning)
	19	Granularize soprano pre-recorded reading of sec 6 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7); control frequency

		using knob 1 (K1), control grain duration using knob 2 (K2), control density using knob 5 (K5), control panning using knob 7 (K7, turn knob clockwise for anticlockwise panning)
VII	20	Granularize soprano pre-recorded reading of sec 7 through speakers 1, 2, 5, 6 (ch 0, 1, 4, 5)
	21	Granularize soprano pre-recorded reading of sec 7 through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7); control panning using knob 6 (K6)
VIII	22	Amplify soprano through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
	23	Granularize soprano pre-recorded reading of sec 6 of text through speakers 1, 2 (ch 0, 1)
IX	24	Granularize soprano pre-recorded reading of sec 5 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
	25	Granularize soprano pre-recorded reading of sec 9 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
X	26	Amplify soprano through speaker 5 (mic 1 to ch 4)
	27	Amplify soprano through speaker 7 (mic 1 to ch 6)
	28	Amplify soprano through speaker 4 (mic 1 to ch 3)
	29	Amplify soprano through speaker 6 (mic 1 to ch 5)
	30 (also piano button trigger 3)	Amplify soprano through speaker 9 (mic 1 to ch 8); amplify piano mic signal through speakers 1, 2 (mic 2 to ch 0, 1)
XI	31	Granularize soprano pre-recorded reading of sec 3 of text through speaker 9 (ch 8); amplify piano mic signal through speaker 8 (mic 2 to ch 7); control density using knob 5 (K5); amplitude control o slider 7
	32	Granularize soprano pre-recorded reading of sec 3 of text through speaker 9 (ch 8); amplify piano mic signal through speaker 7 (mic 2 to ch 6); control density using knob 5 (K5); amplitude control o slider 7
	33	Granularize soprano pre-recorded reading of sec 3 of text through speaker 9 (ch 8); amplify piano mic signal through speaker 6 (mic 2 to ch 5); control density using knob 5 (K5); amplitude control o slider 7
	34	Granularize soprano pre-recorded reading of sec 3 of text through speaker 9 (ch 8); amplify piano mic signal through speaker 5 (mic 2 to ch 4); control density using knob 5 (K5); amplitude control o slider 7
	35	Granularize soprano pre-recorded reading of sec 3 of text through speaker 9 (ch 8); amplify piano mic signal through speaker 4 (mic 2 to ch 3); control density using knob 5 (K5); amplitude control o slider 7
XII	36	Granularize soprano pre-recorded reading of sec 2 of text through speaker 9 (ch 8); amplify piano mic signal through speaker 1, 2, 3, 4, 5, 6, 7, 8 (mic 2 to ch 0, 1, 2, 3, 4, 5, 6, 7); control frequency using knob 1 (K1)
	37	Granularize soprano pre-recorded reading of sec 2 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)

XIII	38	Granularize soprano pre-recorded reading of sec 1 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
	39	Granularize soprano pre-recorded reading of sec 1 of text through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7); script content different from button 38
	40	Amplify soprano pre-recorded reading of text 'huí yīng' through speakers 5, 6 (ch 4, 5)
	41	Amplify soprano pre-recorded reading of text 'shēng xiǎng' through speakers 5, 6 (ch 4, 5)
	42	Amplify soprano pre-recorded reading of text 'dòng jìng' through speakers 5, 6 (ch 4, 5)
	43	Amplify soprano pre-recorded reading of text 'qì xī' through speakers 5, 6 (ch 4, 5)
	44	Amplify actor (mic 3) through speakers 1, 2, 3, 4, 5, 6, 7, 8 (ch 0, 1, 2, 3, 4, 5, 6, 7)
I	45	Playback pre-recorded clarinet G note for 8 seconds
IV	46	Playback pre-recorded clarinet C note for 13 seconds
VII	47	Playback pre-recorded clarinet F# note for 9 seconds
	48	Playback pre-recorded clarinet B note for 8 seconds
IX	49	Playback pre-recorded clarinet Bb note for 8 seconds
X	50	Playback pre-recorded clarinet F note for 3 seconds
XI	51	Playback pre-recorded clarinet Eb note for 8 seconds
	52	Playback pre-recorded clarinet F note for 8 seconds
XIII	53	Playback pre-recorded clarinet Bb note for 8 seconds
	54	Playback pre-recorded clarinet Eb note for 8 seconds
	55	Playback pre-recorded clarinet Eb note for 8 seconds (different recording from button 54)
	56	Amplify soprano through speakers 1, 2, 5, 6, 9 (ch 0, 1, 4, 5, 8)
I	B1	Spectral filter on soprano voice through speakers 1, 2 (mic 1 to ch 0, 1); broad range frequency
IX	B2	Spectral filter on soprano voice through speakers 1, 2 (mic 1 to ch 0, 1); high frequency output
	B3	Spectral filter on soprano voice through speakers 1, 2 (mic 1 to ch 0, 1); low frequency output
XIII	B4	Spectral filter on soprano voice through speakers 1, 2 (mic 1 to ch 0, 1); broad range frequency; script content different from B1

Source of tape recording

Clare Lesser, the soprano of the premier (June 7th, 2019) of this piece, recorded her reading of the text. The vocal recording is used as part of the electronics' sound generating process. During the last minutes of completing the piece, with generous help, clarinetist Chiung-Yu Ku recorded his clarinet playing for the pre-recorded parts of the electronics.

Contact

If you have question concerning the code, you may get in touch with Shu-Yu by email at shuyuslin@gmail.com

Hardware setup

Audio interface

1. Connect audio interface to your laptop computer.
2. Choose output to be the audio interface. If are using Mac OS, go to Systems Preferences -> Sound -> Output -> [name of the audio interface].
3. Choose the sample rate of the interface. If you are using a Mac OS, open Audio Devices application. On the left panel choose your interface. Click on Output, and from the Format drop down menu, choose 44,100 Hz as sampling rate. Also choose 44,100 Hz as Input sampling rate. Click on Input and select 44,100 Hz as input sampling rate.

Speakers

1. Place 2 speakers on the left and the right of the stage, 6 speakers around the performance space evenly and 1 speaker underneath the piano facing up to the piano soundboard. Open the piano to its largest without taking it apart.
2. Place the subwoofers in front of the L and R mid-range speakers on the stage. Chain each subwoofer to each L and R speaker.
3. Connect the speakers to your audio interface sequentially. Make sure the speaker number and the interface output correspond. For example speaker 1 connects to interface output 1 etc.
4. Connect power to the 9 speakers and 2 subwoofers.
5. Turn on the speakers.

Microphones

1. Place a dynamic microphone in front of the position of the soprano.
2. Place another dynamic microphone in front of the position of the actor.
3. Place a condenser microphone at the piano, aiming at the soundboard.
4. Connect microphones to audio interface. Soprano microphone connects to interface input 1, piano microphone connects to input 2, and the actor microphone connects to input 3.
5. Turn on phantom power for the condenser microphone (piano mic) on the interface.

Midi controllers

- Midi controller with at least 7 turn knobs:
 - > If you are using Akai LPD 8:
 1. Label the knobs as follows:
 - K1 knob 'gran rate' (rate of granular synthesis)
 - K2 knob 'grain dur' (duration of each grain)

- K4 'sop pan' (panning of the soprano voice)
- K5 'cloud density' (density of the granular cloud)
- K6 'gran pan clockwise' (clockwise panning of the granular result from the position of the electronics performer facing the stage)
- K7 'gran pan anticlock' (anticlockwise panning of the granular result from the position of the electronics performer facing the stage)
- K8 'freq mod sop sig' (amount of frequency modulation on the signal receiving from the soprano microphone)

Each knob label corresponds to the instructions in the electronics part of the score.



2. Connect the device to the computer.

-> If you are using a different device with at least 7 knobs:

1. You may choose to label the 7 knobs on your device the way you wanted as long as it is comprehensible that knob 1 corresponds to the rate of granular synthesis, knob 2 corresponds to the duration of each grain etc. This is because the knob instructions written on the score make use of the labels K1, K2, K4 etc.

Note that the knob 3 on the LPD 8 is skipped. If you would want to use the knobs without skipping any, make sure you correspond your labels with the instructions written on the score. For example, if you label knob 3 on your device as 'cloud density', which on the LPD 8 would be knob 4, then note that the instruction for K4 on the score refers to your K3.

If you choose to label your device the same way as the image shows, skipping knob 3, then knob instructions will correspond.

2. Find and note the midi numbers of each knob. These numbers will be used to modify the midi numbers in the *Requiem* code that was written for LPD 8. The next section (Supercollider code instruction) specifies which parameters at which lines in the code need to be modified.
3. Because the electronics part of the score was written with the LPD labels, find the corresponding processes to the specific knob that you will be using. For example, if you label knob 4 as 'cloud density', find the corresponding knob for controlling

cloud density on LPD label. In this case, knob 4 corresponds to knob 5 on LPD. On the score, instruction for K5 refers to the instruction for your K4 knob. ()

4. Connect the device to the computer.

- Push button controller:

-> If you are using Akai APC mini:

1. Label the buttons as shown in the image.

Buttons 1 to 64 triggers electronics processing written on the score.

Buttons 65 to 73 play back the noise sound file for speaker testing where 65 plays back noise through speaker 1 and 66 plays back through speaker 2 etc.

Buttons b1, b2, b3 and b4 correspond to the foot pedal processing b1, b2, b3 and b4.

Button 1 and 2 include encircled 1 and 2. Encircled numbers are instructions for the pianist to trigger specific button on the MPK controller. Buttons 1 and 2 on APC mini corresponds to the buttons 1 and 2 on the MPK, and corresponds to the encircled 1 and 2 on the score in the piano part.

Notes for some of the electronics processing are written on the labels. For example, button 8 and 10 consist of REC notes. This is to indicate these two buttons triggers live recording. You may choose not to include notes on the labels, or write your own notes on the labels.

The explanatory notes for score and the electronics numbering in the performance instruction section describes the number symbols on the score corresponding to the button numbers, and the electronic process of each number.



2. Label the 8th slider foot. This slider controls the output volume of the sound processes trigger by the foot pedal.

3. Connect the device to your computer.

-> If you are using a different push button midi controller with at least 77 buttons and 8 sliders:

1. Label the buttons from numbers 1 to 73 and b1, b2, b3 and b4.

2. Label one of the slider 'foot' to indicate slider control for the output volume of sound processing from the foot pedal.

3. Find and note the midi numbers of each button. These numbers will be used to modify the midi numbers in the *Requiem* code that was written for APC mini. The next section (Supercollider code instruction) specifies which parameters at which lines in the code need to be modified.

4. Connect the device to your computer.

- Foot midi controller with at least 4 pedals:

-> If you are using ActioN 4 button midi foot pedal:

1. Label each pedal button from left to right b1, b2, b3 and b4.



2. Connect to your computer.

-> If you are using a different device with at least 4 foot pedals:

1. Label 4 of the pedals b1, b2, b3 and b4.

2. Find and note the midi numbers of each pedal. These numbers will be used to modify the midi numbers in the *Requiem* code that was written for ActioN 4. The next section (Supercollider code instruction) specifies which parameters at which lines in the code need to be modified.

3. Connect the device to your computer.

- Midi controller with at least 3 push buttons:

-> If you are using Akai MPK mini:

1. Label pads 5, 6 and 7 to be 1, 2 and 3.



2. Connect it to the computer.

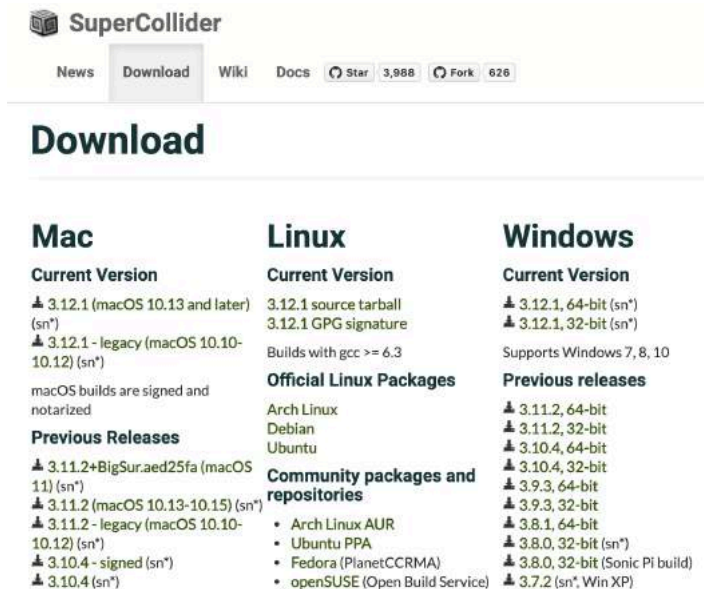
-> If you are using a different device with at least 3 push buttons:

1. Label 3 of the pads to be 1, 2 and 3.

2. Find and note the midi numbers of each pedal. These numbers will be used to modify the midi numbers in the *Requiem* code that was written for Akai MPK mini. The next section (Supercollider code instruction) specifies which parameters at which lines in the code need to be modified.
3. Connect the device to your computer.

SuperCollider code instruction

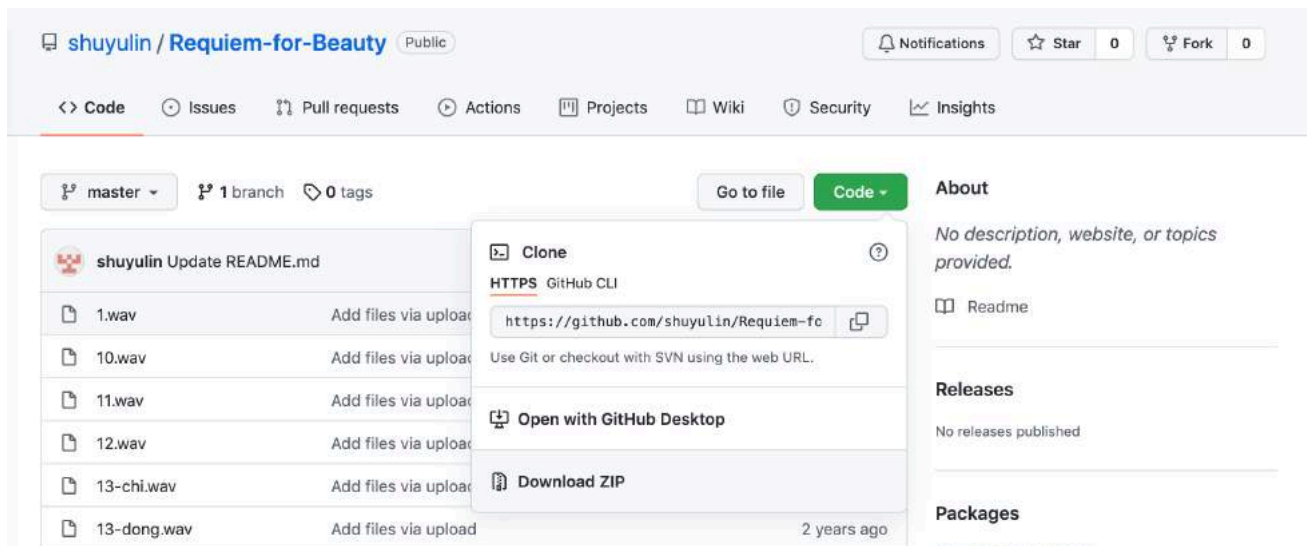
1. Download SuperCollider at <https://supercollider.github.io/download> Make sure the version that you download is compatible with your computer's operating system. Click open the downloaded zip file and install SuperCollider.



The screenshot shows the 'Download' page of the SuperCollider GitHub repository. It is organized into three columns: Mac, Linux, and Windows. Each column lists the current version and previous releases with their respective operating system requirements and signatures. The Linux column also includes official and community packages for various distributions like Arch Linux, Debian, Ubuntu, Fedora, and openSUSE.

Mac	Linux	Windows
Current Version 3.12.1 (macOS 10.13 and later) (sn*) 3.12.1 - legacy (macOS 10.10-10.12) (sn*) macOS builds are signed and notarized	Current Version 3.12.1 source tarball 3.12.1 GPG signature Builds with gcc >= 6.3 Official Linux Packages Arch Linux Debian Ubuntu Community packages and repositories • Arch Linux AUR • Ubuntu PPA • Fedora (PlanetCCRMA) • openSUSE (Open Build Service)	Current Version 3.12.1, 64-bit (sn*) 3.12.1, 32-bit (sn*) Supports Windows 7, 8, 10 Previous releases 3.11.2, 64-bit 3.11.2, 32-bit 3.10.4, 64-bit 3.10.4, 32-bit 3.9.3, 64-bit 3.9.3, 32-bit 3.8.1, 64-bit 3.8.0, 32-bit (sn*) 3.8.0, 32-bit (Sonic Pi build) 3.7.2 (sn*, Win XP)

2. Download files for electronics from github at <https://github.com/shuyulin/Requiem-for-Beauty> Click on 'Code' and select 'Download ZIP'.

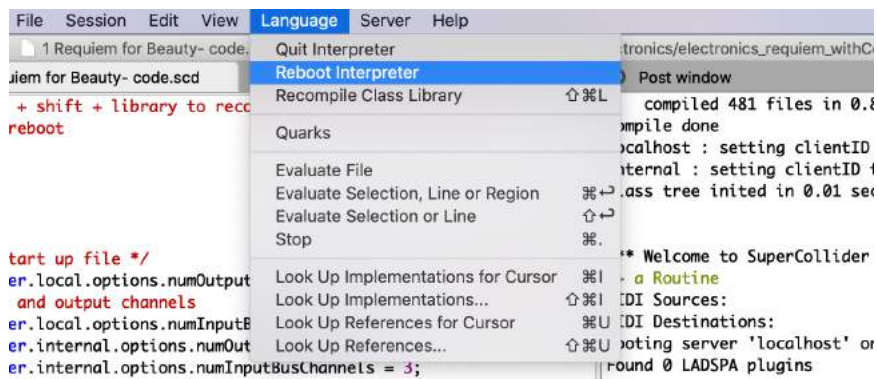


The screenshot shows the GitHub repository page for 'shuyulin / Requiem-for-Beauty'. The 'Code' button is highlighted, and a dropdown menu is open, showing options to 'Clone' (with HTTPS and GitHub CLI options), 'Open with GitHub Desktop', and 'Download ZIP'. The repository content shows a list of audio files (1.wav to 13-dong.wav) and a README.md file.

3. Open zip and move to a directory where you wanted it to be.

4. Click open the SuperCollider script 'rb.scd'.

Note: The Interpreter should indicate Active after you opened the .scd file. Reboot the Interpreter if it is inactive. To do this, go to Language drop down menu and select Reboot Interpreter.



5. Modify the numbers of channel output if necessary.

-> If you are using 9-speaker setup with 2 subwoofers (original plan for the *Requiem*)
 The number of output and input channels from lines 26 to 29 do not need to be changed. If you would want to have exact number of channel correspond to the input and output, modify the number accordingly.

1. Change number of output bus channels to 9 at lines 26 and 28. The output channel 0 and 1 includes the subwoofer outputs because they are chained to the L and R speakers (speakers 1 and 2) on the stage, so it is not necessary to have channel number for subwoofers.

```

25  /* start up file */
26  Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27  Server.local.options.numInputBusChannels = 3;
28  Server.internal.options.numOutputBusChannels = 11;
29  Server.internal.options.numInputBusChannels = 3;
  
```

2. Change number of input bus channels to 3 at lines 27 and 29.

```

25  /* start up file */
26  Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27  Server.local.options.numInputBusChannels = 3;
28  Server.internal.options.numOutputBusChannels = 11;
29  Server.internal.options.numInputBusChannels = 3;
  
```

-> If you are using stereo setup with additional speaker at the piano
 If 9-speaker setup with 2 subwoofers is not possible at the time to rehearsal, however, 3 speakers are available (stereo and another speaker for the piano), modify the number of input and output channels and their corresponding global variables.

1. Change number of output bus channels to 2 at lines 26 and 28.

```

25  /* start up file */
26  Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27  Server.local.options.numInputBusChannels = 3;
28  Server.internal.options.numOutputBusChannels = 11;
29  Server.internal.options.numInputBusChannels = 3;
  
```

2. Change number of input bus channels to 2 at lines 27 and 29.

```

25  /* start up file */
26  Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27  Server.local.options.numInputBusChannels = 3;
28  Server.internal.options.numOutputBusChannels = 11;
29  Server.internal.options.numInputBusChannels = 3;
  
```

3. Modify numbers assigned to the global variables. Change ~ch2, ~ch3 and ~ch4 to 1 for stereo right output. Change ~ch5, ~ch6 and ~ch7 to 0 for stereo left output. Change ~ch8 to 2 for speaker output at the piano.

```

121 ///////////////////////////////////////////////////* function *////////////////////////////////////
122 /* global variables for Motu setup */
123 ~ch0 = 0; // channel outputs
124 ~ch1 = 1;
125 ~ch2 = 2;
126 ~ch3 = 3;
127 ~ch4 = 4;
128 ~ch5 = 5;
129 ~ch6 = 6;
130 ~ch7 = 7;
131 ~ch8 = 8;
132 //~ch9 = 9; // optional variable for sub 1
133 //~ch10 = 10; // optional variable for sub 2
134 ~mic1 = 0; // mic inputs
135 ~mic2 = 1;
136 ~mic3 = 2;

```

-> If you are using built-in input and output

If you do not have microphone and external speakers at the time of running the code, you may choose to use the computer's built-in input and outputs.

1. Since no signal will come into input channels 1 and 2, you may indicate the input to be 1 for the single build-in microphone input at lines 27 and 29.

```

25 /* start up file */
26 Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27 Server.local.options.numInputBusChannels = 3;
28 Server.internal.options.numOutputBusChannels = 11;
29 Server.internal.options.numInputBusChannels = 3;

```

2. Change number of output bus channels to 3 at lines 26 and 28. However, without the 3rd speaker available, no signal can be output through channel 2 (counting from 0, speaker output 1 to channel 0, output 2 to ch 1 and output 3 to ch 2).

```

25 /* start up file */
26 Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27 Server.local.options.numInputBusChannels = 3;
28 Server.internal.options.numOutputBusChannels = 11;
29 Server.internal.options.numInputBusChannels = 3;

```

3. Modify numbers assigned to the global variables. Change ~ch2, ~ch3 and ~ch4 to 1 for stereo right output. Change ~ch5, ~ch6 and ~ch7 to 0 for stereo left output. Change ~ch8 to 2 for the output that is suppose to be at the piano.

```

121 ///////////////////////////////////////////////////* function *////////////////////////////////////
122 /* global variables for Motu setup */
123 ~ch0 = 0; // channel outputs
124 ~ch1 = 1;
125 ~ch2 = 2;
126 ~ch3 = 3;
127 ~ch4 = 4;
128 ~ch5 = 5;
129 ~ch6 = 6;
130 ~ch7 = 7;
131 ~ch8 = 8;
132 //~ch9 = 9; // optional variable for sub 1
133 //~ch10 = 10; // optional variable for sub 2
134 ~mic1 = 0; // mic inputs
135 ~mic2 = 1;
136 ~mic3 = 2;

```

6. Indicate audio device.

-> If you are using an external audio interface

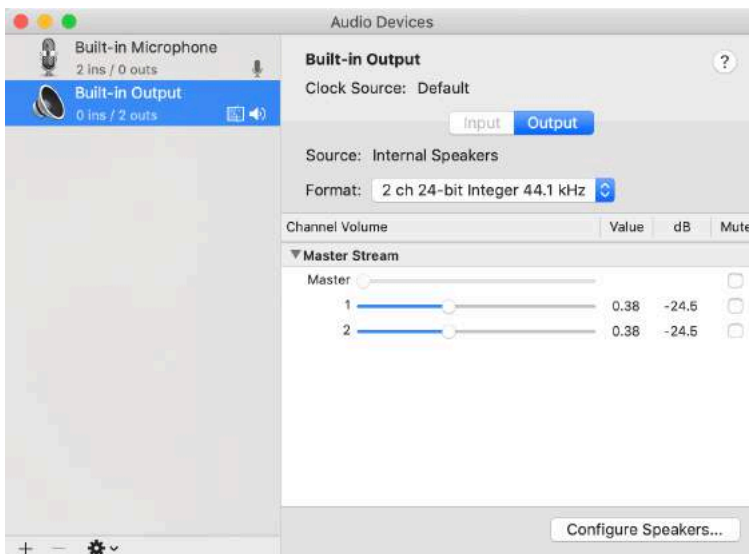
1. Find the name of your audio interface. If you are using a Mac OS, open Audio Devices application, the names of audio input and output devices are displayed at the left panel.

2. Go to line 30 and modify the name of the audio interface to the one you are using. You may use different input and output audio interface. If so, indicate your input device at line 31 (.inDevice) and indicate your output device at line 32 (.outDevice).

```
25  /* start up file */
26  Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27  Server.local.options.numInputBusChannels = 3;
28  Server.internal.options.numOutputBusChannels = 11;
29  Server.internal.options.numInputBusChannels = 3;
30  Server.local.options.device = "MOTU 896mk3 Hybrid";
31  //Server.local.options.inDevice = "Scarlett 2i4 USB"; // other possible audio interfaces
32  //Server.local.options.outDevice = "Built-in Output";
33  //Server.local.options.outDevice = "Orpheus (0979)";
```

-> If you are using computer's built-in audio

1. Find the name of the name of the built-in microphone and the name of built-in output. If you are using a Mac OS, open Audio Device application, the names of the built-in input and output devices are displayed at the left panel. The name of the built-in input is 'Built-in Microphone' and built-in output is 'Built-in Output'.



2. Go to line 31 and 32 and modify the names of the in and out devices to the names of the built-in input and built-in output.

7. Change directories of pre-recorded sound files. Find the directory of the downloaded sound files, which should be the same directory as the code if you downloaded files in zip. Go to lines 50 to 110 and change the buffer directories of the corresponding sound files. For example, ~bufSec1 corresponds to 1.wav sound file, ~bufSec2 corresponds to 2.wav etc. Make sure the directories are enclosed by " " and that the directory begins with / .

```

50 ~bufSec1 = Buffer.read(s, "/Users/shuyulin/Desktop/1.wav"); // load soundfile
51 s.sync;
52 ~bufSec2 = Buffer.read(s, "/Users/shuyulin/Desktop/2.wav");
53 s.sync;
54 ~bufSec3 = Buffer.read(s, "/Users/shuyulin/Desktop/3.wav");
55 s.sync;
56 ~bufSec5 = Buffer.read(s, "/Users/shuyulin/Desktop/5.wav");
57 s.sync;
58 ~bufSec6 = Buffer.read(s, "/Users/shuyulin/Desktop/6.wav");
59 s.sync;
60 ~bufSec7 = Buffer.read(s, "/Users/shuyulin/Desktop/7.wav");
61 s.sync;
62 ~bufSec8 = Buffer.read(s, "/Users/shuyulin/Desktop/8.wav");
63 s.sync;
64 ~bufSec9 = Buffer.read(s, "/Users/shuyulin/Desktop/9.wav");
65 s.sync;
66 ~bufSec10 = Buffer.read(s, "/Users/shuyulin/Desktop/10.wav");
67 s.sync;
68 ~bufSec11 = Buffer.read(s, "/Users/shuyulin/Desktop/11.wav");
69 s.sync;
70 ~bufSec12 = Buffer.read(s, "/Users/shuyulin/Desktop/12.wav");
71 s.sync;
72 ~bufSec13 = Buffer.read(s, "/Users/shuyulin/Desktop/13.wav");
73 s.sync;
74 ~bufSec13_hui = Buffer.read(s, "/Users/shuyulin/Desktop/13-hui.wav");
75 s.sync;
76 ~bufSec13_sheng = Buffer.read(s, "/Users/shuyulin/Desktop/13-sheng.wav");
77 s.sync;
78 ~bufSec13_dong = Buffer.read(s, "/Users/shuyulin/Desktop/13-dong.wav");
79 s.sync;
80 ~bufSec13_chi = Buffer.read(s, "/Users/shuyulin/Desktop/13-chi.wav");
81 s.sync;

```

8. Setup midi controllers (from lines 1104)

-> If you are using the same midi controllers as the ones used during the composition and the premier of the *Requiem*.

You do not need to modify the midi setups. But make sure the labels on your controllers correspond to the images of the controllers in the previous instruction, and make sure the devices are properly connected to your computer.

-> If you are using different midi devices.

1. Modify the names of the midi controllers in the code from lines 1110 to 1124.

- To change the name of foot pedal controller, go to lines 1110 and 1112 and replace the ActioN MIDI Controller to the name of your device.

- To change the name of the push button controller with at least 77 buttons and 8 sliders, go to lines 1114 and 1116 and replace APC MINI to the name of your device.

- To change the name of the push button controller with at least 3 push buttons, go to lines 1118 and 1120 and replace MPKmini2 to the name of your device.

- To change the name of the midi controller with at least 7 turn knobs, go to lines 1122 and 1124 and replace LPD8 to the name of your device.

```

1107
1108 /* get midi controller IDs */
1109 MIDIClient.sources.do{|item|
1110   if(item.name == "Actition MIDI Controller", {~footID = item.uid}); }; // foot controller
1111 MIDIClient.destinations.do{|item|
1112   if(item.name == "Actition MIDI Controller", {~footOutID = item.uid}); };
1113 MIDIClient.sources.do{|item|
1114   if(item.name == "APC MINI", {~apcID = item.uid}); }; // pad controller
1115 MIDIClient.destinations.do{|item|
1116   if(item.name == "APC MINI", {~apcOutID = item.uid}); };
1117 MIDIClient.sources.do{|item|
1118   if(item.name == "MPKmini2", {~mpkID = item.uid}); }; // keyboard controller
1119 MIDIClient.destinations.do{|item|
1120   if(item.name == "MPKmini2", {~mpkOutID = item.uid}); };
1121 MIDIClient.sources.do{|item|
1122   if(item.name == "LPD8", {~lpdID = item.uid}); }; // keyboard controller
1123 MIDIClient.destinations.do{|item|
1124   if(item.name == "LPD8", {~lpdOutID = item.uid}); };
1125
1126 /* set midi outputs */
1127 m = MIDIOut(0, ~apcOutID);
1128 n = MIDIOut(0, ~mpkOutID);
1129 //f = MIDIOut(0, ~footOutID);
1130 l = MIDIOut(0, ~lpdOutID);
1131 "MIDIOut done".postln;

```

* Do not change the name of the global variables eg. ~apcID because these are the names of the containers. Each contains the name of your device and is used in other parts of the code. Assigning the names of your controllers to these global variables (however the name could be) should be sufficient to link your device to the code. If you want the names of the global variables to correspond to your devices, make sure you change all the names of the corresponding global variables in the code.

2. Change the midi numbers of each device.

- For a push button controller with at least 77 buttons and 8 sliders (in place of Akai APC mini), go to line 1133 where commented */* APC slider controls */*, modify the midi number associate with each slider in each MIDIDef.cc from lines 1133-1694. The midi number is in purple before srcID.

```

1133 /* APC slider controls */
1134 /* slider 1 */
1135 MIDIDef.cc(\slider1_button1, {
1136   arg num, id, chan, src;
1137   //[num, id, chan, src].postln;
1138   Ndef(\bar1SopSp9).set(\vol, num.linlin(0, 127, 0, 1)); // control
1139   amplitude output from mic 1 (sop mic) to speaker 9 (piano speaker)
1140   }, 48, srcID: ~apcID);

1190 /* slider 2 */
1191 MIDIDef.cc(\slider2_button2, {
1192   arg num, id, chan, src;
1193   //[num, id, chan, src].postln;
1194   Ndef(\bar1PnoToSp).set(\vol, num.linlin(0, 127, 0, 1));
1195   }, 49, srcID: ~apcID);
1196

```

Go to line 1878 where commented `/* trigger sound; APC */`, modify the midi number associate with each button in each `MIDIdef.noteOn` from lines 1959 to 3029 and lines 3098-3403. The midi number is in purple before the `srcID`.

```

1958 ////////////////////////////////////////////////// bar 1 ////////////////////////////////////// button 1
1959 MIDIdef.noteOn(\apcButton56, {
1960   arg vel, nn, chan, src;
1961   /* output soprano voice to speaker 9 */
1962   if (~countApc1 == 0, {
1963     ~bar1SopSp9 = Ndef(\bar1SopSp9, ~bar1SopToSpeaker9);
1964     m.noteOn(0, 56, 01); //light on
1965     "on 56 button 1".postln;
1966     ~countApc1 = 1;
1967   }, {
1968     ~bar1SopSp9.release(1);
1969     m.noteOn(0, 56, 0); //light off
1970     "off 56 button 1".postln;
1971     ~countApc1 = 0;
1972   });
1973 }, 56, srcID: ~apcID);
1974

```

```

3098 //***** soprano amplification controls *****/
3099 ////////////////////////////////////////////////// soprano amplify through speaker 1 ////////////////////////////////////// button 57
3100 MIDIdef.noteOn(\apcButton0, {
3101   arg vel, nn, chan, src;
3102   if (~countApc57 == 0, {
3103     ~sopThru1 = Ndef(\sopThru1, ~amplifySopThru1);
3104     m.noteOn(0, 0, 01); //light on
3105     "amplify soprano through speaker 1".postln;
3106     "on 0 button 57".postln;
3107     ~countApc57 = 1;
3108   }, {
3109     ~sopThru1.release(1);
3110     m.noteOn(0, 0, 0); //light off
3111     "off 0 button 57".postln;
3112     ~countApc57 = 0;
3113   });
3114 }, 0, srcID: ~apcID);
3115

```

- For a foot midi controller with at least 4 pedals (in place of Actition 4 button midi foot pedal), go to line 3032, modify the midi number in each `MIDIdef.noteOn` from lines 3034-3095. The midi number is in purple before the `srcID`.

```

3031 //***** foot pedal can also control these *****/
3032 ////////////////////////////////////////////////// foot controller B1 //////////////////////////////////////
3033 MIDIdef.noteOn(\apcB1, {
3034   arg vel, nn, chan, src;
3035   if (~countApcB1 == 0, {
3036     ~bar9SpecSop = Ndef(\bar9SpecSop, ~bar9spec);
3037     ~countApcB1 = 1;
3038     m.noteOn(0, 82, 01); //light on
3039     "on 82, apc B1".postln;
3040   }, {
3041     ~bar9SpecSop.release(2);
3042     ~countApcB1 = 0;
3043     m.noteOn(0, 82, 0); //light on
3044     "off 82, apc B1".postln;
3045   });
3046 }, 82, srcID: ~apcID);
3047
3048

```

- For a midi controller with at least 3 push buttons (in place of Akai MPK mini), go to line 1825 where marked `/* Akai MPK mini 2 control mapping; trigger by pianist */`. Modify the midi number in each `MIDIdef.cc` from lines 1832-1873.

```

1831 //////////////// bar 1 //////////////// MPK button 1
1832 MIDIdef.cc(\mpkButton24, {
1833   arg vel, nn, chan, src;
1834   /* button press for pad 5 on MPK */
1835   /* project soprano voice (pick up by Shure SM 58) into piano through speaker 9 (channel 8) */
1836   if (~countMpk1 == 0, {
1837     ~bar1SopSp9 = Ndef(\bar1SopSp9, ~bar1SopToSpeaker9);
1838     "on MPK cc 24, pad 5- 1".postln;
1839     ~countMpk1 = 1;
1840   }, {
1841     ~bar1SopSp9.release(1);
1842     "off MPK cc 24, pad 5- 1".postln;
1843     ~countMpk1 = 0;
1844   });
1845   }, 24, srcID: ~mpkID); // MPK pad 5 midi number 24
1846

```

- For a midi controller with at least 7 turn knobs (in place of Akai LPD 8) go to /* LPD control mapping */ at line 1596. Modify the midi number of each knob in each MIDIdef.cc to the corresponding midi number of the knob of your choice. The midi number is in purple before the srcID.

```

1595
1596 /* LPD control mapping */
1597 /* LPD knob 1 control frequency/ rate of granular synth */
1598 MIDIdef.cc(\lpdKnob1GranRateBar6, {
1599   arg num, nn, chan, src;
1600   //[num, nn, chan, src].postln;
1601   Ndef(\bar6GranMod).set(\maxRate, num.linlin(0, 127, 0, 2));
1602   }, 1, srcID: ~lpdID);
1603

```

-> If you do not want to connect midi controllers.

When testing certain parts of the code, for example, testing the input signals from microphones, you may want to run the script without connecting midi devices. To do this, comment out the part of code that associates with midi by putting a /* at line 1106 and */ at line 3420. Once you put both the /* and */ at the appropriate lines, the part you commented out should appear in red.

```

1104
1105 "func and def done".postln;
1106 /*
1107
1108 /* get midi controller IDs */
1109 MIDIclient.sources.do{|item|
1110   if(item.name == "Actition MIDI Controller", {~footID = item.uid}); }; // foot controller

3417 ~midiMapFoot.();
3418
3419 "midiMap done".postln;
3420 /*
3421 "done!".postln;
3422   };
3423 }.fork
3424

```

If you comment out the midi part of the code, you may not be able to use any midi device to control sound processes or run any part of the sound processes, unless you modify the code in the way that suits your habit of sound testing and performance without using midi but other devices.

* If you do not comment out the midi part of the code when you do not have midi devices connected to your computer, and proceed to run the script, an error will show in the post window with the message, 'ERROR: Message 'uid' not understood.'

```
func and def done
ERROR: Message 'uid' not understood.
RECEIVER:
  nil
```

-> If you want to connect one or more but not all the controllers
When testing certain parts of the code that associate with a particular midi controllers, you may comment out the parts that do not associate with the particular devices.

1. Comment out the associated MIDIclient.sources, MIDIclient.destinations and MIDIOut from lines 1108 to 1131. In the example, the midi controller been tested is APC and the part of the code that do not associate with APC is commented out.

```
1107
1108 /* get midi controller IDs */
1109 /*MIDIclient.sources.do{|item|
1110   if(item.name == "Actition MIDI Controller", {-footID = item.uid}); }; // foot controller
1111 MIDIclient.destinations.do{|item|
1112   if(item.name == "Actition MIDI Controller", {-footOutID = item.uid}); };*/
1113 MIDIclient.sources.do{|item|
1114   if(item.name == "APC MINI", {-apcID = item.uid}); }; // pad controller
1115 MIDIclient.destinations.do{|item|
1116   if(item.name == "APC MINI", {-apcOutID = item.uid}); };
1117 /*MIDIclient.sources.do{|item|
1118   if(item.name == "MPKmini2", {-mpkID = item.uid}); }; // keyboard controller
1119 MIDIclient.destinations.do{|item|
1120   if(item.name == "MPKmini2", {-mpkOutID = item.uid}); };
1121 MIDIclient.sources.do{|item|
1122   if(item.name == "LPD8", {-lpdID = item.uid}); }; // keyboard controller
1123 MIDIclient.destinations.do{|item|
1124   if(item.name == "LPD8", {-lpdOutID = item.uid}); };*/
1125
1126 /* set midi outputs */
1127 m = MIDIOut(0, -apcOutID);
1128 /*n = MIDIOut(0, -mpkOutID);
1129 //f = MIDIOut(0, -footOutID);
1130 l = MIDIOut(0, -lpdOutID);*/
1131 "MIDIOut done".postln;
1132
```

2. Comment out the lines that do not associate with your midi controller(s) from lines 1135 to 3417. In the example, the controller being tested is APC mini, the lines that do not associate with APC are commented out.

```
1590 MIDIdef.cc(\slider9_button89, {
1591   arg num, id, chan, src;
1592   //[num, id, chan, src].postln;
1593   Ndef(\sweep9).set(\amp, num.linlin(0, 127, 0, 0.1));
1594   }, 56, srcID: ~apcID);
1595
1596 /* LPD control mapping */
1597 /* LPD knob 1 control frequency/ rate of granular synth */
1598 /*MIDIdef.cc(\lpdKnob1GranRateBar6, {
1599   arg num, nn, chan, src;
1600   //[num, nn, chan, src].postln;
1601   Ndef(\bar6GranMod).set(\maxRate, num.linlin(0, 127, 0, 2));
1602   }, 1, srcID: ~lpdID);
1603
1604 MIDIdef.cc(\lpdKnob1GranRateBar32, {
```



```

1759 /* foot pedal control mapping */
1760 ~midiMapFoot = {
1761     ~countFt = 0;
1762     ~countFt2 = 0;
1763     ~countFt3 = 0;
1764     ~countFt4 = 0;
1765     /////////////////////////////////////////////////// foot controller /////////////////////////////////////////////////// button 1 (96)
1766     MIDIdef.cc(\footButton1, {
1767         arg vel, nn, chan, src;
1768         if (~countFt == 0, {
1769             ~bar9SpecSop = Ndef(\bar9SpecSop, ~bar9spec);
1770             ~countFt = 1;
1771             "on foot button1, 96".postln;
1772         }, {
1773             ~bar9SpecSop.release(2);
1774             ~countFt = 0;
1775             "off foot button1, 96".postln;
1776         });
1777     }, 96, srcID: ~footID); // pedal button 96
1778
1779     /////////////////////////////////////////////////// foot controller /////////////////////////////////////////////////// button 2 (97)

```

```

1825 /* Akai MPK mini 2 control mapping; trigger by pianist */
1826 ~midiMapMpk = {
1827     ~countMpk1 = 0;
1828     ~countMpk2 = 0;
1829     ~countMpk3 = 0;
1830
1831     /////////////////////////////////////////////////// bar 1 /////////////////////////////////////////////////// MPK button 1
1832     MIDIdef.cc(\mpkButton24, {
1833         arg vel, nn, chan, src;
1834         /* button press for pad 5 on MPK */
1835         /* project soprano voice (pick up by Shure SM 58) into piano through speaker 9 (channel 8) */
1836         if (~countMpk1 == 0, {
1837             ~bar1SopSp9 = Ndef(\bar1SopSp9, ~bar1SopToSpeaker9);
1838             "on MPK cc 24, pad 5- 1".postln;
1839             ~countMpk1 = 1;
1840         }, {
1841             ~bar1SopSp9.release(1);
1842             "off MPK cc 24, pad 5- 1".postln;
1843             ~countMpk1 = 0;
1844         });
1845     }, 24, srcID: ~mpkID); // MPK pad 5 midi number 24
1846
1847     // MIDIdef for cc button 25
1848     MIDIdef.cc(\mpkButton25, {

```

```

1878 /* trigger sound; APC */
1879 ~midiMapApc = {
1880     ~countApc1 = 0;
1881     ~countApc2 = 0;
1882     ~countApc3 = 0;

```

```

1958 /////////////////////////////////////////////////// bar 1 /////////////////////////////////////////////////// button 1
1959 MIDIdef.noteOn(\apcButton56, {
1960     arg vel, nn, chan, src;
1961     /* output soprano voice to speaker 9 */
1962     if (~countApc1 == 0, {
1963         ~bar1SopSp9 = Ndef(\bar1SopSp9, ~bar1SopToSpeaker9);
1964         m.noteOn(0, 56, 01); //light on
1965         "on 56 button 1".postln;
1966         ~countApc1 = 1;
1967     }, {
1968         ~bar1SopSp9.release(1);
1969         m.noteOn(0, 56, 0); //light off
1970         "off 56 button 1".postln;
1971         ~countApc1 = 0;
1972     });
1973     }, 56, srcID: ~apcID);
1974
1975 /////////////////////////////////////////////////// bar 1 /////////////////////////////////////////////////// button 2
1976 MIDIdef.noteOn(\apcButton57, {

```

```

3410
3411      /* run mappings for controllers, and send on and off messages */
3412      // go APC mapping
3413      ~midiMapApc.C);
3414      // go MPK mapping
3415      //~midiMapMpk.C); // run mapping function for MPK mini
3416      // go foot pedal mapping
3417      //~midiMapFoot.C);
3418
3419      "midiMap done".postln;
3420

```

9. Run the code.

After you have setup hardware and modified the code according to your hardware setup, proceed to run the code.

Move cursor to the beginning of line 23. This will highlight lines 23 to 3425. Run the lines by pressing command + return. This will boot the Server and store written variables and functions for sound processes.

```

23 {
24 {
25     /* start up file */
26     Server.local.options.numOutputBusChannels = 11; // change number of input and output channels
27     Server.local.options.numInputBusChannels = 3;
28     Server.internal.options.numOutputBusChannels = 11;
29     Server.internal.options.numInputBusChannels = 3;
30     //Server.local.options.device = "MOTU 896mk3 Hybrid";
31     //Server.local.options.inDevice = "Scarlett Zi4 USB"; // other possible audio interfaces
32     Server.local.options.outDevice = "Built-in Output";
33     //Server.local.options.outDevice = "Orpheus (0979)";
34
35     {s.meter;}.defer; // make sure number of channels is correct
36     MIDIClient.init; // start midi

```

If SuperCollider is successfully booted, the parameters of Server at the lower right of the environment will show in green. If buffers are successfully allocated, functions and midiMaps are successfully stored, the post window will show the messages 'buffers done', 'func and def done' and 'midiMap done'.

```

*** Welcome to SuperCollider 3.9.3. *** For help press Cmd-D.
-> a Routine
MIDI Sources:
MIDI Destinations:
booting server 'localhost' on address: 127.0.0.1:57110
Found 0 LADSPA plugins
Number of Devices: 2
  0 : "Built-in Microph"
  1 : "Built-in Output"

"Built-in Microph" Input Device
Streams: 1
  0 channels 2

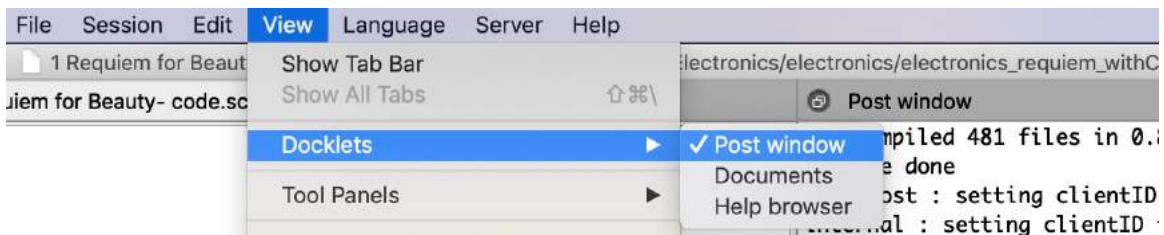
"Built-in Output" Output Device
Streams: 1
  0 channels 2

SC_AudioDriver: sample rate = 44100.000000, driver's block size = 512
SuperCollider 3 server ready.
Requested notification messages from server 'localhost'
localhost: server process's maxLogins (1) matches with my options.
localhost: keeping clientID (0) as confirmed by server process.
Shared memory server interface initialized
buffers
buffers done
func and def done
midiMap done
done!

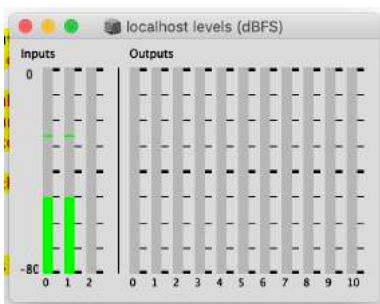
Interpreter: Active Server: 0.18% 0.21% 34u 5s 5g 113d 0.0d8

```

If you do not see the post window, go to View, select Docklets, and select Post window.



A meter should pop up unless the s.meter (line 35) wasn't run. If there is no meter, go to line 35 to see if the line was commented out accidentally, or trouble shoot lines 26 to 35 to see what the problem is. The number of the input and output on the meter corresponds to the numbers of the input and output channels that you indicate at lines 26-29.



```
32 Server.local.options.outDevice = "Built-in Output";
33 //Server.local.options.outDevice = "Orpheus (0979)";
34
35 {s.meter;}.defer; // make sure number of channels is correct
36 MIDIClient.init; // start midi
37 MIDIIn.connectAll; // connect all midi devices
38
39 s.waitForBoot { // go through before booting server
```

10. Test speakers, rehearse and perform the electronics part of the *Requiem*.

After you have successfully ran lines 23 to 3425, and the post window indicates the messages 'buffers done', 'func and def done', 'midiMap done' and 'done!', proceed to the speaker test. The instruction for speaker test is outlined in the next section. Practice the electronics part by following the written instructions on the score.

11. To close the script, click on the x at upper left of the tab (or command + w). A pop up window will appear asking if you would want to save the changes, click on 'Save' so you do not need to modify the code the next time you open the script. Click on the x at the upper left of the program window to close SuperCollider program (or command + q).

Tips

1. If you wish to stop a process before its end, press command + . (period).
2. If you are unable to run certain processes for unknown reason, recompile class library, go to Language and click on Recompile Class Library or press shift + command + l. After doing this, you will need to run lines 23 to 3425 again to store variables and functions.

Speaker test

1. Speaker test by noise and frequency sweep.

If you are using the 9 speakers and 2 subwoofers setup, and if you are using APC mini controller, you can playback the noise sound file by pressing buttons 65-73 where button 65 corresponds to speaker 1 and subwoofer 1, button 66 corresponds to speaker 2 and subwoofer 2, button 67 corresponds to speaker 3 and 67 to speaker 4 etc.

Pressing buttons 65 will playback noise sound file 'test-noise.wav' through speaker 1 and 'test-sub-noise.wav' through subwoofer 1. Pressing button 66 will playback noise sound file 'test-noise.wav' through speaker 2 and 'test-sub-noise.wav' through subwoofer 2.

You may test the speakers by frequency sweep. To do this, modify the global variables that store the noise sound files to the global variable that store the sweep sound files from lines 3238 to 3403. For example, to change sound file for speaker 1 test, uncomment line 3243 and comment out line 3244. Do this also for speakers 2-9. Alternately, you may simply modify the names of the global variables from ~bufTestNoise to ~bufTestSweep at line 3244. Do this also for speakers 2-9.

```
3237 ////////////////////////////////////////////////// sweep sp 1 ////////////////////////////////////////// button 65
3238 MIDIdef.noteOn(\apcButton64, {
3239   arg vel, nn, chan, src;
3240   if (~countApc65 == 0, {
3241     ~sweep1 = Ndef(\sweep1, ~sweep_sp1);
3242     ~sweep1sub = Ndef(\sweep1sub, ~sweep_sp1sub);
3243     //Ndef(\sweep1).set(\bufnum, ~bufTestSweep);
3244     Ndef(\sweep1).set(\bufnum, ~bufTestNoise);
3245     Ndef(\sweep1sub).set(\bufnum, ~bufTestSubNoise);
3246     m.noteOn(0, 64, 01); //light on
3247     "on 64 button 65 sweep speaker 1".postln;
3248     ~countApc65 = 1;
3249   }, {
3250     ~sweep1.release(1);
3251     ~sweep1sub.release(1);
3252     m.noteOn(0, 64, 0); //light off
3253     "off 64 button 65".postln;
3254     ~countApc65 = 0;
3255   });
3256 }, 64, srcID: ~apcID);
3257
```

If you wish to do frequency sweep through the subwoofers, change the global variables from ~bufTestSubNoise to ~bufTestSubSweep at lines 3425 and 3266.

```

3237 ////////////////////////////////////////////////// sweep sp 1 /////////////////////////////////// button 65
3238 MIDIdef.noteOn(\apcButton64, {
3239   arg vel, nn, chan, src;
3240   if (~countApc65 == 0, {
3241     ~sweep1 = Ndef(\sweep1, ~sweep_sp1);
3242     ~sweep1sub = Ndef(\sweep1sub, ~sweep_sp1sub);
3243     //Ndef(\sweep1).set(\bufnum, ~bufTestSweep);
3244     Ndef(\sweep1).set(\bufnum, ~bufTestNoise);
3245     Ndef(\sweep1sub).set(\bufnum, ~bufTestSubNoise);
3246     m.noteOn(0, 64, 01); //light on
3247     "on 64 button 65 sweep speaker 1".postln;
3248     ~countApc65 = 1;
3249   }, {
3250     ~sweep1.release(1);
3251     ~sweep1sub.release(1);
3252     m.noteOn(0, 64, 0); //light off
3253     "off 64 button 65".postln;
3254     ~countApc65 = 0;
3255   });
3256 }, 64, srcID: ~apcID);
3257

```

```

3258 ////////////////////////////////////////////////// sweep sp 2 /////////////////////////////////// button 66
3259 MIDIdef.noteOn(\apcButton65, {
3260   arg vel, nn, chan, src;
3261   if (~countApc66 == 0, {
3262     ~sweep2 = Ndef(\sweep2, ~sweep_sp2);
3263     ~sweep2sub = Ndef(\sweep2sub, ~sweep_sp2sub);
3264     //Ndef(\sweep2).set(\bufnum, ~bufTestSweep);
3265     Ndef(\sweep2).set(\bufnum, ~bufTestNoise);
3266     Ndef(\sweep2sub).set(\bufnum, ~bufTestSubNoise);
3267     m.noteOn(0, 65, 01); //light on
3268     "on 65 button 66 sweep speaker 2".postln;
3269     ~countApc66 = 1;
3270   }, {
3271     ~sweep2.release(1);
3272     ~sweep2sub.release(1);
3273     m.noteOn(0, 65, 0); //light off
3274     "off 65 button 66".postln;
3275     ~countApc66 = 0;
3276   });
3277 }, 65, srcID: ~apcID);
3278

```

You may substitute different sound files for speaker tests by changing the buffer directories from lines 104 to 110.

```

101 s.sync;
102 ~bufCueBar268 = Buffer.read(s, "/Users/shuyulin/Desktop/bar268.wav");
103 s.sync;
104 ~bufTestNoise = Buffer.read(s, "/Users/shuyulin/Desktop/test-noise.wav"); // noise for testing speakers
105 s.sync;
106 ~bufTestSubNoise = Buffer.read(s, "/Users/shuyulin/Desktop/test-sub-noise.wav");
107 s.sync;
108 ~bufTestSubSweep = Buffer.read(s, "/Users/shuyulin/Desktop/test-sub-sweep.wav"); // sweep for testing speakers
109 s.sync;
110 ~bufTestSweep = Buffer.read(s, "/Users/shuyulin/Desktop/test-sweep.wav");
111 s.sync;
112 /*
113 ~bufStab = Buffer.read(s, Platform.resourceDir + "/" + "sounds/a11wlk01.wav"); // default sound file for testing
114 s.sync;*/
115
116 "buffers done".postln;

```

Depending on the position of the speaker and the type (or brand) of speaker (if you are using various brands/models), noise and sweep may sound slightly different from each other.

2. Adjust volume accordingly.