

J74 Assistant

MIDI Assistant

Table of Contents

An introduction to the “J74 MIDI Assistant”	3
Installation and Setup	4
The user interface of the J74 MIDI Assistant plug-in	5
The Work-flow	6
Web AI Tool prompt engineering	8
Work-Flow Examples	11
Prompt Examples	17
Additional guidelines for AI prompting	50
Limitations	50
Licensing, Liability, Support	50

An introduction to the “J74 MIDI Assistant”

The “J74 MIDI Assistant” is a utility plug-in. It operates as a transparent device in Ableton Live and works as a *MIDI clipboard* for *COPY & PASTE* actions, to and from MIDI clips and external applications (such as web-based AI tools).

The tool is *bidirectional*:

- You can *paste* data *into* a MIDI clip from an external application (e.g. ask an AI tool for MIDI generation, like “...create a 5-2-1-1 progression in C Dorian...”, inserting the result in Ableton Live, directly as a MIDI clip).
- You can *copy* data *from* a MIDI clip in Ableton Live, let it be processed outside (e.g. by an AI tool) and then insert back the result with a *paste* action. For instance ask an AI to work on a part, based on something you already have (“...arrange a bassline based on this electric piano part...”), in this case giving the AI the part to analyze and work on, inserting back into the result produced by the AI, directly as a MIDI clip.

Note: MIDI data is exchanged through the “J74 MIDI Assistant” using a *JSON format*.

For both use-cases a set of prompt examples (and prompt design rules) are provided in this user manual.

Background

A generic *Web AI Tool* is like a “genie in a bottle” when it comes to music production: it knows practically everything (harmony, melody, rhythm, drums, styles, whatever) but cannot communicate with your DAW (Ableton Live). It has no direct access to your software. It cannot run code on your local machine. It cannot directly read or write (MIDI) files on your local machine. *It is confined within the boundaries of the web browser it runs on.*

A Web AI Tool can process your prompts via the web browser and can provide results through it, often in the form of an inline answer. But it has “copy & paste” capabilities of its own and *that* is exactly where the data link can be created. For *that*, the “J74 MIDI Assistant” utility can be the answer.

Through this utility, MIDI data can be copied *from* any MIDI clip in Ableton Live into a “piece of text” (technically speaking, a JSON dictionary). With a well-formatted prompt, a Web AI Tool can understand this data and process it. And when you have MIDI data generated (or processed) by a Web AI Tool, you can copy it from the Web AI Tool, as a JSON dictionary (again, a “piece of text”), and *paste* it back into Ableton Live, directly as a MIDI clip.

That’s it, really. A *MIDI clipboard*. It is that simple.

Side Note

I have been working for some time on a related (and much bigger) project, the “J74 Assistant Developers Tools”. This project takes AI integration to a different level (it deals with API integration, data structuring for AI and communication via OSC). For this project I have developed both a “data format” and a “data bridge” for exchanging (packing, unpacking, collecting, exporting) data, from and to Ableton Live, and let outside applications use it (e.g. let an AI Tool, capable of running code & reading and saving files, *co-work* on a Ableton Live project with you).

This data bridge (and the related data format) was initially meant only for use by this “API over OSC” project, but it then expanded, making direct COPY & PASTE actions of (JSON) text (data) possible.

That has essentially become the “J74 MIDI Assistant” package as you see it. A utility which *does not require any system integration* and *can be used with any AI Tool* in a web browser (Gemini, ChatGPT, Co-Pilot, whatever).

Tip: if you are interested on a different level of AI integration (while being brave enough to do some system integration) be free to check the “J74 Assistant Developers Tools” package from <https://www.fabriziopoce.com/>.

Installation and Setup

Installation, setup and use of the “J74 MIDI Assistant” is straight forward. *Download* the package zip from the link you received. Once the download has completed, *unzip* the archive. Once unzipped, you will see the following:

- The *J74 MIDI Assistant*, a Max for Live plug-in, named: “Assistant_MIDI_J74.amxd”
- A set of *PDF documents*, including this user manual.

You can save (copy) the plug-in (and its documentation) anywhere on your system, in any location you like. When you need it in Ableton Live, just drag and drop it from the Ableton Live browser onto the selected track. You can load it and leave it there, as you do not need to configure anything on it (its default will normally work).

Tip: you can include the location of the folder where the “Assistant_MIDI_J74.amxd” is located directly in the Ableton Live browser, go to the “Places” section of the Ableton Live browser and select [+ Add Folder...]

Additional information:

- Requirements: Ableton Live 11 or later, with a Max for Live license (as in Ableton Live “Suite”).
- Limitation: the plug-in supports a maximum of 100 notes per action (both COPY & PASTE).
- Limitation: the plug-in only operates on session view MIDI clips.

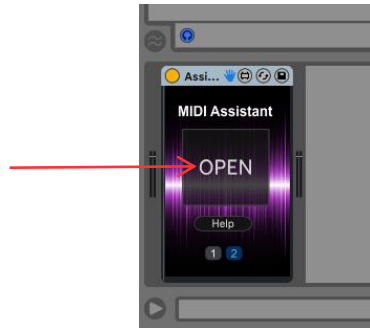
Note: The “J74 MIDI Assistant” plug-in is a transparent audio device. It needs to be present in the project, on any track of choice. The plug-in is completely transparent to audio and does not affect audio passing through. For information, the audio plug-in format (rather than a MIDI device format) has been chosen so that the plug-in can be set on any track. The device does not have to process any MIDI in real-time anyway (it can retrieve MIDI clip data from anywhere, as it follows the selection of the MIDI clip you do on the session view).

No other instructions apply. You can immediately use the device.

Let’s now familiarize ourselves with the work-flow and the user interface and settings of the device. After that we will go through the task of how to create prompts for a Web AI Tool.

The user interface of the J74 MIDI Assistant plug-in

Once dropped in the live set (anywhere is fine), the “Assistant_MIDI_J74.amxd” plug-in will appear as a small “docked” device, as in the picture below:



To see its complete user interface (something you will need to do, if you want to use the COPY & PASTE actions), you need to hit the “OPEN” button. If you find the user interface too small, select “2” on the [1|2] control to magnify the user interface.

IMPORTANT: The tool uses a floating window so that you can use it wherever your view in Ableton Live is.

Below a summary of the controls of the plug-in (once the user interface is open).

[Load from JSON Midi] Allows you to import MIDI content to a clip (selected by clicking on the clip slot), from a JSON Midi file formatted according to the Assistant MIDI JSON file format.

[PASTE] Select the target clip slot in the project (click on the target clip slot of a MIDI or instrument track). The TRACK & CLIP indicator will show your selection. Paste in the [PASTE] field the content of a JSON dictionary for MIDI clip creation (e.g. generated by a Web AI Tool). Press [S] to send the content to the MIDI clip. If you need, you can clear the content pressing [D].

{TRACK & CLIP indicator} shows your current clip slot selection. This is where [COPY], [PASTE], [Load] or [Save] operations will be applied to.

[Auto Clear] If enabled (ON), as on defaults, it automatically clears the PASTE content after it is triggered. If the desired effect is to paste the MIDI JSON content on multiple slots, set this to (OFF).

[COPY] Select the clip slot in the project you wish to copy. The TRACK & CLIP indicator will show your selection. Press [G] to get the content of the clip populated in the [COPY] field. Copy then content (a JSON raw data dictionary for the MIDI clip) to get it into the system clipboard, for processing by an external application, such as a Web AI Tool. If you need, you can clear the content pressing [D].

{TRACK & CLIP indicator} shows your current clip slot selection. This is where [COPY], [PASTE], [Load] or [Save] operations will be applied to.

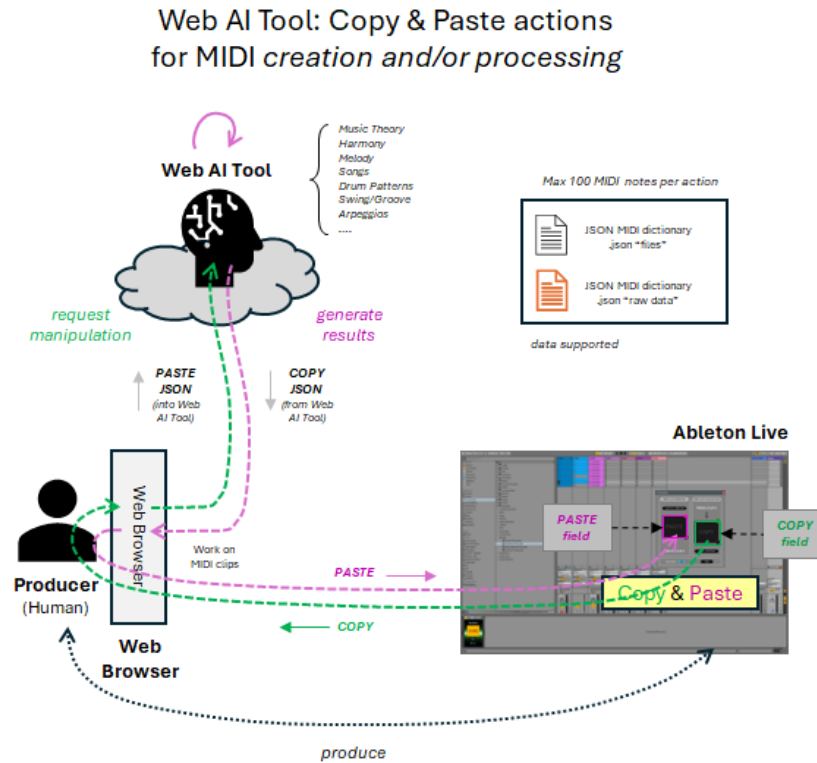
[Help] Opens the help section.

[Step-0] Provides the JSON format definition (content of “Step-0”).

[Save to JSON Midi] Allows you to export the MIDI content of a MIDI clip (selected by clicking on the clip slot) as a JSON Midi file, formatted according to the Assistant MIDI JSON file format.

The Work-flow

The “J74 MIDI Assistant” device is a COPY&PASTE utility for MIDI data in MIDI clips. It is essentially a clipboard for MIDI data. With it you can select a MIDI clip and get the MIDI notes as a **COPY** field. When you have MIDI data coming from an outside application (like a Web AI Tool) you can then use its **PASTE** field to push the data into a MIDI clip in Ableton Live. Below a work-flow diagram which shows how you use the utility in practice.



The work-flow:

- When you have a MIDI clip to *process*, you select the MIDI clip (by clicking on it in the session view) and then head to the J74 MIDI Assistant and prepare a **[COPY]** field using the **[G]** (green, GET) button (so you can have the JSON dictionary ready for copy action and then paste on a Web AI Tool to process).
- When you have MIDI data *processed* or *generated* by a Web AI Tool, you can then copy it from the Web AI Tool (as text) and head back to Ableton Live, where you first select the MIDI clip / slot, as the place you want the MIDI clip to be created (just click on it in the Session view), and finally use the **[PASTE]** field on the “J74 MIDI Assistant” and its **[S]** (magenta button), which creates a new MIDI clip with this data.

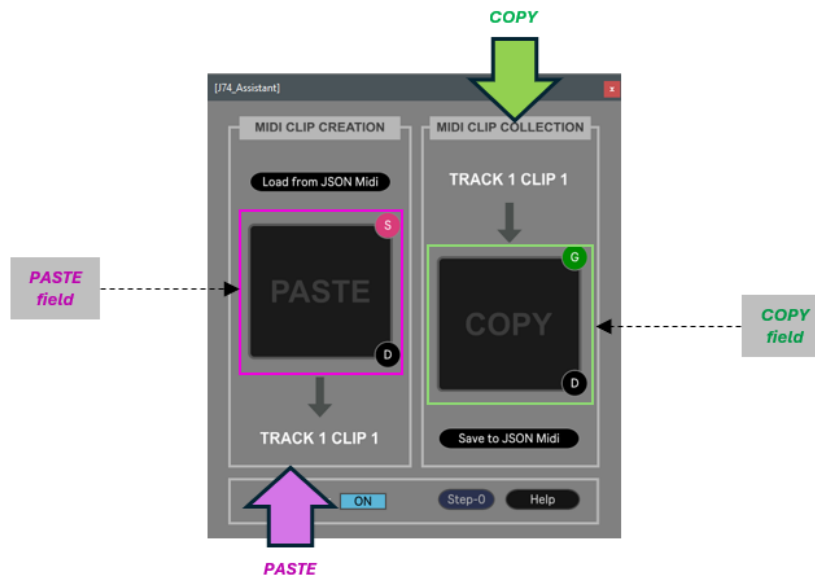
Remark: when in this user-manual you see **[COPY]**, the action refers to the copy action of the J74 MIDI Assistant **[COPY]** field; when you see **[PASTE]** the action refers to the paste action of the J74 MIDI Assistant **[PASTE]** field.

Note: If you want to process a MIDI clip, after the **[COPY]** action, you will proceed on the external application with a normal paste and process it there (e.g. in a Web AI Tool). Then take the result (copy it, in the same format) and use the utility **[PASTE]** action in Ableton Live. This happens quite naturally when using the prompt structures described later. So don't worry now, you will see it in action.

Note: If you just need to generate MIDI on a Web AI Tool (e.g. asking it for some kind of musical progression or melody or drum line, you will just copy from the Web AI Tool) and use the utility **[PASTE]** action in Ableton Live.

Below is a visual representation of the work-flow and details on its working:

- **To process a MIDI clip externally (e.g. with a Web AI Tool), click the clip slot to select it.**
- *The {TRACK & CLIP indicator} shows your selection.*
- **Press the [G] button to “get” the clip MIDI content from the selected clip as JSON data.**
- *The [COPY] field is populated with data.*
- **Click on the [COPY] field and “copy” all the content to the clipboard.**
- *Quick shortcut on a Windows system : [CTRL+A] to select all, [CTRL+C] to copy to the clipboard*
- *Quick shortcut on a Mac OS system : [CMD+A] to select all, [CMD+C] to copy to the clipboard*
- **On the external tool (e.g. a Web AI Tool), paste the content for processing.**
- *Quick shortcut on a Windows system : [CTRL+V] to paste from the clipboard*
- *Quick shortcut on a Mac OS system : [CMD+V] to paste from the clipboard*



- **From the external tool (e.g. a Web AI Tool), copy the generated JSON content.**
- *Quick shortcut on a Windows system : [CTRL+A] to select all, [CTRL+C] to copy to the clipboard*
- *Quick shortcut on a Mac OS system: [CMD+A] to select all, [CMD+C] to copy to the clipboard*
- **Select (click on) a clip slot in your project to define where the MIDI clip needs to be created.**
- *The {TRACK & CLIP indicator} shows your selection.*
- **Click on the [PASTE] field and “paste” all the content from the clipboard.**
- *Quick shortcut on a Windows system : [CTRL+V] to paste from the clipboard*
- *Quick shortcut on a Mac OS system : [CMD+V] to paste from the clipboard*
- **Press the [S] button to “send” the JSON data content to the MIDI clip.**

About the JSON data format

The JSON format is a broadly adopted format for data, which uses structured (but plain) text. Any Web AI Tool knows how to unpack it and, if it has data to fit in the format, to pack it. For a user not interested in the details of formatting and code, just know this is “MIDI data expressed with a certain format” and that this format can be copied & pasted as text, exactly as we will do with this workflow and see in the rest of this document.

Side-Note: the same JSON MIDI dictionary format is used by both the Max for Live “J74 MIDI Assistant” plug-in (this package) and the “J74 Assistant Developers Tools” package (a separate project in the J74 family of tools). Therefore, if you use both, MIDI data can be freely *interchanged* across the two environments.

Web AI Tool prompt engineering

To perform MIDI data exchange (for manipulation or generation of MIDI clips) with a Web AI Tool you need to follow certain rules for making prompts. That is: you need to tell things to a Web AI Tools in a “certain way”. It is an easy format, don’t worry, but you need to use it precisely (the following will guide you).

The next sections will define how to do prompts:

- **An initial [Step 0] prompt for the Web AI Tool (the JSON format agreement):** When you open a Web AI Tool, the AI does not have context. This prompt gives the AI information on the format and needs to be given once per session, at the start (or before you ask for any generative or manipulative action). It essentially tells the AI that a JSON format is used for MIDI notes and how it looks like. A [Step-0] little window can be opened from the J74 MIDI Assistant user interface, so you can easily copy this prompt.
- **Procedure [A] for the generation of MIDI clips:** This is a simple “3-step” process (prompt-copy-create) which you use when you need a Web AI Tool to generate some MIDI notes for your specific goal. You start [1] by giving to the AI a single prompt, describing in natural (musical) language what you want; then [2] you copy the (JSON data) produced by the AI from the Web AI Tool; finally [3] you head over to Ableton Live, where you select the clip slot (by clicking on it) and on the J74 MIDI Assistant use the [PASTE] field (and the [S] button) to create the MIDI clip result.
- **Procedure [B] for the manipulation of MIDI clips:** This is a “5-step” process (collect-copy-prompt-copy-create). You start with [1] selecting a MIDI clip in Ableton Live you wish to process by clicking on it; then [2] you populate the [COPY] field in the J74 MIDI Assistant (with the [G], green “GET” button) and copy the generated text; after that [3] you head over to the Web AI Tool and issue a prompt requesting you action on this data (you will include the data you gathered in Ableton Live as part of the prompt); next [4] once the AI has processed the data, you can copy it from the Web AI Tool; finally [5] you head back to Ableton Live, where you select the clip slot (by clicking on it) and on the J74 MIDI Assistant use the [PASTE] field (and [S] button) to create the MIDI clip result.

Let’s go through them.

“Step 0” - Initial prompt for the Web AI Tool (JSON format agreement)

A Web AI Tool is flexible and can automatically get a formal definition of a data set out of an example. This is exactly what the “Step 0”, the initial prompt, does with the Web AI Tool: it gives the AI an example and agrees with the AI that this (JSON) data format is to be used for musical (MIDI) note processing.

It is a simple prompt; you can just copy and paste from here or use the [Step-0] little window you can open from the MIDI Assistant user interface. It does not need to be updated, just give it once per session, before other actions.

```
# Step 0: one-time prompt (only once per session)
# Description: Prompt which gives the AI the definition of the JSON format for musical (MIDI) notes
# Prompt structure:

Consider from now on the JSON format for musical notes (with a maximum of 100 notes per JSON file): { "notes" :
[ { "note_id" : 0, "pitch" : 93, "start_time" : 1.5, "duration" : 0.5, "velocity" : 100, "mute" : 0, "probability" : 0.2,
"velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 1, "pitch" : 93, "start_time" : 4, "duration" : 0.5,
"velocity" : 100, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, ... , { "note_id" :
99, "pitch" : 103, "start_time" : 14.5, "duration" : 0.5, "velocity" : 100, "mute" : 0, "probability" : 0.2,
"velocity_deviation" : 0, "release_velocity" : 64 } ] }
```

Next, we will formally define the prompt structure for procedures [A] and [B]. These are formal definitions for understanding the structure of prompting. As you will see from the examples (given next) it is just a matter of becoming familiar with their structure, after that it will become second nature.

Procedure [A]: Generate MIDI Clips | 3-step (prompt-copy-create) process

Step	Description	Content / Details
Step 0:	Be sure that the one-time "Step 0" prompt has been initially given to the Web AI Tool.	[See Step-0 definition]
Step 1:	Prompt the Web AI Tool. Gives the AI musical instructions, in one prompt.	# Prompt structure: # [mandatory] 1: <i>"using the specified JSON format (use the same format, do not add any new field)"</i> 2: <i>"create a JSON file with the notes for " <some musical instructions></i> 3: <i>"using a reference pitch" <select reference pitch></i> (e.g. "using a reference pitch C3") # [optional] 4: <i><provide instructions for probability of notes, if desired></i> 5: <i><provide instructions for velocity deviation of notes, if desired></i> 6: <i>"Do not add comments in the JSON file"</i> 7: <i>"Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats"</i> 8: <i>"Make the JSON file downloadable and name it generated.json"</i>
Step 2:	Copy the data generated by the Web AI tool	[Copy from the Web AI Tool copy&paste field] [Quick shortcut on a Windows system : [CTRL+A] to select all, [CTRL+C] to copy to the clipboard] [Quick shortcut on a Mac OS system: [CMD+A] to select all, [CMD+C] to copy to the clipboard]
Step 3:	Run the [PASTE] + [S] action on the J74 MIDI Assistant (paste of the generated data or use of the file).	[Select the MIDI clip slot in Ableton Live (by clicking on it)] [Quick shortcut on a Windows system : [CTRL+V] to paste from the clipboard] [Quick shortcut on a Mac OS system : [CMD+V] to paste from the clipboard] [On the J74 MIDI Assistant use the [PASTE] field + [S] action]

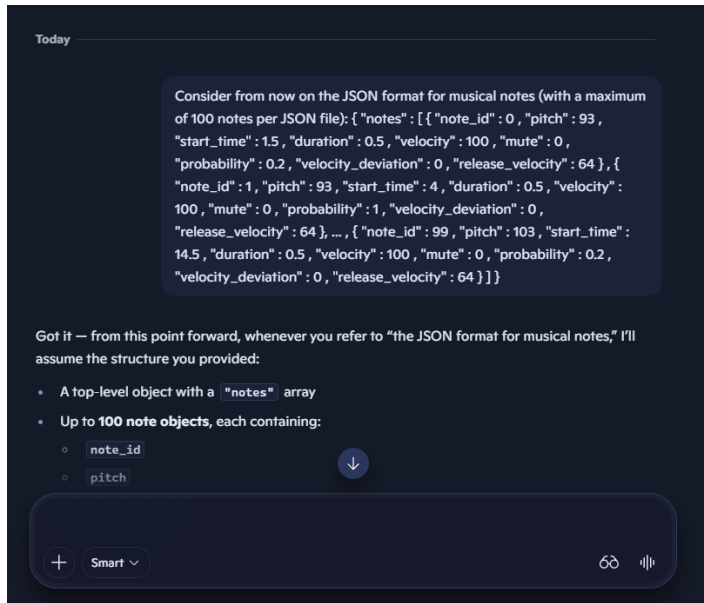
Procedure [B]: Manipulate MIDI Clips | 5-step (collect-copy-prompt-copy-create) process

Step	Description	Content / Details
Step 0:	Be sure that the one-time "Step 0" prompt has been initially given to the Web AI Tool.	[See Step-0 definition]
Step 1:	Collect an existing MIDI clip from Ableton Live with a [COPY] + [G] action on the J74 MIDI Assistant	[Select the MIDI clip slot in Ableton Live (by clicking on it)] [On the J74 MIDI Assistant use the [COPY] field + [G] action]
Step 2:	Copy the data generated by the J74 MIDI Assistant in the [COPY] field	[On the J74 MIDI Assistant use [COPY] field select all text and copy it] [Quick shortcut on a Windows system : [CTRL+A] to select all, [CTRL+C] to copy to the clipboard] [Quick shortcut on a Mac OS system: [CMD+A] to select all, [CMD+C] to copy to the clipboard]
Step 3:	Prompt the Web AI Tool. Gives the AI musical instructions, in one prompt.	# Prompt structure: # [mandatory] 1: <i><formulate instruction for the change> (e.g. "transpose of all pitches for two semitones up")</i> 2: <i>"given the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field):"</i> 3: <i><PASTE_HERE_THE_COLLECTED_JSON_DATA></i> # [optional] 4: <i><provide instructions for probability of notes, if desired></i> 5: <i><provide instructions for velocity deviation of notes, if desired></i> 6: <i>"Do not add comments in the JSON file"</i> 7: <i>"Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats"</i> 8: <i>"Make the JSON file downloadable and name it generated.json"</i>
Step 4:	Copy the data generated by the Web AI tool	[Copy from the Web AI Tool copy&paste field] [Quick shortcut on a Windows system : [CTRL+A] to select all, [CTRL+C] to copy to the clipboard] [Quick shortcut on a Mac OS system: [CMD+A] to select all, [CMD+C] to copy to the clipboard]
Step 5:	Run the [PASTE] action on the J74 MIDI Assistant (paste of the generated data or use of the file).	[Select the MIDI clip slot in Ableton Live (by clicking on it)] [Quick shortcut on a Windows system : [CTRL+V] to paste from the clipboard] [Quick shortcut on a Mac OS system : [CMD+V] to paste from the clipboard] [On the J74 MIDI Assistant use the [PASTE] field + [S] action]

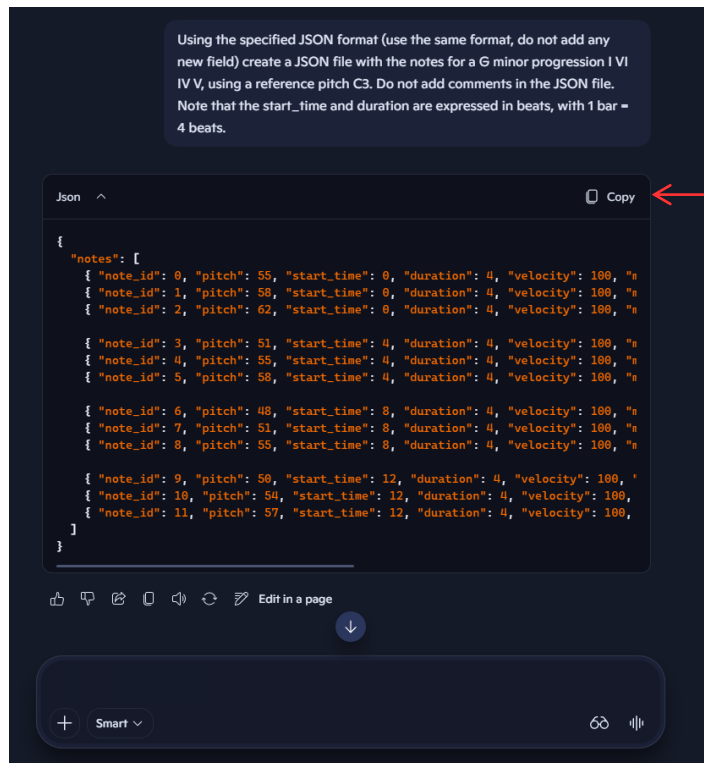
Work-Flow Examples

Examples at this point are better than any more explanation. The first one is a simple work-flow example with a set of screenshots, just to see it done. It is for **procedure [A]**, *generating MIDI data in a Web AI Tool*:

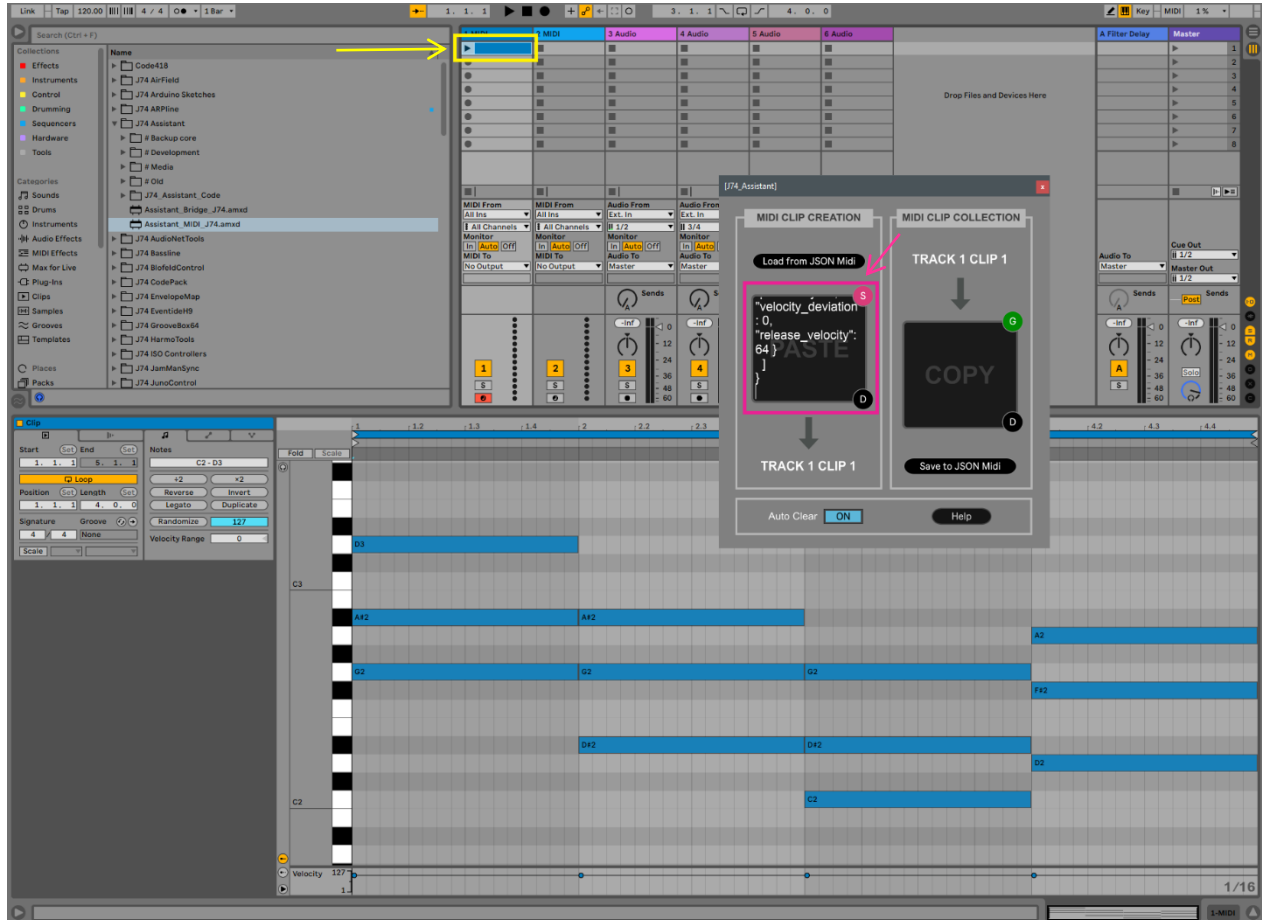
[Below “Step-0”, the initial prompt for the session, just giving the AI context and formats]



[Next, see a “real” prompt as of Procedure “A” and the reply from the AI (with a copy field)]



[Finally, back to Ableton Live, **select the clip by clicking on it** and then on the “J74 MIDI Assistant” use [PASTE] + [S]]

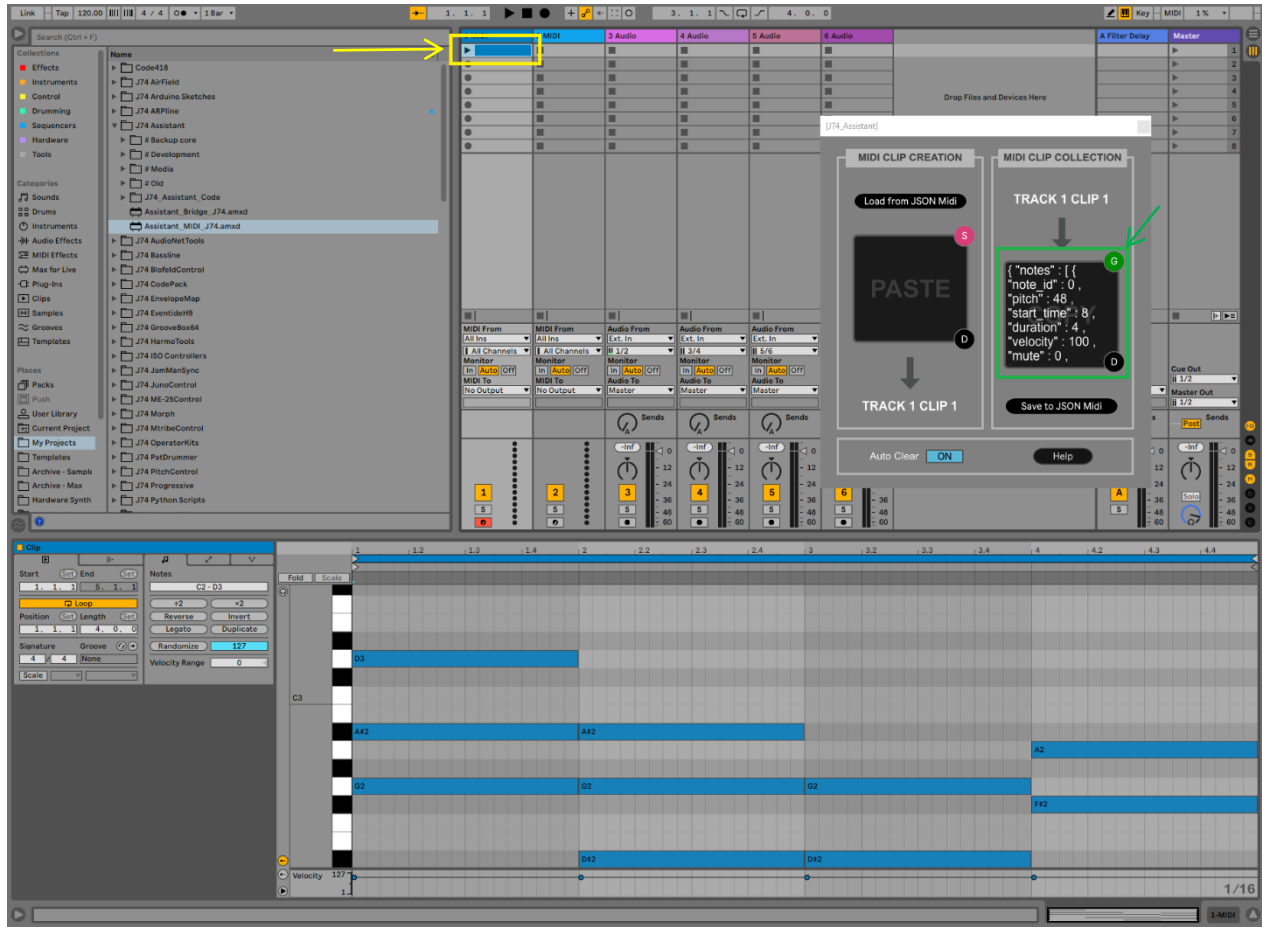


This simple example shows how quick and easy the process is.

The AI created (from scratch) a MIDI progression based on the input given in the prompt, in plain common language, and the result was provided by the AI exactly in the right format, ready to be copied. The [PASTE] action in the “J74 MIDI Assistant” created a MIDI clip immediately.

The next work-flow example is for **procedure [B]**, *manipulating MIDI data with a Web AI Tool*:

[With "Step-0" already done, we select the origin clip in Ableton Live and use the [COPY] on the J74 MIDI Assistant]



[After the copy of the MIDI data we provide the AI with the prompt asking to arpeggiate this progression]



Tip: it might be handy to use a text editor to prepare the above prompt; write the instruction part first and then paste the JSON data collected in the [COPY] stage, at the end. Once done, copy the entire prompt and paste it into the web browser (AI tool). This might be easier than working it all out in sections, directly on the web browser.

[The AI processes our prompt (with the data in it) and provides its answer, again with a *copy field*]

Here is the arpeggiated version of your progression, using:

- **1/8-bar steps = 0.5 beats per note**
- **Each bar = 4 beats → 8 arpeggio notes per chord**
- Pattern per bar (repeating): **root → third → fifth → third → root → third → fifth → third**
- Using only the pitches extracted from your JSON (no new pitches invented)

Chord sets (from your input):

- **Bar 0 (start 0):** 55, 58, 62
- **Bar 1 (start 4):** 51, 55, 58
- **Bar 2 (start 8):** 48, 51, 55
- **Bar 3 (start 12):** 50, 54, 57

All durations = **0.5 beats**.
All velocities = **100**, probability = **1**, mute = **0**, etc.

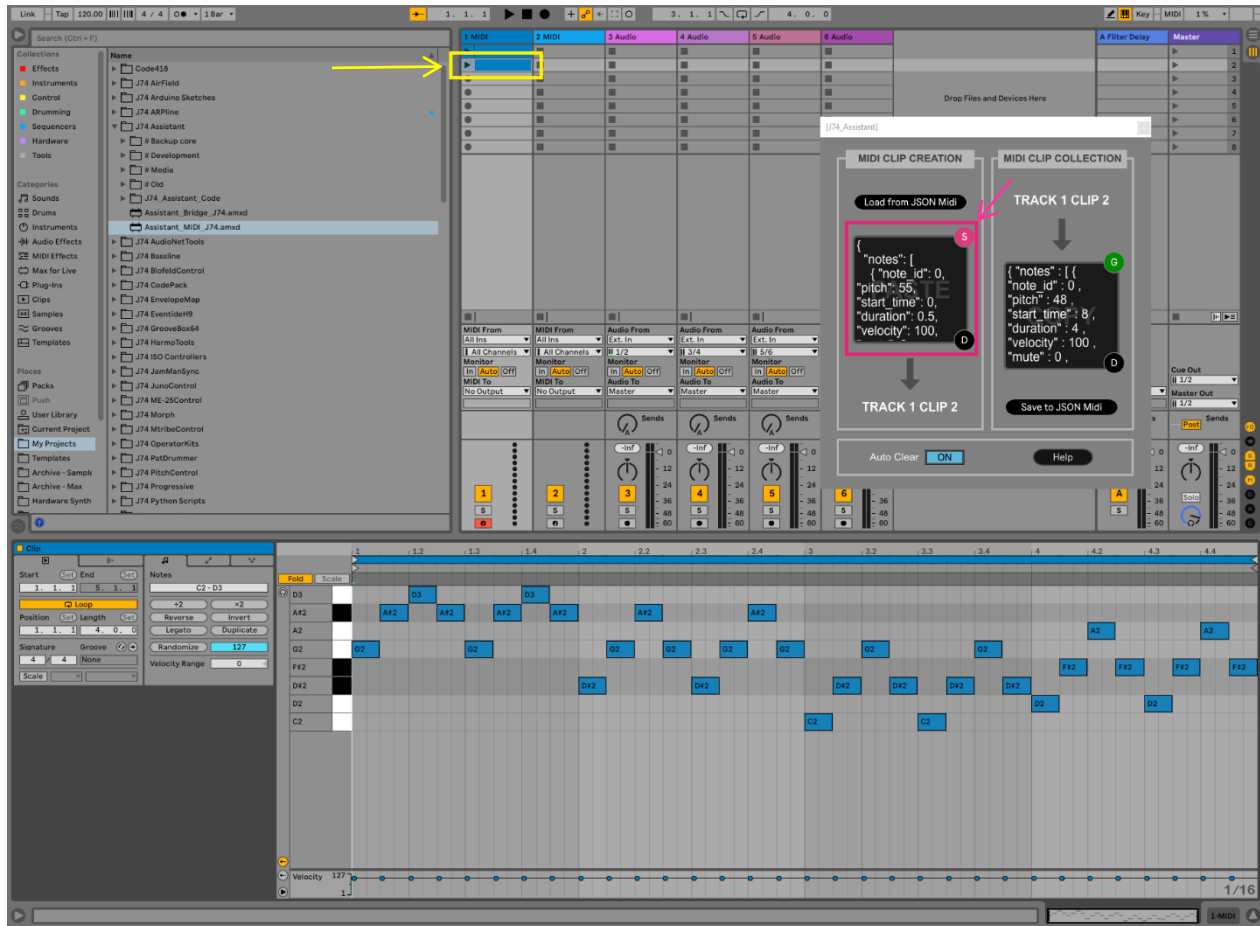
```
Json ^ Copy
```

```
{
  "notes": [
    { "note_id": 0, "pitch": 55, "start_time": 0, "duration": 0.5, "velocity": 100,
    { "note_id": 1, "pitch": 58, "start_time": 0.5, "duration": 0.5, "velocity": 100,
    { "note_id": 2, "pitch": 62, "start_time": 1, "duration": 0.5, "velocity": 100,
    { "note_id": 3, "pitch": 58, "start_time": 1.5, "duration": 0.5, "velocity": 100,
    { "note_id": 4, "pitch": 55, "start_time": 2, "duration": 0.5, "velocity": 100,
    { "note_id": 5, "pitch": 58, "start_time": 2.5, "duration": 0.5, "velocity": 100,
    { "note_id": 6, "pitch": 62, "start_time": 3, "duration": 0.5, "velocity": 100,
    { "note_id": 7, "pitch": 58, "start_time": 3.5, "duration": 0.5, "velocity": 100,

    { "note_id": 8, "pitch": 51, "start_time": 4, "duration": 0.5, "velocity": 100,
    { "note_id": 9, "pitch": 55, "start_time": 4.5, "duration": 0.5, "velocity": 100,
    { "note_id": 10, "pitch": 58, "start_time": 5, "duration": 0.5, "velocity": 100,
    { "note_id": 11, "pitch": 55, "start_time": 5.5, "duration": 0.5, "velocity": 100,
    { "note_id": 12, "pitch": 51, "start_time": 6, "duration": 0.5, "velocity": 100,
    { "note_id": 13, "pitch": 55, "start_time": 6.5, "duration": 0.5, "velocity": 100,
    { "note_id": 14, "pitch": 58, "start_time": 7, "duration": 0.5, "velocity": 100,
    { "note_id": 15, "pitch": 55, "start_time": 7.5, "duration": 0.5, "velocity": 100,
```

+ Smart 60

[Back to Ableton Live, we select an empty slot by clicking on it and on the "J74 MIDI Assistant" use [PASTE] + [S]]



It should be clear now how the procedures [A] and [B] work and the related work-flows.

Next, we will look at other examples, this time providing the actual *text* fields for prompts and for data (so that you can test it by yourself, using these prompts).

Try them by yourself.

Prompt Examples

The following examples show you how to perform the procedures we described. You can replicate exactly these prompts by yourself (just copy them exactly) and learn how to do it and to verify that everything works as expected.

Important: in all cases be sure that the one-time "Step 0" prompt has been initially given to the Web AI Tool.

In the following I have recorded example for these processes using both Gemini (web version, <https://gemini.google.com/>) and ChatGPT (web version, <https://chatgpt.com/>). You will see how both tools reacted to the same prompts. They are extremely close.

Of course, be free and use the AI tool you prefer. The important thing is to verify if the AI understands the context and the data format (you can see it immediately in the "Step-0" reply). As far as I have seen so far, every AI I worked on did fine with this process.

Each example coming next will be given with the following structure:

Prompt:

```
> [Some Prompt Text]
```

Result:

```
JSON data (ready for copy & paste)
```

Examples [A] of create and generate (3-step prompt-copy-create process) using Web Gemini:**# Example/C1: simple progression [Gemini]****Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file with the notes for a G minor progression I VI IV V, using a reference pitch C3. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 43, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 46, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 50, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 3, "pitch": 39, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 4, "pitch": 43, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 5, "pitch": 46, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 6, "pitch": 36, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 7, "pitch": 39, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 8, "pitch": 43, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 9, "pitch": 38, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 10, "pitch": 41, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 11, "pitch": 45, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 }
  ]
}
```

Example/C2: song example from AI knowledge [Gemini]**Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file for the chord progressions in the song "House of the Rising Sun". Add also the notes for the melody one octave higher. Reference pitch is C3. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 45, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 48, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 52, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 3, "pitch": 48, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 4, "pitch": 52, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 5, "pitch": 55, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 6, "pitch": 50, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 7, "pitch": 54, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 8, "pitch": 57, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 9, "pitch": 41, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
      "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 10, "pitch": 45, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 11, "pitch": 48, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 12, "pitch": 45, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 13, "pitch": 48, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 14, "pitch": 52, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 15, "pitch": 48, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 16, "pitch": 52, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 17, "pitch": 55, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 18, "pitch": 40, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 19, "pitch": 44, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
  ]
}
```

```
{ "note_id": 20, "pitch": 47, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 21, "pitch": 40, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 22, "pitch": 44, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 23, "pitch": 47, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 24, "pitch": 69, "start_time": 0.5, "duration": 1, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 25, "pitch": 72, "start_time": 2, "duration": 1, "velocity": 95, "mute": 0, "probability": 1,
"velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 26, "pitch": 74, "start_time": 4.5, "duration": 1, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 27, "pitch": 76, "start_time": 6, "duration": 2, "velocity": 95, "mute": 0, "probability": 1,
"velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 28, "pitch": 74, "start_time": 8.5, "duration": 1, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 29, "pitch": 72, "start_time": 10, "duration": 1, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 30, "pitch": 69, "start_time": 12, "duration": 4, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 31, "pitch": 69, "start_time": 16.5, "duration": 1, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 32, "pitch": 72, "start_time": 18, "duration": 1, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 33, "pitch": 71, "start_time": 24, "duration": 4, "velocity": 95, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 }
]
}
```

Example/C3: arpeggio of a progression (with probability logic) [Gemini]**Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file with the notes for a G minor progression I VI IV V with reference pitch C3. Arpeggiate the notes every 1/8th of a bar (you can repeat notes in the chords). Probability of downbeat notes is always 1. Probability of other notes is 0.33. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 43, "start_time": 0.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 46, "start_time": 0.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 50, "start_time": 1.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 3, "pitch": 43, "start_time": 1.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 4, "pitch": 46, "start_time": 2.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 5, "pitch": 50, "start_time": 2.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 6, "pitch": 43, "start_time": 3.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 7, "pitch": 46, "start_time": 3.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 8, "pitch": 39, "start_time": 4.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 9, "pitch": 43, "start_time": 4.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 10, "pitch": 46, "start_time": 5.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 11, "pitch": 39, "start_time": 5.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 12, "pitch": 43, "start_time": 6.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 13, "pitch": 46, "start_time": 6.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 14, "pitch": 39, "start_time": 7.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 15, "pitch": 43, "start_time": 7.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 16, "pitch": 36, "start_time": 8.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 17, "pitch": 39, "start_time": 8.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 18, "pitch": 43, "start_time": 9.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 19, "pitch": 36, "start_time": 9.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  ]
}
```

```
{ "note_id": 20, "pitch": 39, "start_time": 10.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 21, "pitch": 43, "start_time": 10.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 22, "pitch": 36, "start_time": 11.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 23, "pitch": 39, "start_time": 11.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 24, "pitch": 38, "start_time": 12.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 25, "pitch": 41, "start_time": 12.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 26, "pitch": 45, "start_time": 13.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 27, "pitch": 38, "start_time": 13.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 28, "pitch": 41, "start_time": 14.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 29, "pitch": 45, "start_time": 14.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 30, "pitch": 38, "start_time": 15.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 31, "pitch": 41, "start_time": 15.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": 0, "release_velocity": 64 }
]
}
```

Example/C4: arpeggio of a progression (with probability and velocity logic) [Gemini]**Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file with the notes for a G minor progression I VI IV V with reference pitch C3. Arpeggiate the notes every 1/8th of a bar (you can repeat notes in the chords). Probability of downbeat notes is always 1. Probability of other notes is 0.33. Velocity deviation for downbeat notes is 0. Velocity deviation for other notes is -50. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 43, "start_time": 0.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 46, "start_time": 0.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 2, "pitch": 50, "start_time": 1.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 3, "pitch": 43, "start_time": 1.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 4, "pitch": 46, "start_time": 2.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 5, "pitch": 50, "start_time": 2.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 6, "pitch": 43, "start_time": 3.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 7, "pitch": 46, "start_time": 3.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 8, "pitch": 39, "start_time": 4.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 9, "pitch": 43, "start_time": 4.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 10, "pitch": 46, "start_time": 5.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 11, "pitch": 39, "start_time": 5.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 12, "pitch": 43, "start_time": 6.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 13, "pitch": 46, "start_time": 6.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 14, "pitch": 39, "start_time": 7.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 15, "pitch": 43, "start_time": 7.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 16, "pitch": 36, "start_time": 8.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 17, "pitch": 39, "start_time": 8.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 18, "pitch": 43, "start_time": 9.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
    { "note_id": 19, "pitch": 36, "start_time": 9.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 }
  ]
}
```

```
{ "note_id": 20, "pitch": 39, "start_time": 10.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 21, "pitch": 43, "start_time": 10.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 22, "pitch": 36, "start_time": 11.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 23, "pitch": 39, "start_time": 11.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 24, "pitch": 38, "start_time": 12.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 25, "pitch": 41, "start_time": 12.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 26, "pitch": 45, "start_time": 13.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 27, "pitch": 38, "start_time": 13.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 28, "pitch": 41, "start_time": 14.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 29, "pitch": 45, "start_time": 14.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 30, "pitch": 38, "start_time": 15.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 },
{ "note_id": 31, "pitch": 41, "start_time": 15.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 0.33, "velocity_deviation": -50, "release_velocity": 64 }
]
}
```

Examples [A] of create and generate (3-step prompt-copy-create process) using ChatGPT:**# Example/C1: simple progression [ChatGPT]****Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file with the notes for a G minor progression I VI IV V, using a reference pitch C3. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{ "notes" : [
{ "note_id" : 0 , "pitch" : 55 , "start_time" : 0 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 1 , "pitch" : 58 , "start_time" : 0 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 2 , "pitch" : 62 , "start_time" : 0 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 3 , "pitch" : 63 , "start_time" : 4 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 4 , "pitch" : 67 , "start_time" : 4 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 5 , "pitch" : 70 , "start_time" : 4 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 6 , "pitch" : 60 , "start_time" : 8 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 7 , "pitch" : 63 , "start_time" : 8 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 8 , "pitch" : 67 , "start_time" : 8 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 9 , "pitch" : 62 , "start_time" : 12 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 10 , "pitch" : 66 , "start_time" : 12 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 11 , "pitch" : 69 , "start_time" : 12 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 }
] }
```

Example/C2: song example from AI knowledge [ChatGPT]**Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file for the chord progressions in the song "House of the Rising Sun". Add also the notes for the melody one octave higher. Reference pitch is C3. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{ "notes" : [
{ "note_id" : 0 , "pitch" : 45 , "start_time" : 0 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 1 , "pitch" : 48 , "start_time" : 0 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 2 , "pitch" : 52 , "start_time" : 0 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 3 , "pitch" : 48 , "start_time" : 4 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 4 , "pitch" : 52 , "start_time" : 4 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 5 , "pitch" : 55 , "start_time" : 4 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 6 , "pitch" : 50 , "start_time" : 8 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 7 , "pitch" : 54 , "start_time" : 8 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 8 , "pitch" : 57 , "start_time" : 8 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 9 , "pitch" : 53 , "start_time" : 12 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 10 , "pitch" : 57 , "start_time" : 12 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 11 , "pitch" : 60 , "start_time" : 12 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 12 , "pitch" : 45 , "start_time" : 16 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 13 , "pitch" : 48 , "start_time" : 16 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 14 , "pitch" : 52 , "start_time" : 16 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 15 , "pitch" : 52 , "start_time" : 20 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 16 , "pitch" : 56 , "start_time" : 20 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 17 , "pitch" : 59 , "start_time" : 20 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
```

```
{ "note_id" : 18 , "pitch" : 45 , "start_time" : 24 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 19 , "pitch" : 48 , "start_time" : 24 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 20 , "pitch" : 52 , "start_time" : 24 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 21 , "pitch" : 52 , "start_time" : 28 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 22 , "pitch" : 56 , "start_time" : 28 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 23 , "pitch" : 59 , "start_time" : 28 , "duration" : 4 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },

{ "note_id" : 24 , "pitch" : 69 , "start_time" : 0 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 25 , "pitch" : 72 , "start_time" : 2 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 26 , "pitch" : 74 , "start_time" : 4 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 27 , "pitch" : 76 , "start_time" : 6 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 28 , "pitch" : 77 , "start_time" : 8 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 29 , "pitch" : 76 , "start_time" : 10 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 30 , "pitch" : 74 , "start_time" : 12 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 31 , "pitch" : 72 , "start_time" : 14 , "duration" : 2 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 }
] }
```

Example/C3: arpeggio of a progression (with probability logic) [ChatGPT]**Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file with the notes for a G minor progression I VI IV V with reference pitch C3. Arpeggiate the notes every 1/8th of a bar (you can repeat notes in the chords). Probability of downbeat notes is always 1. Probability of other notes is 0.33. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{ "notes" : [
{ "note_id" : 0, "pitch" : 55, "start_time" : 0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 1, "pitch" : 58, "start_time" : 0.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 2, "pitch" : 62, "start_time" : 1.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 3, "pitch" : 58, "start_time" : 1.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 4, "pitch" : 55, "start_time" : 2.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 5, "pitch" : 58, "start_time" : 2.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 6, "pitch" : 62, "start_time" : 3.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 7, "pitch" : 58, "start_time" : 3.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },

{ "note_id" : 8, "pitch" : 63, "start_time" : 4, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 9, "pitch" : 67, "start_time" : 4.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 10, "pitch" : 70, "start_time" : 5.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 11, "pitch" : 67, "start_time" : 5.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 12, "pitch" : 63, "start_time" : 6.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 13, "pitch" : 67, "start_time" : 6.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 14, "pitch" : 70, "start_time" : 7.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 15, "pitch" : 67, "start_time" : 7.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },

{ "note_id" : 16, "pitch" : 60, "start_time" : 8, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 17, "pitch" : 63, "start_time" : 8.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 18, "pitch" : 67, "start_time" : 9.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : 0, "release_velocity" : 64 },
```

```
{ "note_id" : 19 , "pitch" : 63 , "start_time" : 9.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 20 , "pitch" : 60 , "start_time" : 10.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 21 , "pitch" : 63 , "start_time" : 10.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 22 , "pitch" : 67 , "start_time" : 11.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 23 , "pitch" : 63 , "start_time" : 11.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
  
{ "note_id" : 24 , "pitch" : 62 , "start_time" : 12 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 25 , "pitch" : 66 , "start_time" : 12.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 26 , "pitch" : 69 , "start_time" : 13.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 27 , "pitch" : 66 , "start_time" : 13.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 28 , "pitch" : 62 , "start_time" : 14.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 29 , "pitch" : 66 , "start_time" : 14.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 30 , "pitch" : 69 , "start_time" : 15.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 },  
{ "note_id" : 31 , "pitch" : 66 , "start_time" : 15.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,  
"probability" : 0.33 , "velocity_deviation" : 0 , "release_velocity" : 64 }  
] }
```

Example/C4: arpeggio of a progression (with probability and velocity logic) [ChatGPT]**Prompt:**

> Using the specified JSON format (use the same format, do not add any new field) create a JSON file with the notes for a G minor progression I VI IV V with reference pitch C3. Arpeggiate the notes every 1/8th of a bar (you can repeat notes in the chords). Probability of downbeat notes is always 1. Probability of other notes is 0.33. Velocity deviation for downbeat notes is 0. Velocity deviation for other notes is -50. Do not add comments in the JSON file. Note that the start_time and duration are expressed in beats, with 1 bar = 4 beats.

Result:

```
{ "notes" : [
{ "note_id" : 0, "pitch" : 55, "start_time" : 0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 1, "pitch" : 58, "start_time" : 0.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 2, "pitch" : 62, "start_time" : 1.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 3, "pitch" : 58, "start_time" : 1.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 4, "pitch" : 55, "start_time" : 2.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 5, "pitch" : 58, "start_time" : 2.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 6, "pitch" : 62, "start_time" : 3.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 7, "pitch" : 58, "start_time" : 3.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },

{ "note_id" : 8, "pitch" : 63, "start_time" : 4, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 9, "pitch" : 67, "start_time" : 4.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 10, "pitch" : 70, "start_time" : 5.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 11, "pitch" : 67, "start_time" : 5.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 12, "pitch" : 63, "start_time" : 6.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 13, "pitch" : 67, "start_time" : 6.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 14, "pitch" : 70, "start_time" : 7.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 15, "pitch" : 67, "start_time" : 7.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },

{ "note_id" : 16, "pitch" : 60, "start_time" : 8, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
{ "note_id" : 17, "pitch" : 63, "start_time" : 8.5, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
{ "note_id" : 18, "pitch" : 67, "start_time" : 9.0, "duration" : 0.5, "velocity" : 100, "mute" : 0,
"probability" : 0.33, "velocity_deviation" : -50, "release_velocity" : 64 },
```

```
{ "note_id" : 19 , "pitch" : 63 , "start_time" : 9.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 20 , "pitch" : 60 , "start_time" : 10.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 21 , "pitch" : 63 , "start_time" : 10.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 22 , "pitch" : 67 , "start_time" : 11.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 23 , "pitch" : 63 , "start_time" : 11.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },

{ "note_id" : 24 , "pitch" : 62 , "start_time" : 12 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 },
{ "note_id" : 25 , "pitch" : 66 , "start_time" : 12.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 26 , "pitch" : 69 , "start_time" : 13.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 27 , "pitch" : 66 , "start_time" : 13.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 28 , "pitch" : 62 , "start_time" : 14.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 29 , "pitch" : 66 , "start_time" : 14.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 30 , "pitch" : 69 , "start_time" : 15.0 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 },
{ "note_id" : 31 , "pitch" : 66 , "start_time" : 15.5 , "duration" : 0.5 , "velocity" : 100 , "mute" : 0 ,
"probability" : 0.33 , "velocity_deviation" : -50 , "release_velocity" : 64 }
] }
```

Examples [B] of manipulate (5-step collect-copy-prompt-copy-create process) using web Gemini:**# Example/M1: transpose [Gemini]****Prompt:**

```
> Transpose of all pitches for two semitones up given the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field): {"notes": [{"note_id": 0, "pitch": 71, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 1, "pitch": 71, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 2, "pitch": 71, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 3, "pitch": 74, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 4, "pitch": 74, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 5, "pitch": 74, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 6, "pitch": 74, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 7, "pitch": 76, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 8, "pitch": 77, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 9, "pitch": 77, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 10, "pitch": 77, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 11, "pitch": 77, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 12, "pitch": 77, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 13, "pitch": 77, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 14, "pitch": 79, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 15, "pitch": 79, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 16, "pitch": 81, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 17, "pitch": 81, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 18, "pitch": 81, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 19, "pitch": 83, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 20, "pitch": 83, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 21, "pitch": 84, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 22, "pitch": 84, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 23, "pitch": 86, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }]}
```

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 73, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 73, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 73, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
```


Example/M2: arpeggio [Gemini]

Prompt:

```
> Arpeggiate in 1/8 of a bar (repeating patterns to fill each bar), based on the chords in the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field): { "notes" : [ { "note_id" : 0, "pitch" : 71, "start_time" : 0, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 1, "pitch" : 71, "start_time" : 16, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 2, "pitch" : 71, "start_time" : 28, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 3, "pitch" : 74, "start_time" : 0, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 4, "pitch" : 74, "start_time" : 16, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 5, "pitch" : 74, "start_time" : 20, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 6, "pitch" : 74, "start_time" : 28, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 7, "pitch" : 76, "start_time" : 12, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 8, "pitch" : 77, "start_time" : 0, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 9, "pitch" : 77, "start_time" : 4, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 10, "pitch" : 77, "start_time" : 16, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 11, "pitch" : 77, "start_time" : 20, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 12, "pitch" : 77, "start_time" : 24, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 13, "pitch" : 77, "start_time" : 28, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 14, "pitch" : 79, "start_time" : 8, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 15, "pitch" : 79, "start_time" : 12, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 16, "pitch" : 81, "start_time" : 4, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 17, "pitch" : 81, "start_time" : 20, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 18, "pitch" : 81, "start_time" : 24, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 19, "pitch" : 83, "start_time" : 8, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 20, "pitch" : 83, "start_time" : 12, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 21, "pitch" : 84, "start_time" : 4, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 22, "pitch" : 84, "start_time" : 24, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 23, "pitch" : 86, "start_time" : 8, "duration" : 4, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 } ] }
```

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 71, "start_time": 0.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 74, "start_time": 0.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 77, "start_time": 1.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 3, "pitch": 71, "start_time": 1.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
```

```
{ "note_id": 4, "pitch": 74, "start_time": 2.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 5, "pitch": 77, "start_time": 2.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 6, "pitch": 71, "start_time": 3.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 7, "pitch": 74, "start_time": 3.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 8, "pitch": 77, "start_time": 4.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 9, "pitch": 81, "start_time": 4.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 10, "pitch": 84, "start_time": 5.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 11, "pitch": 77, "start_time": 5.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 12, "pitch": 81, "start_time": 6.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 13, "pitch": 84, "start_time": 6.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 14, "pitch": 77, "start_time": 7.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 15, "pitch": 81, "start_time": 7.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 16, "pitch": 79, "start_time": 8.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 17, "pitch": 83, "start_time": 8.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 18, "pitch": 86, "start_time": 9.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 19, "pitch": 79, "start_time": 9.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 20, "pitch": 83, "start_time": 10.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 21, "pitch": 86, "start_time": 10.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 22, "pitch": 79, "start_time": 11.0, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 23, "pitch": 83, "start_time": 11.5, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 }
}
}
```

Example/M3: change timing (swing) [Gemini]**Prompt:**

```
> Change the timing of notes with no change to downbeat notes and other notes shifted of 1/32th, based on the notes in the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field): { "notes": [ { "note_id": 0, "pitch": 71, "start_time": 0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 1, "pitch": 71, "start_time": 1.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 2, "pitch": 71, "start_time": 3, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 3, "pitch": 74, "start_time": 0.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 4, "pitch": 74, "start_time": 2, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 5, "pitch": 74, "start_time": 3.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 6, "pitch": 77, "start_time": 1, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 7, "pitch": 77, "start_time": 2.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 8, "pitch": 77, "start_time": 4, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 9, "pitch": 77, "start_time": 5.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 10, "pitch": 77, "start_time": 7, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 11, "pitch": 79, "start_time": 8, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 12, "pitch": 79, "start_time": 9.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 13, "pitch": 79, "start_time": 11, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 14, "pitch": 81, "start_time": 4.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 15, "pitch": 81, "start_time": 6, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 16, "pitch": 81, "start_time": 7.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 17, "pitch": 83, "start_time": 8.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 18, "pitch": 83, "start_time": 10, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 19, "pitch": 83, "start_time": 11.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 20, "pitch": 84, "start_time": 5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 21, "pitch": 84, "start_time": 6.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 22, "pitch": 86, "start_time": 9, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 23, "pitch": 86, "start_time": 10.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 } ] }
```

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 71, "start_time": 0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 71, "start_time": 1.625, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 71, "start_time": 3.125, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
```

```
{ "note_id": 3, "pitch": 74, "start_time": 0.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 4, "pitch": 74, "start_time": 2.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 5, "pitch": 74, "start_time": 3.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 6, "pitch": 77, "start_time": 1.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 7, "pitch": 77, "start_time": 2.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 8, "pitch": 77, "start_time": 4, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 9, "pitch": 77, "start_time": 5.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 10, "pitch": 77, "start_time": 7.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 11, "pitch": 79, "start_time": 8, "duration": 0.5, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 12, "pitch": 79, "start_time": 9.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 13, "pitch": 79, "start_time": 11.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 14, "pitch": 81, "start_time": 4.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 15, "pitch": 81, "start_time": 6.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 16, "pitch": 81, "start_time": 7.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 17, "pitch": 83, "start_time": 8.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 18, "pitch": 83, "start_time": 10.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 19, "pitch": 83, "start_time": 11.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 20, "pitch": 84, "start_time": 5.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 21, "pitch": 84, "start_time": 6.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 22, "pitch": 86, "start_time": 9.125, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 23, "pitch": 86, "start_time": 10.625, "duration": 0.5, "velocity": 80, "mute": 0,
"probability": 1, "velocity_deviation": 0, "release_velocity": 64 }
]
}
```

Example/M4: split notes [Gemini]**Prompt:**

```
> Split notes in 1/16th of a bar , based on the chords in the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field or comment): { "notes" : [ { "note_id" : 0 , "pitch" : 60 , "start_time" : 0 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 1 , "pitch" : 60 , "start_time" : 8 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 2 , "pitch" : 60 , "start_time" : 12 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 3 , "pitch" : 62 , "start_time" : 4 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 4 , "pitch" : 64 , "start_time" : 0 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 5 , "pitch" : 64 , "start_time" : 8 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 6 , "pitch" : 65 , "start_time" : 12 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 7 , "pitch" : 67 , "start_time" : 0 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 8 , "pitch" : 67 , "start_time" : 4 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 9 , "pitch" : 69 , "start_time" : 8 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 10 , "pitch" : 69 , "start_time" : 12 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 11 , "pitch" : 71 , "start_time" : 4 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ] }
```

Result:

```
{
  "notes": [
    { "note_id": 0, "pitch": 60, "start_time": 0, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 1, "pitch": 60, "start_time": 1, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 2, "pitch": 60, "start_time": 2, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 3, "pitch": 60, "start_time": 3, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 4, "pitch": 60, "start_time": 8, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 5, "pitch": 60, "start_time": 9, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 6, "pitch": 60, "start_time": 10, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 7, "pitch": 60, "start_time": 11, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 8, "pitch": 60, "start_time": 12, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 9, "pitch": 60, "start_time": 13, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 10, "pitch": 60, "start_time": 14, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 11, "pitch": 60, "start_time": 15, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
    { "note_id": 12, "pitch": 62, "start_time": 4, "duration": 1, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  ]
}
```



```
{ "note_id": 41, "pitch": 69, "start_time": 13, "duration": 1, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 42, "pitch": 69, "start_time": 14, "duration": 1, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 43, "pitch": 69, "start_time": 15, "duration": 1, "velocity": 80, "mute": 0, "probability":
1, "velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 44, "pitch": 71, "start_time": 4, "duration": 1, "velocity": 80, "mute": 0, "probability": 1,
"velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 45, "pitch": 71, "start_time": 5, "duration": 1, "velocity": 80, "mute": 0, "probability": 1,
"velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 46, "pitch": 71, "start_time": 6, "duration": 1, "velocity": 80, "mute": 0, "probability": 1,
"velocity_deviation": 0, "release_velocity": 64 },
{ "note_id": 47, "pitch": 71, "start_time": 7, "duration": 1, "velocity": 80, "mute": 0, "probability": 1,
"velocity_deviation": 0, "release_velocity": 64 }
]
}
```

Examples [B] of manipulate (5-step collect-copy-prompt-copy-create process) using web ChatGPT:**# Example/M1: transpose [ChatGPT]****Prompt:**

```
> Transpose of all pitches for two semitones up given the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field): {"notes": [{"note_id": 0, "pitch": 71, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 1, "pitch": 71, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 2, "pitch": 71, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 3, "pitch": 74, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 4, "pitch": 74, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 5, "pitch": 74, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 6, "pitch": 74, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 7, "pitch": 76, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 8, "pitch": 77, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 9, "pitch": 77, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 10, "pitch": 77, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 11, "pitch": 77, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 12, "pitch": 77, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 13, "pitch": 77, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 14, "pitch": 79, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 15, "pitch": 79, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 16, "pitch": 81, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 17, "pitch": 81, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 18, "pitch": 81, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 19, "pitch": 83, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 20, "pitch": 83, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 21, "pitch": 84, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 22, "pitch": 84, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, {"note_id": 23, "pitch": 86, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }]}
```

Result:

```
{ "notes": [ { "note_id": 0, "pitch": 73, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 1, "pitch": 73, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 2, "pitch": 73, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 3, "pitch": 76, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 }, { "note_id": 4, "pitch": 76, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1,
```


Example/M2: arpeggio [ChatGPT]

Prompt:

```
> Arpeggiate in 1/8 of a bar (repeating patterns to fill each bar), based on the chords in the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field): {"notes": [{"note_id": 0, "pitch": 71, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 1, "pitch": 71, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 2, "pitch": 71, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 3, "pitch": 74, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 4, "pitch": 74, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 5, "pitch": 74, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 6, "pitch": 74, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 7, "pitch": 76, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 8, "pitch": 77, "start_time": 0, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 9, "pitch": 77, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 10, "pitch": 77, "start_time": 16, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 11, "pitch": 77, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 12, "pitch": 77, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 13, "pitch": 77, "start_time": 28, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 14, "pitch": 79, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 15, "pitch": 79, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 16, "pitch": 81, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 17, "pitch": 81, "start_time": 20, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 18, "pitch": 81, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 19, "pitch": 83, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 20, "pitch": 83, "start_time": 12, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 21, "pitch": 84, "start_time": 4, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 22, "pitch": 84, "start_time": 24, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}, {"note_id": 23, "pitch": 86, "start_time": 8, "duration": 4, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64}]}]}
```

Result:

```
{ "notes": [
  { "note_id": 0, "pitch": 71, "start_time": 0.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 1, "pitch": 74, "start_time": 0.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 2, "pitch": 77, "start_time": 1.0, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
  { "note_id": 3, "pitch": 81, "start_time": 1.5, "duration": 0.5, "velocity": 80, "mute": 0, "probability": 1, "velocity_deviation": 0, "release_velocity": 64 },
```


Example/M3: change timing (swing) [ChatGPT]

Prompt:

```
> Change the timing of notes with no change to downbeat notes and other notes shifted of 1/32th, based on the notes in the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field): { "notes" : [ { "note_id" : 0, "pitch" : 71, "start_time" : 0, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 1, "pitch" : 71, "start_time" : 2, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 2, "pitch" : 71, "start_time" : 4, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 3, "pitch" : 71, "start_time" : 6, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 4, "pitch" : 74, "start_time" : 0.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 5, "pitch" : 74, "start_time" : 2.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 6, "pitch" : 74, "start_time" : 4.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 7, "pitch" : 74, "start_time" : 6.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 8, "pitch" : 77, "start_time" : 1, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 9, "pitch" : 77, "start_time" : 3, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 10, "pitch" : 77, "start_time" : 5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 11, "pitch" : 77, "start_time" : 7, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 12, "pitch" : 79, "start_time" : 8, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 13, "pitch" : 79, "start_time" : 10, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 14, "pitch" : 81, "start_time" : 1.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 15, "pitch" : 81, "start_time" : 3.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 16, "pitch" : 81, "start_time" : 5.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 17, "pitch" : 81, "start_time" : 7.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 18, "pitch" : 83, "start_time" : 8.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 19, "pitch" : 83, "start_time" : 9.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 20, "pitch" : 83, "start_time" : 10.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 21, "pitch" : 83, "start_time" : 11.5, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 22, "pitch" : 86, "start_time" : 9, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 }, { "note_id" : 23, "pitch" : 86, "start_time" : 11, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 } ] }
```

Result:

```
{ "notes" : [
  { "note_id" : 0, "pitch" : 71, "start_time" : 0, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
  { "note_id" : 1, "pitch" : 71, "start_time" : 2, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
  { "note_id" : 2, "pitch" : 71, "start_time" : 4, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
  { "note_id" : 3, "pitch" : 71, "start_time" : 6, "duration" : 0.5, "velocity" : 80, "mute" : 0, "probability" : 1, "velocity_deviation" : 0, "release_velocity" : 64 },
```


Example/M4: split notes [ChatGPT]**Prompt:**

```
> Split notes in 1/16th of a bar , based on the chords in the JSON dictionary below which provides the notes in a MIDI clip (use the same format, do not add any new field or comment): { "notes" : [ { "note_id" : 0 , "pitch" : 60 , "start_time" : 0 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 1 , "pitch" : 60 , "start_time" : 8 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 2 , "pitch" : 60 , "start_time" : 12 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 3 , "pitch" : 62 , "start_time" : 4 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 4 , "pitch" : 64 , "start_time" : 0 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 5 , "pitch" : 64 , "start_time" : 8 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 6 , "pitch" : 65 , "start_time" : 12 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 7 , "pitch" : 67 , "start_time" : 0 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 8 , "pitch" : 67 , "start_time" : 4 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 9 , "pitch" : 69 , "start_time" : 8 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 10 , "pitch" : 69 , "start_time" : 12 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } , { "note_id" : 11 , "pitch" : 71 , "start_time" : 4 , "duration" : 4 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ] }
```

Result:

```
{ "notes" : [
  { "note_id" : 0 , "pitch" : 60 , "start_time" : 0 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 1 , "pitch" : 60 , "start_time" : 1 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 2 , "pitch" : 60 , "start_time" : 2 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 3 , "pitch" : 60 , "start_time" : 3 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,

  { "note_id" : 4 , "pitch" : 60 , "start_time" : 8 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 5 , "pitch" : 60 , "start_time" : 9 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 6 , "pitch" : 60 , "start_time" : 10 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 7 , "pitch" : 60 , "start_time" : 11 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,

  { "note_id" : 8 , "pitch" : 60 , "start_time" : 12 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 9 , "pitch" : 60 , "start_time" : 13 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 10 , "pitch" : 60 , "start_time" : 14 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
  { "note_id" : 11 , "pitch" : 60 , "start_time" : 15 , "duration" : 1 , "velocity" : 80 , "mute" : 0 , "probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,
```



```
{ "note_id" : 37 , "pitch" : 69 , "start_time" : 9 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 38 , "pitch" : 69 , "start_time" : 10 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 39 , "pitch" : 69 , "start_time" : 11 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
  
{ "note_id" : 40 , "pitch" : 69 , "start_time" : 12 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 41 , "pitch" : 69 , "start_time" : 13 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 42 , "pitch" : 69 , "start_time" : 14 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 43 , "pitch" : 69 , "start_time" : 15 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
  
{ "note_id" : 44 , "pitch" : 71 , "start_time" : 4 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 45 , "pitch" : 71 , "start_time" : 5 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 46 , "pitch" : 71 , "start_time" : 6 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 } ,  
{ "note_id" : 47 , "pitch" : 71 , "start_time" : 7 , "duration" : 1 , "velocity" : 80 , "mute" : 0 ,  
"probability" : 1 , "velocity_deviation" : 0 , "release_velocity" : 64 }  
  
] }
```

Additional guidelines for AI prompting

Working with an AI it is always important to keep a few things in mind.

- The AI needs to know the “*context*” (what are you and the AI supposed to work on).
- The AI needs to know its “*role*” and level of “*expertise*” for interacting with a human.
- Another important thing to take in mind is that an AI *needs guidance*.
 - You need to be *clear* (and not ambiguous).
 - You need to be *explicit*.
 - You need to be *concise* (and not elaborate unnecessary - it will just consume tokens!)
 - If you have complex instructions, *split them* into more prompts.
 - *Don't assume if you are not sure the AI knows*.
 - Be *careful* in what you ask

Think about the AI as a “*junior assistant producer*”, someone that never says NO, who knows everything musically speaking (from a technique point of view) and knows Ableton Live too (but never made a tune...). In the case of this package someone who has also got the tools to communicate. But it is a “*junior assistant producer*” nonetheless, indeed an assistant and a junior, so *it needs to be guided and guard-railed by you*.

Limitations

The following limitations currently apply to the J74 MIDI Assistant plug-in:

- Only MIDI clips in the Session view can be created or exported using the tool. This is the current limitation of the Max for Live API implementation of Ableton Live. Of course, you can always move MIDI clips from the Arrangement view to the Session view and use the tool on any material or generate a MIDI clip in the Session view and then move it manually to the Arrangement view.
- A maximum of about **100** notes can be handled per [COPY] or [PASTE] action

Licensing, Liability, Support

The purchase of a license for the “J74 MIDI Assistant” package gives rights for the use of the package content.

The purchase of the J74 MIDI Assistant package does NOT include rights on the use of the external AI Tool: this must be separately provided by the user.

Liability: The user (producer) always remains responsible for the AI tool actions, which potentially can be disruptive or infringe reserved rights, if left unchecked or abused. Therefore, save often your project when using an external AI and please do not make abuse of it. In any case, no liability goes to the maker of these tools.

In case of issues, please report those via email to: [info \[at\] fabriziopoce.com](mailto:info@fabriziopoce.com)