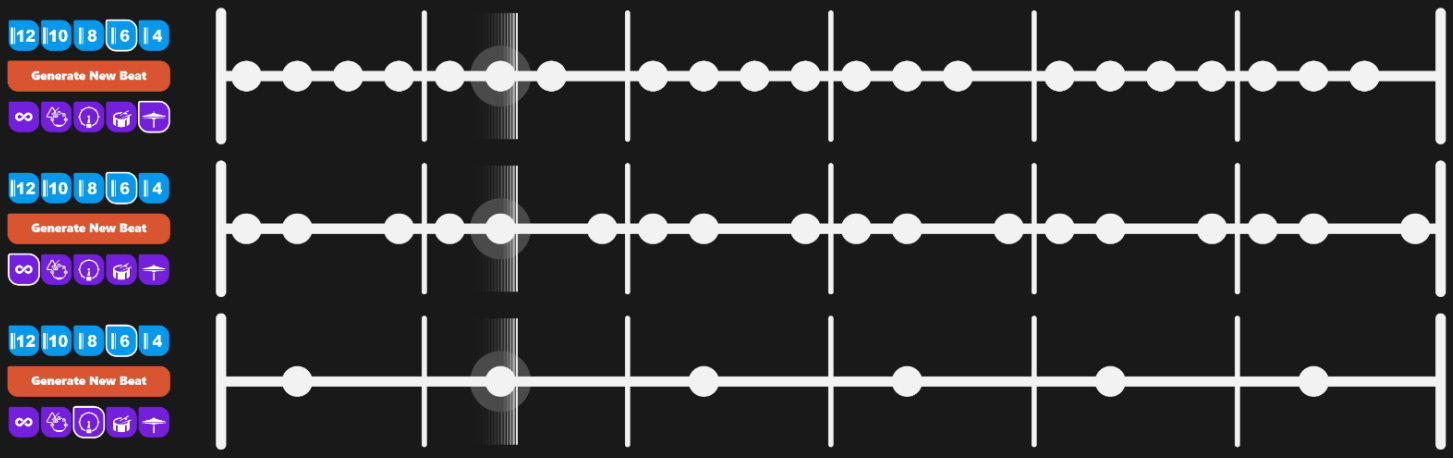
Beat.Gen



Design Description & Concepts

Beat.Gen is a tool for algorithmically generating drum/percussion beats. It is designed not only as an interactive, sound-based algorithmic design, but also as a tool to potentially aid in composition of music, as both object-oriented programming and composition share something in common;

> *"Both are a means of creating objects and structuring relationships between them"*
> – Moore, A. (2016)

Composition is an ongoing process often involving trial and error[1], and Beat.Gen can help reduce the tediousness of this process by allowing for easy, quick generation of rhythmic patterns – unaffected by subconscious influences that hamper originality[2]. These rhythmic patterns can help form a base for a more complex, complete composition. Beat.Gen has several elements to it, which are explained below.

The 'Node'

A node in Beat.Gen describes a single 'beat' that is to be played at a point in time, which is visible as a circle inside a stave. Each node holds information about its position on the stave, whether or not it is a 'rest', and the sound it plays when the 'sweeper' passes over it. These nodes are the fundamental building blocks of the algorithmically generated beats.

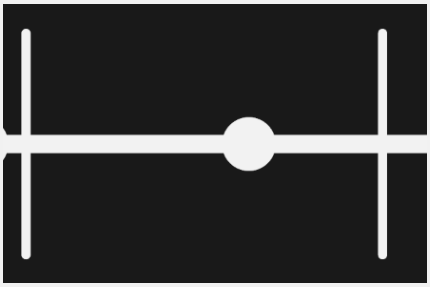


Figure 1: *A single node as seen in Beat.Gen.*

[1] (Moore, 2016) [2] (Delone & Wittlich, 1975)

The 'Stave'

Beat.Gen generates a rhythmic pattern of beats in ostinato (a looping pattern[1]) in 3 'staves'. A single stave consists of a row containing 'nodes'. Each of the staves has its own algorithmically generated set of 'nodes', and generates every node it contains with the same percussive instrument sound.

Figure 2: *An entire stave consisting of 6 'bars', with its buttons visible on the right.*

The 'Sweeper'

The sweeper moves synchronously across every stave at a uniform speed, and every time it passes through the centre of a node, the node plays its sound. Although visually each stave appears to have its own sweeper, all 3 sweepers share the same position at any given time, so that the nodes played from the 3 different staves are predictably synchronised. When a sweeper reaches the end of its stave, it moves back to the start so that the beat loops without interruption.

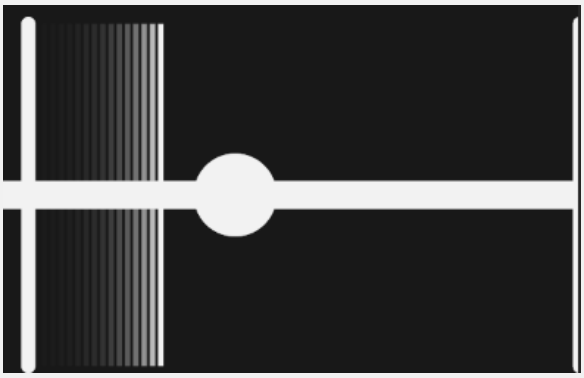


Figure 3.1: *The sweeper, about to pass through a node.*

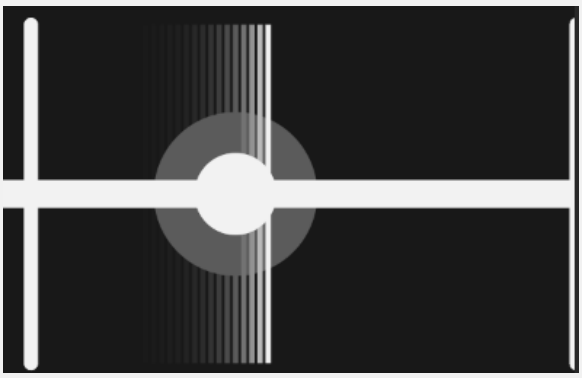


Figure 3.2: *The sweeper as it passes through a node.*

The 'Bars' & 'Rests'

Each of the staves consist of a number of bars, and each bar holds exactly 4 nodes (some of which are rests). This, combined with the sweeper moving at a fixed speed across the staves results in staves with more bars playing at a higher tempo, and staves with fewer bars playing at a lower tempo.

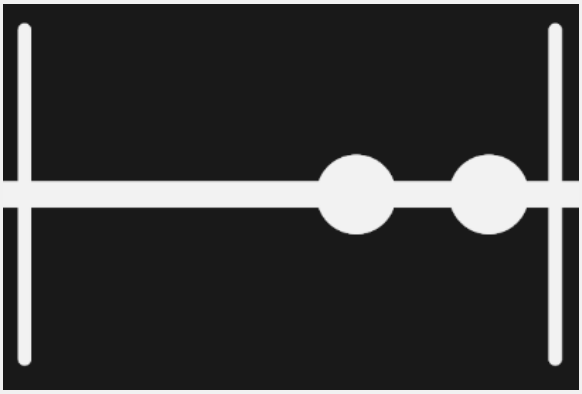


Figure 4: *A bar with 2 rests and 2 normal nodes.*

Rests are nodes which are not meant to be silent, meaning they do not play any sound when the sweeper passes over them, nor are they visible in the stave, as seen in Figure 4, where the first 2 nodes are not visible due to them being rests.

[1] *("Ostinato", 2018)*

The Algorithmic Generation

When coming up with methods to generate the pattern of nodes, a few things had to be taken into consideration; limiting the complexity as much as possible and ensuring minimal resource usage through code optimisation – due to the application running on browsers (which is significantly more limited in resources compared to desktop applications[1]), generating them in a pattern that would be versatile enough to be adapted to most styles, and also ensuring that the end result is simple enough for people to comprehend and without trouble and find pleasant.

After some thought and trail-and-error with rudimentary prototypes, I settled on using a relatively simple algorithm that generated nodes in strophic form, as it is the simplest but most flexible form of music rhythm[2]. The entire pattern generated consists of single duration 'crotchet' notes and rests, in a 4/4-time signature. The generative process is outlined below:

- The algorithm first chooses the current instrument for a stave from the available pool of sound files.
- It then picks the number of bars according to the setting, and calculates the distance between two nodes based on the length of the stave and the number of bars.
- It then chooses between generating a unique pattern of nodes over either one or two bars, and randomly generates each node as either a normal node, or a rest.

As the meter of the stave is already established into 0-4 uniformly distributed accents per bar, the (delineated) random placements of rests and nodes still result in a coherent ostinato[3]; even multiple staves with this generation that are seemingly syncopated to each other combine to form a rational rhythm[4][5].

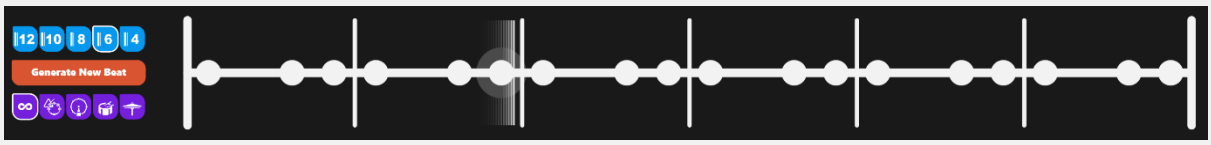


Figure 5: *A stave with the generated nodes. Note how the algorithm has chosen to generate a unique pattern of nodes over a single bar, and the remaining bars repeat the first bar in strophic form. The setting buttons for the stave are also visible on the left.*

The Settings & Buttons

Each of the staves has its own settings for the number of bars it should contain, and the category of percussion instruments it should choose from to play when generated *(see Figure 5)*. Each of the staves have their own set of the following buttons;

- **Bar Buttons:** These are the top row of blue buttons, and they change the number of bars for its stave.
- **Generate Button:** This large orange button in the middle is fairly self-explanatory; when pressed, it generates a new set of nodes for its stave.
- **Instruments Button:** Each of these purple buttons in the bottom row changes the pool of instruments the generation algorithm picks from for the stave.
  Going from left to right – choose from the entire pool of available sound files, choose from the miscellaneous percussion sounds such as triangles and tambourines, choose from kick drums, choose from snare drums, and choose from hat drums.

[1]("Designing for Web or Desktop?", 2018) [2](Grove & Sadie, 1995)
[3](Kamien, 1976) [4](Berry, 1976) [5](Duckworth, 2015)

Evaluations

The First Iteration

The first iteration had a number of differences to the final version, and to get early feedback, heuristic evaluations and Self-Assessment-Manikins (SAM) were carried out. Three major problems were identified with this first iteration of Beat.Gen;

- The elements would be incorrectly placed when the browser window was at different resolutions and aspect ratios.
- The only button available was the reset button, and users felt they needed to be able to have more control over what instruments were being picked and how many bars the stave had.
- The staves would sometimes have incompatible numbers of bars between them (as they were randomly picked upon generation) and resulted in beats that were unpleasant.

The Second Iteration

Taking the feedback from the evaluations into account, the second iteration went through a number of changes and improvements:

- The window size of the browser was considered and elements were scaled accordingly, while keeping the aspect ratio of the canvas fixed to 16:9.
- The bars buttons and instrument buttons were provided, to give users more control.
- Staves all started off at 6 bars so that the initial experience would generally be pleasant.

To get some feedback on these changes, evaluations were carried out one more time. The experience of users this time was generally quite positive, with only one main issue raised – the new settings buttons did not apply any visible changes until the next generation, and this was quite confusing for first time users.

The Final Iteration

The final iteration saw the addition of the tooltip, which displayed an appropriate message at the bottom centre of the screen when hovering over a button.

Concluding Notes

Beat.Gen improved quite a lot from the initial idea to what it is now, and the feedback from my peers and the evaluations were overwhelmingly positive. I'd like to thank everyone who helped with the evaluations for this, as well as the tutors for helpful advice and guidance; it's been a fun assessment! Lastly, a special mention to Ableton[1], for providing the initial source of inspiration for Beat.Gen!

[1] (https://learningmusic.ableton.com/)

Bibliography

- Kamien, R. (1976). *Music: An Appreciation* (7th ed.). New York: McGraw-Hill.
- Berry, W. (1976). *Structural functions in music* (pp. 301-425). New Jersey: Prentice-Hall, INC.
- Moore, A. (2016). *Sonic art*. New York: Routledge.
- Delone, R., & Wittlich, G. (1975). *Aspects of twentieth-century music* (pp. 208-211). Englewood Cliff, N.J.: Prentice-Hall.
- Duckworth, W. (2015). *Creative approach to music fundamentals.* New York: Cengage learning.
- Grove, G., & Sadie, S. (1995). *The new Grove dictionary of music and musicians.* London: Macmillan.
- Ostinato. (2018). *Encyclopaedia Britannica.* Retrieved from https://www.britannica.com/art/ostinato
- Get started | Learning Music (Beta). (2018). Retrieved from https://learningmusic.ableton.com/
- Designing for Web or Desktop?. (2018). Retrieved from https://msdn.microsoft.com/en-us/library/ms973831.aspx

Appendix : Evaluation Data

Self-Assessment Manikin (SAM) – First Iteration

Self-Assessment Manikin 1

It was obvious how to use it	←-0---------→Was difficult to figure out
I found it interesting	←-0---------→I found it boring
I felt involved/important	←-----0-----→It did everything for me, I had nothing to do
It is creative	←--0--------→It's not creative
It can be useful	←-----0-----→I can't see this being useful

Self-Assessment Manikin 2

It was obvious how to use it	←0----------→Was difficult to figure out
I found it interesting	←--0--------→I found it boring
I felt involved/important	←------0----→It did everything for me, I had nothing to do
It is creative	←-0---------→It's not creative
It can be useful	←--0--------→I can't see this being useful

Self-Assessment Manikin 3

It was obvious how to use it	←----0------→Was difficult to figure out
I found it interesting	←-0---------→I found it boring
I felt involved/important	←----0------→It did everything for me, I had nothing to do
It is creative	←---0-------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Self-Assessment Manikin 4

It was obvious how to use it	←---0-------→Was difficult to figure out
I found it interesting	←---0-------→I found it boring
I felt involved/important	←------0----→It did everything for me, I had nothing to do
It is creative	←----0------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Self-Assessment Manikin 5

It was obvious how to use it	←0----------→Was difficult to figure out
I found it interesting	←---0-------→I found it boring
I felt involved/important	←----0------→It did everything for me, I had nothing to do
It is creative	←---0-------→It's not creative
It can be useful	←-----0-----→I can't see this being useful

Self-Assessment Manikin 6

It was obvious how to use it	←---0-------→Was difficult to figure out
I found it interesting	←0----------→I found it boring
I felt involved/important	←-------0---→It did everything for me, I had nothing to do
It is creative	←----0------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Heuristics Evaluation – First Iteration

Heuristics Evaluation 1

Visibility of system status: 1
Match between system and the real world: 2
User control and freedom: 2.5
Consistency and standards: 1
Error prevention: 0
Recognition rather than recall: 1
Flexibility and efficiency of use: 0
Aesthetic and minimalist design: 0
Help users recognize, diagnose, and recover from errors: 0
Help and documentation: 2

Heuristics Evaluation 2

Visibility of system status: 0
Match between system and the real world: 1
User control and freedom: 3
Consistency and standards: 0

Error prevention: 0
Recognition rather than recall: 1
Flexibility and efficiency of use: 0
Aesthetic and minimalist design: 0
Help users recognize, diagnose, and recover from errors: 0
Help and documentation: 1

Heuristics Evaluation 3

Visibility of system status: 0
Match between system and the real world: 0
User control and freedom: 2
Consistency and standards: 0
Error prevention: 0
Recognition rather than recall: 0
Flexibility and efficiency of use: 0
Aesthetic and minimalist design: 0
Help users recognize, diagnose, and recover from errors: 0
Help and documentation: 0

Heuristics Evaluation 4

Visibility of system status: 0
Match between system and the real world: 1
User control and freedom: 3
Consistency and standards: 0
Error prevention: 0
Recognition rather than recall: 0
Flexibility and efficiency of use: 0
Aesthetic and minimalist design: 0
Help users recognize, diagnose, and recover from errors: 0
Help and documentation: 1

Heuristics Evaluation 5

Visibility of system status: 0
Match between system and the real world: 0
User control and freedom: 1
Consistency and standards: 0

Error prevention: 0
Recognition rather than recall: 1
Flexibility and efficiency of use: 0
Aesthetic and minimalist design: 0
Help users recognize, diagnose, and recover from errors: 0
Help and documentation: 0

Heuristics Evaluation 6

Visibility of system status: 0
Match between system and the real world: 1
User control and freedom: 3
Consistency and standards: 1
Error prevention: 0
Recognition rather than recall: 0
Flexibility and efficiency of use: 0
Aesthetic and minimalist design: 0
Help users recognize, diagnose, and recover from errors: 0
Help and documentation: 0

Self-Assessment Manikin (SAM) – Second Iteration

Self-Assessment Manikin 7

It was obvious how to use it	←-----0-----→Was difficult to figure out
I found it interesting	←-0---------→I found it boring
I felt involved/important	←-0---------→It did everything for me, I had nothing to do
It is creative	←----0------→It's not creative
It can be useful	←-0---------→I can't see this being useful

Self-Assessment Manikin 8

It was obvious how to use it	←-0---------→Was difficult to figure out
I found it interesting	←--0--------→I found it boring
I felt involved/important	←--0--------→It did everything for me, I had nothing to do
It is creative	←-0---------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Self-Assessment Manikin 9

It was obvious how to use it	←------0----→Was difficult to figure out
I found it interesting	←-0---------→I found it boring
I felt involved/important	←--0--------→It did everything for me, I had nothing to do
It is creative	←---0-------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Self-Assessment Manikin 10

It was obvious how to use it	←--------0--→Was difficult to figure out
I found it interesting	←---0-------→I found it boring
I felt involved/important	←--0--------→It did everything for me, I had nothing to do
It is creative	←--0--------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Self-Assessment Manikin 11

It was obvious how to use it	←----0------→Was difficult to figure out
I found it interesting	←-0---------→I found it boring
I felt involved/important	←--0--------→It did everything for me, I had nothing to do
It is creative	←---0-------→It's not creative
It can be useful	←---0-------→I can't see this being useful

Self-Assessment Manikin 12

It was obvious how to use it	←------0----→Was difficult to figure out
I found it interesting	←0----------→I found it boring
I felt involved/important	←---0-------→It did everything for me, I had nothing to do
It is creative	←----0------→It's not creative
It can be useful	←-----0-----→I can't see this being useful