Legacy BGA How-To (Japanese How-To)

You may have noticed that older BMS files do not come bundled with actual video files, such as **`.AVI`**, **`.WMV`** or **`.MPEG`**, but instead has each frame of the BGA included as **`.BMP`** files.

This comes from the fact that movie files were too large in size to be conventionally downloaded back in the late 90s and early 2000s, therefore 16-bit BMP files were used instead, as they were much more space-efficient.

Sure, you may have never actually peeked inside a BMS folder to see what is inside, but BMS songs that use bitmap files are easy to notice.
For instance, the BGA will always have extremely good synchronization with the music, as pictures are quicker to load. Movie files need to be decoded and loaded first.

You might also notice that the BGA reuses very many scenes, and that the BGA might have inconsistent framerates. This is because traditional BGAs use the same timing system as the notes, meaning that the framerate is set by BPM and quantization.

So the question remains. How do you create something like this?

#BMP

This tag in the BMS determines what note each picture is assigned to.
The old format ranged from **#BMP01** to **#BMPFF**, using base16. (255 unique frames)
Modern players however support **#BMP01** up to **#BMPZZ**. (1295 unique frames)
#BMP00 is specifically reserved for a frame displayed when the player misses.
You need to list every single frame of the BGA using these tags in the BMS.

Three BGA Channels

In BMS, you are provided three channels for BGA layers:

- BGA
- POOR/MISS
- LAYER

The first, **BGA**, is used for normal BGA display.
Any picture placed in this channel with be displayed normally.

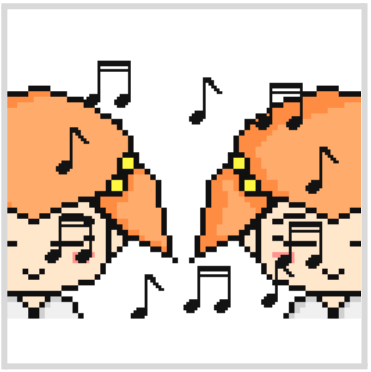
POOR/MISS is another channel that functions the same way as **BGA**, however this channel is only visible when the player either misses or hits a POOR during play.
Before this channel was implemented, miss BGA would be determined by **#BMP00**.

LAYER however is a channel that overlays **BGA**. This means you can use **LAYER** for recurring images or frames. Notable examples of this is Lapis / SHIKI, frozen bond / paraoka and like a dandelion / recognize m.

All these songs use the **LAYER** channel for overlays that appear as frames or shutters.
This image is taken from *Lapis*, and is the overlay that can be seen during the intro of the song.

You'll notice that the supposed opening between the two panels is black, and not transparent.
This is because **true black** is rendered as invisible or transparent by BMS players.

True black is defined as `RGB (00:00:00)`.
This means any black lines drawn need to be one shade lighter than **true black**.
However this is only true for 32-bit BMP.
If the image is 16-bit, `RGB (0F:0F:0F)` is also invisible.



As you can see with the BGA for *like a dandelion*, when **true black** is subtracted or colour keyed out, the contents of the image is much easier to see.

Creating BGAs with this limitation

When making a BGA using this method, you have to be very careful when planning out the scenes and the animation, especially if you plan on using **#BMPFF** to preserve compatibility with old programs, such as BM98.
Hitkey has created a table with more information about what each BMS player supports and what limitations they have. This can be found here.

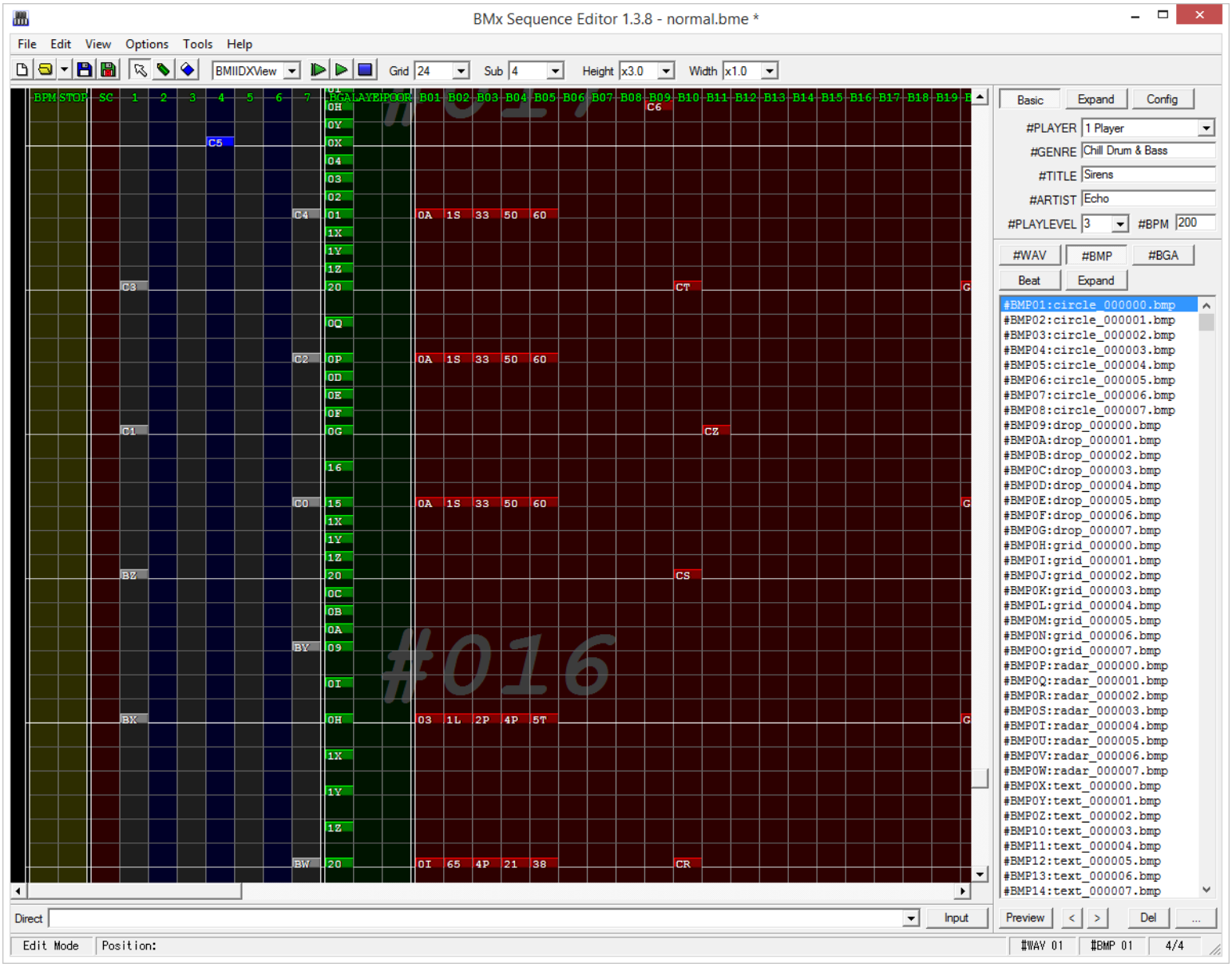
Since traditional BGAs use the same format as regular notes, framerate is decided by BPM and quantization. To be able to create a true to life preview of the BGA in your video editor, you can use grid2sec to calculate rhythmic time to milliseconds.
If possible, use a BPM-based ruler for your video editor's timeline.

Most players reserve a 1:1 space for BGAs in the play screen.
In BM98 this reserved space was **256x256** pixels, and other games followed.
This means you ought to export your frames in this resolution.
For "high quality" BGAs to be used in modern players that are able to scale, stretch and crop images, you can use **512x512** pixels instead.

When planning out the scenes, be aware of what format you are using. 255 frames is not much to work with, meaning that it is wise to reuse many scenes and to find ways to sequence them in a way to not make the BGA too repetitive.
It is also smart to make each scene loopable.

When you have created your scene loops, you can either export each loop one by one to give them unique file names, or you can export the whole thing in one go with each loop in a chronological order or similar.
Make sure that you export the scenes as an image sequence of BMP files.

Once this is done you need to do the rest of the work within a BMS editor.
BMSE is recommended as it supports **#BMP** but also **#BGA** (extended BGA format, not to be confused with the channel).
Keep in mind that BMSE requires Visual Basic 6.



On the left side, you can switch to the **#BMP** tab which shows you the list of all frames added.
If you want to limit yourself to **#BMPFF**, you can adjust this by going to "`Options`" and clicking "`Use Old Format [01-FF]`"
You add frames to **#BMP** by dragging your images to the tag you want to start on, such as **#BMP01**.
Once you have added your frames, you can start adding the sequence to the BMS file.
You do this by adding notes to one of the green columns, depending on what channel you want to use. The most efficient way to do this is to add each scene side by side in the red field, then use copy and paste to paste each scene to where it belongs.

It is also possible to use something like woslicerII to quickly create a BMSE clipboard paste for pasting down a sequence of notes. However to use woslicerII in such a way requires you to find a sound file long enough to place slice points, as if you are slicing keysounds instead.
This is not recommended if you are unfamiliar with woslicerII.
However, if you are interested in learning about using this software, you may refer to the document about BMS Creation, which explains in detail how to use it

#BGA is referred to as extended BGA, however it has very limited support and is not recommended besides novelty reasons..
For more elaboration, please refer to Hitkey's memo about it.

Credits

Thanks a lot to Hitkey for writing the BMS command memo

Written by Dolphin

- Twitter
- Website

Resources

- Lapis / SHIKI
- Frozen Bond / パラノイド正岡
- like a dandelion / recognize m.
- BMx Sequence Editor
- woslicerII
- grid2sec
- #BMP[00-ZZ] - BMS command memo