MKS

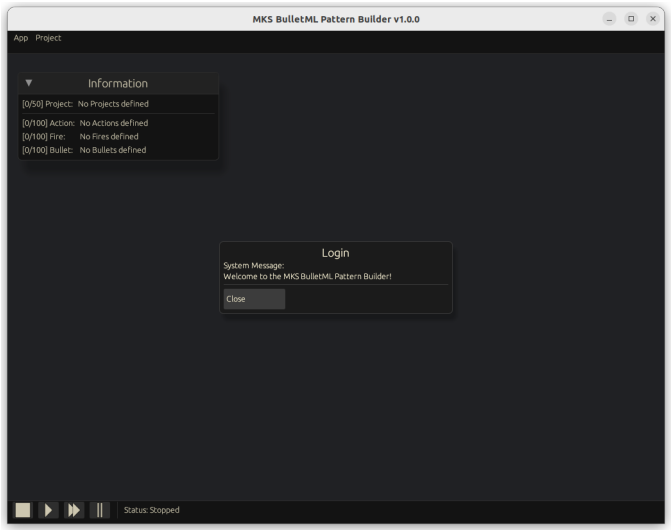
BulletML Pattern Builder

Random Spawn Tutorial

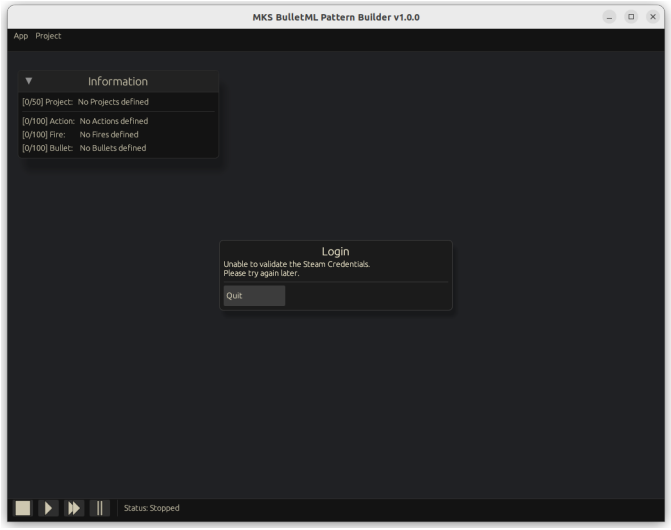
by Micah Koleoso Software, 2025
https://micahkoleososoftware.com

Getting Started

As an introduction, the goal here is to show the quick workflow setting up bullet patterns and previewing the results playing in the Playfield of the editor. On starting the application, the main user interface shows the application after a successful login and providing any system information that may be important for working with the tool. In the case shown below, there is no specific message available and the login has been successful.

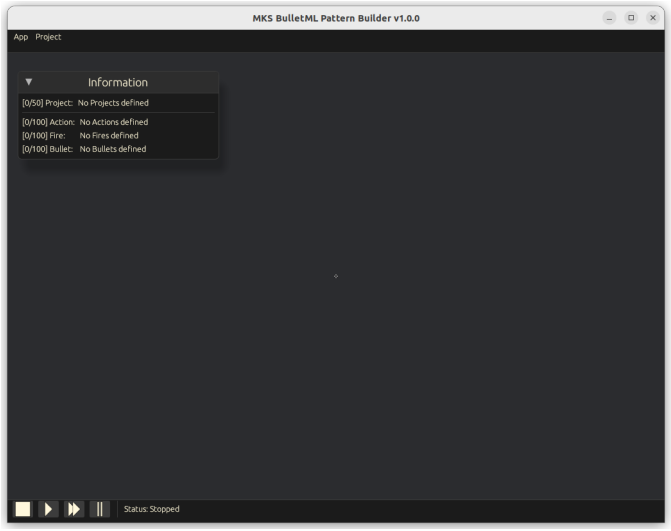


The application is launched through the Steam Launcher and requires a valid purchase on the user's Steam account as well as the execution of the Steam Launcher at all times for validation of the access rights. If the Steam Launcher is not able to verify the correct credentials and purchase or the application account is not in good standing, the Pattern Builder tool will indicate the following on start-up:



Please contact our support per email or come into our Discord Server if you encounter any issues here that we can help with.

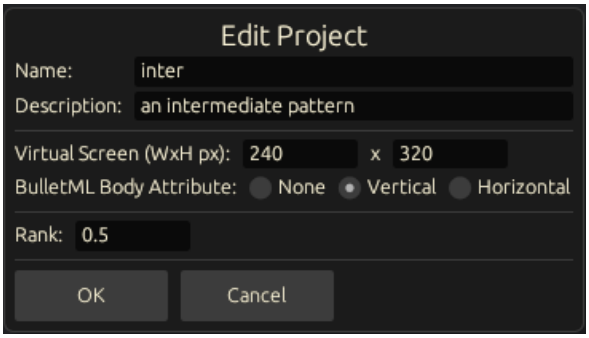
After confirmation of the start-up message dialog, the following screen is presented. As we haven't created any projects yet, the menu shows only relevant next steps.



At this stage, we can go through the introductions as described below. Note that the Information Window can be moved per mouse drag in the window title or toggled visible using the F2 key or the sub-menu option “Show/Hide UI Overlays” under the App menu.

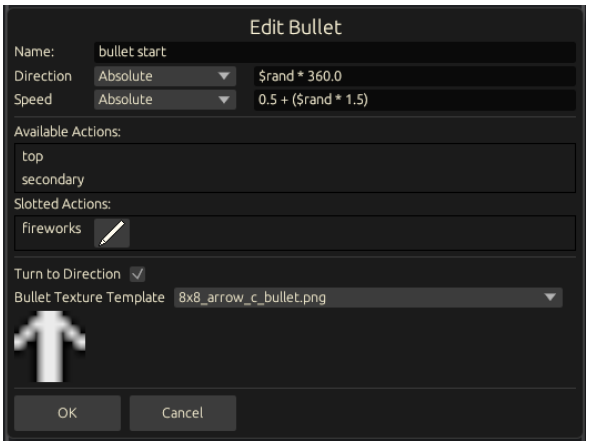
Randomized Pattern

This pattern is more intermediary and shows the basic interactions between Actions, Fires and Bullets as well how their settings influence each other during playback. How to access the menu for creating and editing the project and assets is described in the basic pattern introduction. To keep the steps simple only the main edits required for this example will be shown as screenshots.



Create a new project named “inter” with description “an intermediate pattern” and set the virtual screen width to “240”, the virtual screen height to “320” and the BulletML Body Attribute to “Vertical”. We want something resembling a vertical shooter play area in a virtual screen resolution that was common in machines in the Arcades of the 80s. Any combination of width and height equal or larger than 64 and smaller or equal to 4096 can be used, but for our purposes, we will use the classic resolution of 240 x 320 virtual pixels. The playback display area and all the sprites displayed within that area are scaled appropriately, regardless of the application actual window size or desktop resolution.

First, create three new Actions named “top”, “fireworks” and “secondary” but don’t edit them at this time. We will customize them later when other dependent assets have been created for them.



Create a new Bullet named “bullet start”. This will be the first bullet we want flying outwards from the screen center. To randomize the direction and speed, we are using an Expression and a variable of BulletML called “$rand” within that Expression. Enter the Expressions as displayed. At runtime, the variable “$rand” generates a value between and including 0.0 and 1.0 and can be used as shown above for dynamics when this Bullet is emitted.

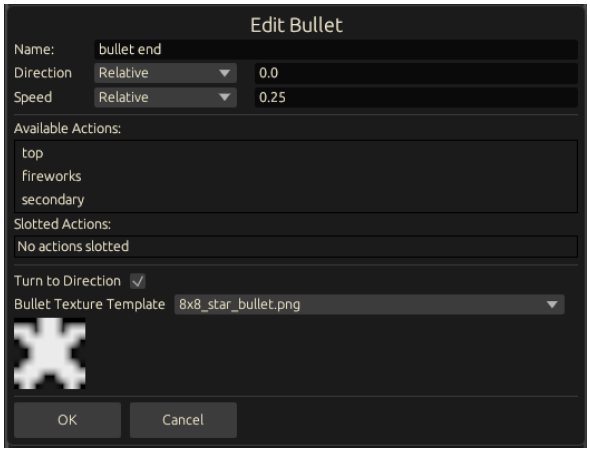
Note that we want to move the Action “fireworks” into the Slotted Actions list by clicking on it in the Available Actions list. Also, set the Arrow as sprite for this bullet to differentiate it visually from another second Bullet we will create later.

Confirm this dialog with “OK” to close it and continue.

Create a new Fire asset named “fire start”. Set the Direction and Speed attributes to “Relative” and make sure “0.0” is set for both. This is to ensure this Fire doesn’t override or influence the Bullet we will emit, as we have custom settings we will enter in the Bullet below.

The “bullet start” Bullet has already been selected per default. Close this too by clicking the “OK” button.

We now edit the Action “top”, which is the first Action we created and will always be the entry point for playback in this project. Add a reference to “fire start” and a Wait of 60 Frames, equivalent to 1 second, here as content for this Action. OK that to leave the dialog.

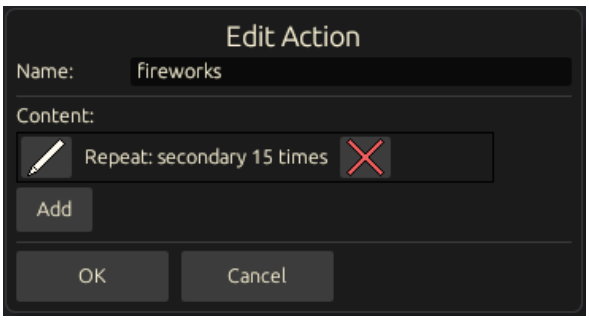


The second Bullet we want to emit is to be created as “bullet end” and contains the parameters as shown above. We have entered Direction and Speed as relative attributes where speed will slightly increase the Bullet velocity when it is spawned, compared to it’s parent Bullet. As this Bullet is a “Child” of the “Parent“ Bullet, it will use that position, direction and speed initially when spawing. The hierarchical elements of the entire BulletML structure is the most creative and complex part of building Bullet patterns and experimentation with these parameters should provide lots of creative and interesting results at playback. For now, we will just use everything as shown above.

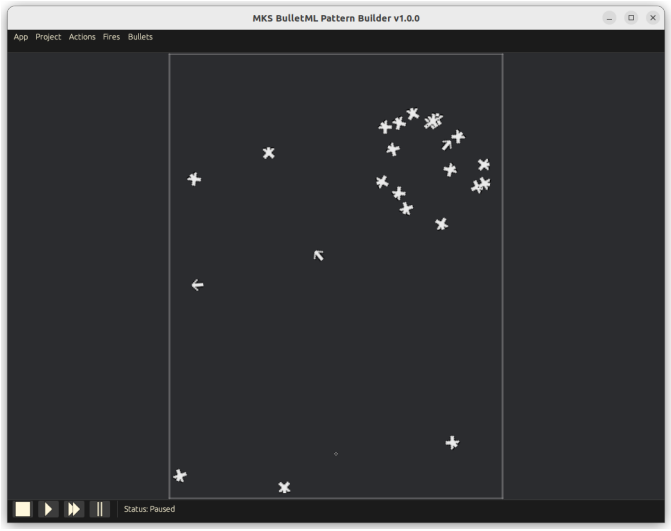
Choose the star sprite as the bullet image to use and leave everything else as is for now. OK out of there to close that dialog.

To emit the second Bullet, create a new Fire named “fire end”. Enter the Expressions as shown above in the appropriate fields and ensure “bullet end” is selected as Bullet to emit. Close this dialog, as we don’t need anything more here.

Edit the Action named “secondary” which we had created earlier and not edited yet at that time. Within it, create a Wait 60 frames asset and reference a Fire named “fire end” as ca FireRef Content here. Close this with “OK” to continue.



As a final part of the setup, edit the Action named “fireworks” to repeat calling the prior Action “secondary” 15 times. This effectively causes 15 Bullets to be spawned through calling the “secondary” Action and each time, a new Bullet of declaration “bullet end” is emitted in a random direction and speed.



Press “Play” in the status bar of the Pattern Builder application. There should be a fireworks of bullets being emitted. Initially the Arrow bullet generates out of the center in a random direction, and after about a second, each arrow sprite emits a ring of 15 star sprites at the location it is at.

This concludes the second more in depth introduction. There is a lot to experiment with and deep hierarchies can be build where bullets depend on their parent bullet locations etc.

15th August, 2025	v1.0 - Initial version.