Artix Engine

Documentation

Table of contents

Table of contents

Table of contents

Will be added soon!

6. Creating a Game

About Artix Engine

Artix Engine is a simple 2D game engine written in Python that empowers beginner game developers to create interactive games with ease. Artix Engine provides a comprehensive set of tools and features to bring your game ideas to life.

Key Features

•**Visual Programming:** Artix Engine offers a user-friendly graphical editor that eliminates the need for extensive coding skills. With its intuitive event system and attribute-based object behaviors, you can easily define interactions and behaviors for your game objects.

•**File Manager:** Importing game assets is a breeze with the built-in file manager. Seamlessly import textures into your projects, keeping everything organized and accessible.

•**Built-in Physics Engine:** Artix Engine incorporates a physics engine based on Pymunk, enabling you to incorporate realistic physics into your games effortlessly. Simulate gravity, collisions, and other physical interactions to add depth and immersion to your gameplay.

•**Project Manager:** The project manager in Artix Engine simplifies the management of your game projects. It provides a streamlined workflow, allowing you to easily navigate and access all your resources, including game assets, scripts, and configurations.

•**Game Export:** Share your games with others by exporting them as Python files. These exported files are using pygame to render the game.

•**High Frame Rates:** Artix Engine ensures smooth gameplay experiences by supporting high frame rates of up to 200 FPS in the exported games.

Requirements

To utilize Artix Engine and its features effectively, ensure that your system meets the following requirements:

Python

Artix Engine requires Python 3.9 or later to run. If you don't have Python installed on your system, follow these steps to install it:

1. Visit the official Python website at python.org.
2. Navigate to the "Downloads" section.
3. Choose the appropriate Python version for your operating system (Windows, macOS, or Linux) and download the installer.
4. Run the installer and follow the instructions to install Python on your system.
5. After the installation is complete, open a terminal or command prompt and type **python --version** to verify that Python is successfully installed. You should see the version number displayed.

If you already have Python installed on your system, make sure it meets the minimum version requirement of 3.9 or later. To check the Python version, open a terminal or command prompt and run **python --version**.

For Artix Engine to work you also need to install the python dependencies here.

Compatibility

Artix Engine has been primarily developed and tested on Windows systems. However, efforts have been made to ensure compatibility with Linux as well. While the engine should work on Linux, please note that Linux compatibility is considered experimental.

System Requirements

To ensure optimal performance and compatibility, make sure your system meets the following requirements:

- **Operating System**: Artix Engine is compatible with Windows and Linux operating systems.
- **Python**: Artix Engine requires Python 3.9 or later. Ensure that you have a compatible version of Python installed on your system.

Windows

Artix Engine has been extensively tested on various Windows platforms. It is designed to be compatible with these operating systems.

Linux

While Artix Engine aims to be compatible with Linux, it is important to note that the level of compatibility may vary depending on the specific distribution and configuration of your Linux system.

License

Summary of the GPL-3.0 License

The GNU General Public License v3.0 is a widely-used open source license that provides certain freedoms to users while ensuring the software remains free and open. Here is a summary of the key points of the license:

- **Distribution**: You are free to distribute copies of Artix Engine, either modified or unmodified, in both source and binary forms.
- **Modification**: You have the freedom to modify Artix Engine's source code to suit your needs. If you distribute the modified version, you must also provide the source code for those modifications under the same GPL-3.0 license.
- **Copyleft**: The GPL-3.0 license requires that any derivative work or modifications made to Artix Engine must also be licensed under the GPL-3.0 or a compatible license. This ensures that the software remains open and free for future users.

- **Attribution**: When distributing Artix Engine or its modified versions, you must include a copy of the GPL-3.0 license and provide clear attribution to the original authors and contributors.
- **No Warranty**: Artix Engine is provided "as is" without any warranty. The authors and contributors of Artix Engine cannot be held liable for any damages or issues arising from the use of the software.

Commercial Use

The GPL-3.0 license allows Artix Engine to be used for both personal and commercial purposes. However, it is important to note that distributing a commercial product based on Artix Engine may have implications for your own product's licensing. If you plan to use Artix Engine for commercial purposes, it is recommended to seek legal advice to ensure compliance with the GPL-3.0 license and any other relevant licenses or obligations.

Disclaimer

This information is provided as a summary of the GPL-3.0 license and does not constitute legal advice. It is your responsibility to read and understand the complete terms and conditions of the license. If you have any legal concerns or questions about the licensing terms of Artix Engine, it is recommended to consult with a legal professional.

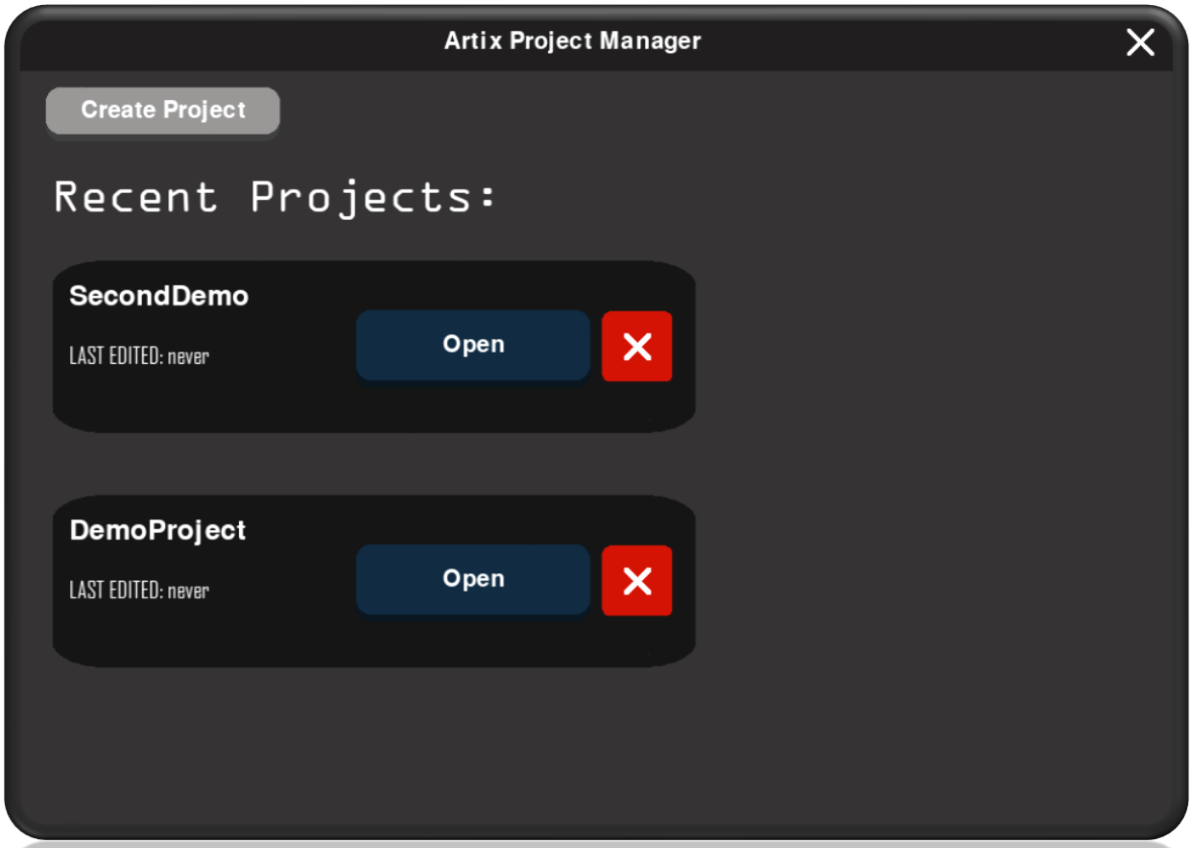
Project Manager

Creating a Project

To create a project you need to open the Project Manager.

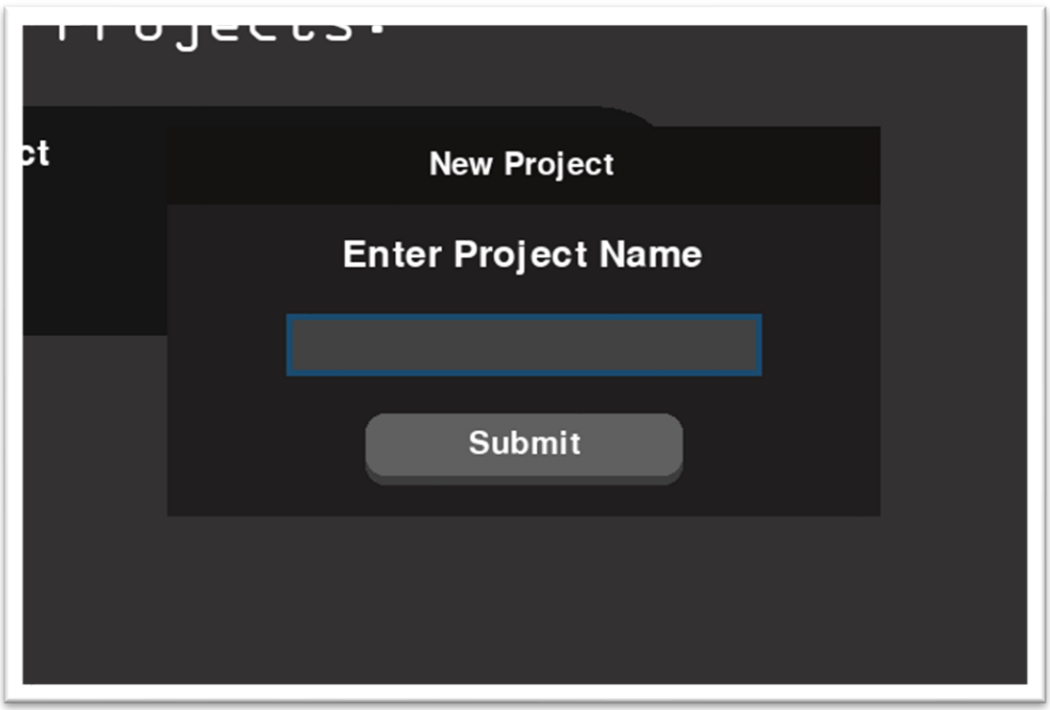
You can do this by running the „main.py“. The main.py file will start with a splash screen. If you want to open the Project Manager directly you can run „project_manager.py“.

In the Project Manager you can see all the projects you ever created. When you run Artix Engine for the first time there will be two demo projects, wich you can delete by pressing the red delete button.



„The Project Manager showing two example projects“

With the project manager open you can press the „Create Project“ button to create a new project. Once the button is pressed a small window will show up. By pressing into the input box you can specify a project name. Note that using letters like „ö, ä, ß, ü“ leads to errors when opening the editor later on.



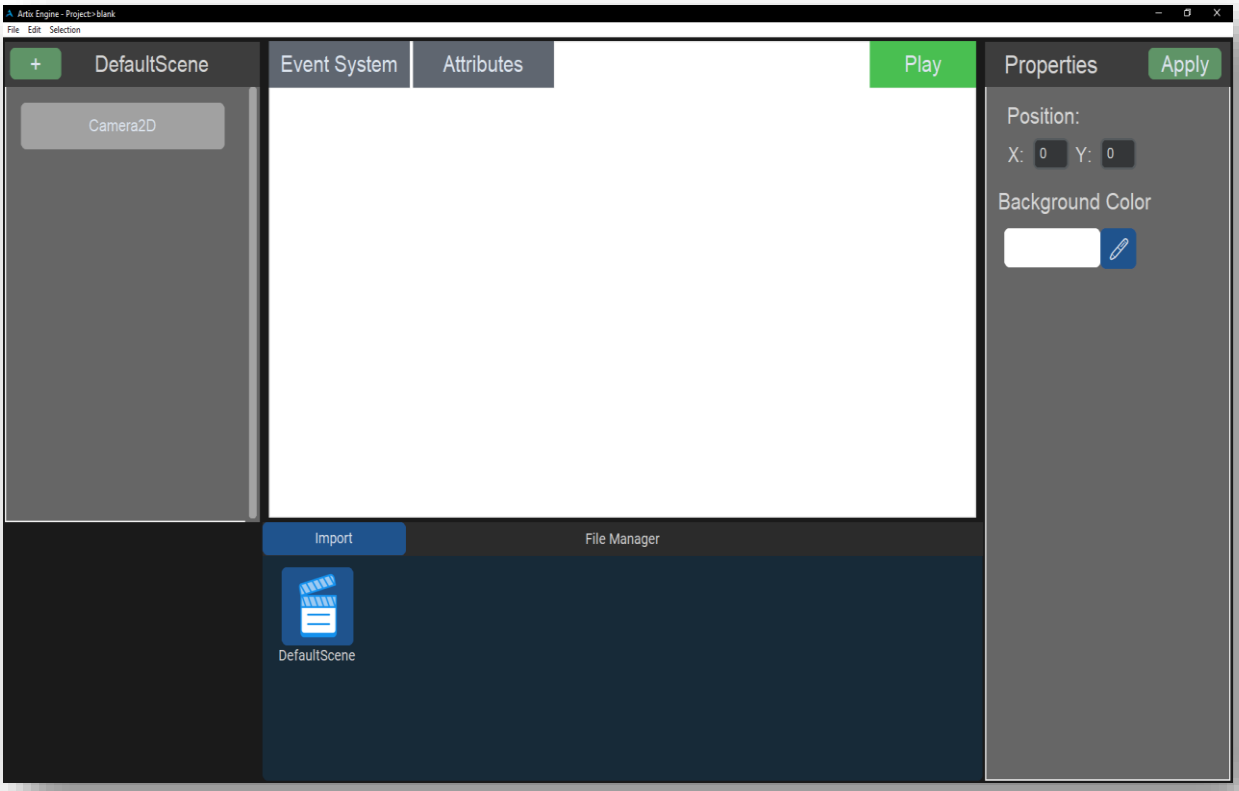
„The window that pops up when you create a new project“

Once you typed in your project name you can click Submit to create the project. The new project will show up and you can open it by clicking on the Open button. This will open Editor.

Editor

What is the Editor?

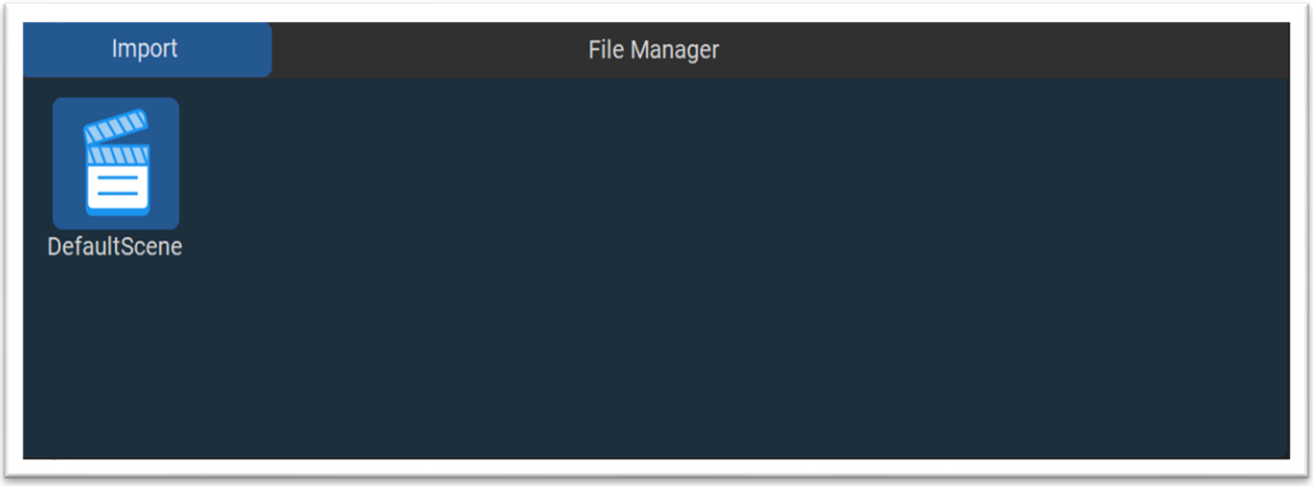
The Editor is a graphical user interface(**GUI**). It enables you to create your Games visually, eliminating the need for coding skills.



„A blank project opened in the Editor“

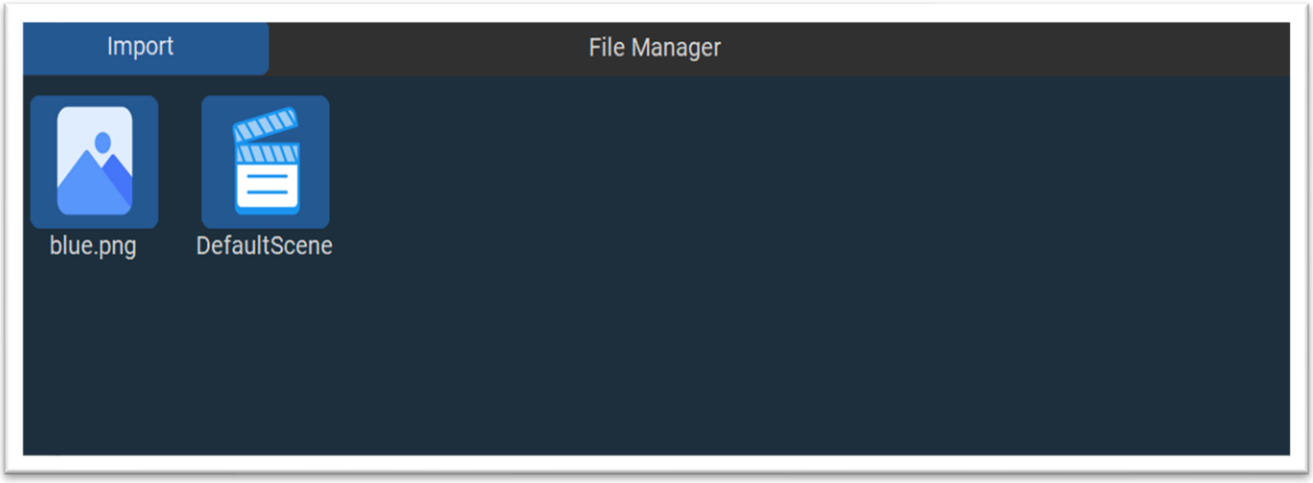
The File Manager

One of the many menus in the Editor is the File Manager. You can use it to import the Image Textures for the objects in your game into the Project. It also shows all the Scenes create in your Game. Click here to learn more about Scenes.



„File Manager of a blank project“

To import a file you can click the import button. This will open up a file explorer where you can select your specific file. Supported file types are png and jpeg(jpg). The file will then show up in the file explorer with a image icon. To delete a file you can right click on it and and click delete.



„File Manager with a imported file“

The Scenetree

In the scenetree all objects of the current Scene are displayed. When you open a project for the first time, it loads the „DefaultScene“.

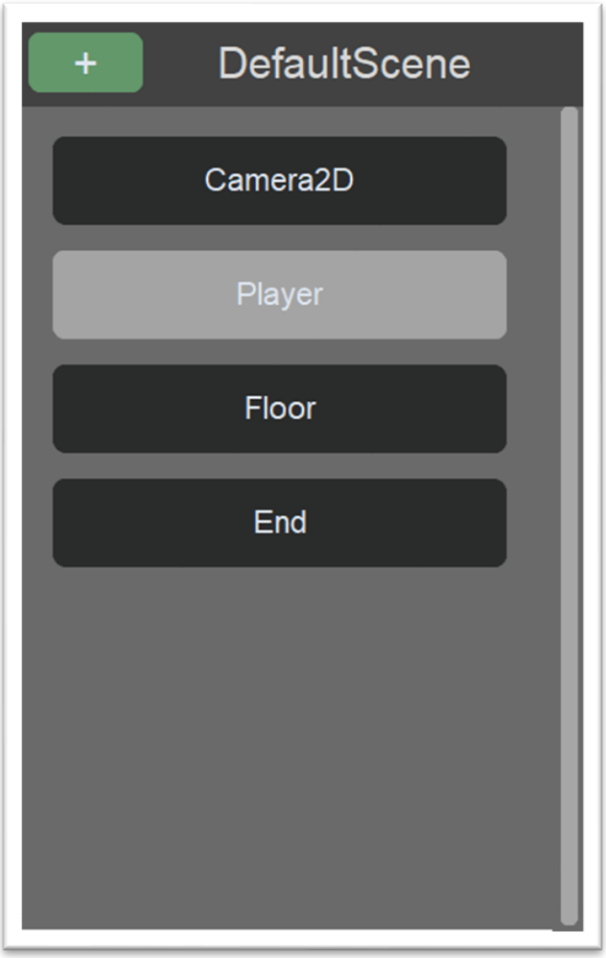
To create a new scene: File->New Scene | Then you can specify the name of the Scene. After creating a scene it will show up in the File Manager. You can open it by double clicking on it.

„Scenetree of the DefaultScene“

Every Scene has a „Camera2D“ object on default. This object cant be deleted, because it is neccesary to render the Scene in the Game.

To create a new object in the scene, you can click the green plus buttonin the Scenetree and then onto „Sprite“. This will open up a little window where you can specify the object name.

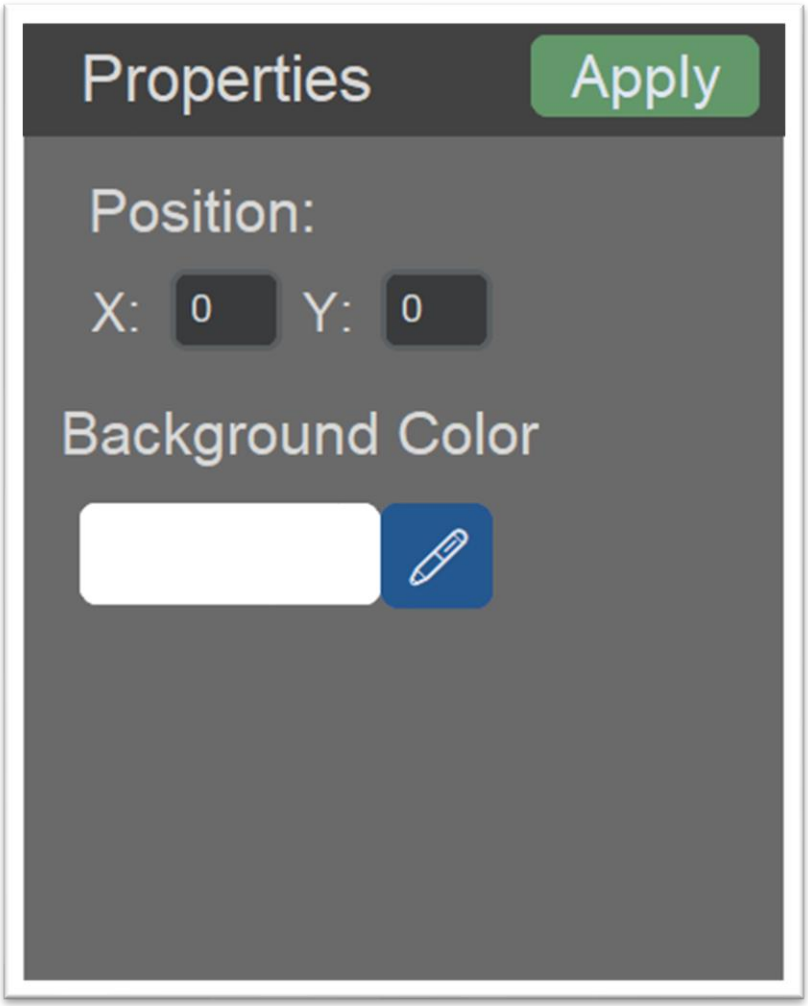
You can select different objects by clicking on them. To delete a object right click on it and then on delete.



„Scenetree showing different game objects“

The Properties Panel

The Properties Panel shows the properties of the current selected object in the Scene Tree. While all the other game objects have properties like a position, scale and image texture, the „Camera2D“ object only has a position and a background color.



„Properties of Camera2D in the Properties Panel“

!! When you change any of the values you need to press the green apply button to save the changes !!

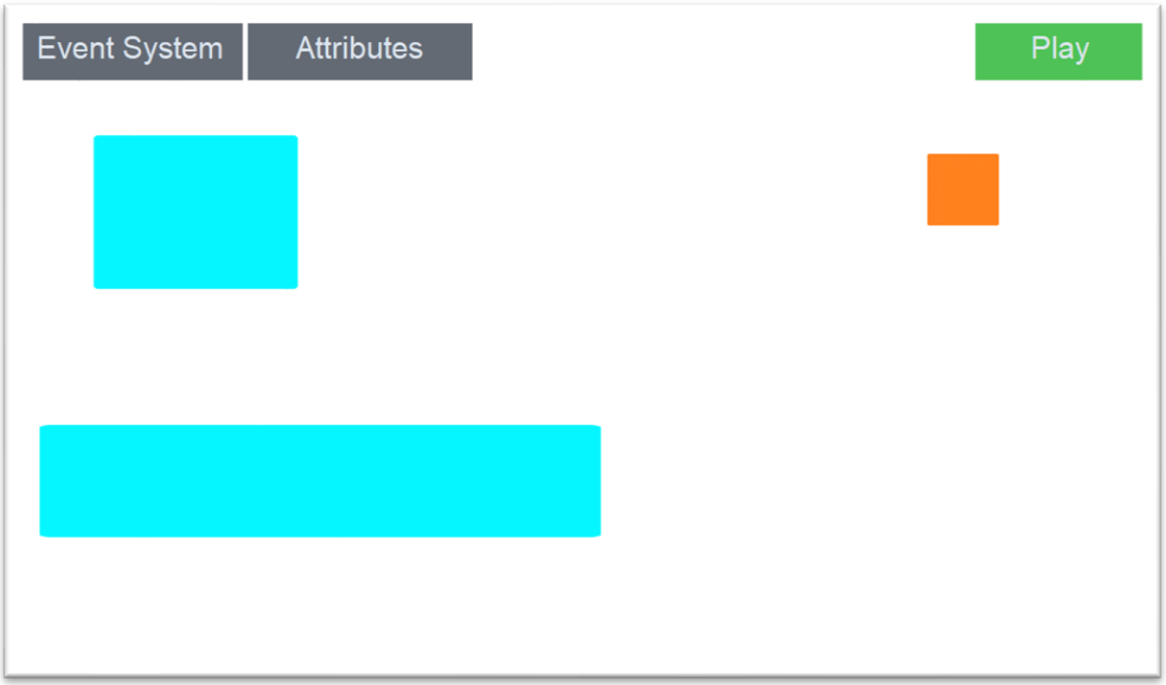
While changing the properties like position or scale is relativly easy, assigning a image texture to a object may not be clear from the beginning.

To assign such an image to the object, you can drag and drop an image from the file manager, into the white image-texture-frame in the „Properties Panel“

Note that this only works for sprites not for the „Camera2D“

The Viewport

In the viewport all the objects in your current loaded scene are displayed. Here you can build the Levels for your game and easily position all of your objects.



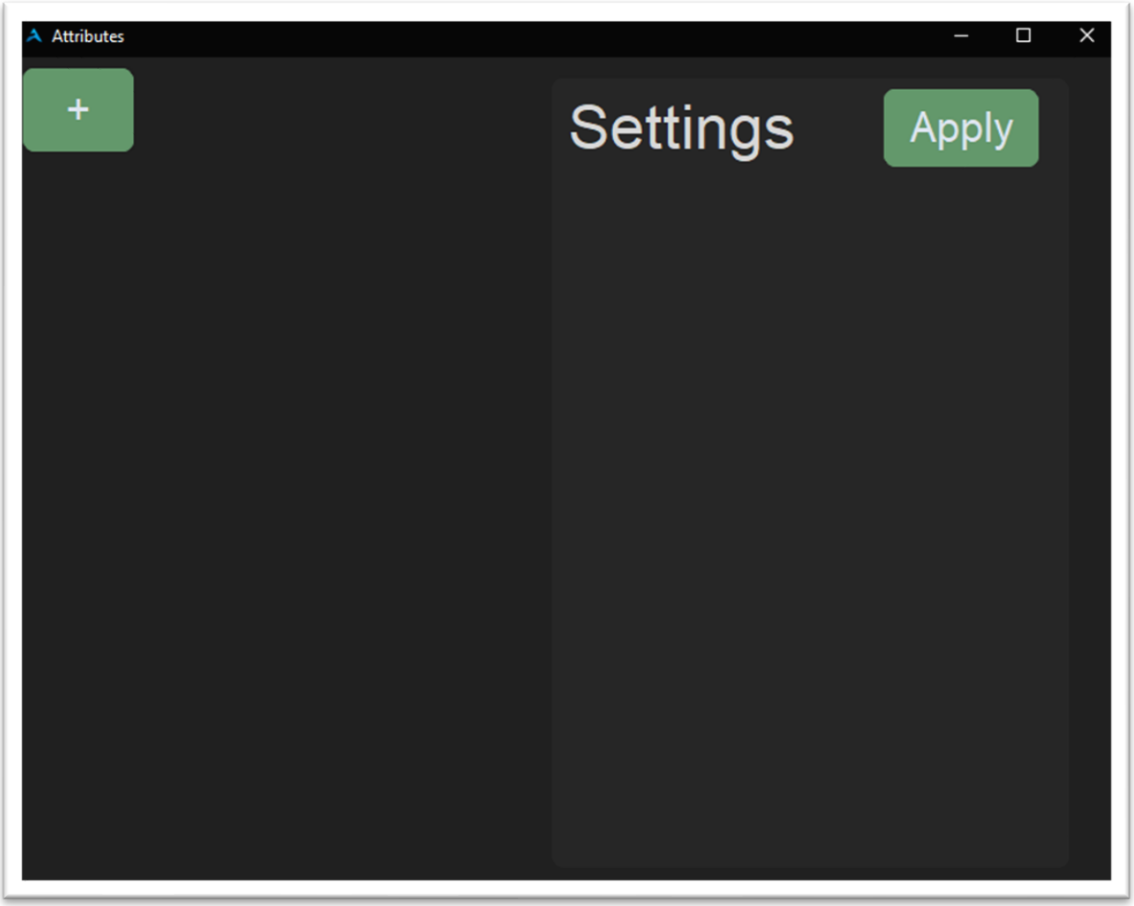
„The viewport showing a example scene“

To easily position your Objects in the Scene you can just drag them around with your cursor. The objects you drag also get selected in the Scene Tree.

The Viewport also includes the „Play“ button with wich you can play test your game. The Event System and Attributes Window can also be accessed trough the viewport.

The Attributes Window

The attributes window is used to assign Attributes to an object and edit the Attributes settings. It can be accessed trough the „Attributes“ button in the Viewport.

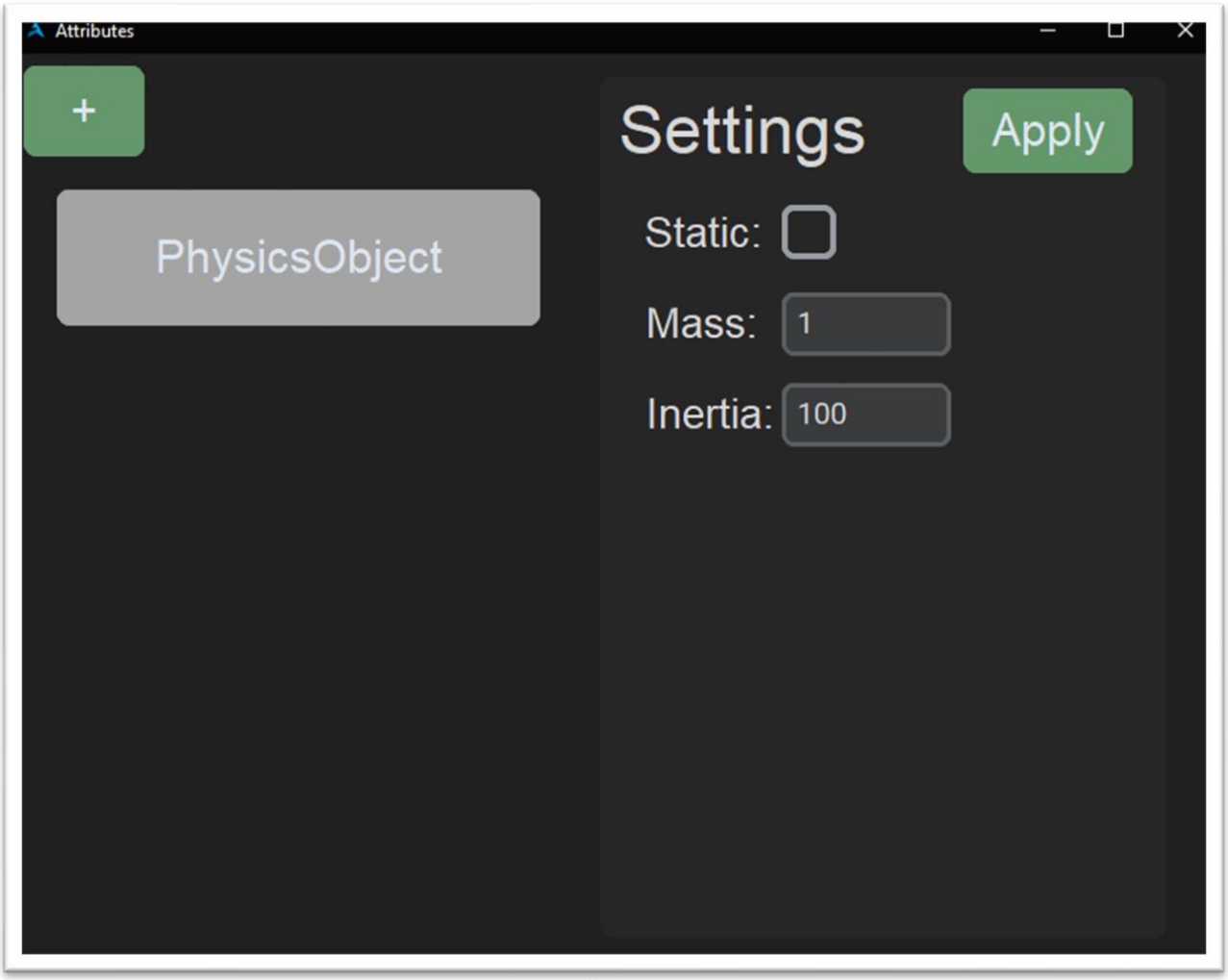


„Attributes Window of a new game object“

To add a new attribute to the selected object you can click the green plus button. Then you need to select your desired attribute.

At the moment only one attribute exists. Read more about it here.

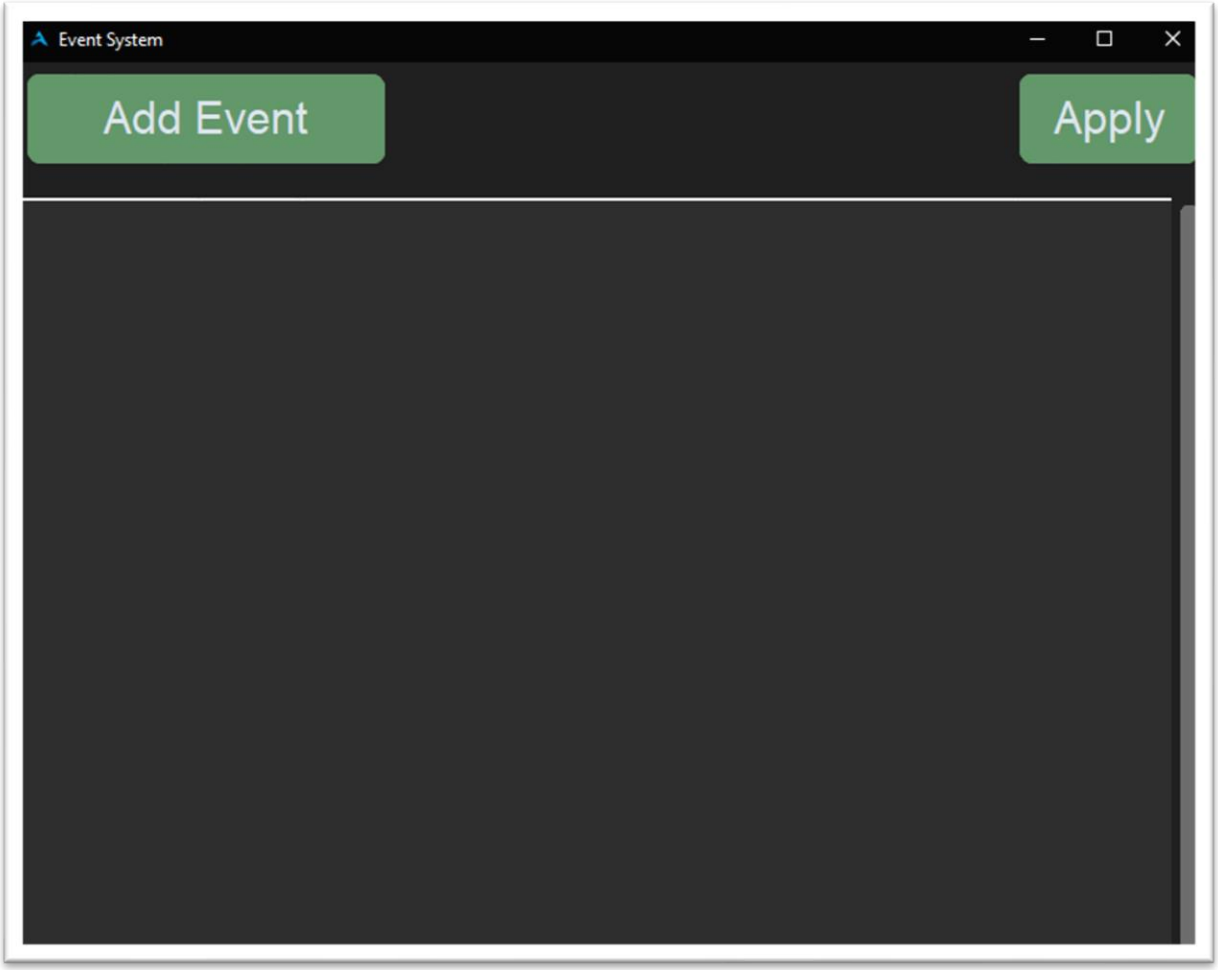
Once you added a new attribute, you can modify its settings in the settings panel. Once you have changed any of the settings you always have to click the green „apply“ button to save the changes.



„The attributes window with a new attribute“

The Event System Window

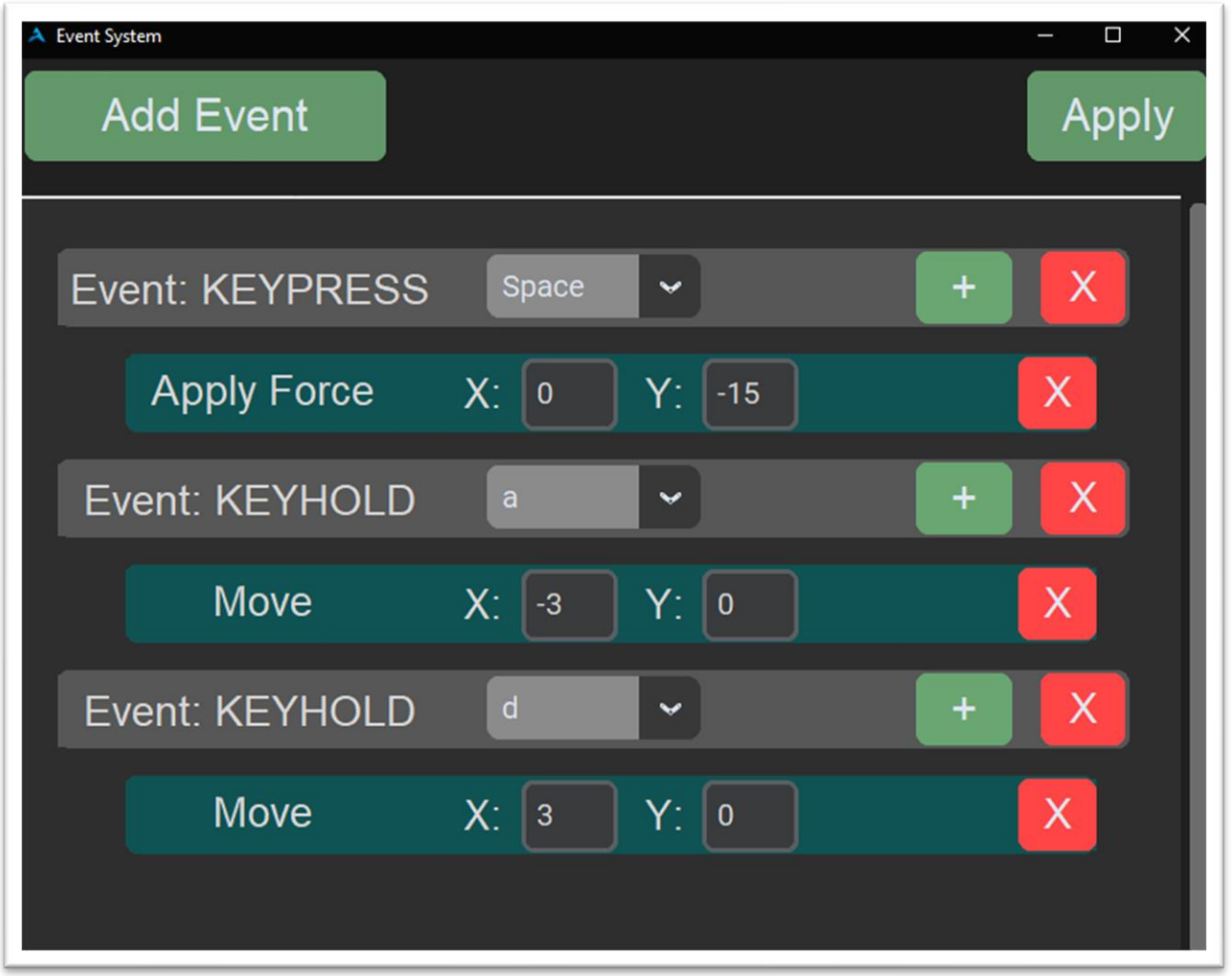
The attributes window serves as the visual programming system of the engine. Here you can define object behavior and game logic.



„The Event System window of a new game object“

To add an Event click the „Add Event“ button. Then you can select from different Events. Once you add an Event you can click onto the green plus button of the event. Once you click on it you will be able to select from different Actions.

!When you change the presets of an event you will have to click apply before adding a action to it or the presets will be reset!



„The Event System window with a Player logic“

Object Attributes

What are Object Attributes

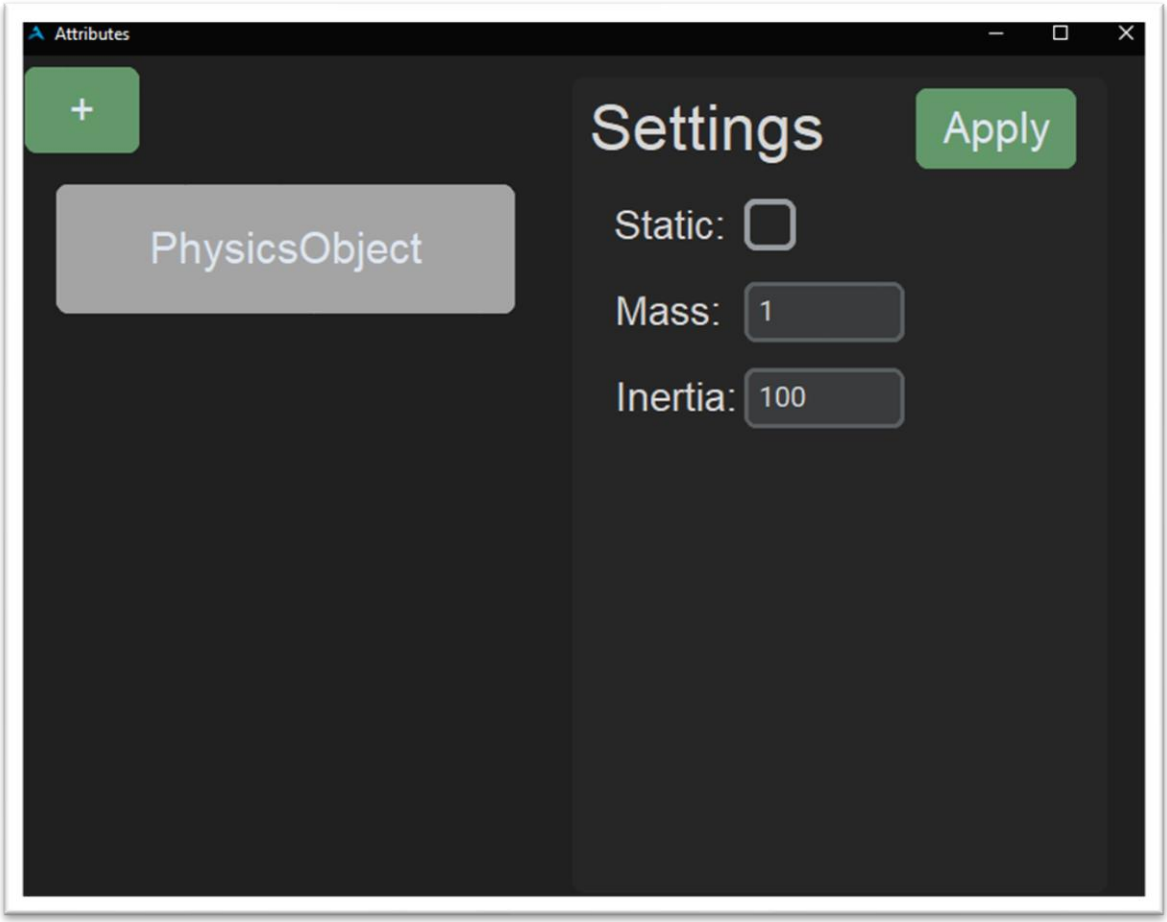
Object Attributes are pre-defined behaviors you can add to a game object. To add Object Attributes you need to use the Attribues Window.

Currently there is only one Attribute but more are going to be added soon.

Physics Object

Warning: This is an experimental feature!

The Physics Object attribute makes a object interact with the Engines Physics System.



„Physics Object Attribute in Attributes Window“

Event System

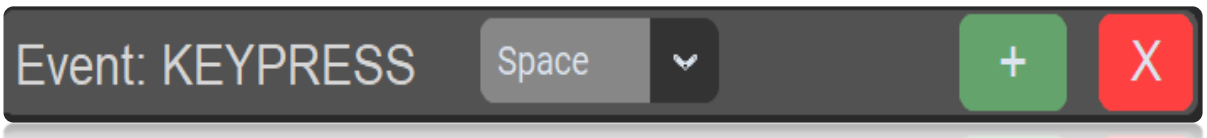
How the Event System works

The Event System enables you to define object behavior by adding different Events. You can also add different Actions to the Events. When the Event gets triggered it runs all the Actions that have been added to it.

Events

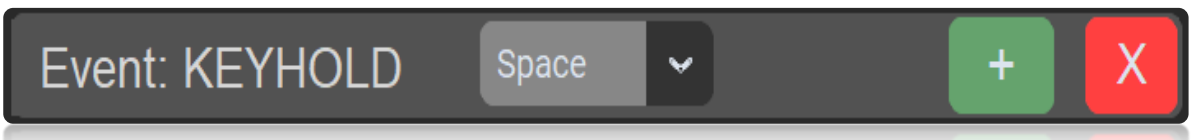
Keypress Event

The keypress event gets triggered when a certain key gets pressed on the keyboard. The difference to the keyhold is that the keypress only runs its actions once. That means, when a key is held down, the action only runs again when the key is released and pressed again.



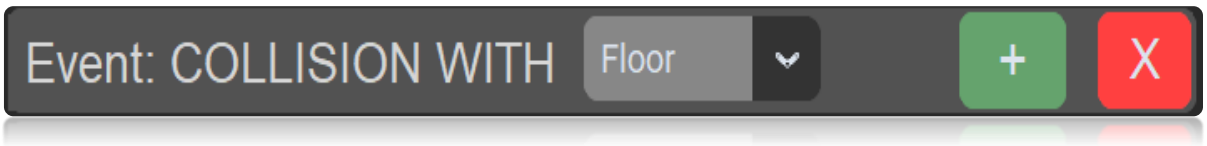
Keyhold Event

The keyhold event gets triggered when a certain key gets pressed on the keyboard. It runs its actions every frame as long as the key is pressed



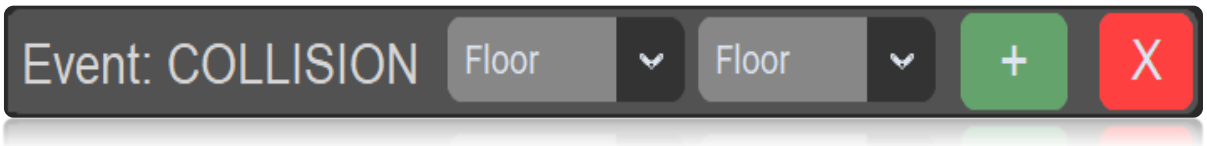
Collision With Event

The Collision with event gets triggered when the object you give the event to collides with another object.



Collision Between Event

The Collision With event gets triggered when two objects collide with each other.



Actions

Set Position Action

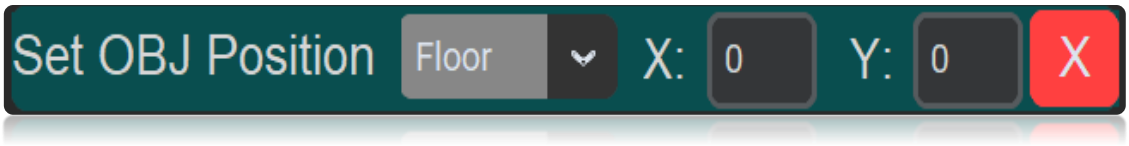
The Set Position Action sets the position of the object you add the action to.

Move Action

The Move Action moves the object you add the action to.

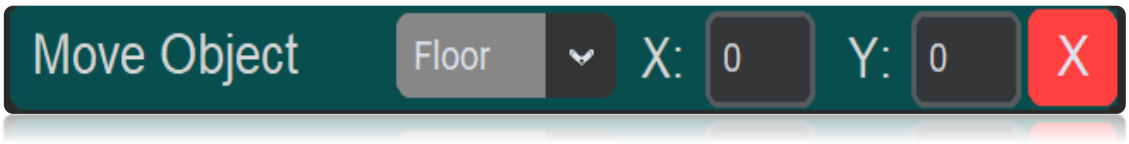
Set Object Position Action

The Set Object Position Action sets the position of a different object in the Scene.



Move Object Action

The Move Object Action moves a different object in the Scene.



Apply Force Action

The Apply Force Action applys a force onto the physics body of the object you add this action to.

Note that this only works if the object has a Physics Object Attribute.

Load Scene Action

The Load Scene Action loads a different Scene.