HORIZONTAL BLOCKS

TEMPLATE FOR BLOCKSENGINE 2

Add the **Horizontal Blocks Template** to your Blocks Engine 2 and change the user experience for your game. Friendly and familiar, based on the most popular block programming languages. Perfect for Educational, puzzle or any type of Game.

This Horizontal Blocks Template for Blocks Engine 2 asset documentation contains the detailed explanations for the implemented features and how to set up the template.

This asset requires Blocks Engine 2

Contact

Asset Store Page

Github

support@meadowgames.com

MeadowGames

MeadowGames.com

CONTENTS

1. Overview

The Horizontal Blocks Template for Blocks Engine 2 is an expansion for the original Blocks Engine 2 that adds a different experience to the game. The block's layout was modified to simplify their identification with a more even look and big icons and create a horizontal way to organize the blocks. **It is important to note that this template asset needs the Blocks Engine 2 as base for it to work properly**.

The asset expands the possibilities by making the environment suitable for different game experiences. The blocks that fit in this layout are already setup with the Horizontal template as well as the inspector's Block Builder, which can be used to create new blocks with the same look.

Even though it is possible to have a horizontal version of every one of the blocks in the original asset, not all of them were used in this template in order to maintain the user experience as simple as possible for the user to learn and understand. The blocks were limited to Trigger, Simple, Loop and Condition blocks that have none or only one input.

2. Getting Started

To start using the Horizontal template, follow the steps below:

1. Previously **import the Blocks Engine 2**;
2. **Import the Horizontal Blocks Template for Blocks Engine 2**;
3. **Open the Horizontal sample scene** at:
   Assets/BlocksEngine2_Horizontal/Scenes/HorizontalBE2SampleScene.unity.

3. User Interface

The interface contains the same elements as the original Blocks Engine 2, the (1) Blocks Selection Panel, (2) Programming Environment and the (3) Game Scene. Below is a print of the sample scene.

See the Blocks Engine 2 documentation for more.

The user interface can be customized to fit your unique project by changing the layout, size, position and other visual characteristics, as well as expanding the functionalities and adding game mechanics.

It is also possible to create more blocks and instructions and customize the blocks characteristics.

3.1. Blocks Layout

The Blocks horizontal layout has a different visual style, they have a more simple look and big icons, also, the horizontal blocks have none or only on input so the user experience is as simple as possible to learn and understand. The used blocks were limited to Trigger, Simple, Loop and Condition blocks that have none or only one input.

Even though it is possible to have Operation blocks and use them along with the horizontal blocks, they were kept out of this template to maintain a direct and easy way to code.

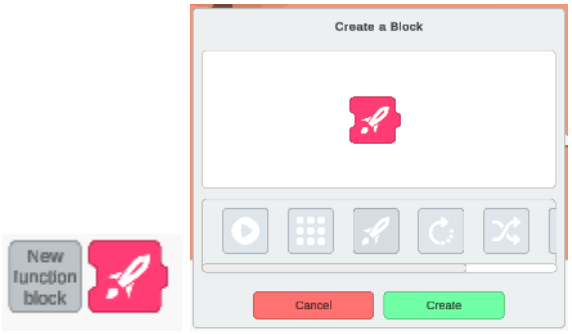
The horizontal blocks keep the same hierarchy structure as the original asset, the same interfaces are used, the way they are connected to each other is different as well, being connected to the right of the previous blocks instead of the bottom.

3.1.1. Categories

Each category of the Block Selection View is related to the general behavior of the instruction.

- **Events:** Trigger blocks that start a sequence of blocks and operation blocks that function as event listeners. Colors #F1C40F and #E74C3C;
- **Motion:** Blocks related to movement, positioning and direction of the Target Object. Color #3498DB;
- **Control:** Blocks that can alter the run cycle, repeating, delaying or conditioning the execution of a group of blocks. Color #F39C12;
- **Looks:** Blocks that alter the visual aspect of the Target Object. Color #E056FD;
- **Sound:** Blocks that execute sounds. Color #9B59B6;
- **Function Blocks**: Blocks that call the execution of the sequence of blocks placed under its corresponding Define Block. Color #FF3D74;
- **Custom:** Additional blocks later created and blocks that act on specific Target Objects. Color #7F8C8D.

The Function Blocks category has a button for creating new Function Blocks, it opens a panel to select the icon and confirm the creation.



3.1.2. Blocks

The blocks are the visual representation of the code functions, operations or variables. The user can place the function blocks in a logic sequence to build a code.

There are four block types (Trigger, Simple, Loop and Condition), those types can be identified by the shape of the block and for its behavior.

Trigger

Blocks that begin each program, sequence of blocks and can behave as event listeners. They indicate how and when the program will start running.

Simple

These blocks are usually responsible for the main actions of the program and execute its function once per cycle, they don't wrap children blocks.

Loop

Blocks that execute loop instruction, they wrap children blocks that are going to be executed if the loop condition is still met. These blocks can have multiple sections, as an example of the If/Else block.

This type of block is by default executed once per frame due to their repetitive behavior, meaning that its ExecuteInUpdate setting of its Instruction is always true.

Conditional

Blocks that execute conditional instruction and wrap children blocks that will be executed if the condition is met. They also can have multiple sections.

Define

Blocks used to define the instructions of a Function Block. The user creates a stack of blocks under it similarly to the Trigger Blocks that is executed when the corresponding Function Block is used. It has one local variable per input.

3.2. Game Scene

The Game Scene is where the code results are exposed to the user. In a 2D or 3D scenario, the Target Object performs the code instructions and can interact with other environment elements.

A virtual joystick is also present in that view, it can be used in composition with specific blocks for listening to the buttons.

There are a great number of possibilities for composing the Game Scene and letting your creativity flow, some examples are coding tutorials, puzzles, top down, combat, racing, coin catch games.

4. Horizontal BE2 Custom Components

The Horizontal blocks use the same interfaces as the default blocks, however, it uses components that are customized specifically for the Horizontal blocks, those are:

- **BE2_BlockHorizontalLayout** : I_BE2_BlockLayout
- **BE2_HorizontalBlockSection** : I_BE2_BlockSection
- **BE2_HorizontalBlockSectionBody** : I_BE2_BlockSectionBody
- **BE2_HorizontalBlockSectionHeader** : I_BE2_BlockSectionHeader
- **BE2_DragHorizontalSelectionFunction** : I_BE2_Drag
- **BE2_HorizontalFunctionBlocksManager** : BE2_FunctionBlocksManager
- **BE2_UI_HorizontalCreateFunctionBlockMenu**

The custom section components are used to adjust the settings for the horizontal block's shape, as well as make it possible to have them be connected laterally. The blocks also contain additional components for adjusting the Dropdown input of the ChangeColor block.

5. Horizontal Block Builder (BE2 Inspector)

The Block Builder in the Horizontal sample scene is set up with the needed prefabs for creating Horizontal blocks. The creation of blocks is done by the setting of variables and the use of the custom header markup to indicate the block header items.

Mind that after the creation of the block, the correspondent instruction should be implemented using C# and making use of the recommended instruction API.

For details on how to build Blocks and more, refer to the Blocks Engine 2 documentation.