Unrealistic Arts

Dice Board Game – Starter kit

User Guide

Version: 2.4.0

Date: 17 March 2022

Table of Contents

List of Tables

List of Figures

Introduction

This system will allow you to create functional map easily for dice and board games and do it really easy. Just click over menu item named 'Dice Board Starter' and create board object, create dice as many as you want, create player as many as you want, create game manager for generate map, then play. Make a relationship between Homes just by one click. Create your custom map in minimum time and make your game logic. Just enjoy it.

Scripts list

There you have all scripts to work.

Table 1 - Main scripts

Main Scripts		
BoardGameManager	Board	PlayerManager
Other Scripts		
DiceManager	Dice	DiceSide
DiceManagerEditor	Player	PlayerHome
PlayerGhost	ElementNode	ElementNodeCreator
Extensions	ArrangingPiecesInElementNode	
Shape makers		
CircleShape	EllipseShape	LineShape
PolygonShape	RectangleShape	SquareShape
triangleShape	DiamondShape	PolygonMaker
LineMaker	CircleMaker	SquareMaker
triangleMaker	RectangleMaker	
Editors		
ElementNodeEditor	ElementOptionNodeEditor	MainMenu

Note:

You don`t have to know each scripts or programming. This list is just for awareness.

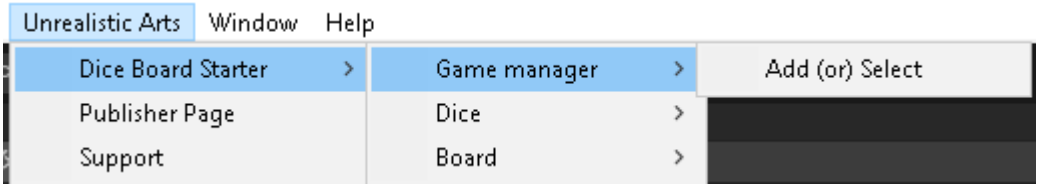
How to use menu?

Here the menu items explained quickly but in Scripts title its explained with more details.

Game Manager

Game Manager is the main core for handling game logic that create the board home(s) and manage all the system. In “(Add or) Select” item you can add game manager script if not exist and select that object automatically.

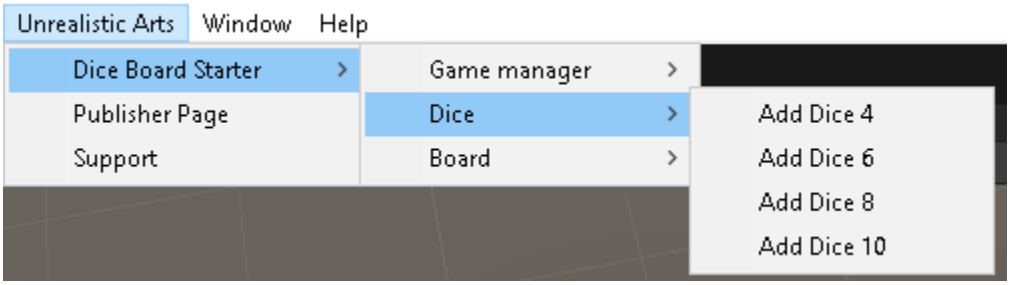
Figure 1 - Game manager menu



Dice

Dice is the core for handling all dice(s). In here you can see some different type of dices. you can choose and add any count of any dices type.

Figure 2 - Dice menu



Board

Board is the core for handling all board(s). In ‘Board’ item you can add any type of board.

Figure 3 - Board menu

Basic scripts and their parameters

BoardGameManager.cs

It`s the main script and connector between dice(s), player(s) and board. It`s attached to project by adding 'Game manager' from menu item. The main function on this script is managing board home(s) and adding player(s). with left click on screen new home will generate. Ctrl + Right click on each node to open node editor window or changing their positions and so on.

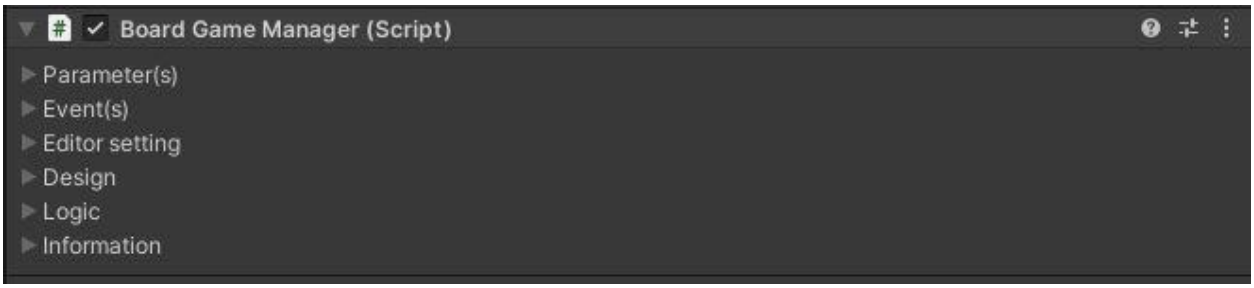
Path: Dice Board Game - Starter Kit\Scripts\BoardGameManager.cs

Here we have some general options. Each section has its own variables.

Table 2 - BoardGameManager (General sections)

Name	Description
Parameter(s)	Game parameters
Event(s)	Game events
Editor setting	Total node and pieces settings
Design	Total node and pieces design
Logic	Game logic parameters
Information	Some useful information

Figure 4 – BoardGameManager inspector (General sections)



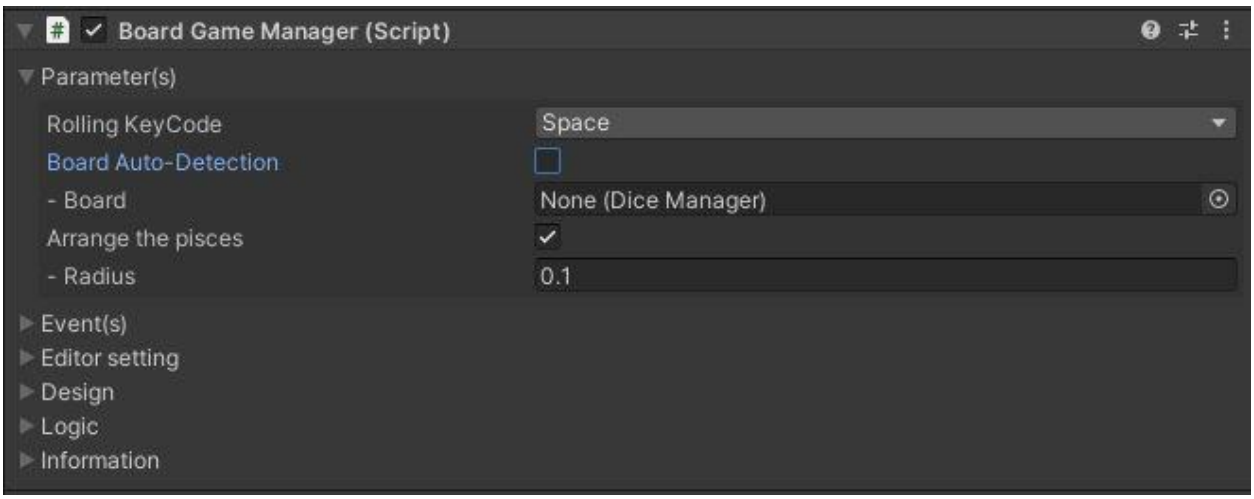
Parameters

Here we have parameter`s variables.

Table 3 - BoardGameManager (parameters)

Name	Description
Rolling Key Code	The key that when it`s press, dice or dices are rolling.
Board Auto-Detection	Auto detection board if exist
-Board	Board you can attach manually
Arrange the Pisces	Arranging the Pisces when they are in same node.
- Radius	Arranging radius (Circle mode)

Figure 5 - BoardGameManager inspector (Parameters)



Events

Here we have Event`s variables.

Table 4 - BoardGameManager (Events)

Name	Description
On player skipping event(s)	The array of events that invoking when player turn is skipping
On next player event(s)	The array of events that invoking when next player turn is coming

Figure 6 - BoardGameManager inspector (Events)

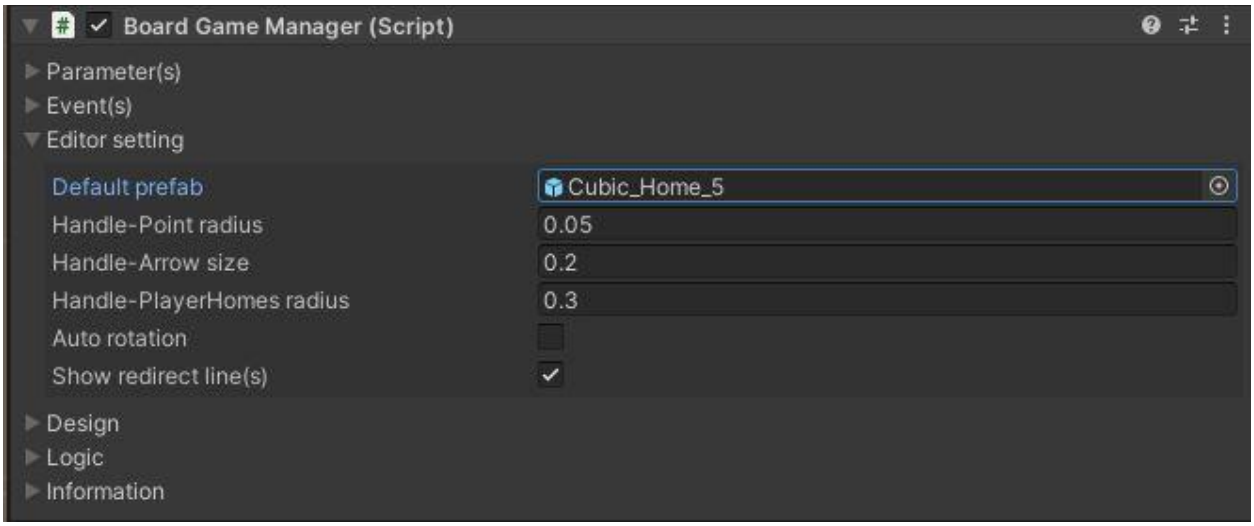
Editor setting

Here we have Editor setting`s variables.

Table 5 - BoardGameManager (Editor setting)

Name	Description
Default prefab	Set a default prefab to auto assign prefab mode in create new home
Handle-Point Radius	The radius of points on screen
Handle-Arrow size	The size of arrow that show the game board path
Handle-Player Homes Radios	The size of player(s) home radius
Autorotation	It manages the prefab rotations automatically
Show redirect lines	Show the redirect line from each home that redirect player to another home

Figure 7 - BoardGameManager inspector (Editor setting)



Design

Here we have Editor Design`s variables. In this section we have two separate options. Design Points and design Player Homes. It be can changed by changing 'Editor mode' property.

Points are the nodes that each player(s) pieces placed in that nodes positions. Each node contains its prefab (like cube or anything else) that can be rendered in game engine and player(s) can see that correctly.

Player homes are the unique node for each player that its piece(s) position(s) at the getting start.

Table 6 - BoardGameManager (Design)

Name	Description
Free movement	Enable free home(s) moving
Reset vector.Y	Reset all home(s) heights
Scale coefficient	Scaling coefficient
Scale -	Decrease designed map distance
Scale +	Increase designed map distance
Buttons	
Square	Create a square shape for board home (s)
Rectangle	Create a rectangle shape for board home (s)
Circle	Create a circle shape for board home (s)
Ellipse	Create a ellipse shape for board home (s)
Diamond	Create a diamond shape for board home (s)
Triangle	Create a triangle shape for board home (s)
N-Gon	Create a N-Gon shape for board home (s)
Line	Create a Line shape for board home (s)
Reset point(s)	Remove all points and board home(s)
Reset Player Home(s)	Remove all Player home(s)

Figure 8 - BoardGameManager inspector (Design - Point)

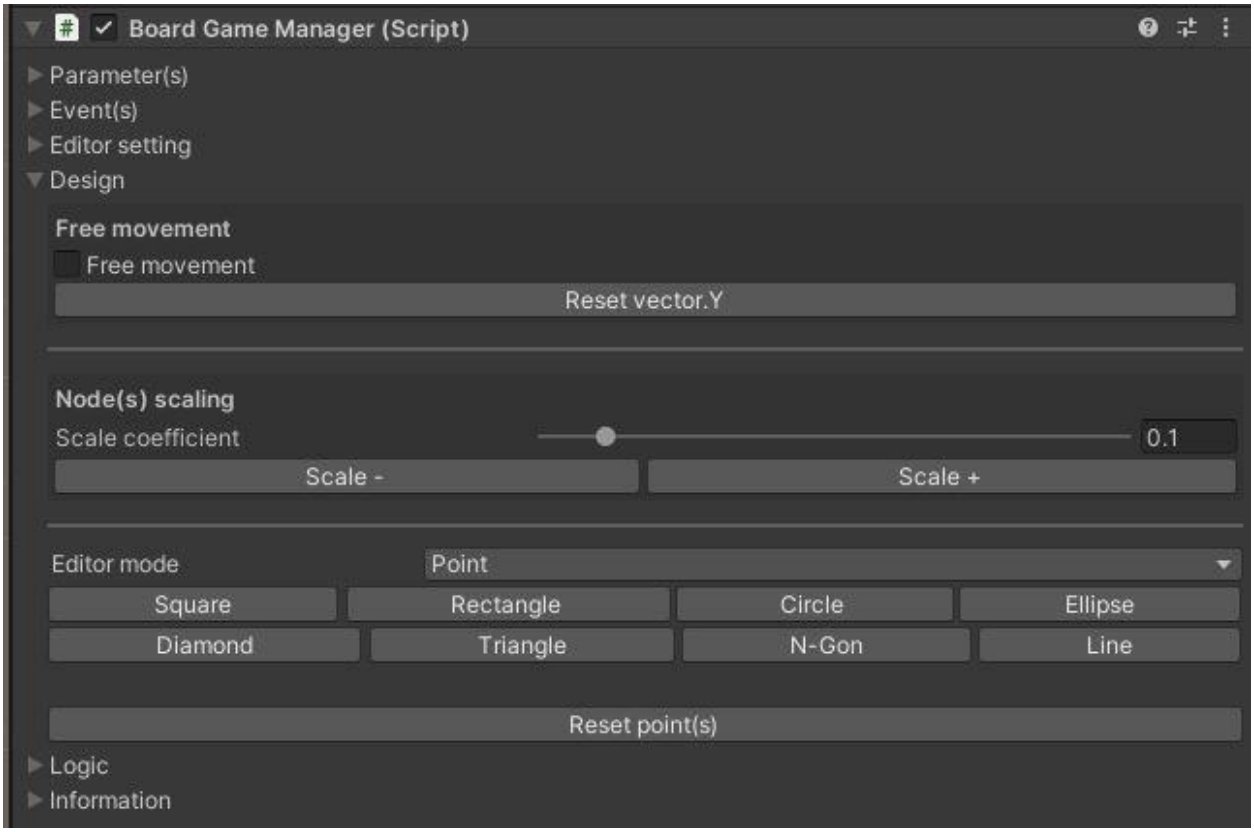
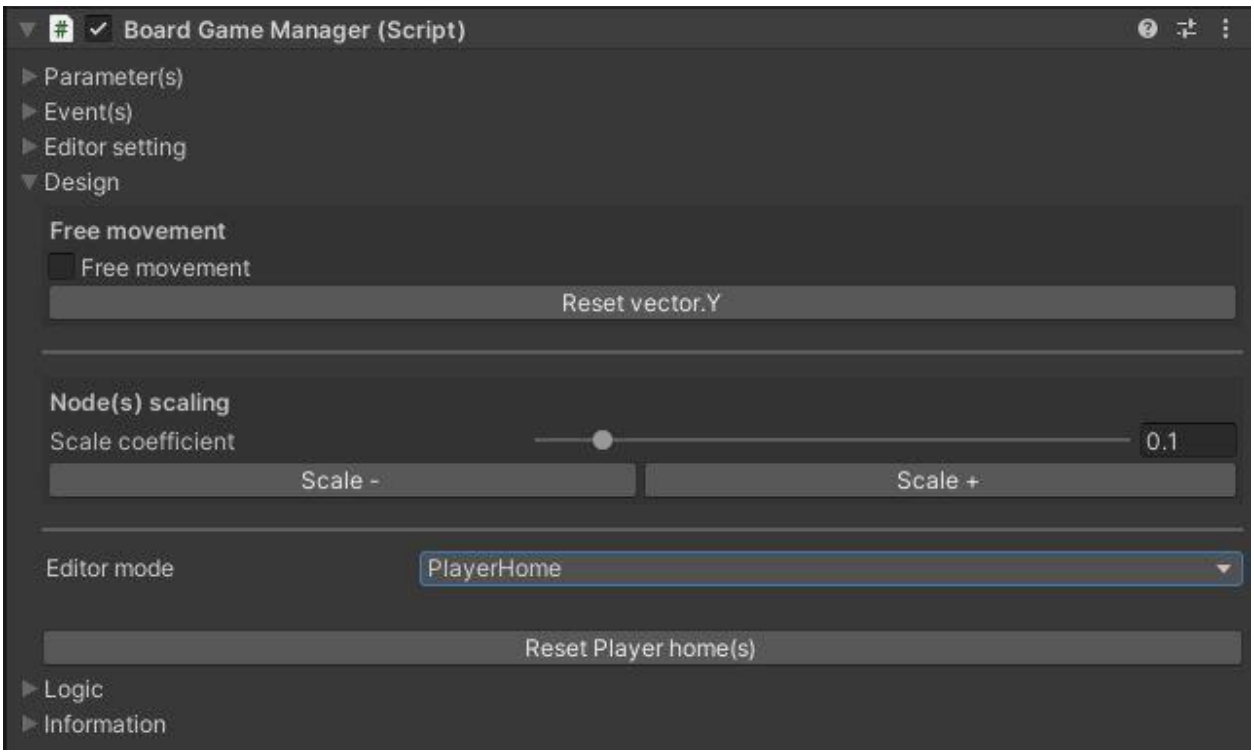


Figure 9 - BoardGameManager inspector (Design - Playe home)



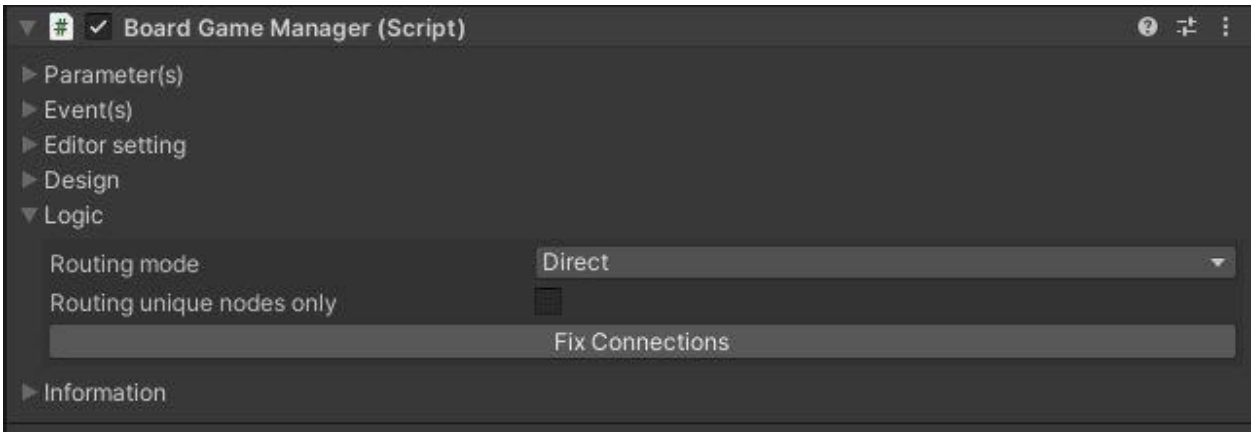
Logic

Here we have Editor Logic`s variables.

Table 7 - BoardGameManager (Logic)

Name	Description
Routing mode	• Direct: forward routing (follow green arrows) • Reverse: backward routing (follow red arrows)
Routing unique nodes only	It means each routing path with has more than one unique node, removed. It`s not necessary option but in some game designs can be useful
Fix connections	In map design you can fix your node connections with this option

Figure 10 - BoardGameManager inspector (Logic)



Information

Here we have Editor Information`s variables.

Table 8 – BoardGameManager (Information)

Name	Description
Point(s) count	The count on board home(s)
Player Homes(s) count	The count on board player home(s)

Figure 11 - BoardGameManager inspector (Information)

Board.cs

The main function of this script is managing the dice(s) over board. All dices will detect each other automatically (if you have more than 1 dice in your game).

Path: Dice Board Game - Starter Kit\Scripts\Board.cs

Table 9 - Board options

Name	Description
Collision object tags	Dices have physics effect over any game object with this tags
Rolling threshold	If dice rolling is more than the threshold, the system alarms programmer by log warning.
Dice(s) auto-detection	Auto dice(s) detection if exist
Dice	Array of dice(s)
State	Board state: • Null • Ready • Rolling • Finish

Figure 12 - Board inspector options

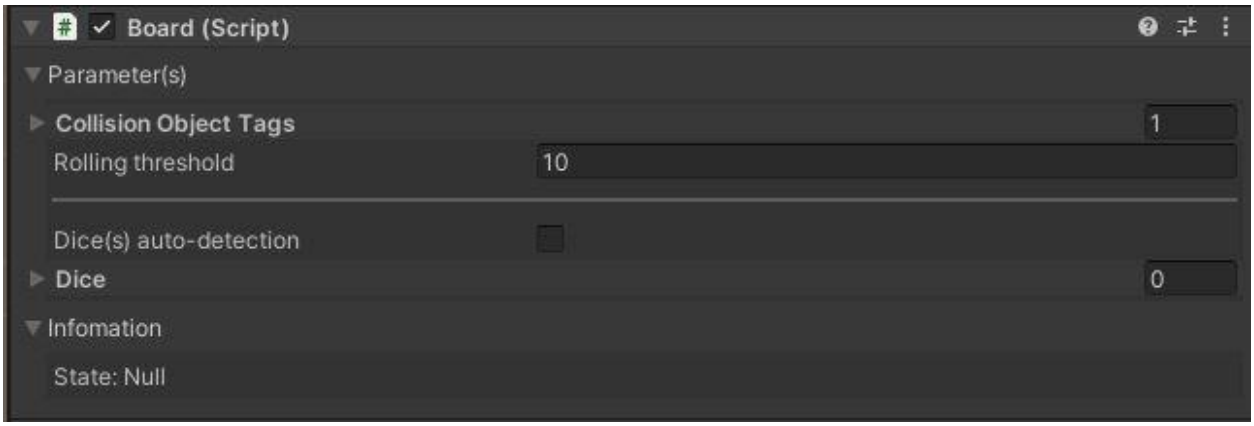


Table 10 – Board.cs Functions

Function	Description
RollDices()	Rolling dice(s)
ResetDices()	Reset dice(s) positions and values
getDicesValues()	Return all dice(s) values as array
onRollingOverFlow()	Override able function for managing over flow rolling

PlayerManager.cs

PlayerManager is the manager for all player(s) in game. It's the player script that manage each player. System is handling many scripts automatically.

Path: Dice Board Game - Starter Kit\Scripts\PlayerManager.cs

Player.cs

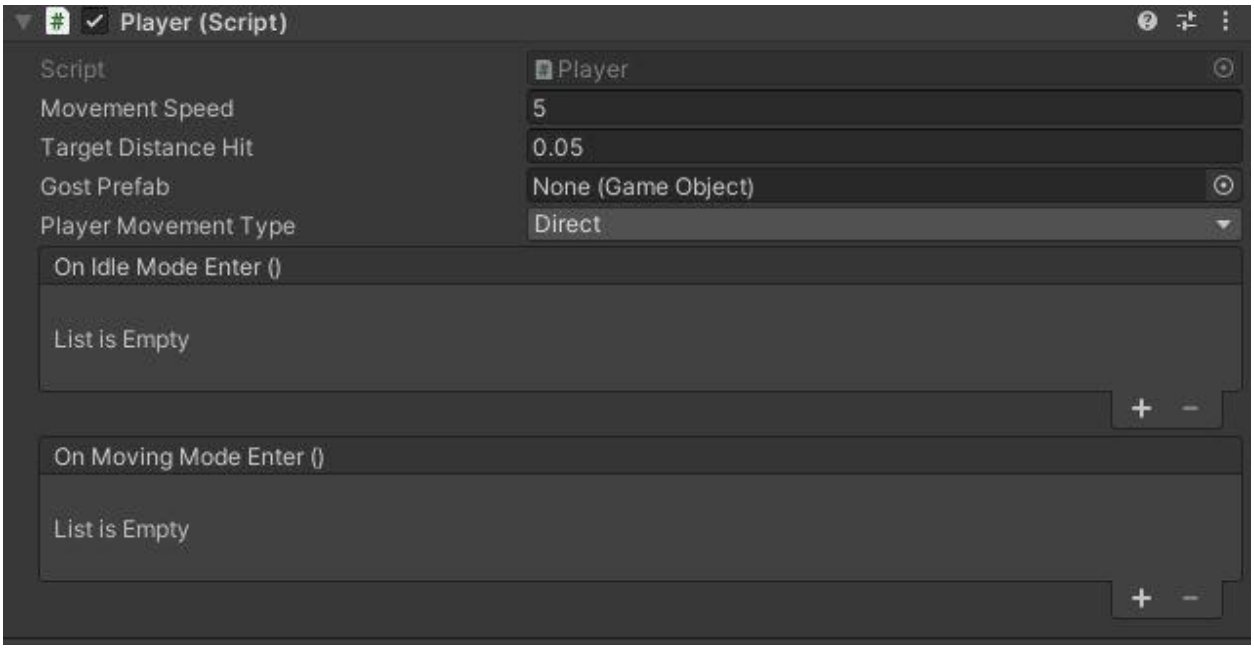
The Player script is handling each piece in game. Each player can have deferent option(s) from others.

Path: Dice Board Game - Starter Kit\Scripts\Player\Player.cs

Table 11 - Player options

Name	Description
Movement speed	The speed of movement
Target distance hit	The threshold of distance that player hit the target
Player Movement type	The type of player moving between board home(s)
On Idle Mode Enter	The unity event when player is entering to idle mode
On Moving Mode Enter	The unity event when player is entering to moving mode

Figure 13 - Player inspector options



Note:

If player movement speed is set to high value and target distance hit is very low, the player will never get the target. It`s not bug, be careful.

Table 12 - Player.cs functions

Function	Description
hitIndex	the array of board home indexes that player hit them in each roll
currentPositionIndex	the position of current board home index
diceValues	return the array of dice values for this player
positionIndex	the data store of board home indexes
GOTO	make player to move its current position to new home position
GOTO_Immediatly	force player to move to new home position
GOTO_CalculatedIndex	get the home nodes and create movement path from them
CalculatePositionIndex	calculated hitIndex value and new position index

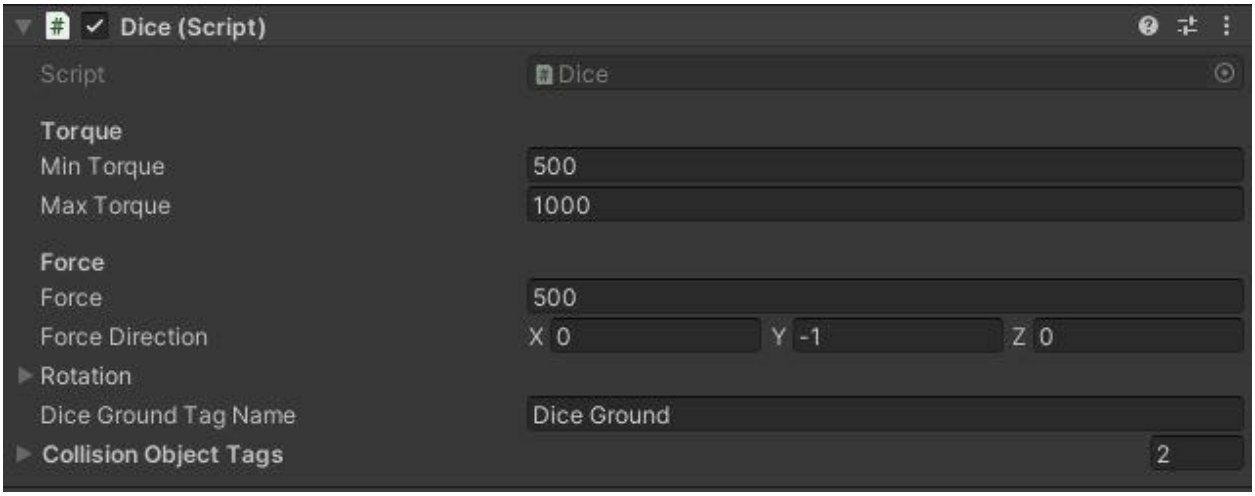
Dice.cs

This script manages the dice by managing the dice side(s) and timing for rolling dice or gathering the correct dice side value.

Table 13 - Dice options

Name	Description
Min Torque	The minimum value of dice torque
Max Torque	The maximum value of dice torque
Force	The directional force in rolling time
Force direction	The fore direction in rolling time
Rotation	The rotation in rolling time
Dice Ground tag name	The ground tag name the dice side will stay on that and create dice value
Collision object tags	Each game object with these tags has physical effect on dice(s)

Figure 14 - Dice inspector option



How to create and design my dreams?

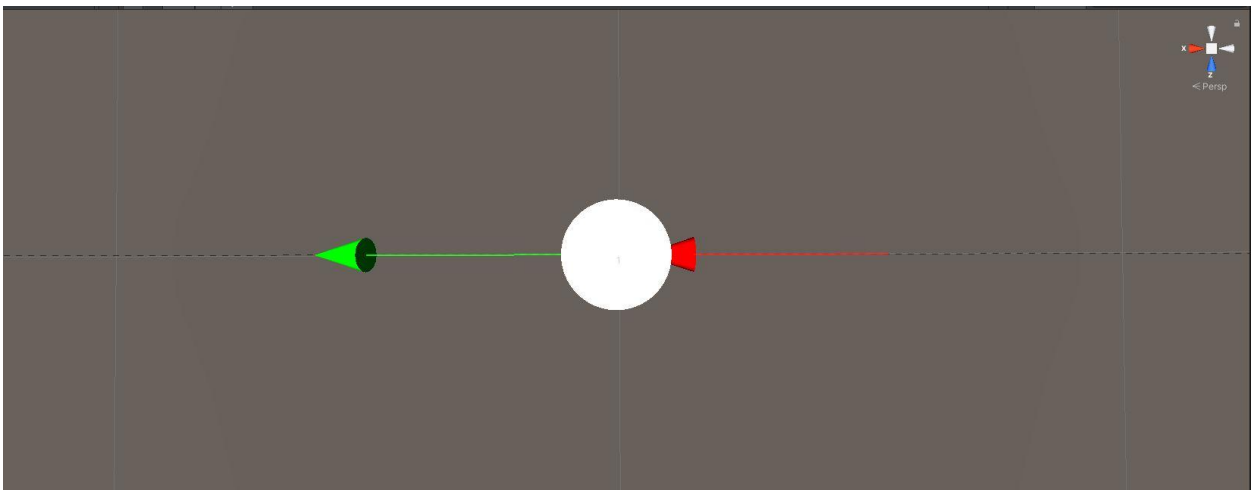
First use menu options and add Game Manager – read "How to use menu?" section – then you can draw your map.

Node

In Game Manager inspector, collapse Design > Editor mode > Point. Point is position of each node. Node is more than position and can include many function and logics.

The empty Node (home) without any prefab look like the blew image. **Red arrow** means incoming path from other node and **Green arrow** means outgoing path from current Calibri (Body).

Figure 15 - Simple Node



Note:

Indexes start from zero.

Each home node can support multi incoming and multi outgoing paths.

In this figures, nodes haven`t any prefab. Player can change this logical node(s) with any prefab.

Node options

For editing each node (Element options), just mouse over the target node and press **Ctrl + RMB** to open editor window.

Table 14 - Node options

Name	Description
Index / count	Index of selected board home / count of all board home(s)
Node type	Type of selected node • None • Reset point • Redirect point • Interrupt point
Hosting type	Type of node reaction when pieces are entering • Enable • Disable • Enable and don't forward
Prefab	The home board prefab
On player stay event	This event will invoke when player reaches this home index
Assign prefab for all node(s)	Assign current prefab for all other board home(s)
Remove node	Remove current board home
<<	Go to previse board home
>>	Go to next board home

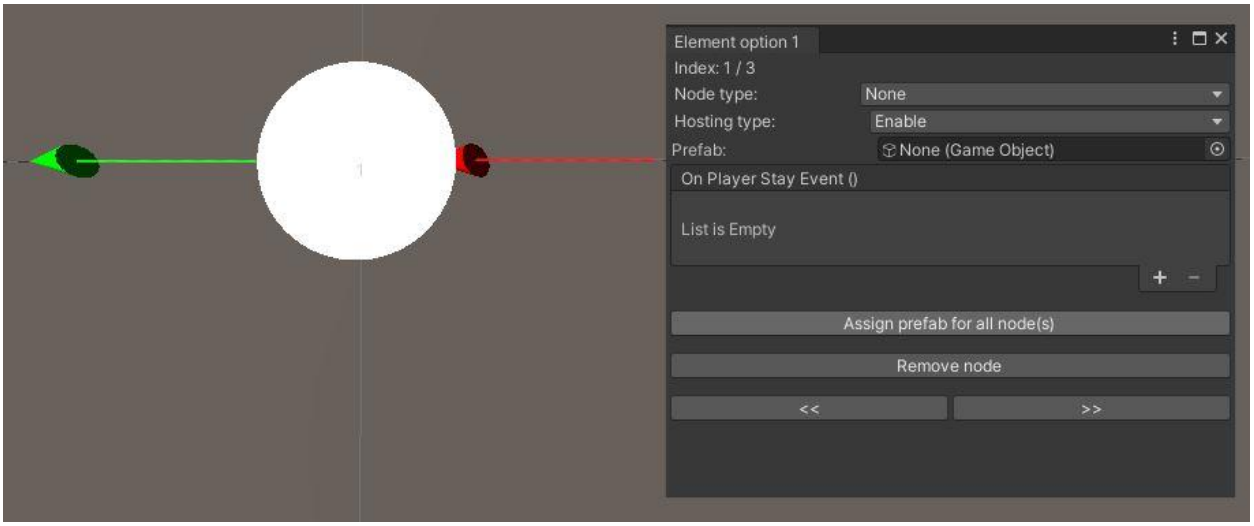
Hosting type

- Enable: node is enable.
- Disable: node is disable.
- Enable and don`t forward: node is enable but don`t allow the piece to go forward or backward.

None type

Node with **none** type is a regular node. It`s follow normal logic instructions.

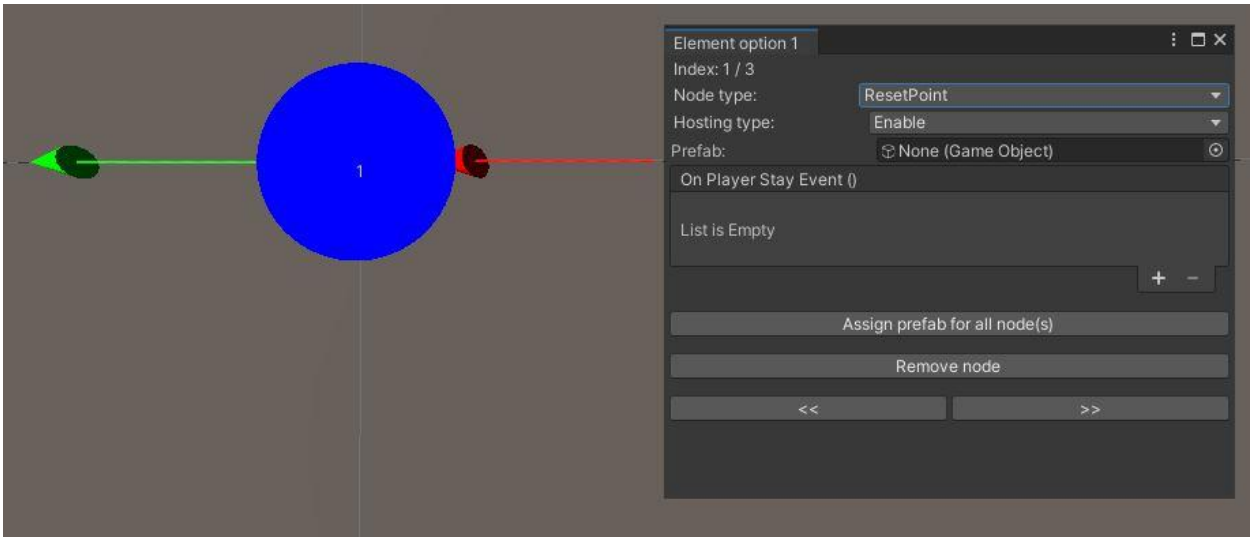
Figure 16 - Node (type: none)



Reset point type

Node with **reset point** type be a final landing home, game piece redirect to its start game position.

Figure 17 - Node (type: ResetPoint)



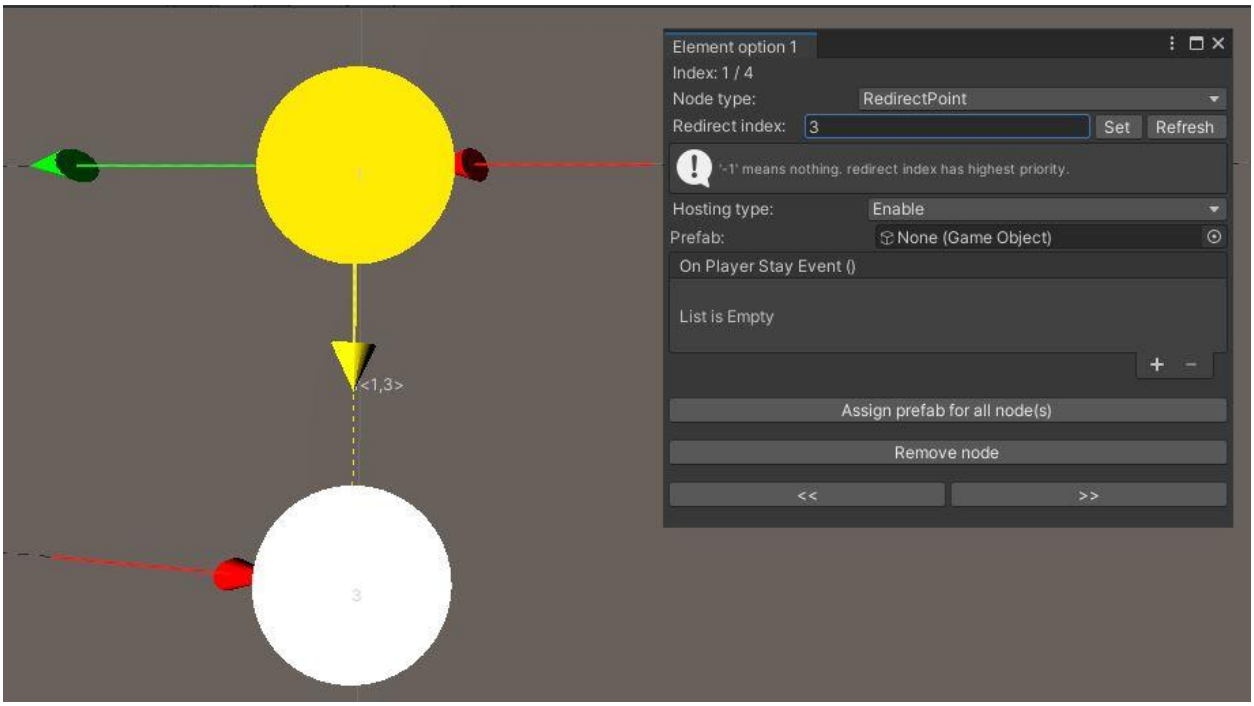
Redirect point type

Node with **redirect point** type is looking like the reset point but player can set the redirect node.

Table 15 - Node (type: RedirectPoint) options

Name	Description
Redirect Index	It means when player sit in this home and have no more moves then player auto redirect to chosen index board home. -1 means no redirect

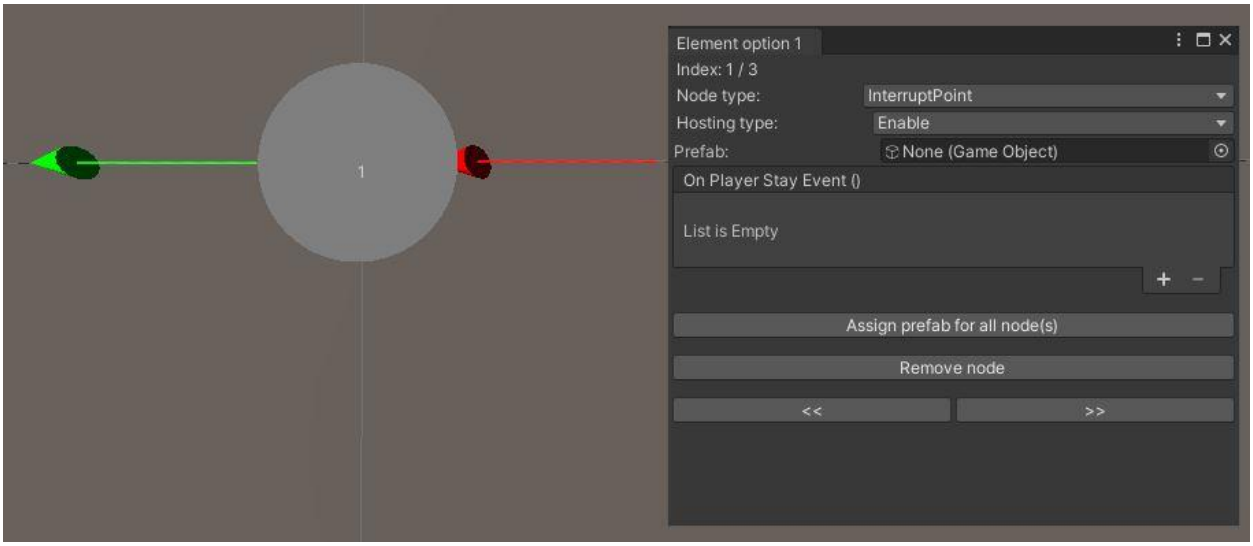
Figure 18 - Node (type: RedirectPoint)



Interrupt point type

Node with **interrupt point**, when piece is stay on this node cause to active interrupt event and functions.

Figure 19 - Node (type: InterruptPoint)



Look at this code:

```
BoardGameManager boardGameManager;

private void Update()
  {
    if (boardGameManager.hasInterrupt())
    {
      //TODO: do some action
      int playerIndex = boardGameManager.PlayerHomeIndex;
      int playerPieceIndex = boardGameManager.playerHomeCandidateIndex;
      //TODO: reset board interrupt
      boardGameManager.resetInetrrupt();
    }
  }
```

Note:

In interrupt mode you must reset interrupt with resetInterrupt() function.

In 'Editor mode' set to "**PlayerHome**":

Player home is the place than all game piece from each unique player will be managed.

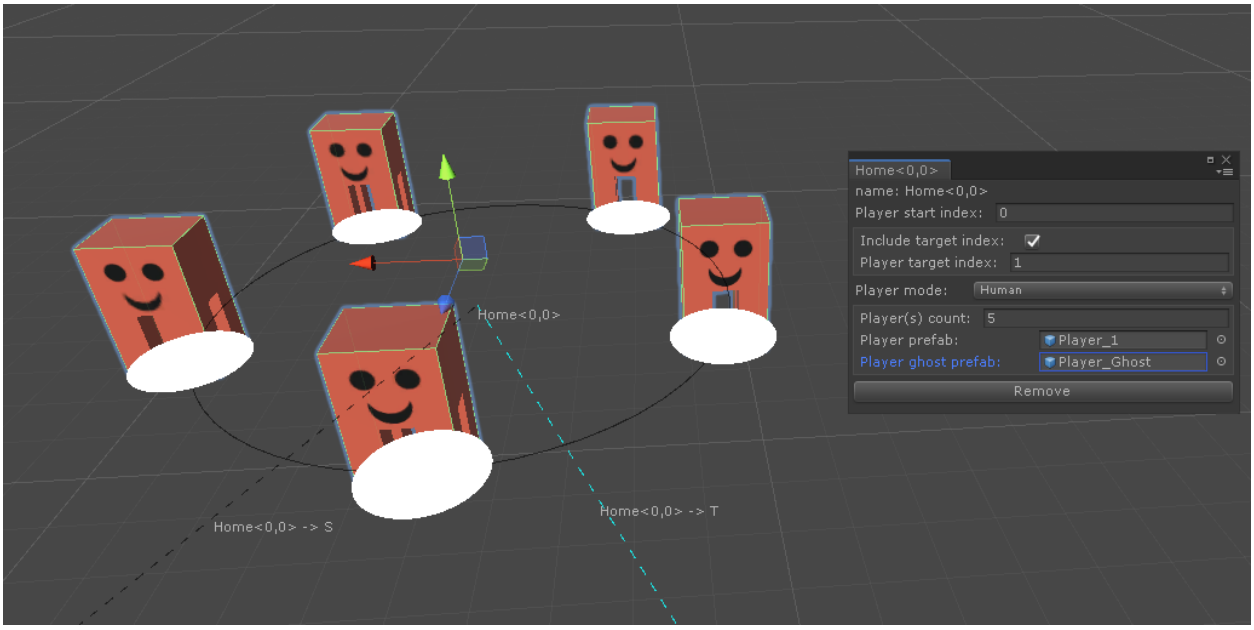
Player Home

In Game Manager inspector, collapse Design > Editor mode > Player home. Player home is position of each player. Player home is more than position and can include many function and logics. For editing each Player home (Element options), just mouse over the target and press **Ctrl + RMB** to open editor window.

Table 16 - PlayerHome options

Name	Description
Name	The player home name
Player start index	The point index that player home redirected new piece s to that position
Include target index	Check it true if these players logic cycle has target home
Player target index	The point index that players target home is
Player mode	In human mode the human is playing and in CPU mode the AI playing
Players(s) count	The count of pieces of that player can be have
Player prefab	The piece s prefab
Player ghost prefab	The ghost prefab

Figure 20 – PlayerHome and its editor



Note:

In editor player homes be labeled like Home<i,j> that ‘i’ is player home index and ‘j’ is point index that player home redirected new pieces to that position.

The Player prefab and ghost prefab can be changed for each deferent piece

Other scripts

DiceManager

This script will manage dice(s). waiting for rolling dices be over (finish) then gather all dice values and return them to BoardGameManager script.

DiceManagerEditor

The editor for DiceManager.

DiceSide

Each dice includes some sides. This script will handle all sides. If any side touch the grand and stay on ground, report the position to Dice script.

PlayerHome

The Player home manager. Its store some information.

PlayerGhost

The Player ghost manager.

ElementNode

This script stores basic board home data.

ElementNodeCreator

The BoardGameManager is inheritance from this script.

ElementOptionNodeEditor

The editor for ElementNode board home script.

LineShape

The editor for handling the pattern of board home(s) as line shape.

CircleShape

The editor for handling the pattern of board home(s) as circle shape.

SquareShape

The editor for handling the pattern of board home(s) as square shape.

RectangleShape

The editor for handling the pattern of board home(s) as rectangle shape.

EllipseShape

The editor for handling the pattern of board home(s) as ellipse shape.

DiamondShape

The editor for handling the pattern of board home(s) as diamond shape.

TriangleShape

The editor for handling the pattern of board home(s) as triangle shape.

PolygonShape

The editor for handling the pattern of board home(s) as N-Gon shape.

Extensions

The extensions functions for mange other scripts.

MainMenu

The main menu item functions for mange other scripts.

LineMaker

The script include function to create line.

CircleMaker

The script include function to create circle.

SquareMaker

The script include function to create square.

RectangleMaker

The script include function to create rectangle.

PolygonMaker

The script include function to create N-Gon.

EllipseMaker

The script include function to create ellipse.

DiamondMaker

The script include function to create diamond.

Sample Code

You can find more sample code and programming in action in below path:

Path: Dice Board Game - Starter Kit\Scene\Sample

Contact us

You need help? Or want to improve this code?

If you need any further assistance, please contact us.

unrealisticarts@gmail.com