Welcome to Connection 2D Documentation. Below you will find a brief explanation of the important scripts and how the game actually works and how to customize it.

This Asset has no platform specific codes so it should work for all platforms it's only tested on Windows and Android, If you face any problems or need further explaining on a specific part of script that I didn't explain please feel free to email me at unitydevrealgear@hotmail.com, and lastly I would like to apologize for my English.

GameManager.cs

The GameManager.cs handles everything about the game, linking the dots, adding dots, removing, enable and disabling the dots, etc.

Script Inspector	
Dots XY	This integer indicates the number of dots that will be created in the X and Y Axis, Example if it's 5 you'll get an 5 x 5 = 25 (Dots) Mobile devices with screens with or less then 720 pixels width I recommend you to use 10 if higher you can use higher
Cur Radius XY	This float determines the current width and height of a dot by checking if the amount of dots will fit in the screen width (will be explain further at the method section)
Max Radius XY	This float is the Max Radius a Dot can be if the amount of dots fit in this will be default, else it will smaller (will be explain further at the method section)
Canvas	This Component is the Canvas where the Grid will be inside, and also to get the screen width and height
Grid Overlay	This is an empty object that's only use for touch drag or mouse drag events. Using just one events for dragging is easier than using multiple events for each and every dot
Grid Parent	This is an transform for where the dots will be instantiated
Dot Prefab	This is the dot object that will be instantiated (we'll look later into how to customize it change sprites or animations)
In Edit Mode	This bool is automatically set by checking if it's still in the editor mode, this will not be enabled if the Script Level Editor is not attached or disabled
Offset Y Axis	If this bool is enabled the dot's will be having an Up and Down offset pattern
Connector Color	This is only used when you're adding a new connector, it will automatically detect dot that's not occupied
Gradient Top Color, Gradient Bottom Color	These can only be used in DemoLevelEditorScene To Change the background gradient

Now that's all public inspector variables explained we can head into the script public and private methods.

Script Public Methods	
Start	This method will delete all the dots from list's and from the Grid Parent Transform that was mentioned in the table above, after this it will create a new set of dots using the variables above
Update	This method only updates every connector script
OnMouseDown	This method is invoked every time the screen is touched or clicked and it will go through every connector script checking if its click and will set the index of the current for loop
SetRadius	This method sets the current radius. Example The screen width is 500 pixels and the dot's Max Radius XY is 50, the DotsXY = 75. 500/75 = 6.6 it's less the the Max Radius so the Cur Radius XY = 6.6 – 5% the 5% is the offset
GetNearesetDot	The method with one argument, gets the dot that's the nearest to the mouse position The method with three arguments, gets the dot that's the nearest to the current dot that's dragged and in the direction of the mouse

Connector.cs

The Connector.cs script handles all the linking, unlinking, connecting, disconnecting of dot's, connecting to portals, or dot's that accepts multi connectors, etc.

Script Variables	
m_Dots	This is a List of dot's class, each time the connector links to a dot, the old dot will be added to the list and the new dot will be the m_ConnectorDot
m_ConnectorDot	This is a dot that's the current connector, it changes each time a dot is linked or disconnected
m_Index	This integer is used later on when you clicked on a dot that was previously linked using this connector the connector should return to that dot as the connector
m_MousePosition	This 2 dimensional vector is the mouse position in viewport
m_DotPosition	This 2 dimensional vector is the current connectors local position
MouseCanvasPosition	This 2 dimensional vector is the mouse position converted to the canvas position(will be explained further on Utils script section)
Distance	This float is shows the distance between the m_DotPosition and MouseCanvasPosition
m_GameManagerInstance	This component is an instance of the component GameManager.cs

Now that's all public inspector variables explained we can head into the script public and private methods.

Script Public Methods

Update	This method only updates the mouse position
IsClicked	This method checks did the mouse clicked this connector and returns a bool
OnDrag	This method is invoke while the mouse is dragging this connector, if the method is invoked it will get the nearestDot in the dragging direction then Connect to it if its not equal to null
ConnectDot	This method links/unlink a dot that's been passed on from the OnDrag method
ResetPosition	This method is invoked by clicking on a dot that's linked using this connector it will be invoked with one argument which is it's index when added in the list m _Dots variable, it will unlink/disconnects all dots that's above is index and dot at its index will be the connect, Eg Connector index = 15 Argument index = 3 (means the third dot is clicked) Disconnects all dots that's at index higher then argument index and Connector = the third dot

Dot.cs

This script Dot.cs it will handle everything about the dot when it's active, inactive, disabled, enabled, linked, unlinked, the curren connector, etc.

Script Variables	
dot	This scripts gameobject
Sprite	This object refers to the Sprite object in the dot object
Fade	This object refers to the Fade object in the dot object
Portal	This object refers to the Portal object in the dot object
Ring	This object refers to the Ring object in the dot object
Multi	This object refers to the child object of the Sprite object
Lines	This array of objects refer to an arrays of line objects inside Lines object
tempLine	This transform refers to the tempLine in the dot object
m _CircleEnabled	This Sprite refers to the Circle-Enable Sprite inside the folder Resources/Sprites
m _CircleDisabled	This Sprite refers to the Circle-Disable Sprite inside the folder Resources/Sprites
m _CircleRing	This Sprite refers to the Circle-Ring Sprite inside the folder Resources/Sprites
activeScale	This 2 dimensional vector is the scale of an active dot
inactiveScale	This 2 dimensional vector is the scale of an inactive dot
m _CurrentColors	This array of Colors is the colors for each connector that has been linked with this dot
m _CurrentLinkedDots	This array of objects is the objects for each connector that has been linked with this dot
portal	This bool defines weather this dot is portal or not
multiConnector	This bool defines weather this dot accepts multi connectors or not

linked	This bool defines weather this dot is linked or not
endOfPortal	This Dot is only important if the current dot is a portal and it must not be null if this is a portal dot
m _CurIndex	This integer is the current index if of how many dots are linked with this one
m _MultiIndex	This integer is always zero if the dot is not a multi connector else it should be 2, it hasn't been tested for more then 2 multiConnectors
GetDotObject	This object returns this dot object
localPosition	This 2 dimensional vector returns this dot object's localPosition
isPortal	This bool returns this portal bool
isActive	This bool returns this active bool
isMultiConnector	This bool returns this multiConnector bool

Now that's all public inspector variables explained we can head into the script public and private methods.

Script Public Methods	
DisconnectDot	This method is invoked when this dot is being disconnect
ConnectDot	This method is invoked when this dot is being initialize as the new connector
LinkDot	This method is invoked when a dot is linked with another dot
SetAsMult	This method can be invoked when you want a specific dot to be a multiConnector
SetAsPortal	This method can be invoked when you want a specific dot to be a portal
Script Private Methods	
RotateLine	This method rotates a specific Line at the current index (m _CurIndex) to look at m _CurrentLinkedDots at the current index (m _CurIndex)
SetColor	This method sets the color of the current Sprite, Ring, Multi the color is from m _CurrentColors at the current index (m _CurIndex)
SetLineColor	This method sets the color of Lines at the current index (m _CurIndex)

Utils.cs

This script is an internal static class which contains helpful methods that is re-usable everytime.

Script Public Methods	
GetChildrens	This method for both objects and transforms , it collects all the objects/transforms inside the value object/transform then add them to list which is later Converted to an array and returned
RemoveChildrens	This method destroys all objects inside the value object
MousePositionToCanvas	This method Converts mousePositionViewPort to CanvasPixelPosition Example. Canvas Width & Height = 500 (500x500)

	MouseViewPort = 0.5, 0.5 As we know pixel in Texture2D Starts from the bottom corner 0, 0 but our rect transform is aligned in the center which is its 0,0 so it's width starts from -250 to 250, Height starts from -250 to 250 tempX = MouseViewPort.x * Width (0.5 x 500) = 250 tempY = MouseViewPort.y * Height (0.5 x 500) = 250 Our mouse is in the center but our temp Position is in the corner now we just remove half of the width and height newX = tempX - (Width/2) = (250 - (500/2)) = 0 newY = tempY - (Height/2) = (250 - (500/2)) = 0 Now our new Position is perfectly following the mouse The rect width & height should always match the screen size to avoid complications and also devices below 400x750 Maximum dots are 10x10 if the pixels is higher it can do to 50x50 without dot dragging errors.
SetWidthHeight	This method sets the new width and height of a RectTransform component

LevelCreator.cs

This script is only working in the Unity Editor and its usage is to save levels that you're creating inside a text file located in Resources/Texts/Levels.txt.

Script Inspector	
GameManager	This is the script component of main GameManager.cs
Level	This class is the current Level that will be saved
DotsXY	This is the amount of dots that will instantiated over the x and y axis, it can be changed inside the GameManager.cs Inspector variables after that click reset level for changes to apply
RadiusXY	This is the radius of the dots can be changed in the cur and max radius, if the screen width is less then the amount radius needed it will be decreased
Selected Dot	This dot shows the information of the current selected dot, and once a dot is edited it cannot be edited twice you'll need reset the level
Levels Text File	This TextAsset is the text file of levels I recommend to leave it as it is, if you don't know much about text files if you wish to change it, also change the locations inside the script class Level , Load method
Is Adding Portal	This bool is enabled if you're adding portals to any dot
Editing Dot Buttons (These Buttons can only be clicked if you selected a dot and the dot color is cyan)	
Add Connector(Button)	Before clicking, set the connector color inside GameManager.cs on top of this script component then click the button, you should see after you've set the current selected dot as connector and when selecting a other dot the color will appear white, it doesn't affect the GamePlay Scene
Add Multi Connector(Button)	This Button will make the current to allow up to 2 connectors to connect

Disable Dot(Button)	This Button disable the dot, dot size will be small sprite will be circle not ring, color will be white faded
Remove Dot(Button)	This Button will make the current dot disappear
Saving Loading Levels Buttons	
Save Level	This Button will save the current Level into the text file in order of other levels
Reset Level	This Button will reset the current Level

This all variables that are public in the inspector, I'll go over the Scenes now and Explaining what are the scenes usages.

Scenes	
DemoGamePlayScene	This is should be the default GamePlay Scene, it loads all levels
DemoLevelEditorScene	This scene should only be use inside the unity editor, if you want to edit the level, play the scene to create a new level

Customize Dot

Add new Sprite

To simply add a new Sprite it's easy all you have to do is have a square image (8x8,16x16,32x32,64x64,256x256,512x512, etc….) and set that Image Texture Type to a Sprite and Replace it with the Sprite you want to change inside Resource/Sprites.