

WAYPOINT INDICATORS

C# NAVIGATION SYSTEM FOR UNITY

USER GUIDE 1.2.0

Outline

Quick Start

The Waypoint Inspector

How it Works

New Project Setup

Tips to Get Started

Removing Game Objects and their Waypoints

Adding Custom Fonts

Converting Images to Sprites

Customized Console Warnings

FAQ

Contact

Quick Start

Adding the Script to your Project

1. Open the “*Scripts*” folder found on the root level (Waypoint Indicators/Scripts)
2. **Select both the folder titled “*Editor*” and the script named “*Waypoint_Indicator.cs*”**
3. Copy and paste both items into your project
4. *NOTE: If you already have a folder titled “*Editor*” in your project, then paste the contents from this asset’s “*Editor*” folder into your project’s “*Editor*” folder.

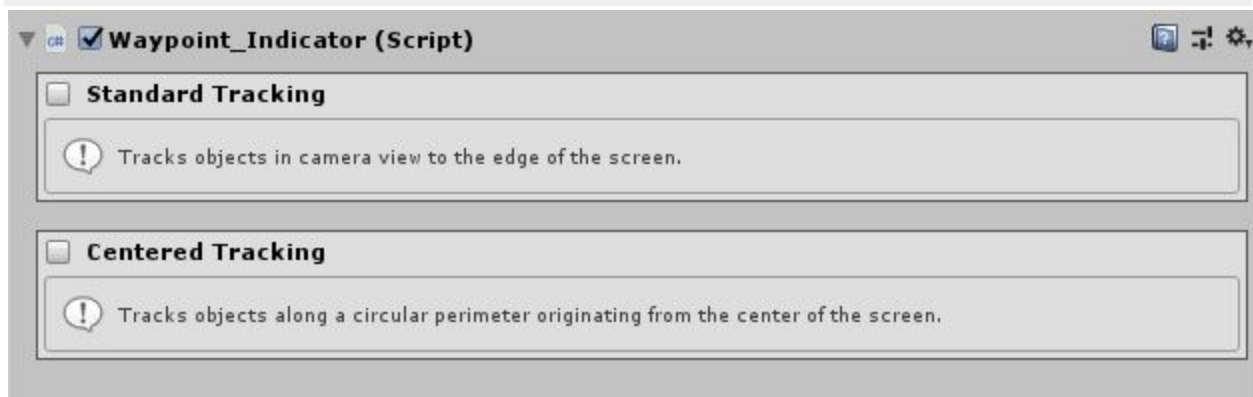
Adding the Script to a Game Object

1. Select Game Object
2. Click “Add Component”
3. Search for: “Waypoint_Indicator”
4. Click and start customizing

The Waypoint Inspector: Waypoint_Indicator.cs

NOTE: Waypoint Indicators is linked to a custom editor script called *Waypoint_Indicator_Editor.cs*. This keeps the inspector organized and must reside inside of a folder labeled “Editor” to work. The screenshots below are based off of that custom inspector.

Script Fully Collapsed



Both tracking types can be active at once for combined customization.

Standard Tracking

Tracks objects within camera view to the edge of the screen. Tracking will continue to follow off-screen objects by sliding along the screen edges.

Centered Tracking

Tracks objects along a circular edge from the center of the screen. An example of this is Far Cry's enemy detection system.

Standard Tracking (Expanded)



Parent Options

The container for your sprite, prefab and text indicator objects. Expand to reveal options.

Sprite Indicator Options

Images/icons used for indicating your game object with a texture type Sprite (2D and UI). Expand to reveal options.

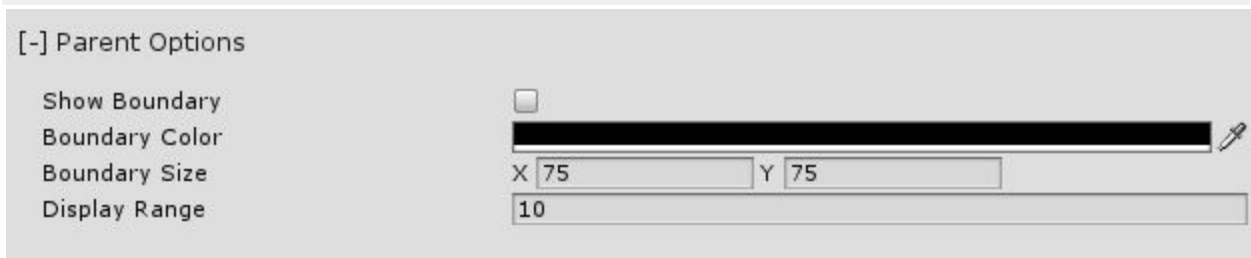
Prefab Indicator Options

Must be an empty game object created in the Canvas with a nested child game object containing a Raw Image component. Add an Animator component for movement. Expand to reveal options.

Text Options

TextMeshPro (TMP) ready text fields for displaying custom text and distance. Expand to reveal options.

Parent Options



Show Boundary

This acts as the waypoint's radius. It shows/hides the screen edge detection boundary box. When the edges of this box meet the edge of your screen, it is considered "off-screen". Turn this option on to visually see when your UI elements are being considered on and off screen. Icon

and Text elements can be moved independently from this box. The boundary box cannot be disabled, only hidden.

Boundary Color

Adjusts the fill color to easily display against various color backdrops.

Boundary Size

Adjusts the width and height of the boundary box. This will not affect the size or positions of the children Icon and Text elements.

Display Range

This is the distance between the main camera and game object at which to hide/display the indicator. Higher values will keep the indicator up longer as the camera moves further away.

Sprite Indicator Options

[-] Sprite Indicator Options

☒ Enable Sprite

Depth

Off Screen Rotates ☐

Sprite On-Screen

Sprite

Color

Size

Position X Y

Rotation

Fade with Range ☐

Scale with Range ☐

Hide ☐

Sprite Off-Screen

Sprite

Color

Size

Position X Y

Rotation

Fade with Range ☐

Scale with Range ☐

Hide ☒

Enable Sprite

Adds or removes the icon UI element completely from the Canvas. This is a performance saving option for Indicators that do not require an Icon element. Checking this enables Sprite options.

Depth

Depth control for indicator display. Use this to stack your sprite, game object or text indicators in any order you like. 0 represents the very bottom of the stack and will be covered by indicators with higher depth numbers.

Offscreen Rotates

Enables the off screen icon to rotate, point and follow the offscreen Game Object.

Sprite On/Off-Screen

Sprite

The sprite image displayed while the target game object is on-screen, this replaces the Off Screen image.

Color

Tints the sprite through a color multiplier. For best results, use solid white sprites for full color control. Shades of grey will result in darkened areas of the selected color. Black areas of the sprite will remain black. The effect is similar to drawing on an image with multiple magic markers.

Size

Resizes the On/Off Screen image, bigger values make for a larger icon.

Position

Moves the icon independently along the X and Y axis for precision positioning.

Rotation

Alters the rotation angle of the on screen image.

Fade with Range

Fades on-screen icons based off of the parent Display Range value. The further the player is, the more transparent the icon becomes and vice versa.

Scale With Range

Scale on-screen icons based off of the parent Display Range value. The further the player is, the smaller the icon and vice versa.

Hide

Completely hides the on screen state of the icon.

(continue reading below)

Object Indicator Options

[-] Prefab Indicator Options

☒ **Enable Object**

Depth
Off Screen Rotates ☐

Prefab On-Screen

Prefab 
Color
Size
Position X Y
Rotation
Fade with Range ☐
Scale with Range ☐
Hide ☐

Prefab Off-Screen

Prefab 
Color
Size
Position X Y
Rotation
Fade with Range ☐
Scale with Range ☐
Hide ☐




See Sprite Indicator Options for descriptions

(continue reading below)

Text Options

[-] Text Options

☒ Enable Text

Depth	0
Description	Pick up
Font Face	<input type="checkbox"/> None (TMP_Font Asset) 
Font Size	15
Font Color	
Text Align	Center 
Line Spacing	20
Width	80
Height	50

Text On-Screen

Hide Desc	<input type="checkbox"/>
Hide Dist	<input checked="" type="checkbox"/>
Position	X 0 Y -16.7

Text Off-Screen

Hide Desc	<input checked="" type="checkbox"/>
Hide Dist	<input checked="" type="checkbox"/>
Position	X 0 Y 0

Enable Text

Adds or removes the text UI element completely from the Canvas. This is a performance saving option for Waypoints that do not require a Text element. Checking this box enables Text options.

Depth

Depth control for indicator display. Use this to stack your sprite, game object or text indicators in any order you like. 0 represents the very bottom of the stack and will be covered by indicators with higher depth numbers.

Description

The name or descriptive text used to call out the game object. If left empty, the description value will default to the name of your game object.

Font

Adds a font face for your text. See Adding Custom Fonts for more info.

Text Size

Adjusts how big or small your text will render. This value affects Distance Counter and Description text at the same time and cannot be changed independently at this time.

Text Color

Changes the color of your font. Colors will affect both on and off screen text states and cannot be changed independently at this time.

Text Align

Aligns both Description and Distance text to the left, center or right. Alignment is set to center by default.

Text Line Spacing

Adjusts the leading or vertical spacing between multi-lines of text.

Enable Description

Shows/hides the description text.

Text Rect Width

Width of the Rect Transform UI text object. Increase this value if your letters/words are wrapping and you'd like them all on one line.

Text Rect Height

Height of the Rect Transform UI text object. Increase this value if the height of your words have gone out of bounds due to size or character length.

Text On/Off-Screen

On Screen Hide Desc

Completely hides the on screen state of the description text.

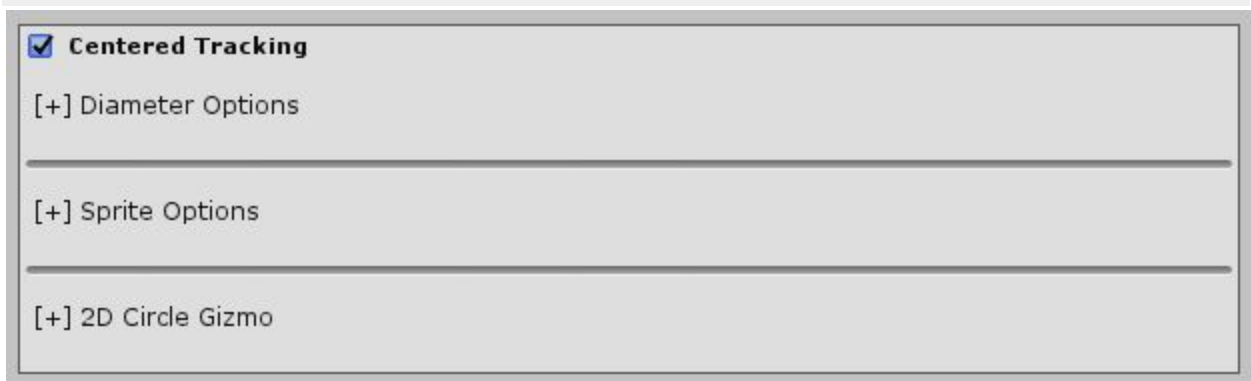
On Screen Hide Dist

Completely hides the on screen state of the distance counter text.

On Screen Text Offset

Moves the text field independently along the X and Y axis for precision positioning.

Centered Tracking (Expanded)



Diameter Options

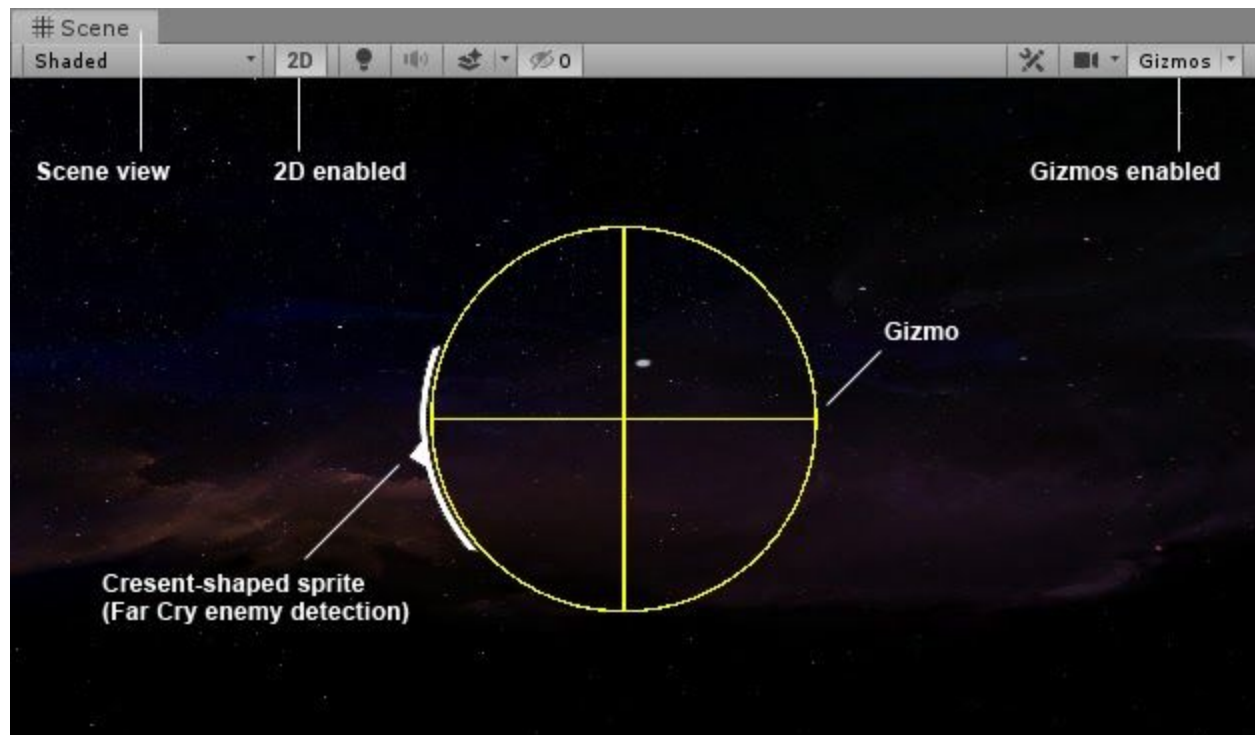
The visual representation of the sprite's circular tracking diameter. Expand to reveal options.

Sprite Options

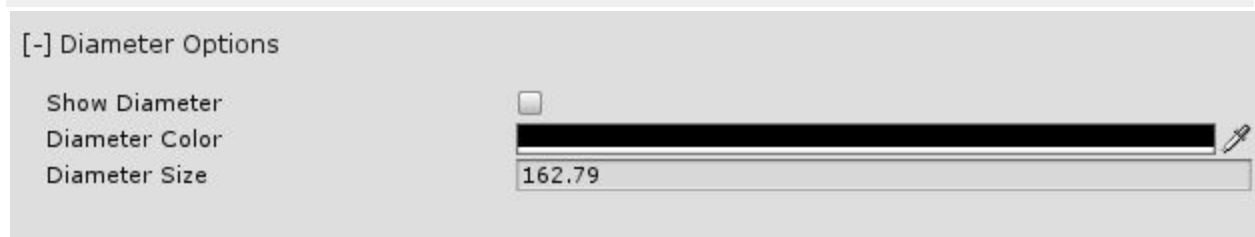
Images/icons used for indicating your game object with a texture type Sprite (2D and UI).
Expand to reveal options.

2D Circle Gizmo

A handy 2D wireframe tool that assists in matching the arc of a sprite to it's circular tracking path. Expand to reveal options.



Diameter Options



Show Diameter

Shows/Hides the visual representation of the sprite's circular tracking diameter.

Diameter Color






Adjusts the diameter's fill color to easily display against various color backdrops.

Diameter Size

Altering the diameter size will tighten or expand the tracking perimeter. Lower values shrink the diameter, bringing the sprite closer to the center of the screen as it rotates around, and vice versa.

Sprite Options

[-] Sprite Options

Display Range	50
Sprite	 white_arrow_shadow_up 
Color	 
Size	0.3
Rotation	 0
Fade with Range	<input checked="" type="checkbox"/>
Scale with Range	<input checked="" type="checkbox"/>

Display Range

This is the distance between the main camera and game object at which to hide/display the indicator. Higher values will keep the indicator up longer as the camera moves further away.

Sprite

An image that represents the indicator for your game object.

Color

Tints the sprite through a color multiplier. For best results, use solid white sprites for full color control. Shades of grey will result in darkened areas of the selected color. Black areas of the sprite will remain black. The effect is similar to drawing on an image with multiple magic markers.

Size

Resizes the sprite, bigger values make for a larger icon.

Rotation

Alters the rotation angle of the sprite.

Fade with Range

Fades sprite based off of the parent Display Range value. The further the player is, the more transparent the sprite becomes and vice versa.

Scale With Range

Scales sprite based off of the parent Display Range value. The further the player is, the smaller the sprite and vice versa.

2D Circle Gizmo

[-] 2D Circle Gizmo



Use this tool to align pointers that rotate around the exterior of a circular path. Scene View only. Must have Gizmos and 2D enabled.

Show Gizmo



Gizmo Color



Gizmo Size

100.8

Show Gizmo

When having Gizmos enabled, this circular wireframe will show up in the Scene view when set to 2D mode.

Gizmo Color

Adjusts the diameter's fill color to easily display against various color backdrops.

Gizmo Size

Adjusts the diameter of the gizmo.

How it Works

The Game Object and the UI Objects

Depending on what's enabled, Game Objects with a Waypoint script will spawn up to four UI objects into the Canvas at runtime. All four of these objects inherit their name prefix from the game object that spawned it.

1. Parent UI Object (contains the Icon and Text UI objects)
 - a. Think of this as a container object to the Icon and Text objects. When the container's edges meet that of a screen edge, it tells the Icon and Text object to perform various actions that can be customized. For example swapping an on-screen icon with an off-screen version.
 - b. The container's dimensions can be customized for greater control. For example, if your icons are running up against the edge of the screen, add padding by giving more width to the parent container and moving the icon and text closer to the center.
2. Sprite UI Object (child of the parent UI object)
 - a. This accepts any graphic converted to a Sprite (See Converting Sprites for more)
 - b. Arrows/Pointers/POI Icons/Targets are good examples of image types
 - c. See Icon Options for additional details

3. Game Object UI Object (also a child of the parent UI object)
 - a. This accepts any empty game object and all of its contents that have been created inside the Canvas.
 - b. These objects can contain animations, images, raw images, buttons, etc.
4. Text UI Object (also a child of the parent UI object)

Naming

All UI objects inherit their name prefix from the game object that it spawned from followed by a "WP" (stands for waypoint) followed by a unique ID followed by the indicator type (Sprite, Object, Text). This naming convention ensures all new indicators will have unique names despite being spawned from a game object of the same name.

Parent Object

- Sprite Indicator Object
- Prefab Indicator
- Text Indicator Object

For example, two game objects both named "Cube" with all 3 indicator types enabled would look like this in the Canvas:

- Cube-WP12345
 - Cube-WP12345-Sprite
 - Cube-WP12345-Prefab
 - Cube-WP2345-Text
- Cube-WP67890
 - Cube-WP67890-Sprite
 - Cube-WP67890-Prefab
 - Cube-WP67890-Text

New Project Setup

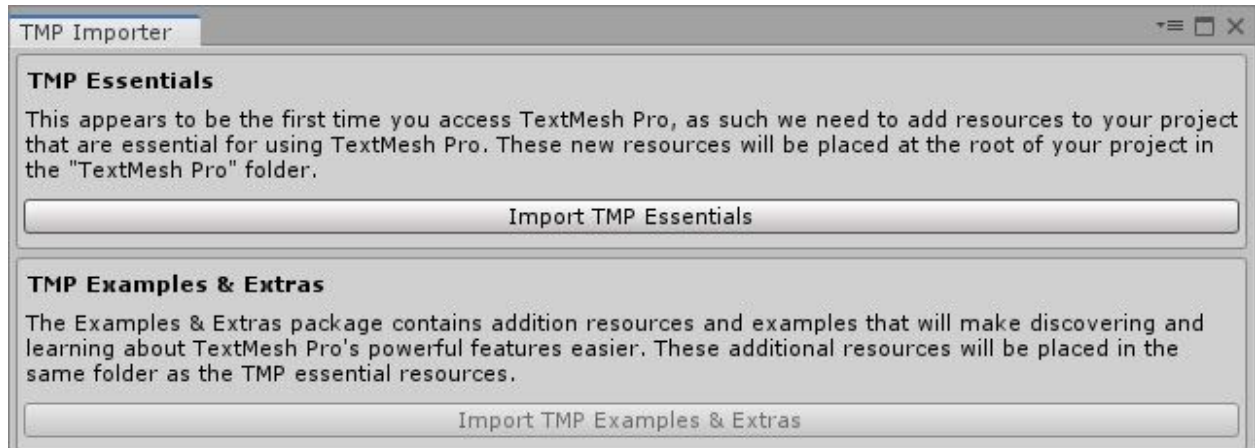
Select Your Project Resolution

This is an important step as your Canvas Reference Resolution will need to mirror this in order to track the waypoint's relationship with the edge of the screen.

From the Game tab (Window > General > Game) select the project workspace resolution (might say "Free Aspect" if this is a new project) and either select an existing resolution or declare your own by clicking the plus (+) sign at the bottom of the window..

Setup TextMeshPro (TMP)

If you try to run a new project with this script WITHOUT Text Mesh Pro, you will see this popup:



Click “Import TMP Essentials” to add Text Mesh Pro

No need to import Extras

TextMesh Pro provides improved control over text formatting and layout and this system uses it as a replacement for Unity's UI Text & Text Mesh. If you do not already have TMP Essential Resources installed, follow the instructions below:

1. Window > TextMeshPro > Import TMP Essential Resources
2. At the bottom of the popup window, click “Import”
3. For more on fonts, see “Adding Custom Fonts”

Add a Canvas

UI elements cannot display without a Canvas game object. Make sure you have one in your scene. To add a Canvas, from the top menu select: GameObject > UI > Canvas

NOTE: If you plan on renaming your Canvas to something other than “Canvas” open the Waypoint Indicator script and update the “canvas_name” value to the same name as your Canvas.

Setup Canvas

In order for the waypoint system to track accurately, it's important that your canvas is setup correctly.

To edit your Canvas, select it from the Hierarchy and find the Inspector window. If you do not see an Inspector window, click Window > General > Inspector

1. Canvas Inspector > Canvas > Render Mode > Screen Space - Overlay
2. Canvas Inspector > Canvas Scaler > UI Scale Mode > Scale With Screen Size
3. Canvas Inspector > Canvas Scaler > Reference Resolution > Same resolution as the project resolution selected in step one from the Game tab
4. Canvas Inspector > Canvas Scaler > Screen Match Mode > Match Width Or Height
5. Canvas Inspector > Canvas Scaler > Match > 0.5
6. Reference Pixels Per Inch > 100

Adding the Script to your Project

1. Due to the amount of options available, Waypoint Indicators is linked to a custom editor script (Waypoint_Indicator_Editor.cs) to keep the inspector organized.
2. Although the Waypoint_Indicator.cs will run smoothly without the custom inspector, it is recommended you use both scripts together for an optimal inspector interface.
 - a. Waypoint_Indicator.cs
 - i. This is the core asset
 - ii. Place this file anywhere you like (typically a folder titled "Scripts")
 - b. Waypoint_Indicator_Editor.cs
 - i. This is the custom inspector interface
 - ii. This file **MUST** be placed in a folder called "Editor". This "Editor" folder can sit anywhere in your project directory.

Add the Script to your Game Object

1. Select your game object and go to the Inspector
2. At the bottom, click "Add Component" and search "Waypoint_Indicator"
3. Click to add
4. Run the project
5. You should see two lines of text attached to your object.
 - a. Hello!
 - b. Object's distance from camera in meters
6. If you do not see any text, you might not have installed Text Mesh Pro or something's off with your Canvas settings. Please read above to make sure you have your project setup correctly.

Tips to Get Started

Use Empty Game Objects For Precision Positioning

Apply the Waypoint script to an empty game object, give it a unique name, then make it a child of the object you wish to track. This allows you to physically place the Waypoint where you like on the object. Waypoint scripts placed directly onto game objects will spawn at the object's local 0,0,0 origin points. Since every game object has a different origin point, this could yield unpredictable positioning results for your Waypoint.

Editing Waypoints on Prefabs at RunTime

When editing in RunTime, edit the script from the scene object directly, not the prefab resource file, your changes will not save. Instead, find and click the game object within the scene hierarchy and edit from there. When you're done, right-click the waypoint script, select "Copy Component". Stop running the project, right-click the waypoint script you just copied and select "Paste Component Values". This will make sure you don't lose your edits during runtime.

Stacking/Layering Indicators Using Depth

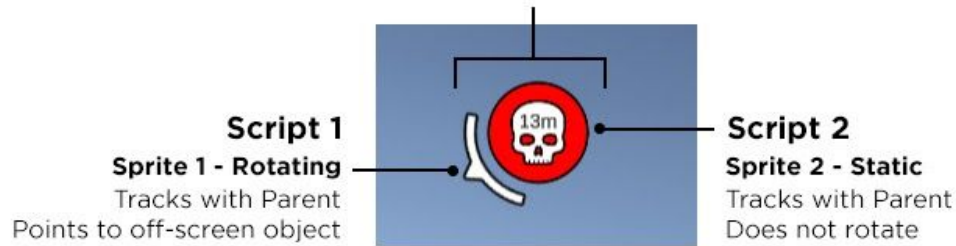
You can achieve some cool effects by placing animated Indicators behind static ones while keeping text indicators on top of everything. Use the depth field to stack your sprite, game object or text indicators in any order you like. 0 represents the very bottom of the stack while 2 is the very top.

Attach Multiple Scripts for Added Functionality

One script is good, but two can be better. Waypoint Indicators work together when multiple scripts are added to a single object.

For example, you might want to show a static sprite coupled with a rotating arrow pointing to its object location off screen. Since a sprite state cannot be both static and allow to rotate at the same time on the same script, you would use one script to handle the static sprite and the other to handle the rotating sprite. See below:

OFF-SCREEN INDICATOR USING TWO SCRIPTS



Removing Game Objects and their Waypoints

Waypoints will destroy themselves when the game object or its parent is either destroyed or deactivated. Code below:

1. `Destroy(gameObject)`
2. `gameObject.SetActive(false);`

Adding Custom Fonts

Although not needed, custom fonts will help establish your game's look and feel. Here's how:

1. Locate the Project tab (Window > General > Project)
2. Create a *FONTS* folder in your assets folder.
3. Unity supported font formats: .ttf and .otf
4. Place supported fonts into the *FONTS* folder.
5. Open Font Asset Creator: Window > TextMeshPro > Font Asset Creator
6. Click "Source Font File" to select font.
7. Click "Generate Font Atlas"
8. Click "Save"
9. Your font is ready to use

Converting Images to Sprites

Although your project contains several sprite examples to experiment with, you may want to add your own.

The instructions below assume you already have icon art saved to your project.

1. Locate your image in Unity via the Project tab (Window > General > Project)
2. Go to the Inspector Window (Window > General > Inspector)

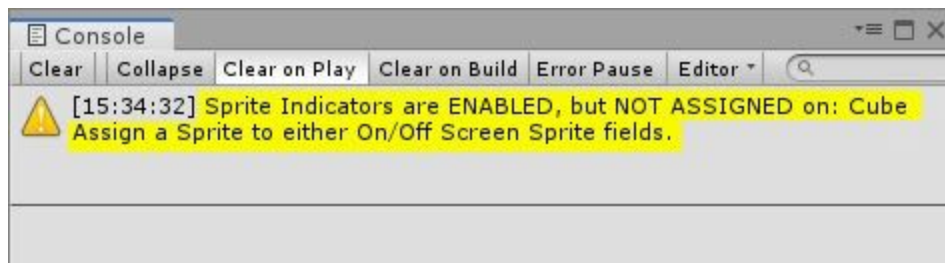
3. Click “Texture Type” at the top
4. Select “Sprite (2D and UI)”
5. Click “Apply” in the bottom right
6. Your icon is ready to use

Customized Console Warnings

If your waypoint isn’t working, check the console

In an attempt to go beyond the console log, detailed warnings for predictable situations have been added to facilitate speedier bug fixing. For example, if no sprites have been selected for the Icon, instead of seeing a white box, the console will tell you there are missing images and which game object its referring to.

Example:



FAQ

Newly added Sprite isn’t showing up during runtime

This happens when adding a sprite for the first time while the game is running. Simply reset the script by disabling and enabling it back. Also check that your camera is within DISPLAY RANGE.

Sprites, Objects, or Text aren’t showing up

Check to see that your camera isn’t out of the default DISPLAY RANGE which is 50. This can be fixed by changing the Display Range to a much larger value. Try putting in 1,000 as a starting point to see if your elements show up.

Icon Object is not rendering on screen or spawning in Canvas when “Enable Icon” is checked

You might be missing sprites. Make sure you have sprites assigned to “On Screen Sprite” and “Off Screen Sprite” in the icon section.

“New Game Object” keeps spawning over and over outside of my canvas when project starts

This happens when there is either no Canvas in your project or your project Canvas is labeled differently than what the Waypoint script is looking for. Basically the script can't find the Canvas. To fix, open the Waypoint Indicator script and update the “canvas_name” value in line 10 to the same name as your Canvas.

Changes aren't updating in real time when the project is running

Make sure you are NOT updating the PREFAB from your project directory, this will yield no results. To see changes in real time, find and click on the Game Object with the Waypoint Indicator script attached either from your Hierarchy window or your scene view and edit directly from there.

How do I control the stacking order of text versus icons

Text and Icon UIs show up in the order that they are spawned. To place a UI object on top of another, simply disable, then enable the icon you want on top. Disabling text automatically disables the description and distance fields. You will need to re-enable them when re-enabling the Text UI.

Sprites look stretched

Disable and re-enable the game object with the waypoint script attached to fix. Do this each time you swap out your sprites with another sprite of a different width and height dimensions. The sprite size is set when the icon becomes enabled. If you swap out a sprite with another sprite of different width and height dimensions, the old values will still be applied to the new sprite and stretching will occur.

Contact

Questions, Comments, Concerns?

If you're having any issues at all with this script, please contact me. I'm not happy unless you are! Also, if the script is working out for you and you're super satisfied, I'd love to hear from you!

Click the contact link below to send me a message.

peter.tracy@yahoo.com

@studio_11508 | <http://ptracy.com/> | [Feedback Form](#)