

WAYPOINT INDICATORS

C# NAVIGATION SYSTEM FOR UNITY

USER GUIDE 1.4.3

Table of Contents

[What's New](#)

[Quick Start](#)

[Canvas Render Modes](#)

[Launching the Demo](#)

[The Waypoint Inspector](#)

[How it Works](#)

[New Project Setup](#)

[Multi-Camera Setup](#)

[Tips to Get Started](#)

[Globally Hiding and Showing Waypoints](#)

[Adding Custom Fonts](#)

[Converting Images to Sprites](#)

[Customized Console Warnings](#)

[FAQ](#)

[Integrating with Curved UI](#)

[Contact](#)

What's New in version 1.4.3

1. Text RectTransform dimensions have been publicly removed in favor of an automated function that adjusts the RectTransform to fit the text size instead. Users are no longer required to adjust the Rect width and height values to accommodate the change in font sizes.
2. Clearly separated *Core* and *Demo* files so that returning users can easily opt out of the extra files that the demo scenes come with.

Quick Start (After Installing the Package)

Setting up Canvas and Camera

1. Canvas - By default, make sure your main canvas is tagged: "Canvas". You can always change this later in the "Setup Options" in the script inspector.
2. Camera - By default, make sure your main camera is tagged: "MainCamera". You can always change this later in the "Setup Options" in the script inspector.

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

Canvas Render Modes

Waypoint Indicators supports all three canvas render modes for maximum customization and flexibility as of the 1.2.7 release. The script will automatically detect the render mode of the canvas tagged in the script's [Setup Options](#).

Screen Space - Overlay

This render mode places UI elements on the screen rendered on top of the scene. If the screen is resized or changes resolution, the Canvas will automatically change size to match this.

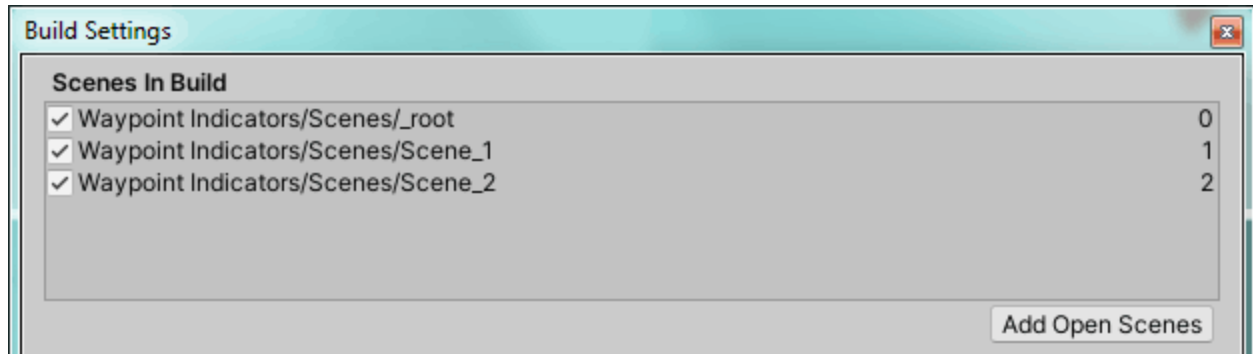
Screen Space - Camera

This is similar to Screen Space - Overlay, but in this render mode the Canvas is placed a given distance in front of a specified Camera. The UI elements are rendered by this camera, which means that the Camera settings affect the appearance of the UI. If the Camera is set to Perspective, the UI elements will be rendered with perspective, and the amount of perspective distortion can be controlled by the Camera Field of View. If the screen is resized, changes resolution, or the camera frustum changes, the Canvas will automatically change size to match as well.

World Space

In this render mode, the Canvas will behave as any other object in the scene. The size of the Canvas can be set manually using its Rect Transform, and UI elements will render in front of or behind other objects in the scene based on 3D placement. This is useful for UIs that are meant to be a part of the world. This is also known as a "diegetic interface".

Launching the Demo



If you plan on using or playing the demo in your new project, be sure to add the respective scenes to your build index under File > Build Settings...

The scenes can be located in Waypoint Indicators > Scenes

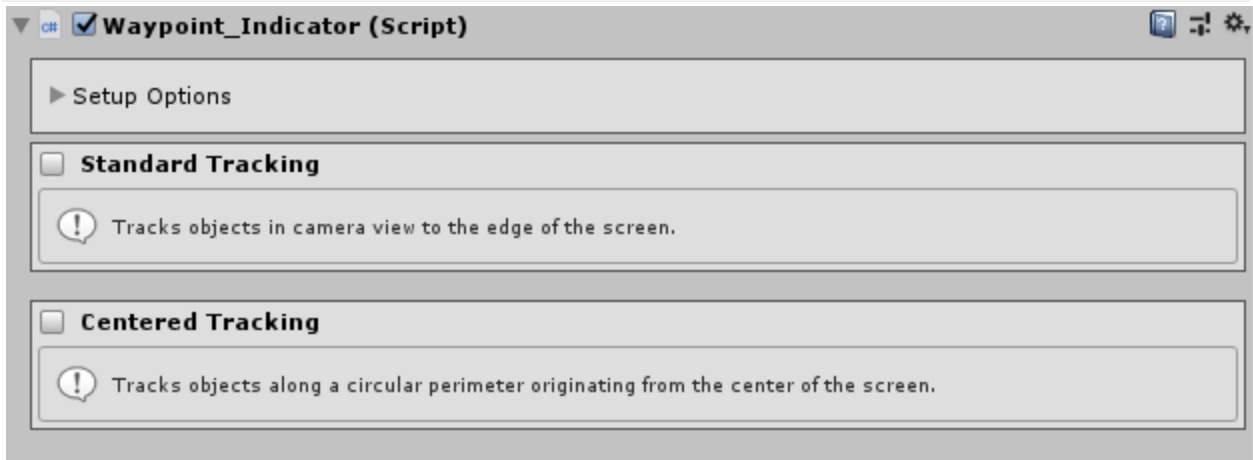
- _root
- Scene_1
- Scene_2

To avoid possible errors and scene index conflicts with the demo and your project, it is recommended to install the package into a brand new unity project.

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.

Setup Options

This is where the Canvas and Camera is linked to the script via public tag labels

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.

Setup Options (Expanded)

Waypoint_Indicator (Script)

▼ Setup Options

Canvas Tag Name: Canvas

Camera Tag Name: MainCamera

DCFT Tag Name: MainCamera

! DCFT (Distance Calculation From Target) is the game object that this game object will track its distance against. In most cases, this would be the same tag name as your main camera (above). In a 3rd person game, this might be the tag name of the player object instead.

☐ **Standard Tracking**

! Tracks objects in camera view to the edge of the screen.

☐ **Centered Tracking**

! Tracks objects along a circular perimeter originating from the center of the screen.

Canvas Tag Name

The unique tag name given to the canvas that will render the UI indicators.

Camera Tag Name

The unique tag name given to the Camera that the UI indicators will render to.

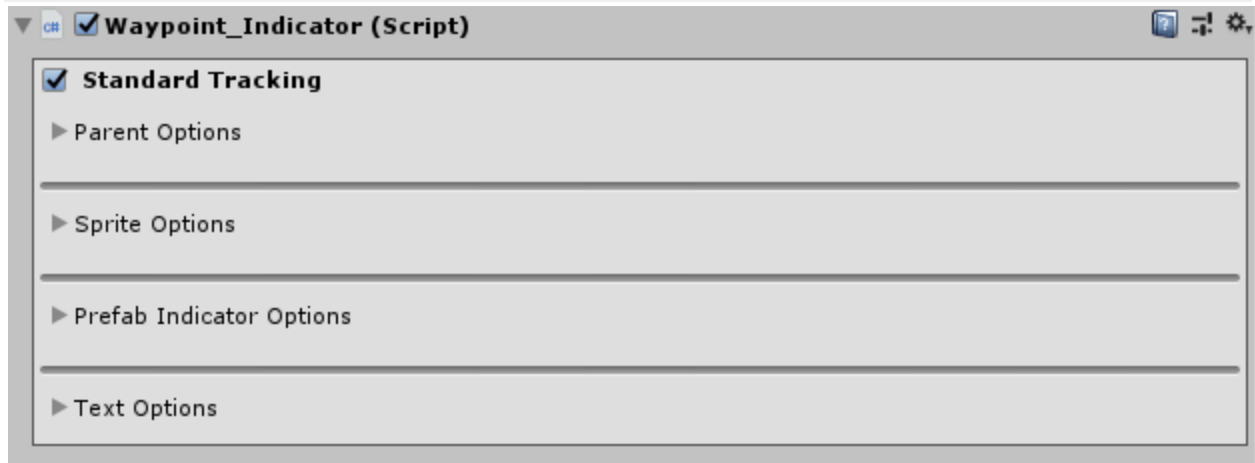
Multiple Cameras

When using multiple cameras, this avoids errors when checked. Requires the *WPI_Manager.cs* script to be included in your project. [Learn more.](#)

DCFT (Distance Calculation from Target)

The tag name of the game object that will be tracking the distance between itself and your waypoint game object. For example: Most FPS games will use the main camera to calculate the distance from what you're looking at. However in 3rd person games, you would want to use the player game object because that's now interacting with the objects in the world, so we would want to know how far things are from the character controller, not the camera.

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.

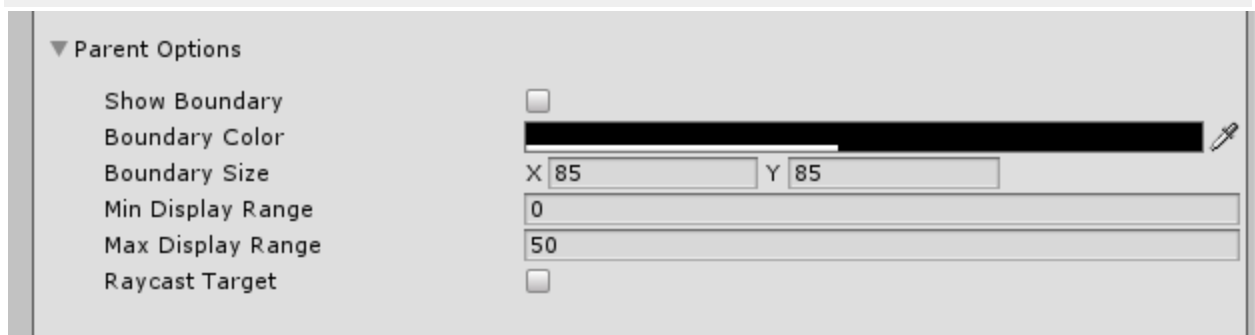
Prefab Indicator Options

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.

Min Display Range

The minimum distance between the main camera and game object at which to display the indicator. Indicators closer to the camera than this range's value will be hidden. Lower values will keep the indicator up longer as the camera moves closer to the player.

Max Display Range

The maximum distance between the main camera and game object at which to display the indicator. Indicators further away from this range will be hidden. Higher values will keep the indicator up longer as the camera moves further away.

Raycast Target

This checkbox turns on or off the Waypoint UI elements ability to block mouse clicks.

Sprite Indicator Options

The screenshot shows a UI panel titled "Sprite Indicator Options". It contains two main sections: "Sprite On-Screen" and "Sprite Off-Screen". Each section has a list of properties on the left and corresponding controls on the right. The "Enable Sprite" checkbox is checked. The "Depth" is set to 0. The "Off Screen Rotates" checkbox is unchecked. The "Sprite On-Screen" section has a "Sprite" dropdown set to "None (Sprite)", a "Color" picker, a "Size" set to 1, and "Position" fields for X and Y both set to 0. The "Rotation" is a slider set to 0. The "Fade with Range", "Scale with Range", "Reverse Scale", and "Hide" checkboxes are all unchecked. The "Sprite Off-Screen" section has identical controls, all of which are also set to their default or off states.

▼ Sprite Options

☒ Enable Sprite

Depth 0

Off Screen Rotates ☐

Sprite On-Screen

Sprite None (Sprite)

Color

Size 1

Position X 0 Y 0

Rotation 0

Fade with Range ☐

Scale with Range ☐

Reverse Scale ☐

Hide ☐

Sprite Off-Screen

Sprite None (Sprite)

Color

Size 1

Position X 0 Y 0

Rotation 0

Fade with Range ☐

Scale with Range ☐

Reverse Scale ☐

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.

Color

Tints the sprite's color and alpha. For best results, use solid white sprites for full color control.

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.

Reverse Scale

Reverses the indicator's scale function so that the further the player is, the larger the icon and vice versa.

Hide

Completely hides the on screen state of the icon.

Prefab 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 ☐

Reverse Scale ☐

Hide ☐

Prefab Off-Screen

Prefab

Color

Size

Position X Y

Rotation

Fade with Range ☐

Scale with Range ☐

Reverse Scale ☐

Hide ☐

See [Sprite Indicator Options](#) for Prefab Indicator descriptions

Text Options

▼ Text Options

☒ Enable Text

Depth: 0

Description: Hello!

Dist Increment: m

Font Face: None (TMP_Font Asset)

Font Size: 15

Font Color: [Color Picker]

Text Align: Center

Line Spacing: 20

Edge Detect Offset: X 0 Y 0

Text On-Screen

Hide Desc: ☐

Hide Dist: ☐

Position: X 0 Y 0

Text Off-Screen

Hide Desc: ☐

Hide Dist: ☐

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 Face

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

Dist Increment

Customize the distance increment displayed ex: "33m", "33ft", "33 miles", etc

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. To force a line break, type `
` just before each break.

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. Text is centered by default.

Text Line Spacing

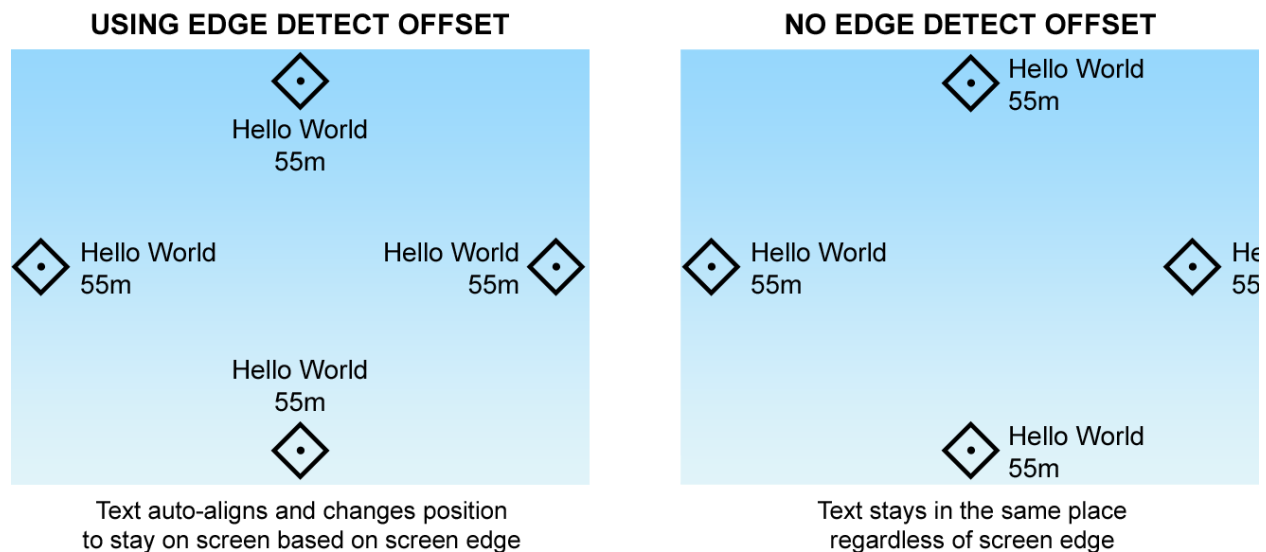
Adjusts the leading or vertical spacing between multi-lines of text.

Enable Description

Shows/hides the description text.

Edge Detect Offset

The amount of x and y offset that the off-screen text will auto align and position itself to the parent off-screen indicator to always stay visible. Set values to 0 to disable this feature.



Text On/Off-Screen

On Screen Hide Desc

Completely hides the on screen state of the description text.

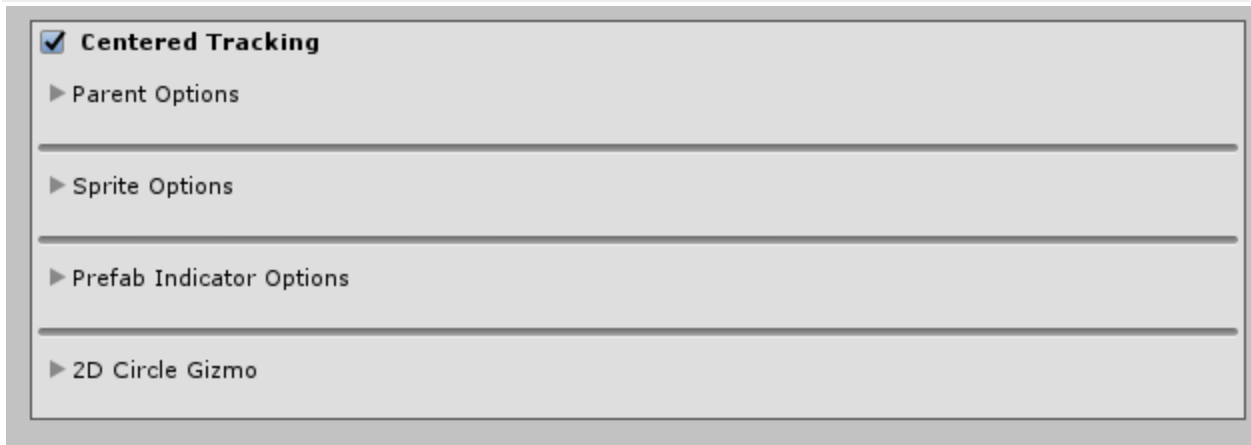
On Screen Hide Dist

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)



Parent Options

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

Sprite Options

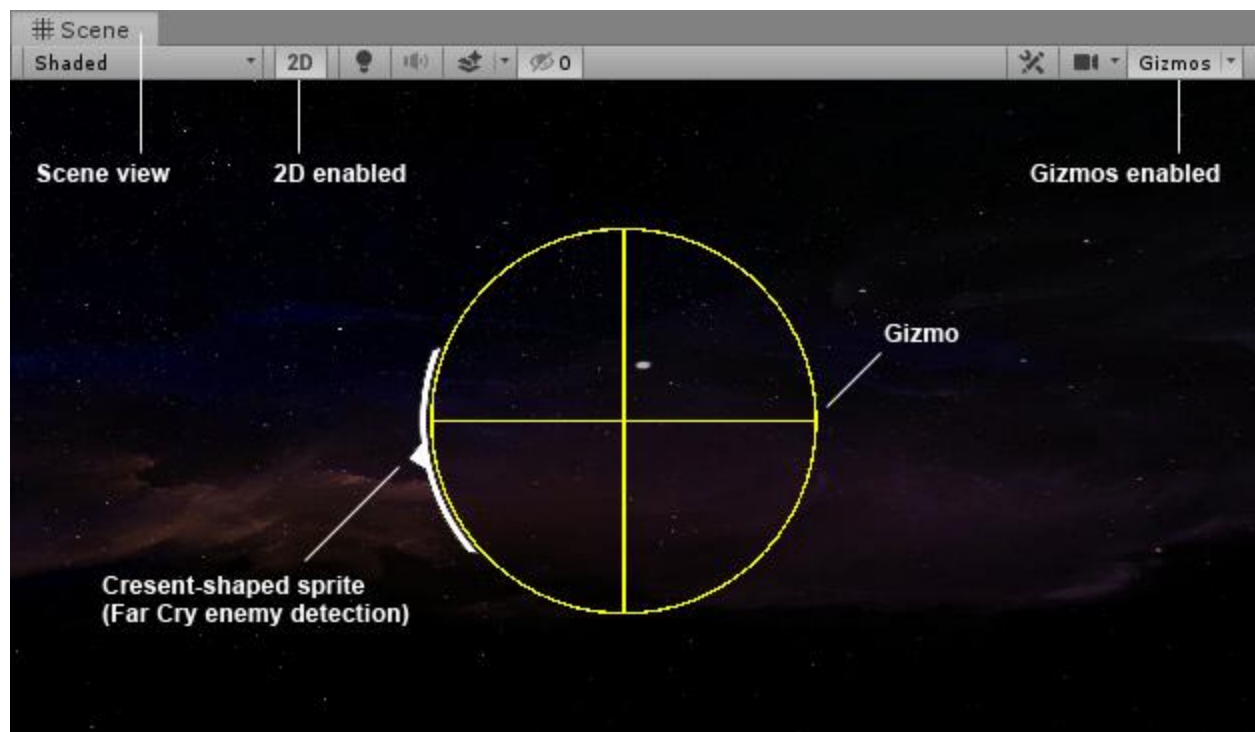
Images/icons used for indicating your game object with a texture type Sprite (2D and UI). Expand to reveal options.

Prefab Indicator Options

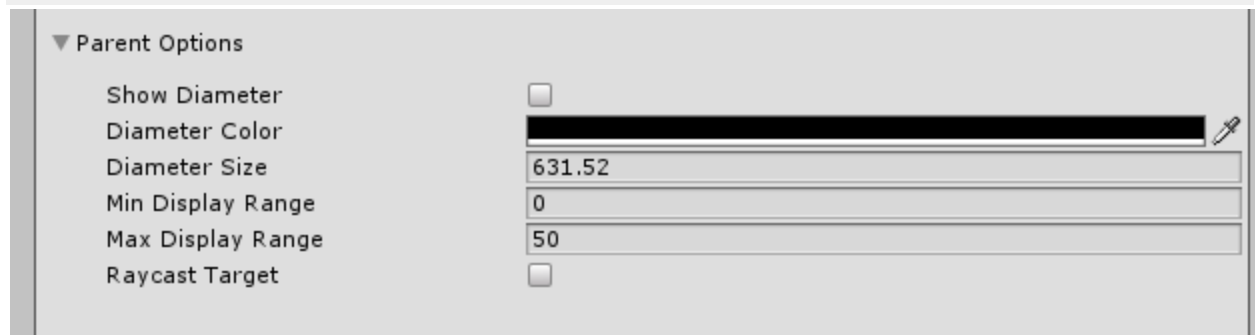
Must be an empty game object created in the Canvas with a nested child game object containing a Raw Image component. Add an Animator component for movement. Expand to reveal options.

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.



Parent Options



The screenshot shows a UI panel titled "Parent Options" with a dropdown arrow. Below the title, there are six settings:

- Show Diameter**: A checkbox that is currently unchecked.
- Diameter Color**: A color selection bar showing a solid black color, with a small pencil icon to its right.
- Diameter Size**: A text input field containing the value "631.52".
- Min Display Range**: A text input field containing the value "0".
- Max Display Range**: A text input field containing the value "50".
- Raycast Target**: A checkbox that is currently unchecked.

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.

Min Display Range

The minimum distance between the main camera and game object at which to display the indicator. Indicators closer to the camera than this range's value will be hidden. Lower values will keep the indicator up longer as the camera moves closer to the player.

Max Display Range

The maximum distance between the main camera and game object at which to display the indicator. Indicators further away from this range will be hidden. Higher values will keep the indicator up longer as the camera moves further away.

Raycast Target

This checkbox turns on or off the Waypoint UI elements ability to block mouse clicks.

Sprite Options

▼ Sprite Options

- ☒ Enable Sprite
- Depth: 0
- Sprite: None (Sprite)
- Color: [Color Picker]
- Size: 1
- Rotation: [Slider at 0]
- Fade with Range: ☐
- Scale with Range: ☐
- Reverse Scale: ☐
- Hide On-Screen: ☐
- Hide Off-Screen: ☐

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.

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. Use the alpha slider to adjust opacity.

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.

Reverse Scale

Reverses the indicator's scale function so that the further the player is, the larger the icon and vice versa.

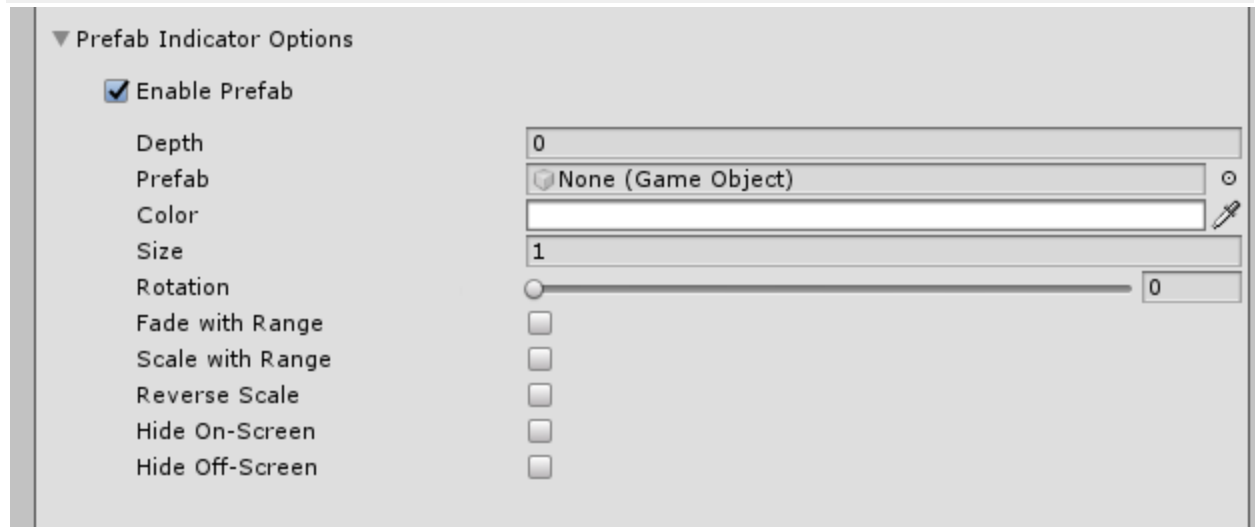
Hide On-Screen

Completely hides the indicator when the game object is on-screen.

Hide Off-Screen

Completely hides the indicator when the game object is off-screen.

Prefab Indicator Options

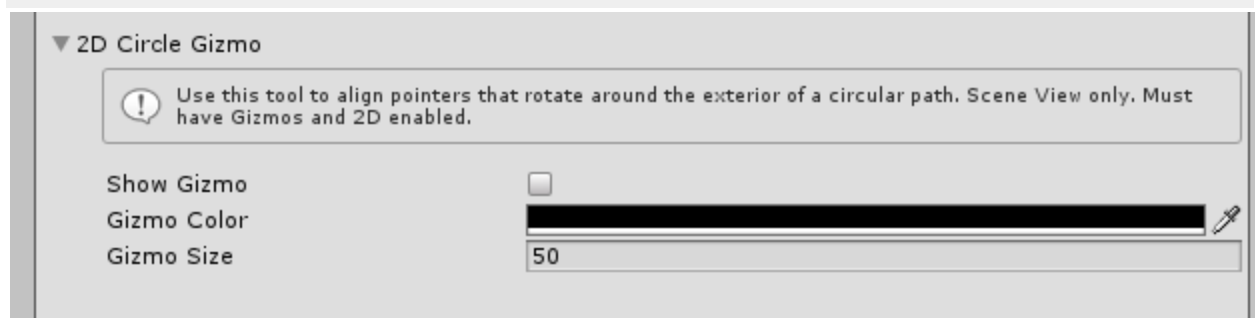


The screenshot shows the 'Prefab Indicator Options' panel. It features a list of settings on the left and their corresponding controls on the right. The 'Enable Prefab' checkbox is checked. The 'Depth' is set to 0. The 'Prefab' dropdown is set to 'None (Game Object)'. The 'Color' field is empty with a color picker icon. The 'Size' is set to 1. The 'Rotation' is set to 0 with a slider. The 'Fade with Range', 'Scale with Range', 'Reverse Scale', 'Hide On-Screen', and 'Hide Off-Screen' options are all unchecked.

Property	Value
Enable Prefab	<input checked="" type="checkbox"/>
Depth	0
Prefab	None (Game Object)
Color	
Size	1
Rotation	0
Fade with Range	<input type="checkbox"/>
Scale with Range	<input type="checkbox"/>
Reverse Scale	<input type="checkbox"/>
Hide On-Screen	<input type="checkbox"/>
Hide Off-Screen	<input type="checkbox"/>

See [Sprite Options](#) for descriptions

2D Circle Gizmo



The screenshot shows the '2D Circle Gizmo' panel. It includes a warning message: 'Use this tool to align pointers that rotate around the exterior of a circular path. Scene View only. Must have Gizmos and 2D enabled.' Below this, the 'Show Gizmo' checkbox is unchecked. The 'Gizmo Color' is set to black. The 'Gizmo Size' is set to 50.

Property	Value
Show Gizmo	<input type="checkbox"/>
Gizmo Color	Black
Gizmo Size	50

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

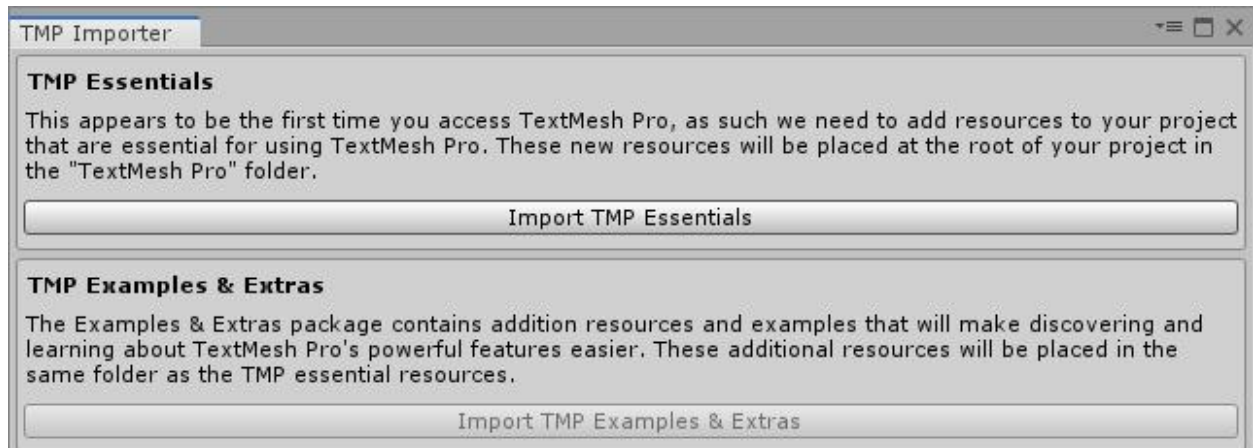
Select Your Project Resolution

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

Setup Canvas

By default, make sure your main canvas is tagged: "Canvas". You can always change this later in the "Setup Options" in the script inspector.

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

Canvas > Pixel Perfect: Disable

Most sprite elements will be scaled or rotated, or use subtle animated position or scaling, therefore it may be advantageous to disable Pixel Perfect, since the movement will be smoother without. Additionally, enabling Pixel Perfect has caused Sprites to not render at all.

Screen Space - Overlay (recommended settings):

1. UI Scale Mode: Scale With Screen Size
2. Reference Resolution: Same resolution as the project resolution selected in step one from the Game tab
3. Screen Match Mode: Match Width Or Height

Screen Space - Camera

1. Render Camera: Make sure you have a camera assigned to this slot

World Space

1. Render Camera: Make sure you have a camera assigned to this slot

Adding the Script to your Project (there are two linked scripts)

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.
3. As of version 1.4.1 Waypoint Indicators is linked to the script (**WPI_Manager.cs**) supporting multi-camera setup and future functionality.

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.

Multi-Camera Setup

WPI_Manager.SwitchCams();

WPI_Manager.SwitchCams(string [CameraTag](#), string [Distance Calculated from Target Tag](#));

Description:

Replaces the current Camera and the Distance Calculated from Target across all active Waypoints in the scene.

Requires the "*WPI_Manager.cs*" script to be in your project (does not have to be in scene or hierarchy). Be sure to click the "[Multiple Cameras](#)" checkbox in the set up options to avoid errors.

Example Trigger Script swapping between 3 cameras using Input.GetKeyUp():

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class camera_manager : MonoBehaviour
{
    //Call cameras by their game object so we can enable/disable them for this example
    public GameObject cam1;
    public GameObject cam2;
    public GameObject cam3;

    //Stores the tag name of the camera you'll be switching to
    private string newCameraTag; //Sets up the new camera tag name
    //Stores the tag name of the game object you'll be calculating distance from
    private string newDistCalTargetTag;
```

```

//Sets cam1 as the default camera on Start
private void Start()
{
    newCameraTag = cam1.tag;
    newDistCalTargetTag = cam1.tag;
    //Tells Waypoints to render to a new cam and calculate dist from a new object
    WPI_Manager.SwitchCams(newCameraTag, newDistCalTargetTag);
    cam1.SetActive(true);
    cam2.SetActive(false);
    cam3.SetActive(false);
}

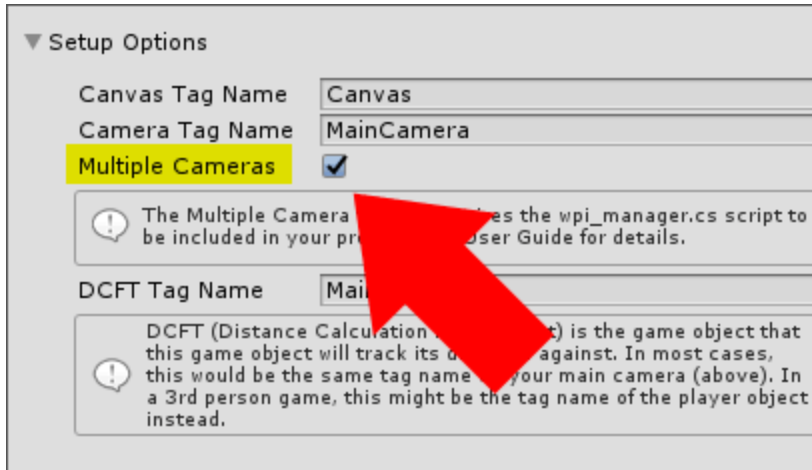
void Update()
{
    if (Input.GetKeyUp("1"))
    {
        newCameraTag = cam1.tag;
        newDistCalTargetTag = cam1.tag;
        //Tells Waypoints to render to a new cam and calculate dist from a new object
        WPI_Manager.SwitchCams(newCameraTag, newDistCalTargetTag);
        cam1.SetActive(true);
        cam2.SetActive(false);
        cam3.SetActive(false);
    }

    if (Input.GetKeyUp("2"))
    {
        newCameraTag = cam2.tag;
        newDistCalTargetTag = cam2.tag;
        //Tells Waypoints to render to a new cam and calculate dist from a new object
        WPI_Manager.SwitchCams(newCameraTag, newDistCalTargetTag);
        cam1.SetActive(false);
        cam2.SetActive(true);
        cam3.SetActive(false);
    }

    if (Input.GetKeyUp("3"))
    {
        newCameraTag = cam3.tag;
        newDistCalTargetTag = cam3.tag;
        //Tells Waypoints to render to a new cam and calculate dist from a new object
        WPI_Manager.SwitchCams(newCameraTag, newDistCalTargetTag);
        cam1.SetActive(false);
        cam2.SetActive(false);
        cam3.SetActive(true);
    }
}
}

```

To avoid debug errors, be sure to check “Multiple Cameras” in the Setup Options when using this function.



The screenshot shows a 'Setup Options' panel with several fields and a checkbox. A red arrow points to the 'Multiple Cameras' checkbox, which is checked. The fields include 'Canvas Tag Name' (Canvas), 'Camera Tag Name' (MainCamera), and 'DCFT Tag Name' (Main). There are also two informational messages: one about the 'wpi_manager.cs' script and another about the 'DCFT' game object.

▼ Setup Options

Canvas Tag Name

Camera Tag Name

Multiple Cameras ☒

! The Multiple Camera... requires the wpi_manager.cs script to be included in your project. See the User Guide for details.

DCFT Tag Name

! DCFT (Distance Calculation...) is the game object that this game object will track its distance against. In most cases, this would be the same tag name as your main camera (above). In a 3rd person game, this might be the tag name of the player object instead.

Tips to Get Started

Make sure your existing project is free of any console errors

Sometimes errors from other scripts will affect the performance of this asset. Please be sure that there are no errors prior to attempting to debug the script.

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.

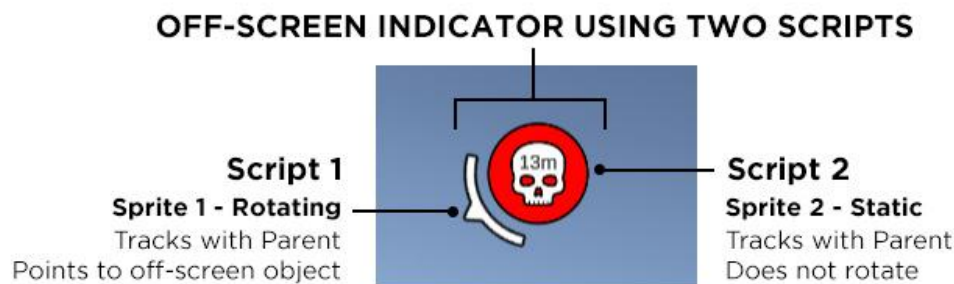
Stacking/Layering Indicators Using Depth

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:



Globally Hiding and Showing Waypoints

WPI_Manager.ToggleVisibility();

Description:

Hides or shows all Waypoints in the scene while keeping their parent game objects visible.

***Note** current and newly spawned waypoints all inherit the same state. This means when waypoints are toggled OFF, any new waypoints spawned into the scene will not display their waypoint until toggled back on and vice versa.

Requires the “WPI_Manager.cs” script to be in your project (does not have to be in scene or hierarchy).

Example Trigger Script using Input.GetKeyUp():

```
using System.Collections;  
using System.Collections.Generic;  
using UnityEngine;  
  
public class toggle_waypoints : MonoBehaviour  
{
```

```

void Update()
{
    if (Input.GetKeyUp("1"))
    {
        //Hides all waypoints if they are active, and shows them if they are inactive
        WPI_Manager.ToggleVisibility();
    }
}
}

```

NOTE: 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 *FONT*S folder in your assets folder.
3. Unity supported font formats: .ttf and .otf
4. Place supported fonts into the *FONT*S 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.

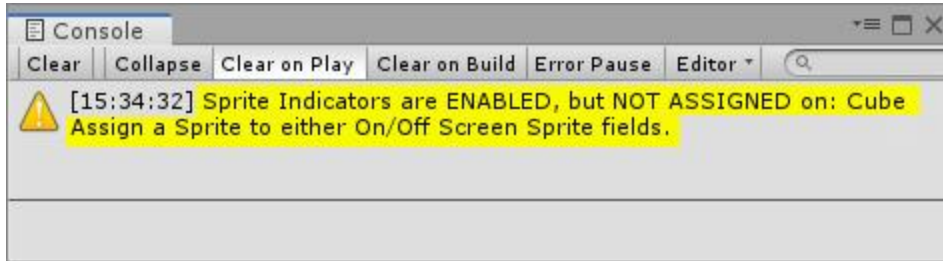
1. Locate your image in Unity via the Project tab (Window > General > Project)
2. Go to the Inspector Window (Window > General > Inspector)
3. Click "Texture Type" at the top
4. Select "Sprite (2D and UI)"
5. Click "Apply" in the bottom right
6. Your icon is ready to use

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

1. 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.
2. Disable Pixel Perfect in your Canvas. Most sprite elements will be scaled or rotated, or use subtle animated position or scaling, therefore it may be advantageous to disable Pixel Perfect, since the movement will be smoother without. Additionally, enabling Pixel Perfect has caused Sprites to not render at all.
3. 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.

Demo won't play (Scene Errors)

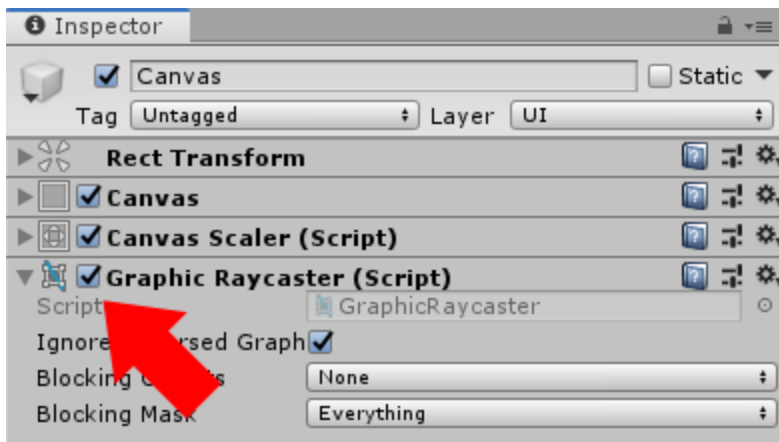
If you plan on using or playing the demo in your new project, be sure to add the respective scenes to your build index under *File > Build Settings...*

The scenes can be located in *Waypoint Indicators > Scenes*

To avoid possible errors and scene index conflicts with the demo and your project, it is recommended to play the demo in a brand new unity project.

Navigation UI elements aren't clicking through

There's a chance that Waypoint Indicator objects might be blocking your Raycast pointer. To fix this, simply go to your Canvas inspector window and disable “Graphic Raycaster”



Jittery UI in Screen Space - Camera Mode

1. If you are using a Camera controller, check to make sure that the Waypoint Indicator Update() function matches that of the camera controller. If your camera controller uses a FixedUpdate, make sure to change that in the Waypoint Indicator script to match.
2. Make a separate Canvas for your Waypoint Indicators and make it a child of the camera that renders the Indicators.

Still Not Working??

Please feel free to email me directly at peter.tracy@yahoo.com

Integrating with Curved UI

Waypoint Indicators supports [Curved UI](#). Follow the instructions below to get your waypoints working properly:

1. Add “using CurvedUI;” to the name space

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5 using TMPro;
6 using CurvedUI;
7
```

- 2.
3. Search for “Set CurvedUI effect manually” and uncomment the line(s) below it for each result (there are probably 9 places to uncomment). There may be instances where two lines need to be uncommented. They will show up clustered together.

```
//Set CurvedUI effect manually  
//wpParentGameObject.AddComponent<CurvedUIVertexEffect>();
```

AND...

```
//Set CurvedUI effect manually  
//gameObjectIndicator.AddComponent<CurvedUIVertexEffect>();  
//gameObjectIndicatorChildGameObject.AddComponent<CurvedUIVertexEffect>();
```

4.

Contact

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

@studio11508 | <http://ptracy.com/> | [Feedback Form](#)