Interaction System: Documentation

Version 1.0

1. Overview

Thank you for choosing Interaction System! This is a lightweight, performant, and data-driven plugin that allows you to create complex, stateful interactions *without writing new code*.

This system is built on the **Strategy Design Pattern** and leverages Unity's powerful `[SerializeReference]` feature. This combination allows you to assign a unique behavior "strategy" (like `OneShotInteraction` or `ToggleInteraction`) to any object and configure all its properties directly in the Inspector.

The custom editor script included in this package automatically finds any new `InteractionStrategy` you create and adds it to the component's dropdown menu, making the system 100% extensible.

2. Core Features

- **Strategy-Driven:** Assign a single, powerful "Strategy" (e.g., `OneShotInteraction`) to any interactable object.
- **Data-Driven:** All logic is self-contained. You no longer need to create separate scripts like `Door.cs` or `Chest.cs`.
- **Custom Inspector UI:** A clean, built-in dropdown menu lets you choose and assign your `InteractionStrategy` in one click.
- **Fully Extensible:** Automatically detects and adds any new C# class you write that inherits from `InteractionStrategy`.
- **Performant Detection:** Uses an efficient `Physics.OverlapSphereNonAlloc` method for zero garbage collection during runtime checks.
- **Physics Layer Filtering:** All detection is filtered by a custom Physics Layer for maximum performance.
- **Clean Architecture:** Includes pre-configured Runtime and Editor Assembly Definitions, as recommended by Unity for all standard plugins.

3. How to Use: Quick Start Guide

This guide will walk you through the setup from a blank scene to a working interaction.

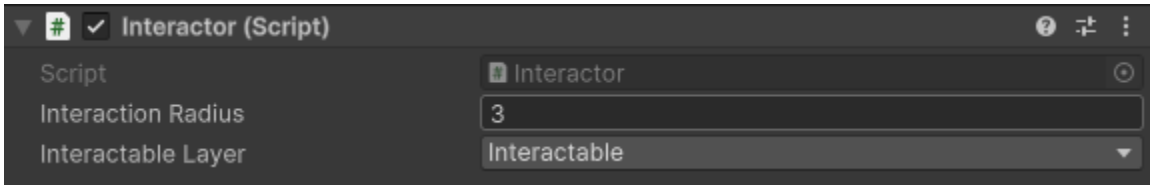
Step 1: Set Up the Physics Layer

This is the most important step for performance.

1. In Unity, go to **Edit > Project Settings...**
2. Navigate to the **Tags and Layers** tab.
3. Under "Layers," add a new user layer. We recommend naming it `Interactable`. Your plugin will *only* detect objects on this layer.

Step 2: Set Up the Player (`Interactor` Component)

1. Select your Player GameObject.
2. Add the `Interactor.cs` component to it.
3. In the Inspector, you will see two fields that you must set:
    a. **Interaction Radius:** The size of your detection "bubble" (e.g., 3).
    b. **Interactable Layer:** From the dropdown, select the `Interactable` layer you just created.



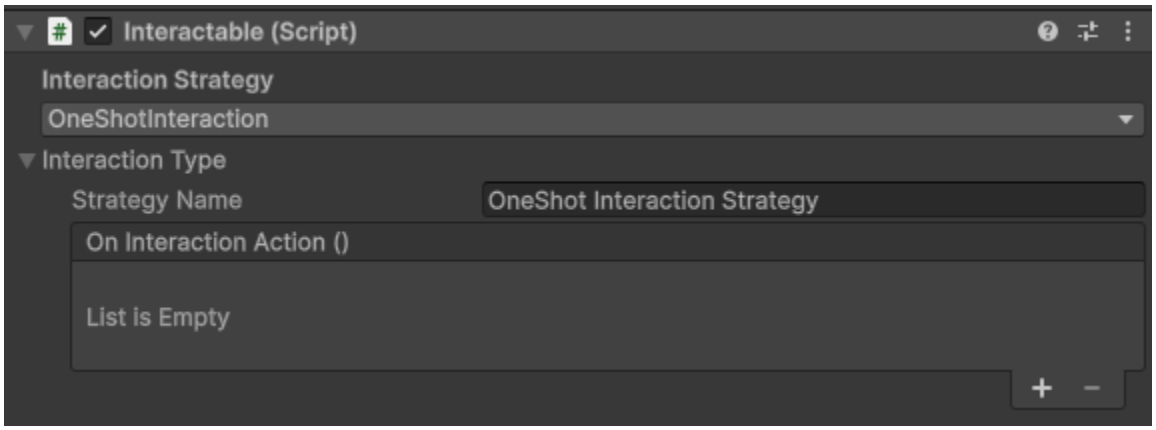
Step 3: Create an Interactable Object (`Interactable` Component)

This is the core of the plugin.

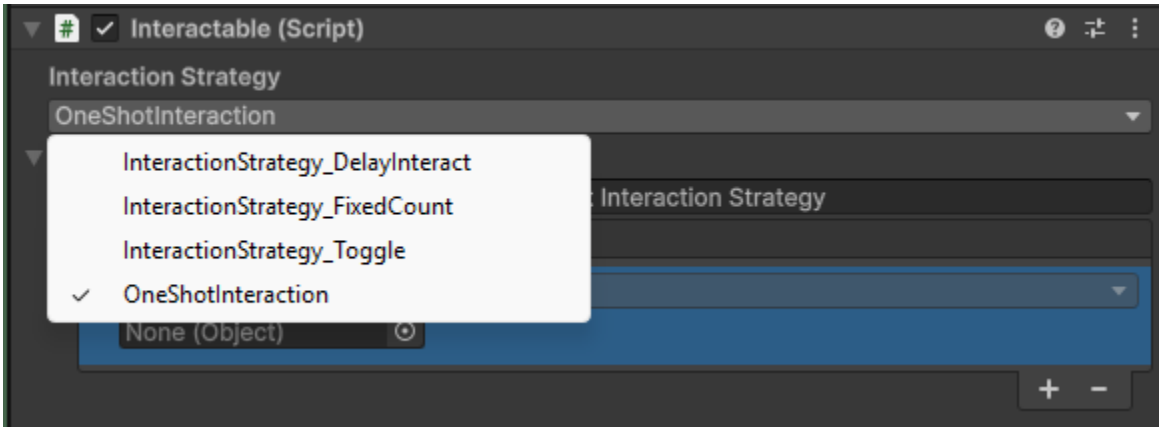
1. Create any object in your scene (like a Cube) that you want to interact with.

2. **Ensure it has a Collider** (e.g., a `BoxCollider`).
3. In the top-right of the Inspector, set the object's **Layer** to **`Interactable`**.
4. Add the **`Interactable.cs`** component to the object.

You will now see the `Interactable` component in the Inspector. It defaults to the `OneShotInteraction` strategy, and its fields are immediately visible and ready to be configured.



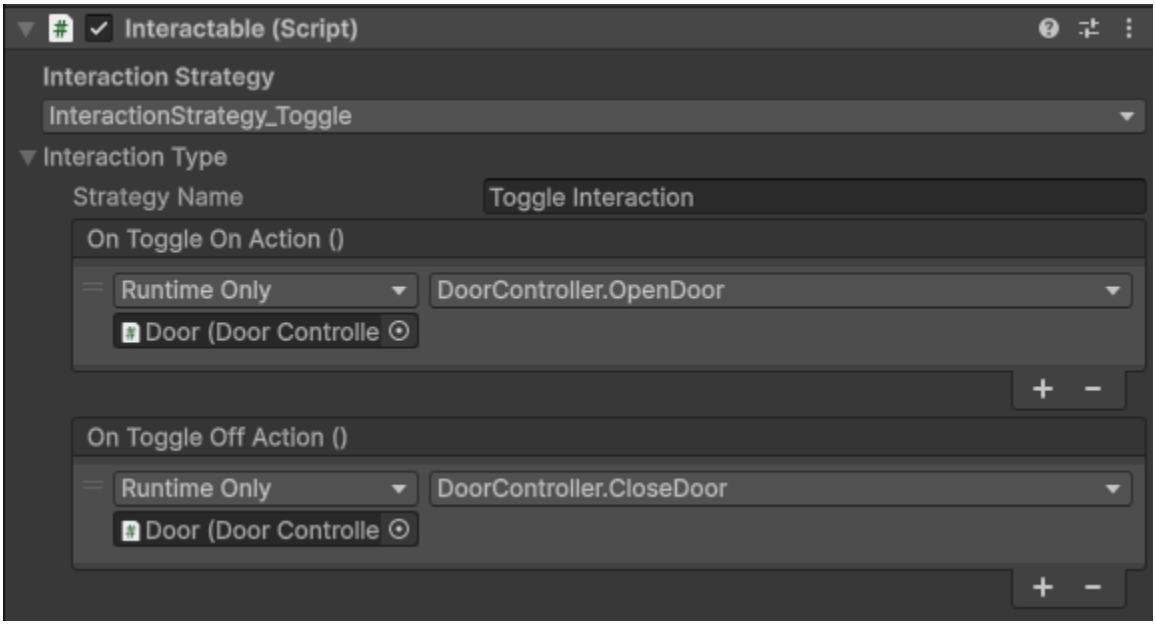
To change the behavior, simply use the **"Interaction Strategy"** dropdown. The custom editor will show you all available strategies (like `ToggleInteraction`) that you can switch to at any time.



Step 4: See It in Action

The included **Sample Scene** (`Assets/InteractionSystem/Samples/DemoScene.unity`) has a pre-built Player and several objects (like a chest and a door) already configured.

Press **Play** and walk up to an object. When you get close, your `Interactor` will detect it. Press the **E** key (or your custom key) to fire the interaction.



4. How to Extend: Creating New Strategies

The plugin automatically detects new strategies.

1. Create a new C# script (e.g., `MyNewStrategy.cs`) inside the `Runtime/Strategies` folder.
2. Make sure it inherits from `InteractionStrategy` and has the `[System.Serializable]` attribute.
3. Save the file. It will **automatically** appear in the "Interaction Strategy" dropdown on all `Interactable` components.

5. Support

If you have any questions, find a bug, or have a feature request, please contact:

Emai:- prakashrandhir16528@gmail.com