App Startup Documentation

Table of Contents

Overview

Startup Manager provides a simple and structured way to manage the application initialization process in Unity. It introduces interfaces such as `IInitializable` and `ICoroutineInitializable` to ensure consistent and ordered initialization of your systems in Unity projects.

Features

- **Asynchronous initialization**: Initialize systems asynchronously using coroutines or UniTasks.
- **Initialization order**: Define the order of initialization for systems.
- **Loading screen**: Show a loading screen while initializing systems.
- **Accumulated Initialization Progress**: Track and display the overall initialization progress to provide feedback to users.
- **Create DontDestroyOnLoad objects**: Instantiate all objects that should not be destroyed on scene load in one place.

Get Started

Basic Usage

1. Create a class that inherits from `EntryPointMonoBehaviour` or `EntryPointScriptableObject`.

```
[CreateAssetMenu(fileName = "MyEntryPoint", menuName = "Startup Manager/MyEntryPoint", order = 1)]
No asset usages  Aleksei Antipin
public class MyEntryPoint : EntryPointScriptableObject
{
}
```

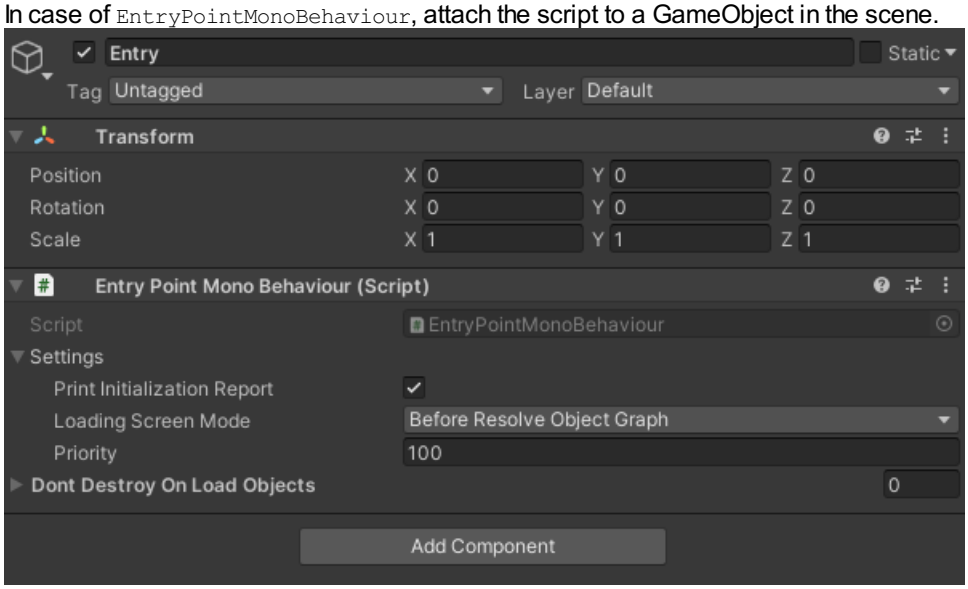
OR

```
No asset usages  Aleksei Antipin *  More...
public class MyEntryPoint : EntryPointMonoBehaviour
{
}
```

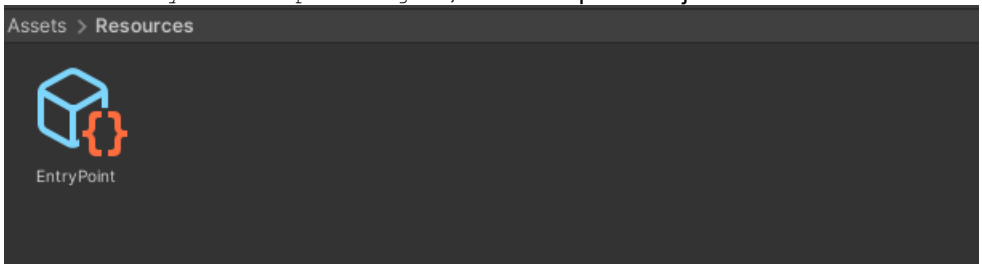
2. Write initialization logic.

```
public class MyEntryPoint : EntryPointScriptableObject
{
    #region Overrides
    Frequently called  0+1 usages  new *
    public override void ApplySettings()
    {
        Application.targetFrameRate = 60;
    }

    Frequently called  0+1 usages  Aleksei Antipin *
    public override void ResolveObjectGraph()
    {
        var initializables = new List<IInitializableInternal>
        {
            new InitializableService1(),
            new InitializableService2(),
            new AsyncInitializableService1()
        };

        initializables.ForEach(i :IInitializableInternal => AddInitializable(i));
    }
    #endregion
}
```

3. In case of `EntryPointMonoBehaviour`, attach the script to a GameObject in the scene.



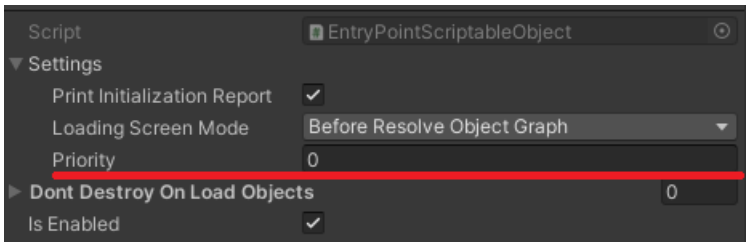
4. In case of `EntryPointScriptableObject`, create scriptable object asset in Resources folder with name EntryPoint.



5. Run the scene.

Entry Point Priority

`EntryPointMonoBehaviour` and `EntryPointScriptableObject` have a priority property that determines which entry point to use. If you have multiple entry points in the scene and in the resources folder, the entry point with the highest priority will be used. This way you can start up some scenes with different initialization logic. This is useful in development and testing ideas.



Examples

EntryPointMonoBehaviour

```
using Abyss.StartupManager;

public class MyEntryPoint : EntryPointMonoBehaviour
{
    #region Overrides
    public override void ResolveObjectGraph()
    {
        var service = new InitializableService1();
        AddInitializable(service);
    }
    #endregion
}
```

EntryPointScriptableObject

```
using Abyss.StartupManager;
using UnityEngine;

[CreateAssetMenu(fileName = "MyEntryPoint", menuName = "Startup Manager/MyEntryPoint", order = 1)]
public class MyEntryPoint : EntryPointScriptableObject
{
    #region Overrides
    public override void ResolveObjectGraph()
    {
        var service = new InitializableService1();
        AddInitializable(service);
    }
    #endregion
}
```

Initializables

```
    public class InitializableService : IInitializable
 {
  #region Interface Implementations
  public void Initialize()
  {
   Debug.Log("Initializing...");
  }
  #endregion
 }
```

Coroutine Initializables

```
public class InitializableWithReportExampleService : ICoroutineInitializable
{
    #region Interface Implementations
    public IEnumerator Initialize(IProgressReceiver progressReceiver)
    {
        for (var i = 0; i < 10; i++)
        {
            var message = $"Initializing: {GetType().Name} step: {i}";
            progressReceiver.Report(0.1f * i, message);

            yield return new WaitForSeconds(0.1f);
        }
    }
    #endregion
}
```

UniTask Initializables

```
public class InitializableWithReportExampleService : IUniTaskInitializable
{
 #region Interface Implementations
 public async UniTask Initialize(IProgressReceiver progressReceiver)
 {
  for (var i = 0; i < 10; i++)
  {
   var message = $"Initializing: {GetType().Name} step: {i}";
   progressReceiver.Report(0.1f * i, message);
   await UniTask.Delay(100);
  }
 }
 #endregion
}
```