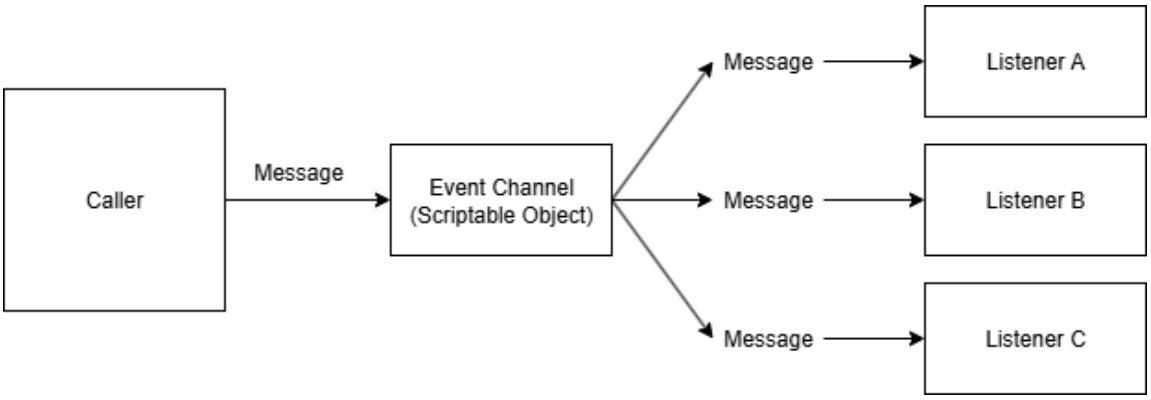
EVENT CHANNEL SYSTEM

I. WHAT DOES IT DO?

- Event Channel System is for decoupling code and creating code that is more encapsulated.
- Event Channel System is using Scriptable Object as a middleman to connect events.
- By doing so, objects don't necessarily know each other and can be easily refactored later.



II. HOW TO USE IT?

- First, create a *custom event channel scriptable object* with a *custom event context*. In this example, a custom event channel will be named: **TestEventChannelSO**, and custom event context will be named: **TestContext**

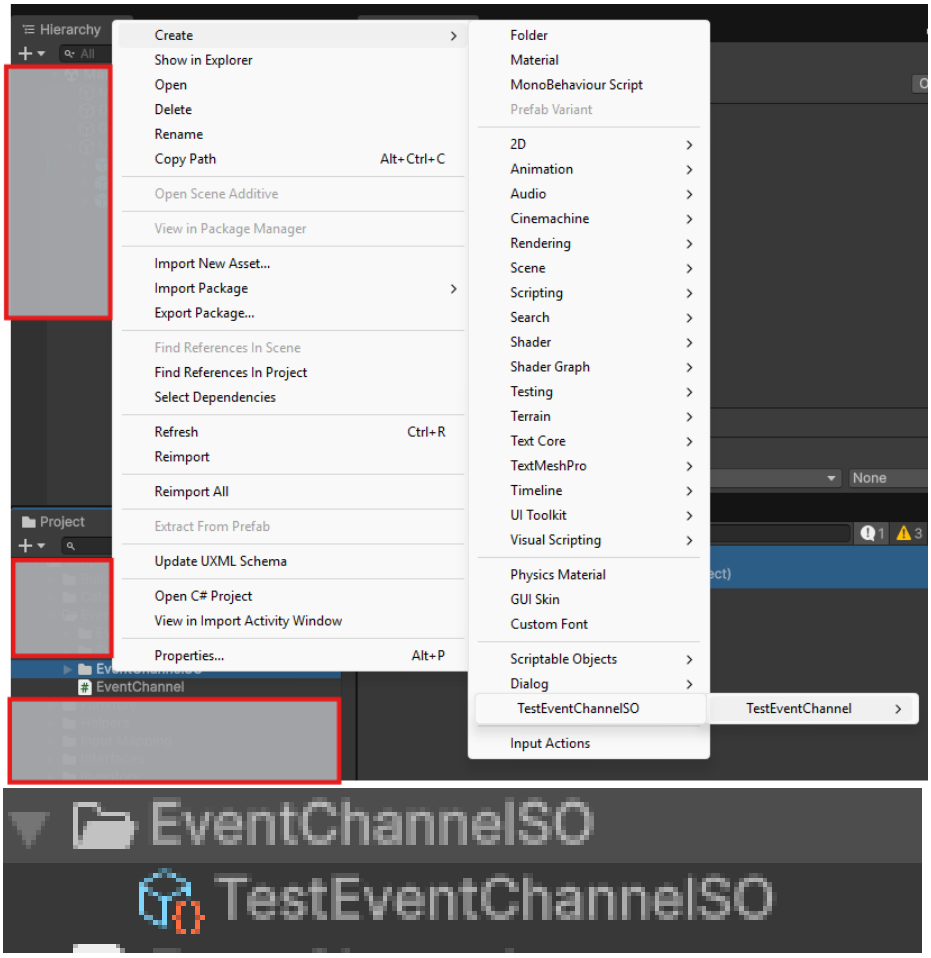
```
[CreateAssetMenu(menuName = "Events/TestEventChannel/TestEventChannelSO", fileName = "TestEventChannelSO")]
No asset usages  1 usage
public class TestEventChannelSO : BaseEventChannelSO<TestContext>
{
}

5 usages
public class TestContext : EventContext
{
    public string msg;
    1 usage
    public TestContext(string msg)
    {
        this.msg = msg;
    }
}
```

  - The **TestContext** in this case will act as a message.

- The **TestEventChannelSO** will act as a middleman event channel

- Note that to make it works, **TestContext** MUST INHERIT from **EventContext** and **TestEventChannelSO** MUST INHERIT from **BaseEventChannelSO**

- Now created a new EventChannelSO:



- Next, create an event listener to listen to the event channel. In this example, let's name it **TestEventChannelListener**

```
No asset usages
public class TestEventChannelListener : BaseEventChannelListener<TestEventChannelSO, TestContext>
{
}
```

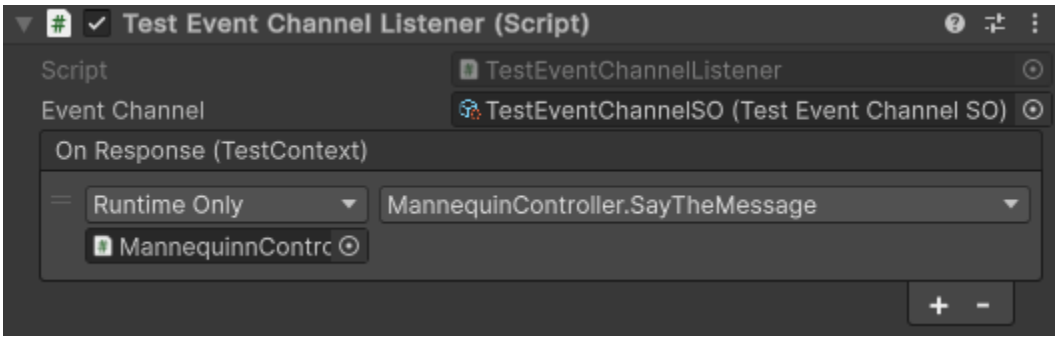
  - The **TestEventChannelListener** will be inherited from **BaseEventChannelListener<TestEventChannelSO, TestContext>**
  - In this case, for the listener to work, we need to specify what *event channel* to listen to and what type of *event context*
- To call the event, simply refer the event channel and call it:

```
[SerializeField] private TestEventChannelSO testChannel;
```

```
testChannel?.RaiseEvent(new TestContext( msg: $"Hello from the {gameObject.name}"));
```

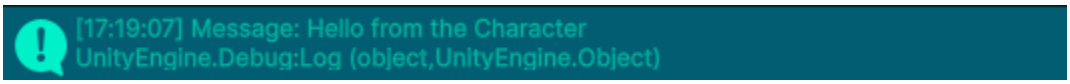
  - To call the channel, use the **RaiseEvent** method.

- Now add the *event channel listener* to any objects that you want to listen to the event.




```
public void SayTheMessage(TestContext ctx)
{
    Debug.Log($"Message: {ctx.msg}", this.transform);
}
```

- And voila, your first custom event channel is now complete! Let's test this out:



It's working really well! And as you can see, these objects don't even need to know each other, but they can still listen to each other really well!