Integración de Unity con los puertos seriales: Ardity

Referencia

https://ardity.dwilches.com/

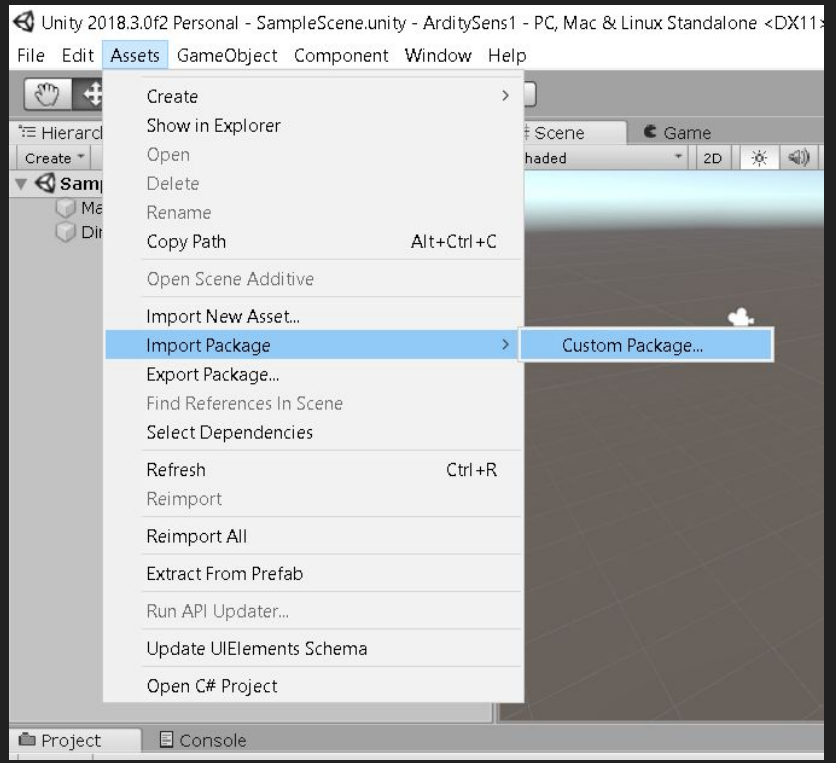
Ejercicio 1: documentación básica

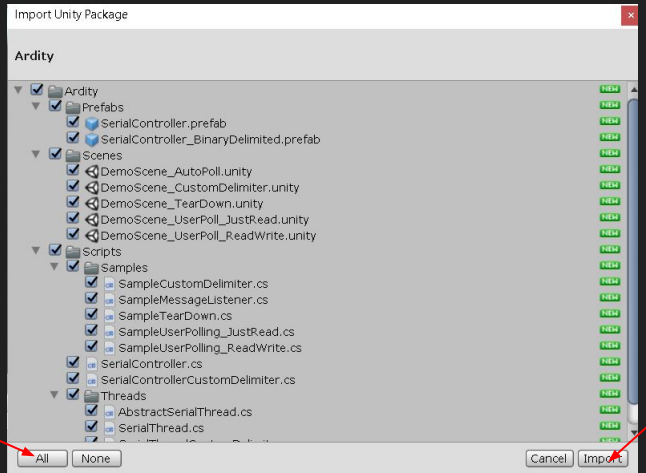
1. Abrir el enlace: https://ardity.dwilches.com/

2. Así funciona Ardity: “Ardity creates a thread in which it polls a COM port, all data it receives is stored in a shared thread-safe queue”. Explique (escriba) qué quiere decir esta frase.

Ejercicio 1 (cont)

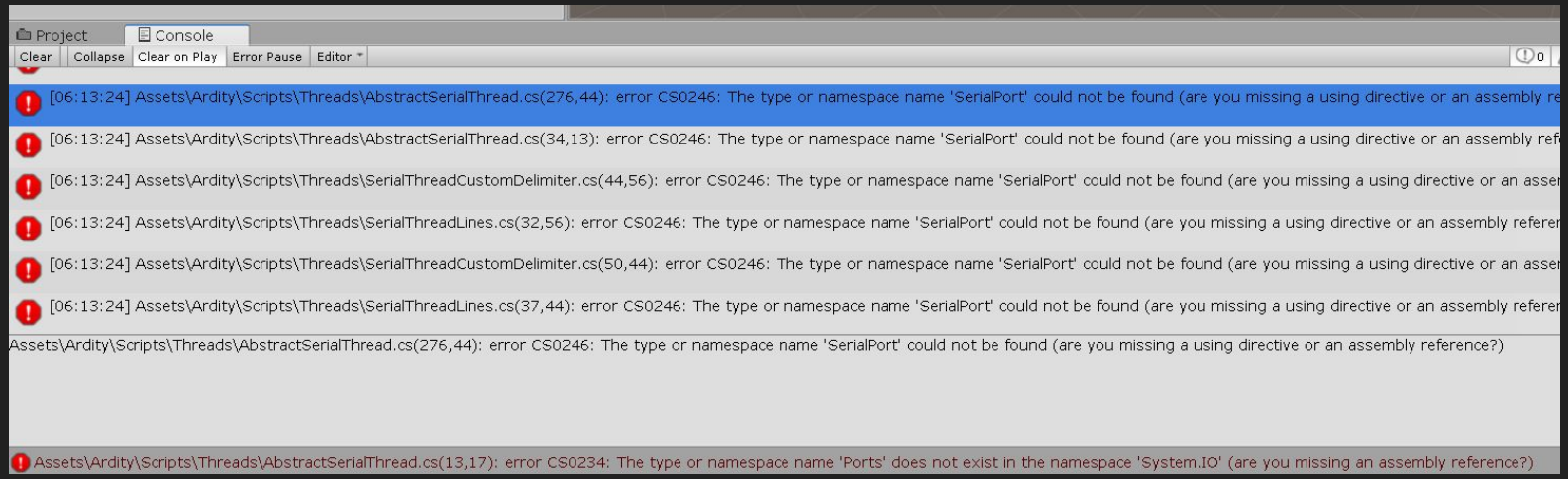
4. Crea un proyecto en Unity
5. Descargua e incluye en el proyecto el paquete Ardity





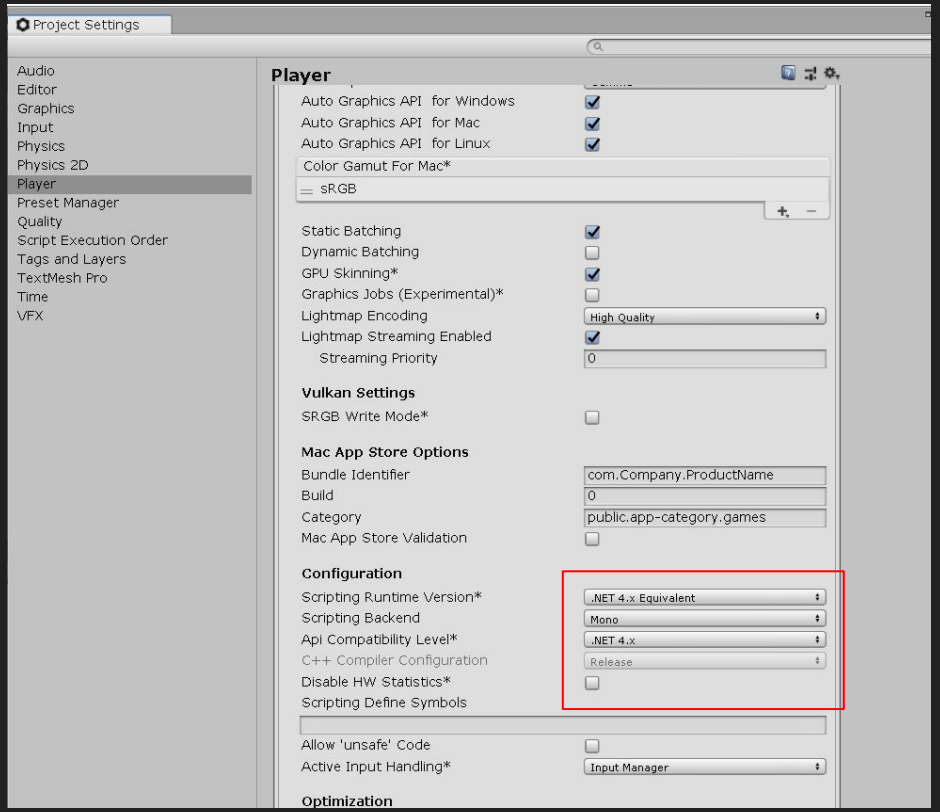
Ejercicio 1 (cont)

6. Al importar el paquete puede salir el siguiente error



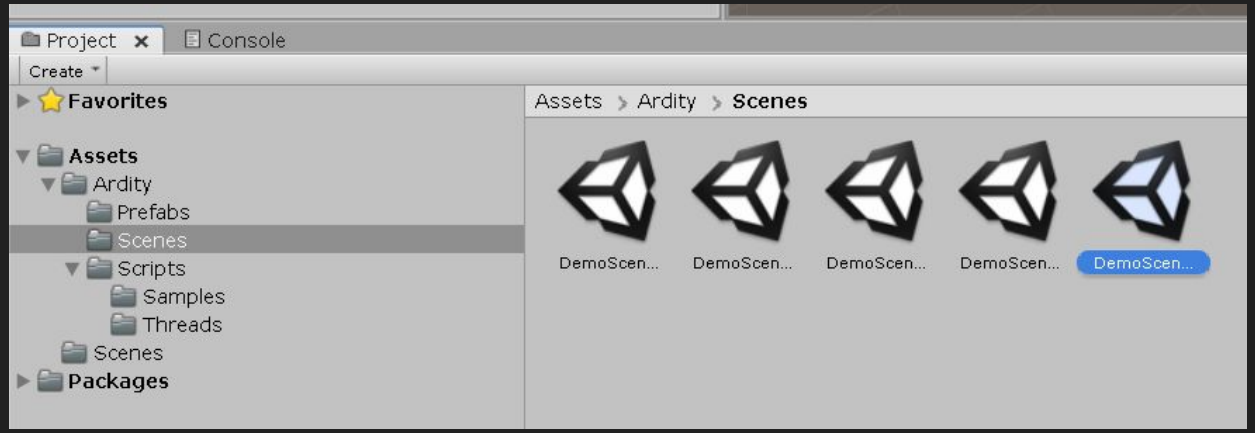
Ejercicio 1 (cont)

7. Para solucionar este problema ingresa al menú Edit / Project Settings y modifica la opción de Player que muestra la figura.
8. ¿Para qué hacemos esta modificación? Para poder utilizar el API de comunicaciones seriales que por defecto no viene habilitada en Unity.



Ejercicio 1 (cont)

9. Una vez solucionado el error, abre el paquete Ardity. Nota las carpetas Prefabs, Scenes y Scripts. En esas carpetas podrás encontrar los recursos necesarios para utilizar la funcionalidades que ofrece el paquete.



Ejercicio 2: demo

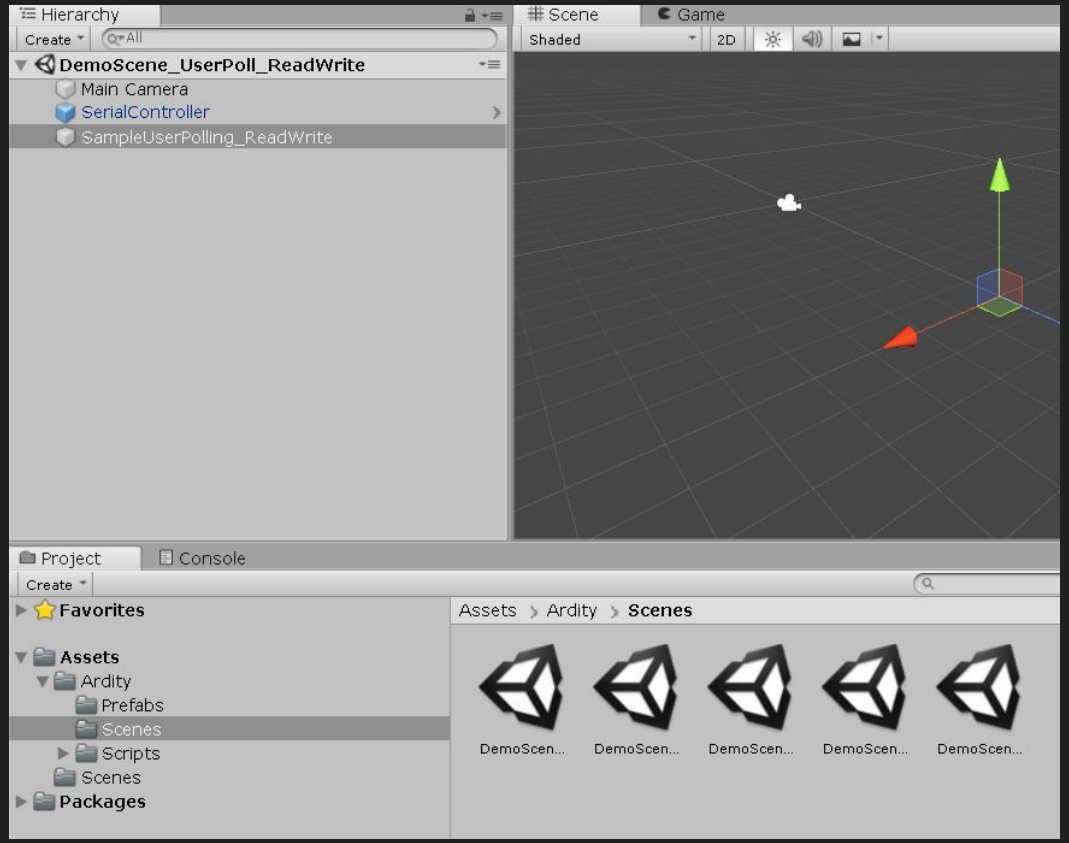
Ahora vamos a reproducir un demo que permitirá ilustrar cómo interactuar con el paquete.

1. Programa un Arduino con el código que está en la sección Arduino sample code del sitio oficial del paquete.
2. **MUY IMPORTANTE**: **el paquete sabe que Arduino envió algo porque recibe el carácter de nueva línea. Este carácter es enviado por Arduino cuando se usa Serial.println.**

```
unsigned long last_time = 0;

void setup()
{
    Serial.begin(9600);
}

void loop()
{
    // Print a heartbeat
    if (millis() > last_time + 2000)
    {
        Serial.println("Arduino is alive!!");
        last_time = millis();
    }

    // Send some message when I receive an 'A' or a 'Z'.
    switch (Serial.read())
    {
        case 'A':
            Serial.println("That's the first letter of the abecedarium.");
            break;
        case 'Z':
            Serial.println("That's the last letter of the abecedarium.");
            break;
    }
}
```

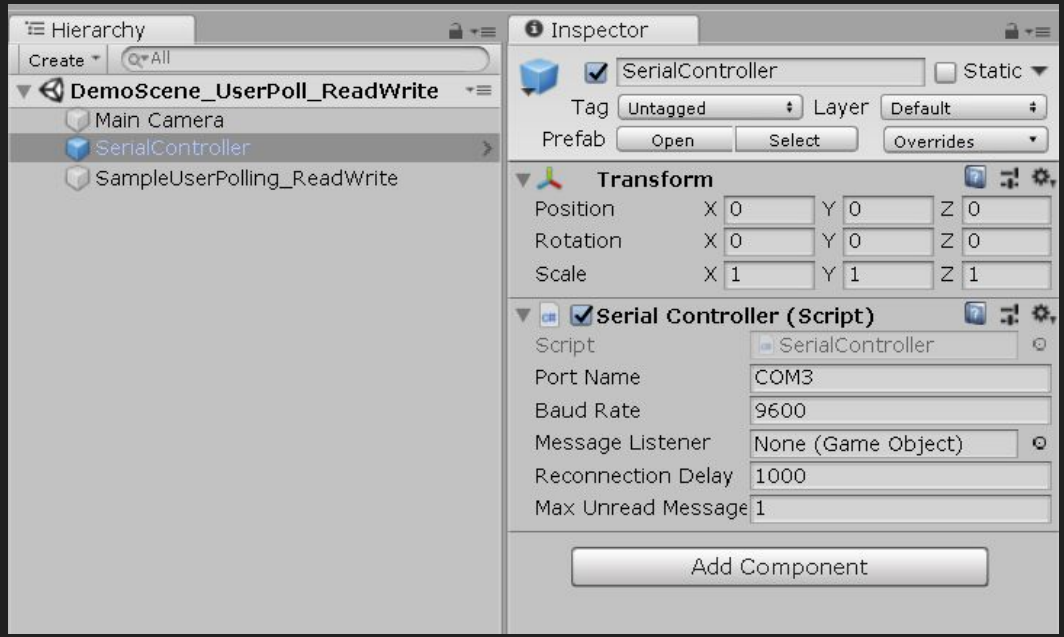
Ejercicio 2: cont

7. Carga la Scene DemoScene_UserPoll_ReadWrite
8. Nota que tenemos tres GameObject: Main Camera, Serial_Controller (prefab) y SampleUserPolling_ReadWrite



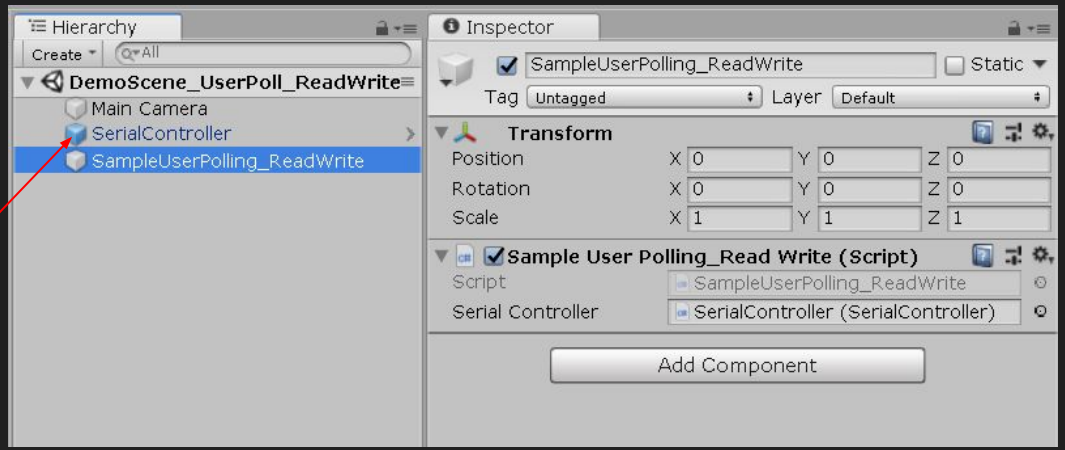
Ejercicio 2: cont

9. Una aplicación en Unity debe tener como mínimo el Script SerialController adicionado como componente a un GameObject. En el caso del demo, tenemos el Prefab SerialController que cumple con ese rol.
10. Nota que SerialController exhibe unas propiedades que debemos configurar según nuestro proyecto: Port Name, Baud Rate (estas por ahora)



Ejercicio 2: cont

12. Ahora señala el GameObject SampleUserPolling_ReadWrite.
13. Nota la propiedad SerialController. En este ejemplo se colocó en ese campo el nombre del GameObject que tiene asociado el Script SerialController, en este caso, el GameObject tiene el mismo nombre: SerialController



Ejercicio 2: cont

14. Abre el componente Sample User Polling_Read Write (en este caso el componente es un Script).
15. Nota que aunque tenemos configurado el SerialController, de todas maneras en Start se busca el GameObject llamado SerialController y luego dentro de ese GameObject se busca el componente SerialController, en ese caso una instancia de la clase SerialController.
16. Nota el mensaje de Debug.Log. ¿Qué significa esto en relación con el código de Arduino?

```
// Initialization
void Start()
{
    serialController = GameObject.Find("SerialController").GetComponent<SerialController>();

    Debug.Log("Press A or Z to execute some actions");
}
```

Ejercicio 2: cont

17. Analiza el método Update.
18. Finalmente, prueba el DEMO.