УДК 004.358

Примеры использования сокетов для проектов виртуальной реальности в Unity средствами библиотеки XR Socket Interactor[1]

Згода И.К.
Алтайский государственный университет, г. Барнаул
playa-ogg@yandex.ru

Аннотация

В данной статье рассматриваются вопросы использования инструментов библиотеки XR Socket Interactor. Возможности библиотеки позволяет реализовать механизмы закрепления объектов в сокеты и их последующее извлечение. Такие инструменты используются в проектах виртуальной реальности, где требуется точное позиционирование и манипулирование объектами.

Ключевые слова: VR, XR, сокет, объект, игра, позиция, последовательность элементов, симуляция, сцена, взаимодействие объектов.

1. Введение

Виртуальная реальность (VR) является быстроразвивающейся областью, которая открывает новые возможности для иммерсивного взаимодействия с виртуальными объектами и становится всё более популярной в различных областях, включая игры, медицину и обучение. Точное позиционирование и манипулирование объектами в среде VR является важной составляющей многих игровых и симуляционных проектов. Поэтому существует необходимость взаимодействия объектов в виртуальном пространстве с реальным оборудованием или другими виртуальными объектами. Одним из способов достижения этой цели является использование сокетов. XR Socket Interactor – это инструмент для кроссплатформенной среды Unity, который позволяет разработчикам реализовывать создание и работу с сокетами. В данной статье рассматриваются особенности применения инструментов XR Socket Interactor в рамках реализации проектов по виртуальной реальности.

2. Работа с сокетами в XR Socket Interactor

Перед тем как углубиться в принципы работы XR Socket Interactor, необходимо понять, что такое сокеты. Сокеты в Unity – это виртуальные контейнеры, представляющие собой невидимые точки привязки, к которым можно прикреплять другие объекты

Библиотека XR Socket Interactor является частью XR Interaction Toolkit – подсистемы Unity, и представляет собой набор компонентов, который можно добавлять к любому объекту виртуального мира.

Ключевой особенностью XR Socket Interactor является его способность соединять объекты в виртуальном мире. Он позволяет создавать различные механики соединения объектов, такие как включение и выключение электрического устройства, присоединение кабелей или механическое соединение различных компонентов. Это позволяет разработчикам создавать более реалистичные виртуальные среды и сцены.

[1]Работа поддержана средствами программы развития ФГБОУ ВО Алтайский государственный университет «Приоритет-2030»

XR Socket Interactor использует разные методы для обнаружения сокетов в пространстве. Один из наиболее распространенных методов – это использование коллайдеров. Разработчик может добавить коллайдеры к объектам-сокетам, чтобы обозначить их местоположение и размеры. XR Socket Interactor использует информацию о коллайдерах для определения возможности взаимодействия с сокетами. Когда сокет находится в пределах досягаемости XR Socket Interactor, пользователь может взаимодействовать с ним, закреплять или извлекать объекты.

Данный набор компонентов позволяет пользователю выполнять различные действия с сокетами, такие как закрепление или извлечение объектов. При взаимодействии XR Socket Interactor выполняет проверку, чтобы убедиться, что объект соответствует требованиям сокета, например, его размер и форма. Если объект удовлетворяет требованиям, он может быть закреплен или извлечен из сокета.

Для того, чтобы закрепить объект в сокет, необходимо определить пользовательский ввод. XR Socket Interactor позволяет использовать контроллеры VR для выбора и размещения объектов в сокеты. Принципы закрепления объектов в сокеты основаны на концепции связей и законов кинематики. При помещении объекта в сокет объект становится связанным с этим сокетом и подчиняется его движениям и вращениям. Если сокет перемещается или вращается, объект, прикрепленный к нему, будет двигаться и вращаться вместе с ним.

Примеры использования сокетов

Использование XR Socket Interactor имеет большой потенциал в обучении и тренировках, особенно в области индустриальной сборки. Представим, что мы разрабатываем тренажер для обучения монтажу сложных изделий, таких как автомобили. С помощью XR Socket Interactor можно создавать виртуальные модели компонентов и инструментов, а затем позволить пользователям собирать их вместе, следуя определенным инструкциям.

Другим примером может послужить использование XR Socket Interactor в медицинском обучении. Медицинское обучение требует от студентов приобретения определенных навыков, включая выполнение манипуляций и процедур, таких как катетеризация и шприцевание. Использование XR Socket Interactor позволяет создавать виртуальные сцены, в которых студенты могут практиковать данные навыки. Например, студент может практиковаться во вставлении виртуального катетера в виртуальное сердце, присоединяя его к нужной анатомической точке. XR Socket Interactor обеспечивает точное и реалистичное взаимодействие с виртуальными объектами, позволяя студентам получить ценный опыт.

В рамках данного исследования реализован тренажёр по сборке автомата из частей, с использованием XR Socket Interactor и Interaction Layers-слоёв. Компоненты данных библиотек позволили не только разработать механику сборки и разборки оружия, но также и заложить в проект специальные инструменты для настройки используемых объектов для сцен и для расширения функциональных возможностей. Ключевой особенностью созданной механики является задание определенной последовательности элементов для сборки, и точное соответствие каждой детали своему сокету. Поэтому, исключается возможность установки элементов оружия не «на своё место».

Последовательность элементов при сборке напоминает стек и работает по такому же алгоритму LIFO (последний вошёл – первый вышел). Например, когда пользователь собирает автомат, он начинает сборку последовательно с установки сначала газовой трубки, потом втулок, и пружины и закрывает все эти детали крышкой, разборка осуществляется полностью в обратном порядке, то есть нельзя вытащить детали из-под закрытой крышки, так же, как и нельзя установить деталь под закрытую крышку, нужно сначала снять её.

Точное соответствие каждой детали своему сокету реализовано по следующему правилу: к детали добавляется компонент XR Grab Interactable, который даёт возможность взаимодействовать с объектом, брать его и т.д., а также присваивать ему слой Interaction Layers. Далее для каждой детали создаётся отдельный слой и устанавливается в свойствах

Interaction Layers компонента XR Grab Interactable, также он устанавливается в свойствах соответствующего слота, только уже у компонента XR Socket Interactor. Слой в этом случае является интерфейсом взаимодействия конкретной детали со своим сокетом.

```
private void OnTriggerEnter(Collider other)
{
    if(other.name == _me.name)
    {
        StartCoroutine(WaitForInstall());

        for (int i = 0; i < _ignoreSlots.Length; i++)
        {
            _ignoreSlots[i].gameObject.GetComponent<Collider>().enabled = false;

            if (CheckInstall(_ignoreSlots[i]))
            {
                _ignoreSlots[i].GetComponent<CustomAK_S>()._me.GetComponent<XRGrabInteractable>().interactionLayers &= ~(1 << _defaultLayer);
            }
        }
    }
}
```

Рисунок 1. Метод OnTriggerEnter

На рисунке 1 представлена функция, которая вызывается триггером сокета, связанного с определенной составной частью оружия. В методе сначала вызывается корутина, представленная на рисунке 3, далее идёт проверка всех элементов последовательности, которые в стеке расположены ниже. У нижестоящих сокетов отключаются коллайдеры, а у установленных деталей отключается слой, позволяющий их взять, то есть отсоединить от слота. Метод для проверки установки деталей в сокеты представлен на рисунке 2.

```
Ссылок: 3
private bool CheckInstall(Transform obj)
{
    return obj.transform.position == obj.GetComponent<CustomAK_S>()._me.transform.position;
}
```

Рисунок 2. Метод CheckInstall

На рисунке 2 представлен метод для проверки установки детали в сокет, который возвращает истину если позиция детали совпадает с позицией сокета.

```
Ссылок: 1
private IEnumerator WaitForInstall()
{
    yield return new WaitUntil(() => CheckInstall(transform));

    IsInstalled = true;
    S_ServiceLocator<IService>.Instance.Get<S_AssemblyManager>().OnAllComponentsInstalled();
}
```

Рисунок 3. Метод WaitForInstall

На рисунке3 представлена корутина, в которой вызывается обработчик события у класса верхнего уровня. На рисунке 4 представлен метод, который вызывается при выходе из триггера сокета. Процесс работы метода организован следующим образом: сначала прекращается выполнение всех корутин, далее у нижестоящих сокетов включаются коллайдеры, у установленных в оружие деталей включаются слои, позволяющие пользователю взаимодействовать с этими элементами.

В данной статье рассмотрен механизм работы XR Socket Interactor в XR приложениях. XR Socket Interactor представляет собой инновационный инструмент, который позволяет более естественно и реалистично взаимодействовать с объектами в XR среде. Он значительно упрощает и улучшает процесс разработки XR приложений, предоставляя возможность более удобного и точного управления объектами. С применением XR Socket

```
Сообщение Unity | Ссылок: 0
private void OnTriggerExit(Collider other)
{
    if (other.name == _me.name)
    {
        StopAllCoroutines();

        if(IsInstalled) IsInstalled = false;

        for (int i = 0; i < _ignoreSlots.Length; i++)
        {
            _ignoreSlots[i].gameObject.GetComponent<Collider>().enabled = true;

            if (CheckInstall(_ignoreSlots[i]))
            {
                _ignoreSlots[i].GetComponent<CustomAK_S>()._me.GetComponent<XRGrabInteractable>().interactionLayers |= (1 << _defaultLayer);
            }
        }
    }
}
```

Рисунок 4. Метод OnTriggerExit

Interactor, разработчики могут создавать более захватывающие, интуитивные и реалистичные XR приложения, что способствует положительному пользовательскому опыту.

Список литературы

1. Unity Technologies. “XR Interaction Toolkit - Socket Interactor.”. — URL: https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.0/manual/xr-socket-interactor.html.