Bow VR Unity Package Documentation

Table of Contents

1. Introduction

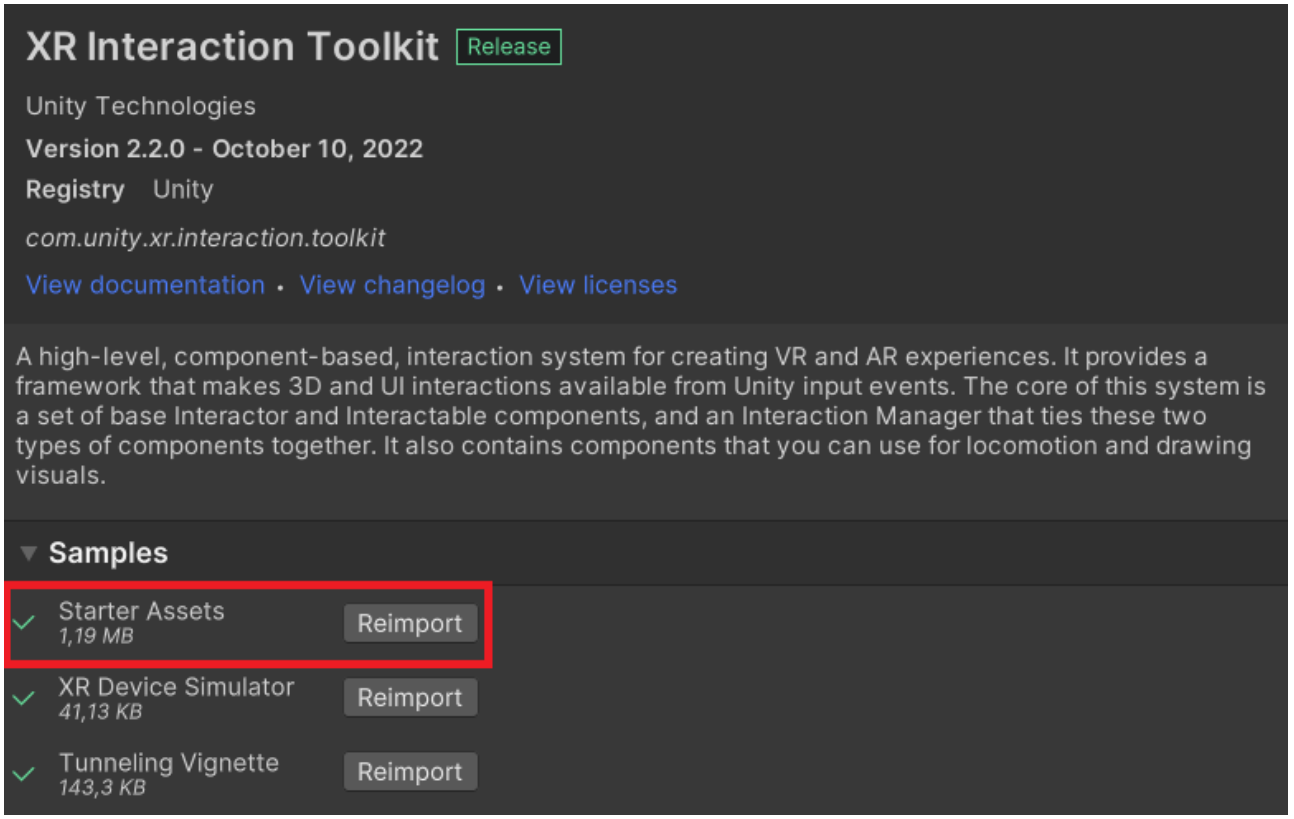
Welcome to the "Bow VR" Unity Package, your gateway to a captivating virtual reality archery experience! This package is designed to empower developers and creators with a set of tools and assets that enable the creation of dynamic bow and arrow mechanics for VR environments. "Bow VR" equips you with everything you need to bring the magic of archery to life.

2. Package Setup

To set up the Bow VR package in your Unity project, please follow these steps:

1. Ensure that you have the Universal Render Pipeline (URP) installed in your project. The Bow VR package was developed using URP and requires it to function properly.
2. If your project is using a different render pipeline, you may encounter pink materials in the demo scenes and on the prefabs. To resolve this, you need to update the shaders on the affected materials to match your render pipeline. Refer to the documentation or resources provided by your chosen render pipeline for instructions on how to update the shaders.

3. In order for the Bow VR package to work, you will need to have a VR setup in your project. This includes having OpenXR installed and configured.
4. Additionally, the package relies on the XR Interaction Toolkit, which is a Unity package for building VR and AR interactions. To use the Bow VR package, you will need to install the XR Interaction Toolkit and include the sample "Starter Assets" provided with it.



3. Features

"Bow VR" offers a wide array of features designed to empower VR developers and creators with the tools they need to create captivating archery experiences:

- Intuitive Bow Handling: "Bow VR" offers a natural and intuitive bow handling system for VR. Players can pull back the bowstring, aiming, and releasing it just like they would in real life, creating a highly immersive archery experience.
- Ready System for Applying Damage: "Bow VR" seamlessly integrates a ready system for applying damage to enemies.
- Realistic Physics: The package enables energy arrows to interact with Rigid Bodies within your VR environment. When an arrow collides with a Rigidbody, it responds realistically, adding a layer of immersion and authenticity to your VR world.

- Example Scene and Prefabs: "Bow VR" provides an example scene showcasing the package's features and functionality. Use the included prefabs to quickly and easily implement the bow mechanics, damage system, and more into your VR project.
- Immersive Visuals: Immerse players in visually captivating sci-fi and fantasy environments. "Bow VR" offers 3 sci-fi/fantasy bow models and particles for arrow collision.
- Immersive Sounds: Experience realistic sound effects for bow handling, shooting and arrow collision.

4. Usage

To use the Bow VR package in your Unity project, follow these steps:

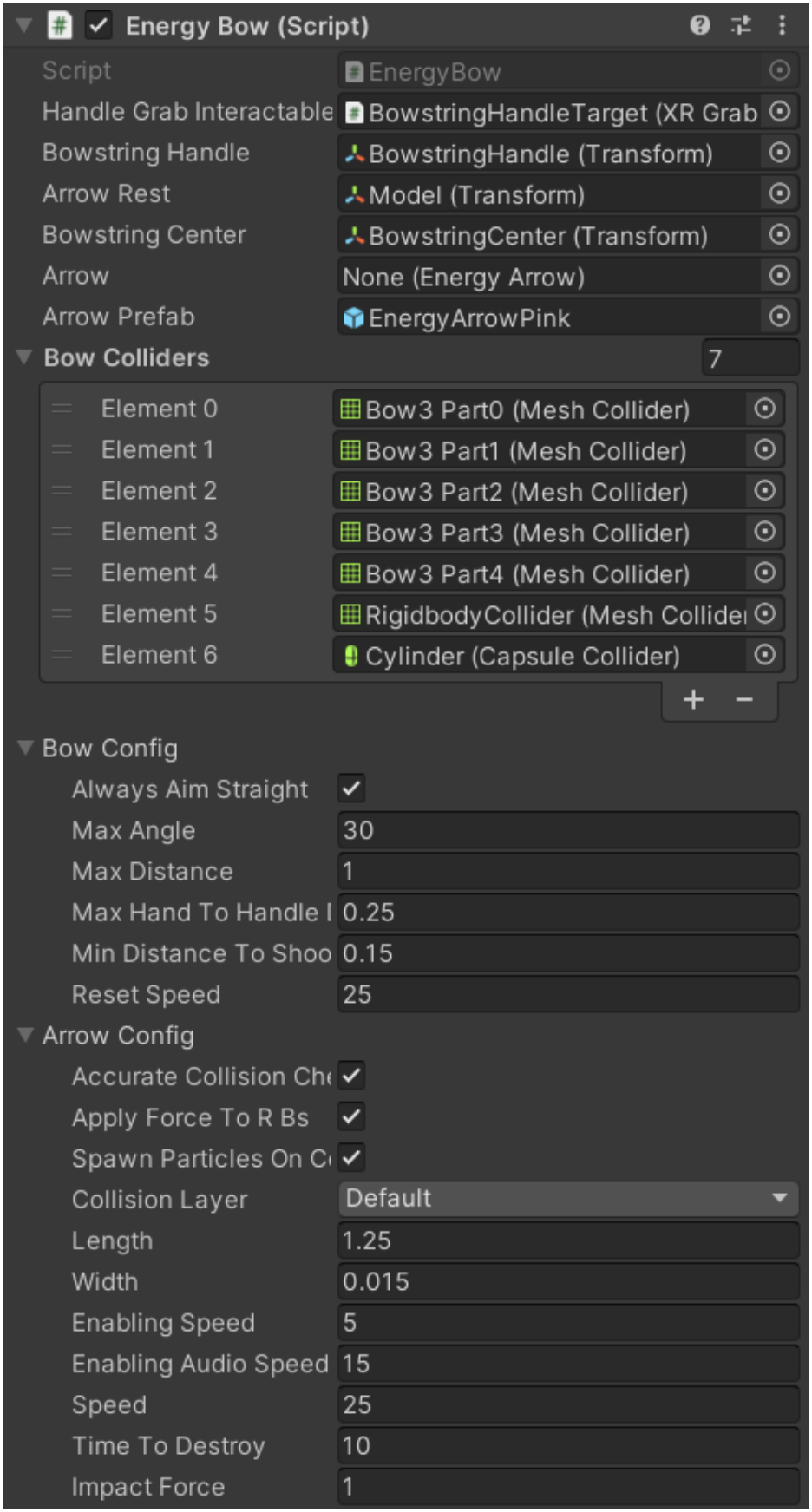
1. Drag and drop the "Bow" prefab into your scene.
2. Drag and drop the "Dummy" prefab into your scene. You can create your own targets, more on this topic in “Making Custom Targets” section.
3. Set up your VR camera rig if you haven't already. (you can use one provided with XR Interaction Toolkit sample assets)

5. Making Custom Targets

You can apply damage when an arrow hits an enemy or any other target by including the IDamagable interface. Don’t forget to include the “BeyondLimitsStudio” namespace in your script! Then you need to implement the ApplyDamage function that takes “DamageData” as a parameter. Below you can see an example of an implemented IDamagable interface that enables arrows to deal damage.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using BeyondLimitsStudios;

Odwołania: 0
public class Enemy : MonoBehaviour, IDamageable
{
    private float health = 100f;

    Odwołania: 2
    public void ApplyDamage(DamageData damageData)
    {
        health -= damageData.Damage;

        if (health <= 0)
            Die();
    }

    1 odwołanie
    public void Die()
    {
        Destroy(this.gameObject);
    }
}
```

6. Bow Settings



- **XRGrabInteractable handleGrabInteractable**: An interactable that is grabbed when you're pulling bowstring.
- **Transform bowstringHandle**: Transform that is used to determine the start of an arrow.
- **Transform arrowRest**: Transform that is used to determine where an arrow is facing. Arrow is facing from bowstringHandle to arrowRest.
- **Transform bowstringCenter**: Transform that is used to reset position of handGrabInteractable and bowstringHandle when bowstring is not being held.
- **GameObject arrowPrefab**: Arrow that will spawn when pulling bow string prefab.
- **List<Collider> bowColliders**: Colliders of the bow. Arrows shot from that bow will not be able to hit those colliders.
- **bool alwaysAimStraight**: If set to true arrow will be always facing arrowRest forward direction.
- **float maxAngle**: Angle between arrowRest.forward and arrow direction that determines when bowstring should stop being held by the player to not allow for unrealistic bow behaviors. (useful only if alwaysAimStraight is set to **False**)
- **float maxDistance**: If distance from bowstringHandle to bowstringCenter is greater than maxDistance bowstring should stop being held by the player to not allow for unrealistic bow behaviors. (useful only if alwaysAimStraight is set to **False**)
- **float maxHandToHandleDistance**: If distance from bowstringHandle to handleGrabInteractable is greater than maxHandToHandleDistance bowstring should stop being held by the player to not allow for unrealistic bow behaviors. (useful only if alwaysAimStraight is set to **True**)
- **float minDistanceToShoot**: If player stops holding bowstring and distance from bowstringHandle to bowstringCenter is greater than minDistanceToShoot arrow while shot. Otherwise it won't.
- **float resetSpeed**: How quickly should handleGrabInteractable and bowstringHandle move to bowstringCenter when bowsting is not being held.
- **bool accurateCollisionCheck**: If set to false it might not detect some arrow hits when framerate is low.
- **bool applyForceToRBs**: Whether the arrow should apply force to Rigid Bodies when it hits them.
- **bool spawnParticlesOnCollision**: Whether to spawn particles when the arrow hits an object.

- **LayerMask collisionLayer**: Layer mask used to determine what objects should the arrow interact with.
- **float length**: Length of the arrow.
- **float width**: Width of the arrow.
- **float enablingSpeed**: How fast should the arrow enable when pulling the bowstring.
- **float enablingAudioSpeed**: How fast should the arrow sound get to target volume.
- **float speed**: How fast is the arrow.
- **float timeToDestroy**: For how long should the arrow fly in case it doesn't hit anything.
- **float impactForce**: How much does the arrow affect Rigid Bodies when it hits them. Also arrow velocity affects how much the Rigid Body is affected.
- **float damage**: How much damage does the arrow deal when it hits an IDamagable.

7. Troubleshooting

If you encounter any issues or have trouble using the Bow VR package, try the following troubleshooting steps:

- Ensure that you have installed the package correctly by following the installation instructions.
- Verify that your VR controllers are properly configured and recognized by Unity.
- Check the Unity Console for any error messages related to the package.

8. Support and Feedback

For support or feedback regarding the Bow VR Unity package, you can reach out to the package developer through the following channels:

- Email: beyondlimitsstudio.gaming@gmail.com