Teleports

Moving locations

CONTENTS

Pickups 1

Basic Teleport 2

Setting Up the teleport 2

Coding the Teleport 3

Plugging in the Information 4

Teleporting Objects with Physics. 5

Exercise: A TWO-WAY teleport 6

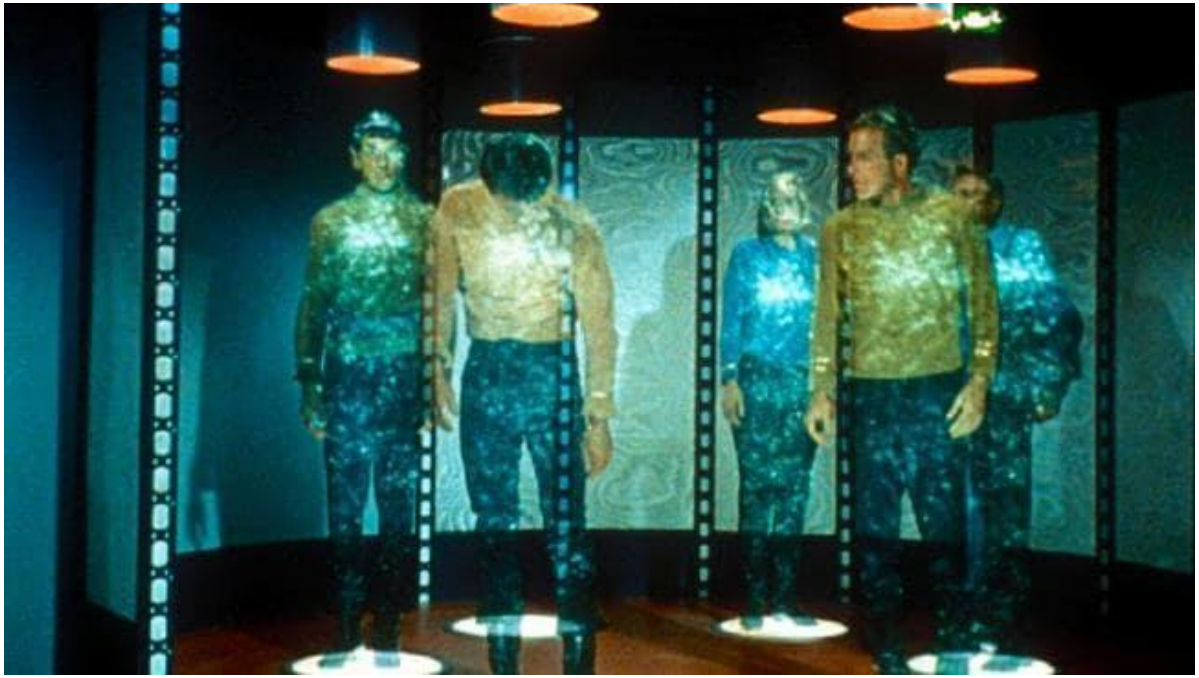
Scripting the Teleport 7

3rd Person Issue, Not teleporting Camera..... 8

PICKUPS

Teleports can be used in many ways

- As a game mechanic to move around the level
- As a way to re-spawn a character
- As a shortcut to move a player to another area on the map without them noticing
    - Was used a lot when we had pseudo 3D games to make the world seem more 3D
    - The game antechamber uses this a lot for its puzzles.



Basic Teleport

To create a basic teleport, you need to know

- What activates the teleport?
- What is being moved?
- Where are they moving to?
- What direction should they be facing?

Setting Up the Teleport

For the teleport, we need a point in the world we want to teleport our object to.

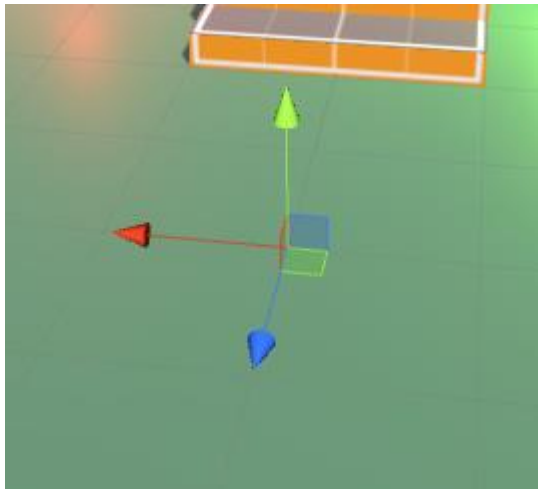
1. Create an empty GameObject
    a. Call it “BasicTeleportPoint”

We will set it up so that the direction of the blue arrow is the rotation they will face when teleported.

2. Rotate the object so that the blue arrow is facing the direction you want the player to move to.

Now, we just need a script to activate the teleport, and a place to store it in the game. For this example, we can place it on the same gameObject.

3. Create a script called “BasicTeleport”
4. Attach it to TeleportPoint



Teleports

Moving locations

Coding the Teleport

We now have our teleport location, let's code our teleport so that it moves an object when called.

- In the BasicTeleport script
  5. Create a public Transform called teleportLocation
  6. Create a public Transform called targetToTeleport

We can use these to set the location of one, to the other.

  7. Create a void function called Teleport
- In the function
  8. Set targetToTeleports position and rotation to be equal to teleportLocations position and rotation

Now that the teleporting function is done, all we need is a way to call it. For Testing, we can use a key press

- In the update function
  9. Create an if statement that checks if a key is pressed
     a. We chose the E key
- If true
  10. Call the Teleport function

```
public class BasicTeleport : MonoBehaviour {
    public Transform teleportLocation;
    public Transform targetToTeleport;

    0 references
    public void Teleport()
    {
        targetToTeleport.position = teleportLocation.position;
        targetToTeleport.rotation = teleportLocation.rotation;
    }
```

```
private void Update()
{
    if(Input.GetKeyDown(KeyCode.E))
    {
        Teleport();
    }
}
```

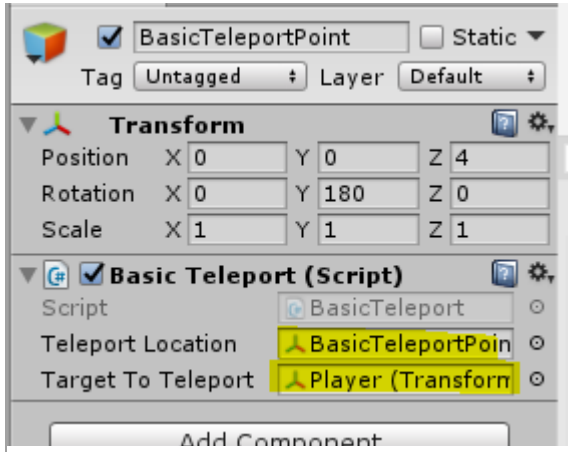
Plugging in the Information

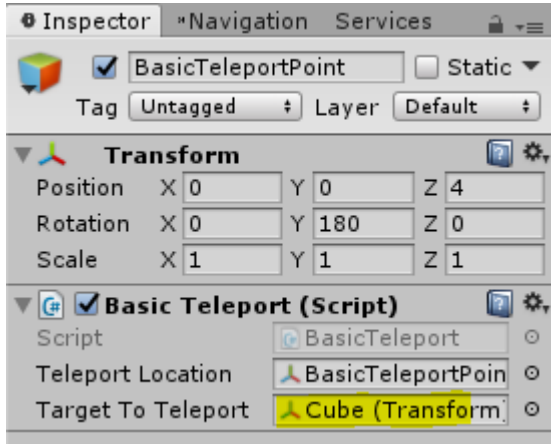
To get our teleport working, we now must pass in the Location and the Target

11. Pass in Teleport location information
12. Pass in what you want to teleport

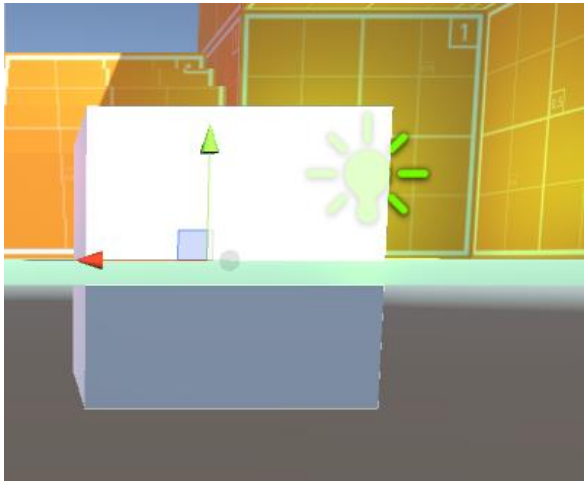
When you test the game, when you press the button, the object you asked to teleport will move to that location.

NOTE: The teleport occurs where the Transform information is, so if the objects transform is in the middle of the object, it will teleport into any objects around it. This is why we make sure that the transform is at the bottom of the object





FIRST IS SETUP TO TELEPORT THE PLAYER, THE SECOND IS SET UP TO TELEPORT A CUBE.



CUBES TRANSFORM IS IN THE MIDDLE OF THE OBJECT, SO WHEN WE TELEPORT THE OBJECT TO THE DESTINATION, IT IS PLACED INSIDE THE FLOOR

Teleporting Objects with Physics.

When you teleport an object with Rigidbody physics. The momentum will not change. This gives us a teleport like in the game portal. “Speedy object goes in, Speedy object comes out”.

This is good in some situations, but sometimes, you just want momentum to reset when teleported.

We can set it up so that when an object teleports we reset velocity

- In the Teleport function
  1. Grab targetToTeleports Rigidbody component
  2. Store in a Rigidbody called target RB

  Now we check to see if we got a Rigidbody
  3. Check if targetRB is not null
- If true
  4. Set targetRBs velocity and angularVelocity to equal Vector3.zero

```
public void Teleport()
{
    targetToTeleport.position = teleportLocation.position;
    targetToTeleport.rotation = teleportLocation.rotation;

    Rigidbody targetRB = targetToTeleport.GetComponent<Rigidbody>();
    if (targetRB != null)
    {
        targetRB.velocity = Vector3.zero;
        targetRB.angularVelocity = Vector3.zero;
    }
```

Now, if an object with Rigid body goes in, it will not keep its momentum going

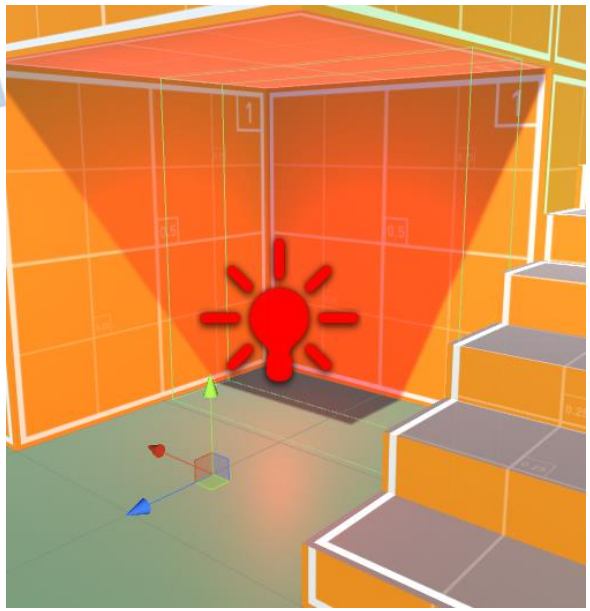
Extra

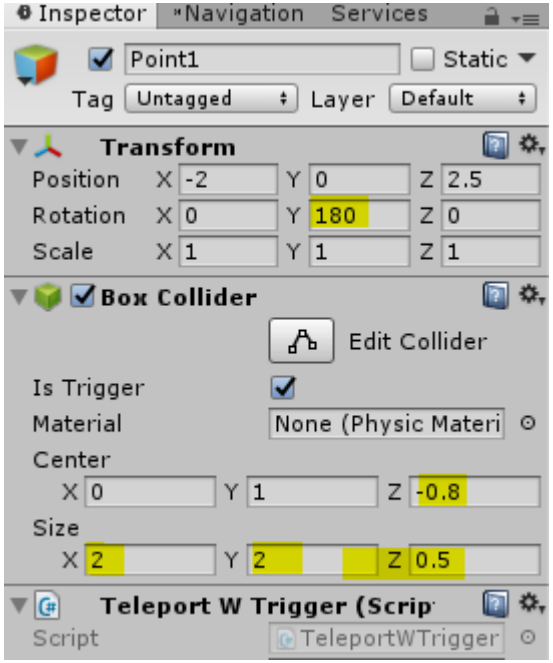
You can do the same with the Players momentum. You need to get players VerticalVelocity to reset if a player enters the teleport.

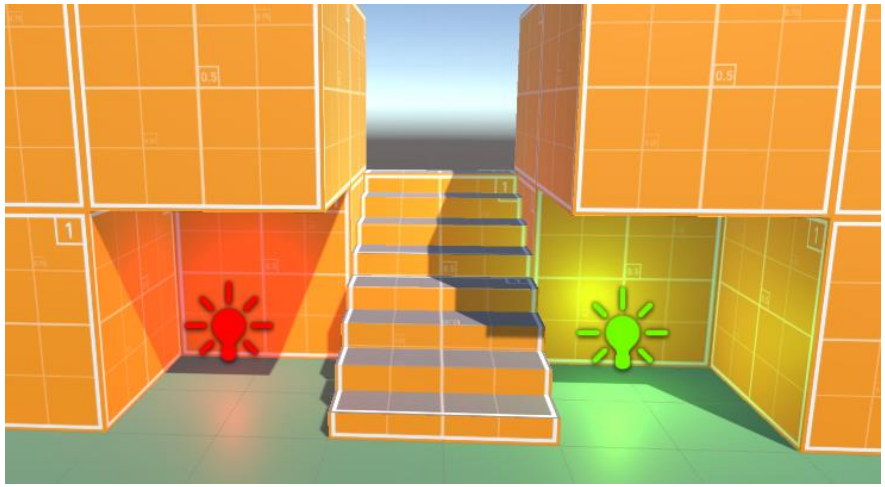
EXERCISE: A TWO-WAY TELEPORT

Now that we have an idea on teleporting, we can create a two-way teleport. When you enter one, you come out the other.

1. Create an empty game object called Point1
2. Make it face the direction you wish to look at when teleported
3. Add a Box collider component
    a. Set it to a trigger
    b. Set it so it is the size you want.
    c. Set it so that it is not on top of the transform position
4. **NOTE: We want the collider further back to the transform point. This is because when we teleport into a trigger it would activate and teleport us again, so we want the trigger away from where we teleport to**
5. Add a way to identify it is a teleport
    a. We added a light
6. Create and add a script called TeleportWTrigger
7. Copy the point and call it Point 2
    a. Position it in the world for the destination







Teleports

Moving locations

Scripting the Teleport

This code is going to be very similar to the previous. However, this time, we are going to add a Trigger volume script, and that will determine what will be teleported.

- in the TeleportWTrigger script
  8. create a public Transform variable called destination – The destination of the teleported object
  9. create a public void function called Teleport
     a. Pass in the parameter Transform target – The target to teleport to the destination
  10. Set it so that the targets position and rotation are the same as destinations

```
public class TeleportWTrigger : MonoBehaviour {
    public Transform destination;

    1 reference
    public void Teleport(Transform target)
    {
        target.position = destination.position;
        target.rotation = destination.rotation;
    }
```

Next we want the OnTriggerEnter event

  11. Create an OnTriggerEnter event
      a. Pass in the Collider other

- In the Trigger Function
  12. Call Teleport
  13. Pass in others transform
  14. Save and Test

```
private void OnTriggerEnter(Collider other)
{
    Teleport(other.transform);
}
```

If the player doesn't get caught in the trigger volume when teleporting, they should be able to teleport from one point to the other.

3RD PERSON ISSUE, NOT TELEPORTING CAMERA.

One issue you can have with 3rd Person camera and character is that the camera does not teleport with the player. It moves to the player instead. This means that the camera is facing the same direction when you exit the teleport.

If the teleports are facing the same way you will walk into the teleport again and flip between the two teleports.

- In the ThirdPersonCamera Script
  1. Create a public void function called RotateBehind Player

We are going to set lookAngle to be behind the targets using the targets y rotation. We need to set it to eulerAngles so its set to a Vector coordinate system

  2. Set lookAngle to equal target.rotation.eulerAngles.y

For testing, I put a key press to “Reset” the rotation whenever the R key was pressed.

- In the Update function
  3. Create an if statement that checks if the R key is pressed
- If true
  4. Call RotateBehindPlayer

```
public void RotateBehindPlayer()
{
    lookAngle = target.rotation.eulerAngles.y;
}
```

```
void Update()
{
    HandleRotation();
    if (Input.GetKeyDown(KeyCode.R))
        RotateBehindPlayer();
}
```

Teleports

Moving locations

- In the TeleportWTriggers script
  - In the OnTriggerEnter function
    5. Check to see if the triggered object is the "Player"
  - If it is
    6. FindObjectOfType ThirdPersonCamera
    7. And call RotateBehindPlayer

If the ThirdPersonCamera script is in the game it should turn the camera around to face the direction the player is facing.

```
private void OnTriggerEnter(Collider other)
{
    Teleport(other.transform);
    if (other.tag == "Player")
    {
        FindObjectOfType<ThirdPersonCamera>().RotateBehindPlayer();
    }
}
```