DOKUMENTATION VR-PROJEKT

Im MTIN Unterricht war es unsere Aufgabe, ein VR-Spiel zu programmerien und zu präsentieren. Als Zielplattform entschied ich mich für die Oculus Rift.

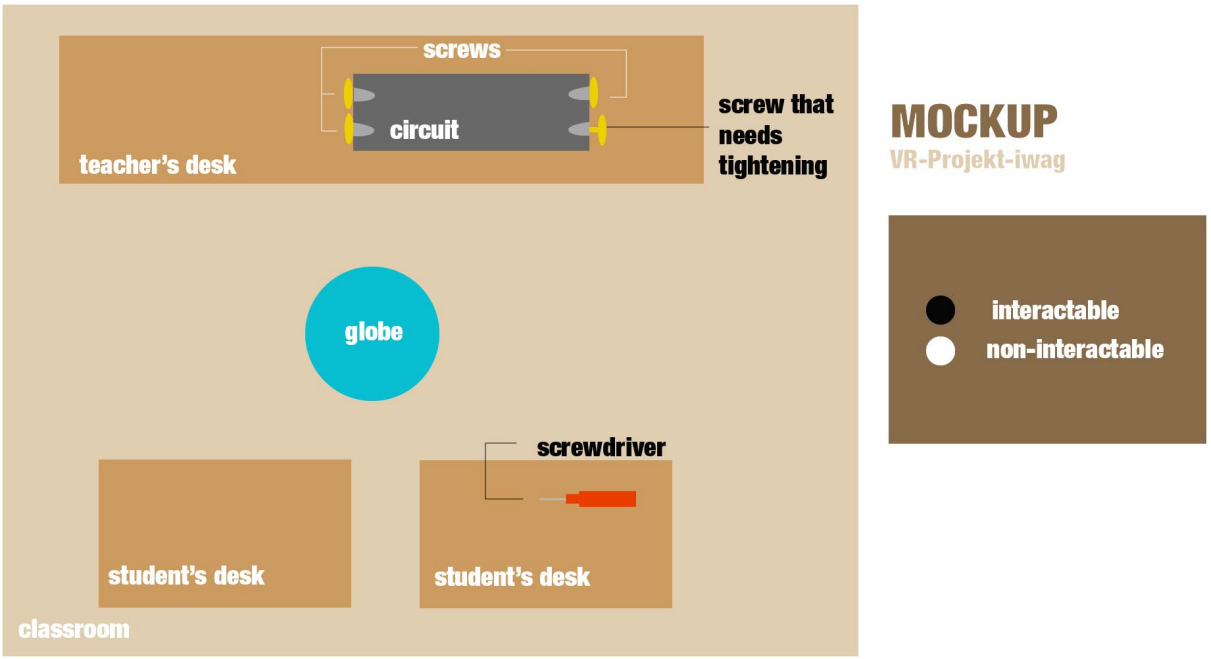
1 – PLANUNG

IDEE

Geplant wurde das Spiel anhand von Begriffen aus einer Wordcloud mit den Anweisungen, dass ein Globus und eine Animation im Spiel vorkommen müsse. Meine Wörter sind "physics", "power" und "search".

Die ursprüngliche Idee war folgende: In der Mitte eines Physikklassenraums steht ein Globus, welcher anfangs unbeleuchtet ist. Auf dem Lehrerpult steht ein elektrischer Kreislauf, der damit verbunden ist, allerdings ist eine Schraube locker.

Die Challenge ist es, im Klassenraum einen Schraubenzieher zu finden, welcher auf einem der Schülertische platziert ist. Diesen muss man grabben und kann dann, in Form einer Drehbewegung mit dem Controller, die Schraube festziehen. Der Globus ist jetzt beleuchtet und das Spiel geschafft.



CONTROLS (planned)

- Press and hold joystick to teleport to any point on the teleport area in the room
- Look around by moving your head with the Oculus Rift headset
- Press and hold the x or a button to grab objects
- Tighten screws by executing a twist motion when in range of a screw whilst grabbing a screwdriver

ENTWICKLUNGSPLATTFORM

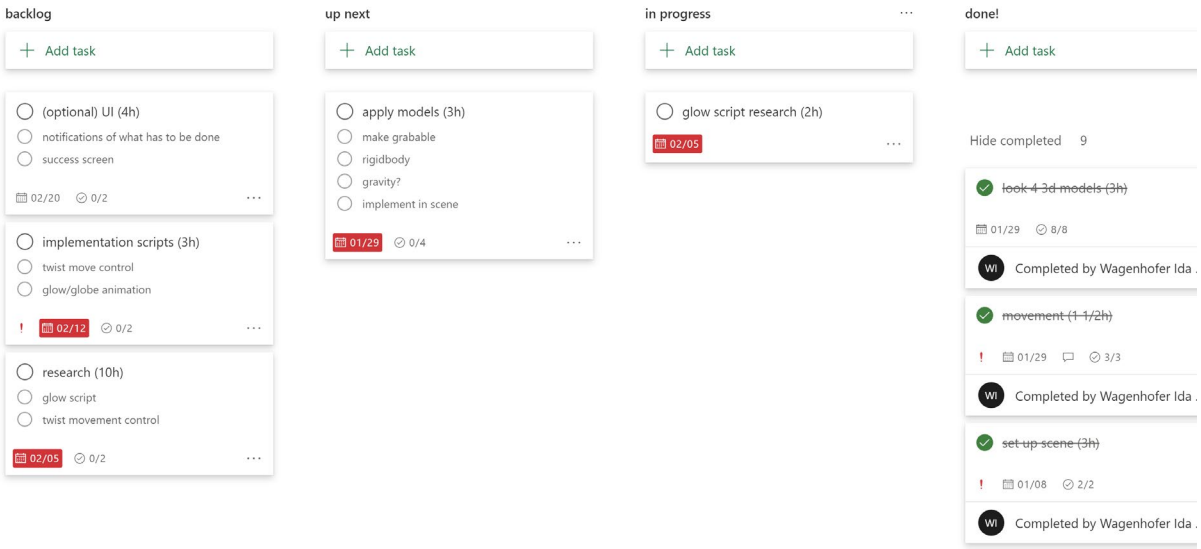
- Windows 10 64 bit, GTX 1070 MaxQ, Intel i7 Hexa-Core
- Unity3D 2019.1.14f, API Compatibility Level .NET Standard 2.0, Scripting Runtime Version .NET 4.x Equivalent
- Visual Studio Community 2019 16.4.0
- Oculus Rift

TARGET PLATFORM

- Oculus Rift

KANBAN BOARD

Jeder der Schüler musste ein Kanban Board auf MS Teams erstellen und darauf die Zeit eintragen, die er/sie vermutlich benötigen würde.



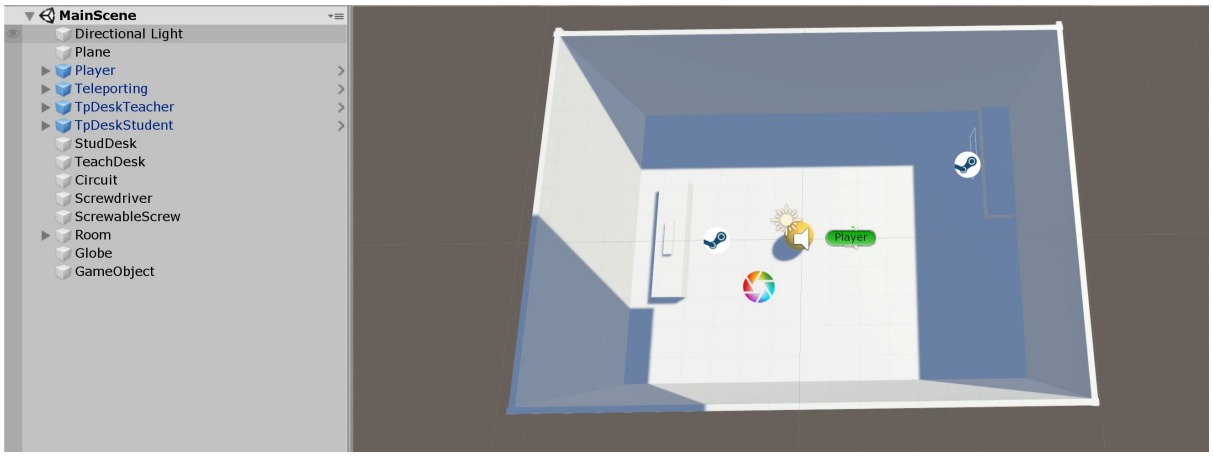
2 – SETUP

Da die Oculus Rift recht ähnlich zur HTC Vive funktioniert, war das Setup nicht all zu schwierig, hat aber dennoch einige Zeit eingenommen, da ich vergessen hatte, wie es funktioniert.

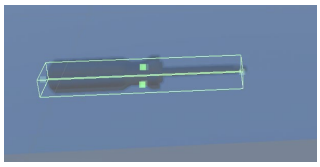
Dafür habe ich folgendes Tutorial verwendet: https://www.youtube.com/watch?v=5C6zr4Q5AlA

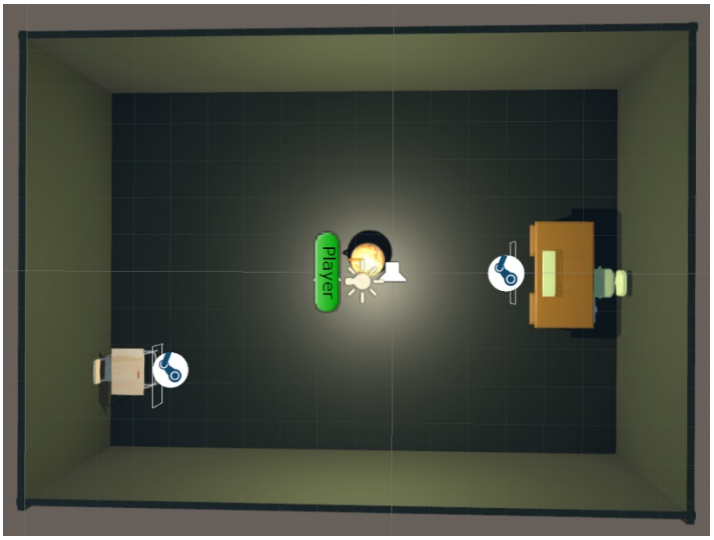
Damit das ganze Projekt mehr „Spaß“ macht, habe ich mich dafür entschieden, die Controller wobbly aussehen zu lassen, was auch diesem Tutorial entnommen ist.

3 – SZENENAUFBAU



Zuerst wurde die Szene aus Whiteboxes aufgebaut, da Models anfangs unwichtig sind und ich mich auf die Interaktion fokussieren wollte.





Am Ende hab ich dann die Whiteboxes noch auf 3D-Models geupdatet und ihnen, wenn notwendig, passende Box Collider gegeben, damit zB der Schraubenzieher nicht durch den Tisch fällt. Die Suche nach passenden 3D-Models bin ich anfangs zu kritisch angegangen und habe mich unnötig lange damit befasst, da die Details der Models eigentlich irrelevant sind.

3.1 LICHTSETUP

Das Licht kommt nur von einem gelblich-getönten Arealight, welches meiner Meinung nach die Stimmung in einem Klassenzimmer simuliert. Dieses ist in der Mitte des Raums, über dem Globus, platzierrt, damit dieser am stärksten beleuchtet ist, weil ja dort die finale Änderung passiert.

4 – PROGRAMMIEREN

Das Programmieren empfand ich als den schwierigsten Part, da ich mir meine Interaktionen leichter vorgestellt hatte, als sie letztendlich waren.

Der Plan war, dass der Globus aufleuchten sollte, das Problem war, dass ich mich davor noch nie mit Post Processing Effekten auseinander gesetzt hatte und daher nicht wusste, dass, wenn man versucht, mit Bloom etwas „glowen“ zu lassen, der Effekt auf alle Objekte in der Szene angewendet und nicht begrenzbar ist. Das fand ich allerdings erst heraus, nachdem ich mir mehrere Tutorials angesehen und Dokumentationen durchgelesen hatte bzw. mich mit Renderpipelines beschäftigt hatte.

Ich versuchte also anfangs trotzdem, mit dem Effekt zu arbeiten, in dem ich die Intensität des Blooms niedrig einstellte und nur die Emission des Materials des zu leuchteden Objekts erhöhte, was allerdings auch nicht funktionierte.

An diesem Punkt hatte ich bereits etwa 5h in diesen Punkt investiert, obwohl ich ihn letztendlich verwerfen musste. Als Alternative dazu ist jetzt im Spiel implementiert, dass sich das Material der Kugel des Globus ändert, anstatt, dass er, wie geplant, aufleuchtet.

Ähnlich ging es mir mit meiner Idee, die Schraube durch ein die Rotation des Schraubenziehers in der Hand, also im Controller, festzuschrauben.

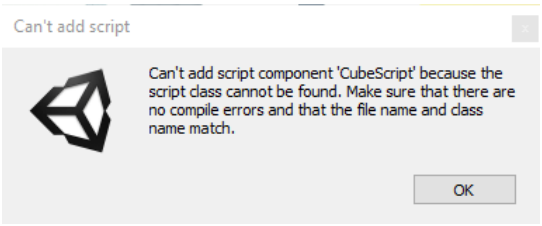
Auch die Interaktion mit dem Schraubenzieher hat nicht wie gewünscht funktioniert, da ich, aufgrund meiner Entscheidung, einfachheitshalber nur zwei Teleportpoints anstatt einer Teleportarea zu machen, nicht wollte, dass er throwable ist, aber doch den grabbable Aspekt eines Throwables gebraucht habe. Daher habe ich versucht, von einem Tutorial (https://www.youtube.com/watch?v=vhRKKAngvAA) den Code zu verwerten, und dafür lt. Tutorial das Setup gemacht, was dann jedoch nicht funktionierte, da das Script Compile Time Errors enthielt und ich es, da es nicht mein Script war und im Tutorial nicht erklärt wurde, nicht verstand. Daher nun die Entscheidung, meinen Schraubenzieher trotzdem throwable zu machen und, wenn der Spieler ihn wegwirft, hat er eben Pech gehabt.

Auch die Twist Movement Control Idee habe ich verworfen, da dies meine Programmierfähigkeiten um einiges überstiegen und der Aufwand dafür sich nicht gelohnt hätte. Stattdessen löste ich es einfach mit einem Collider, wenn also der Schraubenzieher die Schraube berührt, wird die Materialänderungsfunktion ausgelöst.

Letztendlich hielt ich mich dann an folgendes Tutorial mit kleinen Abänderunegn:
https://www.youtube.com/watch?v=dJB07ZSiW7k

```
using UnityEngine;

0 references
public class ChangeMaterialScript : MonoBehaviour
{
    public Material[] material;
    private Renderer rend;

    // Start is called before the first frame update
    0 references
    void Start()
    {
        rend = GetComponent<Renderer>();
        rend.enabled = true;
        rend.sharedMaterial = material[0];
    }

    0 references
    void OnCollisionEnter (Collision col)
    {
        if (col.gameObject.tag == "screwable")
        {
            rend.sharedMaterial = material[1];
        }
    }
}
```

```
using UnityEngine;

0 references
public class ChangeMaterialScript : MonoBehaviour
{
    public Material[] material;
    public MeshRenderer rend;

    // Start is called before the first frame update
    0 references
    void Start()
    {
        rend.sharedMaterial = material[0];
    }

    0 references
    void OnCollisionEnter (Collision col)
    {
        if (col.collider.CompareTag("screwable"))
        {
            rend.sharedMaterial = material[1];
        }
    }
}
```



Im Tutorial wird beschrieben, wie bei einer Collision das Material eines Objekts geändert wird. Zum Verwalten der Materials wird ein Array verwendet. Auch hier hatte ich aber ein Problem und zwar,bekam ich, als ich das Script auf meine ScrewableScrew ziehen wollte, den Error, dass dies nicht ginge, weil das Script nicht auffindbar/einen falschen Klassennamen habe bzw. man zuerst alle Compiler Errors fixen solle. Verwirrt davon suchte ich vergeblich nach Compiler Errors, dabei hatte ich nur den Namen des Scripts im Nachhinein geändert und der Klassen- und Scriptname stimmten nicht überein. Das bemerkte ich jedoch natürlich erst nach einiger Suchzeit.

Das Skript habe ich dann noch insofern überarbeitet, dass ich den private Renderer zu einem public MeshRenderer geändert habe, damit ich im Inspector auswählen konnte, welches Objekt damit verändert wird.

Die Mechanik ist also, dass, wenn der Screwdriver mit einem Objekt mit dem Tag „Screwable“ collidet, sich das Material vom 1. Im Array zum 2. Im Array ändert. Allerdings erst, sobald der Screwdriver losgelassen wird.

5 – OPTIONALES

Von dem, was ich mir optional vorgenommen hatte (einem Text, der einem sagen würde, was zu tun ist), habe ich nichts umgesetzt, da ich schon genug Aufwand hatte, das Spiel so zu finalisieren wie es jetzt ist. Der Grund dafür ist, dass ich in der Anfangsphase des Projekts aufgrund einer OP am Handgelenk erst stationär im Krankenhaus war und daher nicht daran arbeiten konnte und in den darauffolgenden 3 Wochen meine linke Hand aufgrund von Schmerzen beim Tippen nicht verwendbar war.

6 – ENDERGEBNIS

Im Grunde bin ich zufrieden mit meinem Projekt, da es allen Anforderungen entspricht (ein animierter Globus; der/die SpielerIn kann sich fortbewegen; der/die SpielerIn kann mit einem Objekt interagieren; das Spiel funktioniert in VR und lässt sich auf dem definierten Device spielen; das Projekt wurde mitdokumentiert). Nur die pünktliche Abgabe wurde gesundheitsbedingt nicht eingehalten.