VR JÁTÉK FEJLESZTÉS REHABILITÁCIÓS CÉLRA

KERTÉSZ DOMONKOS

TÉMAVEZETŐ: DR. GUZSVINECZ TIBOR

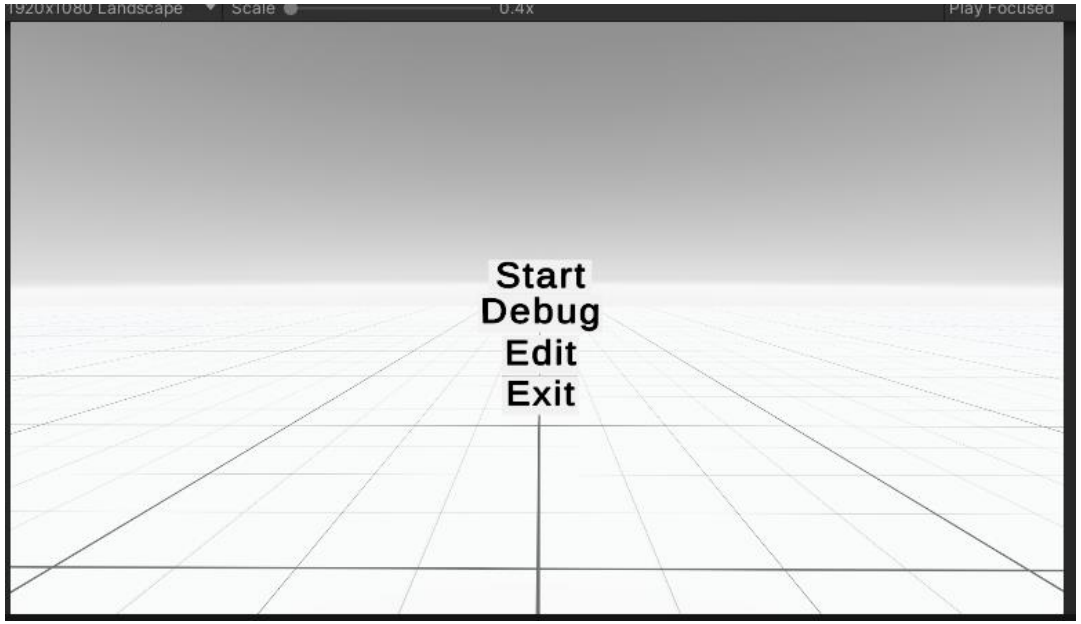
CÉL

- Nyak tornáztató applikáció
- Torna készítő
- Multiplatform (PC/VR)

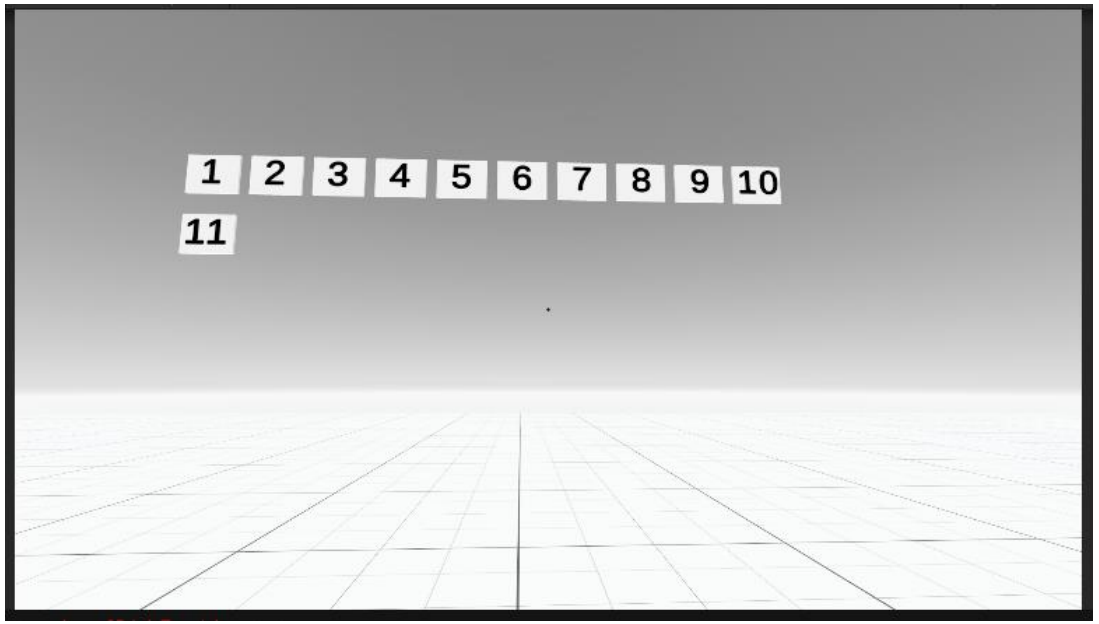
TECHNOLÓGIÁK

- Samsung GearVR – Könnyű, Android alapú, nem igényel PC-t
- Unity Engine – Multiplatform build opció
- Jet Brains Rider – IntelliSense, Unity integráció
- GithHub

Menü rendszer



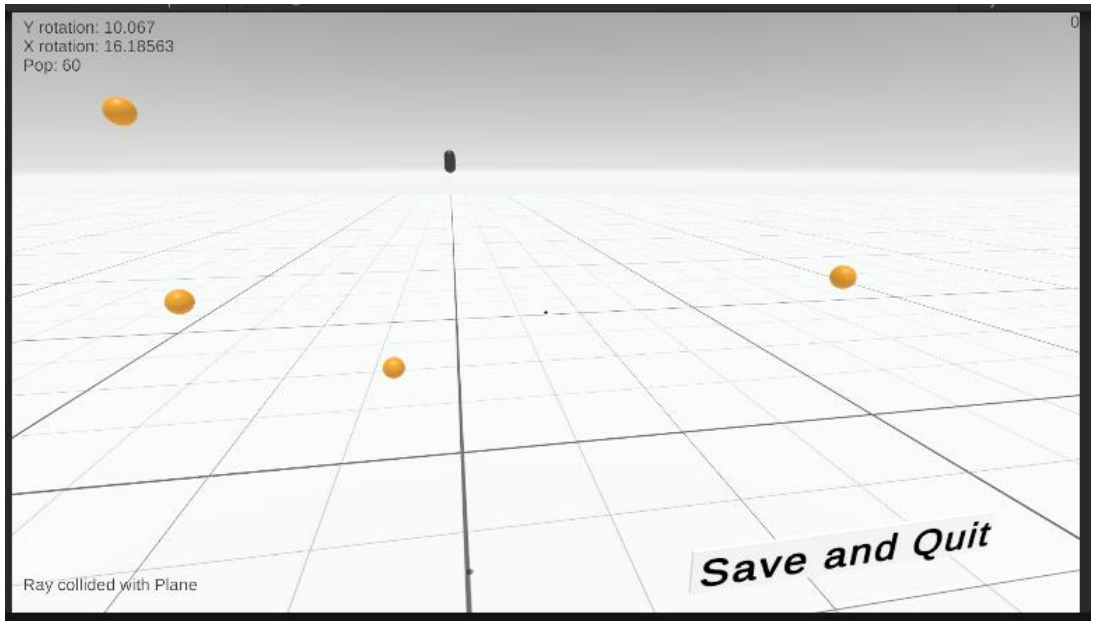
Torna választó



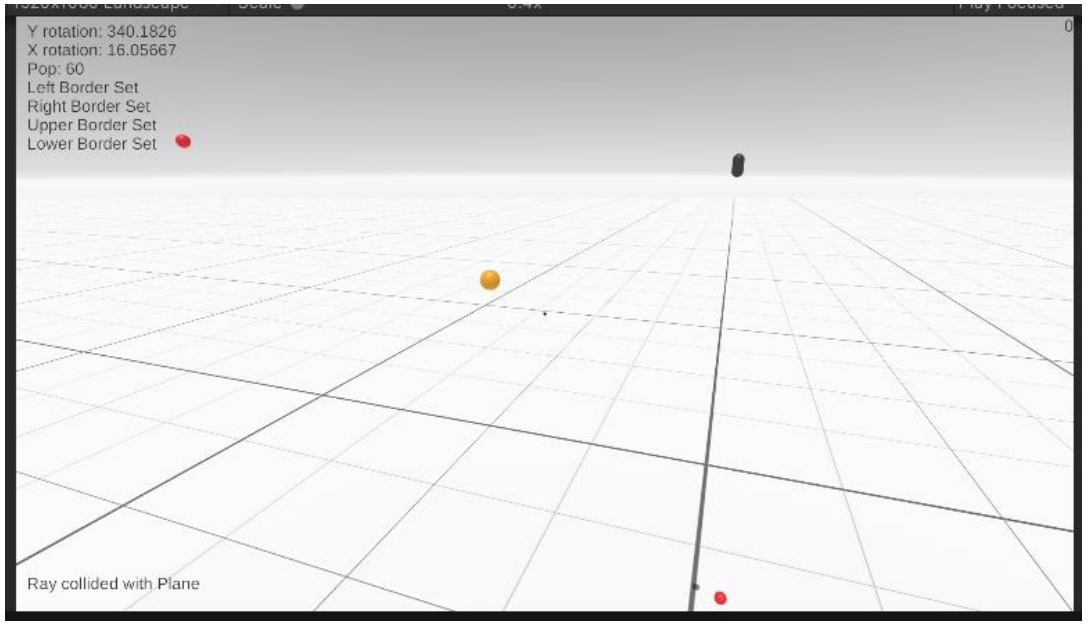
MENÜK

- Nem a UI része, mivel VR-ben nincs kurzor
- Térben lévő objektumok, raycasting segítségével "kattintunk"
- Dinamikusan hozzuk őket létre és töröljük őket (tudnunk kell a pozíciót, és összerakni az objektumot)

Torna készítő mód



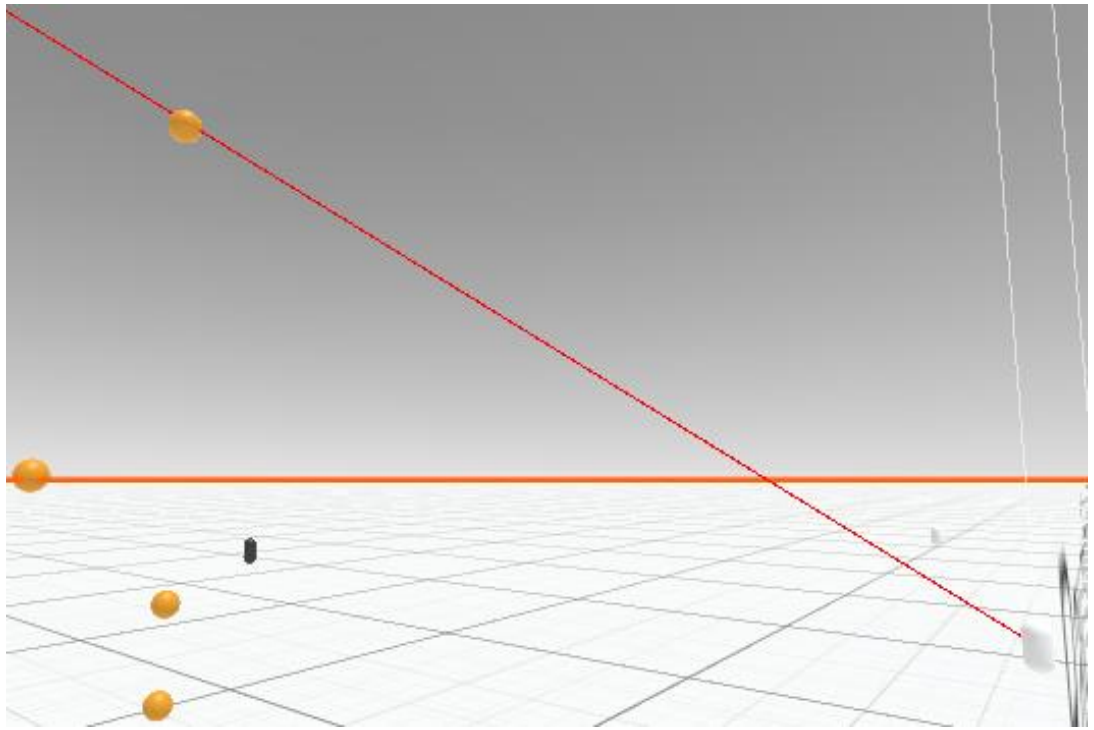
Játék mód



- Objektumok “lerakása” Torna készítőben

```
Vector3 position = PlayerCamRotation.Instance.CameraPosition + PlayerCamRotation.Instance.ForwardVector * ObjectCoordinates.Instance.SpawnDistanceFromPlayer;
```

- Kamera pozíciója + kamera irány vektora * Távolság (Jelenleg 60 egység de módosítható)



```
Ray ray = cam.ViewportPointToRay( pos: new Vector3( x: 0.5f,  y: 0.5f,  z: 0f));
Vector3 rayDirection = ray.direction * 100f;
if (_ui.debug.Enabled)
{
    Debug.DrawRay( start: ray.origin,  dir: rayDirection, Color. red);
}
if (!Input.GetMouseButtonDown(0)) return;
if (Physics.Raycast(ray.origin, rayDirection, out RaycastHit hit))
{
    GameObject o = hit.collider.gameObject;
    _ui.debug.RaycastDebugText = "Ray collided with " + o.name;
    switch (o.name)
    {
        case "DebugClickBox":
            _game.SwitchDebugMode();
            return;
        case "StartClickBox":
```

JÁTÉK MÓD

- A játék sárga gömbökkel vezet végig a tornán
- Raycasting-al tudjuk hogy ránézett-e a felhasználó a megfelelő gömbre

- Futás közbenegy List<List<Vector3>> tárolja a tornákat, egy Singleton osztály tárolja a listát és a segéd függvényeket
- Tornák tárolása JSON formátumban történik két futás között
- Saját megoldás szerializálásra és deszerializálásra

```
public string SerializeVectorListToJson(List<Vector3> vectorList)
{
    StringBuilder sb = new StringBuilder();
    sb.Append("{\"vectors\":[");

    for (int i = 0; i < vectorList.Count; i++)
    {
        Vector3 v = vectorList[i];
        sb.Append("{\"x\":");
        sb.Append(v.x.ToString( format: "F3"));
        sb.Append(",\"y\":");
        sb.Append(v.y.ToString( format: "F3"));
        sb.Append(",\"z\":");
        sb.Append(v.z.ToString( format: "F3"));
        sb.Append("}");

        if (i < vectorList.Count - 1)
        {
            sb.Append(",");
        }
    }

    sb.Append("]}");

    return sb.ToString();
}
```

- JSON-ból Vector3 Listába LINQ segítségével

```
public List<Vector3> DeserializeJsonToVectorList(string json)
{
    int startIndex = json.IndexOf("[{") + 1;
    int endIndex = json.LastIndexOf("}]");

    string[] vectorJsonStrings = json.Substring(startIndex, length: endIndex - startIndex) // string
        .Split( separator: new string[] { "},{" }, StringSplitOptions.RemoveEmptyEntries);

    return (from vectorJson :string in vectorJsonStrings
        select vectorJson.Replace( oldValue: "{", newValue: "")
            .Replace( oldValue: "}", newValue: "") // string
            .Split( separator: new char[] { ',' }, StringSplitOptions.RemoveEmptyEntries) // string[]
        into components :string[]
        let x :float = float.Parse(components[0].Split( separator: ':')[1])
        let y :float = float.Parse(components[1].Split( separator: ':')[1])
        let z :float = float.Parse(components[2].Split( separator: ':')[1])
        select new Vector3(x, y, z)).ToList();
}
```

- Objektumok helyének lekérése a Vector3 listából
- Pozícióhoz prefab hozzárendelése
- Objektum elmentése

```
private void NextObject()
{
    _game.DestroyObject(_currentObject);
    Vector3? nextPos = GetNextVector();
    if (nextPos.HasValue)
    {
        _currentObject = _game.SpawnObject(spherePrefab, nextPos.Value);
        return;
    }
    EndExercise();
}
```

```
private Vector3? GetNextVector()
{
    if (_positions == null || _index >= _positions.Count)
    {
        _index = 0;
        return null;
    }
    Vector3 nextVector = _positions[_index];
    _index++;
    return nextVector;
}
```

fin