UNITY - 1

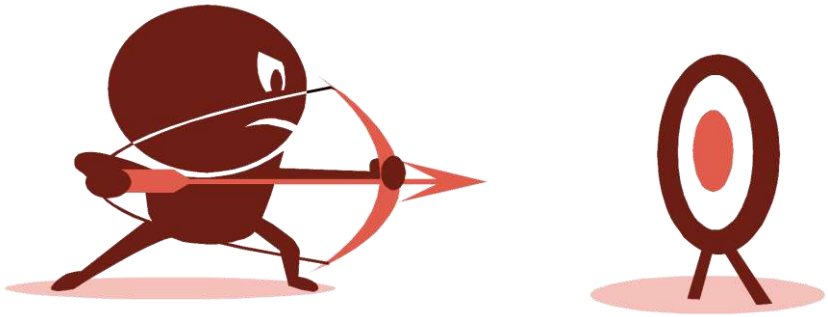
Ing. Timotej Sobota
doc. Ing. Branislav Sobota, PhD.

Katedra počítačov a informatiky, FEI TU v Košiciach

C 07

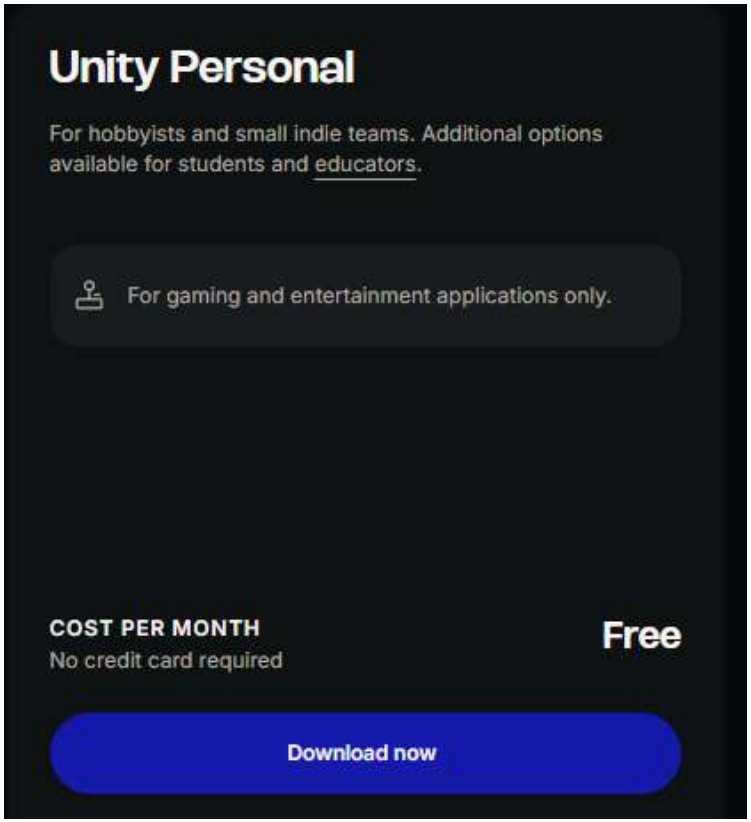
Cieľ cvičenia

- Nastavenie pracovného prostredia
- Oboznámenie sa s pracovným prostredím
- Načítanie modelov (SketchUp)
- Vytvorenie jednoduchej aplikácie



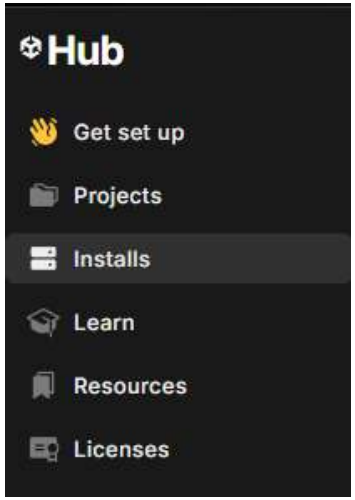
INŠTALÁCIA UNITY

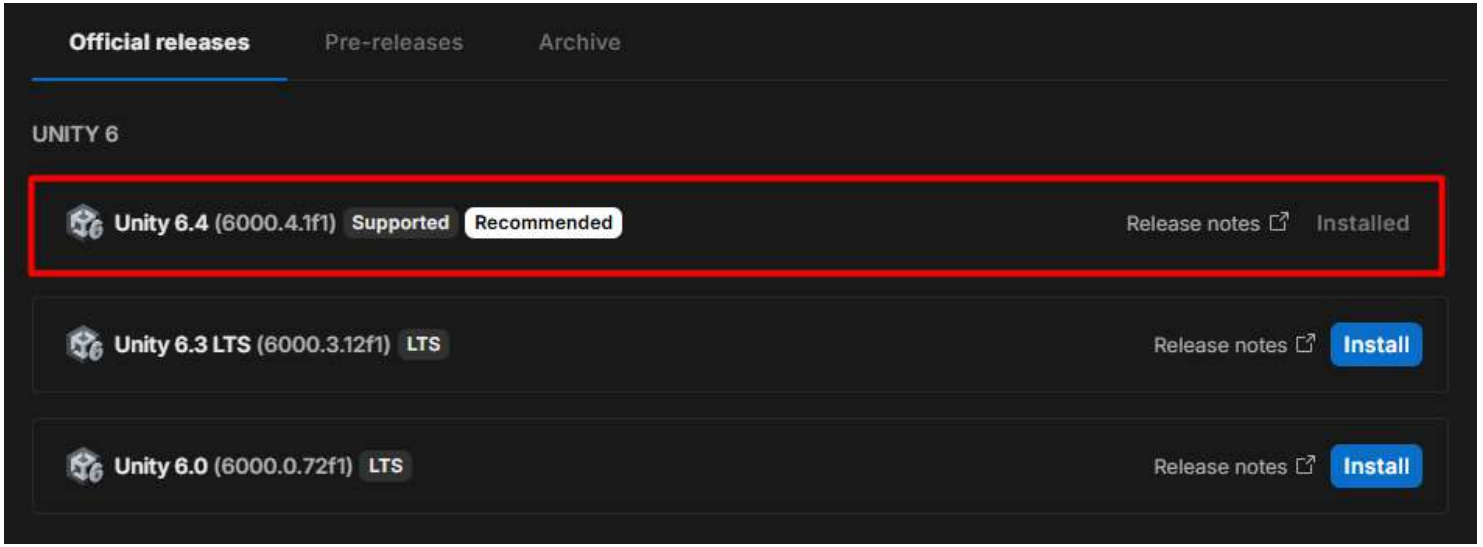
- **Unity Hub**: https://unity.com/products
- **Registrácia**: https://login.unity.com/en/sign-up



Inštalácia Unity

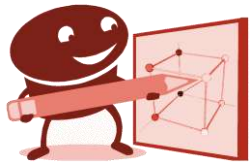
- Inštalácia **Unity Hub** a registrácia na **unity.com**
- Stiahnutie vývojového prostredia **6000.4.1f1**





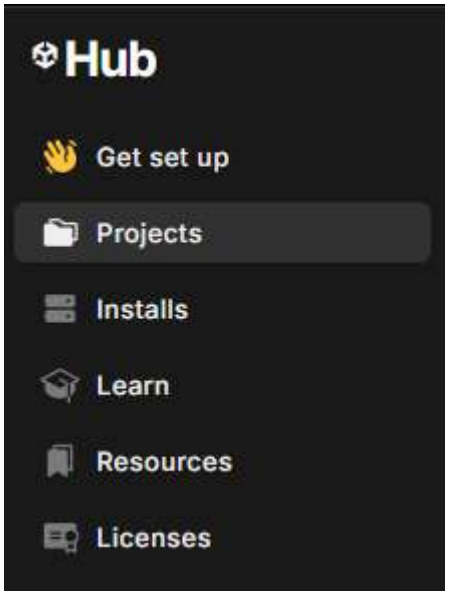
UNITY - VLASTNOSTI

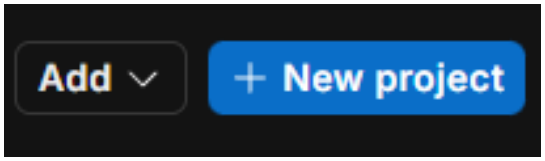
- C#
- Vhodný pre developing mobilných hier
- Templaty 2D/3D
- Store s ukážkami/modelmi
- Podpora viac ako 20 platforiem
- Intuitívne grafické rozhranie
- Veľká aktívna komunita ľudí
- https://unity.com/use-cases

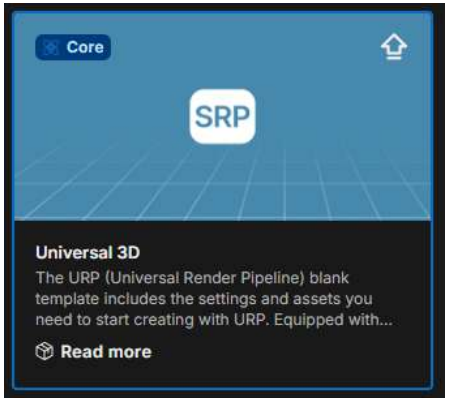


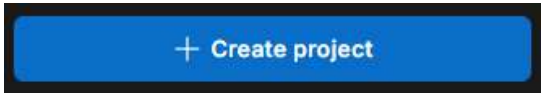
VYTVORENIE NOVEJ APLIKÁCIE

1. Projects
2. New project
3. Universal 3D
4. Nastavenie názvu a umiestnenia projektu
5. Create project



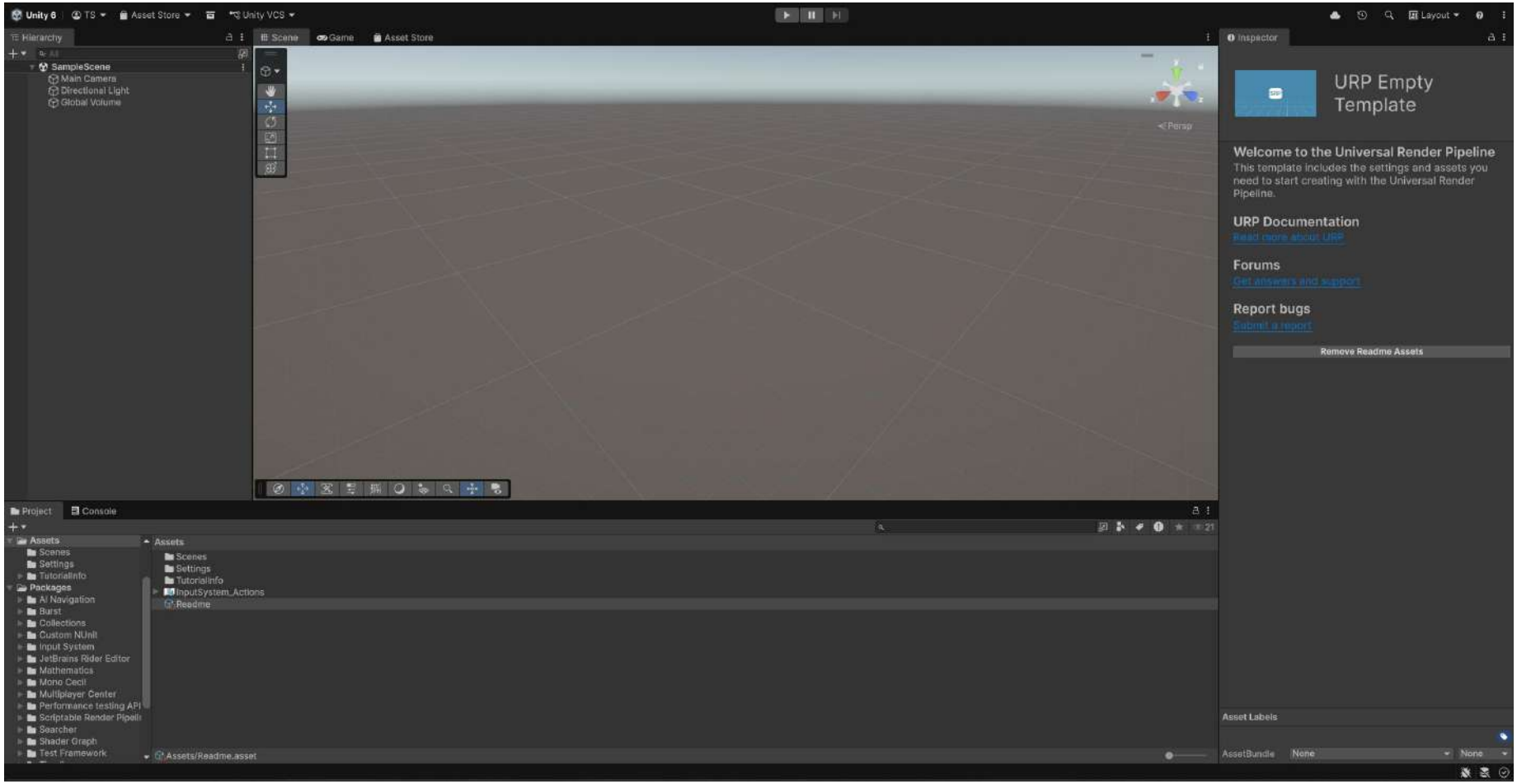






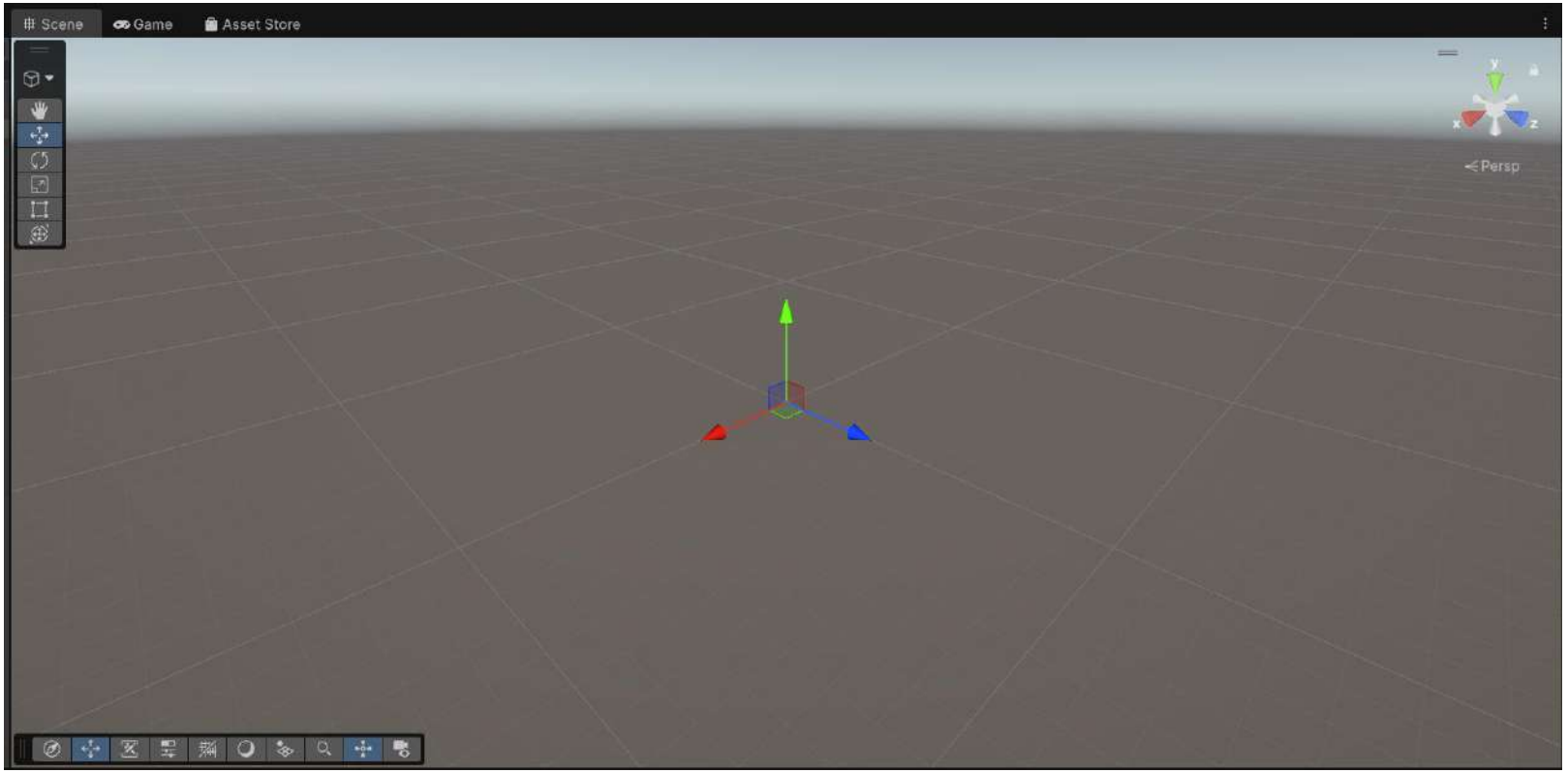
UNITY PROSTREDIE / LAYOUT

- https://docs.unity3d.com/6000.2/Documentation/Manual/urp/urp-introduction.html



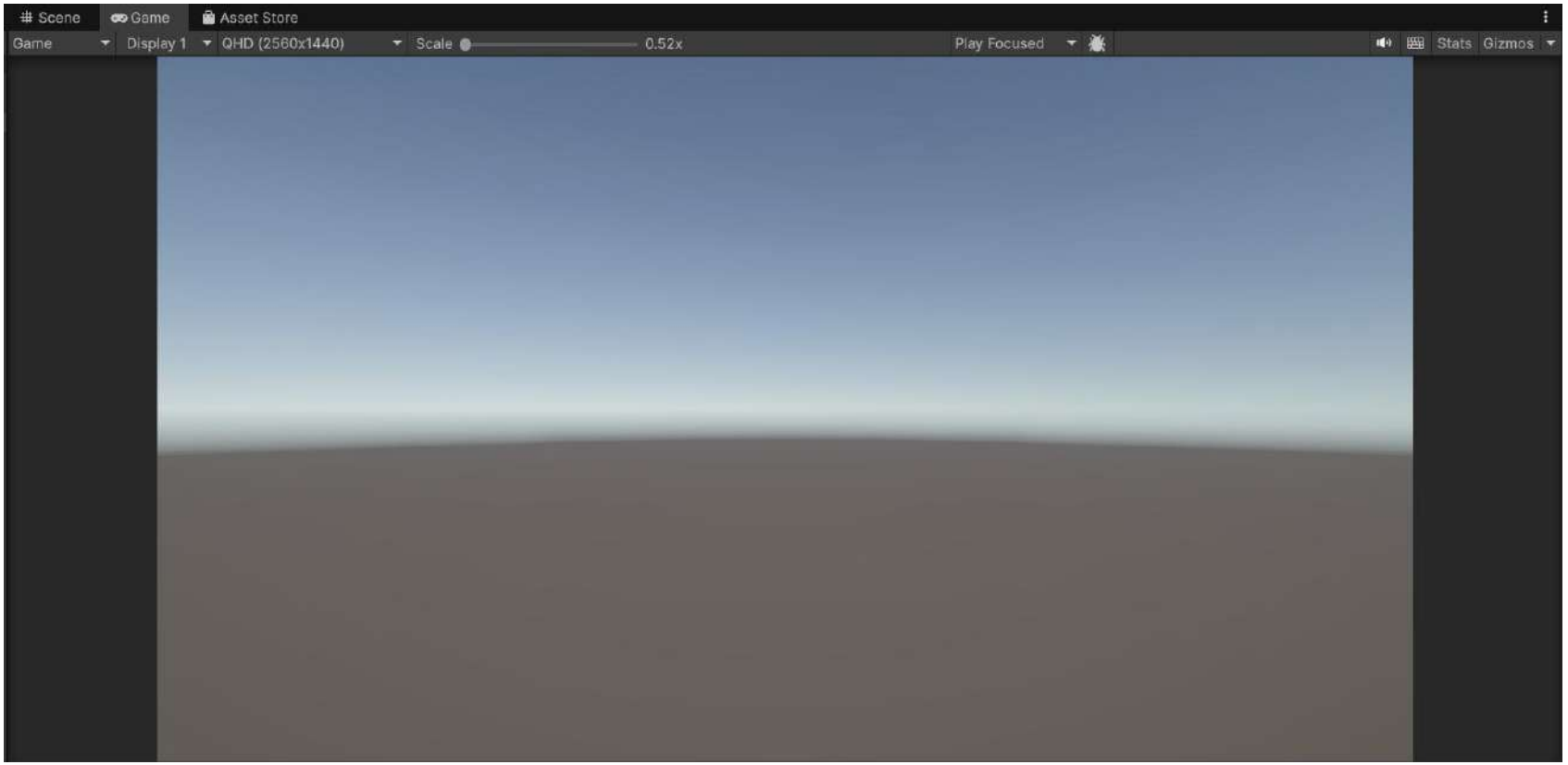
UNITY PROSTREDIE / LAYOUT

- **Scene**
  - pracovný priestor (editor)
  - tvorba a úprava objektov



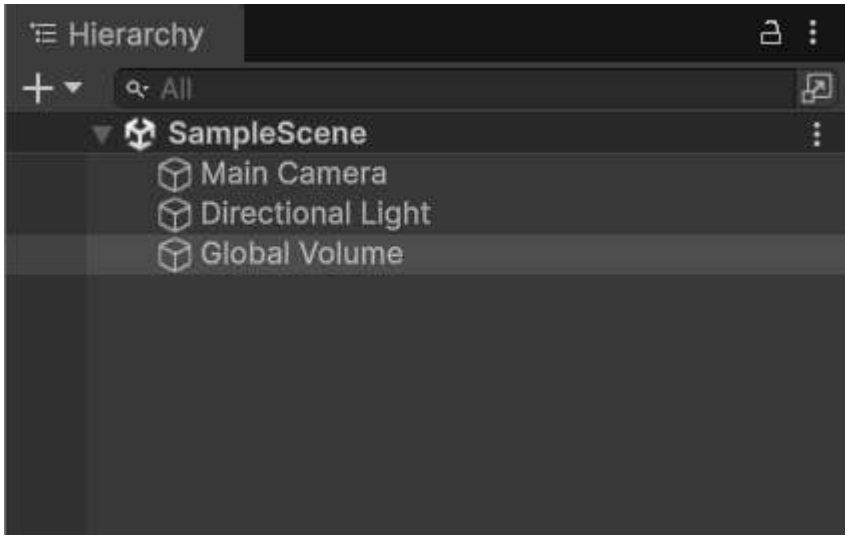
UNITY PROSTREDIE / LAYOUT

- **Game**
  - pohľad hráča (kamera)
  - výsledný výstup hry



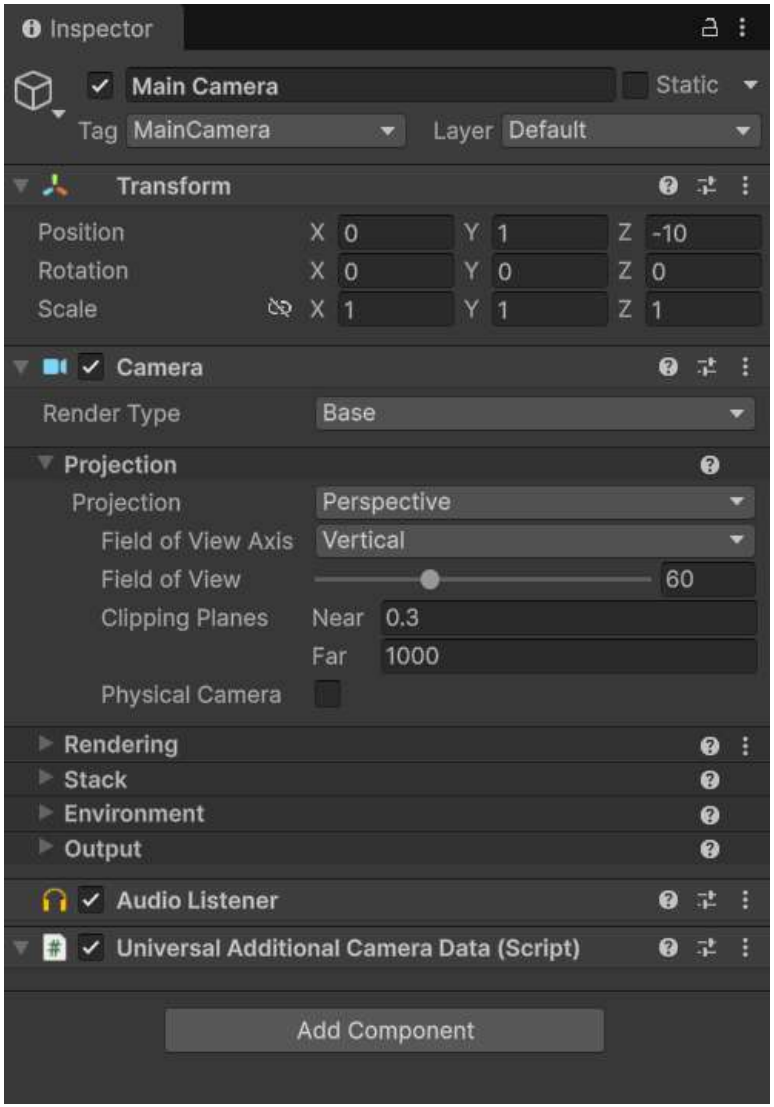
UNITY PROSTREDIE / LAYOUT

- **Hierarchy**
  - zoznam všetkých objektov v scéne
  - organizácia (parent/child)



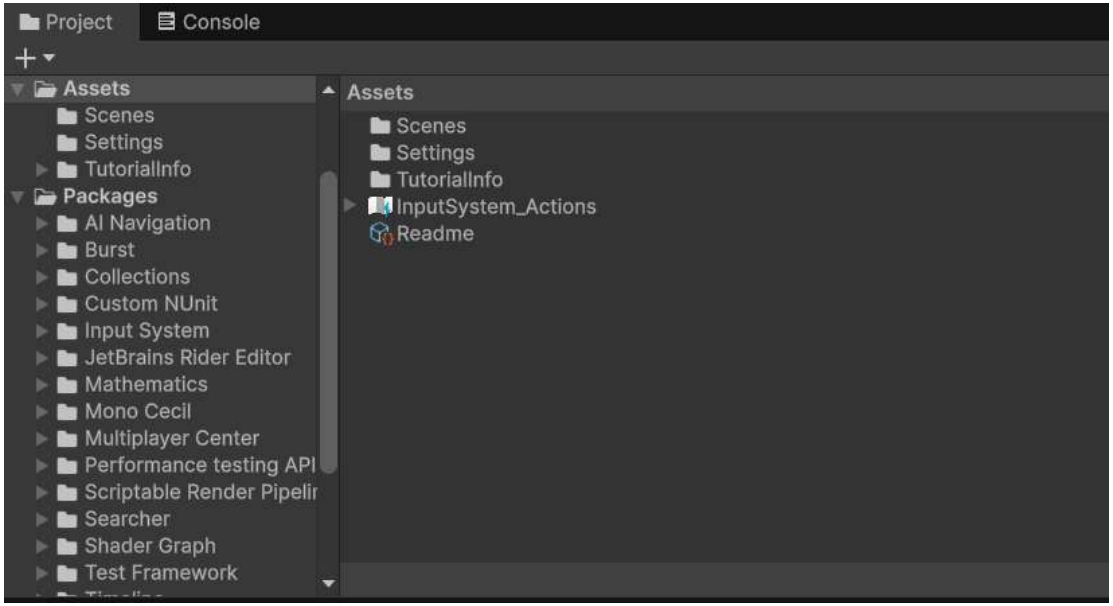
UNITY PROSTREDIE / LAYOUT

- **Inspector**
  - vlastnosti objektu
  - pozícia, rotácia, scale
  - komponenty



UNITY PROSTREDIE / LAYOUT

- **Project**
  - všetky súbory projektu
  - modely, textúry, skripty

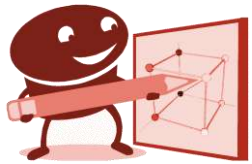


UNITY PROSTREDIE / LAYOUT

- **Toolbar**
  - Hand Tool
  - Move Tool
  - Rotate Tool
  - Scale Tool
  - Rect Tool
  - Move, Rotate or Scale

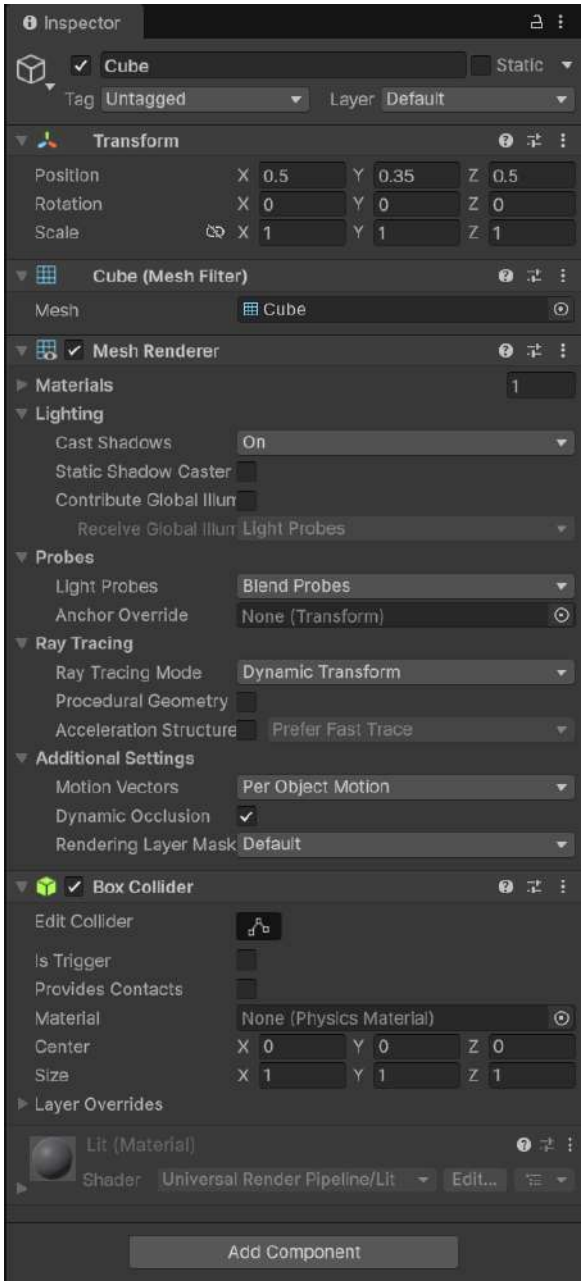
VYTVÁRANIE JEDNODUCHÝCH OBJEKTOV

1. Hierarchy
2. +
3. 3D Object
4. Cube



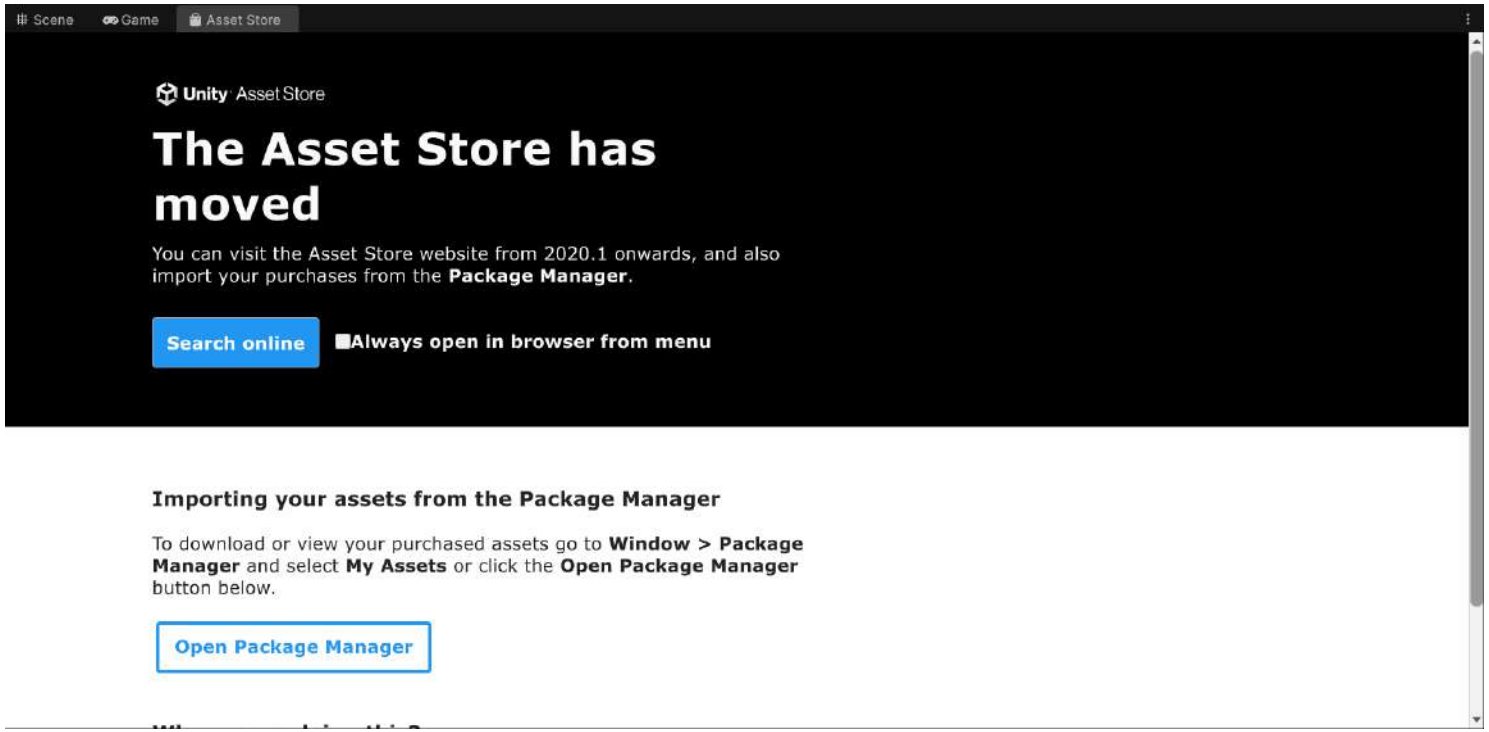
OBJEKT - INSPECTOR

- Štandardne v pravej časti okna
- Sú tu definovane všetky komponenty, ktoré má objekt priradene a aj ich vlastnosti
- Nové komponenty sa dajú pridávať pomocou „Add component“ a taktiež vo väčšine prípadov vo vývojovom prostredí funguje metóda “drag & drop“



IMPORTOVANIE OBJEKTOV

- Možnosť do projektu importovať objekty z externých programov (SketchUp)
- Výhodou je tiež dostupnosť obchodu, ktorý má mnoho objektov aj zadarmo



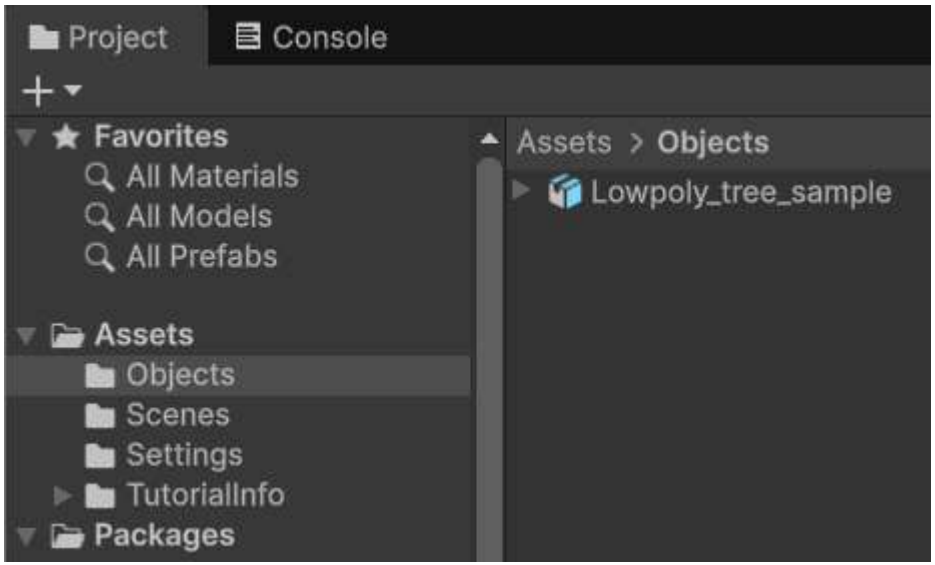
IMPORTOVANIE OBJEKTOV

Úloha:

- Vytvorte v projekte "Assets" priečinok, do ktorého sa budú zhromažďovať modely
- (vzorový model je dostupný aj na moodle)

https://moodle.fei.tuke.sk/mod/url/view.php?id=10750

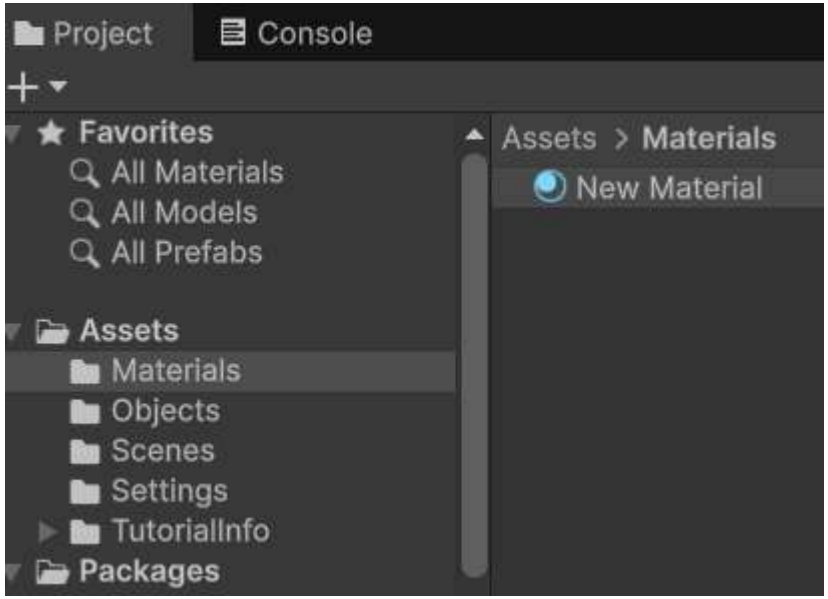
1. RightClick
2. Create
3. Folder

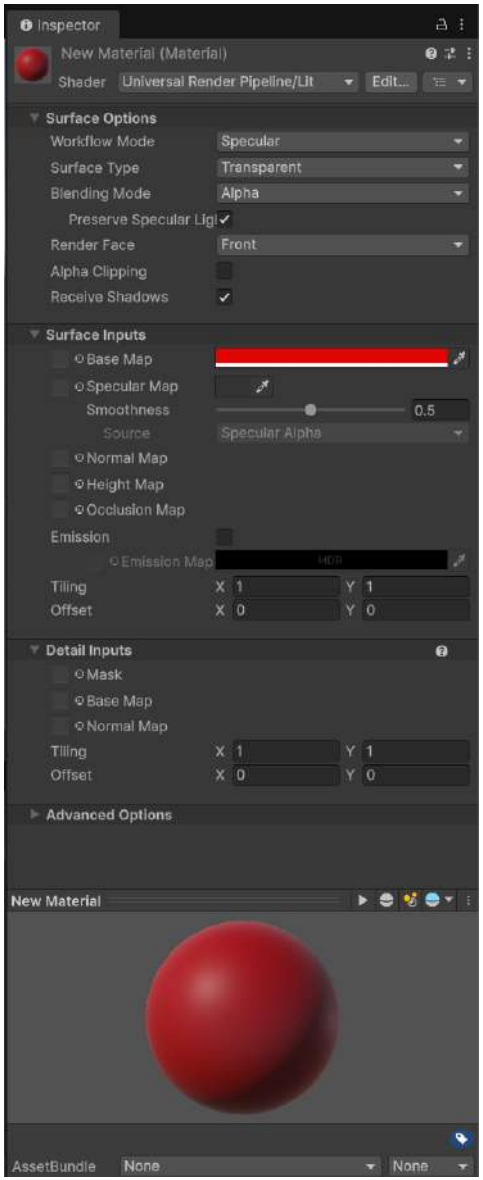


VYTVÁRANIE VLASTNÝCH MATERIÁLOV

- Možnosť vytvárať vlastné materiály priamo cez Unity
- Použitie na objekty

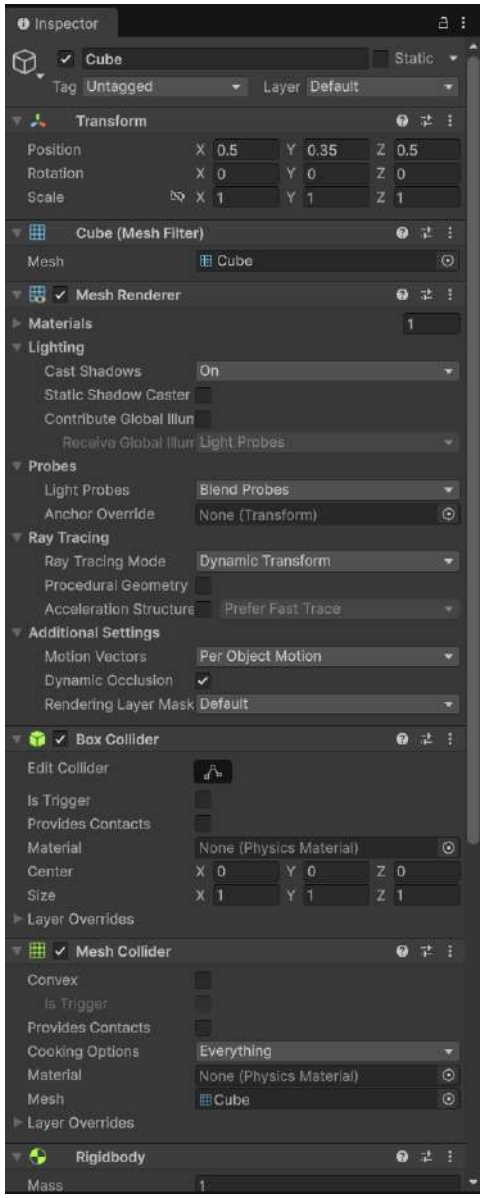
1. RightClick
2. Create
3. Material





Pridávanie Komponentov

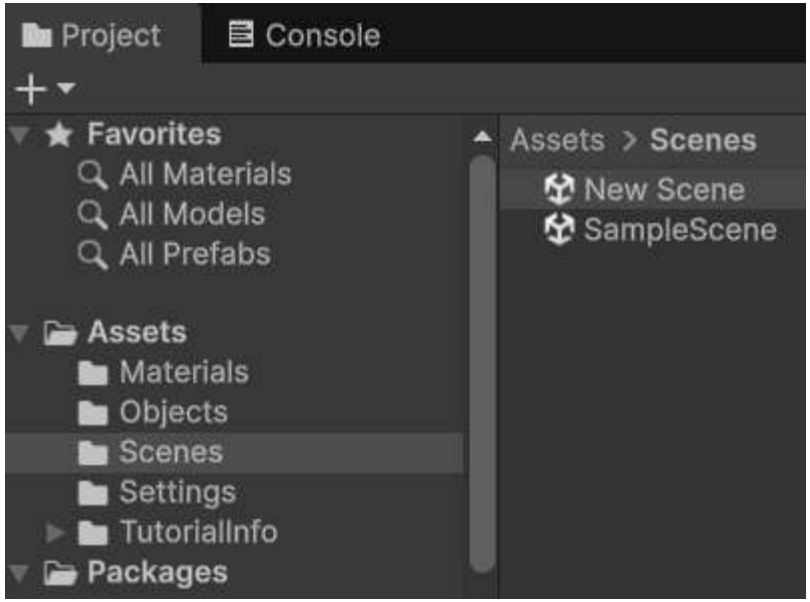
- Transform
  - pozícia (Position)
  - rotácia (Rotation)
  - veľkosť (Scale)
- Mesh Renderer
  - zobrazenie objektu
  - materiály a farby
- Collider
  - detekcia kolízií
  - Box, Sphere, Capsule
- Rigidbody
  - fyzika objektu
  - gravitácia, sily
- Scripts (C#)
  - správanie objektu
  - interakcia, pohyb
- Light
  - osvetlenie scény
  - Directional, Point, Spot
- Camera
  - pohľad hráča
  - renderovanie scény



VYTVÁRANIE NOVEJ SCÉNY

- Použitie scén:
  - Logický kontajner objektov, ktoré môžu rozdeliť aplikáciu do viacerých levelov/sekcii
  - Načítavaním novej scény sa môže uvoľniť pamäť (vymazaním objektov z predošlej scény)

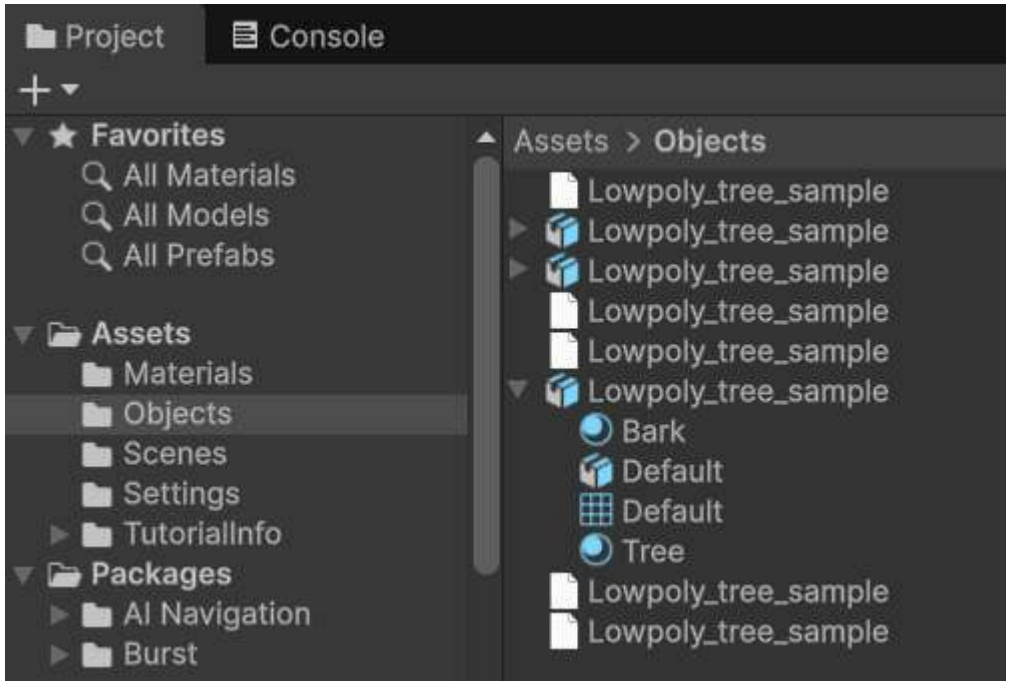
1. RightClick
2. Create
3. Scene



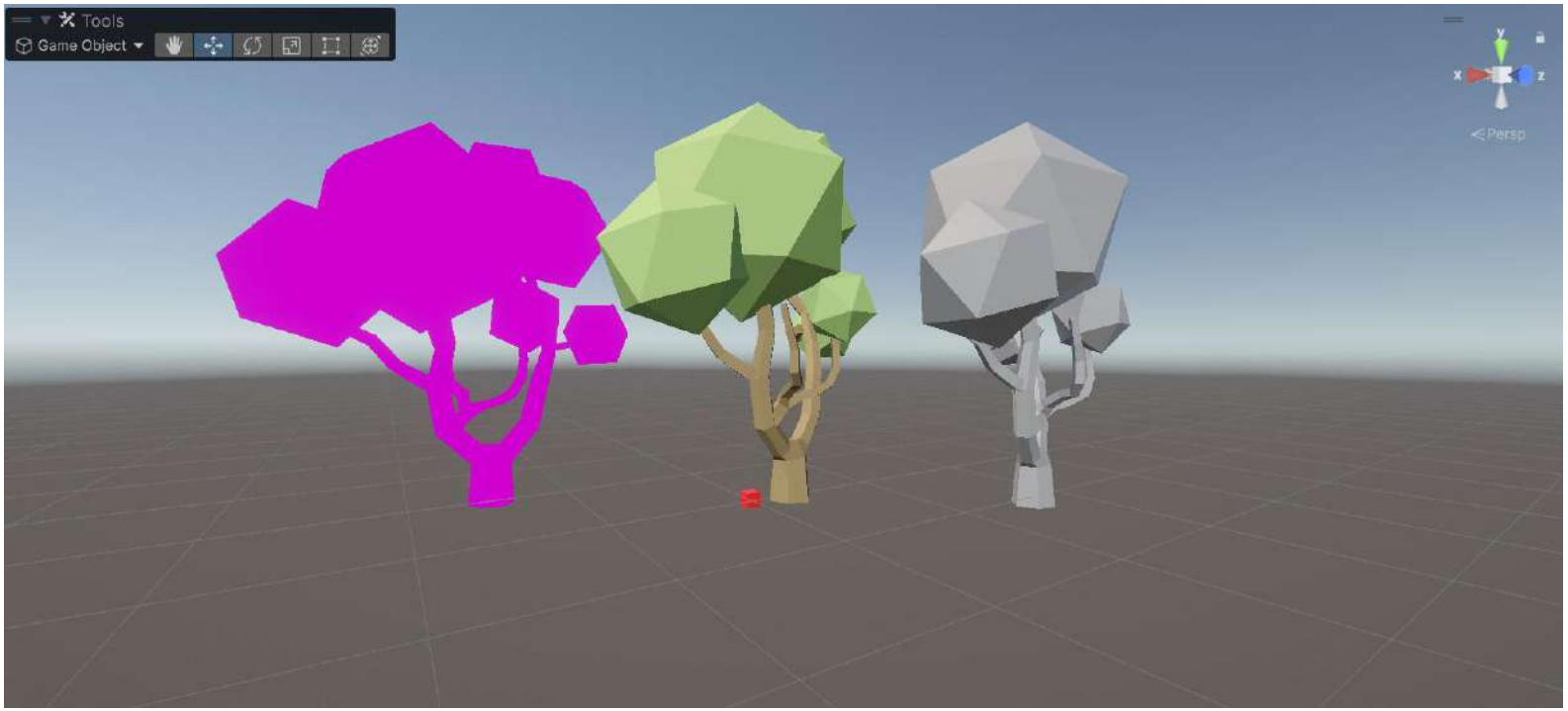
PRIDANIE VLASTNÉHO OBJEKTU NA SCÉNU

Úloha: Pridajte SketchUp objekt do svojej scény

1. **Pomocou "Drag&Drop" vložte objekt na scénu**

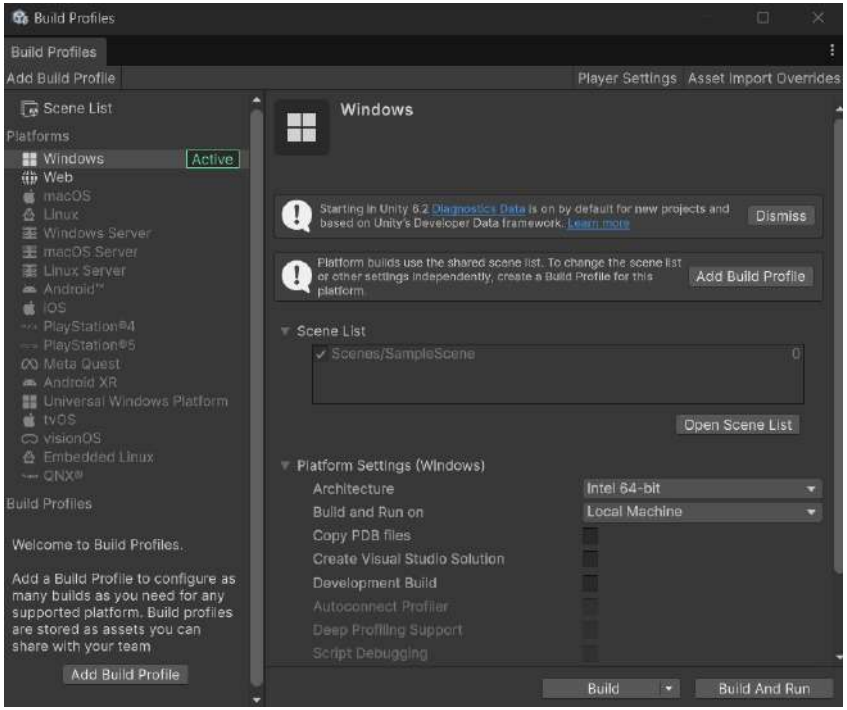


PRIDANIE VLASTNÉHO OBJEKTU NA SCÉNU



BUILD PROJEKTU

1. File
2. Build Profiles
3. Build / Build And Run



DOPLŇUJÚCE ÚLOHY

1. **Vytvorte scénu**, ktorá obsahuje **4 objekty** (napr.: guľa, plocha, 2 kocky) pre objekty vytvorte **rôzne materiály** a použite ich.
2. **Guli** nastavte **gravitáciu**.
3. **Scénu** zostrojte tak, aby guľa pri svojom páde jedným objektom kocky prešla skrz a od druhého sa odrazila až dopadne na plochu, kde sa zastaví.

Q & A

branislav.sobota@tuke.sk
timotej.sobota@tuke.sk

Katedra počítačov a informatiky, FEI TU v Košiciach

UNITY - 2

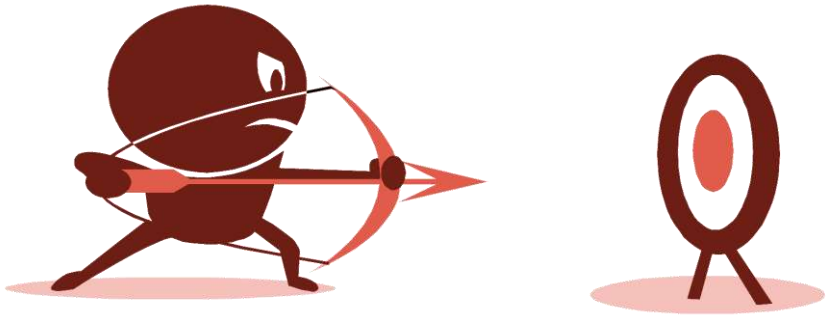
Ing. Timotej Sobota
doc. Ing. Branislav Sobota, PhD.

Katedra počítačov a informatiky, FEI TU v Košiciach

C 09

CIEĽ CVIČENIA

- Zadanie v Unity
- skriptovanie C#
- Input Actions
- Pridávanie tagu
- Vytvorenie jednoduchej aplikácie



Zadanie - Unity

- Naimportované modely (SketchUp / Blender)
  - Vlastné modely sa hodnotia lepšie ako iba stiahnuté
- Dynamika prostredia / pohyb
- Pohyb hráča / kamery
- Box colliders
- Vyplnené prostredie
- Formát odovzdania:
  - Desktop appka
  - Splitscreen appka na telefóne
  - VR headset appka

Skriptovanie v Unity

- **Definuje správanie objektov**
- Používa **jazyk C#**
- Skripty sú **komponenty**
- Pridávajú sa na objekty
- **Funkcie:**
  - **Start()** – spustí sa raz
  - **Update()** – opakuje sa

- **Je potrebné si doinštalovať aj správne Unity balíčky**

Vytvorenie Skriptu

- v **Project**

1. RightClick
2. Create
3. Scripting

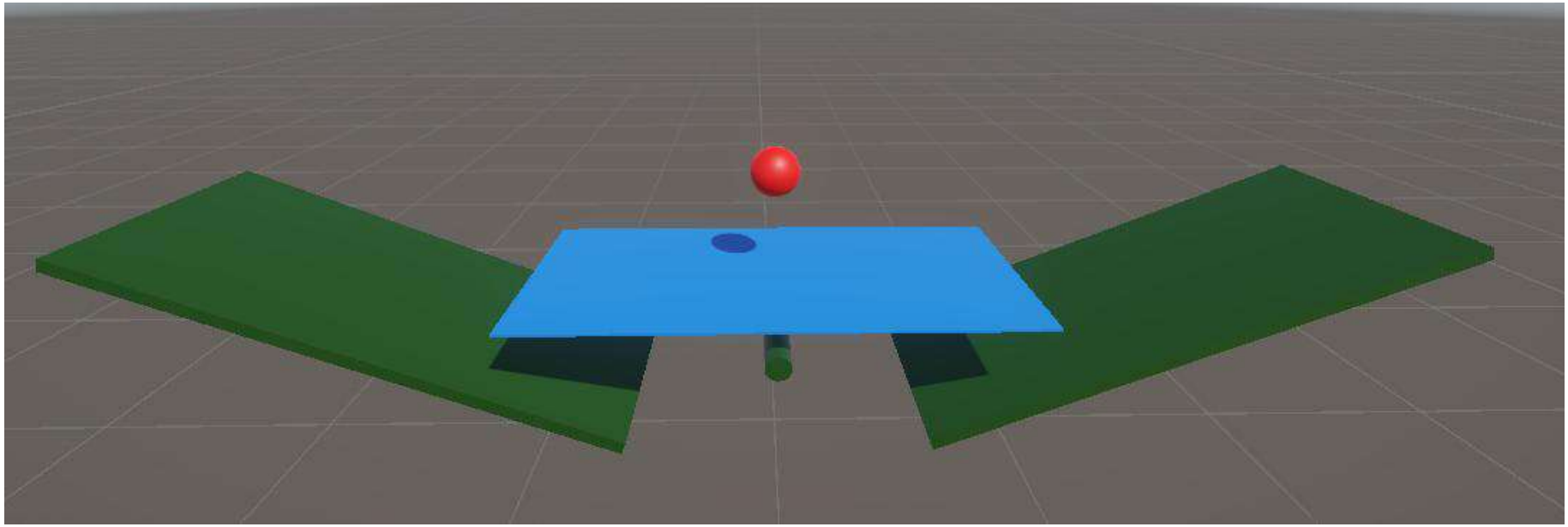
- v **Inspector**-e objektu

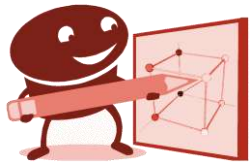
1. Add Component
2. Vypísať názov svojho scriptu
3. Enter

- Drag & Drop -> Inspector objektu -> Add Component

ÚLOHA

- Vytvorenie **novej scény**



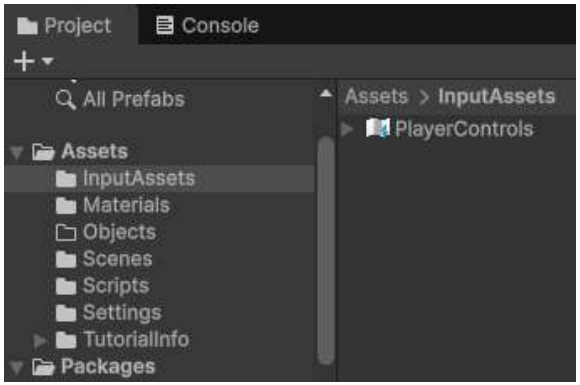


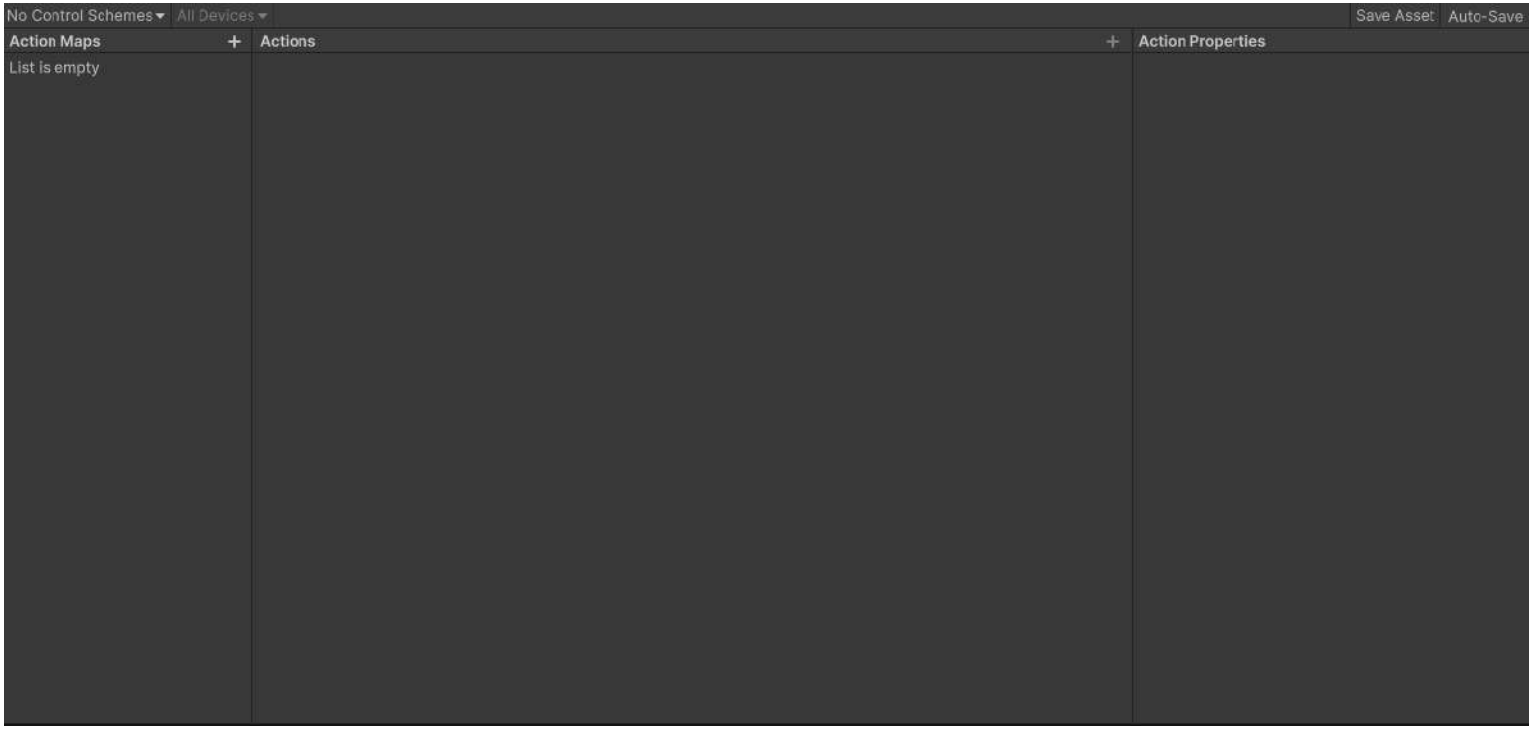
Input Actions (Input System)

- **Nový systém vstupu v Unity**
- Nahrádza starý Input Manager
- **Definovanie vstupov**:
  - **Klávesnica**
  - **Myš**
  - **Gamepad**
- Používa Actions a Bindings
- Ukladá sa ako .inputactions súbor

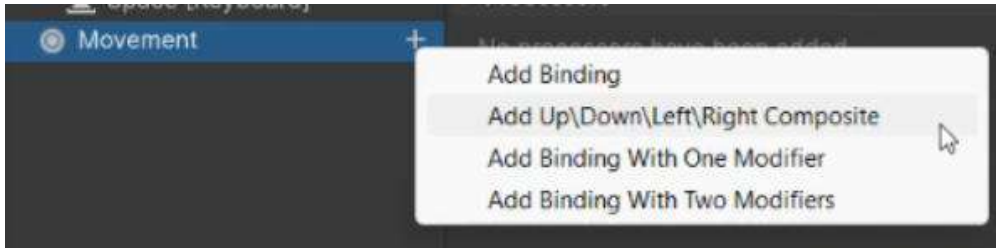
INPUT ACTIONS (INPUT SYSTEM)

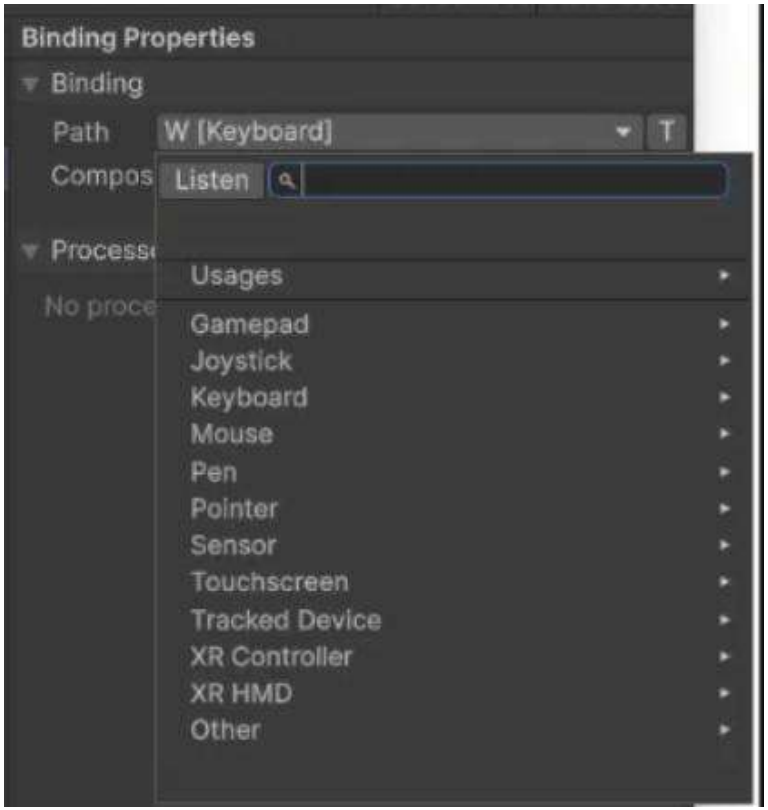
1. RightClick
2. Create
3. Input Actions

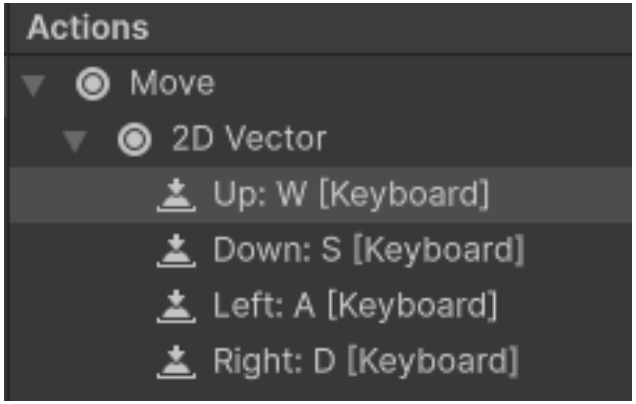


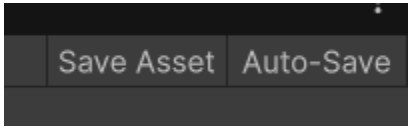


Input Actions (Input System)







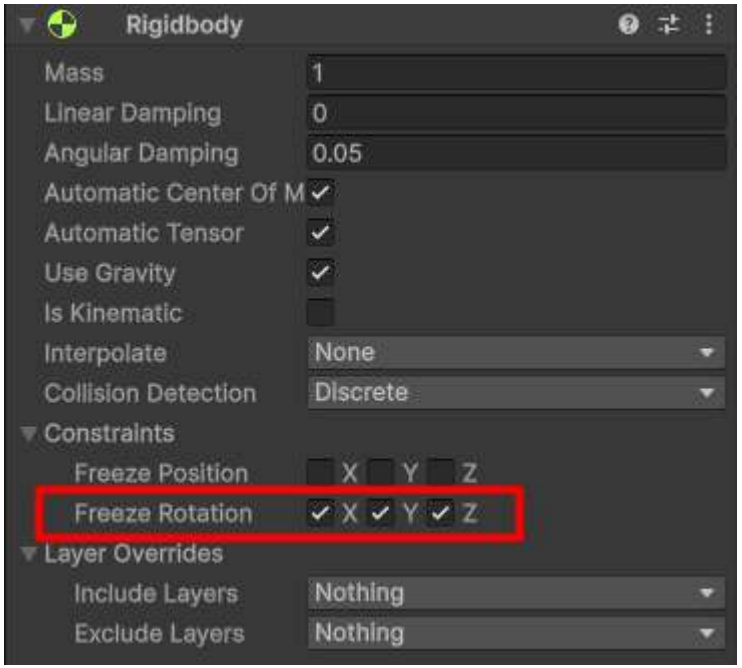


Movement Script

```
using UnityEngine;
using UnityEngine.InputSystem;

public class SphereMovement : MonoBehaviour
{
    public float speed = 5f;
    private Rigidbody rb;
    private Vector2 moveInput;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }

    void FixedUpdate()
    {
        Vector3 movement = new Vector3(moveInput.x, 0f, moveInput.y);
        rb.MovePosition(rb.position + movement * speed * Time.fixedDeltaTime);
    }

    // This MUST match the action name: "Move"
    public void OnMove(InputValue value)
    {
        moveInput = value.Get<Vector2>();
    }
}
```

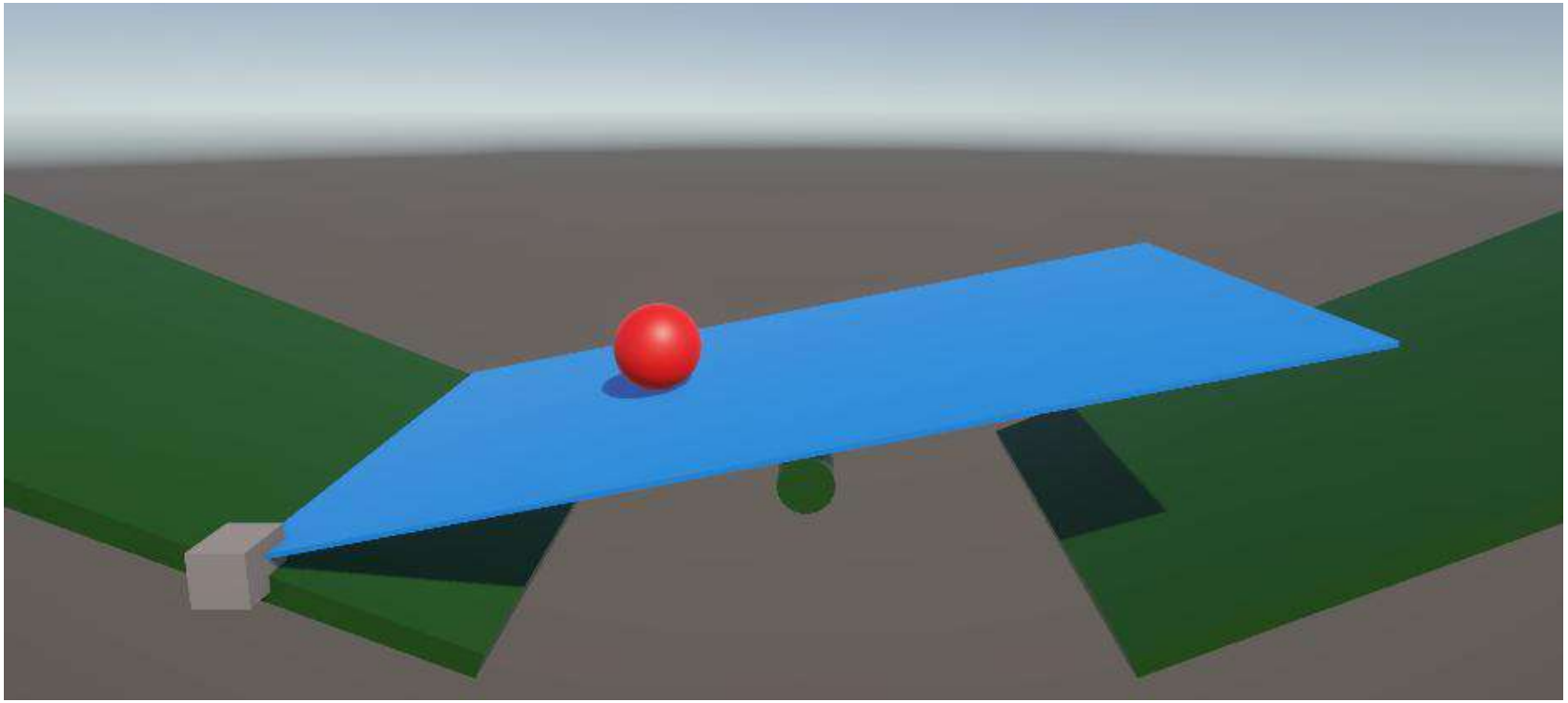
Movement Script

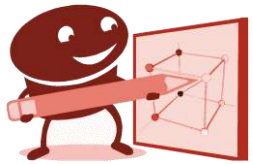
- Overte správanie scriptu



ÚLOHA

- Pridanie detekcie kolízií platforiem



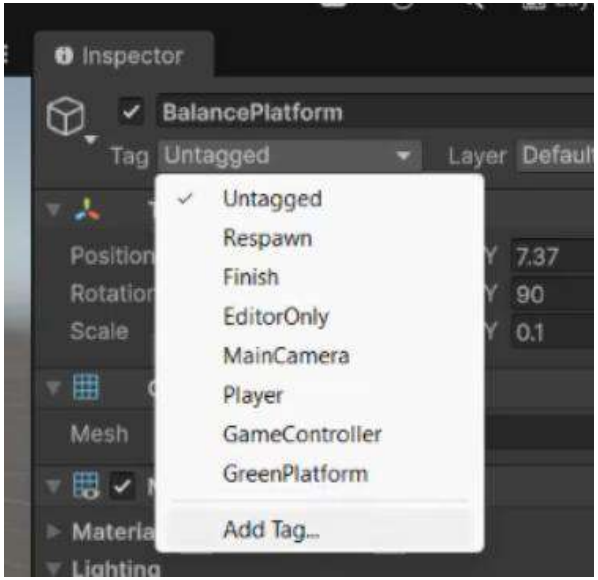


TAGY

- Tag je jednoduchý spôsob, ako si označiť objekty v scéne.
- Tagy používame hlavne v skriptoch, keď chceme:
  - zistiť, **s akým objektom sme sa stretli**
  - reagovať rôzne podľa typu objektu
  - filtrovať objekty (napr. iba nepriatelia)

TAGY

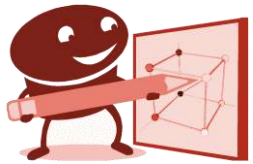
- Ako nastaviť tag v Unity
  - Vyber objekt v scéne
  - V hornej časti **Inspectoru** zvoľ **Tag**
  - Klikni **Add Tag...**
  - Vytvor nový tag
  - Priraď ho objektu
- názov tagu musí byť **presný** (veľké/malé písmená)
- objekt môže mať **iba jeden tag**



ÚLOHA

- Zeleným platformám nastavte tag:

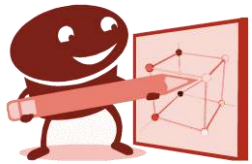
GreenPlatform



ÚLOHA

- Modrej platforme vytvorte skrip pre detekciu kolízie s platformou, ktorá má tag:

GreenPlatform



Collision Script

```
using UnityEngine;

public class SpawnCubeOnCollision : MonoBehaviour
{
    public GameObject cubePrefab;

    void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("GreenPlatform"))
        {
            // Get contact point (where they touched)
            ContactPoint contact = collision.contacts[0];
            Vector3 spawnPosition = contact.point;

            // Spawn cube
            Instantiate(cubePrefab, spawnPosition, Quaternion.identity);
        }
    }
}
```

Úloha

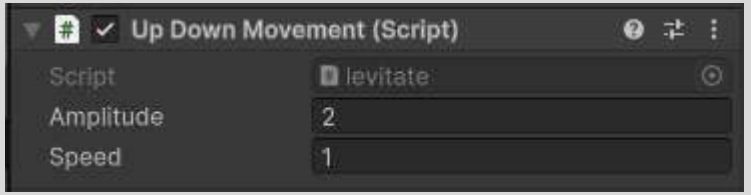
1. Vytvorte na scéne cube objekt
2. Presunte ho z **Hierarchy** do **Project**
3. **Vymažte** objekt cube zo scény (voliteľné)
4. Vyberte si modrú platformu a presuňte prefab kocky do poľa „Cube Prefab“

Collision Script

- Overte správanie scriptu

PRIDANIE DYNAMIKY DO SCÉNY

```
using UnityEngine;

public class UpDownMovement : MonoBehaviour
{
    public float amplitude = 25f;   // how high it goes
    public float speed = 2f;        // how fast it moves

    private Vector3 startPos;

    void Start()
    {
        startPos = transform.position;
    }

    void Update()
    {
        float y = Mathf.Sin(Time.time * speed) * amplitude;
        transform.position = startPos + new Vector3(0, y, 0);
    }
}
```



PRIDANIE DYNAMIKY DO SCÉNY

```
using UnityEngine;

public class Rotate : MonoBehaviour
{
    [Range(1,50)]
    public float speed = 1;

    void Update()
    {
        transform.Rotate(0, 50 * Time.deltaTime * speed, 0);
    }
}
```

ÚLOHY NA SAMOSTATNÉ RIEŠENIE

1. Pridajte možnosť skoku (Jump)
2. Nastavte aby sa rotujúce kocky po určitom čase zničili
3. Zmeňte jednu platformu na inú farbu, pridajte jej iný tag a nastavte aby sa pri kolízii s ňou objavil iný objekt

Q & A

branislav.sobota@tuke.sk
timotej.sobota@tuke.sk

Katedra počítačov a informatiky, FEI TU v Košiciach

UNITY - 3

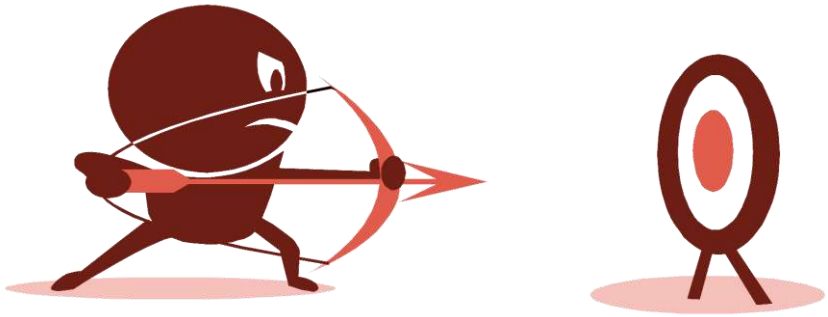
Ing. Timotej Sobota
doc. Ing. Branislav Sobota, PhD.

Katedra počítačov a informatiky, FEI TU v Košiciach

C 10

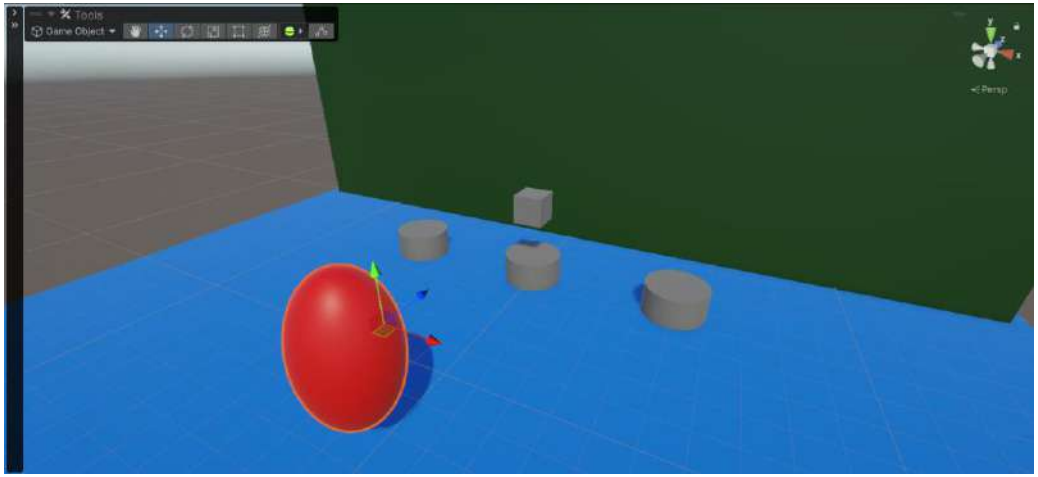
Cieľ cvičenia

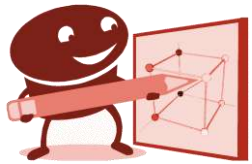
- Pohyb myši
- Look-At mechanika
- Vytvorenie VR aplikácie do telefónu



ÚLOHA

- Vytvorenie **novej scény:**
  - **Podlaha**
  - **Avatar**
  - **Rôzne objekty**

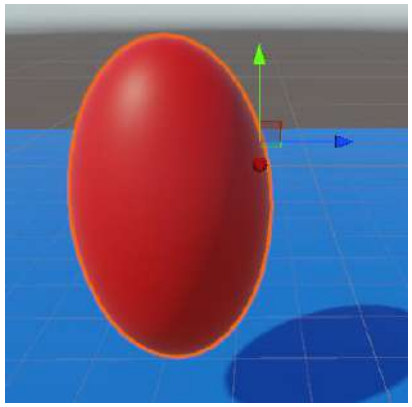




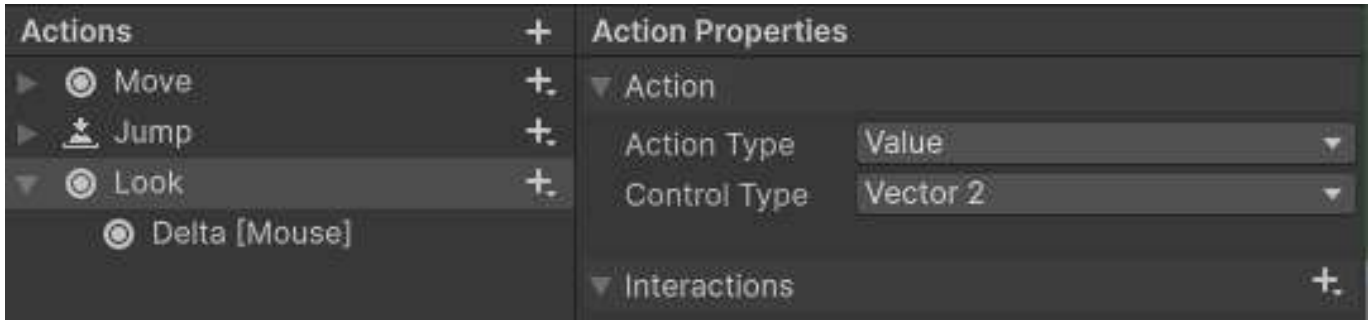
AVATAR / HRÁČ

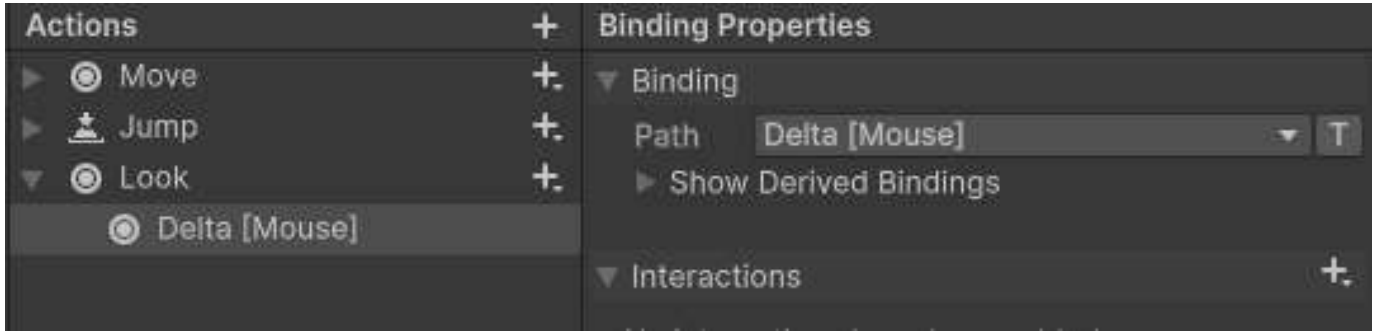
Úloha:

1. Vytvorte objekt a nastavte mu **kameru** ako **child** objektu, s ktorým sa budeme pohybovať
2. Kameru nastavte na vhodné miesto
3. Pridajte **novú action** do vášho **Input Action**
4. Pridajte **script na pohyb myši**
5. Pridajte **script na pohyb avatara**



NOVÁ INPUT ACTION





Mouse Script

```
using UnityEngine;
using UnityEngine.InputSystem;

public class MouseLook : MonoBehaviour
{
    public float sensitivity = 100f;
    public Transform playerBody;

    private Vector2 lookInput;
    private float xRotation = 0f;

    void Start()
    {
        Cursor.lockState = CursorLockMode.Locked;
    }

    public void OnLook(InputValue value)
    {
        lookInput = value.Get<Vector2>();
    }

    void Update()
    {
        float mouseX = lookInput.x * sensitivity * Time.deltaTime;
        float mouseY = lookInput.y * sensitivity * Time.deltaTime;

        // Vertical rotation (camera only)
        xRotation -= mouseY;
        xRotation = Mathf.Clamp(xRotation, -90f, 90f);
        transform.localRotation = Quaternion.Euler(xRotation, 0f, 0f);

        // Horizontal rotation (player body)
        if (playerBody != null)
        {
            playerBody.Rotate(Vector3.up * mouseX);
        }
    }
}
```

- Tento script pridajte na **Camera** object

- **Nezabudnuť** pridať referenciu na **Parent** objekt

Movement Script

```
using UnityEngine;
using UnityEngine.InputSystem;

public class PlayerMovement : MonoBehaviour
{
    public float speed = 5f;
    private Rigidbody rb;
    private Vector2 moveInput;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }

    public void OnMove(InputValue value)
    {
        moveInput = value.Get<Vector2>();
    }

    void FixedUpdate()
    {
        // get direction relative to player rotation
        Vector3 move = transform.forward * moveInput.y + transform.right *
moveInput.x;

        rb.MovePosition(rb.position + move * speed * Time.fixedDeltaTime);
    }
}
```

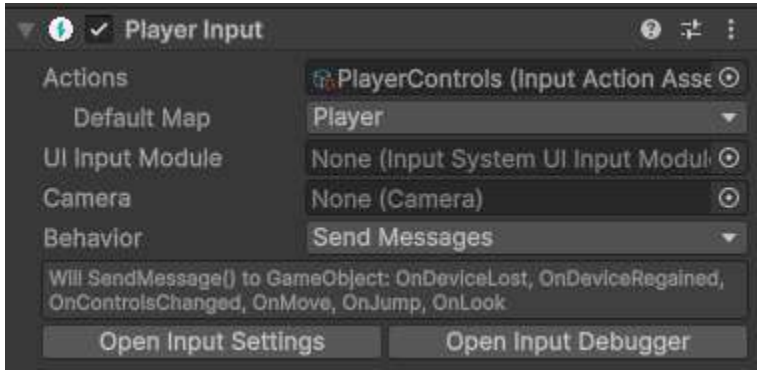
- Tento script pridajte na **Parent** objekt

SCRIPTY

- Pre oba parent a child objekty (Avatar a kamera) pridať komponent

Player Input

- **Actions** → váše **Input Action**
- **Behavior** → **Send Messages**

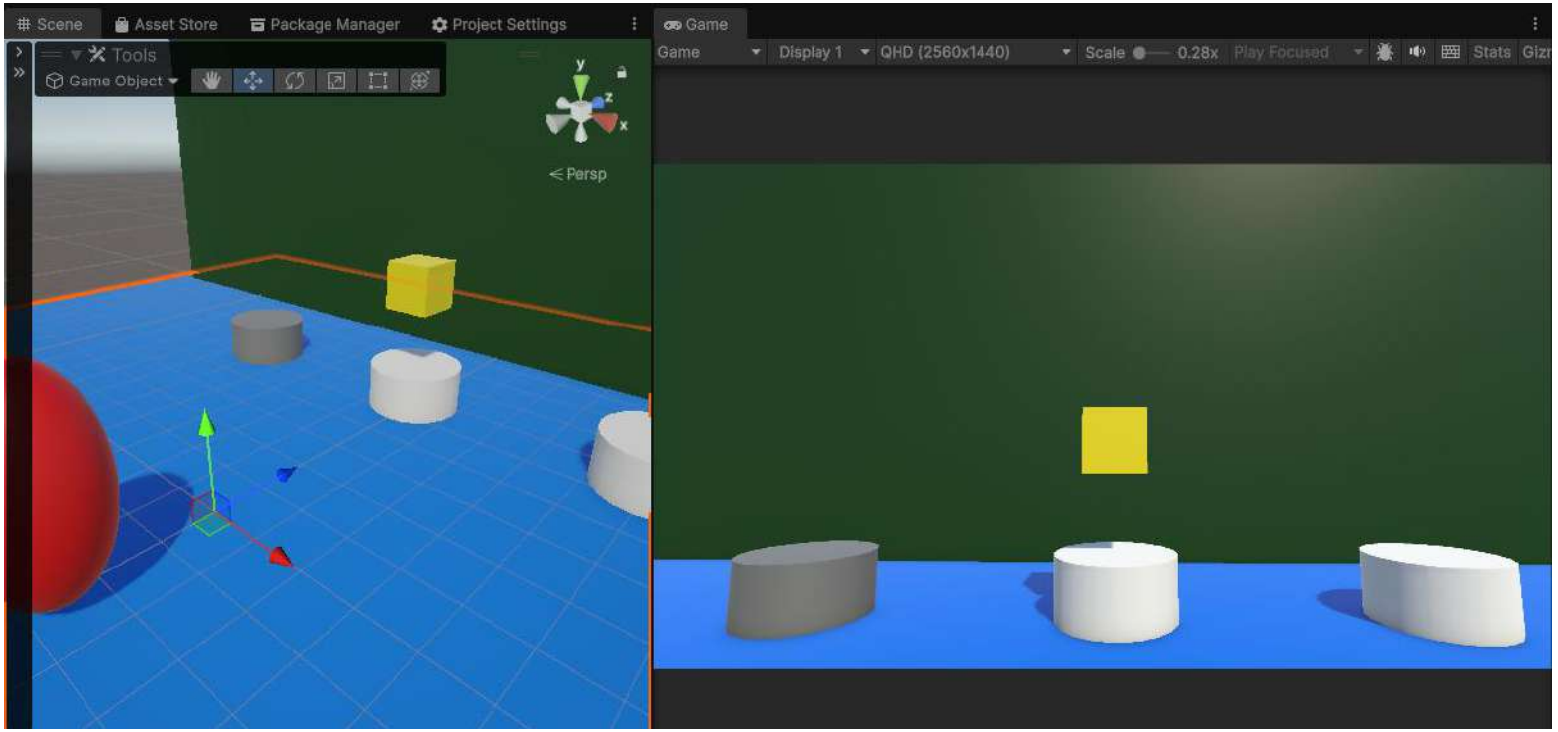


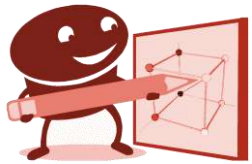
SCRIPTY

- Overte správanie scriptov

ÚLOHA

- Pridanie Look-At mechaniky





SCRIPT

- Tento script pridajte na **Camera** objekt
- Na objekt, ktorý chcete zvýrazniť pridajte tag:

LookedAtObject

- Objekty musia mať
  - Collider (pre raycast)
  - Renderer

```
using UnityEngine;

public class LookHighlight : MonoBehaviour
{
    public float distance = 10f;
    private GameObject lastObject;

    void Update()
    {
        Ray ray = new Ray(transform.position, transform.forward);
        RaycastHit hit;

        if (Physics.Raycast(ray, out hit, distance))
        {
            GameObject obj = hit.collider.gameObject;

            // ONLY objects with this tag
            if (obj.CompareTag("LookedAtObject"))
            {
                // reset previous object
                if (lastObject != null && lastObject != obj)
                {
                    ResetObject(lastObject);
                }
                HighlightObject(obj);
                lastObject = obj;
            }
            else
            {
                ResetLast();
            }
        }
        else
        {
            ResetLast();
        }
    }

    void ResetLast()
    {
        if (lastObject != null)
        {
            ResetObject(lastObject);
            lastObject = null;
        }
    }

    void HighlightObject(GameObject obj)
    {
        Renderer rend = obj.GetComponent<Renderer>();
        if (rend != null)
        {
            rend.material.color = Color.yellow;
        }
    }

    void ResetObject(GameObject obj)
    {
        Renderer rend = obj.GetComponent<Renderer>();
        if (rend != null)
        {
            rend.material.color = Color.white;
        }
    }
}
```

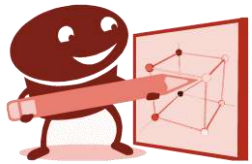
Collision Script

- Overte správanie scriptu

VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Nainštalovanie Gitu

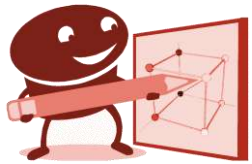
https://git-scm.com/install/



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Import Custom Package

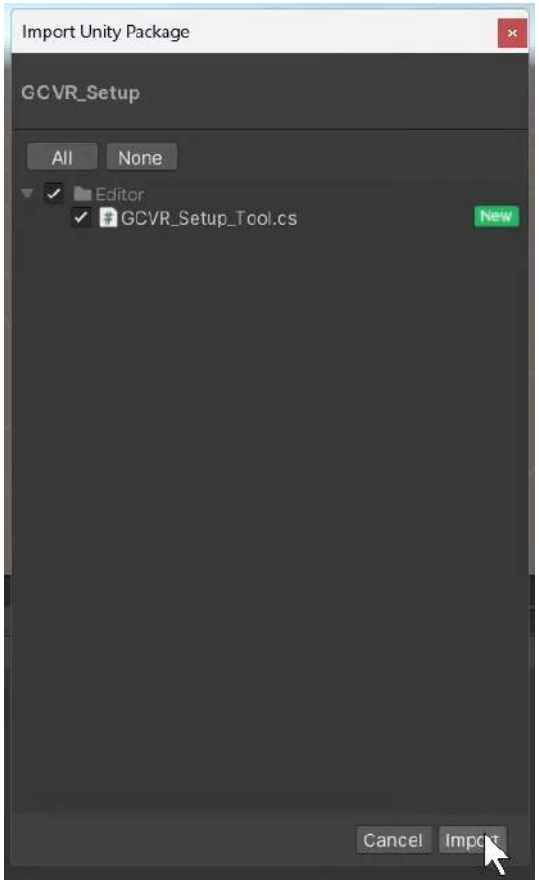
https://moodle.fei.tuke.sk/mod/resource/view.php?id=14522



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Import v Unity:

1. Assets
2. RightClick
3. Import Package
4. Custom Package

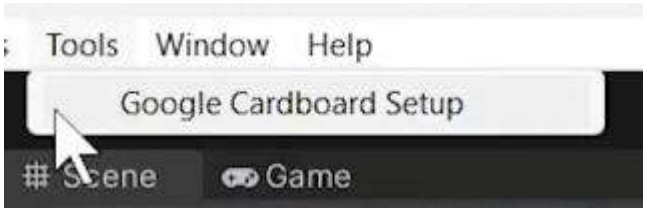


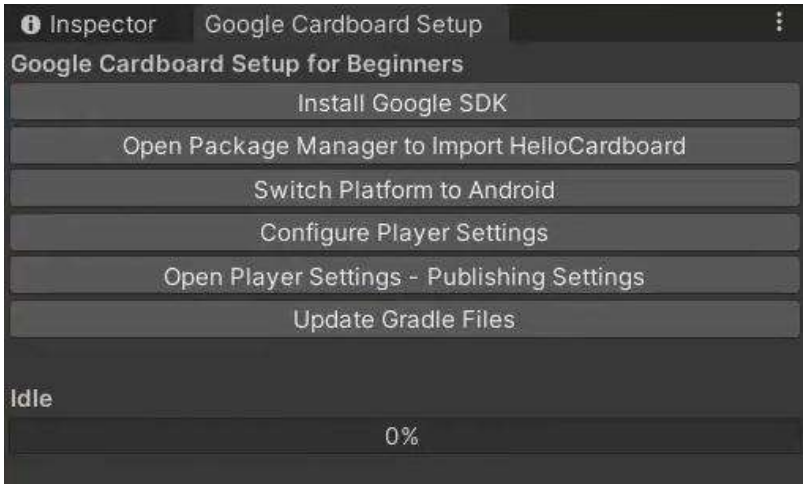
Ak nastanú errory → restart a skúsiť znova

VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Import v Unity:

1. Tools
2. Google Cardboard Setup
3. Install Google SDK



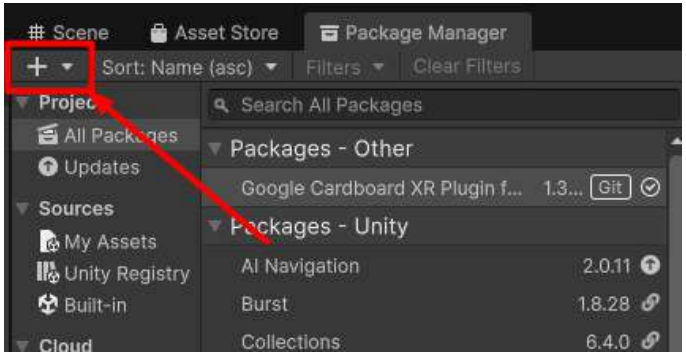


Ak nastanú errory → restart a skúsiť znova

VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Nainštalovanie Google Cardboard XR plugin

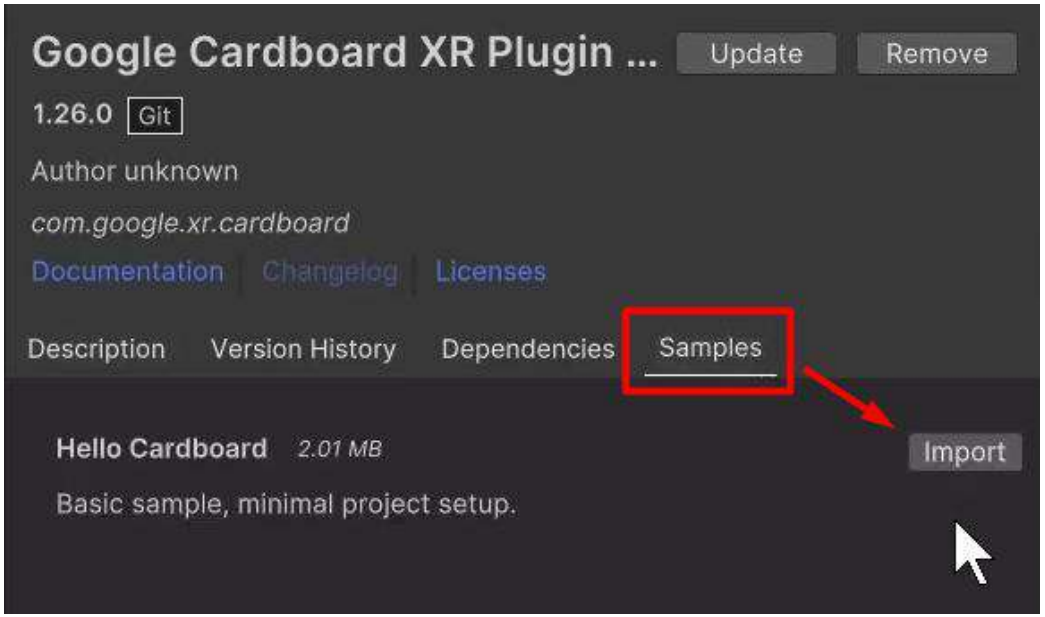
1. Window
2. Package Management
3. Package Manager
4. +
5. Install package from git URL
6. Install



https://github.com/googlevr/cardboard-xr-plugin.git

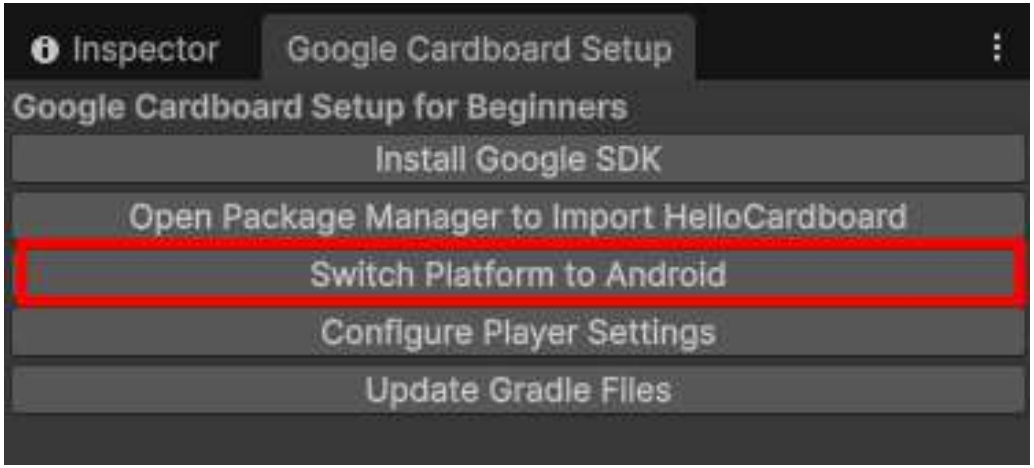
VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Google Cardboard XR plugin Scéna



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

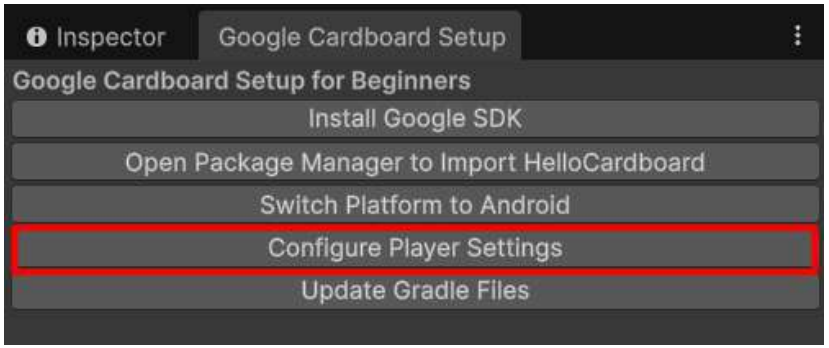
- Zmena platformy na Android



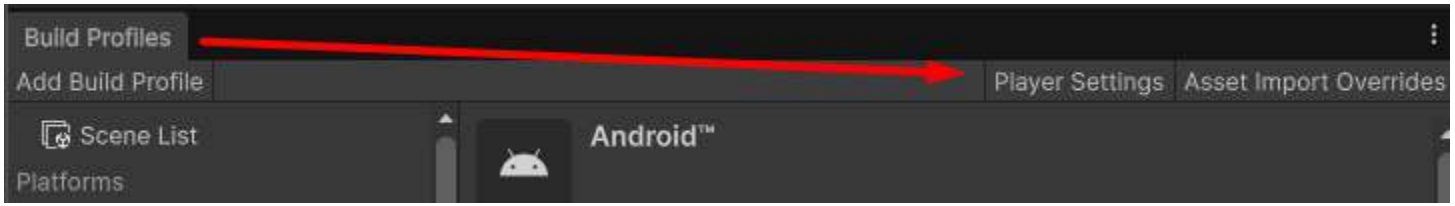
VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Konfigurácia Player Settings

1. Google Cardboard:



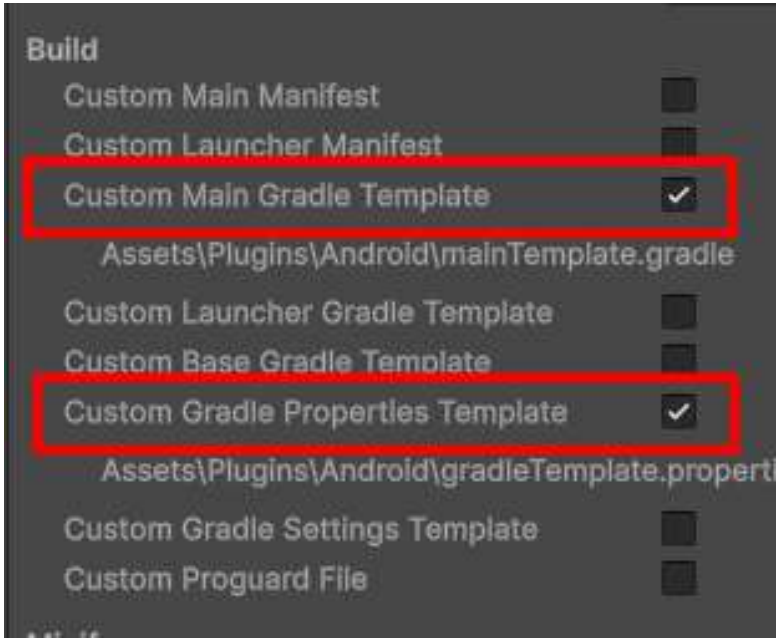
2. Build Settings/Profiles → Player Settings



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

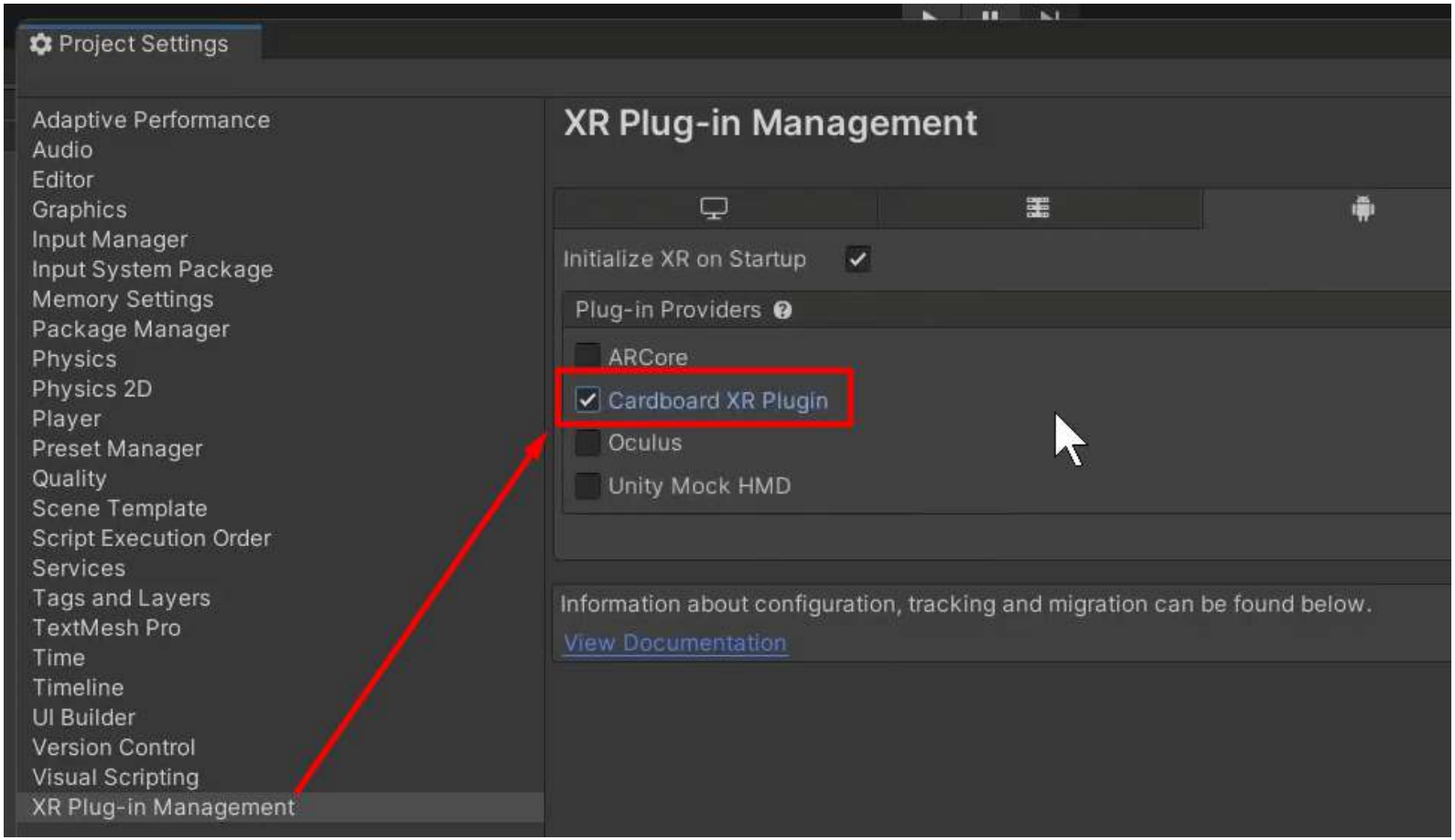
- Konfigurácia Player Settings

1. Player Settings
2. Publishing Settings
3. Build



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

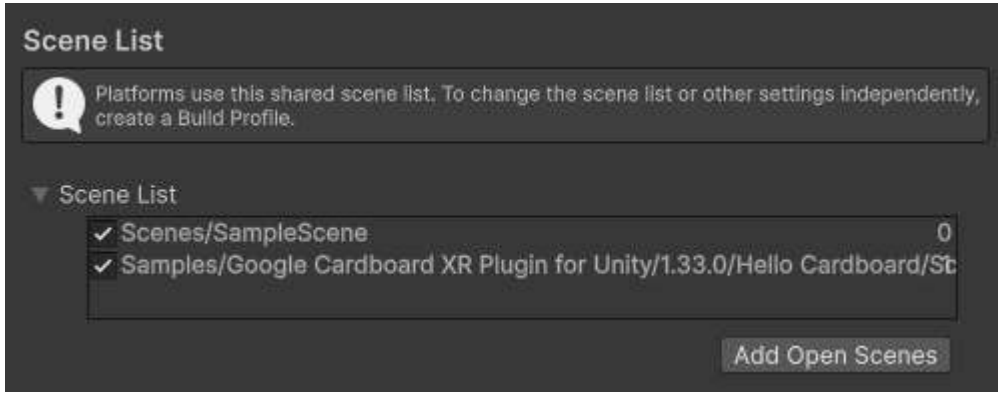
- XR Plug-in Management



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

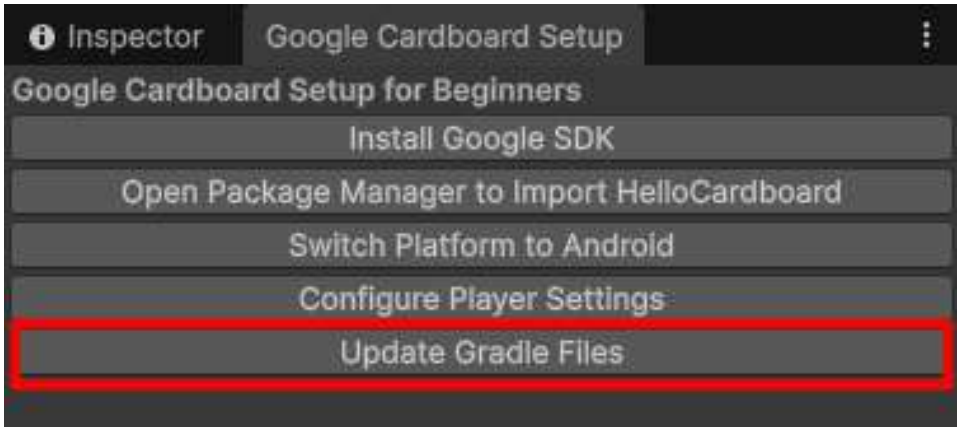
- Pridanie scény do Buildu

1. File → Build Settings/Profiles
2. Open Scene List
3. Add Open Scenes



VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

- Update Gradle Files

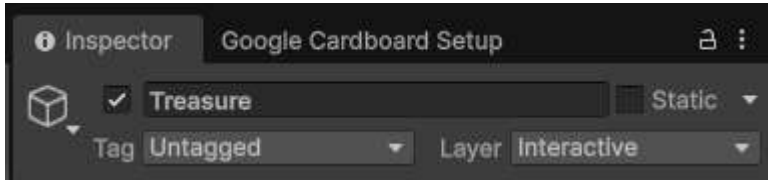


VYTVORENIE VR APLIKÁCIE DO TELEFÓNU

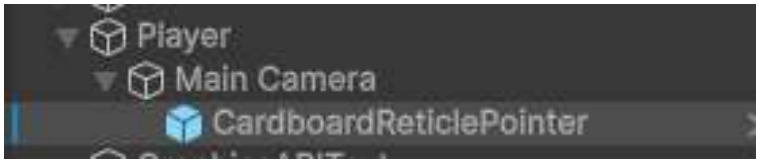
- Nastavenie Layerov na Interactive

1. Hierarchy → Treasure → Inspector
2. Layer → Add Layer
3. User Layer 6: Interactive

4.



5.



→

Q & A

branislav.sobota@tuke.sk
timotej.sobota@tuke.sk

Katedra počítačov a informatiky, FEI TU v Košiciach