Katedra Mechaniki Stosowanej i Robotyki
Wydział Budowy Maszyn i Lotnictwa, Politechnika Rzeszowska

Techniki wirtualnej rzeczywistości w mechatronice

Laboratorium nr 09-10

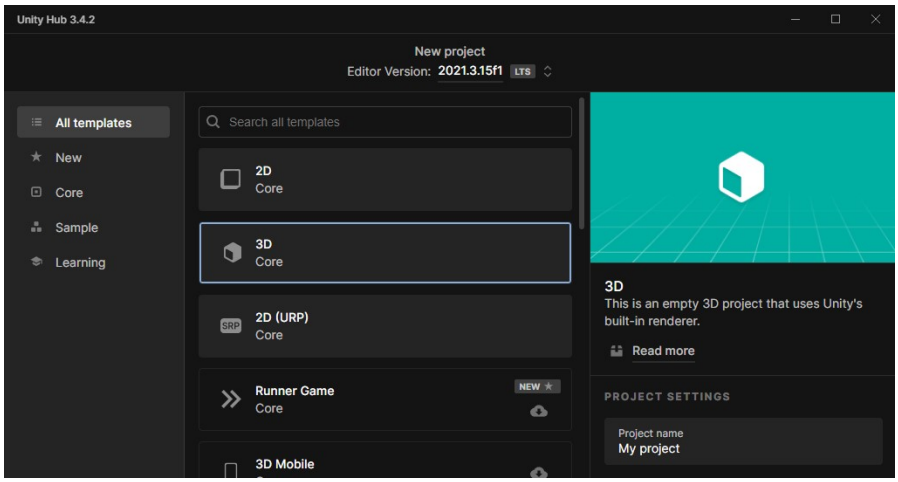
Temat: Unity Engine: podstawy obsługi Unity, pisania skryptów C# i importem assetów z programu Blender

1. Wstęp

Instrukcja ma na celu zapoznanie studenta z podstawami obsługi silnika fizycznego i środowiska Unity, pisania skryptów C# oraz współpracę z programem Blender.

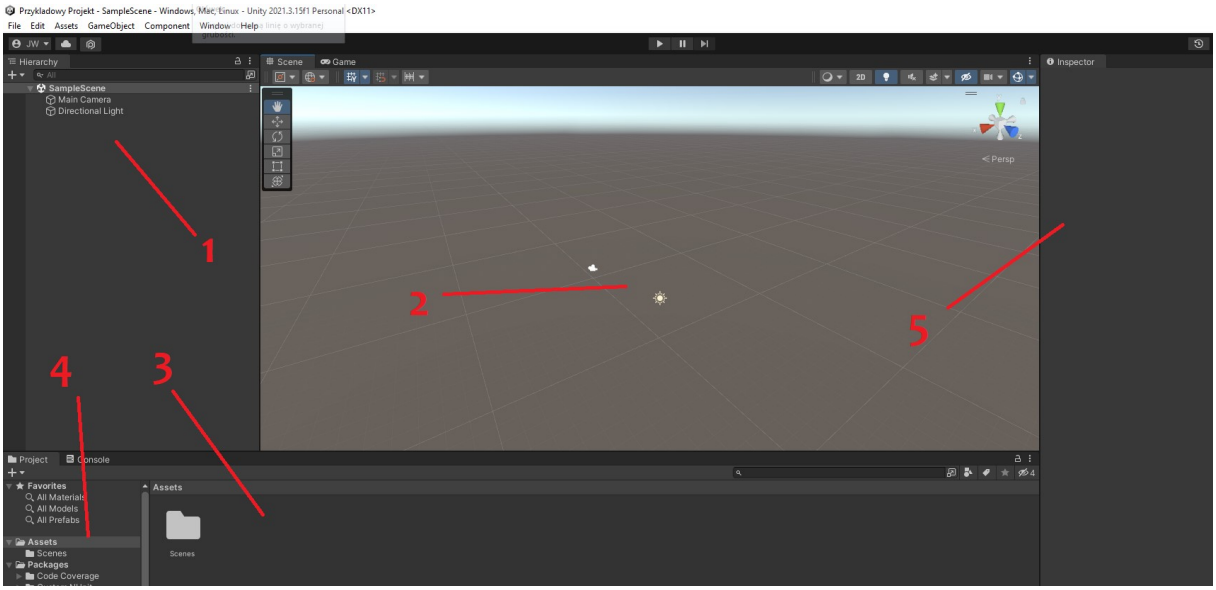
2. Podstawy obsługi Unity v.2021

Uruchamiamy edytor Unity i wybieramy nazwę projektu oraz opcje 3D (rys. 1).



Rys. 1 Utworzenie projektu

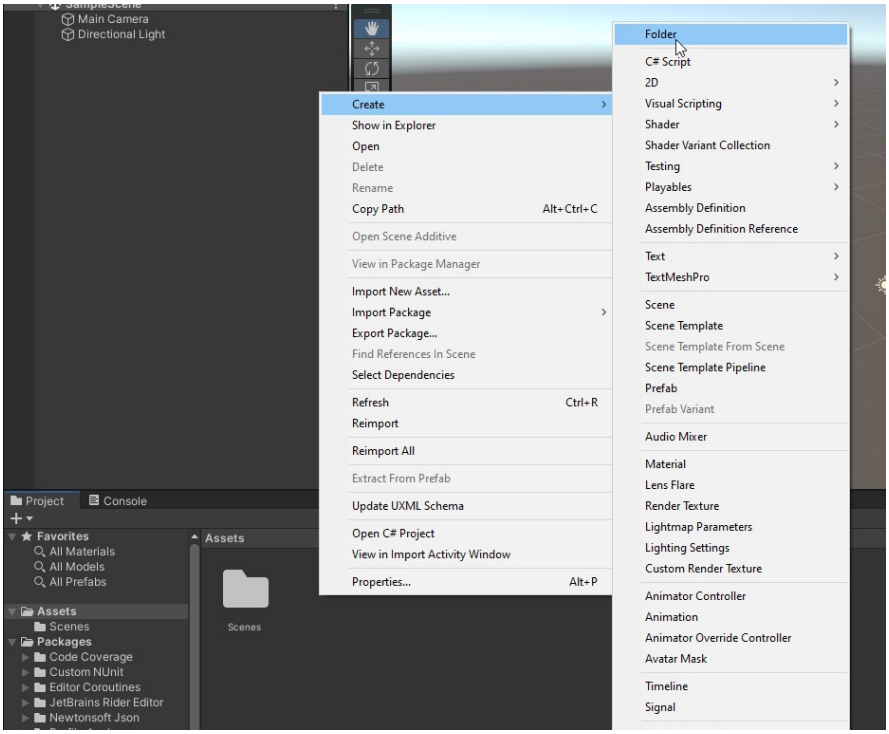
Na ekranie pojawi się podstawowy układ projektu (rys. 2). Po lewej stronie (1) znajduje się drzewko hierarchii obiektów dostępnych w projekcie. Tutaj pojawiają się obiekty, kolekcje, kamery itp. Na środku (2) mamy ekran podglądu projektu (2), który przedstawia rozstawienie początkowe elementów projektu widziane przez projektanta lub przez kamerę. Poniżej ekranu podglądu (3) znajduje się eksplorator plików w projekcie widocznych po lewej (4). Po prawej stronie znajduje się tzw. inspektor (5) służący do zmian ustawień zaznaczonych elementów projektu.



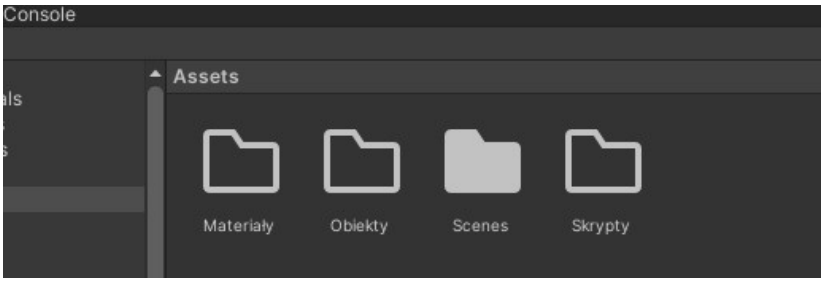
Rys. 2 Ekran główny

Ogólną zasadą w pisaniu programu (gry lub symulacji) w Unity jest przypisywanie obiektom w projekcie ich zachowań z użyciem skryptów w C#. Każdy obiekt oprócz swojego wyglądu wykazuje własne zachowanie definiowane przez skrypt C# lub ogólnie narzucone zachowanie wynikające z silnika fizycznego Unity. Mamy możliwość definiowania zachowania obiektów i ich interakcji z otoczeniem używając poleceń z biblioteki UnityEngine.
Obiekty mogą być fizyczne (np. bryły) lub wirtualne (np. kamery, kolekcje). Obiekty fizyczne można importować oprogramowania modelarskiego takiego jak Blender. Oprócz obiektów można dołączać tzw. sceny czyli np. etapy gry, dźwięki, materiały, skrypty itp. Wszystkie te elementy noszą nazwę Assetów.

W celu utrzymania porządku w projekcie dobrze podzielić assety na kategorie i umieścić je w osobnych folderach. W tym celu tworzymy nowe foldery Skrypty, Obiekty i Materiały w eksploratorze Unity klikając PPM-> Create -> Folder (rys. 3,4).



Rys. 3 Tworzenie nowego folderu



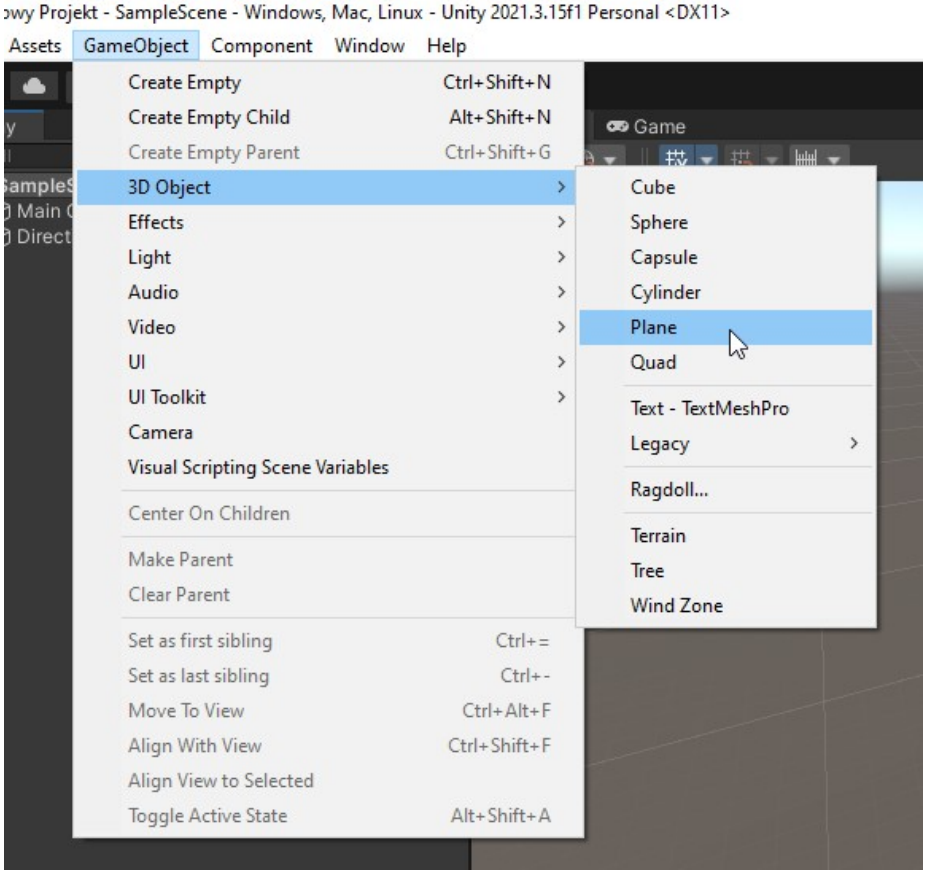
Rys. 4 Foldery

Z tak podstawowo skonfigurowanym projektem możemy przystąpić do pierwszego przykładu.

1. **Przykład 1: definicja obiektu i jego podstawowe poruszanie się**

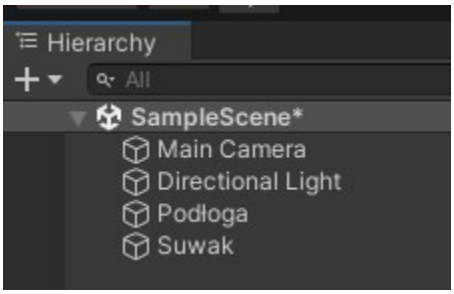
a) **Utworzenie obiektów**

Tworzymy podstawowe dwa obiekty: podłogę i suwak. W tym celu z zakładki „GameObject" wybieramy „3D Object" -> „Plane" oraz „Cube" (rys. 5).



Rys. 5 Tworzenie prymitywów

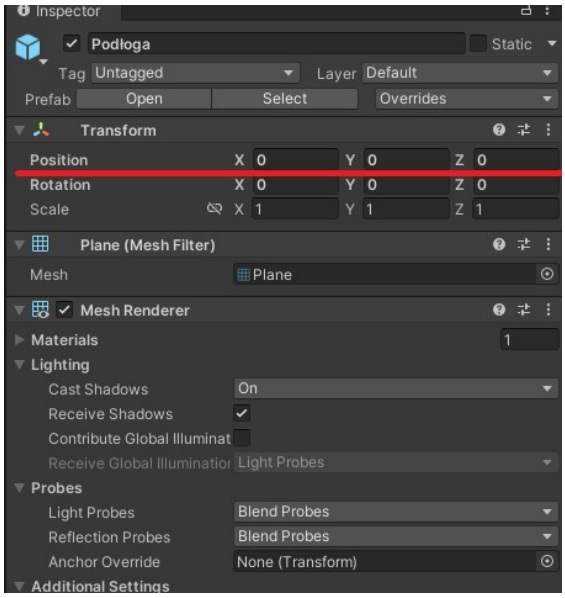
Zmieniamy im nazwę oraz przeciągamy je do folderu Obiekty (rys. 6a,b).

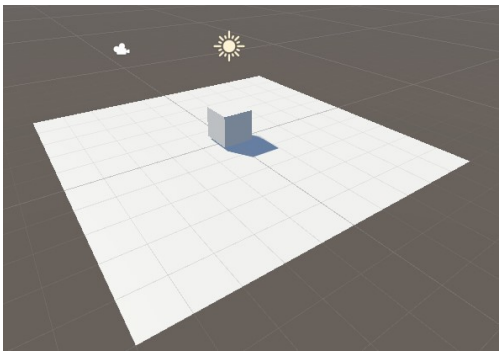
a) 

b) 

Rys. 6 a) edycja nazwy obiektów, b) obiekty w folderze

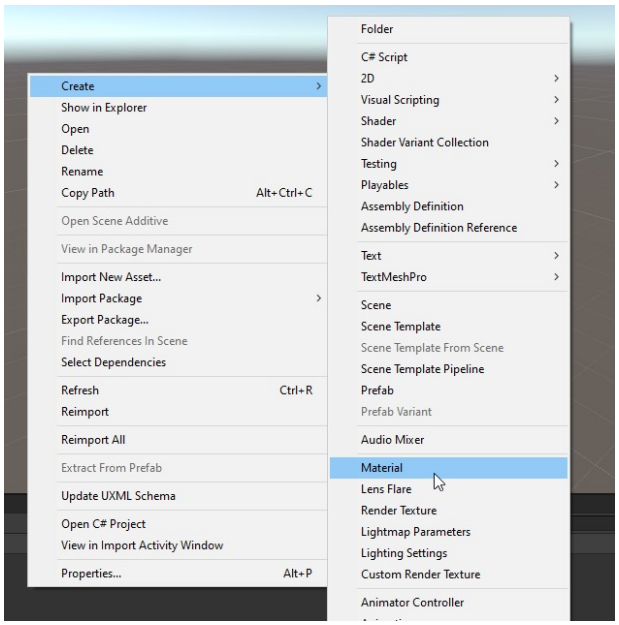
Klikając w obiekt podłoga oraz później w suwak zmieniamy ich pozycję w środowisku. W tym celu w inspektorze w oknie „Transform" przyjmujemy wartości położenia („Position") dla podłogi (0,0,0) a dla suwaka (0,0.6,0) (rys. 7a). Proszę zwrócić uwagę na osie układu współrzędnych. Oś y jest „do góry". Powinniśmy uzyskać efekt jak na rys. 7b.

a) 

b) 

Rys. 7 a) edycja pozycji, b) ustawienie obiektów

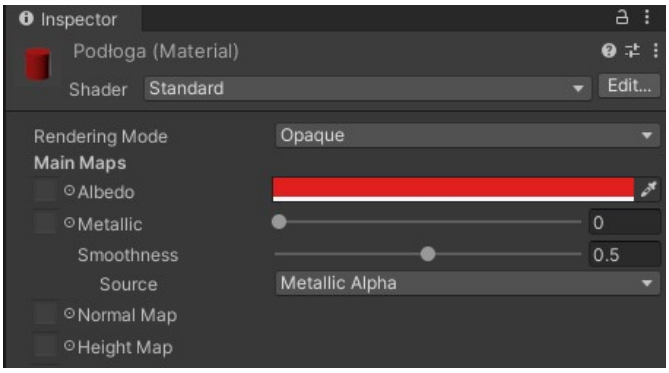
Możemy stworzyć materiały dla podłogi i suwaka. W tym celu otwieramy folder Materiały, klikamy w nim PPM -> Create -> Material (rys. 8a). Tworzymy dwa materiały zmieniając im nazwy na m_podłoga i m_suwak (rys. 8b).

a) 

b) 

Rys. 9 a) tworzenie materiałów ,b) materiały

Zmieniamy kolory materiałów na czerwony dla podłogi i zielony dla suwaka (rys. 10b). W tym celu zmieniamy kolor w opcji Albedo w inspektorze dla zaznaczonego materiału (rys. 10a).

a) 

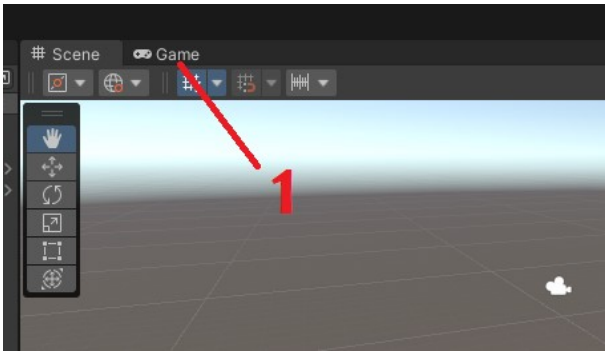
b) 

Rys. 10 a) Zmiana koloru materiału ,b) materiały po zmianie

Aby przypisać kolor obiektowi, przeciągamy materiał na obiekt na ekranie podglądu trzymając LPM.

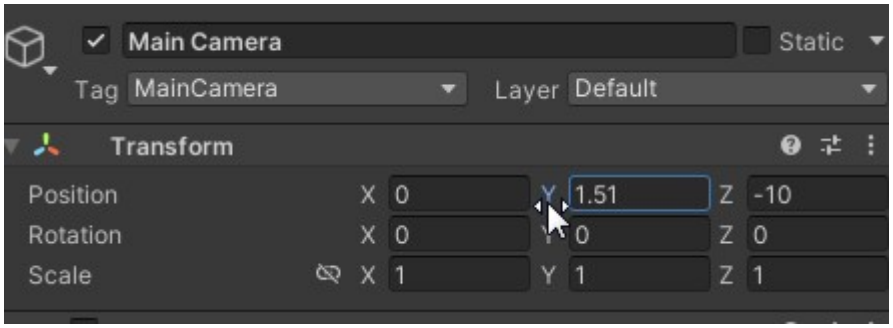
b) Przestawienie pozycji kamery

To co widzimy po uruchomieniu programu jest przekazywane przez kamerę domyślną „Main Camera". Żeby zobaczyć podgląd z kamery wybieramy zakładkę „Game" w ekranie podglądu (rys. 11).

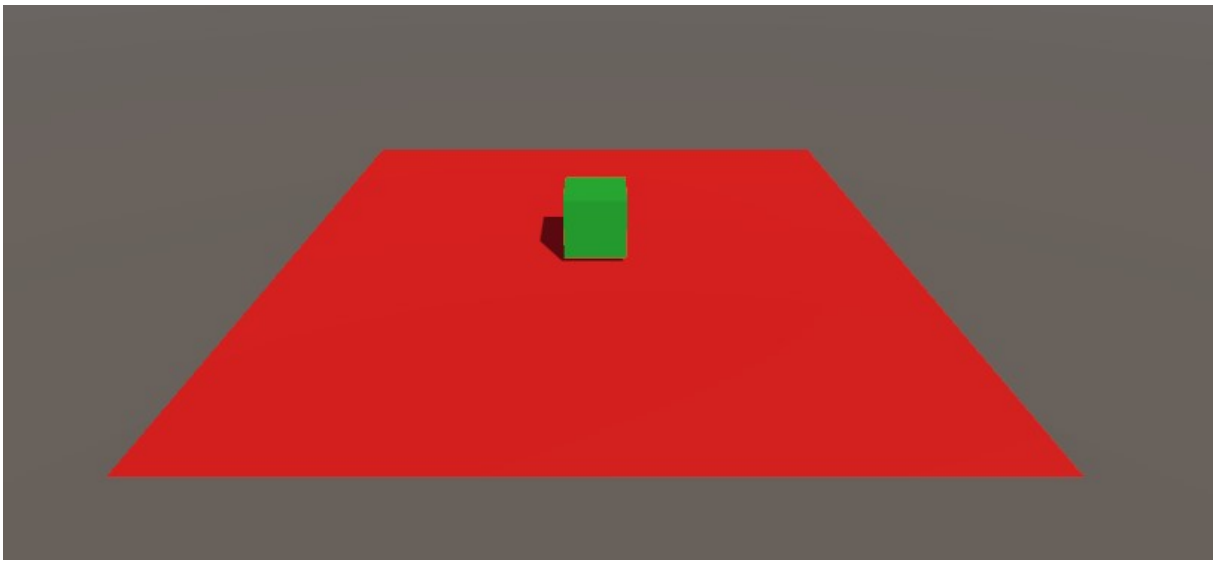


Rys. 11 Zmiana trybu podglądu na podgląd z kamery

Mając uruchomiony podgląd z kamery, zaznaczamy kamerę „Main Camera" w drzewku hierarchii i w inspektorze zmieniamy pozycję i orientację kamery, tak żeby lepiej obejmowała kadrem naszą scenę (rys. 13). W tym celu modyfikujemy opcje „Position" i „Rotation" w inspektorze. Możemy to robić płynnie przez najechanie na odpowiednie oznaczenie osi aż pojawi się symbol <-> co pozwoli nam zmieniać wartość pozycji i kąta przesuwając myszką w lewo i prawo (rys. 12).



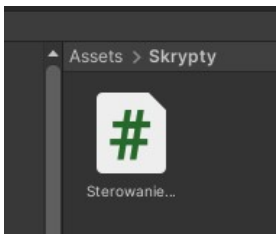
Rys. 12 Płynna zmiana położenia i rotacji kamery

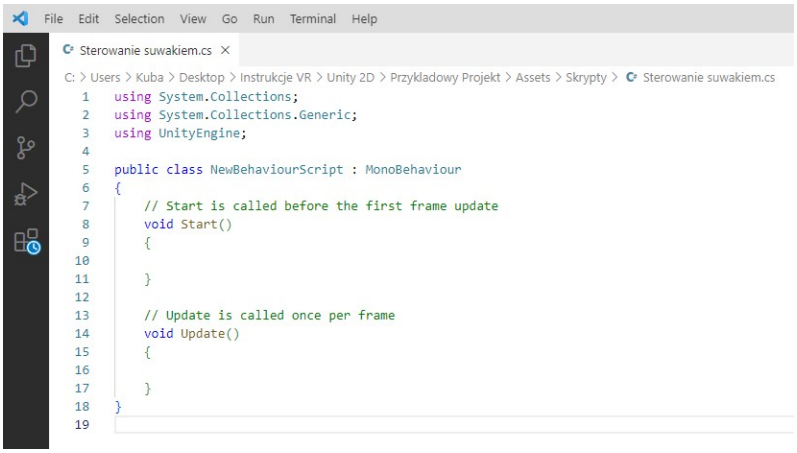


Rys. 13 Zmieniony kadr kamery

c) skrypt C# do sterowania suwakiem

Aby utworzyć nowy skrypt w folderze Skrypty klikamy PPM -> Create -> C# Script. Zmieniamy nazwę skryptu na SterowanieSuwakiem (rys. 14a) i otwieramy go dwukrotnie na nim klikając. Powinien nam się otworzyć skrypt w programie Visual Studio Code (rys. 14b).

a) 

b) 

Rys. 14 a) utworzony skrypt, b) skrypt w programie VSC

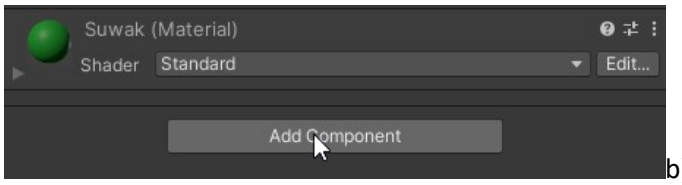
Skrypt składa się z podstawowych bibliotek Unity oraz jednej klasy dziedziczącej z klasy nadrzędnej MonoBehaviour. Jak wskazują komentarze, metoda void Start() wykonywana jest w pierwszej klatce trwania programu, natomiast metoda void Update() wywoływana jest co klatkę programu (tu będzie znajdowała się główna część skryptu). Każdy program, gra czy symulacja wywoływana jest z określoną liczbą klatek na sekundę.

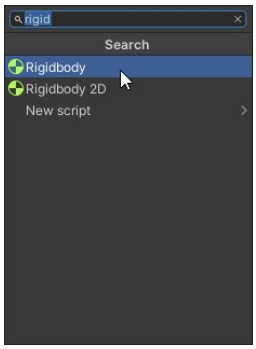
Modyfikujemy skrypt jak poniżej:

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

public class SterowanieSuwakiem : MonoBehaviour
{
```

```
    Rigidbody m_Rigidbody;                          // definicja bryly sztywnej
    [SerializeField] float moveSpeed = 10f;         // utworzenie modyfikowalnej zmiennej
stalej predkosci w Unity

    void Start()
    {
        m_Rigidbody = GetComponent<Rigidbody>(); // zczytanie paramatrow rigidbody z
projektu
    }

    void Update()
    {
        Vector3 m_Input = new Vector3(Input.GetAxis("Horizontal"), 0,
Input.GetAxis("Vertical")); // uwtorzenie wektora 3 elemetowego z klawiszy WASD
        m_Rigidbody.MovePosition(transform.position + m_Input * Time.deltaTime *
moveSpeed);      // przypisanie nowej pozycji dla bryly sztywnej
    }
}
```

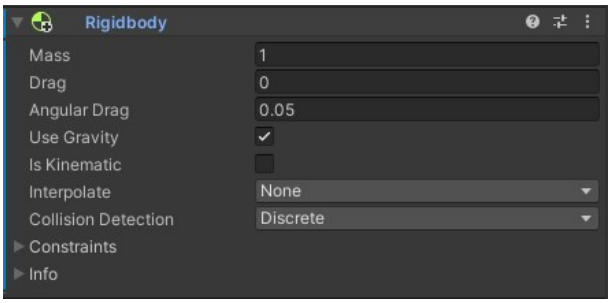
Zapisujemy skrypt skrótem Ctrl+s. Zapisane zmiany automatycznie zostaną zaimportowane do Unity. Obiekt będzie zachowywał się zgodnie z zasadami fizyki silnika Unity gdy dodamy mu komponent Rigidbody. Aby skrypt mógł się odwołać do klasy Rigidbody musimy ją najpierw przypisać do obiektu. W tym celu klikamy na suwak i w inspektorze klikamy przycisk „Add Component" (rys. 15a) i wybieramy „RigidBody" (rys. 15b).

a) 

b) 

Rys. 15 a) Dodawanie nowego komponentu, b) komponent Rigidbody

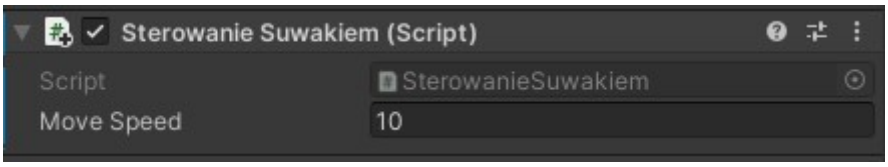
W inspektorze suwaka pojawi się nam nowa zakładka „Rigidbody", w której możemy zdefiniować parametry fizyczne suwaka oraz parametry symulacji fizyki Unity (rys. 16).



Rys. 16 Opcje Rigidbody

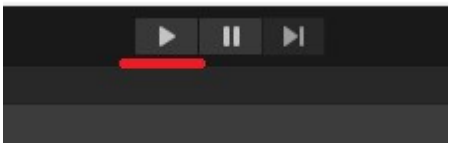
Po skonfigurowaniu komponentu „Rigidbody" przeciągamy skrypt SterowanieSuwakiem LPM na Suwak w drzewku hierarchii lub ekranie podglądu. Tak przeciągnięty skrypt będzie definiował zachowanie Suwaka w projekcie.

Po przypisaniu skryptu do obiektu w inspektorze suwaka pojawi się modyfikowalna zmienna „Move Speed” (rys. 17). Jeżeli chcemy żeby suwak się szybciej poruszał to możemy łatwo zmienić ten parametr w inspektorze.



Rys. 17 Zmienna prędkości w inspektorze

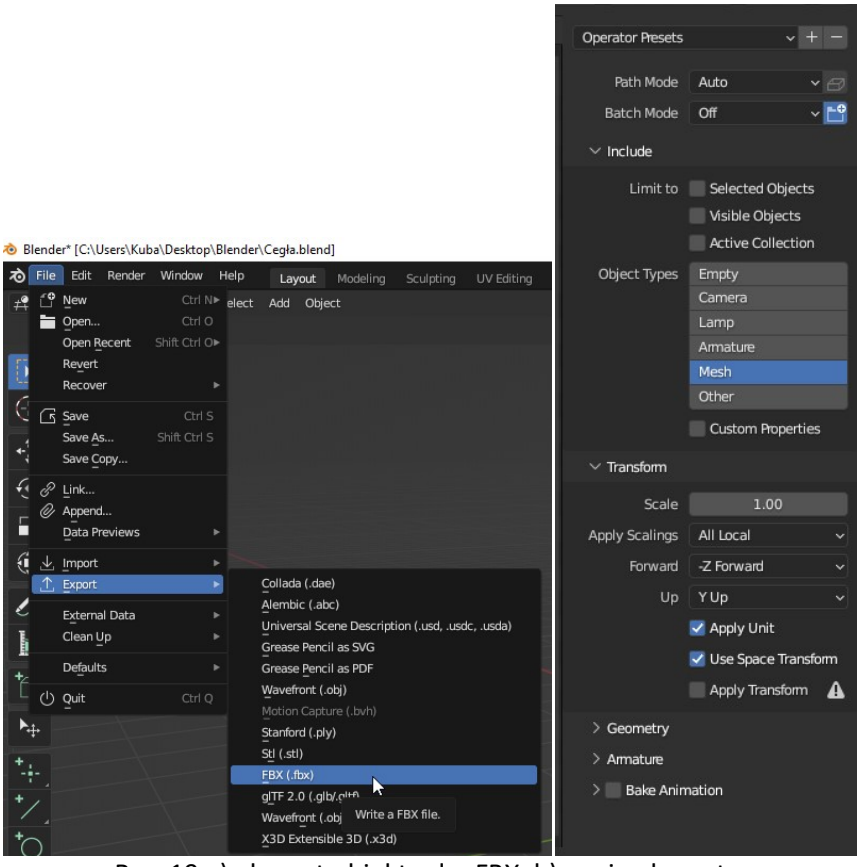
Możemy teraz przetestować zachowanie suwaka. Aby to zrobić klikamy przycisk „play” u góry ekranu poglądu (rys. 18). Przemieszczamy się suwakiem z użyciem klawiszy WASD.



Rys. 18 Przycisk uruchomienia programu w ekranie podglądu

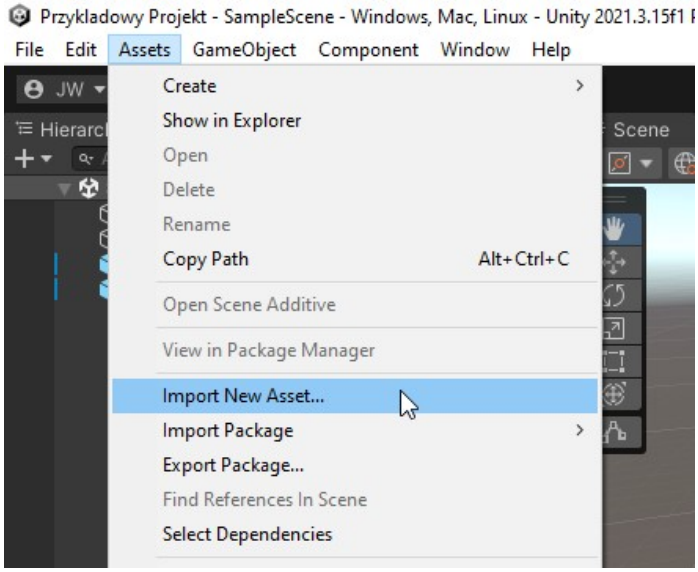
b) Przykład 2: eksportowanie obiektów i materiałów z Blendera do Unity

Jedną z metod eksportu obiekt z Blendera do Unity jest export do pliku .FBX. W tym celu w Blenderze w zakładce File -> Export wybieramy .FBX (rys. 19a) z opcjami jak na rys. 19b (należy tylko zaznaczyć mesh).



Rys. 19 a) eksport obiektu do .FBX, b) opcje eksportu

Następnie w Unity importujemy obiekt z zakładki Assets -> Import New Asset (rys. 20). I wybieramy eksportowany plik z blendera.



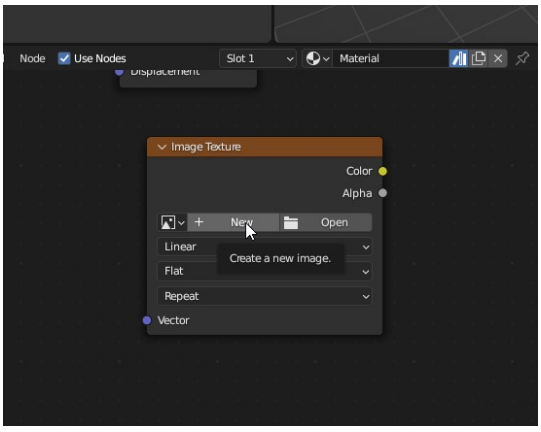
Rys. 20 Import assetu

Obiekt po zaimportowaniu będzie zawierał siatkę wieloboków oraz domyślny materiał, do którego można wstawić teksturę.

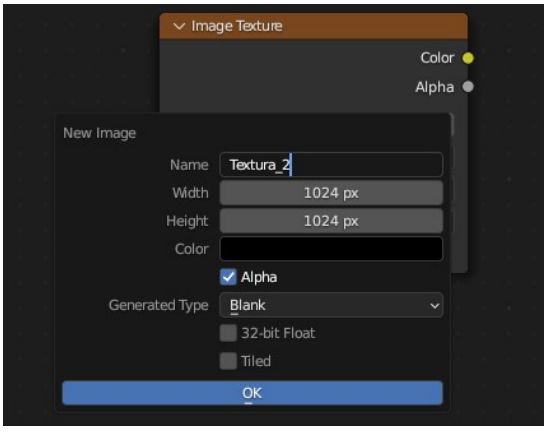
Generowanie i eksportowanie materiałów z Blendera do Unity

Jeżeli mamy już gotową teksturę (patrz poprzednie laboratorium), zrobioną w Blenderze to do zestawu nodów dodajemy node o nazwie „Image Textures" oraz klikamy „New" tak jak na rys. 21a. Nadajemy nazwę teksturze i wybieramy jej parametry (rys. 21b).

a)

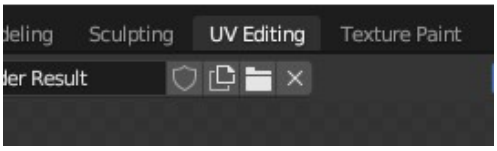


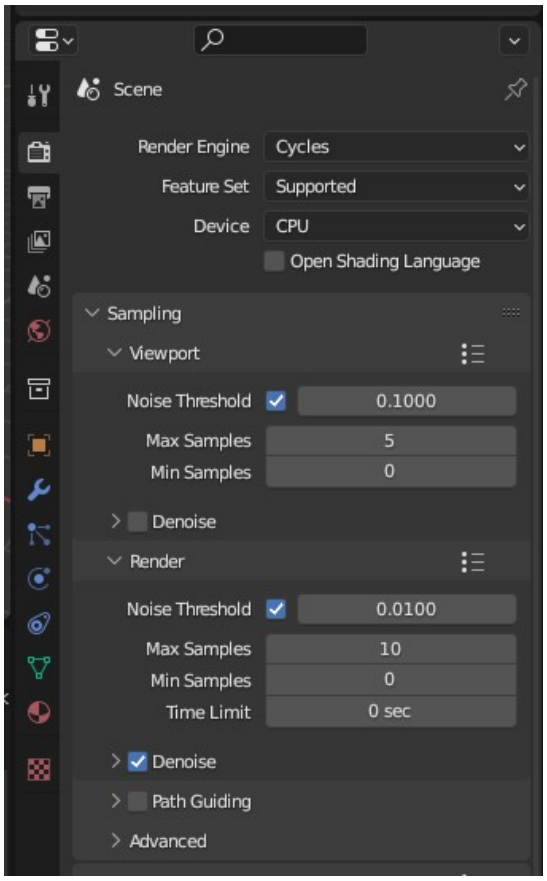
b)



Rys. 21 a) Tworzenie noda tekstury, b) dobór parametrów tekstury

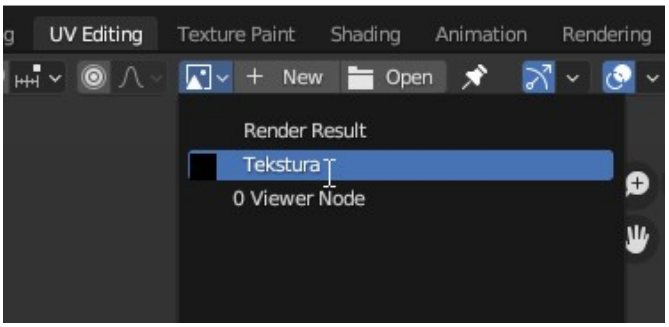
W zakładce „UV Editing" (rys. 22a) wybieramy odpowiednie opcje rendera tak jak na rys. 22b.

a) 

b) 

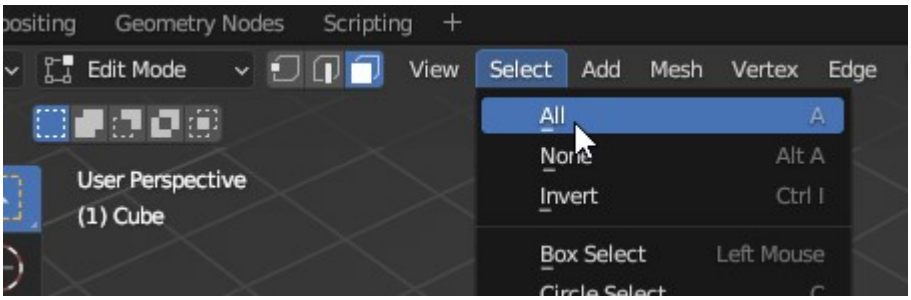
Rys. 22 a) zakładka UV Editing, b) zmiana ustawień rendera

W zakładce „UV Editing” wybieramy wcześniej utworzoną teksturę (ze swoją nazwą) tak jak na rys. 23.



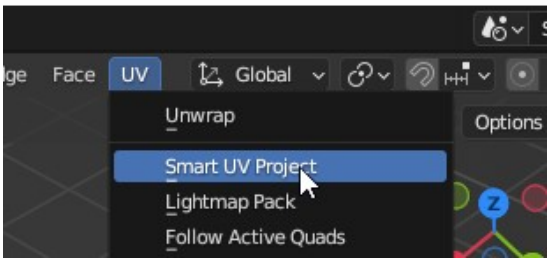
Rys. 23 Wybór tekstury

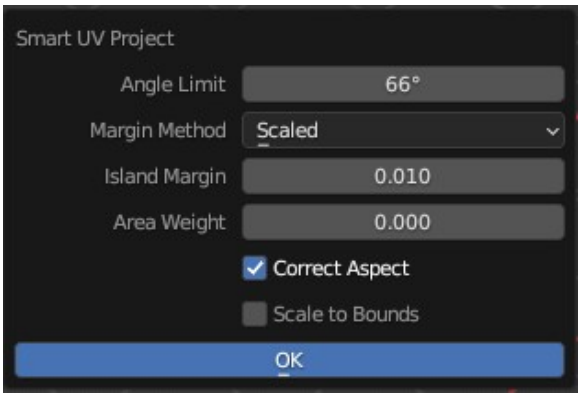
Będąc w zakładce „UV Editing” w trybie „Edit Mode” zaznaczamy wszystkie elementy naszego modelu używając polecenia Select -> All (rys. 24).



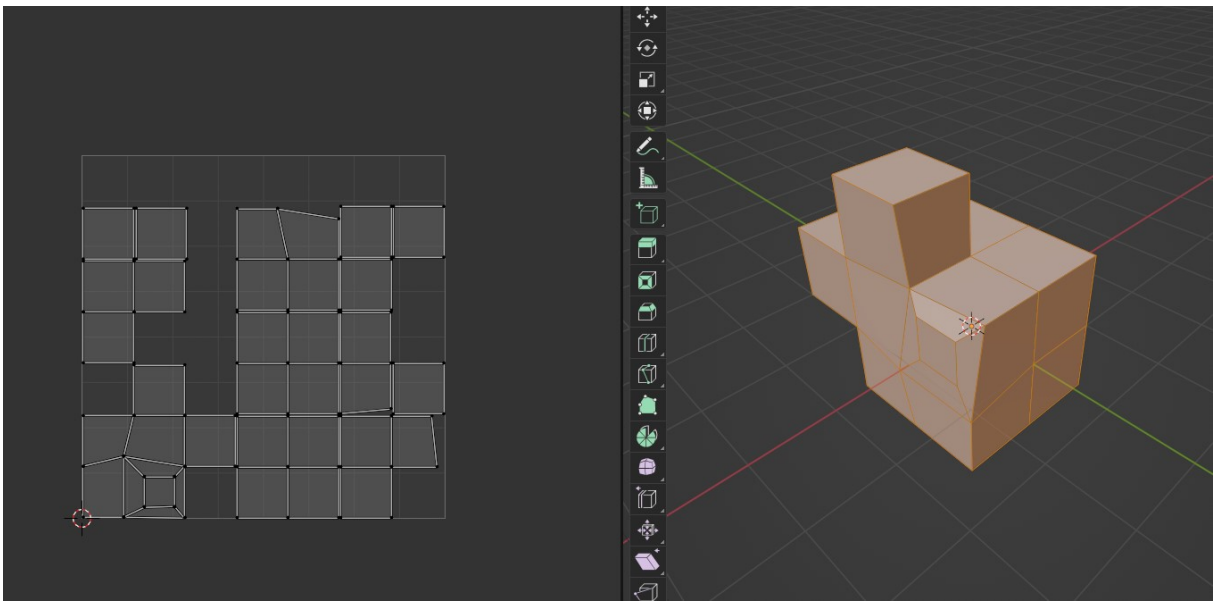
Rys. 24 Zaznaczanie wszystkich elementów bryły

Następnie należy utworzyć mapę UV korzystając z opcji „Smart UV Project” (rys. 25a) z opcjami jak na rys. 25b. Na rys. 26 pokazano przykładową bryłę i jej mapę UV.

a) 

b) 

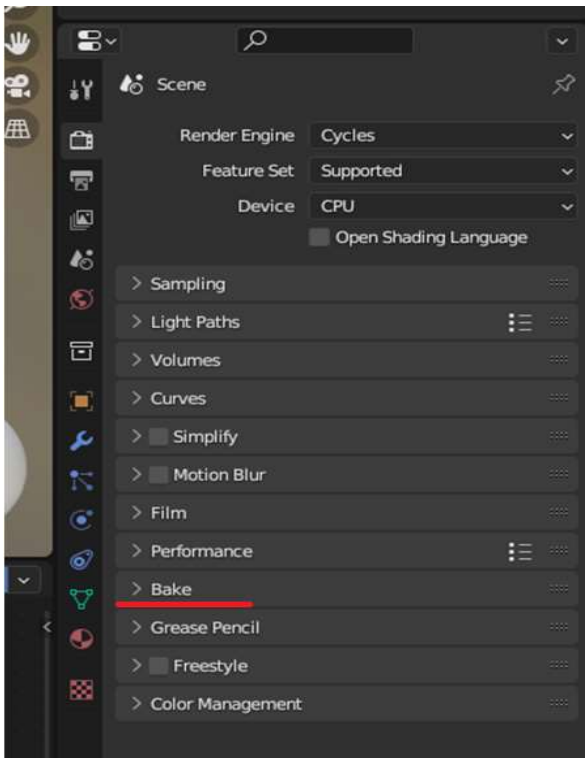
Rys. 25 a) tworzenie mapy UV dla obiektu, b) dobieranie parametrów rzutowania



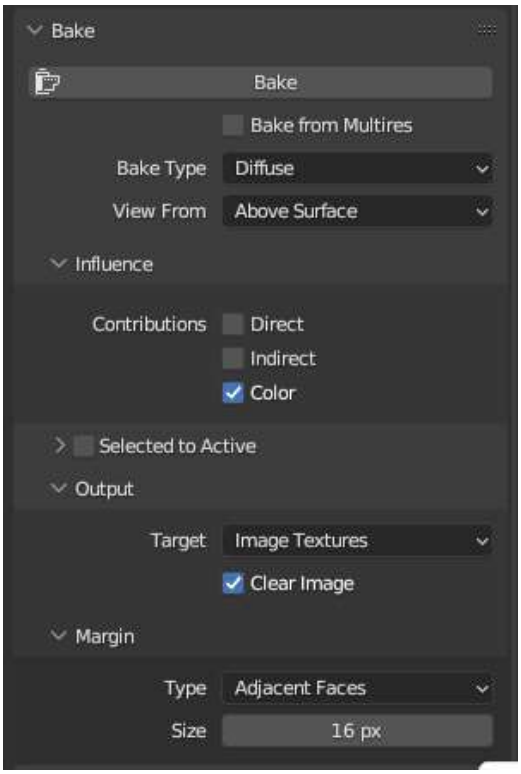
Rys. 26 Przykładowa bryła i jej mapa UV

Dalsza procedura sprowadza się do tworzenia plików elementów tekstury takich jak kolor (color), chropowatość (roughness) i mapa normalnych (normals). W celu utworzenia tych elementów przechodzimy do zakładki „Shading” (tam gdzie nody) i w zakładce „Bake” (rys. 27a) wybieramy odpowiednią część tekstury. Na rys. 27b-d przedstawione są parametry do ustawienia dla koloru, mapy normalnych i chropowatości. Z każdym elementem należy sprawdzić mapę UV w zakładce „UV Editing” czy została wygenerowana oraz należy ją zapisać w pliku w formacie .jpg (rys. 28a-c).

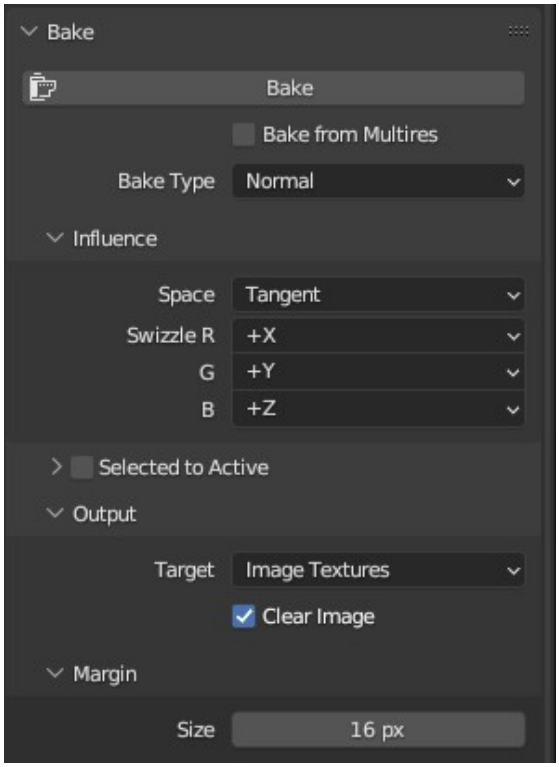
a)



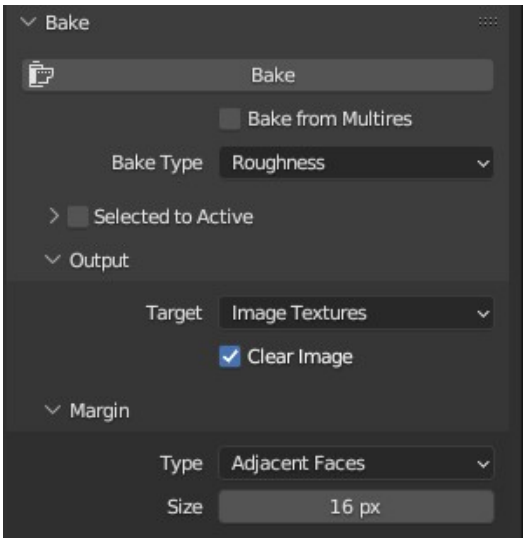
b)



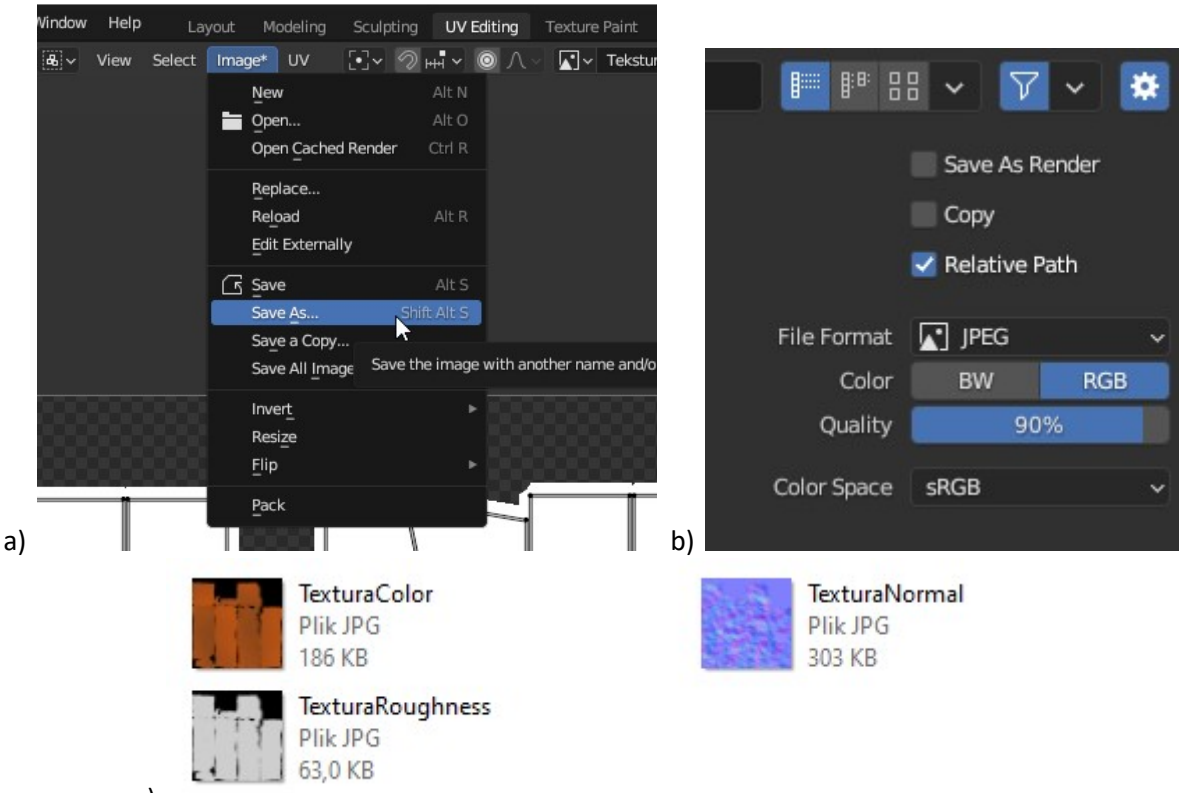
c)



d)



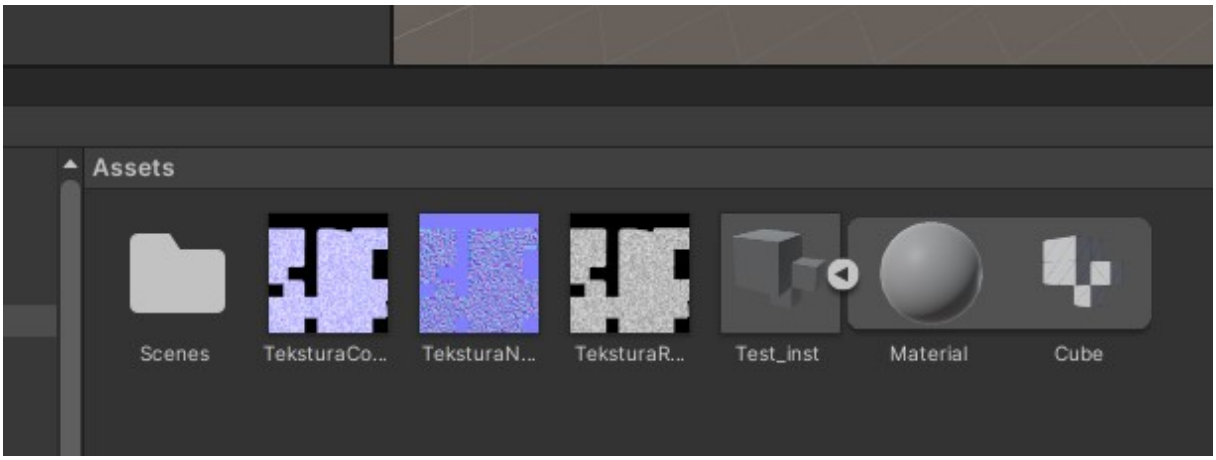
Rys. 27 a) wybór opcji „Bake”, b) opcje koloru tekstury, c) opcje mapy normalnych, d) opcje chropowatości



Rys. 28 a) Zapisywanie elementu tekstury, b) opcje zapisu do .jpg, c) utworzone pliki

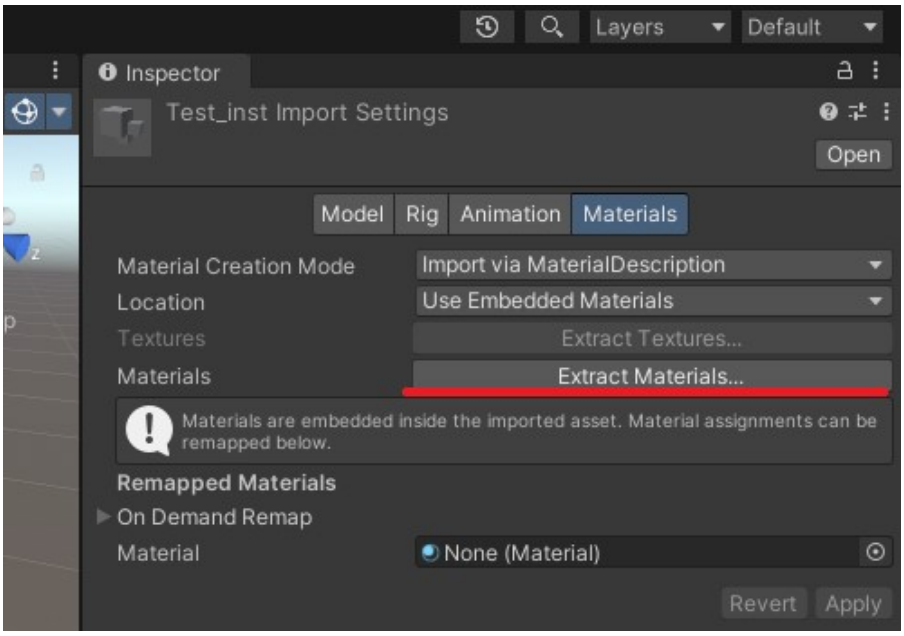
Unity

Przeciągamy elementy tekstury do eksploratora Unity oraz importujemy bryłę tak jak było wcześniej pokazane w instrukcji. W folderze Assets powinniśmy mieć pliki tak jak na rys. 29.



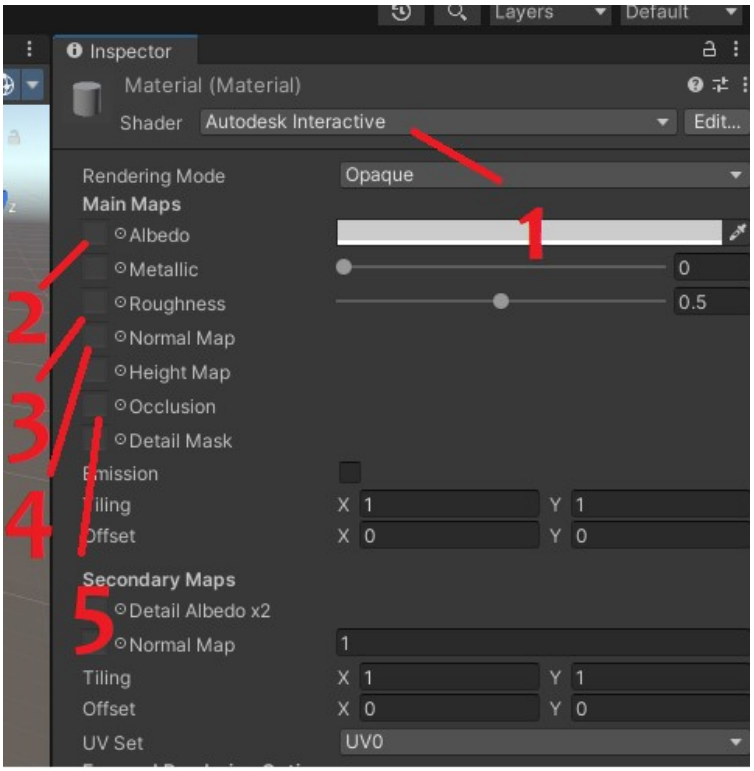
Rys. 29 Zawartość folderu Assets

Klikając na nasz obiekt w eksploratorze powinno się pokazać okno inspektora, gdzie w zakładce materials wybieramy „Extract Materials” (rys. 30). Następnie wybieramy folder projektu, gdzie ma się pojawić plik materiału.



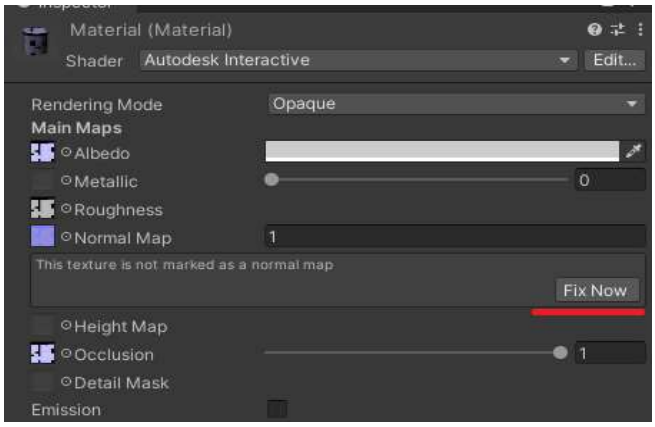
Rys. 30 Właściwości obiektu w inspektorze

Zaznaczamy wyeksportowany materiał i w jego inspektorze zmieniamy shader na Autodesk Interactive (1) i przeciągamy elementy tekstury w odpowiednie miejsca (rys. 31). Kolor (2,5), chropowatość (3), mapa normalnych (4).



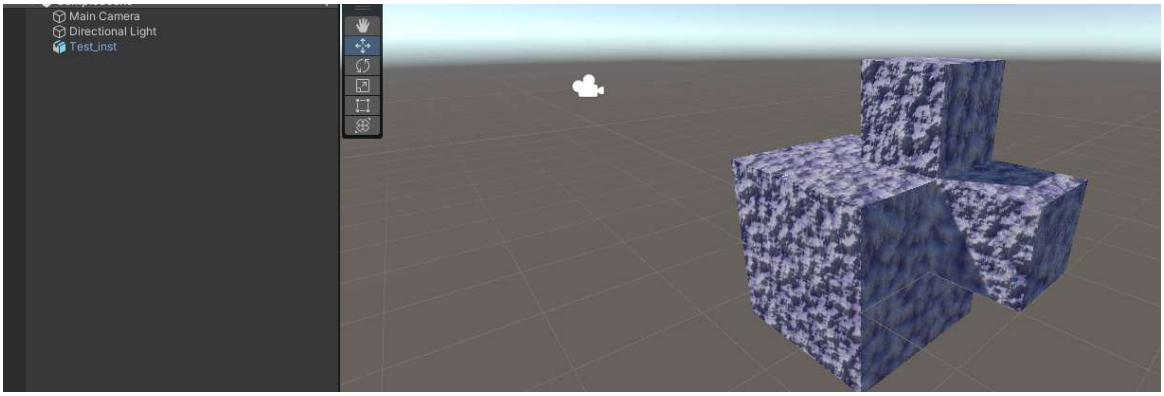
Rys. 31 Modyfikacja materiału z obiektu

Przy przypisywaniu mapy normalnych może wyskoczyć komunikat, że plik nie jest mapą. W tym celu należy kliknąć przycisk „Fix Now” (rys. 32).



Rys. 32 Zatwierdzenie modyfikacji mapy

Po przeciągnięciu obiektu na drzewko hierarchii Unity, powinien się on pojawić z teksturami w podglądzie Unity (rys. 33).



Rys. 33 Obiekt z teksturami

Zadania do wykonania

- Utworzyć nowy projekt w Unity.
- Umieścić w projekcie obiekt zamodelowany w Blenderze z użyciem techniki „rzeźbienia” wraz z jego teksturami.
- Napisać aplikacje, w której użytkownik może poruszać obiektem na platformie z użyciem klawiszy WASD.

Obiekt i platforma powinny posiadać kształty i tekstury stworzone w programie Blender. Do zadania można wykorzystać przykłady z instrukcji i poprzednich laboratoriów.

Student dostaje:

- ocenę 5 za prawidłowo utworzony projekt z modelowanymi w Blenderze obiektami i teksturami oraz sterowaniem w Unity
- ocenę 4 za prawidłowo utworzony projekt z modelowanymi w Blenderze obiektami oraz sterowaniem Unity
- ocenę 3 za prawidłowo utworzony projekt z obiektami oraz sterowaniem z Unity

W rozwiązywaniu zadań można posiłkować się materiałami dydaktycznymi, przykładami z instrukcji i Internetu oraz dokumentacji oprogramowania Blender i Unity na stronach:
https://docs.blender.org/manual/en/latest/modeling/index.html
https://docs.unity3d.com/Manual/index.html