Instrukcja laboratoryjna nr 2

Modele 3D i zasoby w aplikacjach interaktywnych

1. Cel i kontekst ćwiczenia

Każda interaktywna aplikacja 3D jest zbudowana z zasobów (ang. assets): modeli geometrycznych, tekstur, materiałów, dźwięków i skryptów. Jako programiści rzadko tworzymy wszystkie te zasoby od podstaw: częściej integrujemy modele pozyskane z zewnętrznych bibliotek lub od artystów 3D. Umiejętność prawidłowej oceny, importu i optymalizacji gotowych modeli jest zatem kompetencją kluczową dla każdego developera pracującego z silnikami czasu rzeczywistego.

W trakcie tego laboratorium poznamy cały pipeline: od znalezienia legalnego zasobu w Internecie, przez analizę jego licencji, aż po finalne umieszczenie go w działającej scenie Unity ze właściwie skonfigurowanymi materiałami.

Po zakończeniu tego laboratorium student:

- rozumie strukturę modelu 3D: siatkę (mesh), materiały, tekstury i kanały UV,
- zna najpopularniejsze formaty plików modeli i potrafi wskazać preferowany format dla Unity,
- potrafi znaleźć, ocenić i pobrać model 3D z legalnych źródeł,
- rozumie różnice między typami licencji CC i wie, które pozwalają na użycie komercyjne,
- potrafi poprawnie zaimportować model do Unity i skonfigurować materiały oraz tekstury,
- rozumie pojęcie LOD i podstawowe techniki optymalizacji siatki.

2. Wprowadzenie teoretyczne

2.1. Anatomia modelu 3D

Model 3D to matematyczna reprezentacja obiektu w przestrzeni trójwymiarowej. Na potrzeby silników czasu rzeczywistego składa się z następujących warstw:

Element	Definicja	Rola w silniku
Siatka (Mesh)	Zbiór wierzchołków (vertices) połączonych krawędziami w wielokąty (polygons)	Geometria widoczna na ekranie
Siatka UV (UV Map)	Dwuwymiarowe odwzorowanie wierzchołków siatki na płaszczyznę	Określa, jak tekstura „nakrywa” model
Tekstura	Plik graficzny (.png, .jpg, .tga) zawierający dane wizualne	Kolor, chropowatość, emisja itp.
Materiał	Zbiór parametrów shadera: kolor, tekstury, metaliczność, chropowatość	Określa, jak światło oddziałuje z obiektem
Szkielet (Armature)	Hierarchia kości wewnątrz modelu organicznego	Umożliwia animację postaci

W silnikach **PBR (Physically Based Rendering),** którego używa Unity, materiały składają się z kilku map tekstur współpracujących ze sobą. Najważniejsze z nich to:

- **Albedo** (Base Color): kolor bazowy powierzchni bez wpływu oświetlenia

- **Normal Map** : iluzja drobnych nieregularności geometrycznych bez dodawania trójkątów
- **Metallic / Roughness**: określają fizyczne właściwości materiału
- **Ambient Occlusion** (AO): mapa zacienienia wklęśnięć i szczelin modelu
- **Emission**: definiuje obszary samoistnie świecące

2.2. Formaty plików modeli 3D

Różne formaty plików są zoptymalizowane do różnych celów. Warto znać ich specyfikę, aby wybierać właściwy format dla konkretnego zastosowania:

Format	Typ	Obsługa Unity	Charakterystyka
.fbx	binarny / ASCII	Natywna	Złoty standard branży. Obsługuje siatki, materiały, animacje i szkielety. Preferowany format importu do Unity.
.obj	ASCII	Natywna	Prosty, szeroko kompatybilny. Brak animacji. Materiały w osobnym pliku .mtl.
.gltf / .glb	JSON / binarny	Package Manager	Nowoczesny standard open-source (Khronos Group). .glb to wersja binarna all-in-one.
.blend	binarny	Przez konwersję	Format natywny Blendera. Unity może go importować, lecz wymaga zainstalowanego Blendera.
.dae	XML	Natywna	Format Collada, często używany w wymianie między programami. Obsługuje animacje.
.3ds	binarny	Natywna	Stary format 3ds Max. Ograniczenia: brak animacji szkieletowych, limit 65k polygonów na obiekt.

Zalecany format importu do Unity

Do pracy z Unity należy preferować format .fbx: zapewnia on pełną obsługę siatek, materiałów i animacji bez konieczności instalacji zewnętrznych narzędzi. Pliki .gltf/.glb są równie wartym uwagi alternatywnym wyborem dzięki ich otwartości i kompaktowości.

2.3. Licencjonowanie zasobów cyfrowych

Pobieranie modeli 3D z Internetu wymaga uwzględnienia kwestii prawnych. Nieznajomość typów licencji może prowadzić do naruszenia praw autorskich, co w projekcie komercyjnym rodzi poważne konsekwencje. Poniżej przedstawiono zestawienie licencji Creative Commons stosowanych najczęściej w repozytoriach modeli 3D:

Licencja	Darmowe użycie	Modyfikacja	Komercyjnie	Atrybucja
CC0 (Public Domain)	Tak	Tak	Tak	Nie wymagana
CC BY	Tak	Tak	Tak	Wymagana
CC BY-SA	Tak	Tak (na tych samych warunkach)	Tak	Wymagana
CC BY-NC	Tak	Tak	NIE	Wymagana
CC BY-ND	Tak	NIE	Tak	Wymagana
CC BY-NC-ND	Tak	NIE	NIE	Wymagana
Royalty-Free (RF)	Po zakupie	Ograniczona	Często tak	Wg umowy

Na potrzeby zajęć akademickich korzystamy wyłącznie z modeli na licencji CC0 lub CC BY, które pozwalają na użycie bez ograniczeń (lub z obowiązkiem podania źródła). Dokładne źródła i typy licencji każdego pobranego zasobu muszą być udokumentowane.

2.4. Źródła zasobów 3D

Poniżej zestawiono sprawdzone repozytoria modeli 3D z dostępnymi licencjami open-source:

Serwis	URL	Dostępne licencje	Uwagi
Sketchfab	sketchfab.com	CC0, CC BY, CC BY-SA, inne	Filtry licencji, podgląd 3D w przeglądarce. Konto bezpłatne.
Poly Haven	polyhaven.com	CC0 (wszystkie)	W wyście zasoby: modele, tekstury, HDRI. Wysoka jakość.
Kenney.nl	kenney.nl	CC0 (wszystkie)	Zasoby stylizowane (low-poly). Idealne do prototypowania.
Quaternius	quaternius.com	CC0 (wszystkie)	Paczki tematyczne (postaci, pojazdy, środowiska).
OpenGameArt	opengameart.org	CC0, CC BY, GPL	Duża baza; należy weryfikować licencję indywidualnie.
Unity Asset Store	assetstore.unity.com	Unity EULA	Bezpłatne pakiety zaznaczone 'Free'. Licencja powiązana z Unity.

2.5. Optymalizacja modeli 3D

Silniki czasu rzeczywistego muszą renderować całą scenę 60 lub więcej razy na sekundę. Liczba trójkątów (polygonów) modeli bezpośrednio wpływa na koszt renderowania: zbędnie złożone modele obniżają wydajność bez zauważalnej poprawy jakości wizualnej. Kluczowe techniki optymalizacji to:

1. **LOD (Level of Detail):** zastąpienie modelu uproszczoną wersją przy oddaleniu kamery. Unity obsługuje LOD Group jako komponent,
2. **Batching**: łączenie siatek współdzielących ten sam materiał w jedno wywołanie rysowania (draw call),
3. **Atlasowanie tekstur**: scalanie wielu tekstur w jedną (Texture Atlas) redukuje przełączanie stanów GPU,
4. **Decymacja siatki**: algorytmiczne zmniejszanie liczby trojkątów (np. przez Decimate Modifier w Blenderze),
5. **Współdzielenie materiałów**: obiekty używające tego samego materiału są grupowane przez silnik w jedno wywołanie.

Praktyczna reguła: model bohatera w grze mobilnej to ok. 2–5 tys. polygonów; model w grze AAA na PC to 50–200 tys. W aplikacjach VR warto trzymać się dolnych granic, gdyż renderowanie stereoskopowe podwaja koszt.

3. Przygotowanie do ćwiczenia

3.1. Wymagania

Projekt Unity	Projekt z Laboratorium 1 LUB nowy projekt 3D (Core): wg wskazówek prowadzącego
Konto Sketchfab	Założenie bezpłatnego konta na sketchfab.com (pozwala na pobieranie plików)

Przeglądarka	Dostęp do internetu do pobrania modeli (dotyczy zajęć stacjonarnych)

3.2. Rozszerzenie struktury projektu

Jeśli kontynuujemy projekt z Lab 1, dodajemy nowe foldery w Assets:

```
Assets/
├── _Scenes/
├── Scripts/
├── Prefabs/
├── Materials/
├── Models/              ← pliki .fbx / .obj / .gltf
├── Textures/
└── Documentation/       ← nowy folder: źródła i licencje
```

Folder Documentation posiadamy dla zestawienia źródeł pobieranych zasobów. Może to być plik tekstowy, arkusz kalkulacyjny lub plik Markdown: ważne, by znalazło się w nim dla każdego modelu: nazwa, autor, URL, typ licencji.

4. Ćwiczenia praktyczne

4.1. Pozyskiwanie modeli z legalnych źródeł

Przed przystąpieniem do pracy z Unity musimy znaleźć i pobrać odpowiednie modele. Na potrzeby tego ćwiczenia będziemy pracować z trzema modelami reprezentującymi różne kategorie obiektów: meblowy, architektoniczny i jeden dowolny.

Krok 1. Wyszukiwanie modeli na Sketchfab

1. Wchodzimy na stronę sketchfab.com i logujemy się do konta.
2. Klikamy Explore, następnie filtry: Downloadable: Yes oraz License: CC Attribution.
3. Szukamy modelu krzeseł lub stołu (np. wpisujemy „chair low poly” lub „wood chair”).
4. Wybieramy model spełniający kryteria: wersja FBX dostępna, liczba trójkątów poniżej 20k.
5. Klikamy Download i wybieramy format FBX. Notujemy autora i URL strony modelu.

Krok 2. Wyszukiwanie modelu na Poly Haven

1. Wchodzimy na stronę polyhaven.com i przechodzimy do sekcji Models.
2. Przeglądamy dostępne modele (wszystkie są na licencji CC0, więc nie musimy filtrować).
3. Pobieramy jeden wybrany model w formacie FBX lub GLTF w rozdzielczości tekstur 1K lub 2K.

Krok 3. Trzeci model według własnego wyboru

1. Wybieramy model z dowolnego z podanych wyżej repozytoriów (Kenney, Quaternius, OpenGameArt).
2. Sprawdzamy typ licencji i notujemy wymagane informacje o atrybucji.

Krok 4. Zestawienie źródeł

Tworzymy plik tekstowy lub Markdown w folderze Documentation o następującej strukturze:

```
# Zestawienie zasobów: Laboratorium 2

## Model 1: [Nazwa modelu]
- Autor:    [imię i nazwisko / nick]
- Źródło:   [pełny adres URL strony modelu]
```

```
- Licencja: [np. CC BY 4.0]
- Format:   [np. FBX]
- Liczba trójkątów: [liczba, jeśli znana]

## Model 2: [Nazwa modelu]
...
```

4.2. Import modeli do Unity

Import to nie tylko skopiowanie pliku: Unity wykonuje w tle konwersję formatu, generuje Asset Database i tworzy zdefiniowane przez Import Settings ustawienia modelu. Prawidłowe skonfigurowanie Import Settings jest kluczowe dla jakości finalnej sceny.

Krok 1. Kopiowanie plików do projektu

1. Otwieramy folder Assets/Models w Eksploratorze plików systemu operacyjnego.
2. Kopiujemy pobrane pliki modeli .fbx (oraz foldery z teksturami, jeśli są osobno) do tego katalogu.
3. Wracamy do edytora Unity: silnik automatycznie wykryje nowe pliki i przetworzy je.

> **Ścieżki tekstur**
> Wiele modeli FBX spodziewa się, że tekstury będą się znajdować w podfolderze o tej samej nazwie co **plik .fbx** lub w folderze **Textures** obok niego. Umieszczamy dlatego pliki tekstur w **Assets/Textures** lub w podfolderze **Assets/Models/[NazwaModelu]/Textures**, aby Unity automatycznie je powiązał.

Krok 2. Konfiguracja Import Settings

1. W panelu Project zaznaczamy plik .fbx importowanego modelu.
2. W panelu Inspector widzimy okno Import Settings z zakładkami. Przechodzimy przez każdą:
   - Zakładka Model: sprawdzamy Scale Factor (domyślnie 1): jeśli model pojawia się zbyt duży lub zbyt mały, zmieniamy tę wartość (np. 0.01 dla modeli eksportowanych z Blendera w jednostkach centymetrowych).
   - Zakładka Rig: jeśli model nie posiada szkieletu, ustawiamy Animation Type na None.
   - Zakładka Animation: odznaczamy Import Animation, jeśli nie korzystamy z animacji modelu.
   - Zakładka Materials: ustawiamy Material Import Mode na Import via MaterialDescription: Unity spróbuje automatycznie odwzorować materiały z pliku.
3. Klikamy Apply, aby zatwierdzić ustawienia.

Krok 3. Umieszczanie modelu w scenie

1. Tworzymy nową scenę lub otwieramy istniejącą (z Lab 1).
2. Przeciągamy plik .fbx z panelu Project do panelu Scene View lub Hierarchy.
3. Sprawdzamy pozycję modelu w Inspector: ustawiamy Position na (0, 0, 0).
4. Skalujemy model, jeśli jest zbyt duży lub zbyt mały względem reszty sceny.
5. Powtarzamy kroki dla pozostałych dwóch modeli, ustawiając je w różnych miejscach sceny.

4.3. Konfiguracja materiałów i tekstur

Materiał w Unity jest instancją shadera: określa, jak shader interpretuje przypisane mu tekstury i parametry numeryczne. Unity używa domyślnie shadera Universal Render Pipeline / Lit lub (w starszych projektach) Standard, które obsługują cykl PBR.

Krok 1. Diagnostyka stanu materiałów po imporcie

1. Po umieszczeniu modelu w scenie klikamy na niego i sprawdzamy w Inspectorze sekcję Mesh Renderer → Materials.
2. Jeśli materiały są różowe lub jednolicie białe, to oznacza, że Unity nie powiązał ich poprawnie z teksturami. Kontynuujemy poniższymi krokami.

Krok 2. Ręczna konfiguracja materiału

1. W panelu Project klikamy prawym przyciskiem na Assets/Materials → Create → Material.
2. Nadajemy materiałowi opisową nazwę (np. Mat_Krzeslo_Drewno).
3. W Inspectorze w sekcji Albedo klikamy ikonę koła obok pola i wybieramy teksturę Albedo/BaseColor z folderu Textures.
4. Jeśli posiadamy Normal Map: klikamy pole Normal Map i przypisujemy odpowiednią teksturę. Unity automatycznie zapyta o konwersję: zatwierdzamy.
5. Jeśli posiadamy Metallic/Roughness: przypisujemy do pól Metallic i Roughness.
6. Przeciągamy gotowy materiał na obiekt w scenie (lub przypisujemy go w Inspectorze przez pole Materials w Mesh Renderer).

Krok 3. Weryfikacja wizualna

1. Sprawdzamy model w Scene View: klikamy F, by wycentrować widok na zaznaczonym obiekcie.
2. Przełączamy tryb widoku na Shaded (domyślny) i Wireframe (W), żeby zobaczyć topologię siatki.
3. Uruchamiamy scenę (Play) i sprawdzamy wygląd modeli w trybie Game View.

4.4. Organizacja hierarchii i tworzenie prefabrykatów

Trzy importowane modele powinny być logicznie pogrupowane w hierarchii sceny. Dobra hierarchia nie tylko porządkuje projekt, ale też upraszcza programistyczne operowanie grupami obiektów.

Krok 1. Grupowanie obiektów

1. W panelu Hierarchy klikamy prawym przyciskiem → Create Empty. Nadajemy obiektowi nazwę ImportedModels.
2. Przeciągamy wszystkie trzy modele na obiekt ImportedModels, aby stały się jego dziećmi.
3. Sprawdzamy, czy Transform obiektu-rodzica jest na (0, 0, 0): dzięki temu przemieszczanie całej grupy będzie intuicyjne.

Krok 2. Tworzenie prefabrykatów

1. Wybieramy jeden z zaimportowanych modeli (np. krzesełko).
2. Przeciągamy go z hierarchii do folderu Assets/Prefabs.
3. Unity tworzy prefabrykat: oryginalny obiekt w hierarchii staje się instancją prefabrykatu (ikona zmienia się na niebieską).
4. Tworzymy dwie dodatkowe instancje prefabrykatu: przeciągamy go z folderu Prefabs do sceny i rozstawiamy w różnych miejscach.

4.5. Optymalizacja sceny

Optymalizacja to nie tylko zmniejszanie liczby trójkątów: to całościowe podejście do zarządzania zasobami w projekcie. Wykonamy kilka podstawowych działań optymalizacyjnych.

Krok 1. Analiza statystyk renderowania

1. W trybie Play otwieramy okno Stats (prawy górny róg Game View → Stats).
2. Notujemy wartości: Batches, Tris (trianglę), Verts (wierzchołki), SetPass calls.
3. Wracamy do trybu Edit i porównujemy wartości po kolejnych krokach optymalizacji.

Krok 2. Static Batching

1. Obiekty nieporuszające się w scenie oznaczamy jako statyczne: zaznaczamy obiekt → w Inspectorze zaznaczamy checkbox Static (prawy górny róg).
2. Unity połączy siatki statyczne korzystające z tego samego materiału w jedno wywołanie rysowania, redukując Batches.

Krok 3. Współdzielenie materiałów

1. Jeśli dwa lub więcej obiektów używa identycznego materiału, sprawdzamy, czy przypisany jest ten sam Asset materiału (nie tylko wyglądający identycznie).
2. W razie potrzeby przypisujemy współdzielony materiał ręcznie, co redukuje SetPass calls.

Krok 4. Analiza porównawcza formatów

Na koniec przygotowujemy krótką notatkę (może być w pliku .md w folderze Documentation) porównującą trzy zaimportowane modele według:

- formatu pliku i jego wielkości
- liczby trójkątów (Tris, widoczne po zaznaczeniu obiektu w Stats)
- sposobu załączenia tekstur (wbudowane w plik vs. osobne pliki)
- jakości materiałów po imporcie (automatyczna vs. ręczna konfiguracja)

5. Zagadnienia dodatkowe

5.1. Normal Map: iluzja geometrii

Mapa (wektorów) normalnych to technika, która pozwala symulować drobne detale geometryczne (np. rysy, spoiny, pęknięcia) bez dodawania rzeczywistych trójkątow do siatki. Silnik oblicza oświetlenie dla każdego piksela tak, jakby powierzchnia była nieregularna, co tworzy przekonujący efekt trójwymiarowości.

W Unity tekstura Normal Map musi być oznaczona jako typ Normal Map w Import Settings (Inspector → Texture Type). Bez tej zmiany kolorów mapy normalne nie będą poprawnie interpretowane przez shader.

```
// Sprawdzenie, czy tekstura jest ustawiona jako Normal Map:
// Project → wybierz plik tekstury → Inspector
//   Texture Type: Normal Map   ← zmień jeśli ustawione na Default
//   Create from Grayscale: zaznacz jeśli masz mapę wysokości (heightmap)
// Kliknij Apply: Unity przekonwertuje plik.
```

5.2. Kanały UV: dlaczego tekstury się rozciągają

Jeśli po imporcie tekstura jest zniekształcona lub nieproporcjonalna, najczęstszym powodem jest zła mapa UV modelu. UV to dwuwymiarowe odwzorowanie trójwymiarowej siatki: można je porównać do rozkładania pudełka kartonowego na płasko. Niepoprawne rozkładanie UV powoduje rozciąganie, ściski lub powtarzanie wzorów w niewłaściwych miejscach.

W ramach zajęć nie modyfikujemy UV w Unity: ten etap pracy wykonuje się w oprogramowaniu do modelowania 3D (Blender, Maya, 3ds Max). Warto jednak umieć rozpoznać ten problem, by móc go zidentyfikować i zgłosić artyście 3D lub poszukać innego modelu.

6. Lista kontrolna wykonanych zadań

Przed zakończeniem laboratorium sprawdź, czy wszystkie poniższe elementy zostały zrealizowane:

☐	Struktura katalogów poszerzona o folder Documentation
☐	Pobrane co najmniej trzy modele 3D z różnych legalnych źródeł
☐	Zestawienie źródeł i licencji zapisane w folderze Documentation
☐	Wszystkie modele zaimportowane do Unity i widoczne w panelu Project
☐	Każdy model ma poprawnie skalową skalę, orientację i przypisane materiały
☐	Materiały skonfigurowane z poprawnymi mapami: Albedo, oraz Normal Map jeśli dostępna
☐	Hierarchia sceny zorganizowana z grupującym obiektem-rodzicem
☐	Co najmniej jeden model przekształcony w prefabrykat i dwukrotnie użyty w scenie
☐	Scena zoptymalizowana: obiekty statyczne oznaczone, materiały współdzielone
☐	Notatka porównawcza formatów i jakości modeli zapisana w Documentation

7. Materiały uzupełniające

Zasób	Opis
Unity Manual: Importing Models	Oficjalna dokumentacja procesu importu modeli, opis Import Settings.
Unity Manual: Materials and Shaders	Opis systemu materiałów i shaderów w Unity URP.
Unity in Action (J. Hocking)	Książka z listy literatury: rozdział o zasobach i ich integracji.
Poly Haven: polyhaven.com	Baza darmowych zasobów CC0: modele, tekstury, HDRi.
Sketchfab: sketchfab.com	Repozytorium modeli z filtrowaniem po licencji i formacie.
Learn OpenGL: PBR Introduction	Techniczne wprowadzenie do teorii PBR i map tekstur (j. angielski).