Wprowadzenie do silników gier

Karta opisu przedmiotu

Informacje podstawowe

Kierunek studiów Tworzenie Przestrzeni Wirtualnych i Gier	**Cykl dydaktyczny** 2025/2026
Specjalność -	**Kod przedmiotu** WIPPWGS.I2.17391.25
Jednostka organizacyjna Wydział Informatyki	**Języki wykładowe** polski
Poziom kształcenia Studia licencjackie I stopnia	**Obligatoryjność** Obowiązkowy
Forma studiów Stacjonarne	**Blok zajęciowy** Przedmioty kierunkowe
Profil studiów Ogólnoakademicki	**Przedmiot powiązany z badaniami naukowymi** Nie

Koordynator przedmiotu	Bernard Maj
Prowadzący zajęcia	Bernard Maj

Okres Semestr 2	**Forma zaliczenia** Zaliczenie **Forma prowadzenia i godziny zajęć** Wykład: 14 Ćwiczenia laboratoryjne: 28	**Liczba punktów ECTS** 4

Cele kształcenia dla przedmiotu

C1	Zapoznanie studentów z podstawowymi koncepcjami i architekturą popularnych silników gier komputerowych, takich jak Unity, Unreal Engine, CryEngine i Godot. Przekazanie wiedzy z zakresu tworzenia prototypów gier 2D i 3D, obejmującej programowanie skryptowe, zarządzanie assetami, fizykę oraz interfejs użytkownika. Uświadomienie słuchaczom problemów związanych z optymalizacją wydajności oraz projektowaniem mechanik gry w środowisku edytorów wizualnych.

Efekty uczenia się dla przedmiotu

Kod	Efekty w zakresie	Kierunkowe efekty uczenia się	Metody weryfikacji
Wiedzy – Student zna i rozumie:			
W1	Student zna i rozumie pojęcie silnika gier	PPWG1A_W04	Aktywność na zajęciach, Wykonanie ćwiczeń laboratoryjnych
W2	Student zna dobre i słabe strony różnych silników gier. Wie, który z nich zastosować do realizacji gry o zadanym scenariuszu.	PPWG1A_W04	Aktywność na zajęciach, Wykonanie ćwiczeń laboratoryjnych
Umiejętności – Student potrafi:			
U1	Student potrafi wykorzystać różne silniki gier do tworzenia prostych aplikacji.	PPWG1A_U07	Aktywność na zajęciach, Wykonanie ćwiczeń laboratoryjnych
Kompetencji społecznych – Student jest gotów do:			
K1	Student potrafi współpracować w zespole tworzącym oprogramowanie	PPWG1A_K02	Aktywność na zajęciach, Wykonanie ćwiczeń laboratoryjnych

Treści programowe zapewniające uzyskanie efektów uczenia się dla modułu zajęć

Wprowadzenie: różnorodność silników gier. Przetwarzanie projektów w różnych silnikach gier. Tworzenie prostych aplikacji.

Nakład pracy studenta

Rodzaje zajęć studenta	Średnia liczba godzin* przeznaczonych na zrealizowane aktywności
Wykład	14
Ćwiczenia laboratoryjne	28
Przygotowanie do zajęć	25
Samodzielne studiowanie tematyki zajęć	25
Przygotowanie projektu, prezentacji, pracy pisemnej, sprawozdania	15
Łączny nakład pracy studenta	**Liczba godzin** 107
Liczba godzin kontaktowych	**Liczba godzin** 42

* godzina (lekcyjna) oznacza 45 minut

Treści programowe

Lp.	Treści programowe	Efekty uczenia się dla przedmiotu	Formy prowadzenia zajęć
1.	#Wprowadzenie do silników gier Czym jest silnik gier: runtime, edytor, narzędzia i typowe podsystemy (grafika, fizyka, audio, UI). Przegląd Unity, Unreal Engine, CryEngine i Godot – ogólny profil i typowe zastosowania. #Podstawy pracy w edytorze Zakładanie projektu, podstawowe ustawienia, nawigacja po edytorze (widoki, sceny/poziomy). Obiekty w świecie: transformacje (pozycja/obrót/skala), hierarchia, grupowanie. Play mode/testowanie. #Tworzenie Sceny i obiektów gry Co to jest scena/level i jak ją organizować (podział na elementy statyczne i dynamiczne). Podejście komponentowe: obiekt + dołączone zachowania (idea wspólna dla większości silników). Prefab/podobne mechanizmy jako sposób ponownego użycia obiektów. #Skrypty i logika Jak powstaje logika gry: skrypt tekstowy lub narzędzia wizualne (na poziomie koncepcji). Zdarzenia i aktualizacja w czasie: input, prosty ruch, prosta interakcja. Debugowanie podstawowe: logi, inspektor/parametry. #Assety i materiały Co to są assety (modele, tekstury, dźwięki) i jak je importować oraz porządkować w projekcie. Podstawy materiałów i tekstur: przypisanie do obiektu, podstawowe parametry „wizualne”. Dobre praktyki: foldery, nazewnictwo i przygotowanie pod pracę zespołową. #Podstawowe elementy „grywalności” Kolizje i prosta fizyka: obiekty statyczne/dynamiczne, triggery, raycast. UI w wersji podstawowej: HUD/licznik, prosty ekran pauzy lub menu. Audio w wersji podstawowej: odtwarzanie SFX i podstawy audio 3D. #Budowanie i porównanie silników Minimalny „build” na PC: konfiguracje, uruchomienie wersji testowej, checklisty. Proste profilowanie/diagnoza problemów (co mierzyć i jak wyciągać wnioski na poziomie podstawowym). Podsumowanie porównawcze: szybkość prototypu, łatwość nauki, workflow edytora, problemy z silnikami..	W1, W2	Wykład

Lp.	Treści programowe	Efekty uczenia się dla przedmiotu	Formy prowadzenia zajęć
2.	Instalacja i konfiguracja środowiska (Unity, Unreal Engine, CryEngine, Godot + IDE). Repozytorium Git, struktura projektu, konwencje nazewnicze, wspólny pakiet assetów do porównań. Stworzenie prostych gier 2D i 3D w każdym z silników wykorzystując poniższe działania Ustalenie kryteriów porównania silników (czas wykonania, liczba kroków w edytorze, ergonomia, typowe problemy). utworzenie projektu i sceny, nawigacja po edytorze, podstawowe obiekty i transformacje. kamera, input i prosty ruch postaci/obiektu. kolizje i triggery, proste interakcje (np. podnoszenie obiektu, przełącznik). prosty system punktów/stanu gry, podstawowe debugowanie (logi/inspektor). podstawowe UI (HUD: licznik/zdrowie) oraz audio (SFX). przygotowanie i wykonanie builda na PC, podstawowe ustawienia build/paczki. Tworzenie własnej prostej gry w wybranym przez siebie silniku.	U1, K1	Ćwiczenia laboratoryjne

Informacje rozszerzone

Metody i techniki kształcenia :

Wykład, Mini wykład, Metoda warsztatowa (ang. workshop), Kształcenie zdalne, Praca grupowa, Dyskusja

Rodzaj zajęć	Metody zaliczenia	Warunki zaliczenia przedmiotu
Wykład	Aktywność na zajęciach, Wykonanie ćwiczeń laboratoryjnych	
Ćwiczenia laboratoryjne	Aktywność na zajęciach, Wykonanie ćwiczeń laboratoryjnych	

Dodatkowy opis

Wykłady mogą odbywać się w sposób stacjonarny (w salach) lub w trybie zdalnym na platformie MS Teams.

Warunki i sposób zaliczenia poszczególnych form zajęć, w tym zasady zaliczeń poprawkowych, a także warunki dopuszczenia do egzaminu

Ocena końcowa jest równa ocenie z zaliczenia zajęć laboratoryjnych, a ta wyznaczana jest skali określonej w regulaminie AGH na podstawie sumy punktów uzyskiwanych za uczestnictwo zajęciach i prezentowane rozwiązania postawionych problemów.

Sposób obliczania oceny końcowej

Aby uzyskać pozytywną ocenę końcową niezbędne jest uzyskanie pozytywnej oceny z zaliczenia zajęć laboratoryjnych – ocena końcowa jest taka jak ocena z zaliczenia zajęć laboratoryjnych.

Sposób i tryb wyrównywania zaległości powstałych wskutek nieobecności studenta na zajęciach

Zaległości powstałe wskutek nieobecności studenta na zajęciach warsztatowych można uzupełniać poprzez realizację ćwiczeń i dodatkowych zadań po wcześniejszym uzgodnieniu z prowadzącym.

Wymagania wstępne i dodatkowe

brak

Zasady udziału w poszczególnych zajęciach, ze wskazaniem, czy obecność studenta na zajęciach jest obowiązkowa

Wykład: obecność obowiązkowa, studenci uczestniczą w zajęciach poznając kolejne treści nauczania zgodnie z sylabusem przedmiotu. Studenci winni na bieżąco zadawać pytania i wyjaśniać wątpliwości. Rejestracja audiowizualna wykładu wymaga zgody prowadzącego.
Laboratoria: obecność obowiązkowa, studenci wykonują ćwiczenia warsztatowe zgodnie z materiałami udostępnionymi przez prowadzącego. Student jest zobowiązany do przygotowania się w przedmiocie wykonywanego ćwiczenia, co może zostać zweryfikowane podczas zajęć. Zaliczenie zajęć odbywa się na podstawie wyniku prac w postaci rozwiązania postawionego problemu.

Literatura

Obowiązkowa

1. Jacek Ross, Unity i C#. Praktyka programowania gier, Helion 2020

Dodatkowa

1. Kumsal Obuz, Game Development with Blender and Godot. Leverage the combined power of Blender and Godot for building a point-and-click adventure game, Helion 2022
2. Aram Cookson, Ryan Dowling Soka, Clinton Crumpler, Unreal Engine w 24 godziny. Nauka tworzenia gier, Helion 2017
3. Sam Howels, Richard G Marcoux, Riham Mohamed F Aly Aly Toulan, Samuel Howels, Chris Goodswen, Riham Toulan, Richard Marcoux III, CRYENGINE Game Development Blueprints. Perfect the art of creating CRYENGINE games through exciting, hands-on game development projects, Helion 2015

Kierunkowe efekty uczenia się

Kod	Treść
PPWG1A_K02	Jest przygotowany do współdziałania i pracy z innymi osobami w ramach zespołu projektowego gier, umie zorganizować pracę własną i zespołową w ramach realizacji wspólnych zadań i projektów.
PPWG1A_U07	Potrafi stosować narzędzia z zakresu projektowania graficznego, grafiki komputerowej i komunikacji wizualnej.
PPWG1A_W04	Zna i rozumie trendy rozwojowe różnych gatunków gier, modeli wirtualnej rzeczywistości i technik grafiki komputerowej i animacji.