Projekt gry platformowej 2.5D z wykorzystaniem X3DOM

Przedmiot: Języki Programowania Webowej Grafiki 3D
Autorzy: Adrian Krupa, Patryk Lesiak, Maciej Pieniążek **Data:** Listopad 2025

1. Wprowadzenie i cel projektu

Niniejszy dokument przedstawia koncepcję projektu interaktywnej gry platformowej 2.5D działającej w środowisku przeglądarki internetowej. Projekt ma na celu zademonstrowanie możliwości technologii X3DOM w zakresie tworzenia aplikacji interaktywnych łączących elementy grafiki trójwymiarowej z klasyczną mechaniką gier dwuwymiarowych.

Gra, roboczo nazwana **"Marek"**, inspirowana jest kultową serią Super Mario Bros. i stanowi przykład zastosowania deklaratywnego podejścia do programowania grafiki 3D w kontekście aplikacji rozrywkowych. Głównym założeniem projektowym jest połączenie intuicyjnej mechaniki platform 2D z nowoczesną prezentacją wizualną wykorzystującą renderowanie trójwymiarowe w czasie rzeczywistym.

2. Koncepcja rozgrywki

Projekt zakłada implementację gry platformowej, w której użytkownik kontroluje awatara poruszającego się w przestrzeni dwuwymiarowej (oś X - ruch poziomy, oś Y - ruch pionowy/skoki), podczas gdy wszystkie elementy gry reprezentowane są jako obiekty trójwymiarowe z pełnym oświetleniem, cieniami i perspektywą. Takie podejście, określane mianem "2.5D", pozwala zachować prostotę kontroli charakterystyczną dla klasycznych platformówek, jednocześnie oferując współczesną estetykę wizualną.

Cel rozgrywki: Użytkownik przemierza poziom od punktu startowego do mety, zbierając po drodze przedmioty (monety), pokonując lub unikając przeciwników oraz pokonując przeszkody. System punktacji będzie uwzględniał liczbę zebranych monet, czas przejścia poziomu oraz sposób ukończenia (wysokość dotarcia na maszt mety).

Interakcja z użytkownikiem: Kluczowym elementem projektu jest zapewnienie responsywnej i intuicyjnej kontroli poprzez obsługę klawiatury oraz realistyczną symulację fizyki ruchu (inercja, grawitacja, kolizje).

3. Uzasadnienie wyboru technologii

3.1. X3DOM jako framework graficzny

Projekt zostanie zrealizowany z wykorzystaniem technologii **X3DOM** - deklaratywnego frameworka do renderowania grafiki trójwymiarowej w środowisku przeglądarki. X3DOM stanowi implementację standardu X3D osadzoną w DOM (Document Object Model), umożliwiając tworzenie scen 3D poprzez znaczniki XML bezpośrednio w dokumencie HTML.

Argumenty za wyborem X3DOM:

- **Integracja z HTML/DOM** - scena 3D może być definiowana deklaratywnie w strukturze HTML i manipulowana standardowymi metodami JavaScript DOM API

- **Abstrakcja WebGL** - framework automatycznie zarządza kontekstem WebGL, kompilacją shaderów, macierzami transformacji oraz renderowaniem, eliminując konieczność niskopoziomowego programowania grafiki
- **Dostępność** - rozwiązanie działa natywnie w przeglądarkach bez konieczności instalacji dodatkowego oprogramowania czy wtyczek

3.2. Dodatkowe technologie wspierające

TypeScript zostanie wykorzystany jako główny język programowania logiki aplikacji. Statyczne typowanie oferowane przez TypeScript zwiększa niezawodność kodu oraz ułatwia rozwój większych projektów poprzez wykrywanie błędów na etapie kompilacji.

Vite posłuży jako narzędzie budujące projekt, zapewniając szybki serwer deweloperski z funkcją hot module replacement oraz optymalizację produkcyjnej wersji aplikacji.

3.3. Architektura oprogramowania

Kod aplikacji zostanie zorganizowany zgodnie z wzorcem **Entity-Component-System**, gdzie:

- **Encje** reprezentują obiekty gry (gracz, przeciwnicy, przedmioty)
- **Komponenty** definiują właściwości i zachowania encji (fizyka, rendering, AI)
- **Systemy** przetwarzają grupy encji z określonymi komponentami w każdej iteracji pętli gry

4. Projektowane mechaniki gry

4.1. System kontroli i fizyki

Obsługa wejścia użytkownika: Sterowanie zostanie zrealizowane poprzez obsługę zdarzeń klawiatury (klawisze strzałek lub WSAD dla ruchu poziomego, spacja dla skoku). System wejścia będzie agregował stan klawiszy w każdej klatce animacji, zapewniając responsywną kontrolę.

Symulacja fizyki: Implementacja uproszczonej fizyki 2D obejmie:

- Symulację grawitacji (stałe przyspieszenie w dół)
- Bezwładność i przyspieszenie postaci (stopniowe osiąganie maksymalnej prędkości)
- System wykrywania i rozwiązywania kolizji oparty na bryłach AABB (Axis-Aligned Bounding Boxes)

4.2. System przeciwników

Zaplanowane są trzy typy encji wrogich o zróżnicowanych wzorcach zachowań:

- **Typ A (Patrol):** Przeciwnik poruszający się w poziomie po określonej trasie, zawracający przy końcu platformy
- **Typ B (Skoczek):** Przeciwnik wykonujący skoki w regularnych odstępach czasu
- **Typ C (Żółw):** Przeciwnik z możliwością przyjęcia formy skorupy po ataku gracza, która może następnie zostać wprawiona w ruch i wykorzystana do eliminacji innych przeciwników

Mechanika interakcji: kolizja gracza z przeciwnikiem z góry (atak) eliminuje wroga, kontakt z boku powoduje utratę punktu życia gracza. W celu optymalizacji wydajności, przeciwnicy poza określonym promieniem od gracza będą usypiani, ich logika nie będzie przetwarzana.

4.3. Struktura poziomów

Środowisko gry będzie składać się z następujących elementów:

- **Platformy statyczne** - podstawowa geometria poziomu (podłoże, ściany, przeszkody)
- **Platformy dynamiczne** - elementy wykonujące ruch oscylacyjny lub obrotowy, przenoszące gracza
- **Przedmioty kolekcjonerskie** - monety implementowane jako triggery (wykrywają kolizję bez blokowania ruchu)
- **Obiekt mety** - maszt zakończenia poziomu z systemem premiowania wysokości dotarcia

Format poziomów: Definicje poziomów zostaną zapisane w formacie JSON, umożliwiając szybkie tworzenie i modyfikację zawartości bez ingerencji w kod źródłowy aplikacji.

4.4. Prezentacja wizualna

Scena będzie oświetlona kilkoma źródłami światła kierunkowego symulującymi naturalne warunki (światło słoneczne, ambientowe). Kamera będzie śledzić pozycję gracza z wygładzeniem ruchu, dodatkowo nieznacznie wyprzedzając go w kierunku poruszania się dla lepszej widoczności przestrzeni przed postacią.

5. Architektura systemu

5.1. Interfejs użytkownika

Aplikacja będzie zawierać:

- Menu główne z wyborem poziomów i wyświetlaniem wyników
- Nakładkę HUD (Heads-Up Display) prezentującą stan gry w trakcie rozgrywki
- System pauzy i opcji

Interfejs zostanie zrealizowany jako warstwa HTML/CSS renderowana nad sceną X3DOM, zapewniając łatwość stylizacji i responsywność.

5.2. Zarządzanie stanem gry

System zarządzania stanem będzie obejmował:

- Śledzenie postępu gracza (zebrane monety, ukończone poziomy, wyniki)
- Zapisywanie danych w localStorage przeglądarki
- Obsługę tablicy najlepszych wyników

5.3. Pętla gry

Główna pętla gry zostanie oparta na metodzie `requestAnimationFrame`, zapewniając synchronizację z odświeżaniem ekranu (docelowo 60 FPS). Wykorzystanie delty czasu pomiędzy klatkami zagwarantuje spójną symulację niezależnie od wydajności sprzętu.

6. Podsumowanie

Projekt zakłada stworzenie prostej gry platformowej działającej w przeglądarce, wykorzystującej X3DOM do renderowania grafiki 3D. Dzięki połączeniu klasycznej mechaniki platformówek 2D z trójwymiarową prezentacją wizualną, gra będzie intuicyjna w obsłudze, a jednocześnie atrakcyjna wizualnie.

Format poziomów oparty na JSON ułatwi tworzenie nowej zawartości, a modułowa struktura kodu pozwoli na rozbudowę gry o dodatkowe funkcjonalności. Projekt pokazuje praktyczne wykorzystanie X3DOM do tworzenia interaktywnych aplikacji webowych.