Politechnika Wrocławska

Wydział Informatyki i Zarządzania
kierunek studiów: Informatyka

Praca dyplomowa - inżynierska

Muzyczna gra na platformę Android

Daniel Kończyk

słowa kluczowe:
Android
Gra
Rytm

krótkie streszczenie:
Celem pracy jest projekt i implementacja gry rytmicznej na system Android. Przedstawiono istniejące na rynku rozwiązania oraz sposób projektowania aplikacji jaką jest gra, w tym emocjonalne podejście do wymagań funkcjonalnych, szczegóły mechanizmów rozgrywki, stylu wizualnego oraz interfejsu użytkownika, a także metody implementacji oraz napotkane problemy.

opiekun pracy dyplomowej	..... *Tytuł/stopień naukowy/imię i nazwisko*	..... *ocena*	..... *podpis*

Ostateczna ocena za pracę dyplomową

Przewodniczący Komisji egzaminu dyplomowego	..... *Tytuł/stopień naukowy/imię i nazwisko*	..... *ocena*	..... *podpis*

*Do celów archiwalnych pracę dyplomową zakwalifikowano do:**
a) kategorii A (akta wieczyste)
b) kategorii BE 50 (po 50 latach podlegające ekspertyzie)
** niepotrzebne skreślić*

pieczątka wydziałowa

Wrocław 2017

Dla moich rodziców, za całą wiedzę, którą mi przekazali, a także duszę artysty i inżyniera, którą we mnie ukształtowali, która to wiedzie mnie po wyboistej, aczkolwiek wynagradzającej ścieżce życia. A także dla mojej siostry Wiktorii… za nieograniczone wsparcie.

Streszczenie

Gry muzyczne zazwyczaj kojarzą się z wszelkimi odmianami karaoke lub z prostymi zabawami wymagającymi prostej interakcji w rytm granej muzyki. Jednakże, wraz z rozwojem branży na rynku pojawiają się nowe ich wcielenia, wykorzystujące innowacyjne mechaniki i wprowadzające nowych graczy do świata gier. Znaczną rolę odgrywają w tym platformy mobilne, będące jednymi z najbardziej przystępnych i wręcz niezbędnych do życia w tych czasach.

W pracy przedstawiony został projekt innowacyjnej gry muzycznej, będący nowym sposobem interpretacji tego gatunku, bazującym na analizie podbijających rynek produkcji zarówno wielkich studiów, jak i małych twórców. W ramach projektu sformułowane zostały wymagania gry, bazujące na sferze emocjonalnej gracza, a także opisany został koncept gry i rządzące nią mechaniki. Scharakteryzowany został także styl wizualny i współpracujący z nim interfejs użytkownika, będący nieodłącznym elementem aplikacji multimedialnych tego typu. Zaprezentowane zostały także wnioski dotyczące realizacji implementacji gry na platformy wykorzystujące jeden z najpopularniejszych systemów mobilnych – system Android.

Efektem pracy jest działająca gra muzyczna w wersji Alfa, wykorzystująca sporządzone ręcznie od zera grafiki, skrypty i shadery, prezentująca zaprojektowane mechaniki wykorzystujące rytm muzyki skomponowanej specjalnie na potrzeby tej aplikacji.

Abstract

Music video games are usually associated with karaoke kind of games as well as with simple plays that require basic interaction in a rhythm of music. However, as far as the game industry grows, new types of games arrive and present a variety of innovative mechanics, introducing new, fresh players into the world of gaming. Mobile platforms play a significant role in this, being the most accessible and approachable devices that are now an essential part of our lives.

In this paper, a project of an innovative music video game is presented, being a new approach to rhythmic video games, based on an analysis of market-conquering productions from big studios, as well as from single creators. Additionally, both formal and informal requirements are formulated within the framework of the project and built on an emotional sphere of the player. The game's design concept as well as core mechanics are listed and described thoroughly along the main foundations of the visual style, which are directly tied to the user interface, both being inseparable components of a multimedia application of this type. Furthermore, conclusions concerning the implementation of this project on one of the most popular mobile systems – the Android mobile system - are drawn and presented.

This work results in a well-working music game in version Alpha, consisting of graphical assets, shaders and scripts created manually from scratch, which introduces the designed mechanics and concepts utilising the rhythm of music, which was composed specifically for this application.

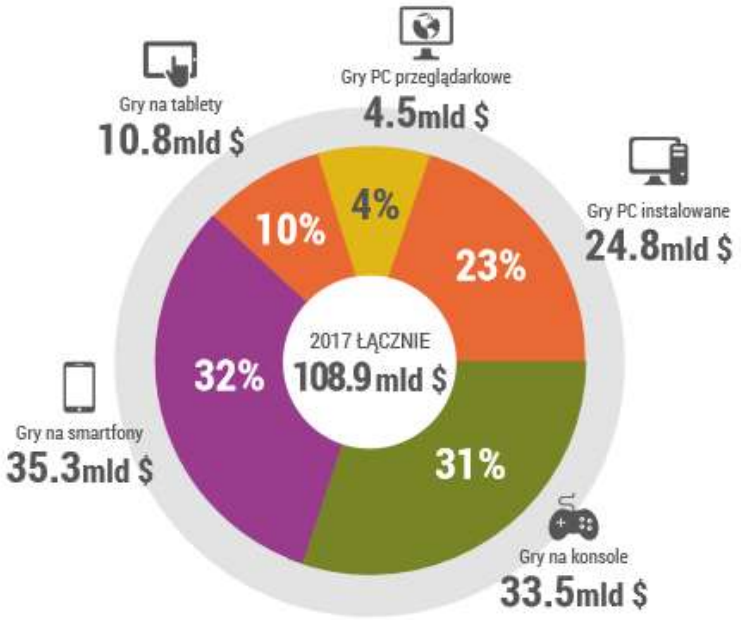
Spis treści

1. Wstęp

1.1. Wprowadzenie

Dwudziesty pierwszy wiek to okres rozwoju zarówno branży gier, jak i świata sprzętu komputerowego. Widoczna gołym okiem miniaturyzacja urządzeń codziennego użytku sprawia, że ludzie coraz częściej sięgają po smartfony i tablety, jako że cechują się one większą przystępnością dla przeciętnego konsumenta. Moc obliczeniowa tego sprzętu rośnie z roku na rok, co przekłada się bezpośrednio na atrakcyjność tych platform dla twórców gier.

Wartość rynku gier w roku 2017 wyceniona została na 108,9 miliardów dolarów, czego gry mobilne (na smartfony i tablety) to aż 42% tej wartości, co widać niżej na rysunku (Rysunek 1.1) [6]. Popularność gier na te platformy jest zatem faktem, a jako że ta część branży multimedialnej jest stosunkowo nowa (porównując do branży filmowej i muzycznej) istnieje wiele możliwości wprowadzenia innowacji i wykorzystania wszystkich dostępnych atutów urządzeń przenośnych, w tym przede wszystkim dotykowego ekranu.



Rysunek 1.1 - Wartość rynku gier wideo na rok 2017 (źródło: [6])

Gry muzyczne jako gatunek nie należą do najczęściej wybieranych przez deweloperów. Jako, że branża ta jest stosunkowo młoda, wiele pomysłów zostało po prostu jeszcze nie odkrytych – wielkie studia niechętnie eksperymentują z gatunkami nie gwarantującymi zysków bez podejmowania wielkiego ryzyka, a zespoły małe i niezależnie często zderzają się z problemem zdobycia odpowiednich licencji od wielu producentów muzyki, lub też z samym problemem innowacji koncepcji gatunku gier muzycznych – podstawowe interpretacje tego typu gier już powstały w postaci karaoke czy prostych gier wykorzystujących interakcje w rytm muzyki [18]. Niekiedy jednak pojawiają się nowe produkcje, łączące w sobie nowe mechaniki i innowacyjne, indywidualne podejście do wykorzystania rytmu muzycznego [18]. Gry te

najczęściej spotykają się z przytłaczająco pozytywnym odbiorem i sukcesem komercyjnym na miarę tych projektów. Rynek gier wzbogacany jest zatem o produkcje łączące w sposób nowoczesny różne gatunki, wplatając w podstawowe pojęcie rozgrywki dodatkowe elementy znane z innych rodzajów zabaw. Najczęstszym zjawiskiem jest wplatanie do gier elementów RPG („Role Playing Game") – dodanie do świata gry postaci, z którą identyfikować ma się gracz oraz mechanizmów, mających na celu wzmocnienie więzi emocjonalnej z bohaterem/bohaterami rozgrywki, takich jak posiadanie ekwipunku, rozwój umiejętności, możliwość prowadzenia dialogu i dokonywania wyborów najczęściej skutkuje większym zaangażowaniem gracza w rozgrywkę, co bezpośrednio przekłada się na lepsze doświadczenie z gry. Innym sposobem wprowadzania innowacji jest produkcja gier na nowe platformy, lub z wykorzystaniem innych kontrolerów, takich jak prawdziwe instrumenty, ekrany dotykowe lub kontrolery wykorzystujące rzeczywistość wirtualną. Kwestią wartą poruszenia przy tym jest też popularność takich platform i kontrolerów – smartfony i ekrany dotykowe jako systemy do gier są już bardziej popularne niż tradycyjne wykorzystywane do tego konsole.

Zaprojektowanie gry muzycznej wymaga zatem rozpatrzenia wielu możliwości. Nie jest to jedynie twór programistyczny, ale także dzieło wymagające podejścia kreatywnego i poddania się procesom temu podejściu towarzyszącym [7]. W konfrontacji z aplikacjami codziennego użytku, prym w definiowaniu wymagań gry wiodą potrzeby emocjonalne potencjalnego gracza [2][19] zamiast tradycyjnych, suchych funkcjonalności produktu. Gra jako wyrafinowana wersja aplikacji multimedialnej odbierana jest indywidualnie przez gracza, a to wymaga przeznaczenia uwagi na skrzętne zaprojektowanie stylu graficznego i języka wizualnego. Każda postać, wszystkie elementy i zajścia muszą być czytelne, mówić o swoich funkcjach i przeznaczeniach za pomocą samego wyglądu. Zarówno styl grafiki i interfejs użytkownika przekazywać muszą informacje szybko i skutecznie, zwłaszcza w grach muzycznych, jako że podejmowanie decyzji i interakcji odbywa się pod presją rytmu. Część audiowizualna gier ma na tyle duże znaczenie, że wymagają one takiej samej uwagi na etapie projektowania, co część programistyczna. Jest to przyczyną znacznych różnic w podejściu do inżynierii tego typu aplikacji.

Prototypowanie i implementacja gier wspomagane są przez oprogramowanie, zwane silnikami gier [7], będące zarówno narzędziami wyspecjalizowanymi, a także uniwersalnymi dla wszystkich rodzajów gier. Niektóre z tych technologii zakładają nawet możliwość lepienia gry z gotowych komponentów, bez potrzeby programowania albo produkcji grafiki. Rzadko jednak gry budowane w ten sposób osiągają znaczne sukcesy komercyjne, jako że brak im elastyczności na etapie produkcji, co wiąże się z trudnością wprowadzenia jakichkolwiek innowacji lub mechanik unikalnych, bez których gra może nigdy nie zostać zauważona na rynku. Inne, bardziej zaawansowane narzędzia dają szerokie spektrum możliwości oraz narzędzia, pozwalające urzeczywistnić nawet najodważniejsze pomysły, aczkolwiek opanowanie wszystkich funkcjonalności takich technologii zazwyczaj wymaga całego zespołu specjalistów. Ostatnimi czasy jednak pojawiły się silniki łączące w sobie prostotę i przystępność z szerokim wachlarzem możliwości, ułatwiając małym twórcom pracę nad ich projektami i czyniąc branżę gier wideo jeszcze bardziej popularną.

1.2. Cel pracy

Celem pracy jest projekt oraz implementacja gry muzycznej (rytmicznej) na platformy z systemem Android. Aplikacja projektowana zgodnie z nowoczesnymi zasadami projektowania gier . Jej celem zaspokojenie potrzeb emocjonalnych jak najszerszego grona graczy poprzez zagwarantowanie im rozrywki [1], rozróżniając potrzeby zgodnie z teorią

francuskiego intelektualisty i filozofa Rogera Caillois [2]. Mechanika gry wykorzystywać ma możliwości gwarantowane przez urządzenia z systemem Android (smartfony oraz tablety).

Zakres pracy obejmuje przygotowanie projektu gry w zakresie mechaniki, stylu graficznego oraz interfejsu użytkownika, a także zbudowanie aplikacji będącej realizacją tego projektu przynajmniej w wersji Alfa, czyli rzeczywistej, działającej gry, prezentującej zaprojektowane mechaniki, nie będącą jednak finalną i skończoną wersją gry gotową do wydania, wraz z przygotowaniem wszystkich potrzebnych grafik, utworów muzycznych, skryptów oraz shaderów (ang. „cieniowanie" – programy opisujące właściwości pikseli i wierzchołków). Projekt opracowany jest w oparciu o przeprowadzoną w pracy analizę istniejących już na rynku gier oraz dokonany jest wybór docelowego odbiorcy. W implementacji gry wykorzystane są najnowsze technologie oferujące łatwość w kreowaniu interakcji pomiędzy grą a graczem oraz wspomagające projektowanie szaty audiowizualnej, a także polepszające samą kulturę i komfort programowania i debugowania kodu, co przekłada się na mniejszą ilość błędów powstałych z powodu samego środowiska i kultury programowania.

Do realizacji pracy użyty został sprzęt oraz oprogramowanie w następujący sposób:

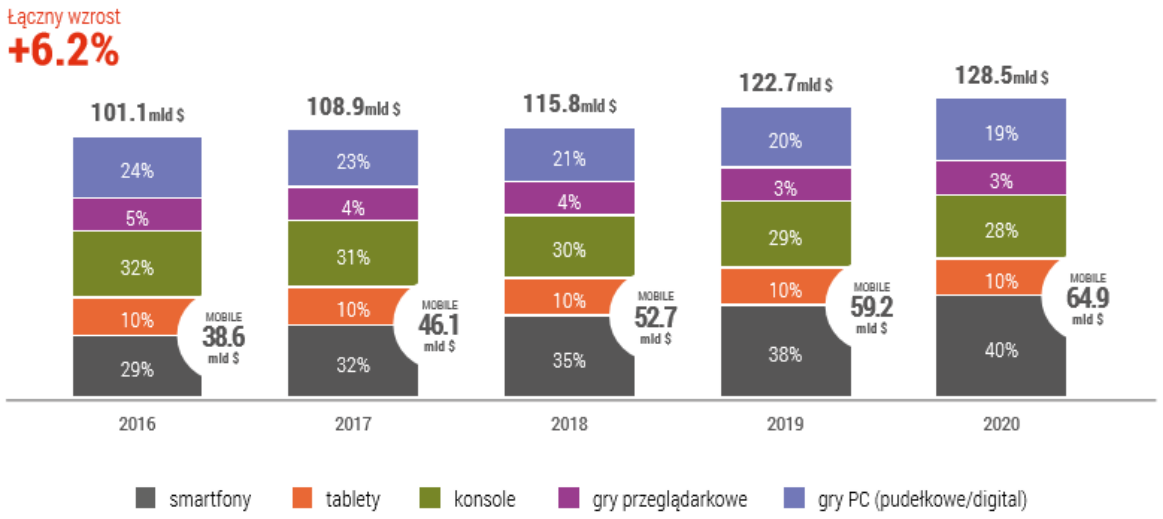
- Komputer PC (budowa niestandardowa)
  - System operacyjny: Windows 10 Pro
  - Środowisko programistyczne: Unreal Engine 4 wykorzystujący wbudowane środowiska „Material Editor", „Blueprints" oraz środowisko zewnętrzne Microsoft Visual Studio 2017
  - Wykorzystane oprogramowanie: Audacity (obróbka dźwięku), Adobe Photoshop CC (wykonanie elementów graficznych), TexturePacker (produkcja atlasów tekstur), Blender (obrabianie modeli 3d)
- Tablet Apple iPad Pro (2017)
  - System operacyjny: iOS 11
  - Wykorzystane oprogramowanie: Garage Band (kompozycja muzyki), Procreate (rysunki koncepcyjne)
- Smartfon Huawei P8 Lite
  - System operacyjny: Google Android 6.0 (wykorzystany do testowania gry)

Praca zawiera 4 główne rozdziały, podzielone na istotne podrozdziały. W rozdziale pierwszym zawarty jest wstęp, będący zagajeniem w kwestii projektowania gier oraz ogólnym wprowadzeniem w tematykę pracy. Rozdział drugi zawiera szczegółowy projekt aplikacji multimedialnej, jaką jest gra. W poszczególnych podrozdziałach przedstawiona jest analiza istniejących już na rynku gier, sformułowanie wymagań projektowych, wybór gatunku oraz docelowego odbiorcy gry, a także koncept samej gry, szczegółowy opis mechanik gry, pierworys stylu graficznego i ogólnych zasad języka wizualnego, a także plan interfejsu użytkownika. Trzeci rozdział prezentuje technologie i środowiska wykorzystane do implementacji wraz z unikalnym podejściem do programowania wykorzystującym te technologie. Dodatkowo omówione są tam także problemy, które wystąpiły podczas wytwarzania aplikacji oraz zastosowanie względem nich rozwiązania. Ostatni, czwarty rozdział zawiera istotne możliwości rozwoju gry a także ogólne podsumowanie, zawierające analizę postawionych założeń względem aplikacji wynikowej, a także impresje końcowe na temat działania samej gry. Praca zakończona jest spisem wykorzystanych źródeł w postaci literatury, artykułów, prezentacji i dokumentów.

2. Projekt

2.1. Analiza istniejących gier

Aż do niedawnej przeszłości gry były co najwyżej pasją wąskiej grupy osób szukających nowych metod rozrywki. Dziś natomiast jest to jedna z największych branż rozrywkowych na świecie. Wartość rynku gier na rok 2017 szacowana jest na 108,9 miliarda dolarów, natomiast prognozy na kolejne lata przewidują dalszy wzrost tej wartości, a także coraz to większy udział platform mobilnych (a zwłaszcza smartfonów) na rynku.



Rysunek 2.1 - Prognoza wartości globalnego rynku gier w latach 2016-2020 (źródło: [6])

Gry muzyczne i rytmiczne zajmują w tym swoje miejsce. Do przykładów gier muzycznych, który odniosły znaczący sukces w ostatnich latach należą między innymi:

1. **Crypt of the NecroDancer (**Rysunek 2.2**)**

Jest to innowacyjna, wygrywająca wiele nagród gra będąca wynikiem działalności studia Brace Yourself Games. Studio składa się zaledwie z sześciu osób. Polega ona na przejściu sterowaną postacią do wyjścia wykonując kroki w rytm. W wykonaniu tego zadania przeszkadzają graczowi przeciwnicy, którzy także poruszają się zgodnie z muzyką. W związku z tym gracz musi sobie z nimi poradzić – walczyć (oczywiście w rytm muzyki), uciekać i wykorzystywać elementy swojego ekwipunku, który zdobywa w czasie gry.

Crypt of the NecroDancer to hybryda gatunkowa – jest to połączenie gry RPG akcji z grą rytmiczną. Elementy RPG oraz akcji to przede wszystkim walka, zbieranie ekwipunku oraz sam cel gry. Część rytmiczna wywiera na graczu presję – muzyka leci i gracz nie może czekać i zastanawiać się, musi wykonywać ruchy w rytm, tak samo jak w tańcu. Takie połączenie elementów dwóch gatunków zagwarantowało powstanie produktu cieszącego się przytłaczająco pozytywnymi recenzjami.

Aby zrozumieć sukces tej gry trzeba zrozumieć sekret jej innowacji – na rynku jest wiele gier muzycznych wyprodukowanych przez potężne studia będących prostą interpretacją podstawowego gatunku jakim są gry rytmiczne/muzyczne. Crypt of the NecroDancer nie opiera więc swoich mechanik na tylko tej jednej mechanice, ale łączy ją z inną mechaniką, kreując z

dwóch gatunków coś nowego, co jeszcze nie zostało spopularyzowane. Takie hybrydy gatunków zapewniają innowację już na samej warstwie gatunku gry [1].

Kolejną warstwą potrzebującą innowacji jest sam „Core Gameplay" (ang. „rdzeń rozrywki") [1]. Jest to główna mechanika wokół której budowana jest cała gra. W tym przypadku jest to wykonywanie kroków postacią w rytm – można poruszyć postacią o jedno pole, jak pionkiem w szachach, tylko w dowolnym kierunku i to w każdym interwale rytmu grającej muzyki. Niewykonanie takiego ruchu skutkuje jego utratą – przeciwnicy zyskują wtedy na bezczynności gracza. Mechanika ta nie należy do spopularyzowanych już sposobów na rozgrywkę, zatem jest to innowacja, która czyni tę grę unikalną i wyróżnia ją na rynku. Do popularnych mechanik rozgrywki w sferze gier muzycznych należą między innymi mechanika naciskania wskazanego przycisku w rytm, mechanika zapamiętywania sekwencji przycisków i odgrywania ich w rytm (tzw. zabawa o nazwie „Szymon mówi"), czy mechaniki typu karaoke polegające na śpiewaniu odpowiednim tonem.

Rysunek 2.2 – Zdjęcie z gry „Crypt of the NecroDancer", Brace Yourself Games (źródło: Steampowered.com)

2. Rocksmith (Rysunek 2.3)

Rocksmith to gra wyprodukowana przez studio Ubisoft – jedną z wielkich korporacji odgrywających kluczową rolę w kształtowaniu całej branży. Posiada ono niezliczone środki i zasoby, które przeznacza na eksperymenty i produkcję gier, zawieranie kontraktów oraz analizę rynku. Gra Rocksmith powstała w kooperacji z producentem gitar Epiphone – polega ona na zwyczajnym graniu znanych i popularnych utworów, głównie rockowych, na autentycznej, prawdziwej gitarze elektrycznej. Razem z grą dostarczona mogła zostać także takowa gitara elektryczna oraz kabel Rocksmith Real Tone Cable, który pozwalał na podłączenie gitary (sprzęt analogowy) do wejścia cyfrowego komputera USB oraz kompatybilność tej gitary z grą.

Pod względem gatunku jest to zwykła gra muzyczna polegająca w rzeczywistości na wciśnięciu odpowiedniego przycisku w odpowiednim czasie i ewentualne zapamiętanie całej sekwencji przycisków (nauczenie się piosenki na pamięć). Elementem innowacyjnym jest w tym wypadku kontroler gry – zwyczajna gitara elektryczna. W porównaniu ze znanymi wszystkim matami do tańca gitary są jednak popularniejsze i praktyczniejsze – wiele osób

posiada gitary już po to, aby na nich po prostu grać, a to, że mogą podłączyć je do komputera i grać także w grę komputerową z jej pomocą to dodatek i przywilej każdego posiadacza gitary. Natomiast maty do tańca, poza zastosowaniem w grach, nie mają żadnej praktycznej wartości w większości przypadków.

Rysunek 2.3 - Zdjęcie z gry "Rocksmith", Ubisoft (źródło: Steampowered.com)

3. Audioshield (Rysunek 2.4)

Jest to gra zaprojektowana przez pojedynczą osobę - Dylan Fitterer. Powstała ona na platformy wykorzystujące wirtualną rzeczywistość – polega ona na strącaniu lecących w kierunku gracza w rytm muzyki obiektów za pomocą kontrolerów, które gracz trzyma w rękach. Na oczach założone ma gogle wirtualnej rzeczywistości, także widzi jak obiekty do strącenia lecą prosto na niego, a on się przed nimi broni.

Gatunkowo jest to zwykła gra zręcznościowo rytmiczna/muzyczna, lecz jej platforma – wirtualna rzeczywistość – zapewnia innowację, jako że takich gier na tego typu platformy po prostu jeszcze nie ma. Tradycyjne wciskanie przycisków w rytm zmienione jest tutaj na machanie kontrolerami w powietrzu i strącanie lecących w kierunku gracza obiektów i to wystarcza, aby zaoferować graczom coś nowego i innowacyjnego, co odróżni tę grę na tle innych.

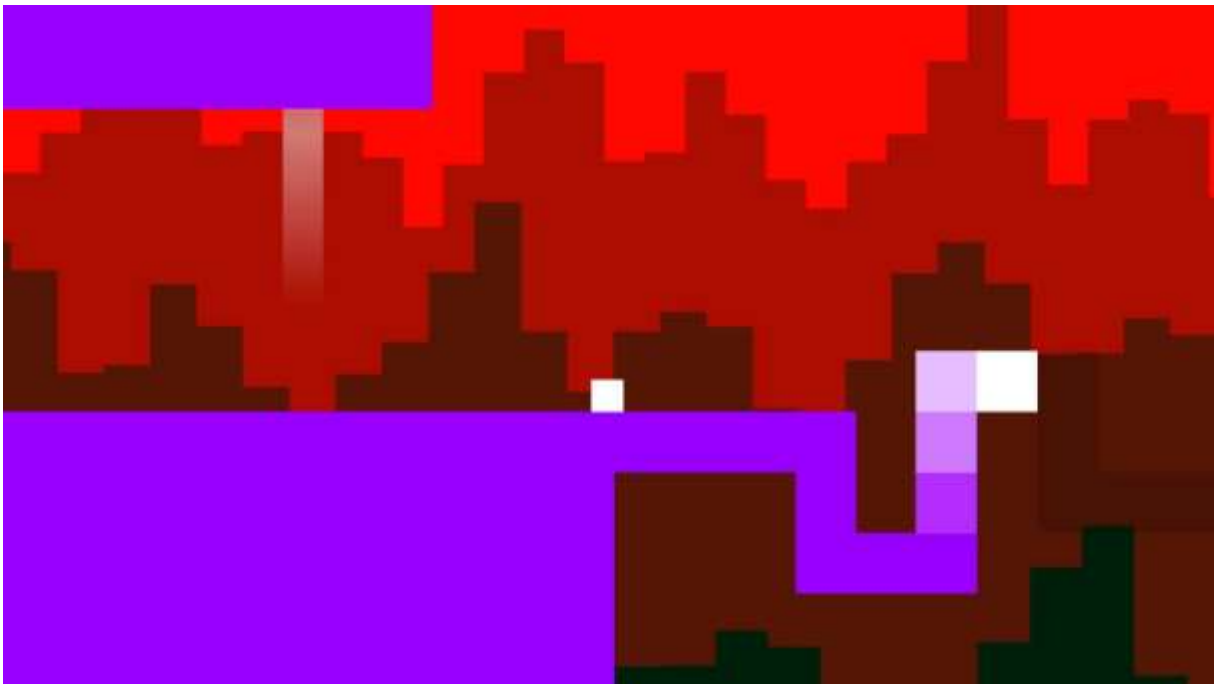
Dodatkowym elementem gry wartym uwagi jest system analizowania muzyki oraz system rankingowy – gra nie posiada własnych utworów muzycznych, lecz takie utwory do gry dostarczają użytkownicy za pomocą linku do platformy YouTube z utworem, lub bezpośrednio w postaci pliku. Gra analizuje muzykę pod kątem rytmu i akcentów rzuca w gracza obiektami zgodnie z opracowanym rytmem. Gdy gracz zakończy utwór umieszczany jest w rankingu, który kreowany jest na podstawie tytułów utworów / linków do platformy YouTube. Dzięki temu każdy może się dobrze bawić przy swojej własnej muzyce, a twórca gry nie odpowiada za to, jakie utwory są w grze wykorzystywane przez graczy.

Rysunek 2.4 - Zdjęcie z gry "Audioshield", Dylan Fitterer (źródło: Steampowered.com)

4. 140 (Rysunek 2.5)

140 to gra muzyczno-platformowa wyprodukowana przez mały zespół – Carlsen Games, sygnowany nazwiskiem głównego projektanta i programisty gry, Jeppe Carlsena. Jest to gra o całkowicie symplistycznej oprawie graficznej, skupiającej się jedynie do kształtów i płaskich kolorów, polegająca na przeprowadzeniu sterowanej figury geometrycznej do końca poziomu wykorzystując dostępne możliwości poruszania się oraz środowisko. Gra swoją prostotą przekonała do siebie duże ilości graczy, ciesząc się przytłaczająco pozytywnymi recenzjami oraz nagrodami.

W grze prezentowane jest środowisko składające się z różnych platform, ścian i wyrzutni, które aktywują i poruszają się w rytm grającej muzyki. Wykorzystana muzyka została skomponowana specjalnie do tej gry aby nadać odpowiedni klimat i odpowiedni rytm całemu doświadczeniu. Gra jest bardzo minimalistyczna, w związku z czym nie posiada żadnych innych mechanik, co także w pewien sposób ją wyróżnia.



Rysunek 2.5 - Zdjęcie z gry "140", Carlsen Games (źródło: Steampowered.com)

Dodatkowo, cechą wspólną wszystkich powyższych gier jest ich prezencja na serwisie YouTube – gracze jako fani muzyki lubią oglądać, jak ktoś gra ich ulubione utwory, tańczy do nich lub wykonuje inne czynności z nimi powiązane lub też po prostu gra w bardzo trudne gry. Nie jest to jednak tak popularne, jak gry angażujące emocjonalnie – gry posiadające dodatkową warstwę wymagającą zaangażowania emocjonalnego i podejmowania trudnych decyzji przyciągają graczy do oglądania i poznawania świata gry. Przykładem takiej gry jest **Undertale** (autor – Toby Fox) będąca grą małą i prostą, zdobywającą jednak przytłaczające ilości pozytywnych opinii dzięki właśnie dodatkowej warstwy angażującej emocjonalnie. Gry rytmiczne z reguły angażują emocjonalnie jedynie dzięki używanej muzyce – mogą jednak poszerzyć ten wachlarz możliwości dzięki połączeniu z innym gatunkiem. Podsumowując zatem wymienione wyżej gry wydzielić można cechy charakterystyczne gier w zależności od posiadanych środków, sytuacji na rynku czy też wielkości zespołu:

- **Gatunek gry i rodzaj muzyki** – innowacja samego gatunku gry jest bardzo prostym sposobem na wyróżnienie się na rynku gier tanim kosztem. Zaprojektowanie hybrydy gatunków otwiera szeroki wachlarz możliwości zamieszczenia nowych elementów rozgrywki – w przeciwnym wypadku pełny potencjał gry zależeć będzie tylko i wyłącznie od muzyki. Wielkie studia, takie jak Ubisoft mogą zawrzeć wiele umów z wytwórniami muzycznymi umożliwiając sobie użycie ich w grze (Rocksmith). Mali twórcy zmuszeni są polegać na utworach dostarczanych przez graczy (Audioshield) lub wyprodukowanych przez samych twórców (140). W przypadku hybrydy gatunków muzyka nie musi odgrywać głównej roli w grze, a jedynie wspomagać ją jako całość, co odciąża z producenta z potrzeby projektowania gry pod muzykę.

  Wynika z tego, iż wielkie studia jak Ubisoft mogą wykorzystać znane utwory muzyczne, które znajdują się poza zasięgiem małych studiów - te z kolei brną w innowację w postaci hybrydy gatunków albo stosują algorytmy pozwalające graczom dostarczać muzykę do gry.

- **Rdzeń rozgrywki i kontroler** – innowacja w tej sferze polega na zaprojektowaniu unikalnej mechaniki, która wyróżni grę na rynku. W Crypt of the NecroDancer i 140 wykorzystane zostały możliwości dostępne dzięki połączeniu gatunków, w grze Rocksmith oraz Audioshield mechaniki są jak najbardziej tradycyjne, aczkolwiek z wykorzystaniem nowej technologii. W przypadku Ubisoftu, studio pozwoliło sobie na wyprodukowanie własnego kabla przejściowego umożliwiającego użycie gitar elektrycznych, natomiast w Audioshield wykorzystany jest sprzęt do wirtualnej rzeczywistości. W przypadku gier na system Android dostępne będą jedynie możliwości gwarantowane przez smartfony – ekran dotykowy, akcelerometr, wibracje.

  Wynika z tego między innymi to, iż wielkie studia, takie jak Ubisoft, mogą wyprodukować własny sprzęt dołączany do gry, pozwalający na innowacyjną rozgrywkę. Małe studia zmuszone są poszukiwać innowacji w elementach hybrydy gatunków lub algorytmów, wykorzystując przy tym sytuację na rynku (tak jak świeżość gier muzycznych na platformach do wirtualnej rzczywistości).

2.2. Wymagania projektowe

Aplikacja projektowana na potrzeby tej pracy to gra. Charakteryzują się one bardziej kreatywnym i cięższym do zaplanowania podejściem niż zwykłe aplikacje użytkowe – w przeciwieństwie do programów i systemów działających w bankomatach, galeriach, sklepach, restauracjach i tym podobne, gry nie mają ściśle zdefiniowanego zestawu wymagań i funkcji.

Podejście do wymagań gier opiera się na sferze emocjonalnej oraz kreatywnej [2][19]– gra ma spełniać potrzeby należące do czterech kategorii w różnych natężeniach, zależących od docelowego odbiorcy oraz rodzaju gry [2][19]:

- „Konkurencja” – potrzeba konkurencji i porównywania się z innymi;
- „Szansa” – potrzeba wygrania czegoś poprzez przypadek, doświadczenie szczęścia;
- „Odgrywanie roli” – potrzeba wcielenia się w postać własną albo dostosowania się do nadanej roli;
- „Vertigo” (ang. „Zawrót głowy”) – potrzeba doświadczenia prędkości, nagłych zwrotów ruchu, czystej akcji.

Biorąc pod uwagę dzisiejsze możliwości technologiczne oraz wszystkie atuty, które gwarantuje ogólny rodzaj gry – gra rytmiczna – może ona spełniać wszystkie wyżej wymienione potrzeby, z różnym naciskiem na każdą z nich. Zapewnienie doznań ze wszystkich tych kategorii może uczynić grę atrakcyjną dla najszerszego grona odbiorców, a odpowiedni balans skieruje tę grę określonego odbiorcy.

Ponadto, projektując grę można kierować się kilkoma filozofiami:

- „Gameplay First” (ang. „Najpierw rozgrywka”) [1] – filozofia polegająca na zaprojektowaniu przede wszystkim jak najlepszej rozgrywki, stawiając wydajność, przekaz i inne wymagania na drugi plan;
- „Story First” (ang. „Najpierw historia”) – filozofia polegająca na zaprojektowaniu przede wszystkim jak najlepszej historii oraz przekazu gry, stawiając rozgrywkę, wydajność i inne wymagania na drugi plan;

- Kierowanie się innymi wymaganiami, jak wydajność, platforma docelowa i tym podobne – filozofia stosowania często w przeszłości, gdy sprzęt był bardzo ograniczony i dyktował sposób w jaki powstawały gry, lub też i dzisiaj, lecz rzadko – stosowana zazwyczaj przy implementacji gry na określone platformy na zamówienie, wydajność nie ogranicza już dziś funkcjonalności gier w stopniu zauważalnym.

W przypadku gry rytmicznej na system Android gra musi przede wszystkim być projektowana na ten system, natomiast gdy te wymagania są spełnione, najczęściej wybieranym wyjściem w branży jest aktualnie kierowanie się filozofią „Gameplay First". Projekt opisywany w pracy spełniać musi zatem następujące założenia:

- Jest to gra – szczególny rodzaj aplikacji multimedialnej, będącej interaktywnym źródłem rozrywki;

- Zaprojektowana jest ona specjalnie na platformy mobilne (wykorzystuje zatem zalety i możliwości smartfonów oraz ich unikalne właściwości), ze szczególnym naciskiem na system Android, kierując się filozofią „Gameplay First".

- Należy ona do gatunku gier rytmicznych – wykorzystuje elementy rytmu i/lub muzyki w mechanice oraz możliwie ma odzwierciedlenie w stylu wizualnym;

- Dostarcza szerokie spektrum emocji zaspokajających każdą kategorię potrzeb – potrzeba „konkurencji", „szansy", „odgrywania roli" oraz „vertigo".

Wymagania niefunkcjonalne są trudne do określenia – gra uruchamiana być może na każdym sprzęcie, jakim dysponuje potencjalny użytkownik. Stare smartfony mogą być skrajnie niekompatybilne z podstawową wersją technologii używanej do implementacji gier mobilnych, natomiast z biegiem czasu nowsze urządzenia mogą nie obsługiwać przestarzałych metod używanych w owych technologiach [17]. Dlatego też projekt można sformułować jedno podstawowe wymaganie niefunkcjonalne, charakterystyczne dla oprogramowania jakim są gry – **minimalne wymagania sprzętowe, dla których gra będzie oficjalnie kompatybilna**. W przypadku tego projektu za wyznacznik wymagań minimalnych uznany został smartfon wykorzystywany do testowania aplikacji Huawei P8 Lite – jest to telefon komórkowy będący przedstawicielem warstwy pośredniej pomiędzy smartfonami drogimi i wydajnymi, a smartfonami przestarzałymi. Gra uznana może zostać za oficjalnie grywalną i kompatybilną, jeżeli osiąga co najmniej 30 klatek renderowanych na sekundę, a opóźnienie w responsywności aplikacji jest niezauważalne gołym okiem.

Podsumowując zatem, aplikacja spełniać powinna następujące wymagania niefunkcjonalne:

- Gra powinna działać i być kompatybilna ze sprzętem określonym w „minimalnych wymaganiach sprzętowych" (wyznacznikiem takiego sprzętu jest w tym wypadku smartfon Huawei P8 Lite) – gra powinna osiągać co najmniej 30 klatek na sekundę i być responsywna – opóźnienia interakcji powinny być niezauważalne gołym okiem.

- Gra oraz użyta w niej technologia musi być kompatybilna z systemem Android oraz smartfonami.

2.3. Gatunek i docelowi odbiorcy

Zgodnie z wymaganiami projekt powinien być grą rytmiczną, a docelowe grono odbiorców, celem zapewnienia jak największego sukcesu biznesowego, jak najszersze [1]. Posiłkując się analizą aktualnego stanu rynku gier rytmicznych i mobilnych oraz zasad na tym rynku panujących (opisaną szczegółowo w punkcie 2.1) wyciągnąć można następujące wnioski:

- Gra implementowana przez małe studio – w tym przypadku, w zespole jednoosobowym – powinna być grą małą i prostą;

- Projektując grę z bardzo małą można podjąć duże ryzyko i poszukiwać innowacji w postaci hybrydy gatunków;

- Gry od małych i nieznanych studiów powinny angażować graczy emocjonalnie w jak najszerszym spektrum, aby mieć szansę trafić do odbiorcy bez potrzeby korzystania z kampanii reklamowych – gry bardzo angażujące emocjonalnie reklamują się same poprzez serwisy internetowe (jak YouTube lub Twitch.tv) i poprzez pocztę pantoflową, gdyż gracze chętnie oglądają reakcje oraz dzielą się swoimi reakcjami z innymi;

- Gry czysto rytmiczne/muzyczne wymagają uzyskania stosownych licencji na używanie utworów muzycznych lub instrumentów, gdyż to właśnie instrumenty i znana muzyka może zachęcić potencjalnych odbiorców gry do zakupu;

Gry hybrydowe, łączące elementy rytmu z innymi rodzajami rozgrywki są większym ryzykiem, lecz mogą zdobyć graczy dzięki swojej indywidualności, odwadze oraz właśnie istnieniu elementów rozgrywki innych niż muzyka i rytm – wiąże się to z przekonaniem, jako że muzyka jest jedynie dodatkiem, gdyż często muzyka towarzyszy ludziom w pracy lub przy jeździe samochodem, czy też wykonywaniu innych czynności, lecz bardzo rzadko jest jedynym i głównym medium na którym skupia się uwagę.

Biorąc pod uwagę powyższe wnioski, projekt powinien kierować się wytycznymi dla małych studiów, wymagających jak najmniejszego budżetu. Gra powinna angażować emocjonalnie, być odważna i innowacyjna poprzez połączenie elementów właściwych dla gry rytmicznej z innymi, znanymi gatunkami gier, zapewniając tym samym szersze spektrum możliwości rozwoju i poszerzając grupę odbiorców gry.

Gra zaprojektowana została zatem jako **dwuwymiarowa rytmiczna gra platformowa**, zawierająca jednocześnie **elementy RPG, pamięciowe oraz zręcznościowe.** Takie połączenie pozwala graczom na szkoleniu się i porównywaniu się, aby być lepszym (potrzeba „konkurencji"), pozwala wcielić się graczom w rolę sterowanej postaci dzięki elementom RPG (potrzeba „odgrywania roli"), wprowadza bazę do zaprojektowania pewnej losowości w elementach zręcznościowo platformowych (potrzeba szansy), oraz pozwala graczom doświadczyć prędkości i nagłości (potrzeba „vertigo").

Tego typu gra kierowana będzie zatem do szerokiego grona **osób znających dwuwymiarowe gry platformowe** (w tym klasyczne tytuły jak „Super Mario Bros" od Nintendo, lub „Megaman" od firmy Capcom). Jest to grupa wiekowa zrzeszająca jednocześnie nastolatków oraz ludzi dorosłych, a czasem także i starsze dzieci. Jest to także grupa, która najczęściej i najchętniej korzysta ze smartfonów, na które gra jest projektowana – według badań opublikowanych w styczniu 2017 roku aż 92% obywateli Stanów Zjednoczonych z przedziału wiekowego 18-29 posiada smartfony [11].

	Any cellphone	Smartphone	Cellphone, but not smartphone
Total	95%	77%	18%
Men	96%	78%	18%
Women	94%	75%	19%
White	94%	77%	17%
Black	94%	72%	23%
Hispanic	98%	75%	23%
Ages 18-29	100%	92%	8%
30-49	99%	88%	11%
50-64	97%	74%	23%
65+	80%	42%	38%
Less than high school graduate	92%	54%	39%
High school graduate	92%	69%	23%
Some college	96%	80%	16%
College graduate	97%	89%	8%
Less than $30,000	92%	64%	29%
$30,000-$49,999	95%	74%	21%
$50,000-$74,999	96%	83%	13%
$75,000+	99%	93%	6%
Urban	95%	77%	17%
Suburban	96%	79%	16%
Rural	94%	67%	27%

Source: Survey conducted Sept. 29-Nov. 6, 2016.

Rysunek 2.6 – Procent dorosłych w Stanach Zjednoczonych posiadających telefony komórkowe / smartfony, (źródło: [11])

2.4. Koncept gry

Gra projektowana jest zgodnie z zasadą „Gameplay First”, tak więc rozgrywka jest elementem do którego dostosowywane są inne właściwości gry. Natomiast aby rozgrywka zapewniać powinna wszystkie elementy gwarantujące jak najlepsze spełnienie czterech głównych potrzeb graczy, będąc jednocześnie czytelną w odbiorze, lekką i zrozumiałą w interakcji oraz wywołującą jak najmniej emocji negatywnych, psujących ogólny odbiór gry. Gra powinna zatem posiadać następujące cechy:

- „Easy to learn, hard to master” (ang. „Łatwa do nauczenia się, trudna do opanowania”) – gracz bardzo szybko wciąga się w rozgrywkę i uczy się podstawowych jej zasad, lecz opanowanie zaawansowanych technik wymaga poświęcenia – doświadczenie gry zależne jest od wymagań i umiejętności gracza.

- Unikalna szata graficzna – gra prezentuje się w sposób unikalny w porównaniu do innych gier oraz ułatwia i podkreśla odbiór emocjonalny. Szata graficzna podkreśla i wzmacnia rytmiczną część gry.

- Wolność poruszania się – gra pozwala graczowi decydować, gdzie i w jakim tempie iść, ograniczając jego spektrum poruszania się jedynie konstrukcją poziomów i umiejętnościami gracza.

- Zmienne tempo rozgrywki – pomiędzy fragmentami pełnymi akcji i rozgrywki rytmicznej gracz jest nagradzany odpoczynkiem, spowolnieniem rozgrywki i spokojną muzyką aby nigdy nie czuł się ani przytłoczony pod naporem akcji, ani nigdy nie znudził się brakiem stymulacji.

- Rytmiczne środowisko – Elementy środowiska rozgrywki zależne są od rytmu i wymagają od gracza takiej samej, rytmicznej interakcji. Dostosowanie się do rytmu nagradza gracza, niedostosowanie się do niego skutkuje karą.

- Dynamiczne łamigłówki – proste gry pamięciowe i poziomy wymagające myślenia przyczynowo-skutkowego pozwalają zróżnicować rozgrywkę, nawiązując z graczem dyskurs także na poziomie intelektualnym.

- Innowacyjność – implementacja nowych podejść i świeżych pomysłów pozwala doświadczyć graczom czegoś, czego nie zaoferuje im żadna inna gra.

- Rozgrywka bez sztucznych przerw – gra nie ma przerw (w postaci ekranów wczytywania), wytrącających gracza z rytmu rozgrywki czy błogości spokoju pomiędzy fragmentami akcji.

- Wielokrotna rozgrywka – grę można ukończyć wiele razy, na wiele sposobów, zależnie od poziomu wtajemniczenia czy umiejętności gracza.

2.5. Mechanika gry

2.5.1. Rozgrywka

Istotą gry jest nawigowanie przydzieloną postacią poprzez odpowiednio zaprojektowane poziomy i działające w rytm muzyki środowisko, a także podejmowanie interakcji ze światem w rytm owej muzyki. Gracz uznawany jest za zwycięzcę (wygrywa grę), gdy przejdzie odpowiednią ilość poziomów, w tym jeden z poziomów końcowych.

2.5.2. Postać i poruszanie się

Graczowi przydzielona jest odgórnie zdefiniowana na potrzeby rozgrywki postać główna, będąca jego awatarem w grze (Rysunek 2.7). Gracz może wchodzić w interakcję ze światem za pośrednictwem tej postaci i celem gry jest przeprowadzenie właśnie tej postaci na koniec poziomu. Postać zawiera następujące cechy:

- Zdrowie – wartość liczbowa całkowita, przyjmująca wartości od 0 do 100 włącznie, będąca odzwierciedleniem poziomu zdrowia awatara gracza. Gdy wskaźnik ten osiągnie 0 gracz uznawany jest za przegranego, rozgrywka jest przerywana i gracz cofany jest do miejsca specjalnie oznaczonego (patrz punkt 2.5.3).

- Mobilność – gracz może poruszać postacią w stylu gier platformowych. Dostępny jest ruch w prawo, w lewo oraz różne wariacje skoku, w szczególności skok pojedynczy, podwójny oraz skok na ścianie. Na styl poruszania się ma także wpływ środowisko – może go modyfikować lub odbierać możliwość ruchu. Poruszanie się (modyfikowane lub nie) daje graczowi poczucie prędkości, zwłaszcza pod presją rytmu, co zaspokaja jego potrzebę „vertigo".

- Interakcja z otoczeniem – postać wpływa na środowisko, a także środowisko może wpływać na postać – może ona zostać zraniona (obniżenie poziomu zdrowia), uleczona (podniesienie poziomu zdrowia), przeniesiona/zatrzymana (zmiana mobilności postaci).

Rysunek 2.7 – Postać gracza

Postać jest jednym z najważniejszych elementów rozgrywki dla gracza – od sposobu w jaki gracz steruje postacią zależny jest wynik końcowy, wygrana i przegrana gry. Postać jest także podmiotem, z którym identyfikuje się gracz – zaspokaja to jego potrzebę „odgrywania roli".

Dodatkowo, postać może nawiązywać dialog z postaciami pobocznymi, będącymi elementami środowiska (patrz punkt 2.5.3).

2.5.3. Środowisko i interakcja ze światem

Środowisko w którym porusza się gracz zależy od tempa rozgrywki. W sekcjach rytmicznych, pełnych akcji, środowisko zsynchronizowane jest z rytmem muzyki i wymaga od gracza interakcji w ten rytm. Można wyszczególnić następujące elementy środowiska:

- Podłoga (Rysunek 2.8) – są to stałe elementy środowiska, takie jak ziemia, po których poruszać się może postać bez przeszkód.

Rysunek 2.8 - Podłoga

- Platformy (Rysunek 2.9) – dynamiczne fragmenty podłogi, które poruszają i zmieniają swoje właściwości w zależności od rytmu. Postać gracza może po nich się poruszać, jednakże mogą one zmieniać pozycję gracza, przenosić go do innych miejsc. Wyrafinowanym przykładem platformy niezależnej od rytmu może być zwykła winda.

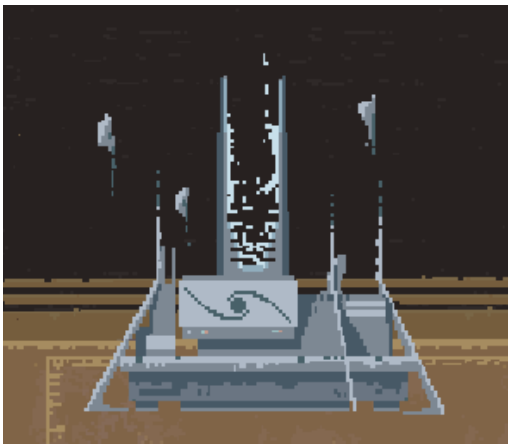
Rysunek 2.9 - Platforma

- Pułapki (Rysunek 2.10) – dynamiczne elementy środowiska, zależne od rytmu, w różny sposób podejmujące próbę interakcji z postacią gracza, skutkującej obniżeniem jej poziomu zdrowia, zmianą jej mobilności w sposób nagły/niespodziewany/niechciany przez gracza, ogólnie utrudniające graczowi dotarcie do końca poziomu. Nieznajomość przyszłych pułapek oraz ich funkcji wzmaga ciekawość gracza a także zaspokaja jego potrzebę „szansy", jako że przypadek może zadecydować o jego ominięciu pułapki.

Rysunek 2.10 – Pułapka (wieża strzelająca pociskami)

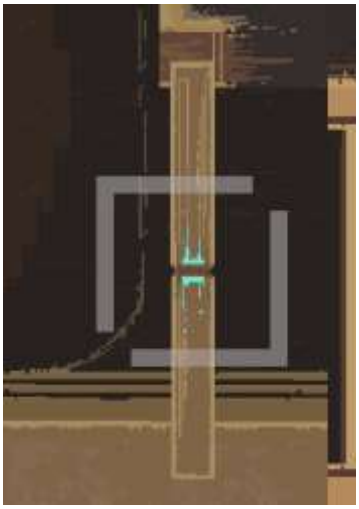
- Stacje zapisu (Rysunek 2.11) – stałe instancje podejmujące interakcję z postacią gracza, skutkującą odnowieniem poziomu zdrowia postaci oraz zapisaniem stanu gry. W wyniku

przegranej postać gracza przenoszona będzie do ostatniej stacji zapisu, z którą podjął interakcję.



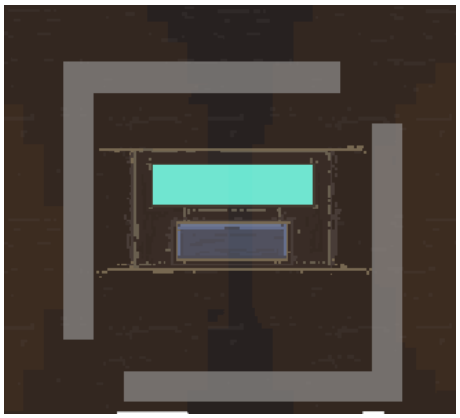
Rysunek 2.11 – Stacja zapisu

- Drzwi (Rysunek 2.12) – elementy interaktywne, ograniczające dostęp do poszczególnych części poziomu. Drzwi wymagać mogą jedynie interakcji w rytm, aby je otworzyć i uzyskać dostęp do kolejnych fragmentów poziomu, mogą też wymagać wciśnięcia przycisku, rozwiązania zagadki, posiadania odpowiedniego przedmiotu, poziomu zdrowia, odwiedzenia odpowiedniego fragmentu poziomu lub innych akcji angażujących gracza do eksploracji i pomysłowości.



Rysunek 2.12 - Drzwi

- Przyciski (Rysunek 2.13) – elementy interaktywne, z którymi gracz może wejść w interakcje. Interakcja z przyciskiem może otwierać/zamykać drzwi, aktywować/deaktywować pułapki lub wywoływać inne akcje, wpływające na stan postaci i poziomu.



Rysunek 2.13 - Przycisk

- Ściany (Rysunek 2.14) – stałe elementy środowiska, ograniczające możliwości poruszania się gracza, budujące swoiste pokoje. Ściany mogą być nieinteraktywne (całkowicie stałe), lub interaktywne (mogą zniknąć). Interakcja z taką ścianą może skutkować jej zniszczeniem i odblokowaniem dostępu do innych pokojów/poziomów.

Rysunek 2.14 - Ściana

- Postaci poboczne – elementy środowiska, z którymi postać wchodzi w interakcję. Skutkuje to przywołaniem okna dialogowego, przekazującego informacje i/lub pozwalające graczowi wybrać odpowiednią opcję (patrz punkt 2.7). Postaci poboczne pomagają zbudować historię świata i zaspokoić potrzebę „odgrywania roli” przez gracza (wcielając się w swojego awatara podejmuje za niego decyzje).

Środowisko jest głównym elementem rytmicznym w grze. To właśnie charakter środowiska, użyte pułapki oraz łamigłówki budują wyzwanie dla gracza.

2.5.4. Projekt poziomów

Projekt poziomów jest bezpośrednio połączony z projektem środowiska oraz interakcji gracza i jego postaci z nim. Projekt poziomów decyduje też o tempie gry, ustala warunki zwycięstwa i poziom trudności gry w zależności od drogi wybranej przez gracza i jego umiejętności [16]. Wymienić można zatem główne założenia projektu poziomów [16]:

- Użyteczność elementów – wszystkie elementy środowiska składające się na cały poziom muszą mieć swój cel, tak znaczący jak odcięcie graczowi dostępu do kolejnych poziomów, lub jedynie cel pomocniczy, jak wywoływanie odpowiednich walorów estetycznych czy emocjonalnych.

- Charakter poziomu – każdy poziom ma swój określony charakter i emocję, którą ma przekazywać. Emocja może być skomplikowanym odczuciem, którego trudno opisać słowami, aczkolwiek projekt poziomu nie może być nastawiony antagonistycznie do stylu i tempa rozgrywki (chyba, że celem takiego zabiegu jest właśnie wywołanie zakłopotania u gracza i odczucia niedopasowania elementów).

- Tempo poziomu – poziomy mają swoje określone tempo: odpoczynkowe, nie wymagające od gracza interakcji w rytm i pozwalające na swobodną eksplorację oraz tempo akcji, wymagające od gracza ciągłej uwagi, interakcji rytmicznej i podejmowania decyzji w czasie rzeczywistym.

- Głębia poziomu – poziom ma kilka możliwości przejścia poziomu, zależnych od stylu rozgrywki i umiejętności gracza lub jego zaangażowania w historię i fabułę gry. Nagradzanie gracza nowymi fragmentami rozgrywki za jego poprawę umiejętności gry zaspokaja potrzebę „konkurencji" gracza – może on stawać się coraz lepszy i odkrywać coraz więcej zawartości gry.

- Wiele poziomów końcowych – styl rozgrywki gracza, jego umiejętności oraz ciekawość świata gry świadczy o tym, jakie zakończenie powinien otrzymać. Natomiast zakończenie zależy bezpośrednio od poziomu końcowego, które są przygotowane na różne oczekiwania gracza.

- Sekretne pomieszczenia – aby wzmocnić ciekawość gracza na poziomach umieszczone są sekretne pomieszczenia, które można ujrzeć jedynie rozwiązując specjalne zagadki lub poddając się szczególnej eksploracji poziomu. Uznając, że w poziomie wszystko ma swoją funkcję i cel można prowadzić gracza do różnych konkluzji, które sprawią, że podejmie się dodatkowych wyzwań i użyje innych ścieżek w oczekiwaniu na nagrodę.

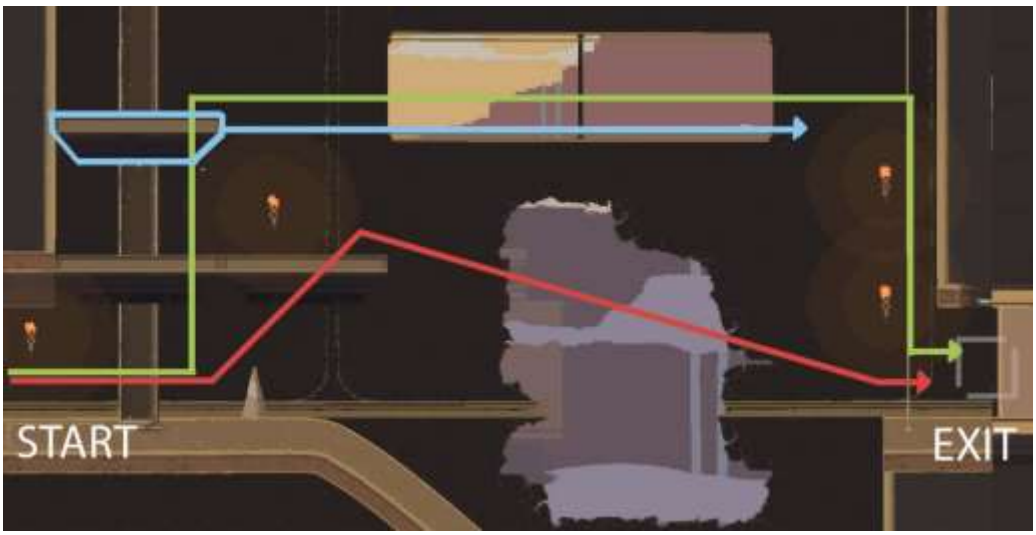
- Uczenie gracza – poziomy stopniowo przedstawiają graczowi nowe elementy środowiska, ucząc go o ich funkcji, pozwalając mu jednak zaryzykować (zaspokajając potrzebę „szansy").

Sposób projektowania poziomów jest głównym czynnikiem wpływającym na końcową jakość rozgrywki. Nieodpowiednie ułożenie punktów zapisu może skutkować negatywnymi emocjami gracza, tak samo jak zbyt duża ilość lub zła kompozycja pułapek czy platform może wydać się graczom za trudna, niezrozumiała lub nieodpowiadająca ich stylowi, co negatywnie wpłynie na ich odbiór rozgrywki. Dlatego też **poziomy zaprojektowane są z myślą o balansie stopnia trudności, nasycenia, czytelności i zaangażowania gracza.**

Jednym z zaprojektowanych poziomów przedstawiony jest na Rysunek 2.15. Jest to poziom mający na celu nauczenie gracza mechanizmów działania platform, dlatego też użycie ich jest wymagane do zakończenia tego poziomu. Jednakże, dla graczy mających doświadczenie i rozumiejących działanie platform zaprojektowana została druga, alternatywna ścieżka przejścia tego poziomu:

- Kolorem **zielonym** oznaczona jest ścieżka główna, dla graczy zapoznających się z mechanizmem platform. Polega ona na wykorzystaniu ruchomej platformy (oznaczonej kolorem **niebieskim**, wraz ze ścieżką jej poruszania się) do dotarcia do końca poziomu oznaczonego jako EXIT (ang. „wyjście") z miejsca startowego START.

- Kolorem **czerwonym** oznaczona jest ścieżka alternatywna, przygotowana dla graczy mających doświadczenie z platformami lub znających ten poziom. Ścieżka ta pozwala na bezpośrednie przeskoczenie do wyjścia (EXIT). Wymaga ona jednak znajomości mechanik rozgrywki (dalekie skakanie) oraz poziomu.

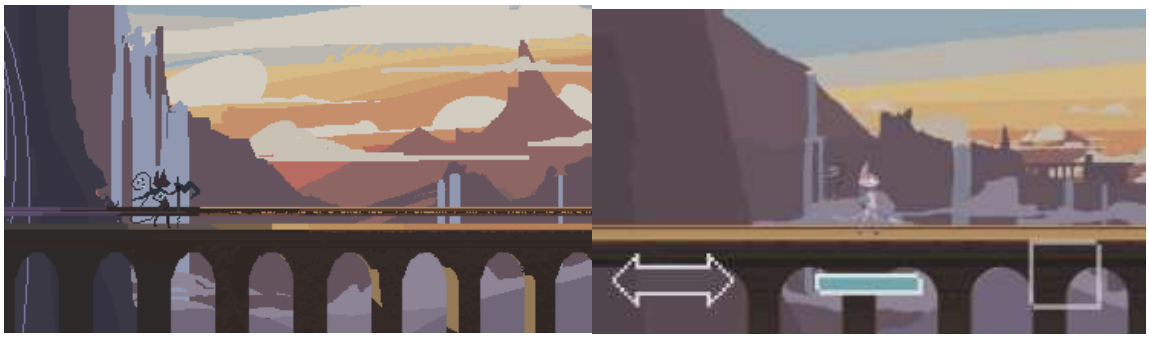


Rysunek 2.15 – Przykład zaprojektowanego poziomu

2.6. Styl wizualny

W grach styl wizualny jest jednym z głównych mediów przekazu, jaki twórcy wykorzystują aby przekazać emocje oraz stan sytuacji graczowi. Innym medium do tego służącym jest interfejs użytkownika (patrz punk 2.7), jednakże jest on zazwyczaj metodą bezpośrednią – tekstem, tabelkami, wskaźnikami, paskami, przyciskami, które nie są częścią świata gry, a jedynie przekazują graczowi informacje lub pozwalają informacje wprowadzić. Styl wizualny jest natomiast metodą pośrednią – jest częścią świata gry, komunikuje graczowi wygląd całego uniwersum oraz jego awatara, ale też może przekazywać informacje, tak jak interfejs użytkownika.

Projektowanie stylu wizualnego zaczyna się od narysowania grafik koncepcyjnych – prostych i szybkich rysunków mających na celu zagajenie wyglądu całej gry i uchwyceniu emocji [14] (Rysunek 2.16). W oparciu o grafiki koncepcyjne powstają projekty wszystkich poszczególnych elementów środowiska (pod względem wizualnym, mając na uwadze funkcję), wygląd postaci i charakter ogólnego klimatu.



Rysunek 2.16 – Porównanie grafiki koncepcyjnej (lewa strona) z finalnym ekranem gry (prawa strona)

Jednym z głównych zaprojektowanych elementów jest **„color code”** (ang. „kod koloru”). Jest to zestaw zasad kolorowania elementów środowiska gry (patrz punkt 2.5.3), mających na celu nauczyć gracza i przekazywać mu informacje o rodzaju i charakterze tych elementów. Tak samo jak w wypowiedziach pisemnych używa się wcięć, akapitów i pogrubień w ramach polepszenia czytelności, tak samo stosuje się odpowiednie jasności oraz kolory, aby ułatwić czytelność gry. W związku z tym ustalone zostały następujące zasady:

- Pułapki są dynamiczne (Rysunek 2.10, Rysunek 2.17) – w przeciwieństwie do innych elementów gry, **pułapkom towarzyszą dodatkowe efekty specjalne i wizualne**, odróżniając je od statycznego środowiska;

- Odróżnienie elementów raniących (Rysunek 2.10, Rysunek 2.17) – są one oznaczane kolorem **czerwonym**, dzięki czemu gracz nauczany jest, że czerwonych elementów trzeba unikać;

Rysunek 2.17 – Czerwony promień, może zranić gracza

- Czytelność akcji (Rysunek 2.18) – tło, na którym dzieje się akcja jest mniej szczegółowe i **ciemniejsze/jaśniejsze pod względem waloru**, aby ułatwić czytelność. Dotyczy to głównie fragmentów rozgrywki pełnej rytmu i akcji;

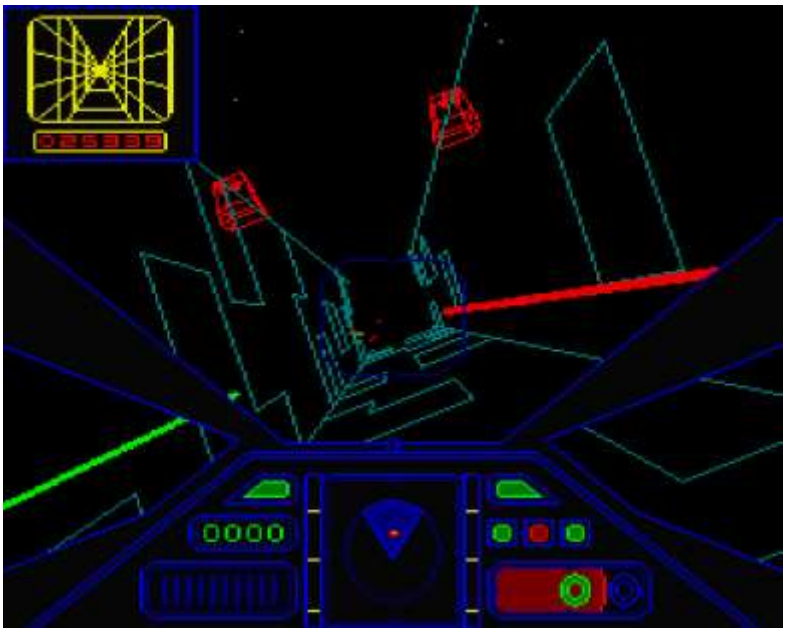
Rysunek 2.18 - Postać gracza na tle przeznaczonym dla akcji

- Rozpoznawalność punktów zapisu (Rysunek 2.11) – są one zaprojektowane inaczej niż inne elementy środowiska, przez co odcinają się od reszty świata i są łatwo rozpoznawalne przez gracza;

- Kolor bezpieczeństwa gracza – kolor **cyjanowy** jest kolorem, który oznacza coś dobrego dla gracza – bezpieczeństwo, interesujący go przedmiot lub odnowę punktów zdrowia. Uczy go to, że tego koloru nie należy się bać i można przy nim odpocząć. Kolor ten reprezentuje też zdrowie gracza.

Reszta kolorów ma jedynie funkcje estetyczne oraz emocjonalne – zbudowanie pięknej kompozycji kolorystycznej jest także ważne, gracze są wrażliwi na walory takie jak piękno. Dlatego też zastosowany „color code" ma także ugruntowanie w zbudowanej kompozycji.

Kolejnym elementem stylu wizualnego jest sam sposób budowania formy na ekranie. W grach trójwymiarowych formy wymagają oświetlenia, żeby zrozumieć ich trójwymiarową naturę mając do dyspozycji jedynie dwuwymiarowy ekran – w przeciwnym wypadku gra może stracić na czytelności głębi, albo wymagane są dodatkowe zabiegi (jak stosowanie grafiki typu „wireframe" (ang. „szkielet") (Rysunek 2.19)). W grach dwuwymiarowych natomiast ten problem nie występuje – głębia jest względna i abstrakcyjna, można zatem obyć się bez oświetlenia, a formy budowane są jedynie poprzez ich sylwetkę, kolory i walory – tak samo jak dzieje się to w dziełach sztuki, plakatach, animacjach dwuwymiarowych czy komiksie. Dlatego też dla dwuwymiarowej gry ważny jest wybór stylu budowania form.



Rysunek 2.19 - Zdjęcie z gry "Star Wars: Attack on the Death Star", Victor Musical Industries, Inc., 1991 – Grafika typu „wireframe" (źródło: Google.pl)

Rysunek 2.20 - Zdjęcie z gry "Super Mario Bros.", Nintendo, 1985 – Grafika typu „pixel art." (źródło: Google.pl)

Celując w odbiorców znających klasyczne gry platformowe (jak „Super Mario Bros od Nintendo") styl wizualny powinien nawiązywać do tego typu gier, dlatego projekt powstał w stylu **„pixel art"** (ang. „sztuka pikselowa") (Rysunek 2.20) [12]. Założeniem tego stylu jest to, że wszystkie elementy środowiska składają się z małej ilości pikseli mimo tego, że rozdzielczość ekranu jest zdecydowanie większa. Wykreowane w ten sposób formy pozwalają między innymi na:

- Oszczędność pamięci – każda grafika odpowiadająca poszczególnym elementom świata gry powstaje docelowo w bardzo małej rozdzielczości, co pozwala na zmieszczenie o wiele większej ilości takich grafik jak najmniejszym kosztem pamięci. Zestaw grafik widoczny na rysunku (Rysunek 2.21) pozwala na zaprojektowanie całych poziomów, a wymaga jedyne 37,4 KB pamięci;

Rysunek 2.21 - Zestaw grafik użytych w projekcie

- Zwiększona wydajność – gra renderowana jest w mniejszej rozdzielczości, co znacząco zmniejsza obciążenie smartfonu i pozwala na szybsze wczytywanie grafik i płynniejszą rozgrywkę, nie wymagając przy tym realistycznych sposobów przetwarzania obrazu [10], zmniejszając przy tym wymagania minimalne gry i otwierając ją na szersze grono odbiorców;

- Impresjonistyczne wrażenia [12] – mała ilość pikseli sprawia, że formy są jedynie umowne, a najbardziej widoczne stają się same kolory i uproszczone sylwetki. Nawiązuje to do impresjonistycznego kierunku w sztuce;

- Odczucie „retro” – styl ten nawiązuje do czasów, gdy sprzęt był na tyle ograniczony, że możliwe było jedynie renderowanie gier w bardzo niskich rozdzielczościach;

- Prostota wykonania [12] – wykreowanie pełnej animacji, zaprogramowanie realistycznego shadera czy wymodelowanie postaci w 3d zajmuje duże ilości czasu, podczas gdy zbudowanie elementów świata z zaledwie kilku pikseli jest bardzo szybkie i proste. Pozwala to na narysowanie zdecydowanie większej ilości grafik i prostych animacji w ograniczonym okresie czasowym.

Podsumowując zatem główne założenia stylu wizualnego – gra zaprojektowana jest w stylu „pixel art”, co zapewnia wydajność oraz prostotę i szybkość wykonania. Dodatkowo szczególny nacisk położony został na barwy i ich znaczenie – ułatwia to odbiór rozgrywki i buduje piękne kompozycje kolorystyczne.

2.7. Interfejs użytkownika

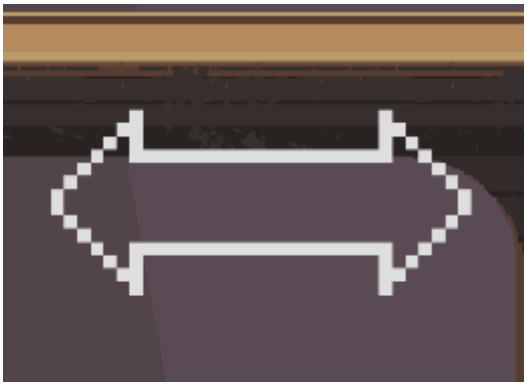
W grach interfejs użytkownika rozmywać się może ze światem gry, przeplatać się z nim i łączyć. Od niego zależy często, czy gra jest czytelna, łatwa w obsłudze i zrozumiała dla użytkownika [13]. Jest to jeden z głównych kanałów komunikacyjnych, który razem ze światem gry składa się na całość ekranu rozgrywki [13].

Interfejs użytkownika zaprojektowany jest z myślą o urządzeniach mobilnych (smartfony z systemem android), a interfejs taki wymaga specyficznego podejścia, powinien być więc między innymi [4]:

- Minimalistyczny – tabelki z tekstem i skomplikowane komunikaty utrudniają odbiór gry na smartfonach;
- Lekki – zajmuje mało miejsca w pamięci, szybko się ładuje, ma mały wpływ na wydajność;
- Spójny – nie odstaje od świata gry, a wręcz przeciwnie – pasuje do niego i do innych elementów spełniających funkcję interfejsu;
- Ikoniczny – ikony i kształty są łatwiejsze do zrozumienia i przekazują więcej informacji niż tekst;
- Znaczący – użyte kolory oraz kształty powinny spełniać funkcję i uczyć gracza (jak „color code", patrz punkt 2.6);
- Czytelny – użyte kolory, tekst i ramki powinny być duże i czytelne, aby na małym ekranie smartfonu użytkownik mógł bez problemu go zrozumieć.

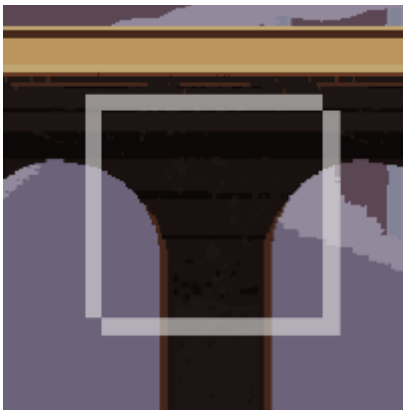
Stosując się do powyższej listy cech zaprojektowany jest interfejs gry – do oznaczania interfejsu wykorzystane są barwy bardzo jasne i bardzo ciemne, odcinające je od świata gry. Zastosowany jest także język kształtów oraz ikon. Spośród elementów interfejsu wymienić można:

- Przycisk odpowiadający za poruszanie się (Rysunek 2.22) – kształt wykorzystujący strzałki w obie strony mówi graczowi o kierunku, w którym poruszać się będzie postać. Naciśnięcie przycisku po lewej stronie skutkować będzie przesunięciem postaci w lewo, natomiast po prawej stronie przycisku – w lewo;



Rysunek 2.22 - Przycisk odpowiadający za poruszanie się

- Przycisk skoku (Rysunek 2.23) – pozwala postaci wykonać skok w górę gdy zostanie przyciśnięty. Wysokość/moc skoku zależy od tego, jak długo przycisk jest wciśnięty. Przyjęty jest także język kształtów – elementy interfejsu opakowywane są w ramki z uciętymi rogami, co ma graczowi pozwolić odróżnić elementy interfejsu od świata gry (wyjątkiem od tej reguły jest przycisk poruszania się ze względu na swój specyficzny kształt).



Rysunek 2.23 - Przycisk skoku

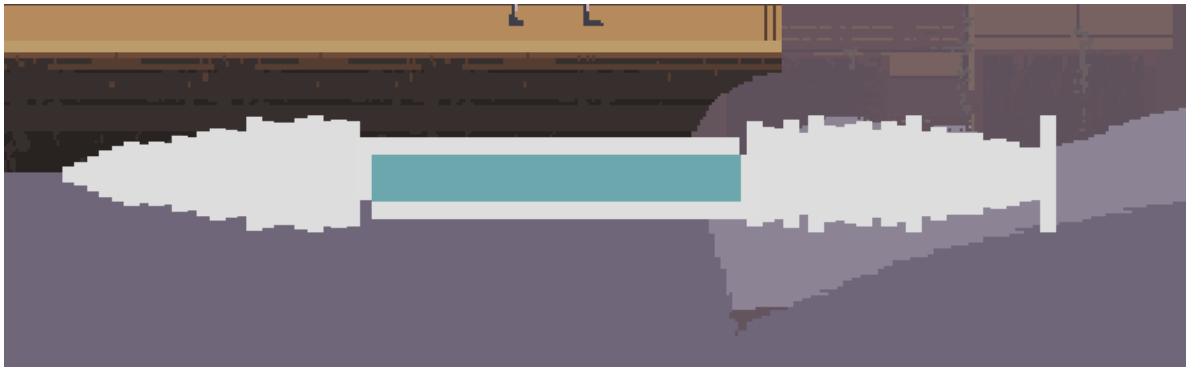
- Wskaźnik zdrowia (Rysunek 2.24) – przekazuje graczowi informację o stopniu zdrowia postaci. Zdrowie oznaczone jest kolorem cyjanowym, gdyż jest to kolor bezpieczeństwa gracza.

Rysunek 2.24 - Wskaźnik zdrowia postaci

- Okno dialogowe (Rysunek 2.25) – odpowiada za komunikację pomiędzy postaciami pobocznymi a postacią gracza. Przekazuje treść dialogów, a także jest wrażliwe na rytm – litery, słowa i zdania poruszać mogą się w rytm aby związać ten element interfejsu ze światem gry i ubogacić przekaz informacji o dodatkowe emocje – emocja i charakter wiadomości jest tak samo ważny, jak jej treść, tak samo jak język ciała jest ważny przy wymawianiu treści na głos w rozmowie z innymi ludźmi lub przy jakiejkolwiek prezentacji.

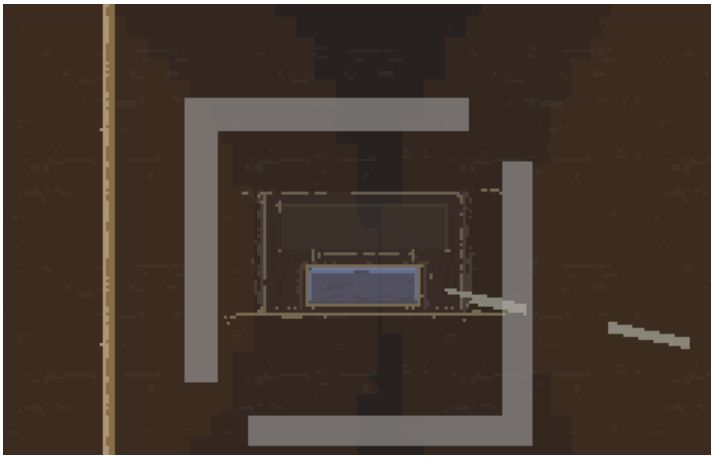
Rysunek 2.25 - Okno dialogowe

- Wizualizacja dźwięku (Rysunek 2.26) – wizualizuje muzykę w postaci dynamicznego spektrum dźwiękowego. Pomaga to graczowi zrozumieć naturę rytmu gry przekazując go także w warstwie interfejsu.



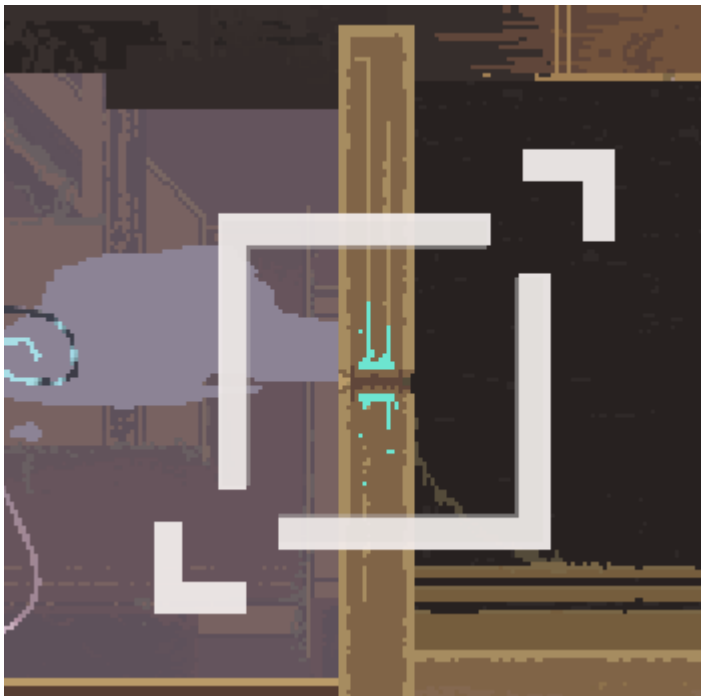
Rysunek 2.26 - Wizualizacja dźwięku

- Elementy interaktywne (Rysunek 2.27) – oznaczone jako kwadrat z mocno uciętymi rogami budują język kształtów gry – są to elementy wymagające bezpośredniej interakcji z graczem. Gracz może nacisnąć oznaczony tym rodzajem interfejsu element (dotknąć go na ekranie smartfonu), co skutkować będzie wykonaniem jakiejś akcji.



Rysunek 2.27 - Element interaktywny (brak rytmu)

- Elementy interaktywne w rytm – oznaczone są tak samo jak zwykłe obiekty interaktywne (Rysunek 2.27), aczkolwiek w rytm muzyki pulsują jak widać to na rysunku poniżej (Rysunek 2.28). Gracz informowany jest w ten sposób kiedy może podjąć interakcję z tym obiektem poprzez dotyk w rytm muzyki, za co jest nagradzany.



Rysunek 2.28 - Element interaktywny (w rytm)

Użycie stylu „pixel art” (patrz punkt 2.6) zdecydowanie ułatwia kwestię interfejsu – użycie grafik o niskiej rozdzielczości gwarantuje lekkość i minimalizm, a także wymaga ikoniczności, żeby całość przekazu zmieściła się na ekranie smartfonu. Użycie odpowiednich kolorów zapewnia spójność z ustalonymi w stylu wizualnym barwami i gwarantuje czytelność na tle reszty gry. Dodatkowo użyty jest odpowiedni język kształtów i ikon, mający na celu nauczyć gracza interakcji ze światem w odpowiedni sposób.

3. Implementacja

3.1. Wykorzystane technologie

Gry w dzisiejszych czasach są znacznie bardziej skomplikowanym utworem niż w przeszłości. Do produkcji gier odpowiednich na dzisiejszy rynek wymagana jest ekspertyza nie tylko programistów, ale też artystów, projektantów, grafików i specjalistów dziedzinowych (czasami fizyków, lekarzy, historyków) [10]. Wszyscy ci specjaliści (a w szczególności programiści, graficy, artyści i projektanci) mają szczególnie duży wkład w projekt, potrzebne jest zatem środowisko, które umożliwiłoby im pracę w swoich dziedzinach w jednym miejscu, przy tym samym projekcie. Z myślą tą powstały **silniki gier.**

Silniki izolują warstwę niskopoziomową gier (lub aplikacji użytkowych, gdyż takie także powstają z użyciem silników gier) od warstwy wysokopoziomowej, przystosowanej dla artystów i projektantów, którzy niekoniecznie rozumieją języki programowania, lub potrzebują odpowiedniej wizualizacji projektu. Silniki pozwalają także oglądać poszczególne elementy gry, zmieniać ich właściwości w czasie rzeczywistym bez potrzeby kompilacji całego projektu. Jest to narzędzie, które ze względów biznesowych oraz kultury pracy jest niemal niezbędne do powstania gry.

Rynek odpowiada na potrzeby twórców gier oferując szeroki wachlarz płatnych z góry, ale też i darmowych silników gier o możliwościach na najwyższym światowym poziomie [8]. Przykładami takich technologii jest między innymi silnik **Unity**, który zdobył sławę wśród twórców małych gier jako jeden z najprzystępniejszych i najprostszych w obsłudze. Innym przykładem silnika, który widnieje na rynku jest **Game Maker**, silnik bardzo prosty i o podstawowych funkcjach, przeznaczony jednak przede wszystkim dla osób początkujących w programowaniu.

Do implementacji gry użyty został jednak silnik **Unreal Engine 4**. Jest to zdecydowanie najbardziej zaawansowany i jeden z najpopularniejszych silników gier na rynku. Zapewnia on wsparcie dla systemów mobilnych (między innymi dla systemu Android), pozwalając na wykorzystanie pełnego potencjału smartfonów, jest też najbardziej zintegrowanym i pełnym środowiskiem, nie potrzebującym zewnętrznego oprogramowania lub dodatkowych wtyczek czy integracji, a oferującym wiele możliwości zarówno od strony programistycznej, jak i projektowej i artystycznej. Unreal Engine 4 współpracuje bezpośrednio z Visual Studio od Microsoftu, pozwalając na ingerencję w kod silnika, jak i dodawanie własnego kodu i metod bezpośrednio do gry w języku C++.

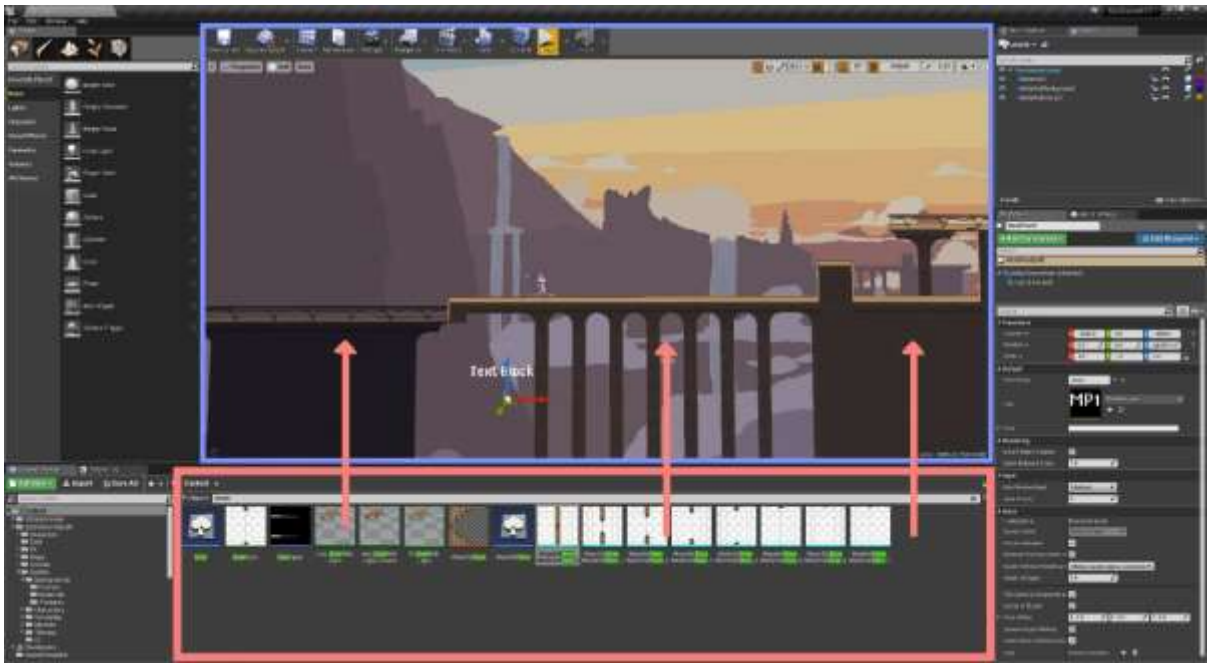
Grafika do gry narysowana została z użyciem programu Adobe Photoshop CC oraz aplikacji Procreate. Aplikacje te pozwoliły na malowanie cyfrowe zarówno w sposób zwykły, jak i bezpośrednio w stylu „pixel artu”. Do kompozycji muzyki użyta została natomiast aplikacje Garage Band oraz Audacity.

3.2. Środowisko i sposób implementacji

Środowiskiem, w którym implementowana jest aplikacja, jest środowisko silnika Unreal Engine 4. Silnik ten dostarcza wysokopoziomowych narzędzi, które ułatwiają i przyspieszają pracę nad grą. Do narzędzi udostępnianych przez ten silnik i wykorzystanych w tym projekcie należą między innymi:

- „Viewport” (ang. „widok/rzutnia”) (Rysunek 3.1) [3][5][7][9][15] – na bieżąco pokazuje świat gry, pozwala na jego edycję i projektowanie w czasie rzeczywistym. Ze spisu dostępnych, przygotowanych elementów środowiska gry (oznaczonego kolorem

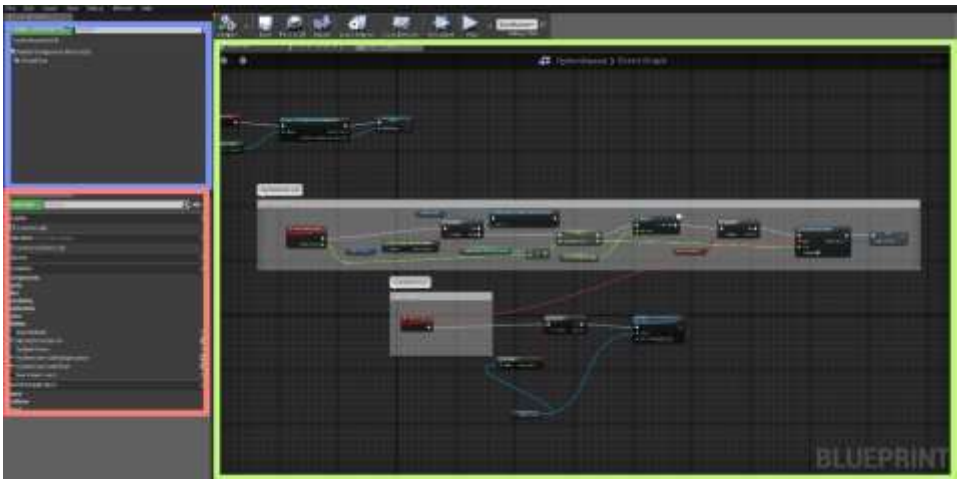
czerwonym) można przeciągać poszczególne instancje na ekran, który jest swoistym oknem na świat gry (oznaczony kolorem niebieskim). Korzystając z narzędzi umożliwiających precyzyjne ustawianie elementów można w ten sposób projektować i budować poziomy z przygotowanych wcześniej elementów, a także je ozdabiać.



Rysunek 3.1 - Widok środowiska Unreal Engine 4

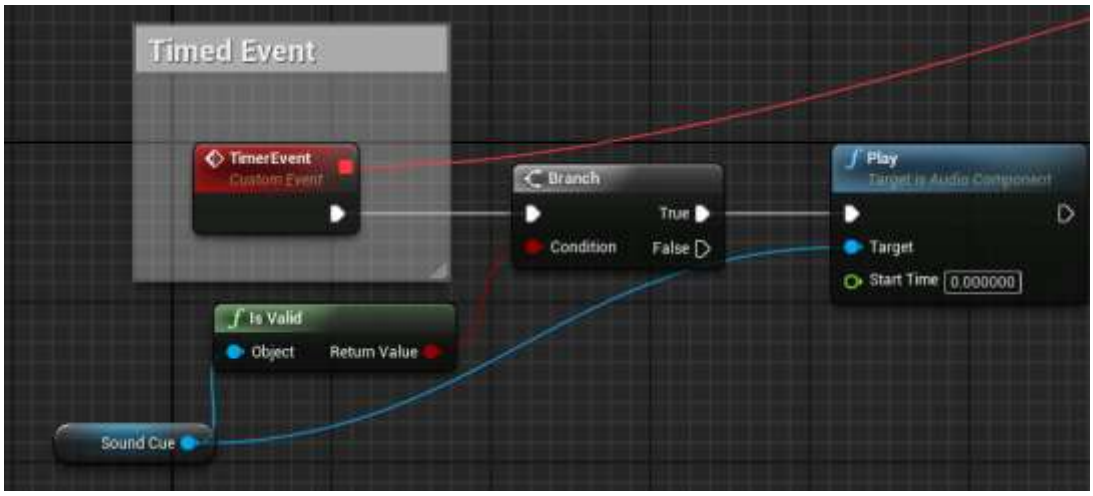
- „Blueprints" (ang. „plany") (Rysunek 3.2) [3][5][7][9][15] – jest to skomplikowany system wizualnego programowania klas gry. Tak samo jak tradycyjne klasy w znanych językach programowania, pozwalają na dziedziczenie, mają swoje zmiennie oraz metody. System „Blueprints" pozwala na łatwe debugowanie kodu gry powstającej w tym systemie w czasie jej działania oraz wizualizuje przepływ logiki całej aplikacji.

  „Blueprint" (Rysunek 3.2) to skomplikowana struktura, która składa się z **grafów, funkcji, zmiennych** (oznaczonych kolorem czerwonym) oraz **komponentów** (w ramce oznaczonej kolorem niebieskim). Środek ekranu zajmuje podgląd wybranego grafu lub funkcji (widok w ramce oznaczonej kolorem zielonym). Komponenty to elementy wizualne oraz mechaniczne, dla których „Blueprint" jest kontenerem. Można zatem do niego dodawać komponenty oraz logikę, jaka zachodzić będzie pomiędzy nimi samymi oraz światem w postaci skryptów w grafach oraz funkcji.



Rysunek 3.2 – Blueprint

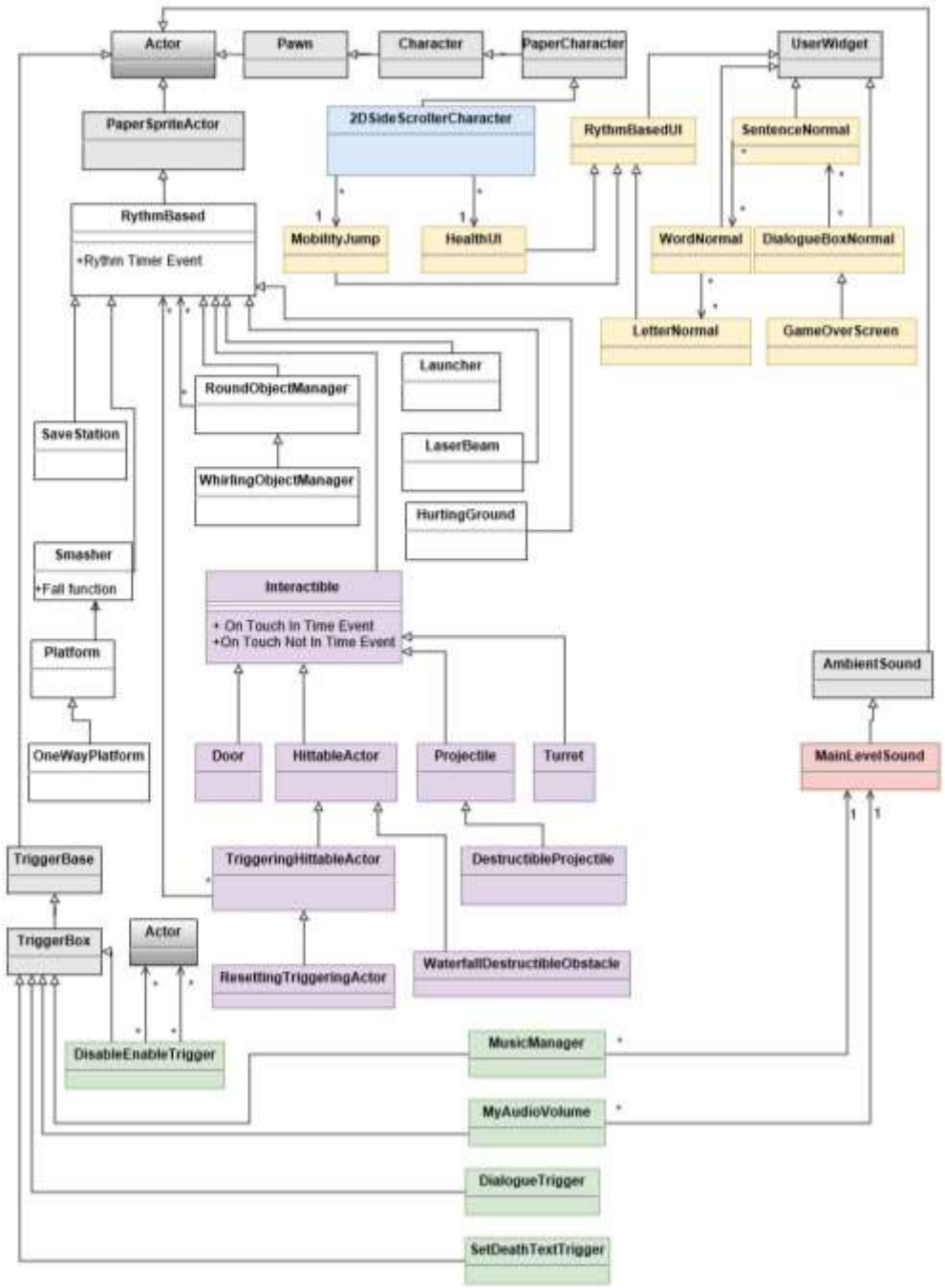
Graf „Blueprintu” (Rysunek 3.3) składa się z trzech rodzajów bloków: **„Eventów”** (ang. „event” – „zdarzenie”, blok czerwony), **funkcji** (bloki niebieskie/zielone z literką „f”) oraz **zmiennych** połączonych ze sobą liniami oznaczającymi przepływ logiki programu lub wartości zmiennych [9]. Na rysunku (Rysunek 3.3) przedstawiona jest implementacja odgrywania dźwięku w rytm muzyki przez obiekty klasy „RythmBased” i klas po niej dziedziczących, oznaczonych kolorem białym i fioletowym na uproszczonym diagramie klas (Rysunek 3.4). Działanie tego fragmentu grafu wygląda następująco: w rytm muzyki wywoływane jest zdarzenie „TimerEvent”, które odgrywa dźwięk zapisany w zmiennej „Sound Cue” wtedy i tylko wtedy, gdy funkcja „Is Valid” (walidująca, czy zmienna jest zdefiniowana) zwróci wartość prawdziwą. Gdy dźwięk nie jest zdefiniowany, nic się nie dzieje.



Rysunek 3.3 - Fragment grafu "Blueprintu"

„Blueprinty” mogą dziedziczyć po klasach napisanych w C++ lub po innych „Blueprintach” na tych samych zasadach [5]. Jedynie klasy w C++ nie mogą dziedziczyć po „Blueprintach”, jako że są to twory wewnętrzne i charakterystyczne tylko dla środowiska wewnętrznego silnika Unreal Engine 4. Istnieje możliwość konwersji całych „Blueprintów” do kodu w C++, aczkolwiek wygenerowany kod może być skrajnie niezrozumiały dla człowieka [5]. Na uproszczonym diagramie klas (Rysunek 3.4) przedstawiona jest struktura

dziedziczenia klas w projekcie. Klasy oznaczone **kolorem szarym** to bazowe klasy silnika Unreal Engine 4. Klasy **koloru żółtego** to „Blueprinty" interfejsu użytkownika, **kolor biały** odpowiada klasom, których działanie uzależnione jest od rytmu („eventu TimerEvent", Rysunek 3.3). **Kolor fioletowy** oznacza „Blueprinty" pozwalające na interakcję w rytm muzyki, **kolor zielony** to kolor klas wchodzących w interakcję z postacią użytkownika bez jego wiedzy (obiekty wykonujące jakąś akcję przy kolizji z postacią gracza). **Kolor niebieski** odpowiada „Blueprintowi" postaci gracza, a **kolor czerwony** to kolor klas dźwięku.

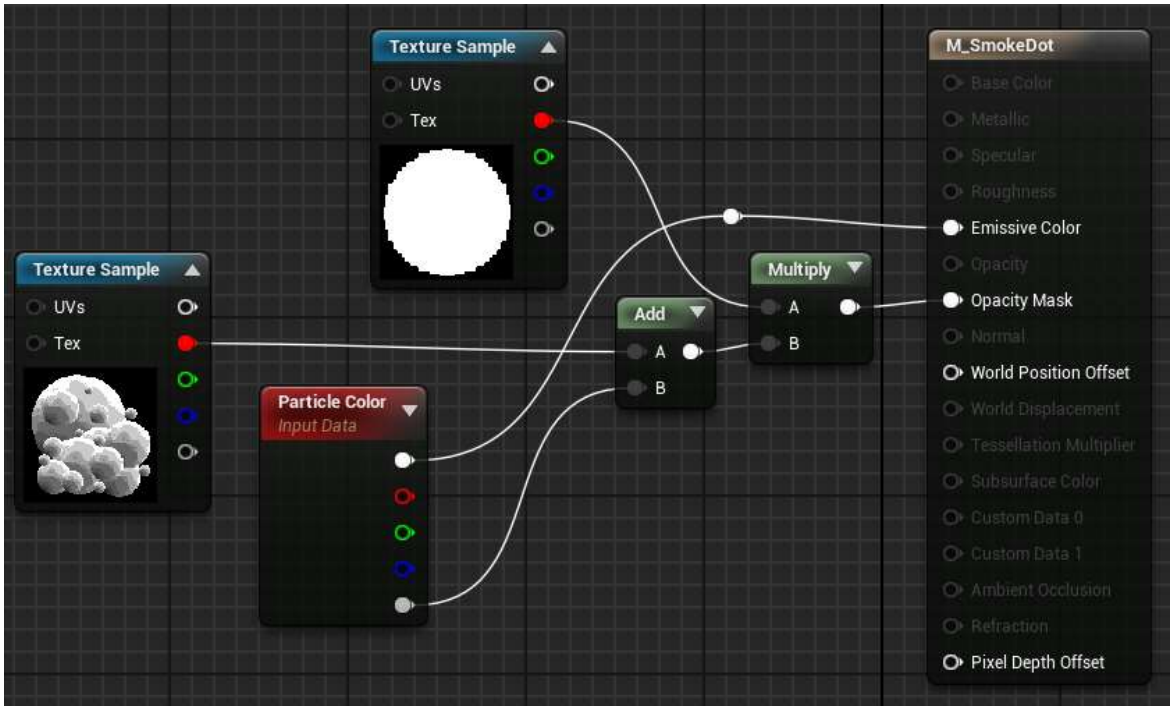


Rysunek 3.4 - Uproszczony diagram klas gry

- „Material Editor" (ang. „edytor materiałów") [5][7][9][15] – podobnie jak system „Blueprints", jest to wizualny system programowania shaderów. Pozwala na podgląd działania shadera w czasie rzeczywistym, przez co produkowane mogą być one z myślą

artystyczną, a nie programistyczną – artysta projektujący shadery nie musi znać semantyki języka czy zasad programowania, a jedynie za pomocą prostych operacji matematycznych oddziałuje na finalne działanie shadera oraz opakowuje go razem z teksturami w instancję zwaną „materiałem”.

Jednym z materiałów przygotowanych do gry jest materiał M_SmokeDot (Rysunek 3.5). Jest to materiał niereagujący na światło, posiadający maskę przeźroczystości oraz jest przystosowany do użycia w efektach specjalnych. Jak widać na grafie tego materiału, używane są dwie tekstury czarno-białe. Są to maski przeźroczystości, które poprzez serię matematycznych działań („Add” – ang. „dodawanie” oraz „Multiply” – ang. „mnożenie”) wraz z parametrem efektu cząsteczkowego (czerwony blok „Particle Color” – ang, „kolor cząstki”) dają w wyniku końcowym obraz w kształcie dymu oraz o kolorze pobieranym z efektu cząsteczkowego (Rysunek 3.6).

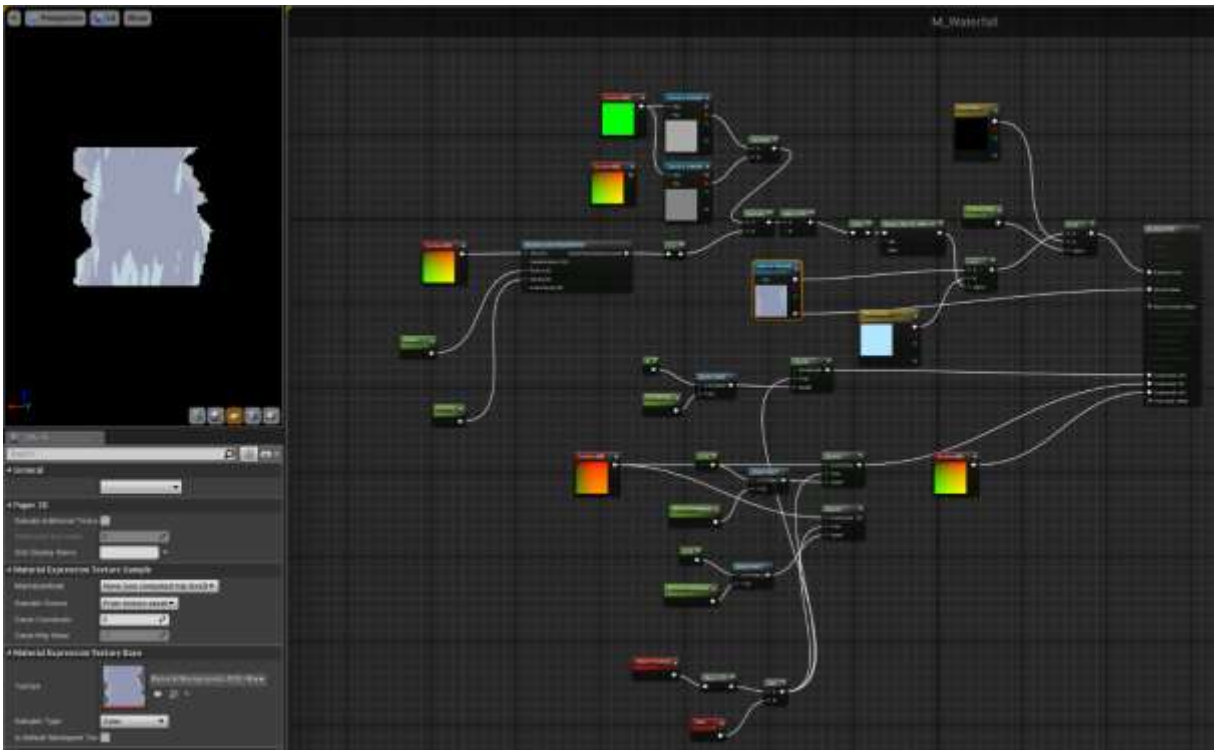


Rysunek 3.5 - Graf materiału M_SmokeDot

Rysunek 3.6 - Efekt cząsteczkowy dymu

Dzięki takiemu podejściu, artyści są w stanie projektować materiały i shadery bez potrzeby poznawania semantyki i specyfikacji kodu, służącego do pisania shaderów, jak i z pomocą w postaci wizualizacji finalnego efektu materiału w czasie rzeczywistym. Na

poniższym grafie (Rysunek 3.7) widoczny jest bardziej skomplikowany materiał, którego efekt finalny przedstawiony jest w lewym górnym rogu rysunku. Taki materiał odpowiada 2128 liniom kodu w języku HLSL nastawionym na wieloplatformowość oraz kompatybilność z silnikiem i jego funkcjami.



Rysunek 3.7 - Graf materiału M_Waterfall

3.3. Problemy implementacyjne i platformowe

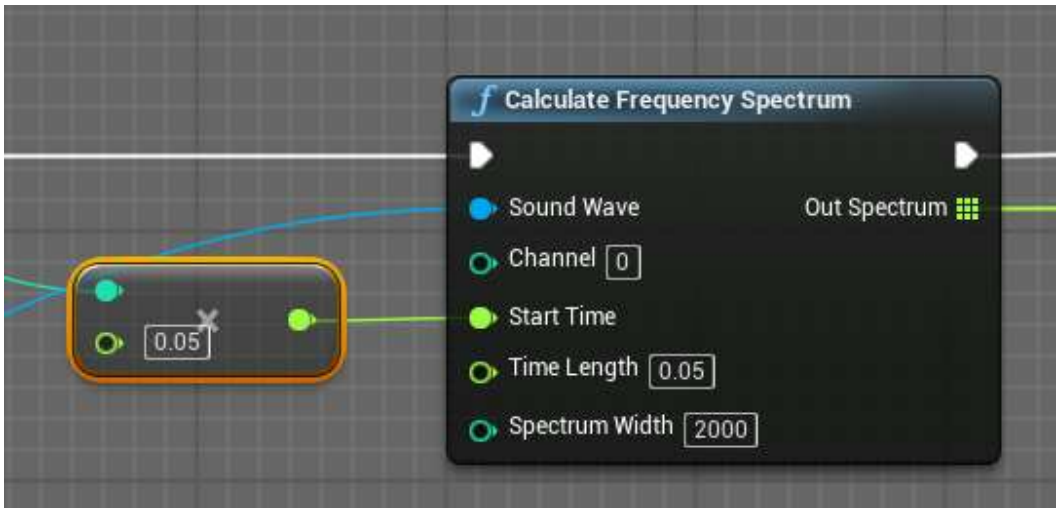
Podczas implementacji rozgrywki wystąpiło wiele problemów oraz ograniczeń powiązanych z wykorzystywaną technologią oraz platformą docelową. Niektóre z nich można bez problemu wyeliminować, jednak część z nich zaważyła negatywnie na końcowej jakości gry. Jednym z problemów ogólnych związanych z użytą technologią jest ogólna niska wydajność gry na platformach mobilnych. Silnik projektowany jest priorytetowo dla komputerów PC oraz konsol do gier, wiele rozwiązań użytych w rdzeniu silnika opiera się na optymalizacjach obecnych dla takiego sprzętu, jednak nieobecnych na smartfonach. W związku z tym także część funkcjonalności silnika jest niemożliwa do wykorzystania na Androidzie.

Specyfikacja samego systemu Android także okazała się przeszkodą lub utrudnieniem w zaimplementowaniu aplikacji. W większości przypadków są to problemy nie do ominięcia, wymagające kompromisu. Poniżej opisane są główne problemy implementacyjne i platformowe:

1. Problem wizualizacji dźwięku

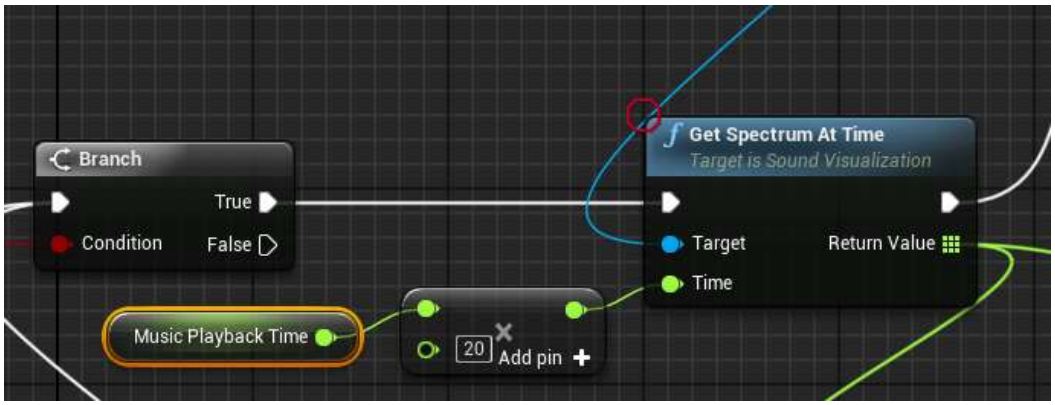
Główna natura tego problemu leży w wydajności analizowania spektrum dźwięku w czasie rzeczywistym. Warunkiem otrzymania takiego spektrum w postaci możliwej do przeanalizowania jest odpowiedni, nieskompresowany format dźwięku. Niekompresowane pliki dźwiękowe zajmują wielkie ilości pamięci, podczas gdy konwersja pliku w czasie

rzeczywistym jest skrajnie niewydajna. Na fragmencie grafu (Rysunek 3.8) przedstawiona jest funkcja dokonująca analizy dźwięku w nieskompresowanym formacie.



Rysunek 3.8 - Funkcja "Calculate Frequency Spectrum", skrajnie niewydajna w czasie rzeczywistym.

Rozwiązaniem tego problemu okazało się przetworzenie całego pliku (za pomocą tej samej funkcji, co wcześniej) jeszcze na etapie produkcji i dostarczenie jedynie gotowych danych wizualizacyjnych do końcowej wersji gry. Dzięki temu wizualizacja dźwięku ma jedynie marginalny wpływ na wydajność, jako że wizualizacja skupia się jedynie do odczytywania danych z pliku, jak widać na grafie (Rysunek 3.9).



Rysunek 3.9 – Funkcja "Get Spectrum At Time" odczytująca dane wizualizacyjne z pliku

2. Ograniczona liczba kanałów dźwiękowych

Przyczyna tego problemu związana jest ze specyfikacją platformy docelowej – systemu Android. Mnogość dźwięków odgrywana naraz może powodować utraty wydajności oraz jakości dźwięku, mogą się one zacinać i opóźniać, co rozsynchronizowuje rozgrywkę z rytmem.

Sposobem na wyeliminowanie tego problemu jest dynamiczne odcinanie dźwięków najmniej znaczących, aby liczba dźwięków odtwarzanych naraz była możliwie jak najmniejsza. Na poniższym rysunku (Rysunek 3.10) widać ustawienia mechanizmu dynamicznego odcinania dźwięków, ustawiony aktualnie na 4 dźwięki oraz odcinanie najpierw dźwięków najdalszych, a w następnej kolejności dźwięków najstarszych.

Rysunek 3.10 - Dynamiczne odcinanie dźwięków

3. Niekonsekwentna latencja dźwięku

Jest to kolejny problem związany z samym Androidem oraz smartfonami. Zależnie od wersji systemu czy też producenta smartfonu pojawia się znaczna latencja odtwarzanego dźwięku, a w szczególności muzyki. Ponadto, muzyka może być odtwarzana odrobinę wolniej, a takie spowolnienie rozsynchronizowuje rozgrywkę z rytmem.

Rozwiązanie tego problemu jest bardzo skomplikowane i nie istnieje takie, które wyeliminowałoby ten problem całkowicie dla wszystkich platform. Jednym z środków mogących załagodzić ten problem jest dokonywanie kalibracji przed rozpoczęciem rozgrywki, innym sposobem może być ingerencja w niskopoziomowy kod silnika.

4. Obracanie obiektów względem osi OY

Problem obrotu względem jednej z osi jest w produkcji gier wideo bardzo znany. Jest on związany ze sposobem reprezentacji rotacji obiektów w przestrzeni. Dla popularnego zrozumienia obrotu zastosowany jest format kątów RPY, czyli tak zwane „roll, pitch, yaw", wyrażane w stopniach. Niestety, reprezentacja w takim formacie związana jest ze zjawiskiem o nazwie „Gimbal Lock" – polega ono na tym, że układ obrotów ograniczony jest jedynie do dwóch wymiarów swobody, czyniąc obrót o 360 stopni względem osi OY praktycznie niemożliwy, jako że kąt względem osi OY podawany jest w zakresie [-90, 90] stopni.

Rozwiązanie tego problemu wymaga przejście na inny sposób reprezentacji obrotów – przejście na kwaterniony. Wykorzystywana technologia pozwala na zamianę w prosty sposób jednego formatu reprezentacji kątów na drugi, aczkolwiek tylko bezpośrednio w kodzie C++. Zaimplementowana została zatem funkcja, pozwalająca na rotację względem osi OY i odpowiedniej prędkości za pomocą kwaternionów (Rysunek 3.11), będąca bezpośrednio gotowa do wykorzystania później jako funkcja w systemie „Blueprints". Operowanie na kwaternionach pozwoliło zlikwidować tak zwany „Gimbal Lock" i obracać obiekty względem osi OY dowolnie w czasie działania gry.

```
FRotator UQuatBlueprintFunctions::rotateBySpeed(FRotator
    actualRotation, float speed, float deltaSeconds)
{
    FQuat actualQuat(actualRotation);
    FQuat rotation(FVector(0, 1, 0), -speed*deltaSeconds*PI);
    return (rotation * actualQuat).Rotator();
}
```

Rysunek 3.11 - Implementacja funkcji rotateBySpeed pozwalającej na obrót względem osi OY za pomocą kwaternionów

5. **Niska jakość shaderów na platformach mobilnych wywołuje artefakty graficzne** (Rysunek 3.12)

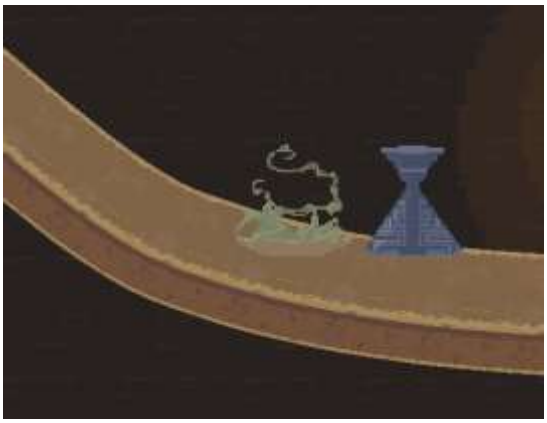
Ten problem związany jest z dwoma aspektami: niską precyzją obliczeń oraz ograniczeniem technicznym ilości i jakość samplerów, używanych do odczytywania grafik gry. Są to niskopoziomowe zagadnienia, niemożliwe do rozwiązania ze względu na ograniczenia smartfonów. Programowane shadery muszą zatem być projektowane specjalnie pod platformy mobilne, mając na uwadze ograniczone możliwości oraz specyficzny sposób, w jaki działają, co wiąże się z istnieniem swoistych błędów graficznych, które mogą być jedynie maskowane.

Poniżej na rysunku (Rysunek 3.12) widoczny jest jeden z artefaktów graficznych – nie jest to jednak znaczącym problemem, jako że gra na smartfonach wyświetlana jest w niskiej rozdzielczości, a artefakty te są zaledwie drobnymi elementami tła, co czyni je niezauważalnymi dla przeciętnego użytkownika.

Rysunek 3.12 - Artefakty graficzne – poucinane i niedokładnie wyrysowane piksele różnych wielkości

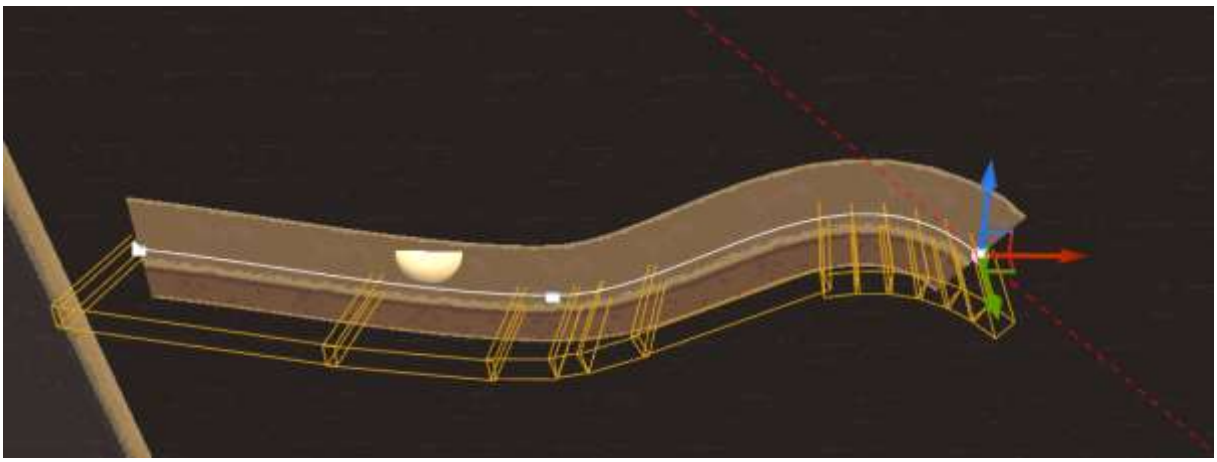
6. Implementacja krzywizny terenu

Obiekty wymagające krzywizn nie pojawiają się w grach często – implementacja obiektów wyginających się względem krzywej zawsze była problemem zarówno w kwestii wydajności jak i samego sposobu implementacji. Krzywy teren w świecie dwuwymiarowym wymaga zatem rozpatrzenia dwóch składowych – warstwy wizualnej, która musi ukazywać krzywiznę w sposób naturalny, oraz warstwy fizycznej, używanej do obliczeń fizycznych przy kolizji z postacią gracza i obiektami. Krzywy teren (Rysunek 3.13) jest bardzo atrakcyjny dla gracza, jako że wprowadza on dodatkową możliwość parametryzacji terenu – dzięki temu teren pustynny można będzie scharakteryzować za pomocą delikatnych krzywizn, a w innych miejscach stosować jedynie płaskie podłoże, z wariacjami w postaci ramp. Zmieni to zarówno rozgrywkę jak i odczucia gracza.



Rysunek 3.13 - Krzywizna terenu – wygląd w grze

Problem ten został rozwiązany poprzez implementację narzędzia-skryptu w systemie „Blueprints", pozwalającego na generację tego typu terenu po krzywej Beziera. Jako warstwa wizualna posłużył specjalny rodzaj obiektu w Unreal Engine 4, pozwalający na wygięcie obiektu względem parametrów krzywej. Obiekty te ustawiane są w równych odstępach po krzywej z odpowiednimi parametrami krzywizny. Warstwa fizyczna natomiast, z uwagi na ograniczenia obliczeń fizycznych, nie może być wykrzywiana, dlatego krzywizna przybliżana jest za pomocą zwykłych prostopadłościanów („boxów") fizycznych – im mocniejsza krzywizna, tym większa częstotliwość „boxów", co zobrazowane jest na rysunku poniżej (Rysunek 3.14).



Rysunek 3.14 - Krzywizna terenu - widok edytorski z reprezentacją fizyczną oraz styczną krzywej

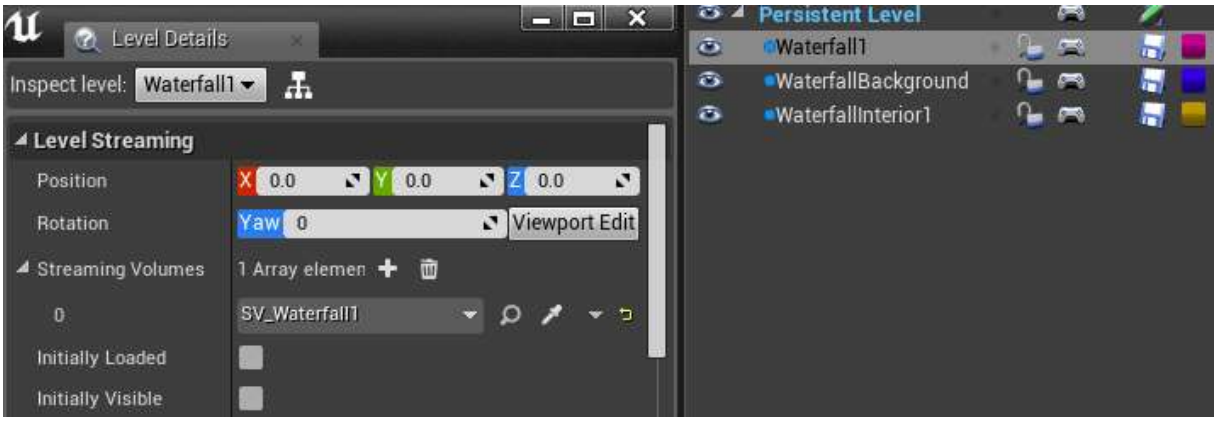
Taka implementacja krzywych terenów pozwala na wytworzenie szerokiej gamy dowolnie wykrzywionych terenów ręcznie, lub na podstawie szumów i algorytmów. Jest to rodzaj narzędzia, zatem obliczenia generujące taki teren wykonywane są podczas procesu powstawania gry, zaś do finalnego produktu trafią jedynie wygenerowane, statyczne, gotowe do użycia obiekty, niewymagające żadnych obliczeń na krzywych, a jedynie będące zestawem grafik i obiektów fizycznych.

7. Dynamiczne ładowanie poziomów

Jedną z rzeczy, która może negatywnie wpłynąć na odbiór gry przez gracza jest zbędny czas poświęcony czekaniu na załadowanie się poziomów gry. W branży pojawiły się odpowiednie rozwiązania, mające na celu albo zmniejszyć efekt tego problemu, albo

wyeliminować go całkowicie. Jednym z powszechnie stosowanych rozwiązań jest najzwyklejsze ubogacanie ekranów ładowania w proste formy rozrywki lub ekrany informacji, mające na celu zająć gracza w czasie oczekiwania na załadowanie poziomu. Czasami są to zwykłe informacje, przekazywane w postaci napisów w rogu ekranu, czasami forma przekazu treści jest zdecydowanie bardziej kreatywna – wykorzystywane są wymyślne animacje i rysunki, treść recytowana jest przez narratora jak krótki wycinek opowieści, a całość wykonana jest z najwyższą dbałością, aby nie odrywać gracza od ogólnej jakości rozgrywki. Mimo wszystko jednak sztuczny podział świata gry na odrębne lokacje, wymagające wczytania od nowa w ten sposób nowych danych przy przechodzeniu z jednego do drugiego, są pewną niedogodnością w rozgrywce.

Wykorzystana technologia – Unreal Engine 4 – pozwala na całkowicie ominięcie sztucznego podziału lokacji poprzez system umożliwiający strumieniowanie poziomów w czasie rzeczywistym. Poziomy mogą zatem być strumieniowane i ładowane w tle wtedy, gdy postać gracza się do nich zbliża, i natychmiast wyładowywane z pamięci, gdy tylko znacznie się od nich oddali. Jest to system wbudowany – wymaga jedynie sumiennego podziału świata na poziomy (Rysunek 3.15), mogące być oddzielnie ładowane i rozładowywane z pamięci, a następnie obsługi strumieniowania ich za pomocą metod wbudowanych (wyznaczenie obszarów, gdzie poziom ma być aktywny, a gdzie może być wyładowany z pamięci) lub bezpośrednio za pomocą kodu, czy też w systemie „Blueprints”.



Rysunek 3.15 - Podział poziomów na podpoziomy (prawa strona) oraz ustawienia poziomu (lewa strona)

8. Zastosowanie filtru dolnoprzepustowego dla tłumienia muzyki

Oczywistym zjawiskiem jest, iż przebywanie na zewnątrz szczelnie zamkniętego pomieszczenia (pokoju, samochodu) w którym gra głośna muzyka skutkuje słyszeniem przytłumionej, zniekształconej jej wersji, obejmującej głównie jej basową część. Zastosowanie tego zjawiska ma także miejsce w grach – ma to dźwiękowo i podświadomie zasygnalizować graczowi, iż znajduje się w środku lub na zewnątrz budynku/pomieszczenia, co może mieć znaczenie w projektowanej grze – zwłaszcza w grze muzycznej, gdzie to od muzyki i rytmu zależy całe środowisko.

Do takiego przekształcenia dźwięku wykorzystywany jest filtr dolnoprzepustowy, zostawiający jedynie basową część dźwięków. Jest on dostępny do użytku w silniku Unreal Engine 4, jednak nie funkcjonuje on na platformie Android ze względu na jej ograniczenia. Problem ten został rozwiązany poprzez zastosowanie tego filtru na etapie produkcji gry na wszystkich wymagających tego plikach muzycznych, a następnie wstawienie do gry statycznych, przetworzonych już utworów. Ostatnim krokiem jest zastosowanie przeniku pomiędzy utworami w czasie rzeczywistym podczas przechodzenia do lub z odpowiednich pomieszczeń, w których gra muzyka.

4. Podsumowanie

4.1. Możliwości rozwoju

Gatunek będący hybrydą dwóch innych gatunków – gier rytmicznych i platformowych – pozwala na podjęcie wielu ścieżek rozwoju całej gry. Do hybrydy dodawać można kolejne elementy znane z innych rodzajów gier, między innymi typu RPG albo horroru. Pozwala to na zaprezentowanie szerszego spektrum emocji i zawartości graczowi jednocześnie będąc po części zawartością opcjonalną, dzięki czemu użytkownik będzie mógł doświadczyć jedynie tego, na co ma ochotę. Bazując na tym wymienić można główne ścieżki rozwoju dostępne dla gry:

- **Zbieranie przedmiotów** – gracze lubią zbierać przedmioty, ulepszać swój ekwipunek. Chęć posiadania wyspecjalizowanych narzędzi i korzystania z nich uwarunkowana jest potrzebą „odgrywania roli". Włożenie do świata gry elementów rynsztunku oraz przedmiotów specjalnych jest naturalnym mechanizmem rozwoju postaci oraz przedstawianej opowieści. Do używania takich przedmiotów zastosowany może być też nowy sposób kontrolowania postaci – na przykład użycie akcelerometru. Użycie takich przedmiotów mogłoby pozwalać na docieranie do niedostępnych wcześniej miejsc lub wpływanie na losy świata i postaci oraz opowiadaną historię.

- **Przeciwnicy specjalni** – w świecie gier znani jako „bossowie" są kamieniami milowymi w rozwoju historii oraz postaci gracza. Każdy przeciwnik specjalny charakteryzować się może własnym mechanizmem, sprawdzającym umiejętności gracza lub jego dedykacje do podjętych przez siebie decyzji, w związku z czym pokonanie go będzie równoznaczne ze zdaniem sprawdzianu przedstawionego graczowi w sposób naturalny i czasami spontaniczny. Dodanie takich przeciwników do rozgrywki nada graczowi wrażenie postępu – zdobywanie kamieni milowych, rozpoczynanie nowych rozdziałów.

- **Nowe krainy** – rozwinięcie świata gry poprzez dodanie nowych krain mających własny charakter i mechaniki zadziała pozytywnie na ciekawość gracza i pozwoli wywołać u niego nowe emocje. Dodanie do gry nowych krain podzieli także świat gry na strefy, które różnić się mogą poziomem trudności, charakterem prezentowanych zagadek i wyzwań oraz pochodzeniem utworów muzycznych, mających inne właściwości rytmiczne.

- **Algorytmy analizujące utwory muzyczne** – zastosowanie takich algorytmów może dać możliwość graczom do używania własnych utworów muzycznych do sekcji gry specjalnie do tego przygotowanych. Dzięki temu użytkownicy sami decydować będą o charakterze spotykanej muzyki. Może być to jednak trudne do zastosowania w czasie rzeczywistym, zwłaszcza na platformach mobilnych takich jak system Android.

- **Postaci poboczne** – dodanie większej ilości postaci w grze może bardzo pozytywnie wpłynąć na zaangażowanie emocjonalne gracza. Mogą one prezentować wybory moralne, zadania i wyzwania dla gracza oraz przekazywać mu wiedzę na temat mechanik rozgrywki, sekretów świata i jego historii lub też oszukiwać go. Reakcja gracza w odpowiednich sytuacjach socjalnych pozwoli mu poznać samego siebie i swoją naturę – gry posiadające tego typu cechy są bardzo popularne na serwisach typu YouTube i Twitch.tv, gdyż gracze lubią oglądać reakcję ich ulubionych prezenterów na różne sytuacje socjalne.

- **Rozbudowanie historii** – nadanie sensu całej sytuacji, prezentowanej w grze, w postaci powieści opowiadanej wizualnie i poprzez interakcje gry z graczem jest niezbędne, aby gracz mógł zostać zaangażowany emocjonalnie. Jest to bezpośrednio połączone z dodaniem do gry postaci pobocznych oraz nowych krain – każda z tych postaci to odpowiedni zestaw funkcji i interakcji, ale także każda jest wycinkiem całej historii, którą gra opowiada.

- **Interakcje poboczne** – dodanie szerokiego spektrum interakcji ze światem, nie mających wpływu na rozgrywkę a jedynie pozwalających graczowi na zabawę otoczeniem może mieć jak najbardziej pozytywny wpływ na immersję gracza w świat gry.

- **Inne platformy docelowe –** zastosowana technologia – Unreal Engine 4 – pozwala na zbudowanie gry na różne inne platformy niż system Android. Dostępne platformy to między innymi: system iOS, systemy Windows, Linux oraz komputery Mac, przeglądarki internetowe za pomocą HTML5, konsole Xbox One, PlayStation 4 oraz Nintendo Switch. Wymaga to jedynie dostosowania sposobu kontrolowania postaci i interakcji ze światem, jako że nie wszystkie urządzenia posiadałyby ekran dotykowy.

4.2. Podsumowanie właściwe

Podsumowując, projekt oraz implementacja zostały wykonane na potrzeby pracy zgodnie z założeniami. Powstała gra (Rysunek 4.1) jest hybrydą gry rytmicznej i platformowej, a gdy nawiązany zostanie odpowiedni dyskurs, gra przekazuje emocje spełniające potrzeby graczy. Implementacja projektu została wykonana w wersji alfa, zawierającej jedynie prezentację poszczególnych zaprojektowanych elementów działających razem jako gra oraz zagajenie do świata gry, natomiast możliwości rozwoju gwarantują rozrost samego projektu jak i implementacji do wersji będącej kandydatem do osiągnięcia sukcesu komercyjnego. Całość projektu, włączając w to wszystkie grafiki, utwory muzyczne, skrypty i shadery wykonana została od zera specjalnie dla tego projektu, wyłączając jedynie niektóre efekty dźwiękowe.

Rysunek 4.1 - Finalny wygląd zaimplementowanej gry

Powodem podjęcia się pracy o tej tematyce była przede wszystkim chęć wyrażenia siebie – gra, oprócz bycia aplikacją multimedialną o zdefiniowanym celu, jest także dziełem artystycznym. Oprócz zaprogramowania mechanik i sekwencji gry powstać musiały także grafiki i animacje poklatkowe, a także skomponowana została specjalna muzyka o odpowiednich walorach rytmicznych. Także sam kod, będący nie tylko sercem całej aplikacji, służył wyrazowi artystycznemu w postaci shaderów i symulacji. Całość tego wyrazu nie byłaby możliwa gdyby użyta technologia na to nie pozwalała – wybrany do realizacji pracy silnik projektowany był z myślą zarówno o części programistycznej gier, ale także o integracji tego z poszukiwanym wyrazem artystycznym.

Wszystkie założenia dotyczące stylu graficznego oraz interfejsu użytkownika zostały w pracy spełnione. Mechanika gry została zaprojektowana z szerokimi możliwościami rozwoju i zaimplementowana w całości poza jednym elementem – projektem poziomów. Jest on zarówno częścią mechaniki, ale także głównym sposobem opowiadania historii i realizowania scenariusza gry, jednocześnie jest on najłatwiej rozszerzalną i modyfikowalną jej częścią, dlatego w wersji alfa ograniczony został jedynie do liniowej serii poziomów prezentującej założenia projektu poziomów jedynie w obrębie ich (między innymi zawiera sekretne pomieszczenie, kilka sposobów przejścia, stara się uczyć gracza – choć jest to trudne ze względu na mały rozmiar poziomu). Niestety w czasie implementacji pojawiały się także problemy – część z nich udało się rozwiązać, natomiast inne problemy powiązane są z samą specyfikacją systemu Android i próba naprawy tych błędów może wymagać ingerencję w niskopoziomowe sekcje silnika, użytego w produkcji gry. Także biorąc pod uwagę brak dostępności szerokiej gamy sprzętu do testowania (gra sprawdzana była głównie na smartfonie Huawei P8 Lite, a także kilku innych, lecz w znacznie mniejszym stopniu) gra może posiadać błędy, które pojawiają się tylko na innych platformach z systemem Android (jak tablety albo smartfony innych firm).

Podjęcie się tej pracy umożliwiło także pokazanie różnic w podejściu do projektowania gier względem projektowania aplikacji użytkowych:

- **Wymagania funkcjonalne i niefunkcjonalne** - w tradycyjnym podejściu do inżynierii oprogramowania wymagania formułowane są w sposób ścisły – aplikacja powinna mieć odpowiednie funkcje i przeprowadzać sekwencje w zależności od wkładu użytkownika. Tymczasem w produkcji gier ważne jest jedynie to, aby gra sprawiała radość – a ta może zostać osiągnięta spełniając potrzeby emocjonalne użytkownika.

- **Przypadki użycia** – projektowanie gry wymaga ich użycia w ich oryginalnej postaci głównie podczas projektowania serwisów naokoło gry – tak jak menu główne, system dołączania do gry wieloosobowej, panel administracyjny. Sam rdzeń gry natomiast skupia się do podejmowania interakcji z elementami świata i sterowania postacią – projekt skupia się więc głównie na tych elementach, a one same mogą nie wywoływać sekwencji zdarzeń. Przypadki użycia i historyjki użytkownika zostały zatem zamknięte w mechaniki gry i możliwości interakcji gracza ze światem.

- **Styl wizualny i projekt poziomów** – elementy te w aplikacjach multimedialnych, jakimi są gry, mają szczególne znaczenie i wymagają projektowania od samego początku – mają bezpośredni wpływ na odbiór emocjonalny i przekaz gry.

Gra zaprojektowana została z myślą o innowacji – choć gry muzyczno-platformowe istnieją już na rynku i odnoszą sukcesy, czego przykładem jest sukces gry „140” (rozdział 2.1), ich nasycenie jest bardzo małe, a także założenia projektowania są inne – w przypadku gry

„140” nacisk położony jest na minimalizm i prostotę rozgrywki. Względem takich gier dodane zostały odpowiednie innowacje:

- **Elementy RPG** – przeprowadzanie dialogów, wybór moralny albo wybór ścieżki gry, odkrywanie sekretów i świata – zapewnia graczowi immersję i więź emocjonalną z postaciami i grą;

- **Interakcja poprzez dotyk** – gracz podejmuje bezpośrednią interakcję ze światem poprzez dotykanie (naciskanie) odpowiednich elementów środowiska na ekranie (oznaczonych odpowiednio za pomocą interfejsu użytkownika);

- **Styl wizualny** – odpowiedni projekt postaci oraz krain ma za zadanie narzucić klimat i wywołać u gracza emocje, których nie doświadczy w innych grach;

- **Możliwości rozwoju –** planowany rozwój aplikacji zakłada wprowadzenie do gry kolejnych innowacji w postaci przeciwników specjalnych, rozwinięcia warstwy historycznej gry, dodania kolejnych, unikalnych krain, algorytmów i interakcji.

Biorąc pod uwagę wielkość i stan branży gier można stwierdzić, iż zaprojektowanie i implementacja gry mobilnej daje możliwość na znaczny sukces komercyjny i tym samym zaistnienie w świecie biznesu. Rozwinięcie gry na większą ilość platform może jej jedynie pomóc, dalej popularyzując grę i docierając do nowych odbiorców. Gra muzyczna na system Android może zatem odnieść sukces jako aplikacja na rynku, ale może też przekonać do siebie nowych odbiorców, którzy dopiero stawiają pierwsze kroki w świecie gier, poprzez nawiązanie odpowiedniej więzi emocjonalnej, a także wymusić refleksje na jej temat, tym samym spełniając się jako dzieło artystyczne.

Bibliografia

[1] Adams E., *Fundamentals of Game Design*, New Riders, 2010

[2] Caillois R., *Man, Play and Games*, University of Illinois Press, 1961

[3] Doran J. P., *Unreal Engine Game Development Cookbook*, Packt Publishing Ltd., 2015

[4] Dunniway T., *Mobile Game UI/UX Top 10 Best Practices*, 2016 https://www.linkedin.com/pulse/mobile-game-uiux-top-10-best-practices-troy-dunniway (29.10.2017)

[5] Epic Games Inc., *Unreal Engine 4 Documentation*, https://docs.unrealengine.com (20.11.2017)

[6] Krakowski Park Technologiczny, *Kondycja Polskiej Branży Gier'17*, 2017 http://www.kpt.krakow.pl/badania-kondycja-polskiej-branzy-gier17/ (20.11.2017)

[7] Lee J., *Learn Unreal Engine Game Development*, Packt Publishing Ltd., 2016

[8] Michael J., *Why You Should Choose the Unreal Engine as an Indie*, 2016 https://www.gameskinny.com/r5bgx/why-you-should-choose-the-unreal-engine-as-an-indie (4.11.2017)

[9] Misra N., *Learning Unreal Engine Android Game Development*, Packt Publishing Ltd., 2015

[10] Olivares R. C., *Unreal Engine 4, Rendering and Mobile Platforms*, 2014 https://www.unrealengine.com/en-US/resources *(28.10.2017)*

[11] Pew Research Center, *Mobile Fact Sheet*, 2017 http://www.pewinternet.org/fact-sheet/mobile/ (28.10.2017)

[12] Podgorski D., *Style by Necessity: On FTL: Faster Than Light, and Pixel Art as an Art Movement, 2015* http://thegemsbok.com/art-reviews-and-articles/video-game-reviews-mid-week-mission-ftl-faster-than-light-subset-games/ (29.10.2017)

[13] Quintans D., *Game UI By Example: A Crash Course in the Good and the Bad*, 2013 https://gamedevelopment.tutsplus.com/tutorials/game-ui-by-example-a-crash-course-in-the-good-and-the-bad--gamedev-3943 (29.10.2017)

[14] Ruppel R., *Graphic L.A.*, Design Studio Press, Los Angeles 2014

[15] Sewell B., *Blueprints Visual Scripting for Unreal Engine*, Packt Publishing Ltd., 2015

[16] Sheppard M., Burns R., *Empowering the Player: Level Design in N++*, 2016 https://www.gdcvault.com/play/1023282/Empowering-the-Player-Level-Design (28.10.2017)

[17] Smedber N., *UE4 Mobile Performance, 2014* https://www.unrealengine.com/en-US/resources *(28.10.2017)*

[18] Webster A., *Roots of rhythm: a brief history of the music game genre*, https://arstechnica.com/gaming/2009/03/ne-music-game-feature , 2009

[19] Yasuhara H., *How to make a game „fun”*, Digital Dragons, 2017 https://www.youtube.com/watch?v=J-3avMBqJ9s (20.11.2017)

Spis ilustracji