Projekt Grafika Komputerowa

Raport beta z postępów prac.

1. Treść zadania

Temat: Gra komputerowa wzorowana na oryginalnej grze “Plants vs Zombies”.

Technologie: Unity, C#, VS Code/VS

2. Analiza zadania

Opis

Gra będzie klonem oryginalnej gry “Plants vs Zombies”, w której wykorzystamy istniejące w oryginale mechaniki rozgrywki. Gra będzie rozgrywała się na płaskiej mapie, na której po jednej stronie umieszczone zostaną nasze obiekty broniące, a po drugiej spawnowane będą potwory lub inni przeciwnicy. Całość gry polegać będzie na wykorzystaniu obiektów broniących, aby nie pozwolić przeciwnikom przedostać się na drugą stronę mapy.

Rozgrywka

Gra będzie miała możliwość rozgrywki w dwóch językach, z możliwością tłumaczeń na inne języki.

Obiekty rozmieszczone będą na mapie o postaci siatki (na kształt szachownicy), gdzie w wolnych polach możliwe będzie stawianie swoich obiektów.

Warunki zwycięstwa i porażki

Gracz przegrywa rozgrywkę w momencie gdy odpowiednia liczba przeciwników przedostanie się na jego stronę (przekroczy granicę). Wygrać rozgrywkę może w momencie przejścia wszystkich poziomów dostępnych w grze.

3. Opis postępu prac

Wykorzystane zagadnienia z grafiki komputerowej

W grze wykorzystane zostanie zagadnienie związane z efektem cząsteczek (particles system w unity), animacje oraz wykrywanie kolizji z przeciwnikami.

- Particles system - Wykorzystany został jako tło w menu głównym gry, cząsteczki rozchodzą się w bocznych kierunkach z emitera w kształcie koła.
- Animacje - W grze istnieją liczne animacje pierwszym przykładem są poruszające się postaci oraz latające pociski, każda z postaci posiada kilka stanów, w których może się znaleźć w zależności od sytuacji w grze. Stan ataku pokazuje animację ataku postaci, Stan ‘idle’ w zależności od postaci jest chodzeniem lub statycznym kołysaniem się na boki symbolizujące pozostanie w ruchu.
- Wykrywanie kolizji - W grze kolizje wykrywane są między przeciwnikami na dwa sposoby pierwszy z nich to tradycyjny sposób wykrywania uderzenie kolidera o kolider innym sposobem, który znalazł zastosowanie dla postaci, które swoje ataki wykonują z dystansu jest wypuszczenie tzw. “Raycast” wykrywającego kolidery na zadanej odległości oraz w zadanym kierunku.

Wielojęzykowość oraz serializacja

Gra umożliwia rozgrywkę w języku angielskim oraz polskim z możliwościa zmiany ustawień w każdej chwili w menu głównym gry. Istnieje bezinwazyjna możliwość dołożenia do gry kolejnych tłumaczeń na inne języki. Wszystkie teksty w grze przechowywane są jako pliki o formacie .csv (dane rozdzielone przecinkami) przez co możliwe jest ich generowanie bezpośrednio z programu np. Excel.

Poniżej przedstawiamy przykład jednego z plików reprezentujących teksty w grze oraz przypisane do nich klucze lokalizacyjne umożliwiające ich odszukanie.

KLUCZ LOKALIZACYJNY	NAZWA PL	NAZWA EN
BUTTON_ENDGAME_PLAY	Zagraj Ponownie	Play Again
BUTTON_GAME_LOAD	Załaduj Grę	Load Game
BUTTON_GAME_MAINMENU	Menu	Main Menu
BUTTON_GAME_SAVE	Zapisz Grę	Save Game
BUTTON_MM_FB	Facebook	Facebook
BUTTON_MM_LOADGAME	Załaduj Grę	Load Game
BUTTON_MM_NEWGAME	Nowa Gra	New Game
BUTTON_MM_OPTIONS	Ustawienia	Options
BUTTON_MM_QUIT	Wyjście	Quit
STRING_ENDGAME_LOSE	Przegrałeś	You Lost
STRING_ENDGAME_WIN	Wygrałeś	You Win
STRING_GAME_COINS	Monety	Coins
STRING_GAME_LIVES	Życia	Lives
STRING_GAME_WAVE	Koniec Fali	Waves Cleared

Serializacja danych w grze jest w pełni zaimplementowana oraz umożliwia zapis rozgrywki po każdej poprawnie pokonanie fali przeciwników. Zapis jest realizowany za pomocą wzorca projektowego “Memento”, każda z klas, której dotyczy zapis korzysta z odpowiedniego managera, który zajmuje się przeprowadzenie serializacje oraz deserializacji danych. Wczytanie rozgrywki jest możliwe z poziomu menu głównego o ile wcześniej został wykonany zapis stanu gry.

Porównanie postępów do planu pracy.

- Etap 1 - Utworzenie grafiki mapy oraz implementacja w grze wraz z obsługą ustawiania postaci w odpowiednich miejscach na mapie.
    - Grafika została zaimplementowana główna scena jest renderowana poprawnie oraz istnieje pełna obsługa ustawiania postaci w odpowiednich miejscach na mapie. W przypadku rozwoju gry siatka umożliwiająca ustawienie obiektów może być dostosowywana do rozmieszczenia mapy.

- Etap 2 - Dodanie systemu obsługi sklepu w trakcie rozgrywki aby umożliwić graczom zakup postaci w celu ustawieniu na odpowiednim miejscu mapy.
    - Sklep został obsłużony w pełni istnieje możliwość wyboru postaci oraz umieszczenie jej, jeżeli gracz posiada na to odpowiednie środki finansowe w wybranym polu. W sklepie zaimplementowano obsługę graficzną podświetlenia wybranej postaci oraz selekcji. Sam sklep automatycznie pobiera wszystkie dostępne postaci z gry z odpowiedniego tzw. “Scriptable Object” będącego kontenerem danych. Wyświetlenie kolejnych postaci wymaga jedynie utworzenie odpowiedniego pliku prefab oraz przypisanie do kontenera od tego momentu będzie widoczny w całej przestrzeni gry.

- Etap 3 - Dodanie systemu odpowiedzialnego za spawnowanie przeciwników oraz ich zdefiniowanie jako obiekty. Przeciwnicy pojawiać się będą z prawej strony mapy w tzw. “Falach”.
    - Spawnowanie przeciwników zostało w pełni obsłużone oraz wszystkie ich zachowanie takie jak wykrywanie kolizji oraz ataki. Przeciwnicy analogicznie jak postaci pozytywnej muszą zostać podpięte do kontenera zawierającego postaci z gry i od tego czasu będą widoczne w całej przestrzeni gry.

- Etap 4 - Wykrywanie kolizji agentów w trakcie rozgrywki oraz dodanie mechaniki ich niszczenia.
    - Pełna implementacja szczegółowy opis został już przedstawiony w punkcie powyżej.

- Etap 5 - Stworzenie UI dotyczącego ekranu przegranej oraz zwycięstwa oraz wykrywanie tych zdarzeń.
    - Pełna implementacja ekrany zostały zaimplementowane przy podejściu oraz zastosowaniu modelu MVC zostają wyświetlone w przypadku wystąpienia odpowiedniego eventu w trakcie rozgrywki. Kontrolery ekranów wywołują następnie odpowiednie metody z klas managerów.

- Etap 6 - Stworzenie systemu obsługi upgradów w grze dotyczącego możliwości zakupu przez gracza ulepszeń swoich postaci.
    - Ta mechanika nie została zaimplementowana, zamiast niej został stworzony system umożliwiający zmienne wersje językowe opisany w punkcie powyżej.

- Etap 7 - Serializacja danych niezbędnych do stworzenia systemu zapisu oraz wczytywania rozgrywki.
    - Pełna implementacja przy zastosowaniu wzorca projektowego “Memento” oraz odpowiedniej klasy managera. Takie podejście umożliwiło każdemu z członków zespołu utworzenie jednej klasy serializacji. Dokładny opis znajduje się w punkcie powyżej.

- Etap 8 - Polisz w tym wykorzystanie particles system oraz animacji postaci.
    - Dokładny opis znajduje się w punkcie powyżej. Particle system został wykorzystany oraz liczne animacje w grze.

Porównanie postępów do planu pracy.

- Fabian Berda - Część programistyczna, implementacja logiki oraz modułów w grze oraz implementacja kontenerów danych dla części designu w celu podpinania nowych elementów bez ingerencji w kod z poziomu inspektora. Zaprojektowanie architektury całości programu oraz utrzymanie jej założeń. - Lider
- Kamil Susek - Część programistyczna, implementacja logiki oraz modułów w grze oraz wsparcie implementacji części programistycznej UI oraz modułów sterowania dla designu.
- Piotr Adamski - Projektowanie UI/UX, projekt oraz odpowiednia implementacja w tym utworzenie plików prefab oraz lokazlizacja w odpowiednim miejcu na scenia całości UI. Design oraz wykonanie wybranych kontrolerów do obsługi stworzonych widoków.
- Wojciech Waleszczyk - Grafika znajdująca się w grze oraz implementacja wybranych skryptów np. stworzenie memento dla wybranego managera oraz pełna jego obsługa. Stworzenie animacji.

- Tomasz Wienchor - Grafika znajdująca się w grze oraz implementacja wybranych skryptów np. stworzenie memento dla wybranego managera oraz pełna jego obsługa. Projekt tabel w skoroszycie. Testowanie gry.