31.03.2016

Projekt PZTGK

Architektura

Temat: Kosmiczna strzelanka 2.5D typu Vertical Scroller

Skład sekcji:

Katarzyna Orzechowska
Przemysław Grela
Rafał Loska
Dariusz Fredyk
Piotr Kunicki
Michał Kiełczewski
Marcin Griszbacher
Denis Wychowałek

Projekt architektury systemu

W tym dokumencie przedstawiona zostanie architektura jaka została zaplanowana do stworzenia projektu gry. Poniżej przedstawione zostały klasy użyte w projekcje wraz z ich opisem oraz powiązaniami.

1. Klasy odpowiadające za Menu oraz tabelę wyników

Bedzie to pierwszy zestaw klas wykorzystany po uruchomieniu gry. Klasy odpowiedzialne będą za zarządzanie profilem gracza, wgląd do jego tabeli wyników oraz działanie menu gry. Skrypty w tej kategorii:

Menu - odpowiadającą za wyświetlanie interfejsu menu i reakcje na kliknięcia przycisków. Klasa wykorzystująca skrypt "Profile" do zarządzania profilem gracza

Profile - będzie wczytywać/zapisywać z zewnętrznych plików dane graczy

Skrypty powiązane będą w następujący sposób: Menu na start wczytuje listę profili. Przewidziana będzie możliwość tworzenia i usuwania profili oraz wybranie jednego z nich i przejście do menu właściwego gry. Po wczytaniu lub stworzeniu profilu utworzy obiekt klasy Profile reprezentujący wybranego gracza.

Jeśli gracz zechce uzyskać informacje na temat najwyższego wyniku w danej planszy, skrypt "Menu" będzie wykorzystywał obiekt profilu gracza do pobrania informacji. I jeśli któraś z danych gracza zostanie zmieniona podczas rozgrywki lub w inny sposób, informacja ta zostanie dostarczona do klasy "Menu", która wyśle te informacje do profilu gracza w celu jego zaktualizowania. Menu gry powiązane będzie również z innymi scenami zawierającymi poziomy gry. Będzie w stanie przejść do danej sceny i pobierać z niej informacje o statusie danych gracza.

2. Klasy odpowiadające AI przeciwników

Grupa klas odpowiedzialnych za zachowanie się przeciwników podczas rozgrywki. W skryptach tego rodzaju można ustalać ruch poszczególnych jednostek wroga jak i całej fali.

CAITask - klasa bazowa, która sama w sobie zawiera jedynie wirtualne uniwersalne pola. Najważniejsza jest tu metoda o roboczej nazwie DoStuff(void), która będzie musiała być przeciążona w każdej klasie dziedziczącej. Wywołanie każdej dziedziczącej klasy wygląda identycznie, różnią się tylko sposobem działania i możliwymi ustawieniami zniennych

Klasy dziedziczące po CAITask:

CStraightMove - ruch przeciwnika po linii prostej
CCircularMove - ruch przeciwnika po łuku okręgu
CTrailPlayer - podążanie przeciwnika za graczem
CWait, **CStop**, **CPause** - zatrzymanie i postój przeciwnika

Wróg - posiada tablicę CAITask[], w której znajdują się poszczególne zachowania jakie ma realizować. Drugie pole to typ zachowania po zakończeniu tablicy. Przewidziano opcje destroy, repeat albo random

Wave - klasa określająca liczbę wrogów w fali, odstęp między falami, moment w którym rozpoczyna się pojawianie nowych przeciwników, zdefiniowane AI dla każdego wroga w konkretnej fali

Dodatkową modyfikacją powyższych skryptów jaka może zostać dodana jest kierowanie pocisków wroga w stronę gracza.

3. Klasy odpowiadające za działanie systemu ulepszeń i broni statku

Podczas rozgrywki pojawiają się obiekty będące przeciwnikami oraz ulepszeniami statku. Każdy z obiektów posiada jeden skrypt. Obiekty rozpoznają się po posiadanych skryptach. Klasy w skryptach nie dziedziczą po niczym i posiadają wiele parametrów do ustawienia w inspektorze dla późniejszego zbalansowania rozgrywki przez level designera.

BlackHoleScript - Skrypt odpowiedzialny za pojawienie się przeciwnika czarna dziura

BuildingScript - Skrypt odpowiedzialny za budynki. Zawiera parametry o ilości życia, i o typie i prawdopodobieństwie paczki

BulletScript - Skrypt odpowiedzialny za mechanikę pocisków. Można ustawić różne właściwości pocisków takie jak jego prędkość, obrażenia, czas propagacji, podstawowy numer broni, kasowanie po uderzeniu (pocisk zadający obrażenia wszystkim wrogom w danej linii)

CameraScript - Skrypt odpowiedzialny za poruszanie się kamery

CleanerScript - Skrypt usuwający przedmioty za plecami gracza wychodzące poza mapę

EnemyScript - Skrypt odpowiedzialny za wrogów, zawiera parametry przeciwnika i informacje o typie i prawdopodobieństwie paczki

GranadeScript - Skrypt odpowiedzialny za granat, który niszczy przeciwników

MeteroidScript - Skrypt odpowiedzialny za pojawienie się przeciwnika meteor

NamePickupScript - Skrypt paczki z zaopatrzeniem które nie ma odpowiednika w postaci prefabrykatu. Należy wybrać ilość oraz wartość enum np. punkty życia, maksymalne punkty życia, prędkość ataku

ObjectPickupScript - Skrypt paczki z zaopatrzeniem który ma odpowiednik w postaci prefabrykatu. Należy podać obiekt (Np. amunicja, tarcza) i jego ilość

PlayerScript - Skrypt zawierający mechanikę i parametry gracza, w tym też informację o prędkości planszy, ograniczeniach swobody ruchu gracza i ilości punktów

ShieldScript - Skrypt odpowiedzialny za pojawienie się ulepszenia bariera ochronna. Po zebraniu tworzy pole siłowe dookoła gracza

WeaponPickupScript - Skrypt odpowiedzialny za paczki ulepszające lub dodające nowe bronie. Jako parametry trzeba podać pociski w kolejności ulepszania

4. Diagramy klas UML

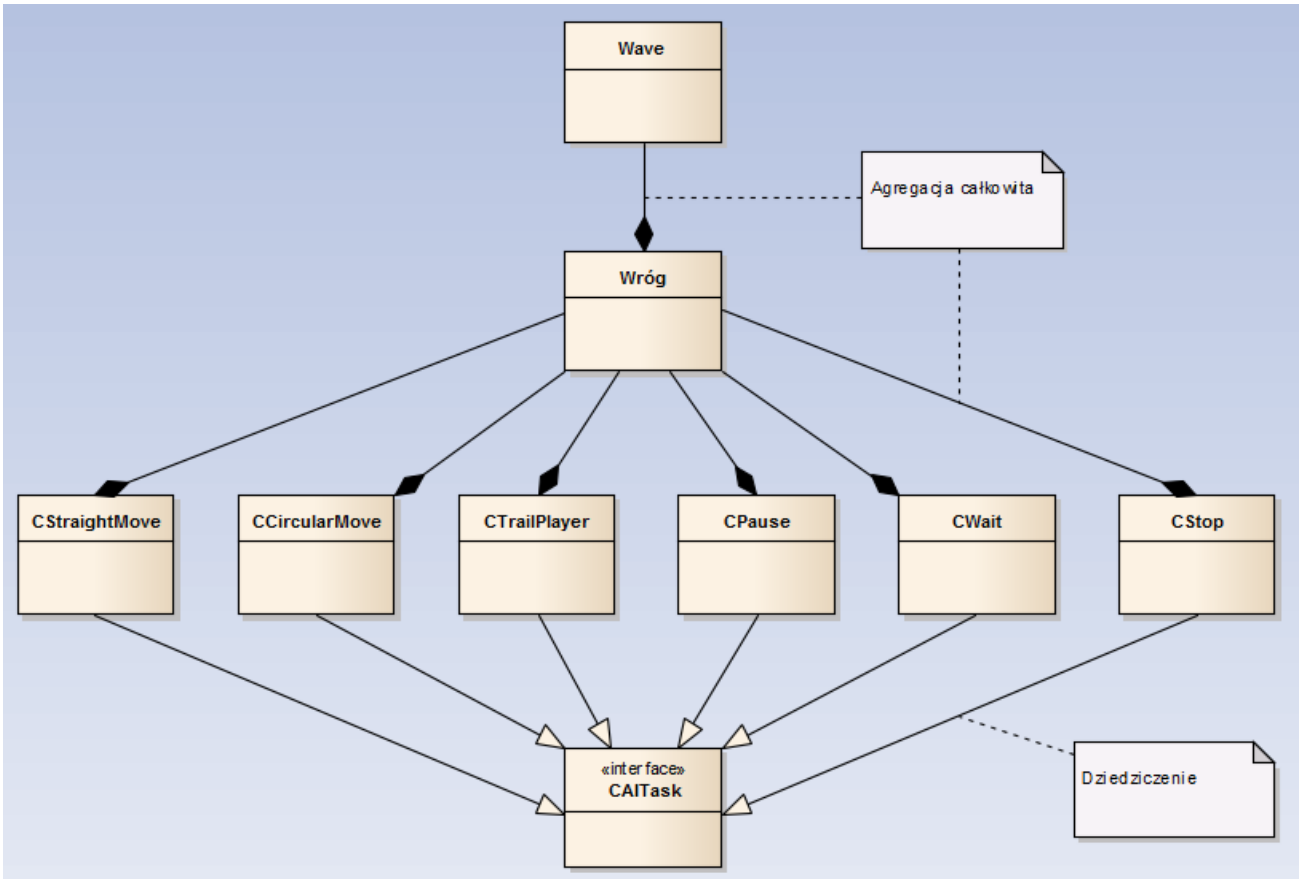

Diagram klas odpowiedzialnych za sztuczną inteligencję

Na powyższym diagramie widzimy klasę "CAITask" będącą interfejsem odpowiedzialnym za ruch przeciwników. Dziedziczą z niej wszystkie klasy o specyficznych

właściwościach ruchu jednostek wroga. Następnie mamy klasę "Wróg", która ma możliwość agregowania obiektów klas dziedziczących z "CAITask". Klasa "Wróg" posiada w swojej strukturze listę takich obiektów, z których korzysta aby zdeterminować po jakiej ścieżce będzie się poruszał jej obiekt na ekranie. Klasa "Wave" służy do wstawiania zdefiniowanej dla danego poziomu liczby i rodzaju obiektów klasy "Wróg", od której jest zależna. Zależność ta informuje o tym, iż aby klasa "Wave" mogła używać obiektów klasy "Wróg", musi mieć o niej informacje.

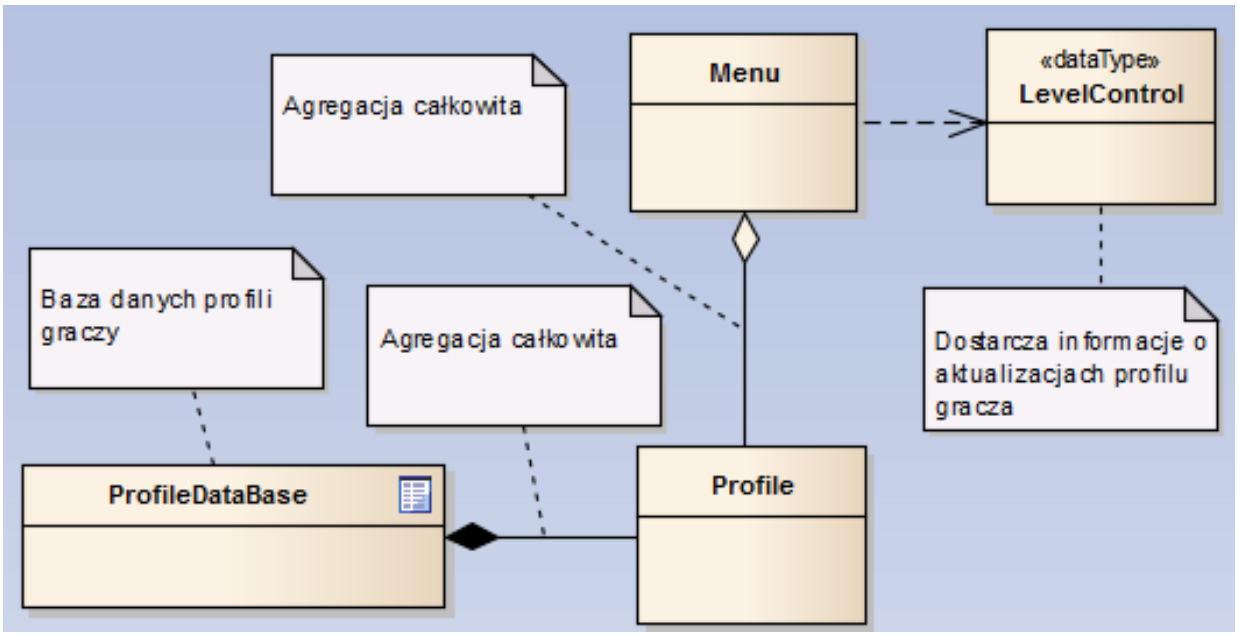

Diagram klas odpowiedzialnych za Menu

Powyżej znajduje się zestaw klas sterujących menu gry. Jak widzimy, klasa "Profile" wczytuje z zewnętrznego źródła dane dotyczące zapisanych profili graczy wraz z ich tabelą rekordów. Klasa "Menu" agreguje częściowo klasę "Profile" co daje jej możliwość odczytania nazw graczy potrzebnych do wyświetlenia w menu wyboru gracza. Za pomocą tej relacji można również modyfikować te informacje. "Menu" sczytuje również rekordy punktowe graczy dla poszczególnych poziomów i wyświetla je jako tabele wyników. Podczas rozgrywki informacje o rekordach punktowych gracza ulegają zmianie. Klasa "Menu" ma wgląd w aktualną sytuację rozgrywki gdyż działa zależnie od informacji napływających z mechanizmu kontrolującego scenę, w której umieszczona jest dana rozgrywka. Dane przechwytywane są przez klasę "Menu", a następnie wydane jest polecenie dla klasy "Profile" aby odświeżyć informacje dotyczące obecnego gracza.