Amazons – Unity, C#

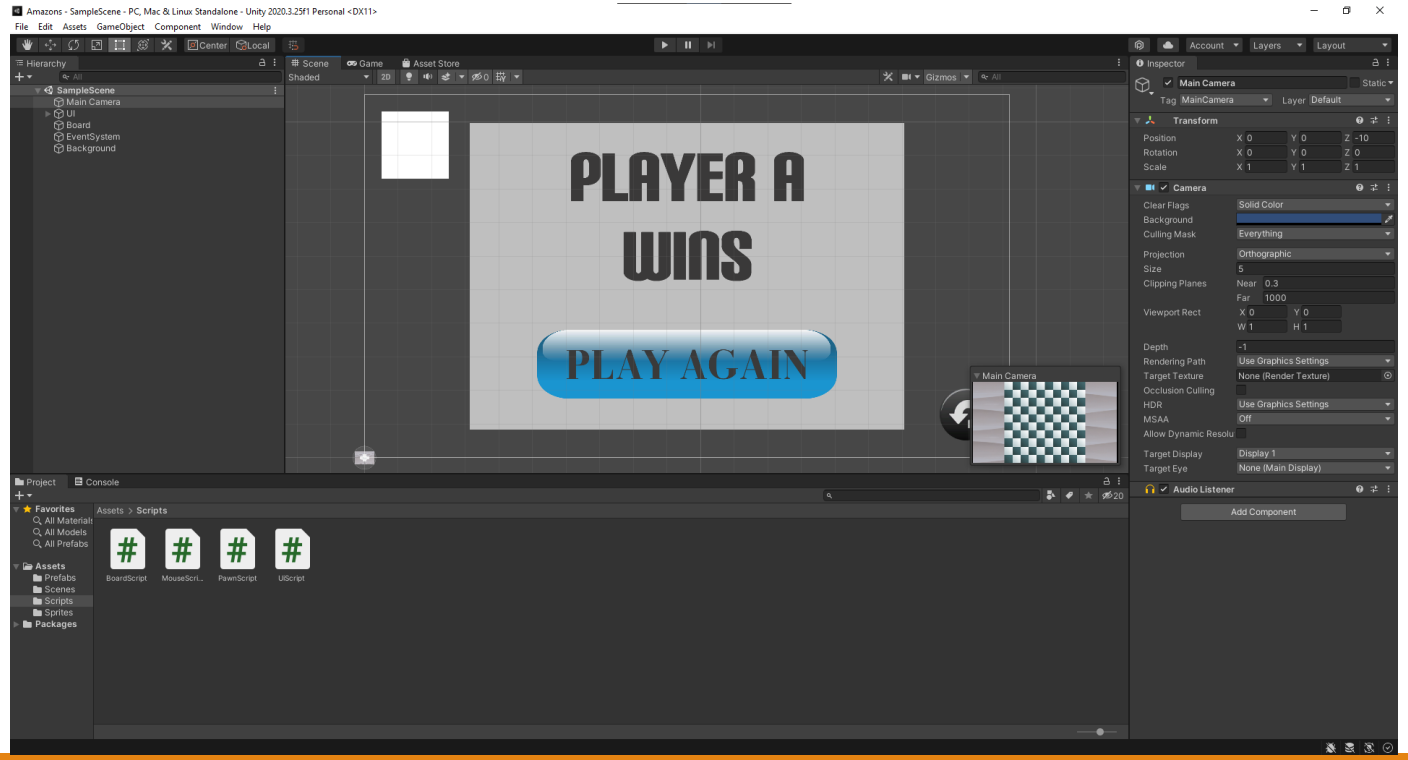
AUTOR: PAWEŁ KUMOROWSKI

Zasady

- Gra Amazons rozgrywa się na planszy 10x10
- Każdy z dwóch graczy kontroluje 4 figury
- Figury poruszają się i strzelają zgodnie z zasadami poruszania królowej w szachach
- Tura składa się z przesunięcia dowolnej figury oraz oddania strzału
- Strzał blokuje pole, nie można się po nim poruszać ani go przeskoczyć
- Gra kończy się w momencie, w którym któryś z graczy nie może poruszyć się żadną figurą, czyli przegrywa

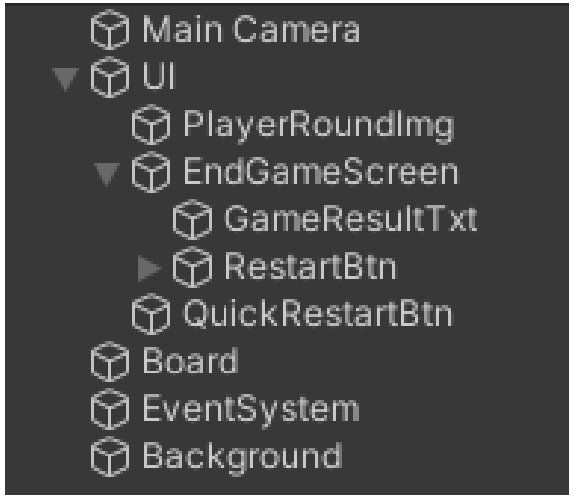
Środowisko Unity

Unity to silnik gier, który w znacznym stopniu usprawnia ich tworzenie. Wizualny edytor pozwala m.in. na tworzenie scen, podgląd i edycję obiektów czy też uruchamianie gry w celu testowania. Za zachowania obiektów odpowiedzialne są skrypty, w których zawarta jest cała logika gry.



Scena

- Main Camera – określa to, jaki fragment sceny jest widoczny dla gracza
- UI – obiekt interfejsu, znajdują się na nim informacje odnośnie: tury, zwycięscy, przyciski do szybkiego restartu oraz zagrania ponownie
- Board – plansza, znajdują się na niej skrypty odpowiedzialne za logikę gry oraz sterowanie
- EventSystem – pozwala na rejestrowanie wciśnięć myszy
- Background – tło



Skrypty

Gra składa się z czterech skryptów:

- BoardScript.cs – inicjalizacja gry, logika rund, ruchy, strzały
- PawnScript.cs – skrypt każdej figury, odpowiada za jej przesuwanie, generuje legalne ruchy
- MouseScript.cs – obsługa sterowania myszą
- UiScript.cs – wyświetlanie, ukrywanie i ustawianie elementów interfejsu

BoardScript.cs

Za inicjalizację plansz odpowiada metoda SetUpBoard(). Usuwa „pozostałości” z poprzedniej rozgrywki (w przypadku ponownej gry) oraz ustawia figury na planszy. Na koniec generuje wszystkie możliwe ruchy dla każdej figury w danej turze.

Za generowanie ruchów odpowiada metoda GenerateAllLegalMoves(), iteruje ona po wszystkich figurach i uruchamia dla każdej z nich metodę generującą dozwolone ruchy.

IsActionLegal() mówi czy dany ruch lub strzał jest możliwy do wykonania, poprzez sprawdzenie czy pole jest wolne oraz czy znajduje się ono na planszy.

GetScenePosition() i GetBoardPosition() konwertują współrzędne sceny oraz współrzędne planszy

BoardScript.cs

Figury poruszane są poprzez wywołanie metody MovePawn(). Po przesunięciu ich początkowa pozycja oraz końcowa dodawane są do historii ruchów dla danej tury (currentMoveHistory). Wywoływana jest również metoda skryptu konkretnej poruszanej figury, również o nazwie MovePawn().

Shoot() tworzy strzałę w podanym miejscu na planszy a następnie ustawia turę na turę przeciwnika. Na koniec koordynaty strzału zostają dodane do currentMoveHistory i całość zostaje dodana do listy zawierającej historię ruchów z każdej tury.

UndoMove() służy do cofania ostatniego ruchu. Na początku sprawdzane są warunki czy jakikolwiek ruch został już wykonany oraz czy gracz nie jest w trakcie poruszania/strzału. Jeżeli nie to usunięta zostaje ostatnio dodana strzała a następnie zostaje odpowiednio przesunięta figura, zgodnie z danymi z listy zawierającej historię rozgrywki.

PawnScript.cs

GenerateLegalMoves() sprawdza kolejno w każdą z 8 stron, czy kolejne pola są puste i mieszczą się na planszy. Każde kolejne pole zostaje dodane do ruchów legalnych, do momentu napotkania niepustego pola lub pola poza planszą.

ShowLegalMoves() wyświetla dozwolone ruchy poprzez renderowanie zielonej poświaty, natomiast HideLegalMoves() je chowa.

MovePawn() „fizycznie” przesuwa figurę, ponownie generuje dozwolone ruchy oraz je wyświetla. Będą one służyły do oddania strzału.

MouseScript.cs

W metodzie Update() (wywoływanej w każdej widocznej klatce gry) sprawdzane jest na co kliknięto myszą, np.: `hit.collider.gameObject.tag == "PlayerA"`

Zmienna hasMoved określa na jakim etapie poruszania się jest gracz. Jeżeli jest ona równa false, znaczy że nie zostało wykonane przesunięcie. Jeżeli natomiast jest ona równa true, znaczy że gracz się poruszył i będzie wykonywał strzał.

Jeżeli hasMoved = false sprawdzamy którego gracza figura została kliknięta. Jeżeli to figura gracza, którego jest obecnie tura, wyświetlamy możliwe dla niej ruchy. Jeżeli kolejne kliknięcie będzie na któreś z tych pól zostanie wykonany ruch i hasMoved zmieni się na true;

Jeżeli hasMoved = true sprawdzamy czy kliknięto na któreś z dozwolonych pól, by oddać w to miejsce strzał.

UiScript.cs

W metodzie Start() wywoływanej raz, przy tworzeniu skryptu w grze, dodajemy listenery na przyciski.

endScreen.SetActive() odpowiada za wyświetlanie i chowanie tablicy po zakończeniu rozgrywki.

SetRound() ustawia ikonę w górnym lewym rogu ekranu informującą o aktualnej turze.