Opis klas

★ `Main`

Klasa `Main` odpowiada za uruchomienie i zarządzanie główną pętlą gry w UJpoly. Inicjalizuje ona wszystkie kluczowe komponenty gry, takie jak okno gry, czas gry, oraz interakcje użytkownika. Klasa ta łączy ze sobą różne elementy gry i koordynuje ich działania. Głównym zadaniem tej klasy jest zbieranie danych z wejścia takich jak kliknięcie przycisków, myszy czy skalowanie okna. Wewnątrz klasy funkcjonuje również licznik czasu gry i tury, który ogranicza długości tur graczy oraz całej rozgrywki.

★ Menu

Klasa `Menu` pełni kluczową rolę w interfejsie użytkownika gry, umożliwiając graczowi wybór różnych opcji, zarządzanie ustawieniami gry oraz personalizację elementów graficznych. Obsługuje ona zarówno zdarzenia interakcji użytkownika, jak i renderowanie graficznych elementów menu.

★ Gra

Klasa `Gra` jest odpowiedzialna za zarządzanie rozgrywką. Klasa integruje różne komponenty gry, obsługuje logikę tury gracza, zarządza interakcjami między graczami a planszą oraz wyświetla elementy gry na ekranie.

Konstruktor klasy `Gra` inicjalizuje różne elementy gry:

- Ustawia początkowe parametry gry, takie jak liczba graczy, kwota początkowa, wymiary ekranu.
- Inicjalizuje wizualizator aby kontrolować kolorystykę i czcionki z jednego miejsca.
- Tworzy graczy za pomocą metody `stworz_graczy`.
- Inicjalizuje różne okna akcji (`AkcjaPolaOkno`, `AkcjaNieruchomosciOkno`, itp.), które są używane do interakcji graczy z planszą i nieruchomościami.
- Ładuje obrazy kostek do gry.
- Tworzy przyciski do przechodzenia do następnej tury i zapisywania oraz wychodzenia z gry.
- I wiele innych

Metoda główna: tura ( )

Klasa ma głównie za zadanie zapewnić turowość rozgrywki i obsługiwać logikę programu. Udostępnia wiele metod pomocniczych głównie używanych w głównej metodzie tura( ), w której odbywają się takie operacje jak: sprawdzenie czy gracz nie jest uwięziony, ustawienie odpowiedniego gracza według kolejki tury, symulacja rzutu kostką i analiza rzutu. Metoda analizuj_rzut( ) może wywołać szereg działań w zależności od tego na jakie pole przesunął się gracz. Finalnie metoda tura( ) sprawdza czy żaden z graczy nie zbankrutował lub wygrał w grze.

Klasa Gra, obsługuje aktualizację rozmiaru okien i przekazuje te informacje do wszystkich klas związanych z wyświetlaniem interfejsu. Ponadto zapewniony jest zapis całości rozgrywki dzięki metodzie zapisz( ) i wczytanie dzięki metodzie wczytaj( ).

★ Gracz

Klasa `Gracz` w przedstawionym kodzie służy do modelowania uczestnika gry. Celem tej klasy jest zarządzanie stanem i interakcjami gracza z różnymi elementami gry. Obejmuje to monitorowanie jego pieniędzy, posiadłości, specjalnych umiejętności oraz wykonywanie operacji finansowych i administracyjnych.

Główne cele:

- Reprezentacja gracza: Klasa przechowuje wszystkie niezbędne informacje o graczu, takie jak jego ID, kwotę pieniędzy, pionek, aktualną pozycję na planszy, licznik tur w więzieniu, liczbę kart wyjścia z więzienia, listę posiadłości, umiejętności i stan aktywności.
- Zarządzanie finansami: Klasa obsługuje wszelkie transakcje finansowe, takie jak płacenie czynszu, dodawanie pieniędzy, czy zastawianie posiadłości. Zapewnia również aktualizację statystyk finansowych gracza.
- Obsługa posiadłości: Klasa umożliwia graczowi zastawianie i zdejmowanie zastawów z posiadłości, monitorowanie liczby posiadłości w poszczególnych kolorach oraz sprawdzanie warunków do budowy domków i hoteli.

★ Pole

Klasa odpowiada za reprezentację i obsługę pojedynczego pola na planszy. Główne zadania tej klasy obejmują zarządzanie położeniem pola na planszy, jego rozmiarem oraz renderowaniem na ekranie. Obejmuje to rysowanie podstawowej grafiki pola, a także dodatkowych elementów, takich jak obramowania wskazujące na właściciela pola czy nakładki półprzezroczyste informujące, że pole jest zastawione. W zależności od statusu pola, metody renderujące mogą zmieniać jego wygląd, co pozwala graczom na łatwe rozpoznanie stanu każdego pola na planszy. Klasa zawiera metody do obliczania pozycji i orientacji pola na planszy w zależności od jego numeru i całkowitej liczby pól na planszy.

★ Posiadłość

Klasa `Posiadlosc` jest rozszerzeniem klasy `Pole` i reprezentuje posiadłości na planszy gry. Klasa ma wszystkie cechy i funkcje pola, takie jak zarządzanie położeniem na planszy i renderowaniem, a dodatkowo posiada specyficzne atrybuty i metody związane z posiadłościami. Główne zadania klasy obejmują zarządzanie własnością nieruchomości, obliczanie czynszu, a także umożliwianie kupna i sprzedaży domów oraz hoteli. Każda posiadłość ma nazwę, kolor, cenę zakupu, czynsz, kwotę zastawu oraz opcjonalną cenę budowy domu. Dodatkowo, posiadłość przechowuje informacje o właścicielu, liczbie domów i hoteli.

★ Moduł Okien

Abstrakcyjna klasa Okno udostępnia metody, które muszą zostać zaimplementowane w klasach pochodnych. Do tych metod należą między innymi metody obsługujące zdarzenia (pygame.events) jak i metody związane z wyświetlaniem czy zamykaniem okna. Moduł okien składa się z zestawu klas pochodnych przeznaczonych do wyświetlania i obsługi konkretnych akcji takich jak stanięcie na polu lub wyświetlenie kart szansy. Każda z tych klas dziedziczy po abstrakcyjnej klasie Okno.

★ Plansza

Klasa `Plansza` pełni rolę centralnego elementu w zarządzaniu planszą gry. Jej głównym zadaniem jest gromadzenie wszystkich pól, które tworzą planszę, oraz udostępnianie funkcji do renderowania tych pól na ekranie gry.

★ Pionek

Klasa, reprezentuje pionki poruszające się po planszy gry. Odpowiada za ich pozycję, wygląd oraz zachowanie podczas przemieszczania się. Klasa Pionek stanowi integralną część mechaniki ruchu i wizualizacji gry planszowej, zapewniając poprawne zachowanie oraz wygląd pionków poruszających się po planszy.

★ Statystyka

Klasa służy do przechowywania i aktualizacji statystyk gracza w grze.

★ KartySzansy

Klasa definiuje obiekt reprezentujący kartę szansy w grze. Każda karta posiada typ, treść oraz opcjonalną wartość. Metody klasy pozwalają na wyświetlanie treści karty oraz wykonywanie akcji związanych z daną kartą. Przy inicjalizacji wczytuje karty z pliku tekstowego, mieszając je, aby uzyskać losową kolejność. Oferuje również metodę `nastepna_karta()`, która pozwala na pobranie kolejnej karty ze stosu.

★ PodatekDochodowy i Zagadki

Klasa `Zagadka` reprezentuje pojedynczą zagadkę w grze. Każda zagadka zawiera treść, trzy możliwe odpowiedzi (A, B, C) oraz poprawną odpowiedź.

Inicjalizacja: Podczas tworzenia instancji klasy `Zagadka`, przekazuje się treść zagadki, trzy odpowiedzi oraz poprawną odpowiedź.

Klasa `Zagadki` zarządza zestawem zagadek w grze. Przy inicjalizacji wczytuje zagadki z pliku tekstowego, mieszając je, aby uzyskać losową kolejność. Oferuje również metodę `nastepna_zagadka()`, która pozwala na pobranie kolejnej zagadki ze stosu.

Klasa `PodatekDochodowy` jest specjalnym typem pola na planszy, które wymaga opłacenia podatku dochodowego przez gracza, który na nie trafi.

Zapłata podatku:

Metoda `zaplac_podatek()` odpowiada za obliczenie i pobranie opłaty za podatek dochodowy od gracza. Jeśli gracz odpowie poprawnie na zagadkę, koszt podatku zostaje pomniejszony o połowę.

- ★ Przycisk

Klasa `Przycisk` definiuje interaktywny przycisk w grze. Pozwala na jego rysowanie, sprawdzanie kliknięć, aktualizację rozmiaru i sprawdzanie czy kursor znajduje się nad przyciskiem.

- ★ Wizualizator

Klasa, przechowuje kolory używane w interfejsie użytkownika oraz czcionki. Zawiera różne zestawy kolorów dla przycisków, tła, napisów i innych elementów interfejsu. Definiuje również ścieżki do plików czcionek używanych w grze.