Projekt programu Gomoku

Aplikacja ma na celu wyznaczenie kolejnego najlepszego ruchu w grze Gomoku, bazując na analizie aktualnego stanu planszy. Gra toczy się na planszy o stałej ustalonej wielkości, a gracz ma za zadanie doprowadzić do układu pięciu znaków pod rząd (w pionie, poziomie lub po skosie). Analizowane są różne przypadki:

- Czy stan planszy jest poprawny?
- Czy można wygrać jednym ruchem?
- Czy przeciwnikowi grozi zwycięstwo i trzeba go zablokować?
- Czy są sytuacje groźby wygranej w 2 lub 3 ruchach?

Stworzyłem projekt aplikacji w oparciu o szereg wzorców projektowych, których zastosowanie pozwoliło na zaprojektowanie elastycznego, rozszerzalnego i czytelnego systemu zgodnego z zasadami dobrego programowania obiektowego. Projekt oparty jest na wzorcach takich jak metoda wytwórcza, pamiątka, odwiedzający, metoda szablonowa, strategia oraz kompozyt.

Opis wzorców projektowych i ich rola w aplikacji

Metoda Wytwórcza

Wzorzec ten został użyty do tworzenia obiektów reprezentujących plansze – w zależności od parametru konfiguracyjnego tworzony jest obiekt planszy z krawędziami stałymi lub planszy z periodycznymi warunkami brzegowymi. Dzięki temu aplikacja nie musi znać klas konkretnych implementacji planszy – korzysta wyłącznie z interfejsu Board.

- Zastosowanie: Uniezależnienie kodu logiki gry od konkretnej reprezentacji planszy.

Pamiątka

Ten wzorzec służy do zachowywania i przywracania stanu planszy. Jest to szczególnie ważne podczas stosowania wielu różnych algorytmów, gdzie mogą one modyfikować planszę (np. symulując ruchy). Zastosowanie pamiątki pozwala po zastosowaniu algorytmu przywrócić planszę do oryginalnego stanu, upewniając się, że niezależnie od zastosowanego algorytmu, plansza będzie niezmieniona.

- Zastosowanie: Bezpieczne korzystanie z algorytmów bez naruszania głównego stanu gry

- Współdziała z algorytmami opartymi o wzorzec metoda szablonowa, gdzie używane są różne strategie na tej samej planszy.

Odwiedzający

Wzorzec odwiedzający umożliwia rozdzielenie operacji od struktur danych – tutaj od klas FixedBoard i PeriodicBoundaryBoard. Dzięki temu logika gry (np. wybór ruchu) nie musi być zaimplementowana w klasach planszy. Ruchy są wykonywane przez odwiedzających, którzy „odwiedzają" planszę i podejmują decyzje.

- Zastosowanie: Oddzielenie logiki algorytmu od reprezentacji planszy.
- Klasa odwiedzająca (Visitor) uruchamia algorytmy zaimplementowane częściowo w metodzie szablonowej.

Metoda Szablonowa

Metoda szablonowa definiuje wspólny schemat działania algorytmu wyboru ruchu, pozostawiając implementację strategii przechodzenia planszy.

- Zastosowanie: Wspólna logika dla różnych odwiedzających, z możliwością wyboru różnych strategii.
- Metoda szablonowa wykorzystywana jest w klasach odwiedzających. Współpracuje również ze wzorcem strategia.

Strategia

Ten wzorzec określa sposób poruszania się po planszy – np. jak znaleźć kolejne pole w każdym kierunku. Różne implementacje strategii są wykorzystywane w zależności od typu planszy - inne reguły dla periodycznych warunków brzegowych, inne dla stałych granic.

- Zastosowanie: Elastyczne określanie reguł przechodzenia między polami planszy.
- Połączenie: Przechowywana w Metodzie Szablonowej oraz używana w strukturach Kompozytu, które analizują układy na planszy.

Kompozyt

Wzorzec kompozyt został użyty do modelowania złożonych wzorców układów pól na planszy, np. 5 w rzędzie, 4 w rzędzie, kształt krzyża itp. Wzorce są połączone w grupy (np. grupa „IlegalStateGroup"), a każda grupa może zawierać inne wzorce lub kolejne grupy.

- Zastosowanie: Reprezentacja i analiza złożonych układów pól na planszy.
- Wzorce są wykorzystywane przez klasy odwiedzające i szablonowe do oceny stanu planszy i wyboru optymalnego ruchu.

Diagram klas