Projektowanie obiektowe oprogramowania
Zestaw 8

Wzorce czynnościowe (3)

2025-04-15

Liczba punktów do zdobycia: **7/57**
Zestaw ważny do: 2025-05-06

1. (**1p**) (**Chain of Responsibility**) Przygotowano system przetwarzania przychodzących wiadomości e-mail. W trakcie analizy wyróżniono cztery rodzaje wiadomości: (1) pochwalne, (2) skargi, (3) zamówienia (4) pozostałe.

   Dodatkowo wszystkie wiadomości powinny być archiwizowane.

   Przygotować implementację takiego systemu obsługi poczty w oparciu o wzorzec Chain of Responsibility. Argumentem przetwarzania jest obiekt `Request`, którego jednym z atrybutów jest wiadomość (obiekt posiadający tytuł i treść).

   Zadaniem pierwszego handlera w łańcuchu jest klasyfikacja - jakiego rodzaju jest to wiadomość. Pierwszy handler ustawia więc dodatkowy atrybut w obiekcie żądania (`Request`), wskazujący na rodzaj wiadomości.

   Kolejne handlery łańcucha nie muszą już więc samodzielnie dokonywać klasyfikacji. Zamiast tego na podstawie klasyfikacji dokonanej przez pierwszy handler, kolejne handlery przetwarzają wiadomość albo nie - do każdego rodzaju wiadomości w łańcuchu jest osobny, dedykowany handler.

   Ostatni handler to handler wiadomości nieprawidłowych - zadbać o to żeby jednym z możliwych efektów działania pierwszego handlera, ustawiającego rodzaj wiadomości, było rozpoznanie że wiadomość jest błędna (np. pusta, zbyt długa) i takie oznakowanie żądania, żeby kolejne handlery rodzajów wiadomości zignorowały tę wiadomość, a przetworzył ją ten ostatni handeler wiadomości nieprawidłowych.

   Taka konstrukcja oznacza, że łańcuch będzie miał co najmniej 7 elementów:

   - klasyfikator
   - 4 handlery na poszczególnych rodzajów wiadomości
   - handler błędnej wiadomości
   - handler archiwizujacy (np. zapisujący wiadomość gdzieś do pliku)

2. (**2p**) (**Command**) Mamy cztery rodzaje poleceń:

   (a) pobranie podanego pliku przez FTP
   (b) pobranie podanego pliku przez HTTP
   (c) utworzenie na dysku pustego pliku i wypełnienie go losową zawartością
   (d) skopiowanie wskazanego pliku pod inną nazwą

Przygotować implementacje wspólnego interfejsu dla każdego z poleceń, następnie zaimplementować cztery konkretne rodzaje poleceń.

Zwrócić uwagę na właściwą delegację polecenia do odbiorcy (Receiver), który zajmuje się samą implementacją - odbiorca nie jest częścią hierarchi określonej przez polecenia. Zwrócić uwagę, żeby polecenie nie było tylko "delegacją" do odbiorcy, tzn. zapewnić poleceniu "stan", który przekazuje odbiorcy (np. dla pobrania pliku stanem jest nazwa pliku).

(część pierwsza zadania za 1p)

Przygotować automat kolejkujący (pełni tu rolę wywołującego (Invoker): jeden wątek będzie tworzy żądania i dodaje do kolejki, dwa inne ściągają je z kolejki i obsługują.

(część druga zadania za 1p)

3. (**1p**) (**Template Method**)

   Klasa dostępu do danych, `DataAccessHandler` ma metodę `Execute`, wykonującą zawsze następującą sekwencję czterech operacji:

   (a) połączenie do danych,

   (b) pobranie danych,

   (c) przetwarzanie danych,

   (d) zamykanie połączenia.

   Opisać tę sekwencję operacji metodą szablonową.

   Przygotować dwie implementacje klas konkretnych : jedna pobiera dane z bazy danych i liczy sumę wartości którejś z kolumn, druga pobiera dane z pliku XML i wyszukuje węzeł o najdłuższej nazwie.

4. (**1p**) (**Strategy**)

   Zrefaktoryzować strukturę klas z poprzedniego zadania do wzorca `Strategy`.

5. (**2p**) (**State**)

   Przygotować implementację maszyny stanowej dla której diagram przygotowywany był w zadaniu 3 z zestawu 2.

   Implementacja powinna być zgodna ze wzorcem State.

   Jeżeli diagram był zbyt złożony - uprościć go do kilku stanów.

Wiktor Zychla