

Projektowanie obiektowe oprogramowania

Zestaw C (ostatni)

ORM/MVC/MVP

2019-06-04

Liczba punktów do zdobycia: **6/80**
Zestaw ważny do: koniec semestru

1. (2p) (ORM + IoC)

Zaprezentować wybraną przez siebie technologię mapowania obiektowo-relacyjnego na prostym przykładzie dwóch tabel (Parent-Child). Mapowanie powinno być częścią prostej aplikacji konsolowej, a za zarządzanie tworzeniem sesji dostępu powinna odpowiadać lokalna fabryka (Local Factory) wykorzystująca jakiś wybrany kontener IoC.

Zastanowić się nad wyborem i właściwą implementacją polityki czasu życia w ramach IoC dla tworzonych sesji dostępu do danych ORM.

Jakie polityki czasu życia sesji są dobre dla aplikacji typu desktop, a jakie dla aplikacji typu web? Kiedy i jak powinno odbywać się inicjowanie sesji dostępu do danych i jej niszczenie?

2. (2p) (MVC)

Rozszerzenie zadania pierwszego.

Zaprezentować wybraną przez siebie implementację wzorca MVC dla tworzenia aplikacji typu web. Aplikacja pokazuje listę danych, pozwala użytkownikowi przywołać widok dodawania/modyfikacji danych, przeprowadza walidację, zapisuje dane do bazy i powraca do widoku listy.

Do dostępu do danych korzysta się poprawnie z infrastruktury z zadania pierwszego - sesja ORM jest do kontrolerów wstrzykiwana przez wybrany kontener IoC.

Pokazać, że możliwy jest test jednostkowy samego kontrolera, bez potrzeby tworzenia pełnego środowiska serwera aplikacyjnego.

3. (2p) (MVP)

Rozszerzenie zadania pierwszego.

Zaprezentować własną implementację wzorca MVP dla aplikacji typu desktop, podobną do tej z wykładu, rozwijającą zadanie pierwsze.

Aplikacja pokazuje listę danych, pozwala użytkownikowi przywołać widok dodawania i modyfikacji danych, przeprowadza walidację, zapisuje dane do bazy i powraca do widoku listy.

Do dostępu do danych korzysta się poprawnie z infrastruktury z zadania pierwszego - sesja ORM jest do prezenterów wstrzykiwana przez wybrany kontener IoC.

Pokazać, że wprowadzenie rozłącznych warstw widoków i prezenterów pozwala na dostarczenie implementacji widoków zastępczych i testy jednostkowe warstwy prezenterów bez potrzeby używania rzeczywistych implementacji widoków.

Wiktor Zychla