Communicator project

Tornado

https://github.com/piotrowskislawomir/Communicator
Adrian Janiak
Michał Masztalerz
Monika Piątkowska
Sławomir Piotrowski

1. `Communicator.BusinessLayer.Services.CommonUserListService` – mogliście pomyśleć o wyrzuceniu stałych tekstowych (np. User, Login) do osobnego pliku. Dobrze widziane jest trzymanie stałych tekstowych w jednym miejscu (łato je potem podmienic) – można potem nawet bawić się w tłumaczenia itd. Takie rzeczy można znaleźć w wielu miejscach u was - `Communicator.Server.Program` np.
2. `Communicator.BusinessLayer.Services.CommonUserListService` – linia 13 – dobrym zwyczajem jest deklarowanie zmiennych, jako interfejsów. Dotyczy to oczywiście wielu miejsc – jak np. Przyjmowanych parametrów, czy też rezultatów z metod. Tutaj macie odpowiedź Erica Lipperta dlaczego warto myśleć o takich rzeczach - http://stackoverflow.com/questions/8717582/why-use-ilist-or-list
3. `Communicator.BusinessLayer.Services.CommonUserListService` – metoda CreateNewUser(..) – komentarze nie są zgodne z konwencją, chociaż przynajmniej są J VS ma skrót do tego – użycie /// powoduje wygenerowania stubów do wpisania komentarzy w odpowiedni sposób (by potem wygenerować dokumentację projektu!). W projekcie w ogole brakuje komentarzy – warto o nich pamietać, na serio... J
4. Rozwiązanie `Communicator.Untils.Services.XmlConfigurationService` jest rozsądne, ale nie spotkałem się z wykorzystaniem plików konfiguracyjnych – rozumiem, że to po prostu nie poleciało na repo?
5. `Communicator.Client.ViewModels.CommunicatorViewModel` – metoda `ProceedCommand` mogłaby spokojnie byc podzielona na kilka innych metod. Tak by każda z metod jednoznacznie opisywała JEDNĄ z wykonywanych funkcji. Zauważyłem, że w wielu miejscach wrzucaliście logikę „przetwarzania commanda” bezpośrednio do ProceedCommand – standardem jest wydzielenie osobnych metod, jak wcześniej.
6. Bardzo dobry podział na logicznie niezależne komponenty.
7. IoC - super!.
8. Aplikacja zgłasza, że użytkownik został zarejestrowany, ale serwer informuje o błędzie.
9. Jeżeli obsługujecie wyjątek - pamiętajcie o zalogowaniu treści, znacznie ułatwia to debugowanie - klientowi można wyświetlić jakąś ogólną informację, ale z punktu widzenia debugowania stacktrace baaardzo pomaga.

10. Jeżeli pliki z których korzystanie nie istnieję (userlist.xml) to aplikacja mogłaby je automatycznie tworzyć.
12. Starajcie się utrzymywać długość metod w granicach 40 linii (1 ekranu) - znacznie ułatiwa to śledzenie co się dzieje.
13. MessageRecognizerService.Timer() - do tego można wykorzystać klasy Timer z System.Threading.Timer albo System.Timers.Timer.
14. Obrazki statusowe mogłyby być zaimplementowane przez konwerter.
15. Super UI!

Świetny projekt - gratuluję!