12.01. 2013

Ksoap StateConnector

Artur Augustyniak

Sem. VII

nr alb. 808

Wstęp

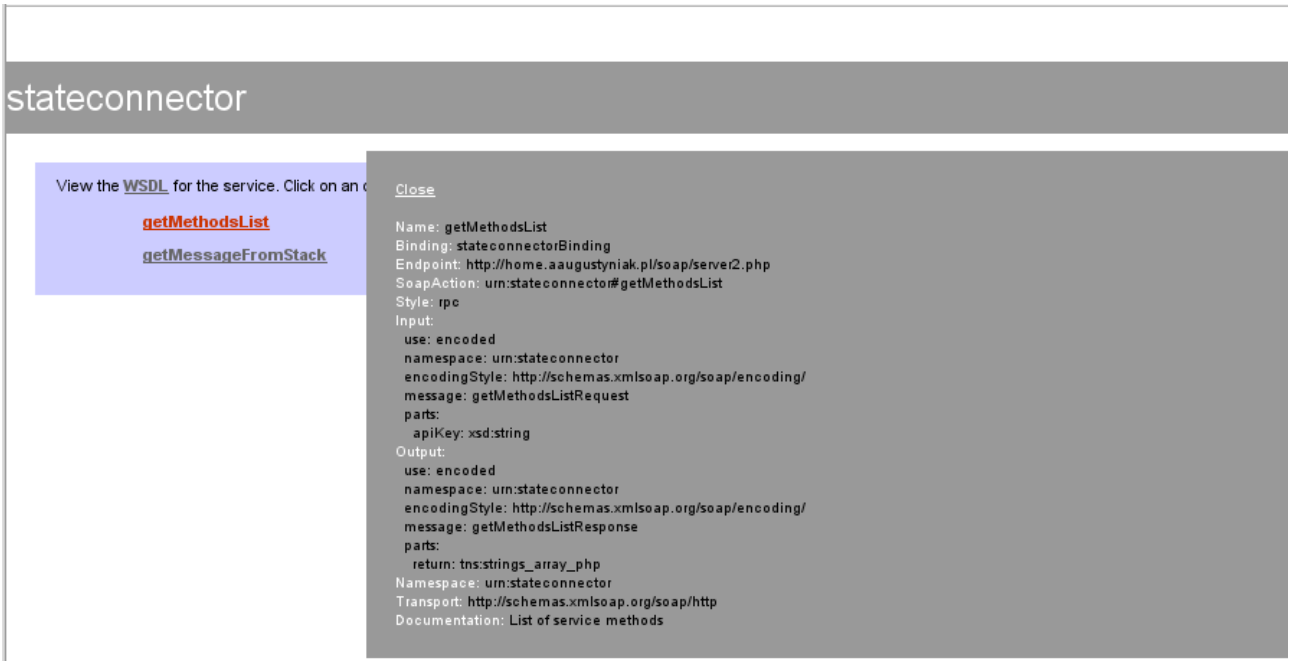
SoapConnector to aplikacja będąca możliwie bezstanowym konsumentem usług web serwisów.
Jedyne dane przechowywane przez klienta działającego w środowisku systemu Android to krotki {url, apiKey}. Wstępnym przeznaczeniem takiego oprogramowania jest zdalna diagnostyka oraz raportowanie w heterogenicznych środowiskach. Izolacja jest osiągnięta Dzięki zastosowaniu warstwy abstrakcji w postaci protokołu SOAP oraz opisu WSDL.

Niestety braki w dokumentacji, bądź niedoskonałość biblioteki ksoap2-android powodują konieczność implementacji pewnego określonego interface'u przez web serwis wykorzystywany przez moją aplikację.

Web Service

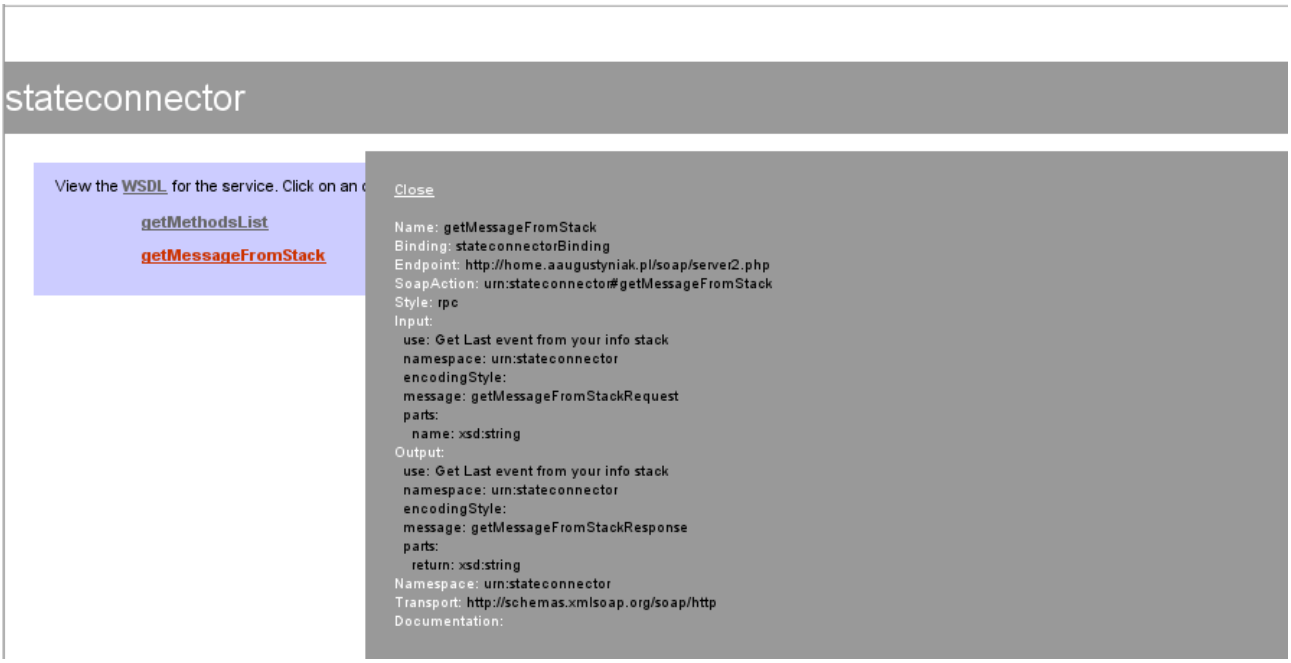
Technologia w której jest zaimplementowane zaplecze danego web serwisu nie gra żadnej roli, jako że jedynym punktem styku klienta działającego na urządzeniu użytkownika jest dokument w standardzie WSDL.

Jak wspominałem wyżej niestety dany web serwis musi posiadać pewien minimalny interface.



Rys. 1 getMethodsList – Metoda zwracająca listę dostępnych w danym serwisie metod „roboczych”

getMethodsList – jest niezbędna do zbudowania dynamicznej listy możliwych działań per zdefiniowany serwer.



Rys. 1 getMessageFromStack – Metoda niezbędna tylko dla działającej w w tle Usługi PrompterService

Każda metoda web serwisu jest wywoływana z parametrem apiKey.
Autoryzacja ogranicza się do sprawdzania poprawności przesłanego klucza.

Testowa implementacja usług po stronie serwera serwera została wykonana przy użyciu PHP oraz biblioteki nu_soap (http://sourceforge.net/projects/nusoap/), są to w istocie skrypty zaślepki generujące bądź losowe dane bądź odczytujące dane ze `stdout` serwera. W realnej implementacji kwestie związane np. z kolejkowaniem wiadomości dla `getMessageFromStack` pozostawiam po stronie implementacji web serwisu.

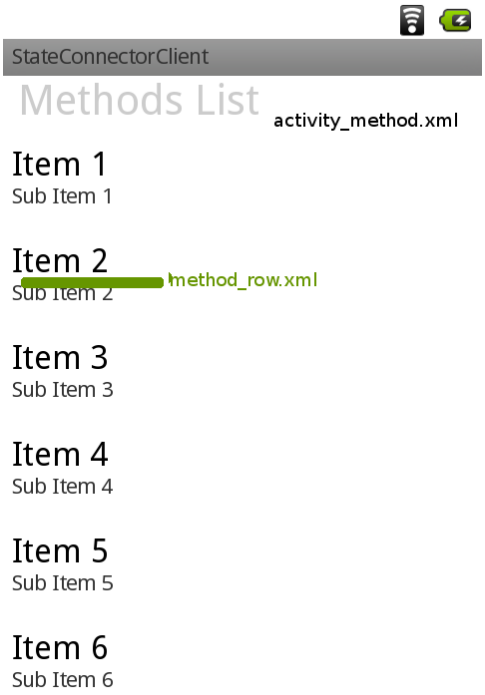
Przykładowa zaślepka: http://aaugustyniak.pl/soap/

Rozpatrywany projekt w swoich założeniach miał realizować następujące cele.

1. Budowa aplikacji z dynamicznym modułowym UI
2. Podział kodu według paradygmatu MVC
3. Przygotowanie ponownie używalnych klas dla tworzenia wygodnych CRUD list
4. Wstęp do programowania web serwisów

Budowa aplikacji z dynamicznym modułowym UI

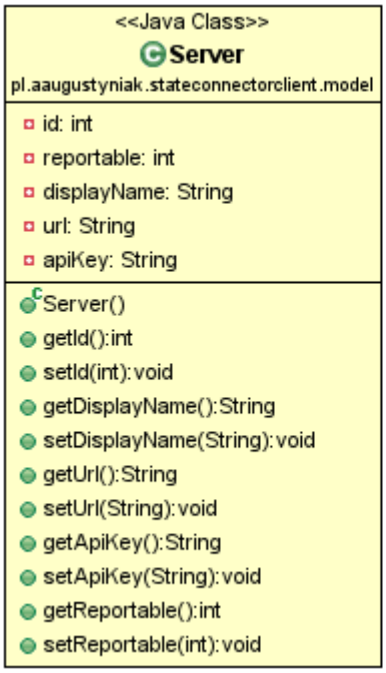
Widoki list zostały przygotowane wg wzorca dekoratora, dysponujemy tu (dla każdej listy) templatką wiersza jak i kontenerem - layoutem. Przy czym taki podział istnieje niezależnie od źródła danych (sqlite lub web serwis).

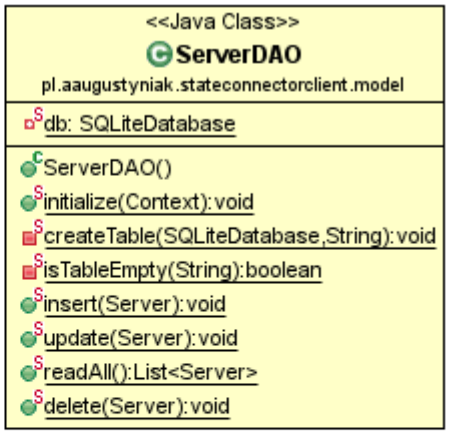


Podział kodu według paradygmatu MVC

Oprócz oczywistego dla aplikacji pisanej przy użyciu framework'u Google podziału na warstwy widoku i kontrolera postanowiłem napisać dla aplikacji adapter obiektowy do bazy sqlite (Wzorując się na absolutnych podstawach koncepcji ORM). Ponieważ moja aplikacja utrzymuje tylko jedną tabelę „servers" dysponuję następującymi klasami.

- Server – obiekt POJO reprezentujący encję tabeli servers.
- ServerDAO – Data Access Object (PROPEL – Peer).

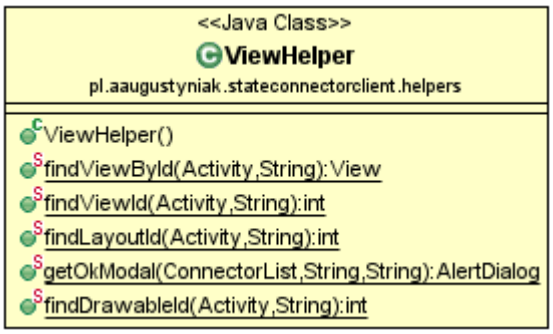




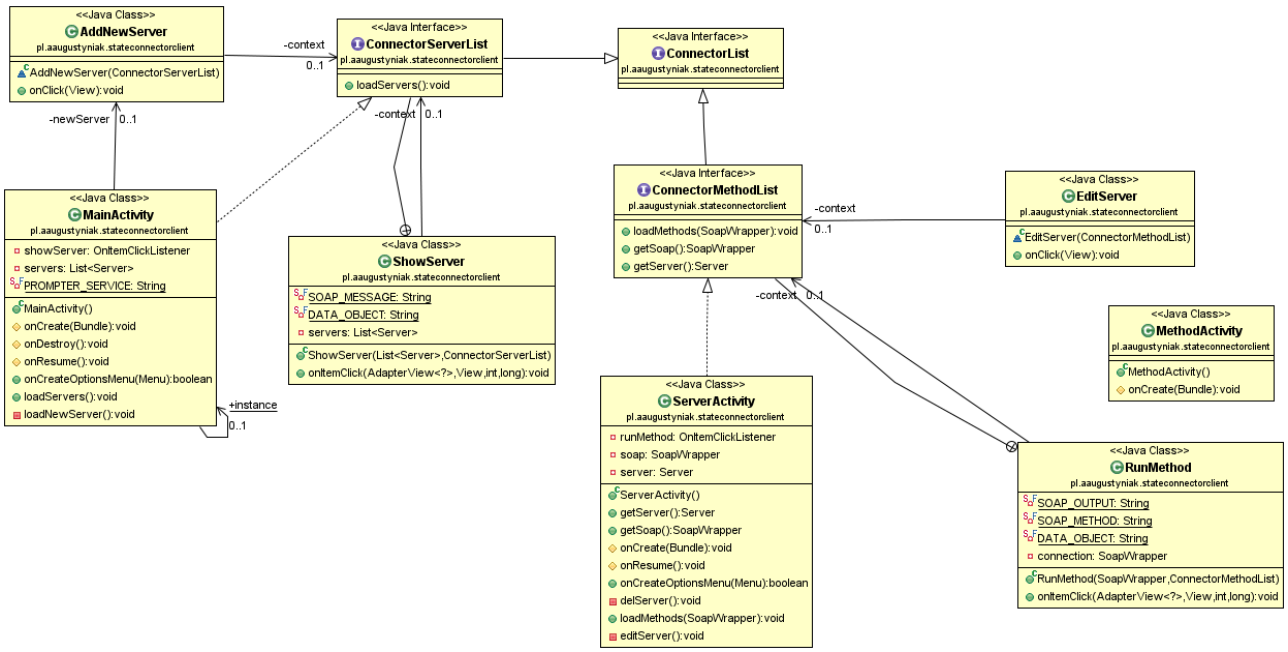
Przygotowanie ponownie używalnych klas dla tworzenia wygodnych CRUD list

W tym celu dostosowałem do własnych potrzeb klasę ViewHelper pochodząca z framework'u OpenMobster (http://code.google.com/p/openmobster/).

W mojej implementacji wygląda ona następująco:

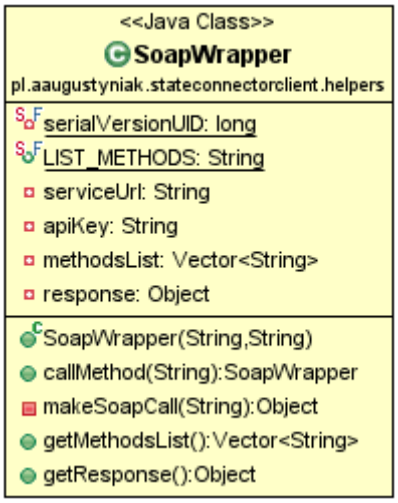


Dla klas obiektów zdarzeń przygotowana została hierarchia interface'ów pozwalających na łatwą zmianę przepływu sterowania .

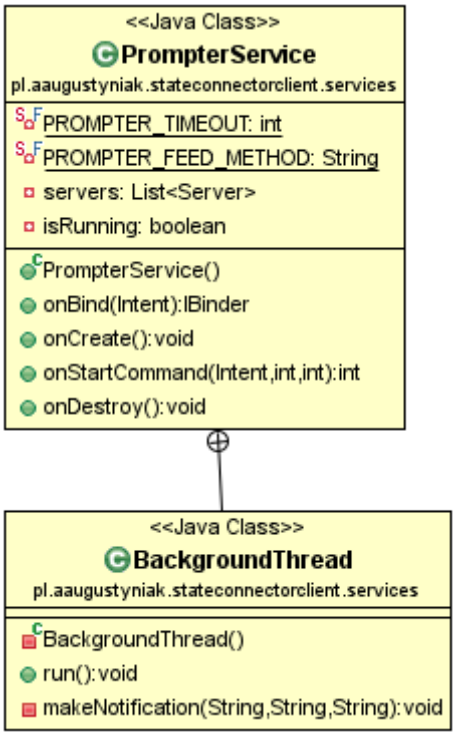


Wstęp do programowania web serwisów

Dla aplikacji przygotowana została klasa agregująca funkcjonalność bliblioteki ksoap (http://code.google.com/p/ksoap2-android/).



Ostatnią funkcjonalnością Klienta jest usługa (Android Service) prompter, pobierająca wiadomość z bufora web serwisu oraz wyświetlająca notyfikację.



Realizując projekt korzystałem z następujących narzędzi/tutoriali:

http://developer.android.com/index.html
http://code.google.com/p/openmobster/wiki/AndroidService
http://code.google.com/p/ksoap2-android/
http://code.google.com/p/openmobster/
http://naveenbalani.com/index.php/2011/01/invoke-webservices-from-android/
http://stackoverflow.com/questions/2530548/browse-data-in-android-sqlite-database
http://stackoverflow.com/questions/9853113/java-lang-noclassdeffounderror-org-ksoap2-serialization-soapobject
http://www.javaranch.com/journal/2002/05/axis.html
http://my-source-codes.blogspot.com/2011/10/php-web-service-return-array.html
http://stackoverflow.com/questions/1052300/how-to-call-a-net-webservice-from-android-using-ksoap2
http://www.objectaid.com/