



Systemy rozproszone

Informatyka, sem. 6 część II

Paweł Kaczmarek



Tematyka części II wykładu

- Zaliczenie
 - przekroczenie progu zaliczenia każdej z części
 - 20 % punktów (za II część) za obecność na wykładzie
 - projekt
- Tematyka (teoria + demo)
 - Programowanie aplikacji rozproszonych
 - specyfika debugowania, "best practices" użycia wątków
 - Wzorce komunikacji
 - send-receive, invoke, messaging, streaming
 - Synchronizacja, zarządzanie procesami



Tematyka, cd

- Tolerowanie uszkodzeń
 - obsługa wyjątków, transakcje, replikacja
- Literatura
 - A.S. Tanenbaum, M. van Steen: Distributed Systems, Principles and Paradigms
 - wykłady do książki (<http://www.cs.vu.nl/~ast/books/ds1/>)
 - G. Coulouris, J. Dollimore, T. Kindberg: Systemy rozproszone, podstawy i projektowanie
 - wykłady do książki
 - J. Bloch Effective Java, Programming Language Guide



Debuggowanie aplikacji zdalnych i wielowątkowych



Debuggowanie aplikacji zdalnej w Javie - demo

- Kiedy
 - np. kod dostępny lokalnie, aplikacja działa w innym środowisku, złożony proces kompilacji, konieczność uruchomienia poza środowiskiem IDE
 - np. aplikacja www na Tomcat, aplikacja na JBoss AS, ...
- Środowisko wykonania:
 - Zdalna aplikacja
 - aplikacja skompilowana z opcjami debuggera
 - VM, która umożliwia połączenie zdalnego debuggera
 - IDE (np. Eclipse IDE) wraz z kodem źródłowym aplikacji
 - możliwość wskazania miejsca kodu źródłowego



Debuggowanie: Eclipse Help

- Wykorzystanie: Remote Java Application launch configuration
 - zamiast parametrów VM - parametry połączenia
- Kroki
 - Wybranie Run > Debug
 - Utworzenie nowego Remote Java Application
 - Ustawienie lokalizacji kodu źródłowego
 - Ustawienie parametrów połączenia
 - "Allow termination of remote VM flag "
 - Debug



Debuggowanie aplikacji zdalnej w Javie, cd

- Uruchomienie aplikacji Java z wiersza poleceń
 - opcje zależne od wersji maszyny wirtualnej
 - `java -Xdebug -Xnoagent -Xrunjdwp :transport=dt_socket,address=8000,server=y,suspend=n -cp MyExample.jar example.pg.gda.pl.MyCall`
- Uruchomienie aplikacji z możliwością debuggowania z poziomu Eclipse IDE
 - opcje dla maszyny wirtualnej
 - `-Xdebug -Xnoagent -Xrunjdwp :transport=dt_socket,address=8000,server=y,suspend=n`
 - Kolejność uruchomienia
 - 1. zdalna aplikacja, 2. proces debugowania



Debugowanie działającego procesu w Visual Studio



- Proces uruchomiony na lokalnym komputerze
 - skompilowany z opcjami debuggera
- Kod źródłowy procesu dostępny w Visual Studio
- Debug -> AttachToProcess
 - wybór procesu do podłączenia
 - podłączenie kodu zarządzanego i nie zarządzanego



Debugowanie aplikacji wielowątkowej



- Ustawienie breakpointów
 - Run > Open Debug Dialog > Java Application
 - > Apply, Debug
- Okno Debug
 - Window > Show View > Other > ... > Debug
 - Konfiguracja aplikacji [Java application]
 - Main class
 - Wątki [Running, Stepping, Suspended]
 - Stack trace



Zasady użycia wątków "Best practices"



Synchronizacja - dostęp do zmieniających się danych



- Dwa cele "synchronized"
 - wzajemne wykluczanie / dostęp tylko do spójnych danych współdzielonych
 - wykonanie programu w ustalonym porządku, jakby wykonał się sekwencyjnie, komunikacja między wątkami
- Synchronized jest konieczny również przy dostępie do danych atomowych
 - np. `counter++`



Zakończenie wątku

- Thread.stop() - funkcja przestarzała i nie zalecana
 - wymuszenie zdjęcia semaforów, potencjalny dostęp do niespójnych obiektów
 - możliwość uszkodzenia innych obiektów
- Lepsze rozwiązanie: znacznik zakończenia wątku
 - wątek sprawdza, czy znacznik jest ustawiony



Unikanie nadmiernej synchronizacji

- Nie należy cedować sterowania do klienta z bloku synchronized - możliwość deadlocku
 - np. metody przewidziane do nadpisania (abstract)
 - przenieść poza blok synchronized
- Ogólnie w sekcji krytycznej należy wykonywać tylko te operacje, które są konieczne
- Klasy bezpiecznie wątkowo (alternatywy)
 - cała klasa synchronized - narzut wydajnościowy
 - dwa warianty klasy: zwykła + thread safe wrapper
 - wewnętrzna synchronizacja wybranych fragmentów



Nie należy wywoływać wait poza pętlą

- Należy wywoływać wait w pętli sprawdzającej
- Możliwe sytuacje bez pętli dla wait
 - inny wątek wykonał coś między wykonaniem notyfy a obudzeniem czekającego wątku
 - inny wątek mógł wykonać błędnie notyfy w sytuacji, gdy warunek nie był spełniony
 - inny wątek mógł być zbyt hojny w budzeniu
- Jeżeli wait są w pętli, lepiej używać notifyAll
 - obudzi wszystkie wątki, które powinien
 - obudzi wątek, jeżeli błędnie śpi (pewien narzut)



Nie należy polegać na schedulerze wątków

- Uzależnienie od schedulera wątków (np. yield)
 - mało przenośna aplikacja
 - wydajność / zagłócenie wątków
 - zapobieganie: tworzenie wątków, które wykonują zadanie i kończą
 - nie gwarantuje poprawności aplikacji
- Busy wait - niepotrzebne zajmowanie procesora
- Thread.yield nie rozwiązuje problemów
 - nieprzenośne wyniki (wydajność, poprawność)
 - przydatny przy testowaniu - przełączanie wątków



Dokumentowanie thread safety



- Słowo synchronized nie jest częścią API tylko "szczegółem implementacyjnym"
- Określenie thread safety w javadoc:
 - immutable - instancje zachowują się jak stałe, niepotrzebna synchronizacja, np. String
 - thread-safe - instancje są zmienne, ale nie jest wymagana zewnętrzna synchronizacja, wewnętrznie zaimplementowano synchronizację, np. `java.util.Timer`
 - cd ...



Dokumentowanie thread safety cd

- Określenie thread safety w javadoc (cd):
 - conditionally thread-safe zawiera metody, które muszą być wykonane w pewnej sekwencji bez przerywania przez inny wątek, konieczność opisanie metod, klient musi użyć synchronizacji
 - thread-compatible - wymagana zewnętrzna synchronizacja podczas wywoływania metod, ale klasa działa poprawnie w przypadku takiej synchronizacji, np. ArrayList, HashMap
 - thread-hostile - nie powinna być używana przez równoległe wątki, obecnie rzadko spotykana, np. System.runFinalizersOnExit (deprecated)



ThreadGroup to klasa przestarzała

- Miała na celu ułatwienie tworzenia apletów
 - w praktyce nie przyjęła się.
 - obecnie prawie nie używana
- Umożliwia zmianę parametrów wątków, ale część z tych parametrów też jest przestarzała
- Wyjątek użycia ThreadGroup:
 - `ThreadGroup.uncaughtException` - złapanie wyjątku, jeżeli wątek go wyrzuca.