Advanced Java technologies: JBoss

Časť 2.

Contexts and Dependency Injection (CDI)

Enterprise JavaBeans

Jozef Hartinger

March 2015

redhat

Projekt

- https://developer.jboss.org/wiki/AdvancedJavaEELabjaro2015
- git clone git@github.com:qa/pv243-2015-cdi-seminar.git
- Branches
  - Počiatočná cdi-00

redhat

Úloha 1: CDI Beans

- Vytvorte CDI komponentu, ktorá bude s použitím komponenty **MathOperations** (injection) implementovať rozhranie **Factorial** a bude zdieľaná vrámci celej aplikácie.
- Riešenie overte pomocou testu **FactorialTest**
  - **Arquillian**

redhat

Úloha 1: CDI Beans

- Vytvorte CDI komponentu, ktorá bude s použitím komponenty **MathOperations** (injection) implementovať rozhranie **Factorial** a bude zdieľaná vrámci celej aplikácie.
- Riešenie overte pomocou testu **FactorialTest**
- Riešenie v branch cdi-01

redhat

Úloha 2: View layer integration

- Prepojte Factorial komponentu s view vrstvou
  - Doimplementujte triedu FactorialForm
    - name
    - scope
    - compute()

redhat

Úloha 2: View layer integration

- Prepojte Factorial komponentu s view vrstvou
  - Doimplementujte triedu FactorialForm
    - name
    - scope
    - compute()
- Riešenie v branch cdi-02

redhat

Úloha 3: Task parallelization

- Upravte komponentu **MathOperations** a vytvorte alternatívnu implementáciu rozhrania **Factorial** tak, aby výpočet prebiehal v dvoch vláknach (s použitím EJB asynchrónneho volania metód).
- Odlíšte vytvorenú implementáciu od predchádzajúcej s pomocou novo-vytvoreného kvalifikátora (**@Parallel**).
- Upravte **FactorialTest** a **factorial.xhtml** tak aby používali paralelnú implementáciu
- Tipy:
  - **@Asynchronous**
  - **Future<BigInteger>**
  - **AsyncResult**

redhat

Úloha 3: Task parallelization

- Upravte komponentu **MathOperations** a vytvorte alternatívnu implementáciu rozhrania **Factorial** tak, aby výpočet prebiehal v dvoch vláknach (s použitím EJB asynchrónneho volania metód).
- Odlíšte vytvorenú implementáciu od predchádzajúcej s pomocou novo-vytvoreného kvalifikátora (**@Parallel**).
- Upravte **FactorialTest** a **factorial.xhtml** tak aby používali paralelnú implementáciu
- Tipy:
  - **@Asynchronous**
  - **Future<BigInteger>**
  - **AsyncResult**
- Riešenie v branch cdi-03

redhat

Úloha 4: Events

- Rozšírte komponentu ParallelFactorial tak, aby po každom dokončení výpočtu vyvolala udalosť **FactorialComputationFinished**
- Funkčnosť overte pomocou **FactorialTest.testEvent()** prípadne pomocou novo-vytvorenej komponenty, ktorá reaguje na udalosť a vypíše výsledok výpočtu na štandardný výstup
- Tipy:
  - **Event<FactorialComputationFinished>**
  - **@Observes**

redhat

Úloha 4: Events

- Rozšírte komponentu ParallelFactorial tak, aby po každom dokončení výpočtu vyvolala udalosť **FactorialComputationFinished**
- Funkčnosť overte pomocou **FactorialTest.testEvent()** prípadne pomocou novo-vytvorenej komponenty, ktorá reaguje na udalosť a vypíše výsledok výpočtu na štandardný výstup
- Tipy:
  - **Event<FactorialComputationFinished>**
  - **@Observes**
- Riešenie v branch cdi-04

redhat

Úloha 5: Singletons

- Vytvorte komponentu zdieľanú vrámci celej aplikácie ktorá bude reagovať na udalosť **FactorialComputationFinished** a bude ukladať dvojice (číslo, číslo!).
- Komponenta bude poskytovať operácie:
  - **get(long number) : BigInteger**
  - **clear() : void**
- Nebude použitá thread-safe dátová štruktúra. Namiesto toho bude prístup k dátovej štruktúre chránený s pomocou read/write zámku (EJB)
- Tipy:
  - @Singleton
  - @Lock

redhat

Úloha 5: Singletons

- Vytvorte komponentu zdieľanú vrámci celej aplikácie ktorá bude reagovať na udalosť **FactorialComputationFinished** a bude ukladať dvojice (číslo, číslo!).
- Komponenta bude poskytovať operácie:
  - **get(long number) : BigInteger**
  - **clear() : void**
- Nebude použitá thread-safe dátová štruktúra. Namiesto toho bude prístup k dátovej štruktúre chránený s pomocou read/write zámku (EJB)
- Tipy:
  - @Singleton
  - @Lock
- Riešenie v branch cdi-05

redhat

Úloha 6: Decorators

- Naimplementujte dekorátor ktorý bude obaľovať volanie **compute()** obidvoch implementácii **Factorial** rozhrania a bude vracať výsledky z cache v prípade, že budú dostupné. V opačnom prípade bude výpočet delegovaný na pôvodnú implementáciu.
- Tipy:
  - @Decorator
  - @Dependent
  - @Delegate
  - @Priority

redhat

Úloha 6: Decorators

- Naimplementujte dekorátor ktorý bude obaľovať volanie **compute()** obidvoch implementácii **Factorial** rozhrania a bude vracať výsledky z cache v prípade, že budú dostupné. V opačnom prípade bude výpočet delegovaný na pôvodnú implementáciu.
- Tipy:
  - @Decorator
  - @Dependent
  - @Delegate
  - @Priority
- Riešenie v branch cdi-06

redhat

Úloha 7: Conversations

- Zoznámte sa s triedami v balíku **cz.muni.fi.pv243.lesson02.students** a preskúmajte akým spôsobom je sú použité konverzácie na realizáciu procesu registrácie študentov.

jharting@redhat.com | www.redhat.com