2. domaća zadaća - JMX

1. Uvod

Java Management Extension, JMX arhitektura omogućava razvoj upravljivih Java aplikacija na standardiziran način. Upravljive aplikacije jesu takve aplikacije čije djelovanje je moguće pratiti i kontrolirati. Kontrola se najčešće odnosi na promjene stanja (vrijednosti atributa) objekata koji čine aplikaciju ili upravljanje životnim ciklusom objekata (uništavanje, suspendiranje, reaktiviranje ili kreiranje). Najjednostavniji način da se Java aplikacija učini upravljivom, a u skladu s JMX arhitekturom, jest da se definira standardni MBean (Standard MBean) objekt. Standardni MBean predstavlja upravljiva svojstva aplikacije ili neke njezine komponente i to preko upravljačkog sučelja koje MBean implementira. Upravljačko sučelje specificira: *atribute* (podatke) upravljive aplikacije čije vrijednosti je moguće pregledavati ili mijenjati te *operacije* koje je nad upravljivom aplikacijom moguće pokrenuti. Atributi i mogućnosti baratanja s njima (read/write) se u sučelju specificiraju indirektno preko tzv. getter i setter operacija. Za dodjelu naziva getter i setter operacija postoje pravila. (Proučite pravila.)

Upravljačka aplikacija pristupa MBean-ovima isključivo preko MBean servera. MBean objekti se moraju prijaviti pod određenim *nazivom* (Object Name) na MBean serveru. Naziv objekta jednoznačno identificira MBean objekt unutar jednog servera. Naziv objekta ima ovaj oblik:

```
NazivDomene:svojstvo1=vrijednost1 [,svojstvo2=vrijednost2 ]*
```

`NazivDomene`, `svojstvo` i `vrijednost` može biti bilo koji alfanumerički string, ali ne smije sadržavati sljedeće znakove: `, = : * ?`. Svi elementi naziva objekta su "case sensitive" tj. razlikuju se velika i mala slova u nazivu. Domena je apstraktna kategorija koja se proizvoljno koristi za grupiranje MBean-ova. Dizajner upravljanja sam odlučuje o nazivima domena unutar kojih će grupirati MBean objekte. Naziv domene je obvezan u nazivu objekta, tj. svaki MBean mora pripadati nekoj domeni. Unutar domene MBeans-i se razlikuju po barem jednom svojstvu koje moraju predstavljati jednoznačni ključ (key properties). To znači da nema dva ili više MBean-a koji imaju istu vrijednost svojstva koje predstavlja ključ. Osim navedenog ne postoje druga ograničenja obzirom na nazive svojstava, njihove vrijednosti i broj svojstava koja ima neki MBean.

2. Model upravljanja

Upravljiva komponenta koju dobivate kao primjer uz ovu DZ je proces čiju "beskonačnu" rutinu izvodi thread. Proces ima identifikator čija se vrijednost može pogledati, ali ne i mijenjati. Također, moguće je pogledati vrijeme pokretanja procesa. Thread koji je zadužen za izvođenje rutine procesa ima i identifikator čija vrijednost se može samo pogledati. Pogledati se može i trenutno stanje tog thread-a. (Moguća stanja Java threda jesu: NEW A thread that has not yet started is in this state. RUNNABLE A thread executing in the Java virtual machine is in this state. BLOCKED A thread that is blocked waiting for a monitor lock is in this state. WAITING A thread that is waiting indefinitely for another thread to

perform a particular action is in this state. TIMED_WAITING A thread that is waiting for another thread to perform an action for up to a specified waiting time is in this state. TERMINATED A thread that has exited is in this state.) Upravljani proces izvodi aktivnosti u beskonačnoj petlji, a iteracije se broje i vrijednosti spremaju u posebnu brojačku varijablu. Vrijednost brojača moguće je pogledati. U svakoj iteraciji izvodi se *CPU bound* aktivnost (Prouči pojmove CPU-bound i I/O bound na Wikipediji.) tijekom koje je thread u stanju RUNNABLE i *nonCPU bound* aktivnost koja je simulirana pomoću `sleep()` metode tijekom koje je thread u stanju TIMED_WAITING. Broj CPU bound operacija te vrijeme "spavanja" threada u svakoj iteraciji određuje se iznova slučajnim brojevima čije je maksimalne vrijednosti moguće pogledati i promijeniti. Za svaku iteraciju mjeri se: vrijeme zauzimanja CPU te ukupno vrijeme izvođenje iteracije. Napomena: tijekom izvođenja nonCPU bound aktivnosti ne troši se CPU vrijeme. Od spomenutih vremena stvaraju se kumulativne vrijednosti koje je moguće pogledati. Nad procesom je moguće je primijeniti operaciju za dobivanje srednje vrijednosti trajanja iteracija te operaciju za dobivanje vrijednosti zauzeća procesora. Također, nad procesom se može primijeniti operacija *reset* čiji je učinak postavljanje vrijednosti brojača i vrijednosti kumulativa trajanja iteracija na nulu. O ovoj promjeni vrijednosti atributa procesa mora se slati notifikacija koju primaju zainteresirani upravitelji.

3. Implementacija

Treba napomenuti da se često sama aplikacija ili njezina komponenta implementira kao standardni MBean. Dakle upravljiva funkcionalnost i njezina MBean reprezentacija predstavljaju cjelinu. To naravno nije obveza; aplikacije ili njezina komponenta te MBean komponenta koji predstavlja njezina upravljiva svojstva mogu se implementirati u različitim klasama. U primjeru u sklopu ove DZ koristi se prva opcija. Upravljivu komponentu predstavlja proces koji je implementiran kao klasa `Runnable` objekta čiju `run()` rutinu će izvršavati thread. Ista klasa predstavlja ujedno i standardni MBean, te objekt koji je sposoban emitirati notifikacije (obavijesti):

```
public class Proces extends NotificationBroadcasterSupport implements
ProcesMBean, Runnable {…}
```

Prije definiranja klase standardnog MBean objekta potrebno je specificirati upravljačko sučelje sa svim atributima tj. pripadnim getter i setter operacijama te ostalim upravljačkim operacijama. Obveza je, prema JMX standardu, da se naziv sučelja, kojeg će standardni MBean objekt implementirati, tvori od dva dijela:

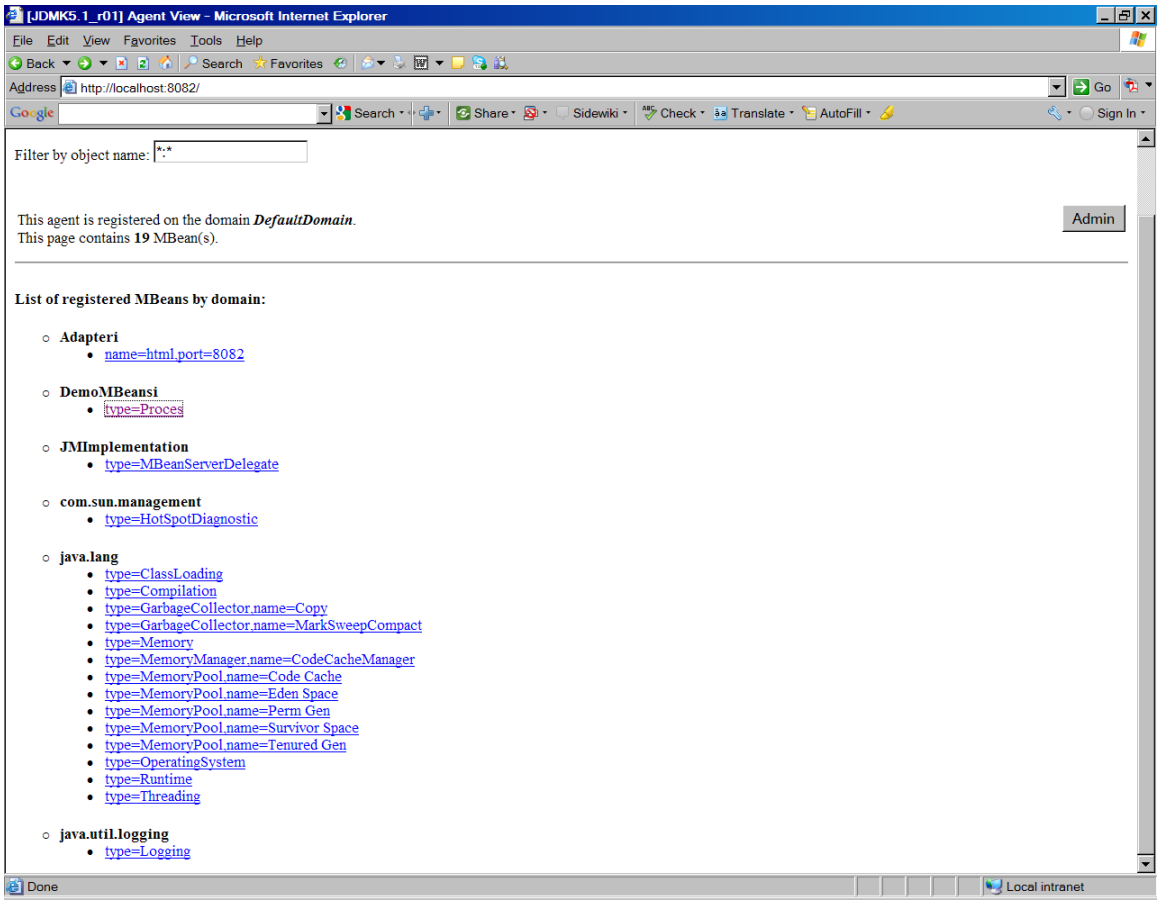
`nazivKlaseKojaPredstavljaStandardMBean+Mbean`.

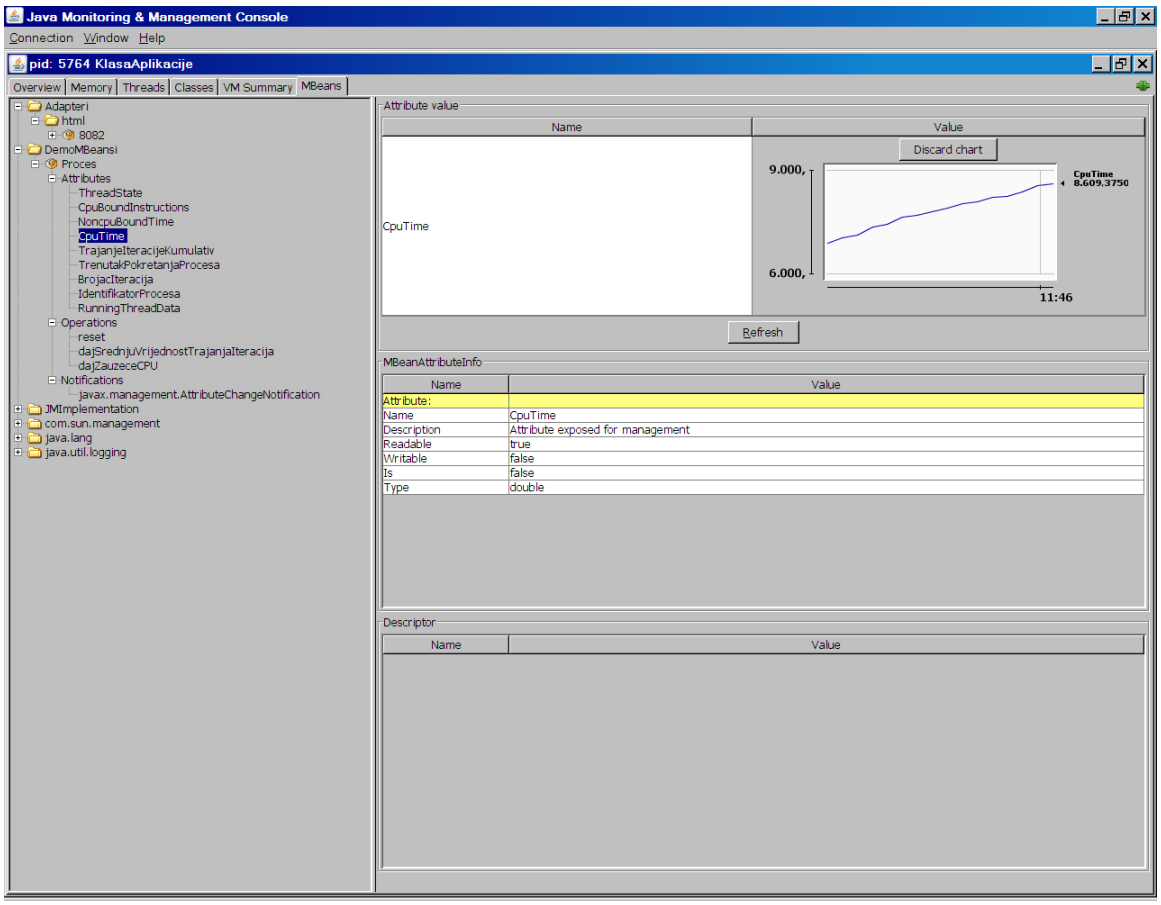
U našem primjeru upravljačko sučelje mora imati naziv: `ProcesMBean`. Također, MBean može slati notifikacije različitog tipa (Proučite tipove notifikacija.) upravljačkoj aplikaciji.

U primjeru demo aplikacije, na server su prijavljena dva MBeana (vidi main klasu `KlasaAplikacije`): Proces i HTML protokol adapter. HTML protokol adapter pruža pogled na MBean server i sve prijavljene MBean-ove preko osnovnog sučelja Web pretraživača. Tj. on osigurava HTTP komunikaciju između Mbean servera i vašeg Web pretraživača uz kreiranje odgovarajućih HTML stranica. Za pristup pokrećete Web pretraživač i upisujete adresu

http://localhost: 8082/ . Drugi način za pristup serveru je upotreba standardne JConsole upravljačke aplikacije (JDK\bin\jconsole.exe).

Zauzeće procesora možete pratiti i na Windows Task Manager-u ako ste primjer pokrenuli pod Windows-ima. Uz napomenu da vrijednost zauzeće dobiven od aplikacije odgovara onom u Task Manageru ako nema drugih procesa koji intenzivnije koriste CPU na vašem računalu te ako postoji samo jedan CPU.





4. Zadaća

- Vaša je zadatak da proučite: JMX arhitekturu (predavanje na tu temu biti će 28.04) te primjer demo aplikacije koji se nalazi u sklopu zadaće. (Napomena: kad pokrećete demo aplikaciju, u pokretačkim datotekama compile.bat i run.bat, prilagodite put do Java prevoditelja i Java interpretera u skladu s instalacijom JDK na vašem računalu.)
- Trebate osmisliti, projektirati i implementirati Java aplikaciju i pripadni model upravljanja kojeg treba dokumentirati u izvještaju (npr. tekstualno, kako je to napravljano u točki 2. za demo aplikaciju). **U osmišljavanju aplikacije imate potpunu slobodu i ona ne mora sličiti demo aplikaciji. Od max 7 bodova, 5 bodova će biti dodijeljeno za ideje i kreativnost.** Upravljački model treba sadržavati upravljive atribute (barem 2), operacije (barem 2) i notifikacije (barem 1).
- Razvijenu aplikaciju treba pokrenuti uz prijavljivanje njezinu(e) MBean komponentu na MBean server. Upravljivost treba isprobati uz pomoć Web pretraživača i JConsole te u izvještaj uključiti barem 2 "capture screanova" slično, kako je to napravljeno za demo aplikaciju u točki 3.
- Poslati pozipano/pojarano: izvještaj (.pdf), source datoteke (.java) aplikacije te dvije pokretačke (.bat) datoteke za prevođenje i pokretanje aplikacije na adresu mirko.randic@fer.hr.