Die Magie von MBeans und JMX

DOAG 2014

Andreas Chatziantoniou - Foxglove-IT BV

Bio – Andreas Chatziantoniou

- Freelance Oracle Fusion Middleware Consultant
- 16 Jahre Oracle Erfahrung/26 Jahre IT (Unix/C)
- Oracle ACE
- andreas@foxglove-it.nl

Agenda

- Übersicht MBeans
- JMX
- WebLogic und FMW
- Demo

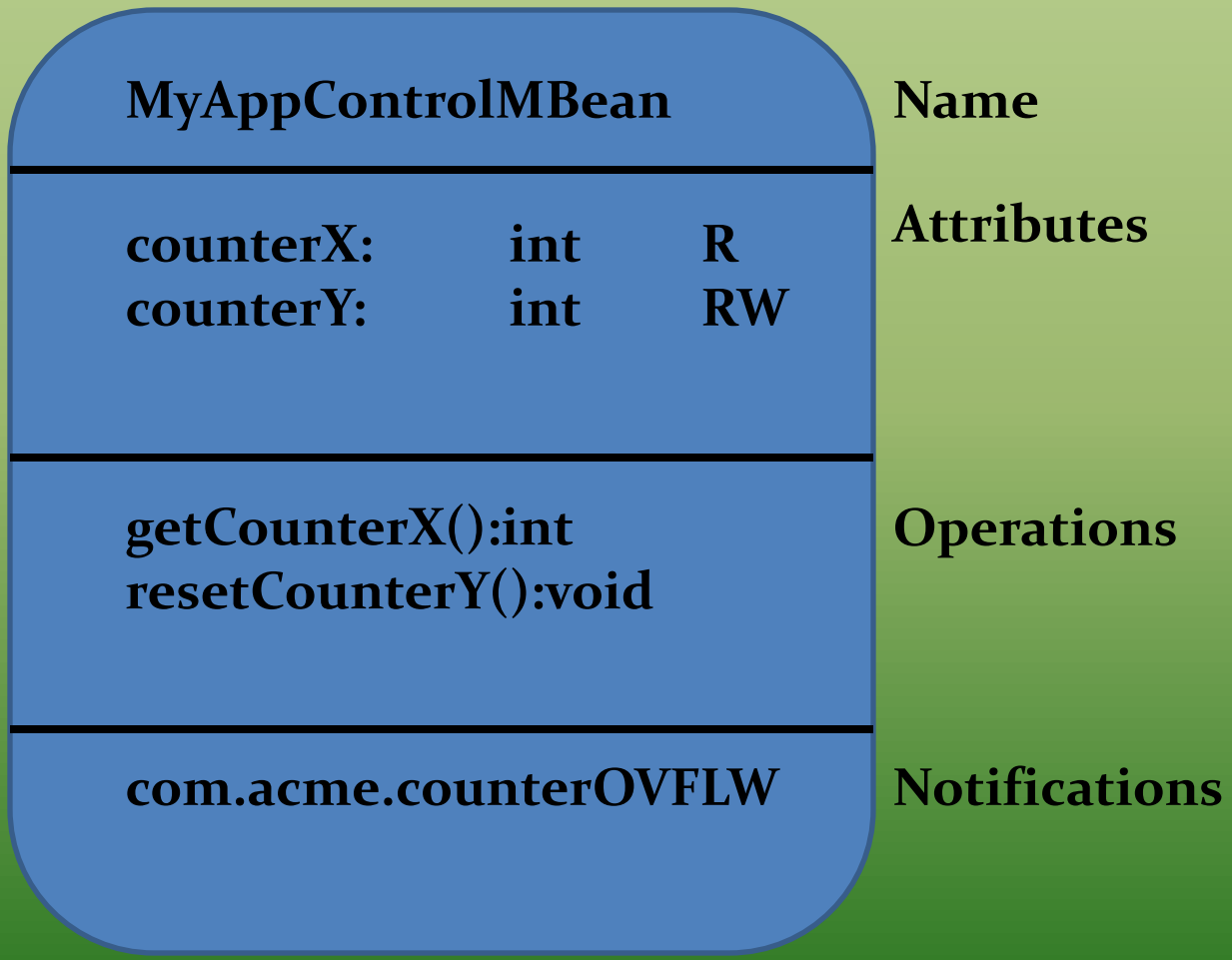
Übersicht MBeans?

- MBean steht für “Managed Bean”
- MBeans können Resources darstellen
  - Konfigurationsdaten einer Anwendung
  - Identität eines Benutzers
  - Apparate

Übersicht MBeans

- MBeans liefern Informationen über die dargestellten Resources
- Hierfür werden Attribute benutzt die gelesen oder beschrieben werden können
- Weiterhin können Operationen auf diesen Resources ausgeführt werden
- MBeans sind in der Lage um Benachrichtigungen (Notifications) zu versenden

Wie sehen MBeans aus?



Erzeugen MBeans

- MBeans können sehr einfach angelegt werden
- Java Interface MyAppControlMBean
  - Definieren von Attributen und Operationen
- Java Class MyAppControl
  - Implementierung des Interfaces

Implementierung MBean Interface

```
public interface MyAppControlMBean {
    public int getCounterX();
    public void resetCounterY();
    // more getters and setters
}
```

Implementierung MBeans Class

```
public  class MyAppControl implements MyAppControlMBean{
    private int counterX;
    private int counterY;

    public int getCounterX(){
        return this.counterX;
    }
    public void resetCounterY(){
        this.counterY = 0;
    }
...
}
```

This is a zero (funny lettertype)

MBean Namen

- Der Name einer MBean repräsentiert die MBean bzw. ein Musters das die Namen mehrerer MBeans angibt
- Syntax:
  - domain:<key-properties>=<value>
- Beispiel:
  - com.acme.myappcontrolmbean:key1=value1, key2=value2

MBean Server

- Der MBean Server ist das Herzstück des Agenten
- Hiermit können clients die Operationen einer MBean entdecken und ausführen
- Jede MBean registriert sich mit ihrem Namen

MBean Server

- Nur bei einem MBean Server registrierte MBeans können von außerhalb der JVM gemanaged werden
- Der MBean Server gibt nur das Management Interface einer MBean nach außen frei, nicht die direkte Referenz

MBean Code

- Connection zum MBean Server

```
InitialContext initCtx = new(InitialContext);
MBeanServer mbs = (MBeanServer) initCtx.lookup("jmx");
```

- Abfrage einer MBean

```
Set names = mbs.queryName(pattern,null);
```

- MBean verändern

```
mbs.setAttribute(name,new Attribute("counterX", 1);
```

Übersicht JMX

- JMX ermöglicht das Management von Java Anwendungen
  - Keine Anpassung des Application Design
  - Die Instrumentalisierung von JMX ist auf Mbeans basiert
- Kann auf verschiedenen Granularitätsebenen angeboten werden
  - Modul, Komponente, etc.

Übersicht JMX

- Das Ziel von JMX ist es um in laufenden Java Anwendungen
  - Objekte managen
    - Attribute lesen/schreiben
    - Operationen aufrufen
    - Benachrichtigungen empfangen
  - Java Objekte hinzufügen

Übersicht JMX

- Lokal oder remote
  - Normalerweise nur innerhalb der JVM
- Connector sorgt für den Zugriff von RMI Clients
- Adaptor sorgt für den Zugriff über Protokolle (HTTP, SNMP)

Übersicht JMX

- JMX kann weiterhin zur Automatisierung von WebLogic Konfigurationen eingesetzt werden
- Das gilt auch für das Monitoring

JMX Agent

- Ein JMX Agent ist ein container für einen MBean Server
- Der JMX Agent bietet (u.a) die folgenden Services um MBeans zu managen:
  - Monitors: Beobachte MBean Attribute und sende eine Notification wenn sich der Wert ändert
  - Timers: Sende eine Benutzer-definiert Notification zu bestimmten Zeiten

Andere Möglichkeiten zum Auslesen von MBeans

- WLST
  - Im Online-Modus können MBeans gelesen werden
  - Ideal für das Arbeiten mit Server/Domain MBeans
  - Sinnvoll als ad-hoc Management Tool
  - MBeans haben eine Baumstruktur
    - Kommandos wie in Unix `cd(); ls();`

Mbeans Bäume

- Die folgenden MBean Bäume sind vorhanden:
  - domainConfig
    - Gesamte Domäne
  - serverConfig
    - MBean des Servers mit dem die Connection besteht
  - edit
    - Gesamte Domäne zum Ändern der Konfiguration
  - jndi
    - JNDI Baum
  - custom
    - Selbstdefinierte MBeans

Weblogic und FMW

- Oft besteht die Notwendigkeit um in einer WLS oder FMW Umgebung extra Daten zu erfassen
- Hier gibt es die folgenden Use-Cases
  - Schnell (bestimmte) Performance Data auslesen
  - Vergleichen von Konfigurationen
  - Aufbauen von Umgebungen

Weblogic und FMW

- Die FMW Console ist ein nützliches Instrument um eine Übersicht zu erhalten, aber für ein Monitoring Tool manchmal nur eingeschränkt nutzbar
- Bei der Vermutung, dass bestimmte Komponenten Performanceprobleme verursachen kann ein kleines JMX Programm schnell diese Daten liefern

Weblogic und FMW

- Sobald in FMW Umgebungen eigener Code (z.B. BPEL Prozess) deployed ist wird die Frage nach MBeans Daten grösser da wir wissen wollen was in der Anwendung passiert
- Daher sollten eigene Anwendungen soviel wie möglich eigene MBeans haben

Demo

- Demo einer einfachen JMX Anwendung um einen Wert aus zu lesen
- Demo des Drill-Down um eine Mbean im JDBC Bereich anzuzeigen

Q?

A!