



JBoss AS advanced - Microcontainer  
Genera Lynx d.o.o.

Aleš Justin  
ales.justin@genera-lynx.com




## Overall agenda

- Contextual injection as a buzzword
- JBoss Microcontainer intro
- Microcontainer advanced
- Conceptual view of the state machine
- Achieving contextual injection in Microcontainer
- Road ahead



## Contextual injection as a buzzword

- What is contextual injection
  - aka autowiring
  - advantages
  - drawbacks
- Usage
  - by type
  - by name
- Example




## Contextual injection example

- Microcontainer
 


```
<bean name="testObject1" class="org.jboss.test.kernel.inject.support.PropertyInjectTestObject">
  <property name="testerInterface"><inject/></property>
</bean>
```
- Springframework
 

```
<bean
  id="testObject1"
  class="org.jboss.test.kernel.inject.support.PropertyInjectTestObject"
  autowiring="byType"
/>
```



## JBoss Microcontainer intro

- JBoss AS as JMX kernel
  - MC kernel replacing JMX heavyweight MBeanServer
  - POJOs replacing MBeans
  - standalone usage; outside AS
- Microcontainer at first glance
  - IoC container
  - state machine
  - improved lifecycle management
    - lifecycle methods invocation
    - mode
    - state
  - additional dependencies
    - demand
    - supply




## JBoss Microcontainer intro

- Support for other JEMS projects
  - JBoss EJB3 standalone
  - JBoss Seam
  - JBoss WS
  - ...

```
<bean name="ServiceEndpointPublisher"
  class="org.jboss.ws.integration.jboss50.ServiceEndpointPublisher">
  <property name="mainDeployer"><inject bean="MainDeployer"/></property>
  <property name="serviceEndpointServlet">
    org.jboss.ws.integration.jbossall.ServiceEndpointServlet
  </property>
</bean>
```

```
<bean name="KernelLocator" class="org.jboss.ws.server.KernelLocator">
  <!--property name="kernel"-->
  <!--inject bean="jboss.kernel.service=Kernel"/>
</property-->
</bean>
```



## Microcontainer advanced

- Using other JBoss projects
  - Common
  - XB
  - AOP
  - Test
  - ...
- Microcontainer as division of concerns
  - clean separation of contract and implementation
    - spi
    - plugin
  - object instance lifecycle
    - metadata; xb, javabean
    - configuration; java.lang.\* abstraction
    - instantiation; annotations, aop

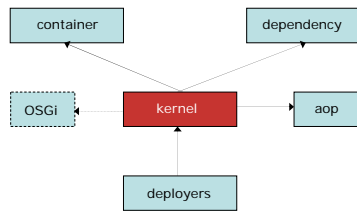


## Microcontainer modules

- Project modularity
  - container
  - dependency
  - kernel
  - aop-int
  - deployers
  - OSGi-int
- Extensive testing
  - JUnit
  - JBoss Test



## Microcontainer modules dependency



## Microcontainer and XB

- Combining XSD with Java
  - XSD as a strict XML rules; single point lookup
  - SingletonSchemaResolverFactory bind to Java

### • Binding example

```

<xsd:complexType name="InjectionType">
public static final QName injectionTypeName = new QName(BEAN_DEPLOYER_NS, "InjectionType");
TypeBinding injectionType = schemaBinding.getType(injectionTypeName);
InjectionType.setHandler(InjectionHandler.HANDLER);

public class InjectionHandler extends DefaultElementHandler
{
    public Object startElement(Object parent, QName name, ElementBinding element)
    {
        AbstractInjectionValueMetaData vmd = new AbstractInjectionValueMetaData();
        if (parent instanceof AbstractPropertyMetaData)
        {
            AbstractPropertyMetaData x = (AbstractPropertyMetaData) parent;
            vmd.setPropertyMetaData(x);
        }
        return vmd;
    }
}
    
```



## Microcontainer binding code

```

<xsd:complexType name="InjectionType">
<xsd:attribute name="bean" type="xsd:string" use="optional"/>
<xsd:attribute name="property" type="xsd:string" use="optional"/>
<xsd:attribute name="state" type="controllerStateType" use="optional"/>
<xsd:attribute name="whenRequired" type="controllerStateType" use="optional"/>
<xsd:attribute name="type" type="InjectionTypeType" use="optional" default="ByClass"/>
</xsd:complexType>

public void attributes(Object o, QName elementName,
    ElementBinding element, Attributes attrs, NamespaceContext nsCtx)
{
    AbstractInjectionValueMetaData injection = (AbstractInjectionValueMetaData) o;
    for (int i = 0; i < attrs.getLength(); ++i)
    {
        String localName = attrs.getLocalName(i);
        if ("bean".equals(localName))
            injection.setValue(attrs.getValue(i));
        else if ("property".equals(localName))
            injection.setProperty(attrs.getValue(i));
        else if ("state".equals(localName))
            injection.setDependentState(new ControllerState(attrs.getValue(i)));
        else if ("whenRequired".equals(localName))
            injection.setWhenRequiredState(new ControllerState(attrs.getValue(i)));
        else if ("type".equals(localName))
            injection.setInjectionType(new InjectionType(attrs.getValue(i)));
    }
}
    
```



## Bean info abstraction

- java.lang.\* abstraction
  - Javassist
  - Introspection / reflection
- goals
  - lazy loading
  - hot deployment
  - aspectization
  - JDK independency
  - ...





## Class dependency context lookup

- Plain installed context lookup
  - result resolves in dependency item resolution
- KernelRegistry entry lookup
  - KernelRegistryPlugin entry lookup
    - AbstractKernelController entry lookup
      - AbstractKernelController.getContextByClass

```
public KernelControllerContext getContextByClass(Class clazz)
{
    Set<KernelControllerContext> contexts = getInstalledContexts(clazz);
    int numberOfMatchingBeans = 0;
    if (contexts != null)
    {
        numberOfMatchingBeans = contexts.size();
    }
    if (log.isDebugEnabled())
    {
        log.trace("Checking for contextual injection, current matches: " + numberOfMatchingBeans + " - " + clazz);
    }
    if (numberOfMatchingBeans != 1)
    {
        if (numberOfMatchingBeans > 1)
        {
            log.warn("Multiple beans match class type: " + clazz);
        }
        return null;
    }
    return contexts.iterator().next();
}
```



## Microcontainer – the road ahead

- Contextual injection
  - currently XML support
  - annotation support
  - 'all you can wire' support
- General Microcontainer road map
  - ...



## Questions?

