Java EE JNDI ENC 注入

[Java EE]

・アプリケーションサーバーにデプロイされたすべての EJB コンテナは、Enterprise Naming Context(ENC) と呼ばれる独自の内部レジストリを持っている
・これは JNDI で実装されている

グローバル JNDI

・EJB3.1 では SLSB、SFSB のビューは以下の構文のグローバル JNDI で入手出来る必要がある

```
java:global[/app-name]/module-name/bean-name [!FQN]
```

項目	内容
app-name	アプリケーション (または EAR) 名 (オプション)
module-name	モジュール (JAR または WAR) 名
FQN	完全修飾インターフェース名

利点

・ベンダにかかわらず、同じ場所で EJB を見つけることができ移植性がある

改善できる点

・ルックアップコードはベンダー固有の JNDI Context の取得に依存
・キャストが必要で、タイプセーフではない
・自分で JNDI 名を作成するため、間違えやすい

EJB コンテナは一連のタイプセーフな注入メカニズムを提供

・多くの場合、利用可能な参照を取得するのに必要なのは以下だけ

```
@EJB
MyEJBLocalBusiness bean;
```

JNDI ENC

・EJB3.x では、ENC が強化され、JNDI ENC 参照を Bean クラスのフィールドに直接注入できるようになった
・このためにアノテーションが主に利用されるがm, XML デプロイメント記述子も利用可能

EJB、インターフェースへの参照、JMS キューまたはトピックの送信先、JMS 接続ファクトリ、データソース、JCA リソース、プリミティブ型など、様々なものを ENC に登録できる

JNDI ENC への投入方法

・2 つの異なる方法で設定できる

■ XML による投入

・ejb-local-ref:MyEJB が MyEJB2 のローカルビジネスインターフェースへの参照を必要としていることを EJB コンテナに通知
・ejbs/referenceToMyEJB2 という名前で JNDI ENC に登録

```
<ejb-jar>
  <enterprise-bean>
    <session>
      <ejb-name>MyEJB</ejb-name>
      <ejb-local-ref>
        <ejb-ref-name>ejbs/referenceToMyEJB2</ejb-ref-name>
        <ejb-ref-type>Session</ejb-ref-type>
        <local>org.ejb3book.example.MyEJB2LocalBusiness</local>
        <ejb-link>MyEJB2</ejb-link>
      </ejb-local-ref>
    </session>
  </enterprise-bean>
</ejb-jar>
```

■アノテーションによる設定

・アノテーションで定義された情報で JNDI ENC に投入

```
@Stateful(name="MyEJB")
@EJB(name="ejbs/referenceToMyEJB2",beanInterface=MyEJB2LocalBusiness.class,beanName="MyEBJ2")
public class MyEJBBean implements MyEJBLocalBusiness {
  :
}
```

ENC からの参照方法

・JNDI ENC に登録したものはすべて、java:comp/env コンテキストから名前で検索できます
・comp は component を表している

```
try {
  javax.naming.InitialContext ctx = new InitialContext();
  bean = (MyEJB2LocalBusiness)ctx.lookup("java:comp/env/ejbs/referenceToMyEJB2");
 } catch(javax.naming.NamingException ne) {
   :
 }
```

■ EJBContext

・EJBContext には ENC ルックアップメソッドがあり、チェック例外を発生させず、相対名を使う
・SessionContext、MessageDrivenContext いずれも EJBContext を拡張

```
@Resource
private javax.ejb.SessionContext ctx;

public void hoge() {
  MyEJBLocalBusiness bean = ctx.lookup("ejbs/referenceToMyEJB2");
}
```

■アノテーションによる注入

・ENC ルックアップの代わりに、EJB 参照をメンバ変数に直接注入できる

```
@EJB
private MyEJB2LocalBusiness bean;
```

・セッターメソッドを使った注入もサポート
・フィールドに直接注入するより冗長だが、単体テストで簡単にモックできるというメリット

```
@EJB
public void setBean(final MyEJB2LocalBusiness bean) {
  this.bean = bean;
}
```

■デフォルト ENC 名

・Bean クラスのフィールドやセッターメソッドにアノテーションをつけると、注入された要素のために JNDI ENC にエントリーを作成することにもなる
・これらはすべての環境アノテーションで起こるが、@EBJ では、注入アノテーションの name() 属性が指定されていると、参照はその名前で ENC に格納される
・名前が指定されない場合、ENC 名はアノテーションづけされたフィールドやメソッドの完全修飾クラス名とフィールド名やメソッドのベース名から付けられる

例

```
org.ejb3book.example.MyEJBBean/otherBean
```

EJB 参照 (上記例は以下で検索できる)

```
java:comp/env/org.ejb3book.example.MyEJBBean/otherBean
```

■ XML による注入

・フィールドの初期化にアノテーションを利用したくない場合、ejb-jar.xml デプロイメント記述子で <jnjection-target> を利用できる

```
<ejb-jar>
  <enterprise-bean>
    <session>
      <ejb-name>MyEJB</ejb-name>
      <ejb-local-ref>
        <ejb-ref-name>ejbs/referenceToMyEJB2</ejb-ref-name>
        <ejb-ref-type>Session</ejb-ref-type>
        <local>org.ejb3book.example.MyEJB2LocalBusiness</local>
        <ejb-link>MyEJB2</ejb-link>
        <injection-target>
          <injection-target-class>
            org.ejb3book.example.MyEJBBean
          </injection-target-class>
          <injection-target-name>otherBean</injection-target-name>
        </injection-target>
      </ejb-local-ref>
    </session>
  </enterprise-bean>
</ejb-jar>
```

■ XML によるオーバーライド

・注入アノテーションを使うと、Bean クラスのコードに構成をハードコーディングすることになるとみなされる場合もある
・EJB 仕様では XML デプロイメント記述子を使って注入アノテーションをオーバーライドできる

XML は常にアノテーションメタデータよりも優先される。XML はハードコーディングされたアノテーションを再構成する手段を提供する

■注入と継承

・Bean クラスをクラス階層に含めることができる
・注入アノテーションは特定の注入規則に従う

```
public class Base {
  private SomeInf bean;

  @EJB(beanName="SomeEJB")
  public void someMethod(SomeInf bean) {
    this.bean = bean;
  }
}


@Stateless
public class MySessionBean extends Base implements MySessionLocalBusiness {
}
```

・上記例で、Base を継承する ステートレスセッション Bean は、Base クラスの someMethod に適切なリソースを注入する。

```
@Stateless
public class MySessionBean extends Base implements MySessionLocalBusiness {
  private SomeInf bean;
  @EJB(beanName="AnotherEJB")
  public void someMethod(SomeInf bean) {
    this.bean = bean;
  }
}
```

・このようにすると、SomeEJB ではなく、AnotherEJB が注入される

someMethod が private の場合、Base には SomeEJB が注入される

参照の注入と型