

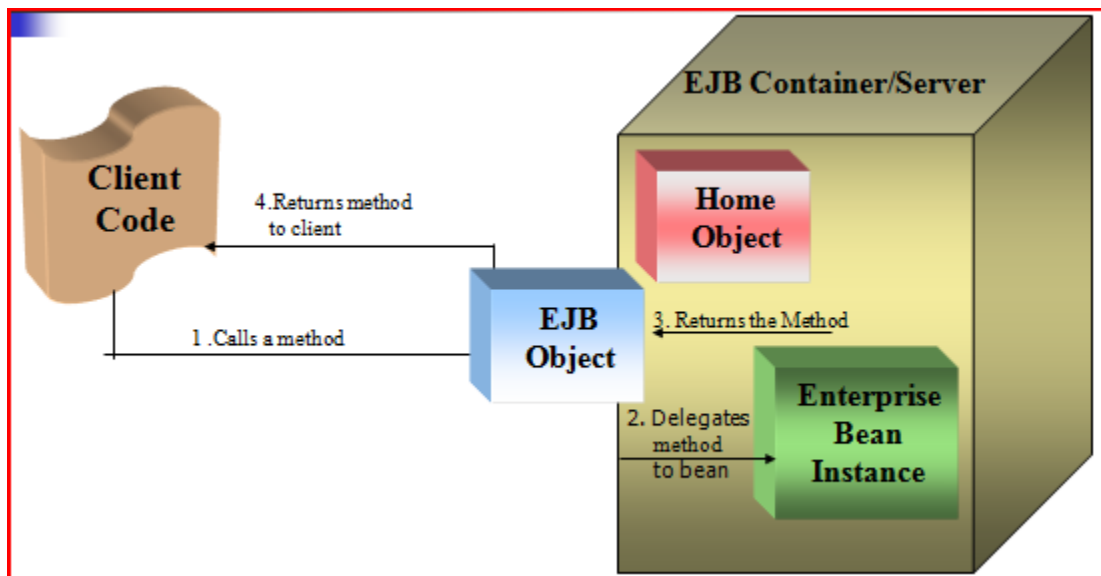
STATELESS SESSION BEAN – EJB2

VoVanHai's blog

I. THIẾT KẾ

1. Tạo remote interface

- Remote interface định nghĩa tất cả các business method của EJB, không chứa các tác vụ cấp hệ thống (persistence, security, transaction, ...). Các business method này sẽ được cài đặt trong lớp implements.



- Remote interface cần phải:

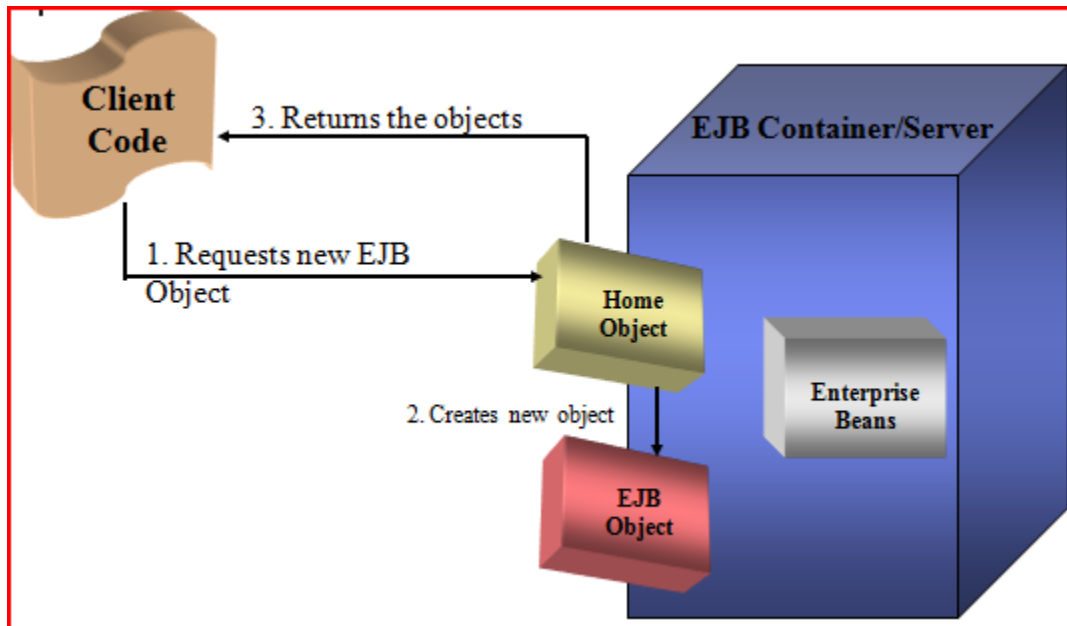
- Có visibility modifier là **public** (để có thể triệu gọi từ xa)
- Thừa kế interface **javax.ejb.EJBObject**.
- Các phương thức trừu tượng ở đây phải throws **java.rmi.RemoteException**.

Ví dụ ở đây ta tạo 1 phương thức đặc tả nghiệp vụ thao tác với các số dựa trên operator được cung cấp.

```
package calc;
import java.rmi.RemoteException;
import javax.ejb.EJBObject;
public interface CalcRemote extends EJBObject{
    public double DoCalculate(double a, double b,
                                char operator) throws RemoteException;
}
```

2. Tạo home interface

- Home interface hoạt động như một factory, định nghĩa các phương thức cho phép client tạo và tìm các đối tượng EJB.



- Home interface cần phải:

- Import các giao diện: **java.io.Serializable**, **java.rmi.RemoteException**, **javax.ejb.CreateException** và **javax.ejb.EJBHome**.
- Phải thừa kế interface **javax.ejb.EJBHome**.
- Có phương thức **create()** throws các exception: **RemoteException** và **CreateException** (Có thể có các exception khác của lập trình viên) và trả về một đối tượng có kiểu remote interface.

```
package calc;
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;
public interface CalcHome extends EJBHome{
    CalcRemote create() throws RemoteException, CreateException;
}
```

3. Tạo lớp EJB

- Lớp EJB cài đặt:

- Tất cả các business method khai báo trong remote interface.
- Các phương thức dùng cho container (phương thức callback).

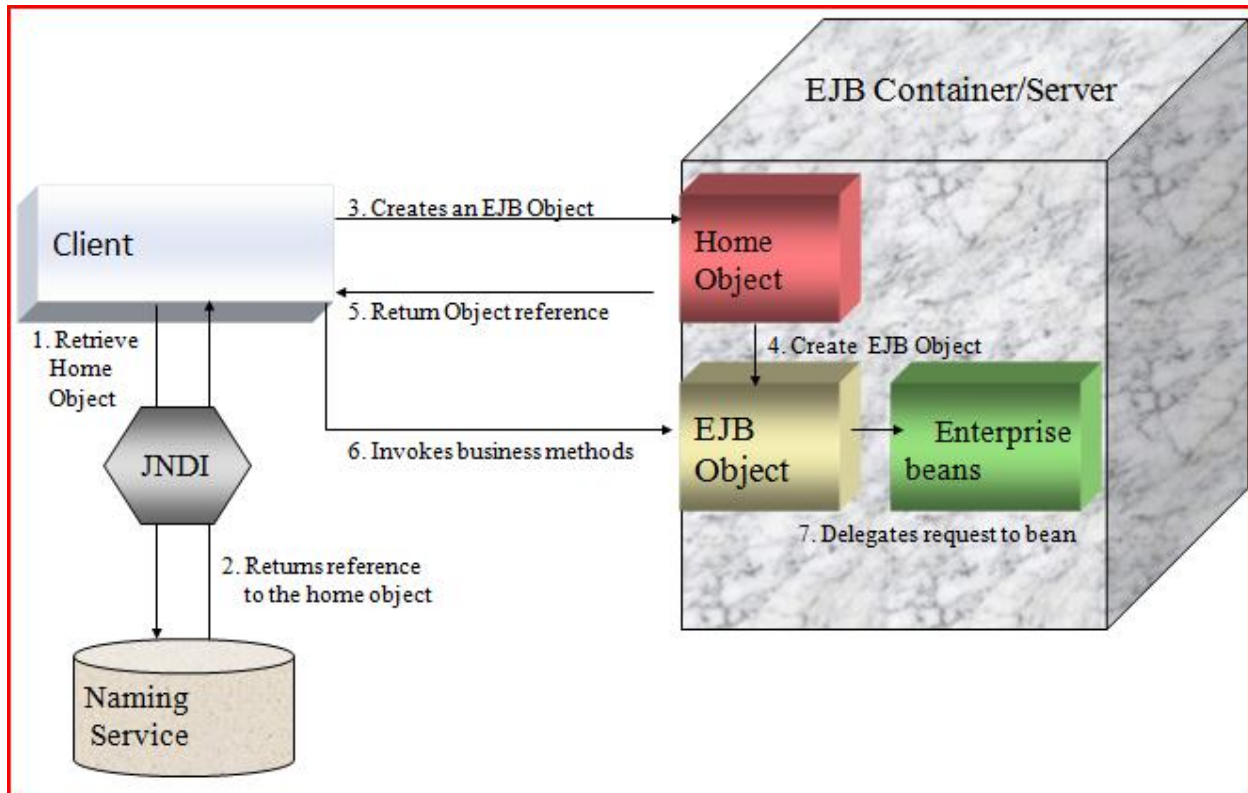
- Lớp EJB cần phải:

- Lớp EJB phải thừa kế interface **javax.ejb.SessionBean**.

```
package calc;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
public class CalcBean implements SessionBean {
    private SessionContext context;
    public void setSessionContext(SessionContext aContext) {
        context = aContext;
    }
    public void ejbActivate() {}
    public void ejbPassivate() {}
    public void ejbRemove() {}
    public void ejbCreate() {}
    public double DoCalculate(double a, double b, char operator){
        double x = 0;
        switch (operator) {
            case '+': x = a + b; break;
            case '-': x = a - b; break;
            case '*': x = a * b; break;
            case '/':
                if (b == 0) {
                    System.out.println("Lỗi chia cho zero");
                    b=1000000000;
                }
                x = a / b; break;
        }
        return x;
    }
}
```

4. Tạo client truy xuất Stateless Session Bean

- Mô hình làm việc tổng quát:



a) Định vị home interface

- Client phải dùng JNDI thông qua Naming Service để định vị một đối tượng home cụ thể.
- Thiết lập thuộc tính môi trường (Initial Context Factory tạo initial context với cấu trúc thư mục JNDI dùng, vị trí server cung cấp dịch vụ Naming, ...).

```
java.util.Properties props = System.getProperties();
props.put(Context.INITIAL_CONTEXT_FACTORY, "org.jnp.interfaces.NamingContextFactory");
props.put(Context.PROVIDER_URL, "127.0.0.1:1099");
props.put(Context.URL_PKG_PREFIXES, "org.jboss.naming");
```

- Tạo JNDI naming context như một giao diện giữa client và JNDI.

```
Context ctx = new InitialContext();
```

b) Tìm đối tượng thông qua JNDI

- Sau khi tạo JNDI context, phương thức **lookup()** của đối tượng lớp **InitialContext** được sử dụng để định vị đối tượng có tên JNDI chỉ định trong tập tin **jboss.xml**.

```
Object obj = ctx.lookup("calc/Calculator");
```

c) Thu hẹp (narrow) tham chiếu thành một đối tượng

- Đối tượng trả về bởi phương thức **lookup()** phải được ép thành kiểu home interface. Đối tượng này giữ một tham chiếu đến home interface. Điều này được thực hiện bởi phương thức **PortableRemoteObject.narrow()**, có hai tham số: đối tượng do **lookup()** trả về và tên file class của home interface. Lớp **PortableRemoteObject** được dùng thay cho lớp **UnicastRemoteObject** trong RMI để bảo đảm tính tương thích với các giao thức khác JRMP, ví dụ RMI/IIOP.

```
CalcHome home = (CalcHome)PortableRemoteObject.narrow(obj,CalcHome.class);
```

d) Sinh ra một thực thể EJB

- EJB sẽ triệu gọi phương thức **create()** của đối tượng home để trả về đối tượng EJB (đối tượng remote interface).

```
CalcRemote calc = home.create();
```

e) Triệu gọi các business method (phương thức nghiệp vụ)

- Remote interface định nghĩa các phương thức nghiệp vụ thực hiện trong lớp EJB, client sẽ triệu gọi các phương thức này thông qua đối tượng EJB vừa được phương thức **create()** trả về.

```
double r=calc.DoCalculate(6,7,'+');
```

f) Code hoàn chỉnh

```
import java.util.Properties; import javax.naming.Context; import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject; import calc.*;
public class Client {
    public static void main(String[] args) {
        try {
            Properties props = System.getProperties();
            props.put(Context.INITIAL_CONTEXT_FACTORY, "org.jnp.interfaces.NamingContextFactory");
            props.put(Context.PROVIDER_URL, "127.0.0.1:1099");
            props.put(Context.URL_PKG_PREFIXES, "org.jboss.naming");
            Context ctx = new InitialContext();
            Object obj = ctx.lookup("calc/Calculator");
            CalcHome home = (CalcHome)PortableRemoteObject.narrow(obj,CalcHome.class);
            CalcRemote calc = home.create();
            double r = calc.DoCalculate(6,7,'+');
            System.out.println("====="); System.out.println(r);
            System.out.println("=====");
        } catch (Exception ex) { ex.printStackTrace();}
    }
}
```

II. TRIỂN KHAI

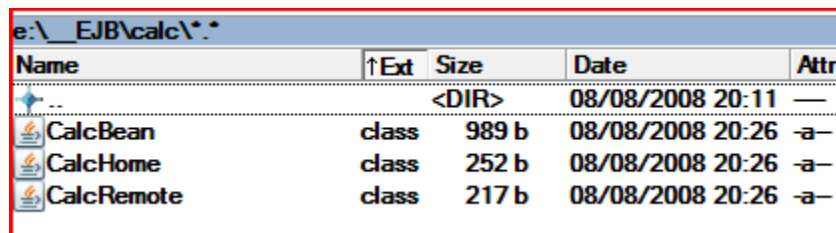
- Cần có gói sau trong **CLASSPATH**, hoặc tham chiếu đến gói này trong IDE cụ thể (các gói này đều nằm trong thư mục client của thư mục cài đặt Jboss)

- jnp-client.jar
- jboss-common-client.jar
- concurrent.jar
- jboss-client.jar
- jboss-serialization.jar
- jboss-remoting.jar
- jbosssx-client.jar
- jboss-transaction-client.jar

1. Biên dịch các file java

- Trong command-line, gõ: **javac *.java**
- Chú ý: Đảm bảo việc biên dịch không xảy ra lỗi nào.

2. Tạo thư mục calc, copy các file class mới biên được vào đây:



Name	Ext	Size	Date	Attr
..	<DIR>		08/08/2008 20:11	—
CalcBean	class	989 b	08/08/2008 20:26	-a-
CalcHome	class	252 b	08/08/2008 20:26	-a-
CalcRemote	class	217 b	08/08/2008 20:26	-a-

3. Tạo thư mục META-INF

- Tạo các file deployment descriptor trong thư mục này:

a. File **ejb-jar.xml** có nội dung

```
<?xml version="1.0" encoding="UTF-8" ?>
<ejb-jar version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
    http://java.sun.com/xml/ns/j2ee/ejb-jar_2_1.xsd">
  <enterprise-beans>
```

```

<session>
  <ejb-name>Calculator</ejb-name>
  <home>calc.CalcHome</home>
  <remote>calc.CalcRemote</remote>
  <ejb-class>calc.CalcBean</ejb-class>
  <session-type>Stateless</session-type>
  <transaction-type>Container</transaction-type>
</session>
</enterprise-beans>
</ejb-jar>

```

b. File jboss.xml có nội dung

```

<?xml version="1.0" encoding="UTF-8" ?>
<jboss>
  <enterprise-beans>
    <session>
      <ejb-name>Calculator</ejb-name>
      <jndi-name>calc/Calculator</jndi-name>
    </session>
  </enterprise-beans>
</jboss>

```

c. Tạo File MANIFEST.MF

```

Manifest-Version: 1.0
Created-By: 1.6.0 (Sun Microsystems Inc.)

```

4. Đóng gói ứng dụng EJB

- Dùng tool jar được cung cấp sẵn của jdk đóng gói ứng dụng

jar cvf calculator.jar calc/*.class META-INF/*.xml

```

E:\_EJB>jar cvf calculator.jar calc/*.class META-INF/*.xml
added manifest
adding: calc/CalcBean.class(in = 989) (out= 570)(deflated 42%)
adding: calc/CalcHome.class(in = 252) (out= 184)(deflated 26%)
adding: calc/CalcRemote.class(in = 217) (out= 169)(deflated 22

```

```
adding: META-INF/ejb-jar.xml(in = 580) (out= 293)(deflated 49%)
adding: META-INF/jboss.xml(in = 223) (out= 135)(deflated 39%)
```

Ta có thể dùng winrar để xem nội dung gói **calculator.jar** có cấu trúc như sau:

```
calc
CalcBean.class
CalcHome.class
CalcRemote.class
META-INF
  ejb-jar.xml
  jboss.xml
MANIFEST.MF
```

5. Chạy JBoss server

- Chạy %JBOSS_HOME%\bin\run.bat trong một console.

,897 INFO [Http11Protocol] Starting Coyote HTTP/1.1 on http-127.0.0.1-8080

```
22:10:16,928 INFO [AjpProtocol] Starting Coyote AJP/1.3 on ajp-127.0.0.1-8009
22:10:16,990 INFO [Server] JBoss (MX MicroKernel) [4.2.2.GA (build:
SVNTag=JBoss_4_2_2_GA date=200710221139)] Started in 22s:869ms
```

6. Triển khai gói calculator.jar

- Sao chép gói **calculator.jar** vào thư mục %JBOSS_HOME%\server\default\deploy\ ,
lập tức thấy chi tiết triển khai gói này trong console chạy JBoss server, đây là khả năng hot
deployment của JBoss.

```
...
22:14:00,492 INFO [AjpProtocol] Starting Coyote AJP/1.3 on ajp-127.0.0.1-8009
22:14:00,523 INFO [Server] JBoss (MX MicroKernel) [4.2.2.GA (build:
SVNTag=JBoss_4_2_2_GA date=200710221139)] Started in 21s:637ms
22:15:00,988 INFO [EjbModule] Deploying Calculator
22:15:01,066 INFO [ProxyFactory] Bound EJB Home 'Calculator' to jndi 'calc/Calculator'
22:15:01,066 INFO [EJBDeployer] Deployed: file:/C:/javaSoft/jboss-
4.2.2.GA/server/default/deploy/calculator.jar
```

7. Sử dụng client để truy xuất EJB

- Tạo thư mục client, copy tập tin client.class vào đấy.
- Tạo thư mục calc trong thư mục client, copy 2 tập tin CalcHome.class, CalcRemote.class
vào đấy.

- Mở cửa sổ command-line, đưa thư mục hiện hành về thư mục client ở trên, chạy lệnh: **java client**.

- Kết quả như sau:

```
Microsoft Windows [Version 6.0.6000]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.
E:\_EJB\client>java client

=====

13.0

=====

E:\_EJB\client>
```