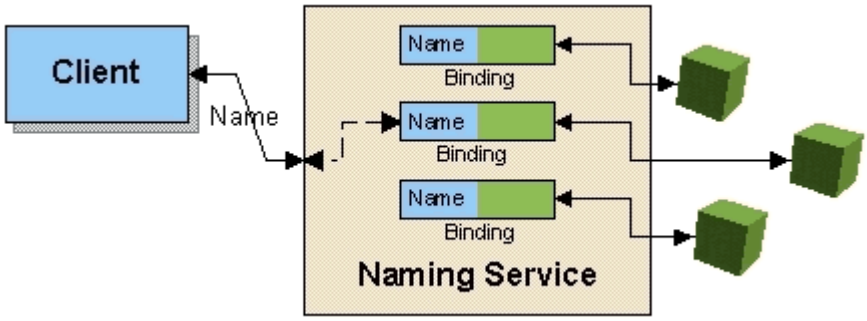
Naming Service

Summary

This service is a tool that helps to find the resource using Java Naming and Directory Interface(JNDI) API. It registers and exposesresources at the Naming server supporting Naming service so that the resources can be used by other applications and can be used by locating the resources registered in the Naming server.



Main Concept

Java Naming and Directory Interface(JNDI)

Java Naming and Directory Interface(JNDI) is a kind of Java API for the directory service that helps to find the data and object using Java software client name.

- **Refer tohttp://en.wikipedia.org/wiki/JNDI.**

Description

How to use Naming service is separated into the type of setting to Spring XML Configuration file and the type of using JndiTemplate class wrapping JNDI API.

- Type of setting to Spring XML Configuration file :
  Type of registering JNDI object in the Spring XML Configuration setting file as bean. Look up JNDI object only. Most commonly used.

- Type of using JndiTemplate class wrapping JNDI API :
  Type of directly using JndiTemplate class provided for easy use of JNDI API at Spring Framework.This type is used if all JNDI API functions should be used.

Spring XML Configuration Setting

The Spring Framework can set the JNDI object to XML Configuration file. Provided, however, that since JNDI object can be lookup through setting file, Using JndiTemplate type should be used if you want to use bind, rebind, unbind function.

The Spring Framework provides jee tags from 2.0 version for simple XML Configuration. Since the e-government development framework is based on Spring 2.5 or over, this guide explains the type that uses jee tag only.

Setting

To use jee tag, namespace and schemaLocation should be added to the head of Spring XML Configuration file.

- namespace: xmlns:jee="http://www.springframework.org/schema/jee"

- schemaLocation : http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-2.5.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:jee="http://www.springframework.org/schema/jee"
xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-2.5.xsd">

<!--<bean/> definitions here -->

</beans>
```

jndi-lookup tag

jndi-lookup tag is the tag that finds JNDI object and register as bean.

tagDescription

```
<jee:jndi-lookup id="bean id"
jndi-name="jndi name"
cache="true or false"
resource-ref="true or false"
lookup-on-startup="true or false"
expected-type="java class"
proxy-interface="java class">
<jee:environment>
name=value
ping=pong
            ...
</jee:environment>
</jee:jndi-lookup>
```

The jndi-lookup tag is mapped to JndiObjectFactoryBean class of Spring Framework at the ratio of 1:1. Attribute value of the tag is as follows:

Attribute	Description	Optional	Data Type	Default Value	Remarks
id	bean id of Spring XML Configuration.	N	String		
jndi-name	Name of JNDI object to search.	N	String		
cache	Indicate the cache for JNDI object searched once.	Y	boolean	true	
resource-ref	Indicate whether to search in J2EE Container or not.	Y	boolean	false	
lookup-on-startup	Indicate whether to perform lookup or not at the startup.	Y	boolean	true	
expected-type	Indicate the type to assign the JNDI object to find.	Y	Class		Ignore if the value is not designated.
proxy-interface	Proxy Interface to use JNDI object	Y	Class		Ignore if the value is not designated.

An environment tag, the element of jndi-lookup tag, is used when registering JNDI Environment variable value. The environment tag has the list in the form of <variable name>=<variable value> as shown in 'foo=bar'.

Example

- Simple

  As the simplest setting, it finds the JNDI object using the name only. This example finds the JNDI object registered in the name of "jdbc/MyDataSource" and performs Dependency Injection to "dataSource" property of "userDao" Bean.

- <jee:jndi-lookup id="dataSource" jndi-name="jdbc/MyDataSource"/>
-
- <bean id="userDao" class="com.foo.JdbcUserDao">
- <!-- Spring will do the cast automatically (as usual) -->
- <property name="dataSource" ref="dataSource"/>
  </bean>

- With single JNDI environment settings

  Following is the example of finding JNDI object using the single JNDI environment setting.

- <jee:jndi-lookup id="dataSource" jndi-name="jdbc/MyDataSource">
- <jee:environment>foo=bar</jee:environment>
  </jee:jndi-lookup>

- With multiple JNDI environment settings

  Following is the example of finding JNDI object using several JNDI environment setting.

- <jee:jndi-lookup id="dataSource" jndi-name="jdbc/MyDataSource">
- <!-- newline-separated, key-value pairs for the environment (standard Properties format) -->
- <jee:environment>
- foo=bar
- ping=pong
- </jee:environment>
  </jee:jndi-lookup>

- Complex

  Following is the example of finding JNDI object through various setting in addition to name.

- <jee:jndi-lookup id="dataSource"
- jndi-name="jdbc/MyDataSource"
- cache="true"
- resource-ref="true"
- lookup-on-startup="false"
- expected-type="com.myapp.DefaultFoo"
  proxy-interface="com.myapp.Foo"/>

local-slsb tag

The local-slsb tag is for referring to EJB Stateless SessionBean.

tag Description

<jee:local-slsb id="bean id"
jndi-name="JNDI name"
business-interface="Java Class"
cache-home="true or false"
lookup-home-on-startup="true or false"
resource-ref="true or false">
<jee:environment>
name=value
ping=pong

...
</jee:environment>
</jee:local-slsb>

locak-slsb tag is mapped to LocalStatelessSessionProxyFactoryBean class of Spring Framework at the ratio of 1:1. Attribute of tag is as follows:

Attribute	Description	Optional	Data Type	Default value	Remarks
id	Bean id of Spring XML Configuration.	N	String		
jndi-name	JNDI name of EJB to search.	N	String		
business-interface	Business interface of EJB for Proxing	N	Class		
cache-home	This indicates a cache for EJB Home object searched once.	Y	boolean	true	
lookup-home-on-startup	This indicates whether to perform lookup at the time of start.	Y	boolean	true	
resource-ref	This indicates whether to search in J2EE Container.	Y	boolean	false	

The environment tag, the element of local-slsb tag, is used to register JNDI Environment variable value. The environment tag has the list in the form of <variable name>=<variable value> as the value as shown in 'foo=bar'.

Example

- Simple

  Example of simple use.

- <jee:local-slsb id="simpleSlsb" jndi-name="ejb/RentalServiceBean"
  business-interface="com.foo.service.RentalService"/>

- Complex

  Example of using various setting value to use local-slsb tag.

- <jee:local-slsb id="complexLocalEjb"
- jndi-name="ejb/RentalServiceBean"
- business-interface="com.foo.service.RentalService"
- cache-home="true"
- lookup-home-on-startup="true"
  resource-ref="true"/>

remote-slsb tag

remote-slsb tag is the tag that refers to remote EJB Stateless SessionBean.

tag Description

<jee:remote-slsb id="bean id"
jndi-name="JNDI name"
business-interface="Java Class"
cache-home="true or false"
lookup-home-on-startup="true or false"
resource-ref="true or false"
home-interface="Java Class"
refresh-home-on-connect-failure="true or false">
<jee:environment>
name=value

```
ping=pong
            ...
</jee:environment>
</jee:remote-slsb>
```

The remote-slsb tag is mapped to SimpleRemoteStatelessSessionProxyFactoryBean class of the Spring Framework at the ratio of 1:1. Attribute of tag is as follows:

Attribute	Description	Optional	Data Type	Default value	Remarks
id	Bean id of Spring XML Configuration.	N	String		
jndi-name	JNDI name of EJB to search.	N	String		
business-interface	Business interface of EJB for proxing.	N	Class		
cache-home	This indicates a cache for EJB Home object searched once.	Y	boolean	true	
lookup-home-on-startup	This indicate whether to perform lookup at the time of start.	Y	boolean	true	
resource-ref	This indicate whether to search in J2EE Container.	Y	boolean	false	
home-interface	EJB Home interface.	Y	Class		
refresh-home-on-connect-failure	This indicate whether to refresh EJB Home if connection fails.	Y	boolean	false	

The environment tag, element of remote-slsb tag is used to register the JNDI Environment variable value. The environment tag has the list in the form of <variable name>=<variable value> as the value as shown in 'foo=bar'.

Example

- Simple

  Example of simple use.

- <jee:remote-slsb id="complexRemoteEjb"
- jndi-name="ejb/MyRemoteBean"
- business-interface="com.foo.service.RentalService"
  home-interface="com.foo.service.RentalService"/>

- Complex

  Example of using various setting value to use remove-slsb tag.

- <jee:remote-slsb id="complexRemoteEjb"
- jndi-name="ejb/MyRemoteBean"
- business-interface="com.foo.service.RentalService"
- cache-home="true"
- lookup-home-on-startup="true"
- resource-ref="true"
- home-interface="com.foo.service.RentalService"
  refresh-home-on-connect-failure="true"/>

Use JndiTemplateClass

JndiTemplate class is the wrapper class providing for easy to use of the JNDI API.

bind

The method of the following JndiTemplateSample class binds the argument 'resource' to JNDI object in the argument ‘name’ using JndiTemplate.

```
importjavax.naming.NamingException;

importorg.springframework.jndi.JndiTemplate;

...

public class JndiTemplateSample
{
privateJndiTemplatejndiTemplate = new JndiTemplate();

    ...

publicboolean bind(final String name, Object resource)
    {
try
        {
jndiTemplate.bind(name, resource);
return true;
        }
catch (NamingException e)
        {
e.printStackTrace();
return false;
        }
    }

    ...
}
```

Lookup

They can look up the resource that is registered as the argument 'name' using JndiTemplate.

```
publicObjectlookupResource(final String name)
    {
try
        {
returnjndiTemplate.lookup(name);
        }
catch (NamingException e)
        {
e.printStackTrace();
return null;
        }
    }
```

Lookup with requiredType

thelookup method of JndiTemplate can designate the type you want as well as the name of resources to find.

```
public Foo lookupFoo(final StringfooName)
    {
try
        {
returnjndiTemplate.lookup(fooName, Foo.class);
        }
catch (NamingException e)
        {
e.printStackTrace();
return null;
        }
```

```
    }
```

Rebind

They can register resources again using the rebind method of JndiTemplate.

```
publicboolean rebind(final String name, Object resource)
    {
try
        {
jndiTemplate.rebind(name, resource);
return true;
        }
catch (NamingException e)
        {
e.printStackTrace();
return false;
        }
    }
```

Unbind

They can unregister the resource registered using unbind method of JndiTemplate.

```
publicboolean unbind(final String name)
    {
try
        {
jndiTemplate.unbind(name);
return true;
        }
catch (NamingException e)
        {
e.printStackTrace();
return false;
        }
    }
```

Reference

- Spring Framework JndiTemplate class API
- The Spring Framework - Reference Documentation A.2.3. The jee schema
- Java SE Guide to JNDI