IceBox Integration with IceGrid

IceGrid makes it easy to configure an IceBox server with one or more services.

On this page:

Deploying an IceBox Server

An IceBox server shares many of the same characteristics as other servers, but its special requirements necessitate a new descriptor. Unlike other servers, an IceBox server generally hosts multiple independent services, each requiring its own communicator instance and configuration file.

As an example, the following application deploys an IceBox server containing one service:

XML

```
<icegrid>
    <application name="IceBoxDemo">
        <node name="Node">
            <icebox id="IceBoxServer" exe="/opt/Ice/bin/icebox" activation="on-demand">
                <service name="ServiceA" entry="servicea:create">
                    <adapter name="${service}" endpoints="tcp"/>
                </service>
            </icebox>
        </node>
    </application>
</icegrid>
```

It looks very similar to a server descriptor. The most significant difference is the service descriptor, which is constructed much like a server in that you can declare its attributes such as object adapters and configuration properties. The order in which services are defined determines the order in which they are loaded by the IceBox server.

The value of the adapter's `name` attribute needs additional explanation. The symbol `service` is one of the names reserved by IceGrid. In the context of a service descriptor, `${service}` is replaced with the service's name, and so the object adapter is also named `ServiceA`.

Service Templates

If you are familiar with templates in general, an IceBox service template is readily understandable:

XML

```
<icegrid>
    <application name="IceBoxApp">
        <service-template id="ServiceTemplate">
            <parameter name="name"/>
            <service name="${name}" entry="DemoService:create">
                <adapter name="${service}" endpoints="default"/>
                <property name="${service}.Identity" value="${server}-${service}"/>
            </service>
        </service-template>
        <node name="Node1">
            <icebox id="IceBoxServer" endpoints="default"
                exe="/opt/Ice/bin/icebox" activation="on-demand">
                <service-instance template="ServiceTemplate" name="Service1"/>
            </icebox>
        </node>
    </application>
</icegrid>
```

In this application, an IceBox server is deployed on a node and has one service instantiated from the service template. Of particular interest is the `property` descriptor, which uses another reserved name `server` to form the property value. When the template is instantiated by the service instance descriptor, the symbol `${server}` is replaced with the name of the enclosing server, so the property definition expands as follows:

```
Service1.Identity=IceBoxServer-Service1
```

As with server instances, you can specify additional properties for the service instance without modifying the template. These properties can be defined in the `service-instance` element, as shown below:

XML

```
<icegrid>
    <application name="IceBoxApp">
        ...
        <node name="Node1">
            <icebox id="IceBoxServer"endpoints="default"
                exe="/opt/Ice/bin/icebox" activation="on-demand">
                <service-instance template="ServiceTemplate" name="Service1">
                    <properties>
                        <property name="Ice.Trace.Network" value="1"/>
                    </properties>
                </service-instance>
            </icebox>
        </node>
    </application>
</icegrid>
```

Advanced Service Templates

A more sophisticated use of templates involves instantiating a service template in a server template:

XML

```
<icegrid>
    <application name="IceBoxApp">
        <service-template id="ServiceTemplate">
            <parameter name="name"/>
            <service name="${name}" entry="DemoService:create">
                <adapter name="${service}" endpoints="default"/>
                <property name="${name}.Identity" value="${server}-${name}"/>
            </service>
        </service-template>
        <server-template id="ServerTemplate">
            <parameter name="id"/>
            <icebox id="${id}" endpoints="default"
                exe="/opt/Ice/bin/icebox" activation="on-demand">
                <service-instance template="ServiceTemplate" name="Service1"/>
            </icebox>
        </server-template>
        <node name="Node1">
            <server-instance template="ServerTemplate" id="IceBoxServer"/>
        </node>
    </application>
</icegrid>
```

This application is equivalent to our first example of service templates. Now, however, the process of deploying an identical server on several nodes has become much simpler.

If you need the ability to customize the configuration of a particular service instance, your server instance can define a property set that applies only to the desired service:

XML

```
<icegrid>
    <application name="IceBoxApp">
        <node name="Node1">
            <server-instance template="ServerTemplate" id="IceBoxServer">
                <properties service="Service1">
                    <property name="Ice.Trace.Network" value="1"/>
                </properties>
            </server-instance>
        </node>
    </application>
</icegrid>
```

As this example demonstrates, the `service` attribute of the property set denotes the name of the target service.

See Also

- IceBox
- IceBox Descriptor Element
- Service Descriptor Element
- Service-Template Descriptor Element
- Server-Template Descriptor Element
- Properties Descriptor Element
- Using Descriptor Variables and Parameters