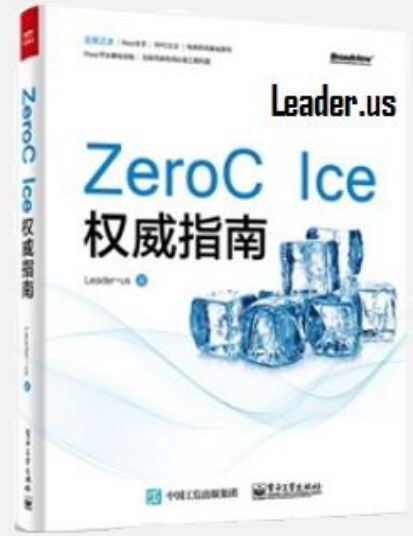
Zeroc Ice微服务架构企业应用实践指南

七：Mycat Ice企业框架（上）



龙果学院 roncoo.com
主讲老师：Leader.us
联 系QQ：719867650

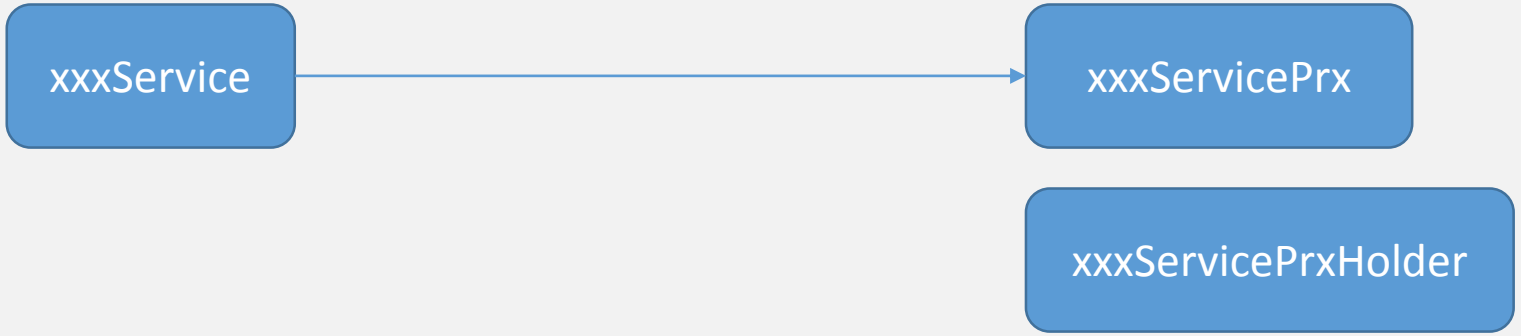
Leader.us高端架构师精品系列课程

本集视频主要内容

- 契约式开发
- 简化客户端调用
- 简化IceBox.Service开发
- 服务调用拦截框架

契约式开发

服务名 生成的客户端代理类



简化客户端调用

```
MyServicePrx mysvcPrx=(MyServicePrx)
ICEClientUtil.getSerivcePrx(MyServicePrx.class);
System.out.println(mysvcPrx.hellow());
```

ICEClientUtil几个特点

一：客户端调用简单方便

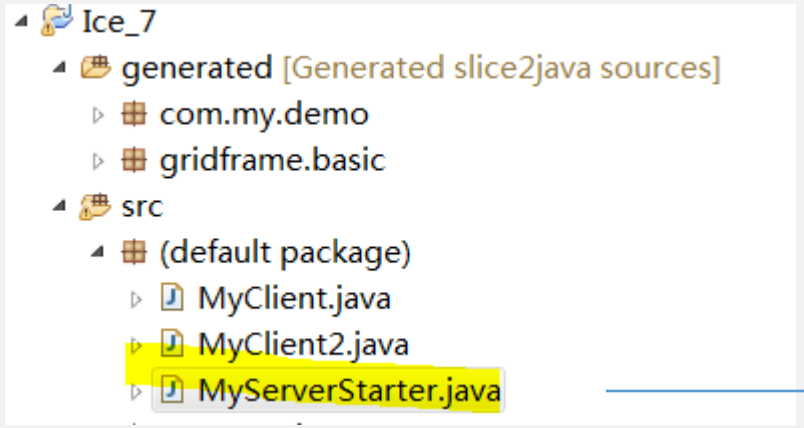
```
TicketServicePrx ticketServicePrx = (TicketServicePrx) ICEClientUtil.getSerivcePrx(TicketServicePrx.class);
```

二：服务之间相互调用的时候，公用相同的Communicator

public static ObjectPrx getSerivcePrx(Ice.Communicator communicator, Class serviceCls)

第三：调用本地部署的或者单独在Grid之外的Service一视同仁

ICEClientUtil例子

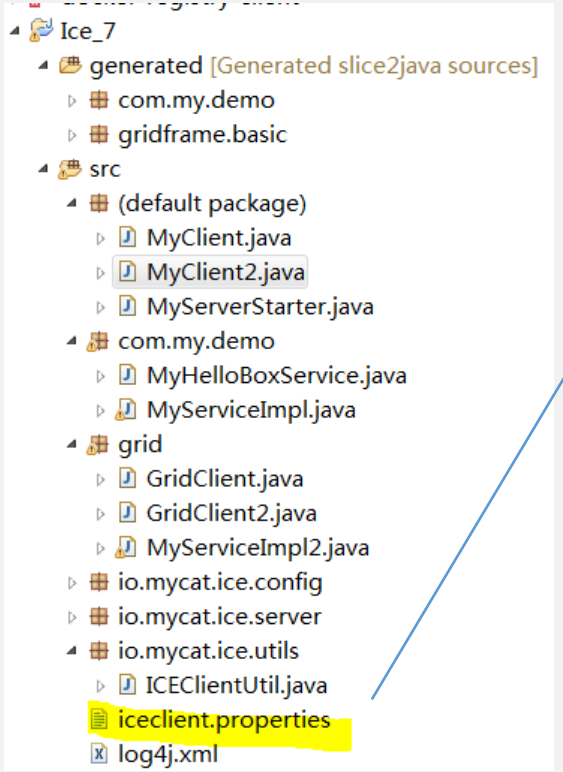


启动服务

```java
public class MyClient2 {

    public static void main(String[] args)
    {
        MyServicePrx mysvcPrx=(MyServicePrx) ICEClientUtil.getSerivcePrx(MyServicePrx.class);
        System.out.println(mysvcPrx.hellow());
    }
}
```

```
Ice client's locator is IceGrid/Locator:tcp -h localhost -p 4061 proxy cache time out seconds :300
find local service:MyService at endpoint:MyService:default -h localhost -p 20000
Hello world
```

ICEClientUtil详解




```
#--Ice.Default.Locator=IceGrid/Locator:tcp -h 16.156.210.211 -
p 4061
--Ice.Default.Locator=IceGrid/Locator:tcp -h localhost -p 4061
idleTimeOutSeconds=300
#stand alone  started servant (out of ice grid)
local.MyService=MyService:default -h localhost -p 20000
```

本地或独立于Grid之外的Ice Object

ICEClientUtil详解（一）

```
public static Ice.Communicator getICECommunictor() {
    if (ic == null) {
        synchronized (ICEClientUtil.class) {
            if (ic == null) {
                if (iceLocator == null) {
                    ResourceBundle rb = ResourceBundle.getBundle("iceclient", Locale.ENGLISH);

                    iceLocator = rb.getString(LocatorKey);
                    idleTimeOutSeconds = Integer.parseInt(rb.getString("idleTimeOutSeconds"));
                    System.out.println("Ice client's locator is " + iceLocator + " proxy cache time out seconds :"
                            + idleTimeOutSeconds);
                    loadStandAloneProxys(rb);
                }
                String[] initParams = new String[] { LocatorKey + "=" + iceLocator };
                // , "--Ice.Default.PreferSecure=1"
                // String[] initParams = new String[] { locatorKey + "="
                // + iceLocator };

                ic = Ice.Util.initialize(initParams);
                createMonitorThread();
            }
        }
    }
```

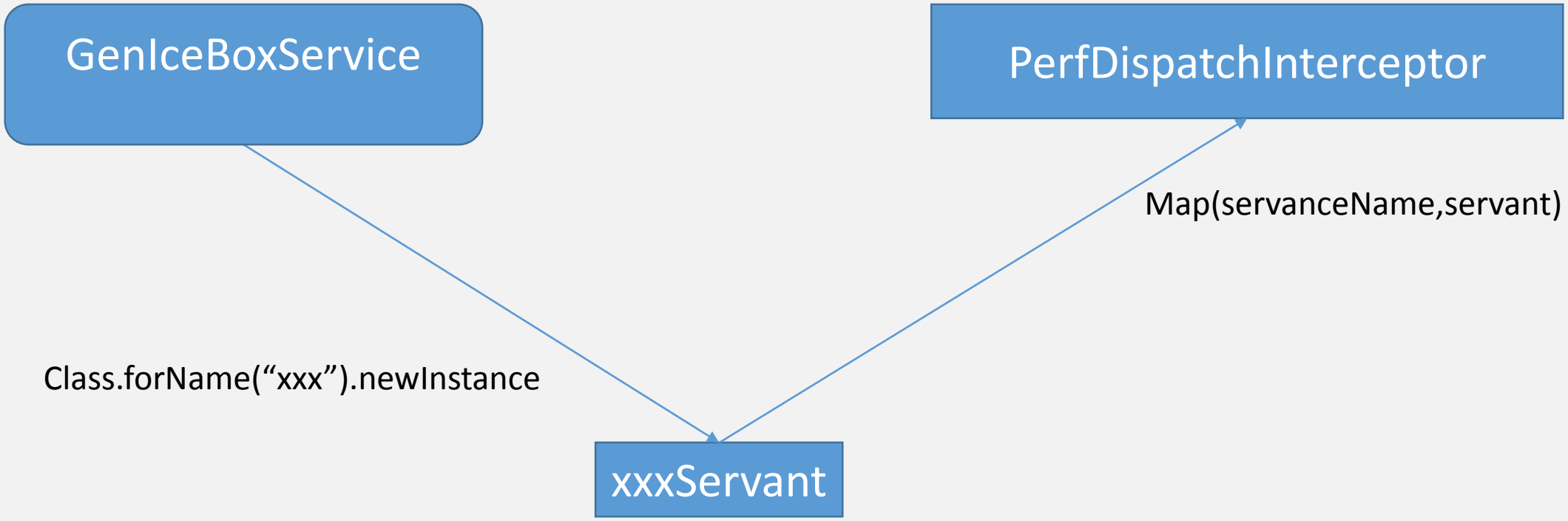
ICEClientUtil详解（二）

```
@SuppressWarnings("rawtypes")
private static Ice.ObjectPrx createObjectPrxFromEndpoint(Ice.Communicator communicator, Ice.ObjectPrx base,
        Class serviceCls) {
    try {
        String clsName = serviceCls.getName();

        ObjectPrx proxy = (ObjectPrx) Class.forName(clsName + "Helper").newInstance();
        Method m1 = proxy.getClass().getDeclaredMethod("uncheckedCast", ObjectPrx.class);
        proxy = (ObjectPrx) m1.invoke(proxy, base);
        return proxy;
    } catch (Exception e) {
        e.printStackTrace();
        throw new RuntimeException(e);
    }
}
```

ICEClientUtil详解（三）

```
    @SuppressWarnings("rawtypes")
    public static ObjectPrx getSerivcePrx(Class serviceCls) {
        ObjectPrx proxy = cls2PrxMap.get(serviceCls);
        if (proxy != null) {
            lastAccessTimestamp = System.currentTimeMillis();
            return proxy;
        }

        proxy = createIceProxy(getICECommunictor(), serviceCls);
        cls2PrxMap.put(serviceCls, proxy);
        lastAccessTimestamp = System.currentTimeMillis();
        return proxy;
    }

    static class MonitorThread extends Thread {
        public void run() {
            while (!Thread.currentThread().isInterrupted()) {
                try {
                    Thread.sleep(5000L);
                    if (lastAccessTimestamp + idleTimeOutSeconds * 1000L < System.currentTimeMillis()) {
                        closeCommunicator(true);
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }

            }
        }
    }
}
```

简化IceBox.Service开发与服务拦截插件



GenIceBoxService用法

```
<icebox id="MyHelloServer${id}" exe="java">
<properties>
<properties refid="props" />
</properties>
<option>IceBox.Server</option>
<env>CLASSPATH=C:\ZeroC\Ice-3.6.1\lib\*;C:\project\Ice_Hellow\bin;C:\project\Ice_Hellow\lib\*</env>
<service name="MyService" entry="com.hp.ice.server.GenIceBoxService">
<property name="servantClassName" value="com.my.demo.MyServiceImpl"/>
<adapter name="MyService" id="MyService${id}" endpoints="default" replica-group="MyServiceRep">
</adapter>
</service>
<service name="MyService2" entry="com.hp.ice.server.GenIceBoxService">
<property name="servantClassName" value="grid.MyServiceImpl2"/>
<adapter name="MyService2" id="MyService2${id}" endpoints="default" replica-group="MyService2Rep">
</adapter>
</service>
</icebox>
```

GenIceBoxService的源码分析

public class GenIceBoxService implements Service {

```
public void start(String name, Communicator communicator, String[] args) {
    String servantClassName = communicator.getProperties().getProperty("servantClassName");
    System.out.println("load servant class "+servantClassName);
    Ice.Util.setProcessLogger(iceLogger);
    _adapter = communicator.createObjectAdapter(name);
    id = communicator.stringToIdentity(name);
    Class<?> servantClass = null;
    try {
        servantClass = ClassHelper.forName(servantClassName);
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
        logger.error(e.getMessage(), e);
    }

    Ice.Object object = null;
    if (null != servantClass) {
        try {
            object = (Object) servantClass.newInstance();
        } catch (InstantiationException e) {
            logger.error(e.getMessage(), e);
        } catch (IllegalAccessException e) {
            logger.error(e.getMessage(), e);
        }
    }

    if (null != object) {
        _adapter.add(PerfDispatchInterceptor.addICEObject(id, object), id);
    }

    _adapter.activate();
    logger.info(name + " service started ,with param size " + args.length + " detail:" + Arrays.toString(args));

}
```

从配置中实例化真正的servant对象

```
<service name="MyService" entry="com.hp.ice.server.GenIceBoxService">
<property name="servantClassName" value="com.my.demo.MyServiceImpl"/>
    <adapter name="MyService" id="MyService${id}" endpoints="default"
        replica-group="MyServiceRep">
    </adapter>
</service>
<service name="MyService2" entry="com.hp.ice.server.GenIceBoxService">
<property name="servantClassName" value="grid.MyServiceImpl2"/>
    <adapter name="MyService2" id="MyService2${id}" endpoints="default"
        replica-group="MyService2Rep">
    </adapter>
</service>
```

Ice服务拦截插件源码解释

public class PerfDispatchInterceptor extends DispatchInterceptor {

```
public static DispatchInterceptor addICEObject(Ice.Identity id,
        Ice.Object iceObj) {
    id2ObjectMAP.put(id, iceObj);
    return self;
}

/**
 * dispatch request to servant
 */
@Override
public DispatchStatus dispatch(Request request) {
    Identity theId = request.getCurrent().id;
    String inf = "dispach req,method:" + request.getCurrent().operation
            + " service:" + theId.name + " server address:"
            + request.getCurrent().con;
    logger.info(inf + " begin");
    try {

        DispatchStatus reslt = id2ObjectMAP.get(request.getCurrent().id)
                .ice_dispatch(request);
        logger.info(inf + " success");
        return reslt;
    } catch (RuntimeException e) {
        logger.info(inf + " error " + e);
        throw e;
    }
}

public static void removeICEObject(Identity id) {
    logger.info("remove ice object " + id);
    id2ObjectMAP.remove(id);

}
```

grid.xml配置

```
<icebox id="MyHelloServer${id}" exe="java" >
<properties>
<properties refid="props" />
</properties>
<option>io.mycat.ice.server.Sl4jIceBoxServer</option>
<env>CLASSPATH=C:\ZeroC\Ice-3.6.1\lib\*;C:\project\Ice_7\bin;C:\project\Ice_7\lib\*</env>
<service name="MyService" entry="io.mycat.ice.server.GenIceBoxService">
<property name="servantClassName" value="com.my.demo.MyServiceImpl"/>
<adapter name="MyService" id="MyService${id}" endpoints="default"
replica-group="MyServiceRep">
</adapter>
</service>
<service name="MyService2" entry="io.mycat.ice.server.GenIceBoxService">
<property name="servantClassName" value="grid.MyServiceImpl2"/>
<adapter name="MyService2" id="MyService2${id}" endpoints="default"
replica-group="MyService2Rep">
</adapter>
</service>
</icebox>
```

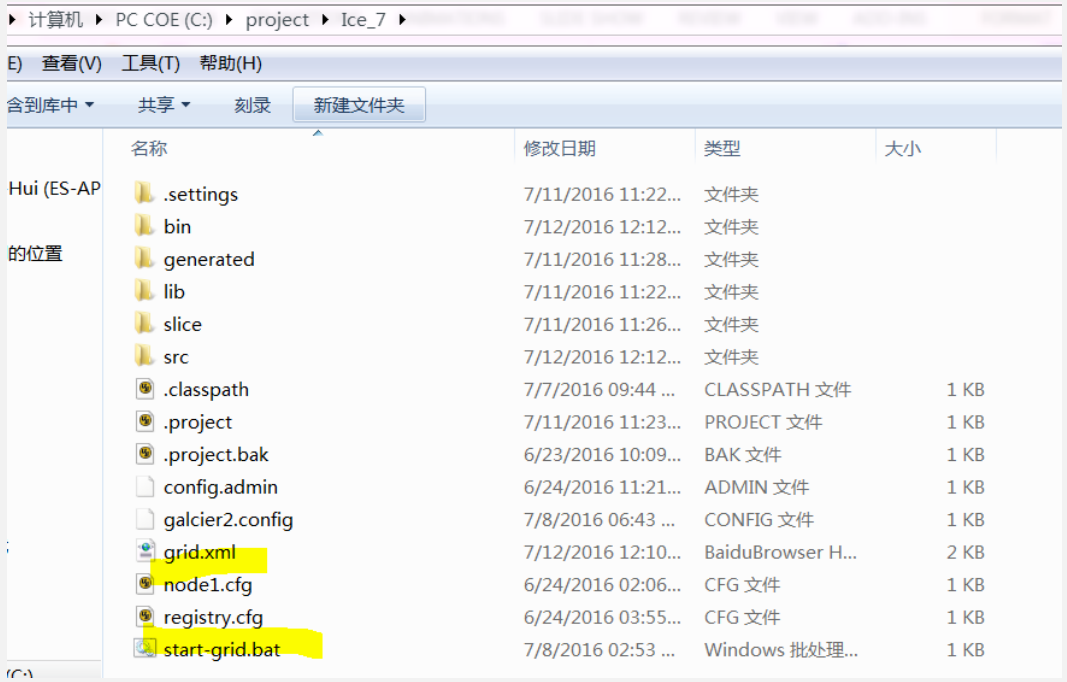
实现了SL4j的IceBoxServer类

```
public class Sl4jLogerI implements Ice.Logger {
private final org.slf4j.Logger logger;

public Sl4jLogerI(String loggerName) {
logger = LoggerFactory.getLogger(loggerName);
}

@Override
public void print(String message) {
logger.info(message);

}

@Override
public void trace(String category, String message) {
logger.debug(category + " " + message);
}

@Override
public void warning(String message) {
logger.warn(message);
}

@Override
public void error(String message) {
logger.error(message);

}
```

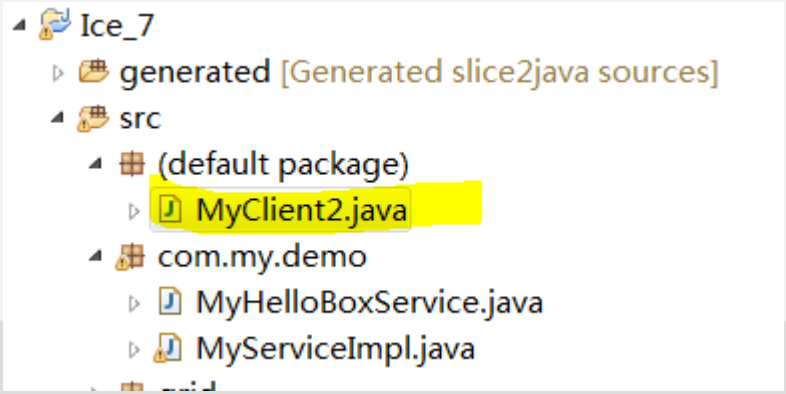
```
public class Sl4jIceBoxServer {
public static void main(String[] args)
{
  Ice.InitializationData initData = new
Ice.InitializationData();
        initData.properties =
Ice.Util.createProperties();

initData.properties.setProperty("Ice.Admin.Dela
yCreation", "1");
        initData.logger=new
Sl4jLogerI("system");

        Server server = new Server();

System.exit(server.main("IceBox.Server", args,
initData));
}
}
```

实践练习

Ice_7工程




```
>>> application remove MyApp
>>>application add C:\project\Ice_7\grid.xml
>>> server list
MyHelloServer1
MyHelloServer2
>>> server start MyHelloServer1
error: the server didn't start successfully:
The server activation timed out.
>>> server start MyHelloServer1
error: the server didn't start successfully:
The server is already active.
>>>
```



下一集：Mycat定制企业框架（下）

谢谢

THANK YOU

主讲老师：Leader.us
联 系QQ：719867650