

Diameter MUX Source Overview

Table of Contents

The Diameter MUX capabilities are defined by the `MBean` interface `org.mobicients.diameter.stack.DiameterStackMultiplexerMBean`. This interface defines two types of methods:

Management

Used by RHQ console. These methods are outside the scope of this documentation.

Stack Accessors

Methods that allow the user to retrieve and use a wrapped stack.

The methods below are of interest to a Diameter MUX user:

```
public Stack getStack();
```

Returns a stack wrapped by the multiplexer. It is present as a convenience method, and the stack should only be changed directly by expert users.

```
public void registerListener(DiameterListener listener, ApplicationId[] appIds) throws  
IllegalStateException;
```

Registers a listener to be triggered when a message for a certain application ID is received.

```
public void unregisterListener(DiameterListener listener);
```

Removes the message listener.

```
public DiameterStackMultiplexerMBean getMultiplexerMBean();
```

Returns the actual instance of MUX.

The listener interface is defined below:

```
package org.mobicients.diameter.stack;  
  
import java.io.Serializable;  
  
import org.jdiameter.api.Answer;  
import org.jdiameter.api.EventListener;  
import org.jdiameter.api.NetworkReqListener;  
import org.jdiameter.api.Request;  
  
public interface DiameterListener extends NetworkReqListener, Serializable,  
    EventListener<Request, Answer>  
{  
  
}
```

MUX can be used as follows:

```

public class DiameterActor implements DiameterListener
{
    private ObjectName diameterMultiplexerObjectName = null;
    private DiameterStackMultiplexerMBean diameterMux = null;

    private synchronized void initStack() throws Exception {
        this.diameterMultiplexerObjectName =
            new ObjectName("diameter.mobicens:service=DiameterStackMultiplexer");

        Object[] params = new Object[]{};
        String[] signature = new String[]{};

        String operation = "getMultiplexerMBean";
        this.diameterMux=mbeanServer.invoke(this.diameterMultiplexerObjectName,
operation,
            params, signature);

        long acctAppIds = new long[]{19312L};
        long acctVendorIds = new long[]{193L};
        long authAppIds = new long[]{4L};
        long authVendorIds = new long[]{0L};
        List<ApplicationId> appIds = new ArrayList<ApplicationId>();
        for(int index = 0;index<acctAppIds.length;index++) {
            appIds.add(ApplicationId.createByAccAppId(acctVendorIds[index],
acctAppIds[index]));
        }

        for(int index = 0;index<authAppIds.length;index++) {
            appIds.add(ApplicationId.createByAuthAppId(authVendorIds[index],
authAppIds[index]));
        }

        this.diameterMux.registerListener(this, appIds.toArray(new ApplicationId
[appIds.size()]));
        this.stack = this.diameterMux.getStack();
        this.messageTimeout = stack.getMetaData().getConfiguration().getLongValue(
            MessageTimeOut.ordinal(), (Long) MessageTimeOut.defValue());
    }
}

```