Table of Contents

Programmers Reference, Adding Custom Channels, Source and sink, Writing Sink

Writing a Sink

To write a custom Sink follow the steps below:

1. Inherit the Sink from the `AbstractSimpleSink` class.
2. If necessary override init method to provide the Sink-specific initialization (if there's any) – this method is called once when the Sink is initialized. The Sink-specific properties would have been entered by the user in the Configuration Tool (see also section 4.3)
3. You must override the following method to provide your own handling of messages sent from ***Aware IM***:

```
public void processMessage (Message msg)
throws PipelineException;
```

You can translate messages from ***Aware IM*** using the following techniques:

1. get the array of DataObject's from the message:

```
DataObject [] dobs = msg.peekDataObjects ();
```

2. use MessageInterpreter class to interpret the message (see also 4.1.3):

```
MessageInterpreter mi = new MessageInterpreter (dobs);
INotification notif = mi.getNotification (); /**or**/
String serviceName  = mi.getServiceName ();
```

As an example you can see the code snippet below that implements the Sink that sends outgoing e-mails from ***Aware IM***. Note that this code is for illustration purposes only and certain details such as e-mail construction and sending and error handling are omitted.

```
 public  class EmailSink extends AbstractSimpleSink
 {
    private Stringm_mailHost= null ;
    private Sessionm_session= null ;
    private Stringm_fromAddress= null ;
    /** Initializethesink. Mail host and 'from address' properties are
specified in the user interface */
    public  void init(Stringname,Propertiesprops,Stringprefix,
        Controllercontroller)
    throws IbafException
    {
        try
        {
            super .init(name,props,prefix,controller);
            m_mailHost=props.getProperty(prefix+"."+"MAIL_HOST");
            if (m_mailHost== null )
                throw  new IbafException("Mail host property for outgoing e-
mails is not specified.");
```

```
            PropertiessessionProps=System.getProperties();
            sessionProps.put("mail.smtp.host",m_mailHost);
            m_fromAddress=props.getProperty(prefix+"."+"MAIL_FROM_ADDRESS");
            m_session=Session.getDefaultInstance(sessionProps,null);
        }
        catch (Exceptione)
        {
            e.printStackTrace();
            throw  new IbafException(e.toString());
        }
    }
    /**  Message handling method – must be provided */
    public  void processMessage(Message msg)
    throws PipelineException
    {
        Stringsubject= null ,toAddress= null ;
        StringsentDateStr= null ;
        try
        {
            DataObject[]dobs=msg.peekDataObjects();
            // get all message parameters from data objects
            MessageInterpretermi= new MessageInterpreter(dobs);
            // get e-mail address from channel values
            DataObjectchvDob=mis.getChannelValues();
            if (chvDob== null )
            return ; // ignore the message
        try
        {
            toAddress=(String)chvDob.getAttributeValue(EMAIL_ADDRESS);
        }
        catch (InvalidParameterExceptionipe)
        {

        }
        if (toAddress== null ) return ;//** ignore the message
        INotificationnotif=mis.getNotification();
        if (notif== null ) return ; // ignore the message
        // we expect notification of a certain structure – it must have
"BODY" and SUBJECT attributes defined
        Stringbody= null ;
        try
        {
            body=(String)notif.getAttributeValue (EMAIL_BODY_ATTR);
        }
        catch (InvalidParameterExceptionipe)
        {

        }
        if (body== null )
        return ; // ignore the message// subject is optional
        try
```

```
        {
            subject=(String)notif.getAttributeValue (EMAIL_SUBJECT_ATTR);
        }
        catch (Exceptione)
        {
        }
        if (subject== null ) subject="";
        // create MAPI e-mail message
        MimeMessageemail=createEmail(body,subject, toAddress);
        if (email!= null )
        {
            // send e-mail using MAPI
            sendEmail(email,toAddress,new Date ());
        }
    }
    catch (Exceptione)
    {
    }
}

/**  Send the e-mail using MAPI. A lot of details are omitted  **/
private  boolean sendEmail(MimeMessageemail,StringtoAddress, StringsentDate)
{
    try
    {
        Transport.send(email);
    }
    catch (Exceptione)
    {
    }
    return true;
}

/** Create e-mailmessage using MAPI. Details are omitted */
private MimeMessagecreateEmail(Stringbody,Stringsubject,String
toAddress,DatesentDate)
throws Exception
{
    MimeMessagemailMessage= new MimeMessage(m_session);
    if (m_fromAddress!= null ) mailMessage.setFrom(InternetAddress.parse
mailMessage.setRecipients(javax.mail.Message.RecipientType.TO,
InternetAddress.parse(toAddress, false ));
        mailMessage.setSubject (subject);
        mailMessage.setSentDate(sentDate);
        return mailMessage;
}
```