

Table of Contents

Writing a Sink 2

[Programmers Reference](#), [Adding Custom Channels](#), [Source and sink](#), [Writing Sink](#), [example](#)

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 **AwareIM**:

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

You can translate messages from **AwareIM** 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 **AwareIM**. 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 String m_mailHost=null ;
    private Session m_session=null ;
    private String m_fromAddress= null ;
    /** Initializethesink. Mail host and 'from address' properties are
specified in the user interface */
    public void init(String name, Properties props, String prefix,
        Controller controller)
        throws IbafeException
    {
        try
        {
            super .init(name, props, prefix, controller);
            m_mailHost=props.getProperty(prefix+"."+ "MAIL_HOST");
            if (m_mailHost== null )
                throw new IbafeException("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 IbafeException(e.toString());
    }
}
/** Message handling method – must be provided */
public void processMessage(Message msg)
throws PipelineException
{
    Stringssubject= 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 (Exception e)
        {
        }
        if (subject== null ) subject="";
        // create MAPI e-mail message
        MimeMessage email=createEmail(body,subject, toAddress);
        if (email!= null )
        {
            // send e-mail using MAPI
            sendEmail(email,toAddress,new Date ());
        }
    }
    catch (Exception e)
    {
    }
}

/** Send the e-mail using MAPI. A lot of details are omitted */
private boolean sendEmail(MimeMessage email,String toAddress, String sentDate)
{
    try
    {
        Transport.send(email);
    }
    catch (Exception e)
    {
    }
    return true;
}

/** Create e-mail message using MAPI. Details are omitted */
private MimeMessage createEmail(String body,String subject,String
toAddress,Date sentDate)
throws Exception
{
    MimeMessage emailMessage= new MimeMessage(m_session);
    if (m_fromAddress!= null ) emailMessage.setFrom(InternetAddress.parse
emailMessage.setRecipients(javax.mail.Message.RecipientType.TO,
InternetAddress.parse(toAddress, false ));
    emailMessage.setSubject (subject);
    emailMessage.setSentDate(sentDate);
    return emailMessage;
}
```

From:

<http://www.awareim.com/dokuwiki/> - **Documentation**

Permanent link:

<http://www.awareim.com/dokuwiki/docs/3500/0400/0410/0411>

Last update: **2025/06/12 02:36**

