Oracle® Java CAPS Email Binding Component User's Guide

Part No: 821–2638
March 2011

120126@25097

Contents

Using the eMail Binding Component

This guide provides an overview of the eMail Binding Component, and includes information and instructions for implementing, configuring, and deploying the binding component in a JBI project. The eMail Binding Component provides connectivity for SMTP, POP3, and IMAP email servers in a JBI environment.

What You Need to Know

These topics provide information you should know before you use the eMail Binding Component:

- "About the eMail Binding Component" on page 6
- "eMail Binding Component Features" on page 6
- "eMail Binding Component Protocols" on page 7
- "eMail Binding Component Property Configuration" on page 7
- "eMail BC Sample Projects" on page 7

What You Need to Do

These topics provide instructions for using the eMail Binding Component:

- "Creating the eMail BC WSDL Document" on page 8
- "Configuring eMail BC WSDL Attributes" on page 16
- "Configuring the eMail Binding Component Runtime Properties" on page 25
- "Creating Application Configurations for Connectivity Parameters" on page 30
- "Using Application Variables" on page 33
- "Using eMail BC Normalized Message Properties in a Business Process" on page 37

Reference Information

These topics provide additional reference information about configuring and using the eMail Binding Component:

- "New WSDL Wizard Properties for the eMail BC" on page 14
- "eMail Binding and Service Level WSDL Attributes" on page 19

- “eMail Binding Component Runtime Property Descriptions” on page 27
- “Normalized Message Properties for the eMail Binding Component” on page 39

About the eMail Binding Component

The eMail Binding Component provides external connectivity for email servers, and supports sending email messages using the SMTP protocol and receiving email messages using the POP3 and IMAP protocols. Binding components implement the protocol transformations between abstract messages handled by the JBI Service Engines and concrete messages of the protocol. Other JBI components, such as the BPEL Service Engine, can leverage the eMail Binding Component to transform and route the messages.

The eMail Binding Component sends messages to, and retrieves messages from, an email server using the JavaMail API. This allows you to automate email operations that currently need to be performed manually. You can log into a server, generate email messages, specify recipients, and add subject lines, content, and attachments. You can also save attachments from incoming email messages.

Note – You can access the JavaMail documentation here: `http://www.oracle.com/technetwork/java/index-138643.html`.

The eMail Binding Component WSDL file defines server connectivity and email handling using a set of extensibility elements. The WSDL extensibility elements are part of the `binding` and `service` sections of WSDL documents. Both the `binding` and `service` sections of a WSDL document must be properly configured to determine the source of a message, how the message is transformed, and the destination of that message.

eMail Binding Component Features

The eMail Binding Component provides the following features and functionality to Java CAPS:

- Gives Java CAPS applications the ability to read and send email messages.
- Provides easy configuration through a wizard that guides you through the steps of creating an email WSDL document.
- Allows you to dynamically set server properties and email handling information using normalized message properties in a BPEL process.
- Allows you to define application variables and configurations so you can port your email applications from one environment to another.
- Supports SMTP, POP3, and IMAP protocols.
- Supports Secure Sockets Layer (SSL), providing secure communication channels for data exchanges, safe from unauthorized users.

- Allows you to store incoming attachments and attach files to outgoing messages, including text, HTML, XML, and image files.
- Supports several character encoding types to enable sending email messages in multiple languages.

eMail Binding Component Protocols

The eMail Binding Component takes advantage of the following widely-used, standard protocols. The protocols you use depend on your email server configuration.

- **POP3 (Post Office Protocol)**: Used to retrieve email message from an email account.
- **IMAP (Internet Message Access Protocol)**: Used to retrieve email message from an email account.
- **SMTP (Simple Mail Transfer Protocol)**: Used to send email messages to email accounts.

eMail Binding Component Property Configuration

You can configure the server connectivity properties and message handling information in several different places for the eMail Binding Component. Each of the following areas configures the binding component in a different scope.

- New WSDL Wizard: When you create the WSDL file for the eMail Binding Component, you can configure all of the connectivity and email handling properties. The wizard generates the appropriate extensibility elements and attributes in the WSDL file.
- WSDL File Elements: Once you create the WSDL file, you can update the extensibility elements and attributes directly in the WSDL file. The values you supply here only apply to the specific eMail Binding Component that the WSDL file represents.
- Application Configurations: You can configure server connectivity properties that override those same properties in the WSDL file and apply them to an endpoint in a Composite Application. This allows you to easily deploy the Composite Application within different environments.
- Normalized Message Properties : When you connect an Email Binding Component to a business process, the normalized message properties are made available in the BPEL Designer Mapper. These properties allow you to assign property values dynamically based on message content.

eMail BC Sample Projects

There are several sample projects for the eMail Binding Component to help you get started with the eMail BC and further understand the information and instructions provided in this document. To download the samples and read information on how to run them, go to `http://wiki.open-esb.java.net/Wiki.jsp?page=EmailBC`.

Working With the eMail Binding Component WSDL Document

The WSDL document defines an eMail interface for the project. You create and configure the WSDL document using the New WSDL Wizard, and you can further configure the interface using the WSDL Editor in NetBeans.

The following topics provides instructions for working with the WSDL document for the eMail BC:

- "Creating the eMail BC WSDL Document" on page 8
- "Configuring eMail BC WSDL Attributes" on page 16

Creating the eMail BC WSDL Document

The following topics provide instructions for creating email WSDL documents for reading and sending email messages under different protocols.

- "To Create a WSDL Document to Read email (IMAP or POP3)" on page 8
- "To Create a WSDL Document to Send email (SMTP)" on page 10

For information about the fields you need to fill in on the New WSDL Wizard, see "New WSDL Wizard Properties for the eMail BC" on page 14.

▼ To Create a WSDL Document to Read email (IMAP or POP3)

1. **In the NetBeans Projects window, right-click the project or a folder within the project where you want to add the WSDL document.**

2. **Point to New and then select WSDL Document.**

   The New WSDL Document Wizard appears.

3. **Enter a name for the WSDL document, and verify or update the folder location for the file.**

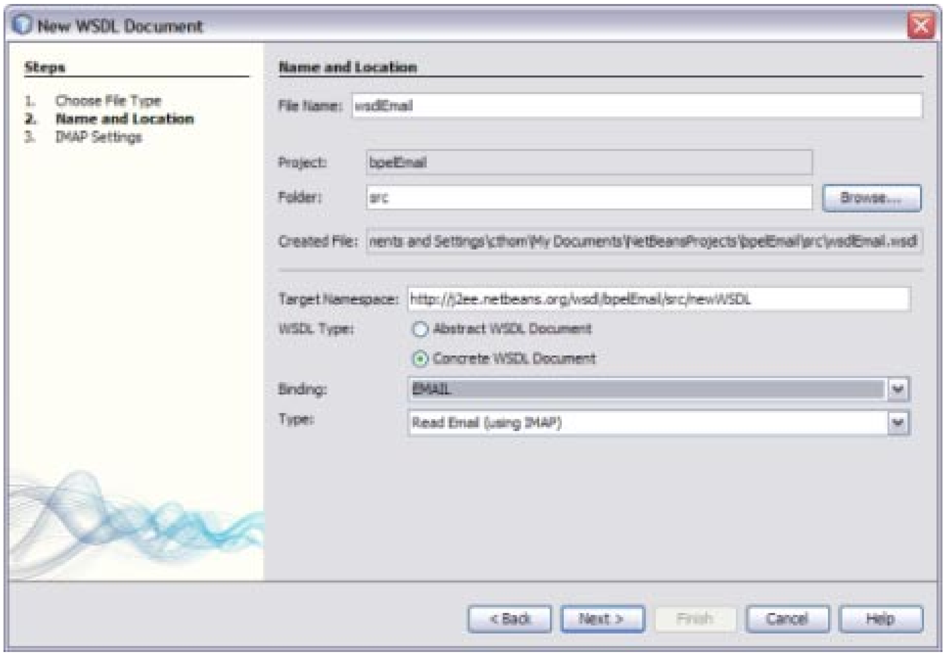
4. **Select Concrete WSDL Document.**

   The Binding and Type fields appear.

5. **For the Binding, select EMAIL; for the Type, do one of the following:**

   - **If your email server uses IMAP, select Read Email (using IMAP).**

- **If your email server uses POP3, select Read Email (using POP3).**


New WSDL Document

Steps
1. Choose File Type
2. **Name and Location**
3. IMAP Settings

Name and Location

File Name: wsdlEmail

Project: bpelEmail

Folder: src

Created File: nents and Settings\cthom\My Documents\NetBeansProjects\bpelEmail\src\wsdlEmail.wsdl

Target Namespace: http://j2ee.netbeans.org/wsdl/bpelEmail/src/newWSDL

WSDL Type: Abstract WSDL Document [✓ Concrete WSDL Document]

Binding: EMAIL

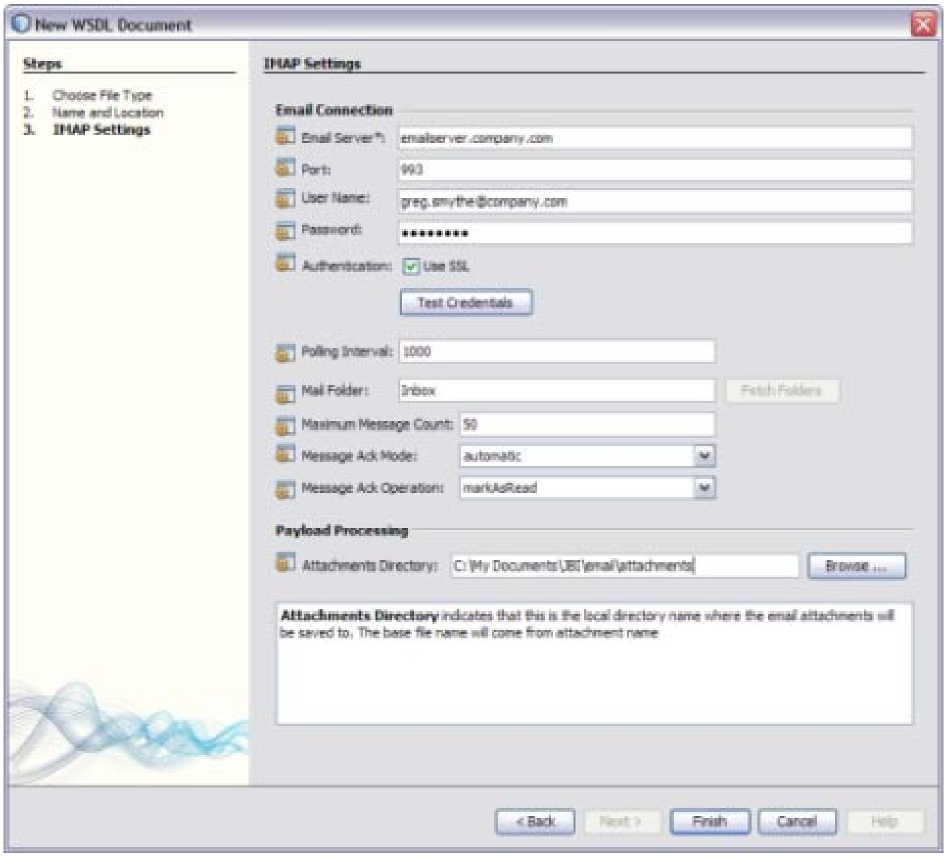
Type: Read Email (using IMAP)

6 Click Next.

The IMAP Settings or POP3 Settings window appears.

7 Fill in the settings according to your email client's IMAP or POP3 server.

For information about these fields, see "New WSDL Wizard Properties for the eMail BC" on page 14.



Email Connection

Email Server*:	emailserver.company.com
Port:	993
User Name:	greg.smythe@company.com
Password:	*********
Authentication:	✓ Use SSL
Polling Interval:	1000
Mail Folder:	Inbox
Maximum Message Count:	50
Message Ack Mode:	automatic
Message Ack Operation:	markAsRead

Payload Processing

Attachments Directory:	C:\My Documents\JBI\email\attachments

Attachments Directory indicates that this is the local directory name where the email attachments will be saved to. The base file name will come from attachment name

8 After you enter the email server connection information, click Test Credentials to be sure you can connect to the server.

9 On the New WSDL Document Wizard, click Finish.

▼ To Create a WSDL Document to Send email (SMTP)

1 In the NetBeans Projects window, right-click the project or a folder within the project where you want to add the WSDL document.

2 Point to New and then select WSDL Document.

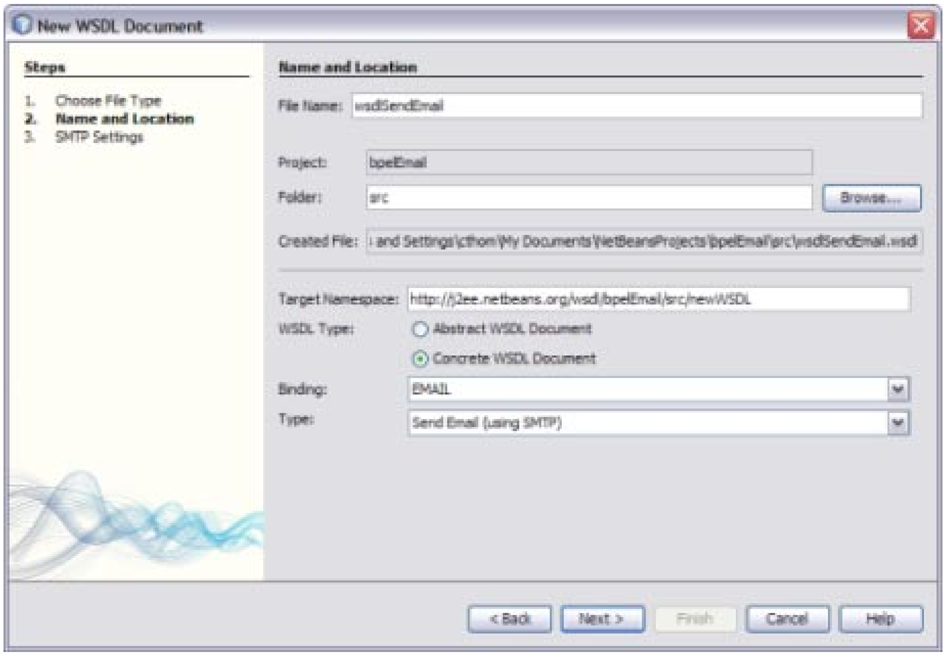
The New WSDL Document Wizard appears.

3 **Enter a name for the WSDL document, and verify or update the folder location for the file.**

4 **Select Concrete WSDL Document.**

The Binding and Type fields appear.

5 **For the Binding, select EMAIL; for the Type, select Send Email (using SMTP).**



Name and Location

File Name: wsdlSendEmail

Project: bpelEmail

Folder: src

Created File: : and Settings\cthon\My Documents\NetBeansProjects\bpelEmail\src\wsdlSendEmail.wsdl

Target Namespace: http://j2ee.netbeans.org/wsdl/bpelEmail/src/newWSDL

WSDL Type: ○ Abstract WSDL Document [✓ Concrete WSDL Document]

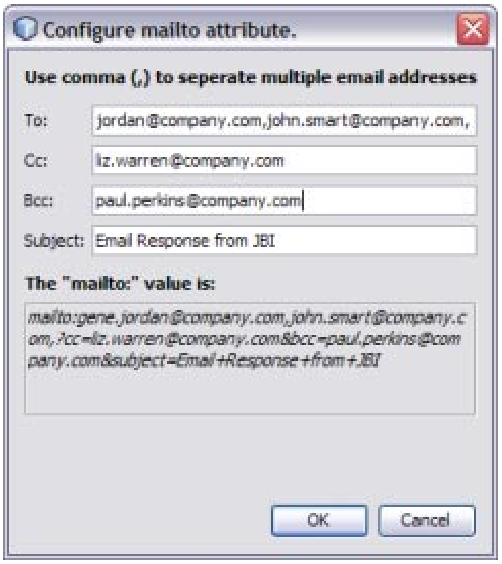
Binding: EMAIL

Type: Send Email (using SMTP)

6 **Click Next.**

The SMTP Settings window appears.

7 For the Mail To field, click "mailto:" fields, and, in the dialog that appears, enter the email addresses and email subject for the email messages sent by the eMail Binding Component.


Configure mailto attribute.

Use comma (,) to seperate multiple email addresses

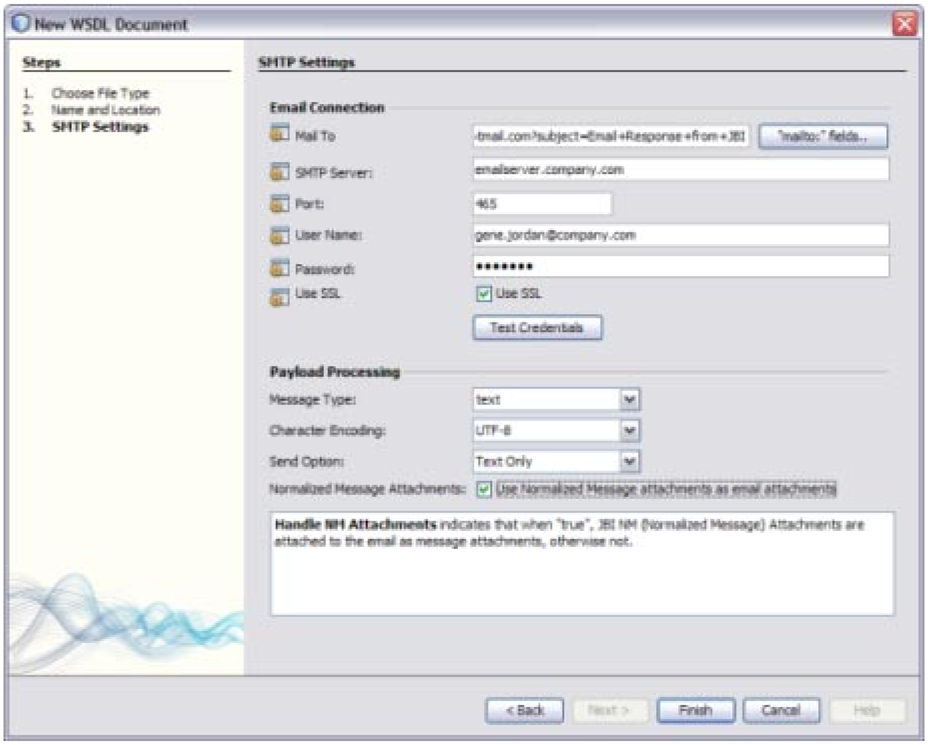
To:	jordan@company.com,john.smart@company.com,
Cc:	liz.warren@company.com
Bcc:	paul.perkins@company.com
Subject:	Email Response from JBI

The "mailto:" value is:

```
mailto:gene.jordan@company.com,john.smart@company.c
om,?cc=liz.warren@company.com&bcc=paul.perkins@com
pany.com&subject=Email+Response+from+JBI
```

8 Fill in the remaining settings according to your email client's SMTP server.

For information about the SMTP Settings fields, see "New WSDL Wizard Properties for the eMail BC" on page 14.



SMTP Settings

Email Connection

Mail To: -tmail.com?subject=Email+Response+from+JBI
"mailto:" fields...
SMTP Server: emailserver.company.com
Port: 465
User Name: gene.jordan@company.com
Password: ••••••••
Use SSL: ✓ Use SSL
Test Credentials

Payload Processing

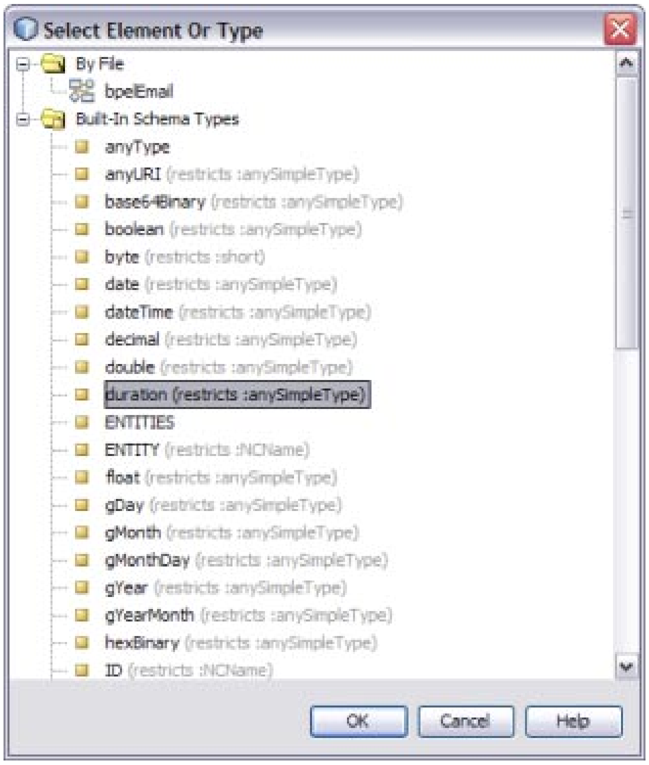
Message Type: text
Character Encoding: UTF-8
Send Option: Text Only
Normalized Message Attachments: ✓ Use Normalized Message attachments as email attachments

Handle NM Attachments indicates that when "true", JBI NM (Normalized Message) Attachments are attached to the email as message attachments, otherwise not.

Note – When you select different message types, the available Payload Processing fields change to accommodate the type.

---

9 **If you select a message type of XML or encoded data, you need to specify the element or type to use. Click the ellipsis button to display a list of options to select from.**


Select Element Or Type
By File
bpelEmail
Built-In Schema Types
anyType
anyURI (restricts :anySimpleType)
base64Binary (restricts :anySimpleType)
boolean (restricts :anySimpleType)
byte (restricts :short)
date (restricts :anySimpleType)
dateTime (restricts :anySimpleType)
decimal (restricts :anySimpleType)
double (restricts :anySimpleType)
duration (restricts :anySimpleType)
ENTITIES
ENTITY (restricts :NCName)
float (restricts :anySimpleType)
gDay (restricts :anySimpleType)
gMonth (restricts :anySimpleType)
gMonthDay (restricts :anySimpleType)
gYear (restricts :anySimpleType)
gYearMonth (restricts :anySimpleType)
hexBinary (restricts :anySimpleType)
ID (restricts :NCName)
OK
Cancel
Help

10 **If you select a message type of encoded data, you need to specify the encoding style used; for example, hl7encoder-1.0 for HL7 encoding, customencoder-1.0 for a custom encoding, or cocoencoder-1.0 for COBOL Copybook encoding.**

11 **After you enter the email server connection information, click Test Credentials to be sure you can connect to the server.**

12 **On the New WSDL Document Wizard, click Finish.**

New WSDL Wizard Properties for the eMail BC

The following tables list and describe the read and send properties for the eMail BC. These properties are accessed from the protocol settings page of the New WSDL Wizard.

TABLE 1 Read email Settings (POP3 and IMAP)

Property	Description
Email Server	The name of the computer that hosts the email server from which you want to read email messages.
Port	The port number of the email server.
User Name	The user name to use to connect to the email server.
Password	The password that corresponds to the user name entered above.
Authentication	An indicator of whether to use SSL authentication. Select the check box to use SSL; otherwise leave it deselected.
Polling Interval	The time period (in milliseconds) the eMail BC will wait between polling for new incoming email messages.
Mail Folder	The name of the email folder from which to read messages.
Maximum Message Count	The maximum number of messages to return from the email server.
Message Ack Mode	An indicator of how the message acknowledgement is handled based on the response from the Normalized Message Router (NMR). Choose either automatic or manual. When the method is automatic, you need to select an operation to perform.
Message Ack Operation	The type of operation to perform on the incoming email messages upon message acknowledgement. Select either markAsRead or delete.
Attachments Directory	A local directory where you want to store any attachments from the incoming email messages being processed. The file name is derived from the attachment name.

TABLE 2 Send email Settings (SMTP)

Property	Description
Mail To	The recipients of email messages sent from the eMail Binding Component. This field uses the standard mailto URL scheme (RFC 2368). Special characters must be encoded (see RFC 2368 and RFC 1378). The corresponding message parts (if any) take precedence over location headers.
SMTP Server	The name of the email server for sending email messages.
Port	The port number of the email server for sending email messages. This property is optional.
User Name	The user name to use to connect to the email server.
Password	The password that corresponds to the user name entered above.

TABLE 2 Send email Settings (SMTP) *(Continued)*

Property	Description
Use SSL	An indicator of whether to use SSL authentication. Select the check box to use SSL; otherwise leave it deselected.
Message Type	The type of message being processed. Select one of the following options: ■ **text** ■ **binary** ■ **xml** ■ **encoded data**
Character Encoding	The encoding to use for body and header text. Select the encoding type from the list of available options. This setting only appears when the Message Type is text.
Send Option	The format of the email content. Select one of the following options: ■ **Text Only** ■ **HTML Only** ■ **XML Only** ■ **Both Text and HTML** ■ **Both Text and XML**
Image Handling	An indicator of whether images are resolved and embedded in an HTML document (with the tag `<img src="`*absolute_URL_to_image*`">`. The URL must be the absolute URL in order to be embedded. This setting only appears when the Send Option is HTML or Both Text and HTML.
XSD Element/Type	The element or type from the schema definition (XSD) for XML or encoded data. Click the ellipsis button to select from a list of elements and types. Note that for encoded data, you must select an element and not a type. This setting only appears when the Message Type is xml or encoded data.
Encoding Style	The encoding style to use. This field only appears when the Message Type is encoded data.
Normalized Message Attachments	An indicator of whether normalized message attachments are included with the email as message attachments. Select the check box to include them as message attachments; otherwise leave the check box deselected.

Configuring eMail BC WSDL Attributes

Once you create the eMail BC WSDL document, you can add and update the WSDL attributes that are specific to the email configuration. The eMail BC includes both service-level WSDL elements and binding-level WSDL elements.

Perform the following tasks to configure the eMail Binding Component WSDL file after it is generated by the wizard.

- "To Configure eMail BC WSDL Attributes" on page 17
- "To Send email Messages with Attachments" on page 18

▼ To Configure eMail BC WSDL Attributes

1. **In the NetBeans IDE, double-click the WSDL document you want to configure.**

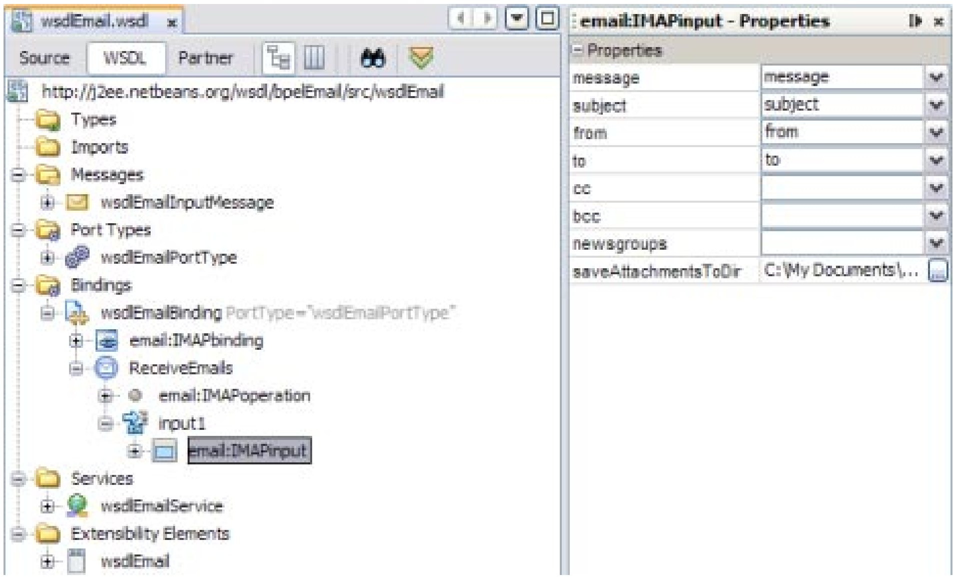
   The document appears in the WSDL Editor in WSDL view.

2. **To configure binding-level attributes, do the following:**

   a. **Expand the Bindings node, the binding name, the operation, and then the input.**

   b. **Click the email:* node.**

   The name of this node begins with "email:" and the rest varies depending on the type of protocol and whether it is receiving or sending. A list of properties appears in the right panel.



message	message
subject	subject
from	from
to	to
cc	
bcc	
newsgroups	
saveAttachmentsToDir	C:\My Documents\...

   c. **Modify the value of any of the properties.**

   For more information about these properties, see "Receiving eMail BC Attributes" on page 19.

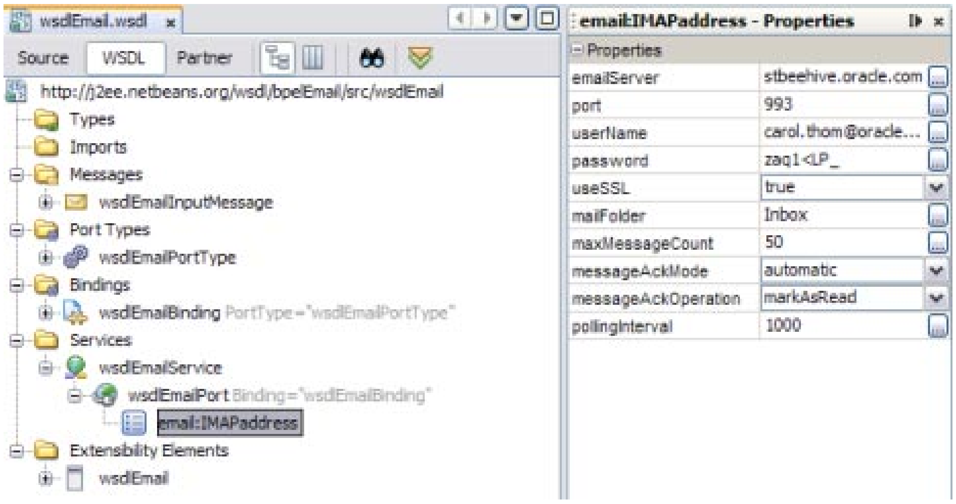
   d. **In the NetBeans toolbar, click Save.**

3 To configure service attributes, do the following:

a. Expand the Services node, the service name, and then the service port.

b. Click on the email address node.

The name of this node varies depending on the type of protocol. A list of properties appears in the right panel.



```
http://j2ee.netbeans.org/wsdl/bpelEmail/src/wsdlEmail
  Types
  Imports
  Messages
    wsdlEmailInputMessage
  Port Types
    wsdlEmailPortType
  Bindings
    wsdlEmailBinding PortType="wsdlEmailPortType"
  Services
    wsdlEmailService
      wsdlEmailPort Binding="wsdlEmailBinding"
        email:IMAPaddress
  Extensibility Elements
    wsdlEmail
```

Properties	
emailServer	stbeehive.oracle.com
port	993
userName	carol.thom@oracle...
password	zaq1<LP_
useSSL	true
mailFolder	Inbox
maxMessageCount	50
messageAckMode	automatic
messageAckOperation	markAsRead
pollingInterval	1000

c. Modify the value of any of the properties.

For more information about these properties, see "Sending eMail BC Attributes" on page 21.

d. In the NetBeans toolbar, click Save.

▼ To Send email Messages with Attachments

1 In the NetBeans IDE, double-click the WSDL document you want to configure.

The document appears in the WSDL Editor in WSDL view.

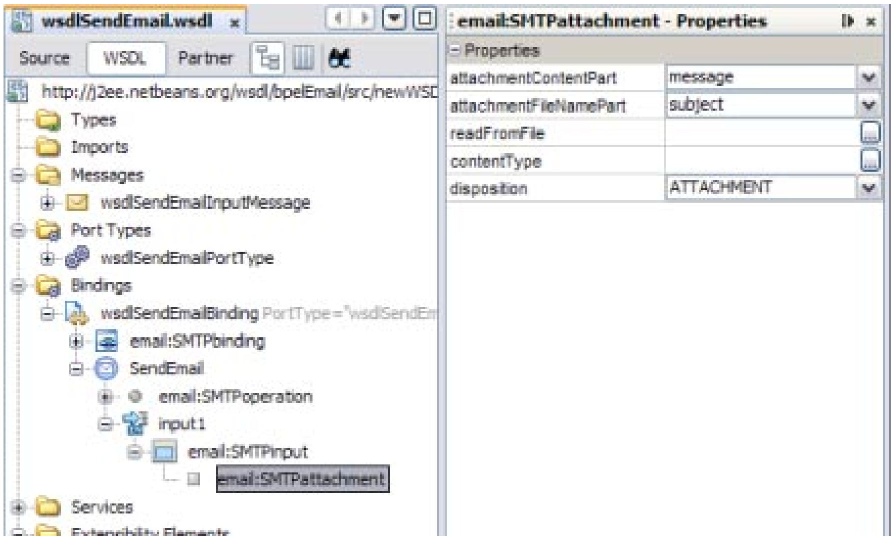
2 Expand the Bindings node, the binding name, the operation, and then the input.

3 Right-click the email:* node, and select Add SMTPattachment.

The name of the email: node varies depending on the type of protocol and whether it is receiving or sending.

4 In the Properties Panel on the right, enter values for the attachment properties.

For more information about these properties, see Table 7.



attachmentContentPart	message
attachmentFileNamePart	subject
readFromFile	
contentType	
disposition	ATTACHMENT

eMail Binding and Service Level WSDL Attributes

The service-level WSDL elements and attributes define connectivity to the email server along with certain processing information. The binding-level WSDL elements and attributes define and configure the operation performed along with information about the email messages and content.

Receiving eMail BC Attributes

The eMail Binding Component binding-level WSDL elements include the binding and services extensibility elements. When you create a WSDL file in the NetBeans IDE, the New WSDL Wizard generates both the binding element and the service elements based on information you supplied in the wizard.

For receiving email messages, the binding element defines where to find specific information in the incoming message and how to handle attachments. The binding element also specifies that the WSDL document is configured for an email server and also indicates the protocol. The service element defines connectivity information, such as the server name, port number, and user authentication. It also defines how often to check for messages, how many messages to poll, and how to handle acknowledgements.

TABLE 3 Binding–Level Attributes for Receiving email Messages

Attribute	Description
message	The name of the message part that contains the body of the email.
subject	The name of the message part that contains the subject line of the email.
from	The name of the message part that contains the email address of the sender.
to	The name of the message part that contains the list of email addresses of the recipients.
cc	The name of the message part that contains the list of recipients who are copied on the email.
bcc	The name of the message part that contains the list of recipients who are blind copied on the email.
newsgroups	The name of the message part that contains the names of recipient news groups.
saveAttachmentsToDir	A local directory where you want to store any attachments from the incoming email messages being processed. Attachments are only saved if this property is defined. The file name is derived from the attachment name.

The following example illustrates the binding element for receiving email messages from an IMAP server:

```
<binding name="wsdlEmailBinding" type="tns:wsdlEmailPortType">
   <email:IMAPbinding/>
   <operation name="ReceiveEmails">
      <email:IMAPoperation/>
      <input name="input1">
         <email:IMAPinput
            message="message"
            subject="subject"
            from="sender"
            to="recipients"
            saveAttachmentsToDir="C:\EmailProject\email\attachments"/>
      </input>
   </operation>
</binding>
```

TABLE 4 Service–Level Attributes for Receiving email Messages

Attribute	Description
emailServer	The name of the computer that hosts the email server from which you want to read email messages.
port	The port number of the server.
userName	The user name to use to connect to the email server.

TABLE 4 Service–Level Attributes for Receiving email Messages *(Continued)*

Attribute	Description
password	The password that corresponds to the user name entered above.
useSSL	An indicator of whether to use SSL authentication. Specify true to use SSL; otherwise specify false.
mailFolder	The name of the email folder from which to read messages.
maxMessageCount	The maximum number of messages to return from the email server.
messageAckMode	An indicator of how the message acknowledgement is handled based on the response from the Normalized Message Router (NMR). Choose either automatic or manual. When the method is automatic, you need to select an operation to perform.
messageAckOperation	The type of operation to perform on the incoming email messages upon message acknowledgement. Select either markAsRead or delete.
pollingInterval	The time period (in milliseconds) the eMail BC will wait between polling for new incoming email messages.

The following example illustrates the service element for receiving email messages from an IMAP server:

```
<service name="wsdlEmailService">
   <port name="wsdlEmailPort" binding="tns:wsdlEmailBinding">
      <email:IMAPaddress
         emailServer="email.mycompany.com"
         useSSL="true"
         port="994"
         userName="gsmythe@mycompany.com"
         password="1qazMKO"
         mailFolder="Inbox"
         maxMessageCount="50"
         messageAckMode="automatic"
         messageAckOperation="markAsUnread"
         pollingInterval="1000"/>
    </port>
</service>
```

Sending eMail BC Attributes

The sending eMail Binding Component binding-level WSDL elements include the binding and services extensibility elements. The eMail binding extensibility element specifies that the WSDL document is configured for an email server and also indicates the protocol. When you create a WSDL file in the NetBeans IDE, the New WSDL Wizard generates both the binding element and the service elements based on information you supplied in the wizard.

For sending email messages, the binding element defines where to find specific information in the incoming message, encoding options, character sets, and how to handle attachments and images. The service element defines connectivity information, such as the server name, port number, and user authentication.

TABLE 5 Binding–Level Attributes for Sending email Messages

Attribute	Description
message	The name of the message part that contains the body of the email.
subject	The name of the message part that contains the subject line of the email.
from	The name of the message part that contains the email address of the sender.
to	The name of the message part that contains the list of email addresses of the recipients.
cc	The name of the message part that contains the list of recipients who are copied on the email.
bcc	The name of the message part that contains the list of recipients who are blind copied on the email.
newsgroups	The name of the message part that contains the names of recipient news groups.
charset	The encoding to use for body and header text. This setting is only applicable when the message type is next.
use	Either literal or encoded.
encodingStyle	The encoding style to use. Only use this property when the use property is set to encoded.
sendOption	The format of the email content. Select one of the following options: ■ **Text Only** ■ **HTML Only** ■ **XML Only** ■ **Both Text and HTML** ■ **Both Text and XML**
embedImagedInHTML	An indicator of whether images are resolved and embedded in an HTML document (with the tag `<img src="`*absolute_URL_to_image*`">`. The URL must be the absolute URL in order to be embedded. Only use this property when the sendOption is HTML or Both Text and HTML.
handleNMAttachments	An indicator of whether normalized message attachments are included with the email as message attachments. Select the true to include them as message attachments; otherwise select false.

The following example illustrates the binding element for sending email messages from an SMTP server:

```
<binding name="wsdlSendEmailBinding" type="tns:wsdlSendEmailPortType">
   <email:SMTPbinding/>
   <operation name="SendEmail">
      <email:SMTPoperation
      <input name="input1">
         <email:SMTPinput
            message="message"
            subject="subject"
            from="sender"
            to="recipients"
            charset="Big5"
            use="literal"
            sendOption="Text Only"
            handleNMAttachments="true"/>
      </input>
   </operation>
</binding>
```

TABLE 6 eMail BC Send WSDL Service Attributes

Attribute	Description
location	The recipients of email messages sent from the eMail Binding Component. This field uses the standard mailto URL scheme (RFC 2368). Special characters must be encoded (see RFC 2368 and RFC 1378). The corresponding message parts (if any) take precedence over location headers.
emailServer	The name of the email server for sending email messages.
port	The port number for the email server.
userName	The user name to use to connect to the email server.
password	The password that corresponds to the user name entered above.
useSSL	An indicator of whether to use SSL authentication. Specify **true**to use SSL; otherwise specify **false**.

The following example illustrates the service element for sending email messages from an SMTP server:

```
<service name="wsdlSendEmailService">
   <port name="wsdlSendEmailPort" binding="tns:wsdlSendEmailBinding">
      <email:SMTPaddress
         emailServer="email.mycompany.com"
         location="mailto:msharpe@mycompany.com?subject=Response+from+JBI&
                   cc=tmiller@mycompany.com"
         useSSL="true"
         port="465"
         userName="gsmythe@mycompany.com"
         password="1qazMKO"
   </port>
</service>
```

TABLE 7 eMail BC SMTP Attachment WSDL Attributes

Attribute	Description
attachmentContentPart	The name of the message part that contains the attachment content. Select from the following options: ■ **from** ■ **to** ■ **subject** ■ **message** If both the `attachmentContentPart` and `readFromFile` attributes are defined, the `attachmentContentPart` takes precedence.
attachmentFileNamePart	The part of the message to use to name the attachment file. If no value is specified, then the value of the `attachmentContentPart` property is used for the file name.
readFromFile	The name of the file that contains the attachment content. If a value is specified for the `attachmentContentPart`, it overrides the value specified here.
contentType	The type of content in the attachment. It might be useful for the receiver to know what type of content is contained in the attachment in order to read it. Example of content types are text/html, message/rfc822, text/plain;charset=UTF-8, and so on. If the content type is invalid, unknown, or empty, the content is treated as application/octet-stream.
disposition	The disposition of the attachment in the content. Select either ATTACHMENT or INLINE.

The following example illustrates the binding element with the SMTP attachment configuration:

```
<binding name="wsdlSendEmailBinding" type="tns:wsdlSendEmailPortType">
   <email:SMTPbinding/>
   <operation name="SendEmail">
      <email:SMTPoperation
      <input name="input1">
         <email:SMTPinput
            message="message"
            subject="subject"
            from="sender"
            to="recipients"
            charset="Big5"
            use="literal"
            sendOption="Text Only"
            handleNMAttachments="true">
              <email:SMTPattachment attachmentContentPart="message"
                 attachmentFileNamePart="subject"
```

```
                disposition="ATTACHMENT"/>
         </email:SMTPinput>
      </input>
   </operation>
</binding>
```

Configuring the eMail Binding Component Runtime Properties

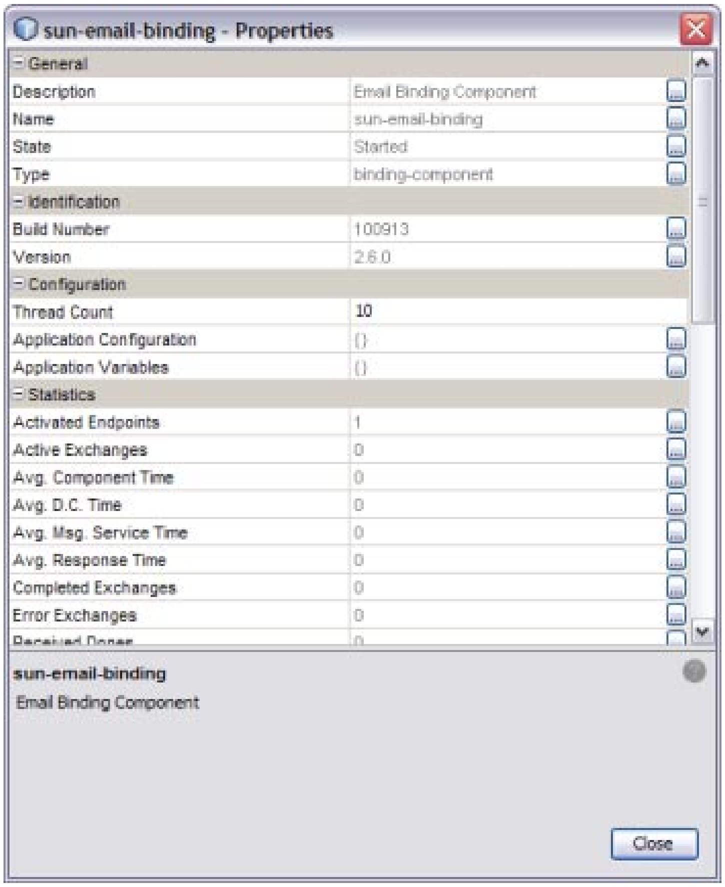
The eMail Binding Component's runtime properties apply to all instances of the binding component in a domain, including all provider and consumer endpoints. The properties can be configured from the NetBeans IDE, GlassFish Admin Console, or from a command prompt during a command line installation. This section describes how to configure the properties in NetBeans.

▼ To Configure eMail Binding Component Runtime Properties

1. **From the Services window of the NetBeans IDE, expand the Servers node.**
2. **If the application server is not already started, right-click the server and then select Start.**
3. **Under the application server, expand JBI and expand Binding Components.**
4. **If the eMail Binding Component is not started, right-click sun-email-binding and select Start.**

5 **Right-click sun-email-binding and select Properties.**

The Properties Editor appears.



General	
Description	Email Binding Component
Name	sun-email-binding
State	Started
Type	binding-component
Identification	
Build Number	100913
Version	2.6.0
Configuration	
Thread Count	10
Application Configuration	{}
Application Variables	{}
Statistics	
Activated Endpoints	1
Active Exchanges	0
Avg. Component Time	0
Avg. D.C. Time	0
Avg. Msg. Service Time	0
Avg. Response Time	0
Completed Exchanges	0
Error Exchanges	0

sun-email-binding

Email Binding Component

6 **Modify any of the properties listed in "eMail Binding Component Runtime Property Descriptions" on page 27.**

Note – General, identification, and statistic properties are automatically updated by the eMail BC. You do not need to modify these properties.

---

7 **To apply the changes, stop and restart the eMail BC.**

eMail Binding Component Runtime Property Descriptions

The eMail Binding Component properties specify the number of threads and reference descriptive information for the Binding Component. You can also use these properties to set application variables and configure application properties. The following tables list and describe each eMail Binding Component runtime property.

TABLE 8 eMail Binding Component General Runtime Properties

Property	Description
Description	A general description of the JBI component.
Name	A unique name for the eMail BC in the JBI environment. If you install more than one eMail Binding Component in a JBI environment, make sure that each has a unique name. When the service unit deploys the component, it is matched with target component name defined in its descriptor file, `jbi.xml`, which can be modified as needed.
State	The current state of the JBI component. This value can be either Started, Stopped, or Shutdown.
Type	The type of JBI component (service-engine or binding-component).

TABLE 9 eMail Binding Component Identification Runtime Properties

Property	Description
Version	The version number of the installed binding component.
Build Number	The build number of the installed binding component.

TABLE 10 eMail Binding Component Configuration Runtime Properties

Property	Description
Thread Count	The number of threads listening on the Normalized Message Router (NMR) for message exchanges. The default value is 10.
Application Configuration	A list of values for a Composite Application's endpoint connectivity parameters, which are normally defined in the WSDL service extensibility elements. The values are applied to a user-named endpoint Config Extension Property. The values defined in an application configuration override the values defined in the WSDL document. For more information, see "Creating Application Configurations for Connectivity Parameters" on page 30.

TABLE 10 eMail Binding Component Configuration Runtime Properties *(Continued)*

Property	Description
Application Variables	A list of name and value pairs for a given type. The application variable name can be used as a token for a WSDL extensibility element attribute in a corresponding binding. For more information, see "Using Application Variables" on page 33.

TABLE 11 eMail Binding Component Runtime Statistics

Property	Description
Activated Endpoints	The number of activated endpoints.
Active Exchanges	The number of active exchanges.
Avg. Component Time	The average message exchange component time in milliseconds.
Avg. D.C. Time	The average message exchange delivery channel time in milliseconds.
Avg. Msg. Service Time	The average message exchange message service time in milliseconds.
Avg. Response Time	The average message exchange response time in milliseconds.
Completed Exchanges	The total number of completed exchanges.
Error Exchanges	The total number of error exchanges.
Received Dones	The total number of received dones.
Received Errors	The total number of received errors.
Received Faults	The total number of received faults.
Received Replies	The total number of received replies.
Received Requests	The total number of received requests.
Sent Dones	The total number of sent dones.
Sent Errors	The total number of sent errors.
Sent Faults	The total number of sent faults.
Sent Replies	The total number of sent replies.
Sent Requests	The total number of sent requests.
Up Time	The up time of this component in milliseconds.

The Loggers properties specify the level of logging for each event. You can set the logging level for each logger to any of the following levels:

- FINEST: provides highly detailed tracing
- FINER: provides more detailed tracing
- FINE: provides basic tracing
- CONFIG: provides static configuration messages
- INFO: provides informative messages
- WARNING: messages indicate a warning
- SEVERE: messages indicate a severe failure
- OFF: no logging messages

By default, these are all set to the INFO level.

TABLE 12 eMail Binding Component Logger Runtime Properties

Property	eMail BC Component
sun-email-binding	com.sun.jbi.binding.email
EmailBC Component Manager	com.sun.jbi.binding.email.EmailBCComponentManager
EmailBC Configuration	com.sun.jbi.binding.email.EmailBCConfiguration
EmailBC Endpoint Manager	com.sun.jbi.binding.email.EmailBCEndpointManager
EmailBC Exchange Handler	com.sun.jbi.binding.email.EmailBCExchangeHandler
Deployment Lookup (QoS)	com.sun.jbi.common.qos.descriptor.DeploymentLookup
Messaging Channel (QoS)	com.sun.jbi.common.qos.messaging.MessagingChannel
EndpointLifeCycle	com.sun.jbi.component.toolkit.endpoint.EndpointLifeCycle
AcceptPoller	com.sun.jbi.component.toolkit.lifecycle. impl.AcceptPoller
DefaultServiceUnitManager	com.sun.jbi.component.toolkit.lifecycle. impl.DefaultServiceUnitManager
PollerConfig	ccom.sun.jbi.component.toolkit.lifecycle. impl.PollerConfig
EmailBC Message Receiver (IMAP)	com.sun.jbi.binding.email.protocol.receive.imap. IMAPMessageReceiver
EmailBC Message Receiver (POP3)	com.sun.jbi.binding.email.protocol.receive.pop3. POP3MessageReceiver
EmailBC Message Sender (SMTP)	com.sun.jbi.binding.email.protocol.send.smtp. SMTPMessageSender

Creating Application Configurations for Connectivity Parameters

Application Configurations allow you to configure the external connectivity parameters for a JBI application and, without changing or rebuilding the application, deploy the same application to a different system. For example, if you have an application that is running in a test environment, you can deploy it to a production environment using new connectivity parameters without rebuilding the application.

The connectivity parameters for the eMail Binding Component are normally defined in the WSDL service extensibility elements. When you create and apply application configurations for these parameters, the values defined for the application configuration override the values defined in the WSDL elements. You apply the configurations to the Composite Application by entering the application configuration name in the Config Extension Name property for the appropriate endpoint in the Service Assembly.

Perform the following procedures to implement application configurations for the eMail Binding Component:

- "To Create Application Configurations" on page 30
- "To Add the Application Configuration to the Endpoint" on page 31

Once you create an application configuration, you can modify it as described in "To Change Application Configuration Values" on page 32.

▼ To Create Application Configurations

You can create several application configurations, which are all referenced by the names you define. Make sure the eMail Binding Component is started before you begin this procedure.

1 **On the NetBeans IDE Services window, expand Servers > GlassFish v2.x > JBI > Binding Components.**

2 **Right-click sun-email-binding, and then select Properties.**

The Properties window appears.

3 **Click the ellipsis next to Application Configuration.**

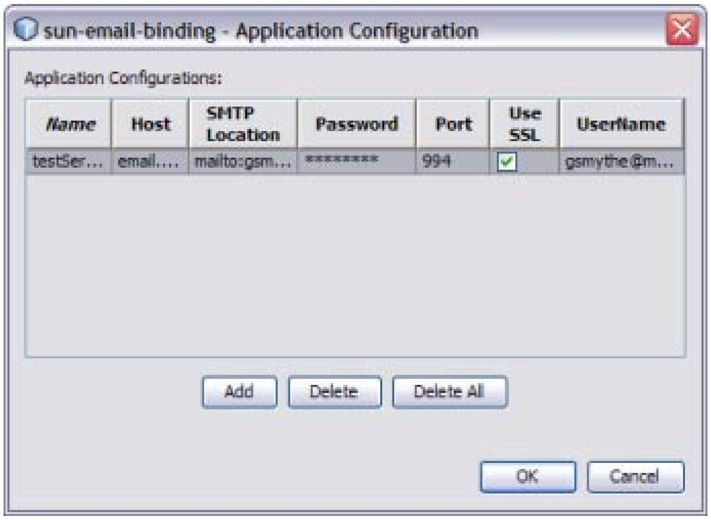
The Application Configuration Editor appears.

4 **Click Add.**

A new row appears in the configuration list.

5 **In the new row, enter a name for this application configuration, and fill in any of the following fields:**

- Host
- SMTP Location
- Password
- Port
- UseSSL
- UserName


sun-email-binding - Application Configuration
Application Configurations:
Add
Delete
Delete All
OK
Cancel

Name	Host	SMTP Location	Password	Port	Use SSL	UserName
testSer...	email....	mailto:gsm...	********	994	✓	gsmythe@m...

6 **Click OK to Close the Application Configuration Editor.**

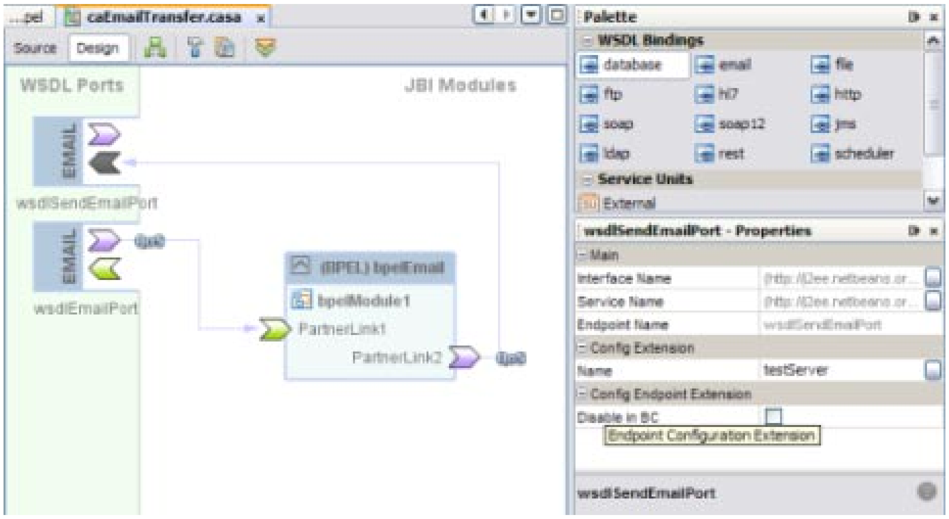
▼ To Add the Application Configuration to the Endpoint

1 **Open the Composite Application Service Assembly in the CASA Editor.**

2 **Select the endpoint to which you want to add the application configuration.**

The Properties Panel appears to the right of the CASA Editor canvas.

3 **In the Name property under Config Extension, enter the name of the application configuration to use for the service assembly.**



4 **Click Save All on the NetBeans toolbar.**

5 **Deploy the application.**

▼ To Change Application Configuration Values

The eMail Binding Component must be started in order to perform this procedure.

1 **On the NetBeans IDE Services window, expand Servers > GlassFish v2.x > JBI > Binding Components.**

2 **Right-click sun-email-binding, and then select Properties.**

The Properties window appears.

3 **Click the ellipsis next to Application Configuration.**

The Application Configuration Editor appears.

4 **Change the value of any of the fields in any of the application configurations.**

5 **To apply the new values, stop and restart any service assemblies that use the application configurations you updated.**

Using Application Variables

Application variables allow you to define a list of variable names and values along with their type. The application variable name can then be used as a token in a WSDL extensibility element attribute for the eMail Binding Component. For example, you could define a string variable named **`ServerName`** with a value of **`MyHost.com`**. To reference this in the WSDL document, you would enter **`${ServerName}`**. When you deploy an application that uses application variables, any variable that is referenced in the application's WSDL document is loaded automatically.

Note – If you start an application and a value is not defined for an application variable, an exception is thrown.

You can define the following four variable types:

- **String** – A string value, such as a path or directory.
- **Number** – A numeric value.
- **Boolean** – A Boolean true or false. When you define a Boolean variable, a check box appears in the value field. Select the check box if the variable value should be true; otherwise, deselect the check box.
- **Password** – A login password. The password is masked and appears as asterisks.

Variables allow greater flexibility in WSDL documents. For example, you can use the same WSDL document for multiple runtime environments by using application variables to specify system-specific information. The variable values can be changed from the binding component runtime properties for each specific environment.

▼ To Create an Application Variable

1. **On the NetBeans IDE Services window, expand Servers > GlassFish v2.x > JBI > Binding Components.**

2. **Right-click sun-email-binding, and then select Properties.**

   The Properties window appears.

3. **Click the ellipsis next to Application Variables.**

   The Application Variables Editor appears.

4. **Click Add.**

   A list of possible variable types appears.

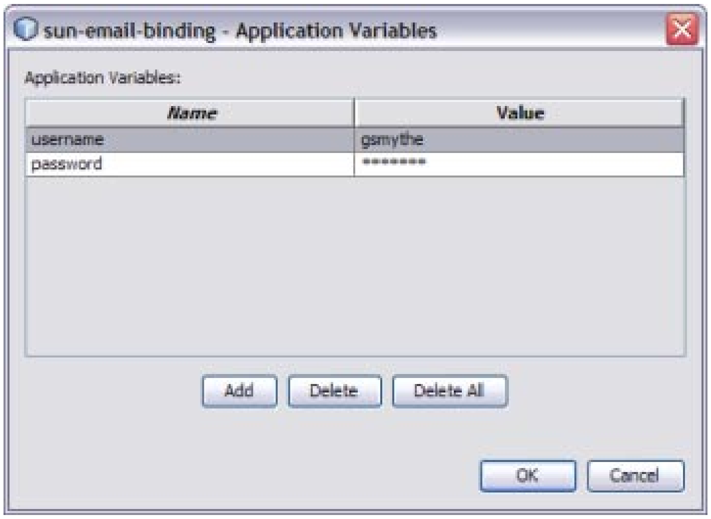
5 **Select String, Number, Boolean, or Password, and then click OK.**

A new row appears in the application list.

6 **In the new row, enter a variable name and then do one of the following:**

- **For a Boolean variable, select the check box if the variable value should be true; otherwise leave it deselected.**
- **For all other variables types, enter the variable value.**

Note – If you created a password variable, the value you enter appears as asterisks.

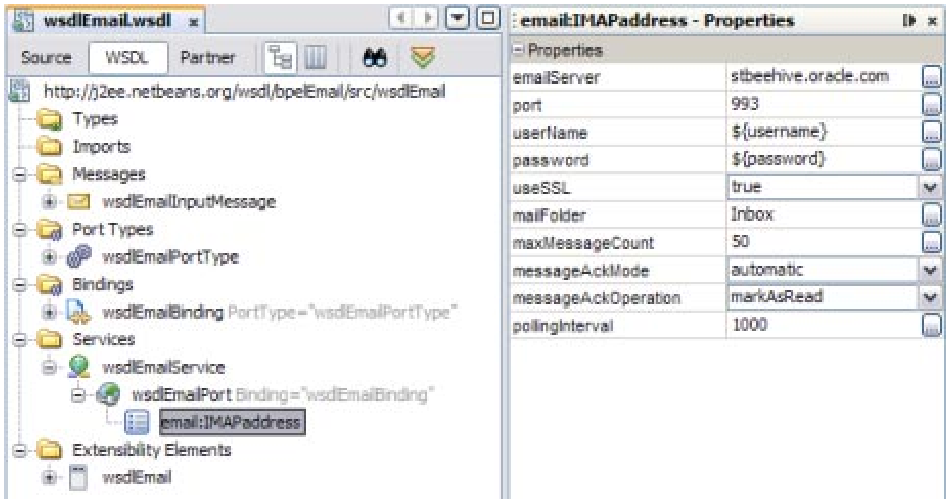


Name	Value
username	gsmythe
password	********

7 **On the Application Variable Editor, click OK; on the Properties window, click Close.**

8 **Reference the variable from the appropriate extensibility element in the WSDL document, using a dollar sign and curly brackets to indicate the variable; for example, `${MyVariable}`.**

You can see an example of this in the userName and password fields in the following image.



emailServer	stbeehive.oracle.com
port	993
userName	${username}
password	${password}
useSSL	true
mailFolder	Inbox
maxMessageCount	50
messageAckMode	automatic
messageAckOperation	markAsRead
pollingInterval	1000

Here is an example of how the username and password variables in the above images appear in the WSDL source:

```
<service name="wsdlEmailService">
   <port name="wsdlEmailPort" binding="tns:wsdlEmailBinding">
      <email:IMAPaddress
         emailServer="email.mycompany.com"
         useSSL="true"
         port="994"
         userName="${username}"
         password="${password}"
         mailFolder="Inbox"
         maxMessageCount="50"
         messageAckMode="automatic"
         messageAckOperation="markAsUnread"
         pollingInterval="1000"/>
    </port>
</service>
```

▼ To Change an Application Variable Value When the Application is Running

1 **On the NetBeans IDE Services window, expand Servers > GlassFish v2.x > JBI > Binding Components.**

2 **Right-click sun-email-binding, and then select Properties.**

The Properties window appears.

3 **Click the ellipsis next to Application Variables.**

The Application Variables Editor appears.

4 **In the variable list, select the value you want to change and then enter the new value.**

5 **On the Application Variable Editor, click OK; on the Properties window, click Close.**

6 **On the Services window, expand Servers > GlassFish v2.x > Service Assemblies.**

7 **Right-click the Service Assembly using the application variables, and click Stop.**

8 **Right-click the Service Assembly again, and click Start.**

The new settings take effect.

▼ To Use an Application Variable for Password Protection

To protect passwords that would otherwise appear as clear text in your WSDL document, you can define a Password application variable to use in the WSDL file as a token. In the following example, a password application variable is created that uses the name SECRET and the password PROTECT.

1 **On the NetBeans IDE Services window, expand Servers > GlassFish v2.x > JBI > Binding Components.**

2 **Right-click sun-email-binding, and then select Properties.**

The Properties window appears.

3 **Click the ellipsis next to Application Variables.**

The Application Variables Editor appears.

4 **Click Add, select Password, and then click OK.**

A new row appears in the variable list.

5 **Enter `SECRET` as the name, and enter `PROTECT` as the value.**

Because this is a password type, the characters appear as asterisks.

6 **Use the application variable name `${SECRET}` as the WSDL password attribute, using the dollar sign and curly braces.**

Using eMail BC Normalized Message Properties in a Business Process

You can define normalized message properties in a BPEL process in order to dynamically assign values to the runtime properties for the eMail Binding Component. The normalized message properties for each JBI component are accessed from the BPEL Designer Mapper view. When you expand a variable's Properties folder it exposes the variable's predefined NM properties, as well as the standard BPEL-specific WSDL properties used in correlation sets, assigns, and expressions . If the specific NM property you need is not currently listed, additional NM properties can be added.

Normalized message properties provide the following capabilities:

- Getting and setting transport context properties.
- Getting and setting request parameters.
- Dynamically configuring eMail properties.

Using Predefined Normalized Message Properties

Predefined normalized message properties are automatically available from the BPEL Designer's Mapper view. You can use additional properties by adding them directly to the source code. You can either define these properties using the BPEL Designer Mapper, or by entering the code directly into the source view.

You can perform additional tasks when working with normalized message properties, such as creating additional properties, deleting properties, creating property shortcuts, and so on. For more information, see "Using Normalized Message Properties" in *Oracle Java CAPS BPEL Designer and Service Engine User's Guide*.

▼ To Define Normalized Message Properties in Mapper View

1 **Open the BPEL process you want to edit in the BPEL Designer.**

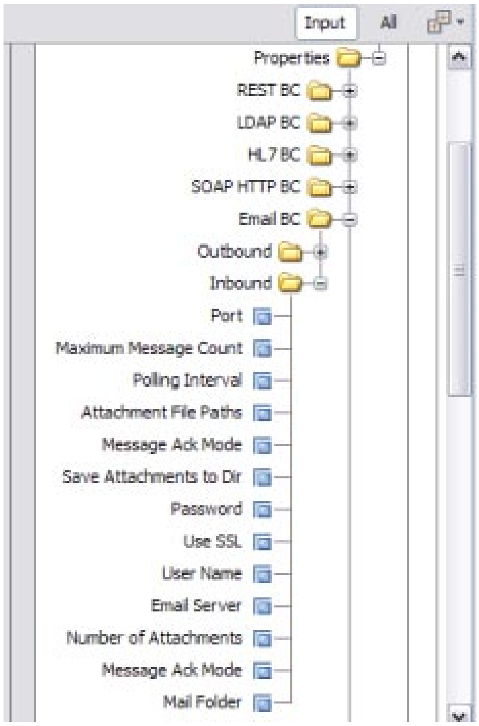
2 **In Design view, select the activity to add the normalized message property to.**

3 **In the BPEL Designer toolbar, click Mapper.**

4 **In the Output pane, expand the variable you want to edit, expand Properties, and then expand Email BC.**

5 **Expand Outbound or Inbound, depending on the message type.**

A list of available normalized message properties appears.



6 **Select the normalized message property you want to use, and use the mapper operands to build an expression or assign a value.**

For a complete list of normalized message properties for the eMail Binding Component, see "Normalized Message Properties for the eMail Binding Component" on page 39.

▼ To Define Normalized Message Properties in Source View

1 **Open the BPEL process you want to edit in the BPEL Designer.**

2 **In the BPEL Designer toolbar, click Source.**

The BPEL source code for the process is now visible.

3 **Declare the `sxnmp` namespace near the beginning of the `process` element; for example:**

```
xmlns:sxnmp="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/NMProperty"
```

4 **Define the property using the property names listed and described in "Normalized Message Properties for the eMail Binding Component" on page 39.**

For example,

```
<copy>
   <from>'true'</from>
   <to variable="SendEmailIn"
       sxnmp:nmProperty="org.glassfish.openesb.email.inbound.usessl"/>
</copy>
```

Normalized Message Properties for the eMail Binding Component

Normalized message properties are either specific to the binding component being used or generally available to all participating JBI components. The following topics describe both types of normalized message properties.

- "General Normalized Message Properties" on page 39
- "eMail Binding Component Normalized Message Properties" on page 40

General Normalized Message Properties

The following table lists and described the general properties that are available to all JBI components. All property values are of the type `java.lang.String`.

TABLE 13 General Normalized Message Properties

Property Name in Source	Property Name in Mapper	Description and Use
org.glassfish.openesb.messaging.groupid	Group ID	Uniquely identifies a message with the group to which a message belongs. This property is optional.
org.glassfish.openesb.messaging.messageid	Message ID	Uniquely identifies a message. For batch processing this might be a record number (for example, a particular record in a file) or a GUID. This property is mandatory.
org.glassfish.openesb.messaging.lastrecord	Last Record	The value is a string representation of boolean ("true" or "false"). This property can be used to signal the last record in a group or the last record in a file. This property is mandatory.
org.glassfish.openesb.exchange.endpointname	Endpoint Name	The value a string representation of the endpoint name set on the exchange. This represents the endpoint name of the "owner" of the message, and could be made available by JBI runtime.

eMail Binding Component Normalized Message Properties

The following properties are specific to the eMail Binding Component. Available properties are different for request messages than for response messages. All property values are of the type `java.lang.String`, except the `Number of Attachments`, which is an integer.

TABLE 14 eMail Binding Component NM Properties (Inbound)

Property Name in Source	Property Name in Mapper	Description
org.glassfish.openesb. email.inbound.emailserver	Email Server	The name of the computer that hosts the email server used for sending or receiving email messages.
org.glassfish.openesb. email.inbound.port	Port	The port number for the email server.
org.glassfish.openesb. email.inbound.username	User Name	The user name to use for authentication on the email server.
org.glassfish.openesb. email.inbound.password	Password	The password to use for authentication on the email server.
org.glassfish.openesb. email.inbound.usessl	Use SSL	An indicator of whether to use SSL for secure connections. Specify either **`true`** or **`false`**.
org.glassfish.openesb. email.inbound. maxmessagecount	Maximum Message Count	The maximum number of message to retrieve for each polling interval.
org.glassfish.openesb. email.inbound.ackmode	Message Ack Mode	The message acknowledgement mode for the response. Specify either **`automatic`** or **`manual`**. For automatic acknowledgements, you need to specify an operation as well.
org.glassfish.openesb. email.inbound. ackoperation	Message Ack Mode	The operation to perform when a message is acknowledged (if the Ack Mode is automatic). Specify either **`markAsRead`** or **`delete`**. **Note –** This is the second Message Ack Node property in the list that appears on the BPEL Mapper.
org.glassfish.openesb. email.inbound. pollinginterval	Polling Interval	The time period (in milliseconds) the eMail BC will wait between polling for new incoming email messages.
org.glassfish.openesb. email.inbound. saveattachmentstodir	Save Attachments to Dir	A local directory where you want to store any attachments from the incoming email messages being processed. The file name is derived from the attachment name.
org.glassfish.openesb. email.inbound.mailfolder	Mail Folder	The name of the email folder from which to read messages.

TABLE 14 eMail Binding Component NM Properties (Inbound) *(Continued)*

Property Name in Source	Property Name in Mapper	Description
org.glassfish.openesb. email.inbound.attachments	Attachments File Paths	The path to the attachment files.
org.glassfish.openesb. email.inbound. attachments.count	Number of Attachments	The number of attachments.

TABLE 15 eMail Binding Component NM Properties (Outbound)

Property Name in Source	Property Name in Mapper	Description
org.glassfish.openesb. email.outbound. emailserver	Email Server	The name of the computer that hosts the email server to which you want to send email messages.
org.glassfish.openesb. email.outbound.port	Port	The port number of the email server.
org.glassfish.openesb. email.outbound.username	User Name	The user name to use for authentication on the email server.
org.glassfish.openesb. email.outbound.password	Password	The password to use for authentication on the email server.
org.glassfish.openesb. email.outbound.usessl	Use SSL	An indicator of whether to use SSL for secure connections. Specify either **`true`** or **`false`**.
org.glassfish.openesb. email.outbound.location	Location	The recipients of email messages sent from the eMail Binding Component. This property uses the standard mailto URL scheme (RFC 2368). Special characters must be encoded (see RFC 2368 and RFC 1378). The corresponding message parts (if any) take precedence over location headers.
org.glassfish.openesb. email.outbound.sendoption	Send Option	The format of the email content. Specify one of the following options: ■ **Text Only** ■ **HTML Only** ■ **XML Only** ■ **Both Text and HTML** ■ **Both Text and XML** If no value is specified, the default is Text Only.

TABLE 15 eMail Binding Component NM Properties (Outbound) *(Continued)*

Property Name in Source	Property Name in Mapper	Description
org.glassfish.openesb. email.outbound. embedimagesinhtml	Embed Images in HTML	An indicator of whether images are resolved and embedded in an HTML document (with the tag `<img src="`*absolute_URL_to_image*`">`. The URL must be the absolute URL in order to be embedded. Only use this property when the Send Option is HTML Only or Both Text and HTML.
org.glassfish.openesb. email.outbound.usessl attachmentFilePaths	Attachment File Paths	A local directory where you want to store any attachments from the incoming email messages being processed. The file name is derived from the attachment name.