JAVAMAIL API - SMTP SERVERS

SMTP is an acronym for **Simple Mail Transfer Protocol**. It is an Internet standard for electronic mail $e-mail$ transmission across Internet Protocol IP networks. SMTP uses TCP port 25. SMTP connections secured by SSL are known by the shorthand SMTPS, though SMTPS is not a protocol in its own right.

JavaMail API has package **com.sun.mail.smtp** which act as SMTP protocol provider to access an SMTP server. Following table lists the classes included in this package:

Class	Description
SMTPMessage	This class is a specialization of the MimeMessage class that allows you to specify various SMTP options and parameters that will be used when this message is sent over SMTP.
SMTPSSLTransport	This class implements the Transport abstract class using SMTP over SSL for message submission and transport.
SMTPTransport	This class implements the Transport abstract class using SMTP for message submission and transport.

The following table lists the exceptions thrown:

Exception	Description
SMTPAddressFailedException	This exception is thrown when the message cannot be sent.
SMTPAddressSucceededException	This exception is chained off a SendFailedException when the *mail.smtp.reportsuccess* property is true.
SMTPSenderFailedException	This exception is thrown when the message cannot be sent.
SMTPSendFailedException	This exception is thrown when the message cannot be sent.The exception includes the sender's address, which the mail server rejected.

The **com.sun.mail.smtp** provider use SMTP Authentication optionally. To use SMTP authentication you'll need to set the *mail.smtp.auth* property or provide the SMTP Transport with a username and password when connecting to the SMTP server. You can do this using one of the following approaches:

- Provide an Authenticator object when creating your mail Session and provide the username and password information during the Authenticator callback. *mail.smtp.user* property can be set to provide a default username for the callback, but the password will still need to be supplied explicitly. This approach allows you to use the static Transport send method to send messages. For example:

```
Transport.send(message);
```

- Call the Transport connect method explicitly with username and password arguments. For example:

```
Transport tr = session.getTransport("smtp");
```

```
    tr.connect(smtphost, username, password);
    msg.saveChanges();
    tr.sendMessage(msg, msg.getAllRecipients());
    tr.close();
```

The SMTP protocol provider supports the following properties, which may be set in the JavaMail
Session object. The properties are always set as strings. For example:

```
    props.put("mail.smtp.port", "587");
```

Here the **Type** column describes how the string is interpreted.

Name	Type	Description
mail.smtp.user	String	Default user name for SMTP.
mail.smtp.host	String	The SMTP server to connect to.
mail.smtp.port	int	The SMTP server port to connect to, if the connect method doesn't explicitly specify one. Defaults to 25.
mail.smtp.connectiontimeout	int	Socket connection timeout value in milliseconds. Default is infinite timeout.
mail.smtp.timeout	int	Socket I/O timeout value in milliseconds. Default is infinite timeout.
mail.smtp.from	String	Email address to use for SMTP MAIL command. This sets the envelope return address. Defaults to msg.getFrom or InternetAddress.getLocalAddress.
mail.smtp.localhost	String	Local host name used in the SMTP HELO or EHLO command. Defaults to InetAddress.getLocalHost.getHostName. Should not normally need to be set if your JDK and your name service are configured properly.
mail.smtp.localaddress	String	Local address *hostname* to bind to when creating the SMTP socket. Defaults to the address picked by the Socket class. Should not normally need to be set.
mail.smtp.localport	int	Local port number to bind to when creating the SMTP socket. Defaults to the port number picked by the Socket class.
mail.smtp.ehlo	boolean	If false, do not attempt to sign on with the EHLO command. Defaults to true.
mail.smtp.auth	boolean	If true, attempt to authenticate the user using the AUTH command. Defaults to false.
mail.smtp.auth.mechanisms	String	If set, lists the authentication mechanisms to consider. Only mechanisms supported by the server and supported by the current implementation will be used. The default is "LOGIN PLAIN DIGEST-MD5 NTLM", which includes all the authentication mechanisms supported by the current implementation.
mail.smtp.auth.login.disable	boolean	If true, prevents use of the AUTH LOGIN command. Default is false.

mail.smtp.auth.plain.disable	boolean	If true, prevents use of the AUTH PLAIN command. Default is false.
mail.smtp.auth.digest-md5.disable	boolean	If true, prevents use of the AUTH DIGEST-MD5 command. Default is false.
mail.smtp.auth.ntlm.disable	boolean	If true, prevents use of the AUTH NTLM command. Default is false.
mail.smtp.auth.ntlm.domain	String	The NTLM authentication domain.
mail.smtp.auth.ntlm.flags	int	NTLM protocol-specific flags.
mail.smtp.submitter	String	The submitter to use in the AUTH tag in the MAIL FROM command. Typically used by a mail relay to pass along information about the original submitter of the message.
mail.smtp.dsn.notify	String	The NOTIFY option to the RCPT command. Either NEVER, or some combination of SUCCESS, FAILURE, and DELAY *separatedbycommas*.
mail.smtp.dsn.ret	String	The RET option to the MAIL command. Either FULL or HDRS.
mail.smtp.sendpartial	boolean	If set to true, and a message has some valid and some invalid addresses, send the message anyway, reporting the partial failure with a SendFailedException. If set to false *thedefault*, the message is not sent to any of the recipients if there is an invalid recipient address.
mail.smtp.sasl.enable	boolean	If set to true, attempt to use the javax.security.sasl package to choose an authentication mechanism for login. Defaults to false.
mail.smtp.sasl.mechanisms	String	A space or comma separated list of SASL mechanism names to try to use.
mail.smtp.sasl.authorizationid	String	The authorization ID to use in the SASL authentication. If not set, the authentication ID *username* is used.
mail.smtp.sasl.realm	String	The realm to use with DIGEST-MD5 authentication.
mail.smtp.quitwait	boolean	If set to false, the QUIT command is sent and the connection is immediately closed. If set to true *thedefault*, causes the transport to wait for the response to the QUIT command.
mail.smtp.reportsuccess	boolean	If set to true, causes the transport to include an SMTPAddressSucceededException for each address that is successful.
mail.smtp.socketFactory	Socket Factory	If set to a class that implements the javax.net.SocketFactory interface, this class will be used to create SMTP sockets.
mail.smtp.socketFactory.class	String	If set, specifies the name of a class that implements the javax.net.SocketFactory interface. This class will be used to create SMTP sockets.

mail.smtp.socketFactory.fallback	boolean	If set to true, failure to create a socket using the specified socket factory class will cause the socket to be created using the java.net.Socket class. Defaults to true.
mail.smtp.socketFactory.port	int	Specifies the port to connect to when using the specified socket factory. If not set, the default port will be used.
mail.smtp.ssl.enable	boolean	If set to true, use SSL to connect and use the SSL port by default. Defaults to false for the "smtp" protocol and true for the "smtps" protocol.
mail.smtp.ssl.checkserveridentity	boolean	If set to true, checks the server identity as specified by RFC 2595. Defaults to false.
mail.smtp.ssl.trust	String	If set, and a socket factory hasn't been specified, enables use of a MailSSLSocketFactory. If set to "*", all hosts are trusted. If set to a whitespace separated list of hosts, those hosts are trusted. Otherwise, trust depends on the certificate the server presents.
mail.smtp.ssl.socketFactory	SSL Socket Factory	If set to a class that extends the javax.net.ssl.SSLSocketFactory class, this class will be used to create SMTP SSL sockets.
mail.smtp.ssl.socketFactory.class	String	If set, specifies the name of a class that extends the javax.net.ssl.SSLSocketFactory class. This class will be used to create SMTP SSL sockets.
mail.smtp.ssl.socketFactory.port	int	Specifies the port to connect to when using the specified socket factory. If not set, the default port will be used.
mail.smtp.ssl.protocols	string	Specifies the SSL protocols that will be enabled for SSL connections. The property value is a whitespace separated list of tokens acceptable to the javax.net.ssl.SSLSocket.setEnabledProtocols method.
mail.smtp.starttls.enable	boolean	If true, enables the use of the STARTTLS command *ifsupportedbytheserver* to switch the connection to a TLS-protected connection before issuing any login commands. Defaults to false.
mail.smtp.starttls.required	boolean	If true, requires the use of the STARTTLS command. If the server doesn't support the STARTTLS command, or the command fails, the connect method will fail. Defaults to false.
mail.smtp.socks.host	string	Specifies the host name of a SOCKS5 proxy server that will be used for connections to the mail server.
mail.smtp.socks.port	string	Specifies the port number for the SOCKS5 proxy server. This should only need to be used if the proxy server is not using the

		standard port number of 1080.
mail.smtp.mailextension	String	Extension string to append to the MAIL command.
mail.smtp.userset	boolean	If set to true, use the RSET command instead of the NOOP command in the isConnected method. In some cases sendmail will respond slowly after many NOOP commands; use of RSET avoids this sendmail issue. Defaults to false.

In general, applications should not need to use the classes in this package directly. Instead, they should use the APIs defined by javax.mail package *andsubpackages*. Say for example applications should never construct instances of SMTPTransport directly. Instead, they should use the Session method getTransport to acquire an appropriate Transport object.

Examples to use SMPT server is demonstrated in chapter [Sending Emails](#).