mail

Using GMail
To connect to GMail the following settings supplied in the properties Array:

```
var userName = 'xxx';
var passWord = 'yyy';
var properties = new Array();
properties[0] = 'mail.smtp.host=smtp.gmail.com';
properties[1] = 'mail.smtp.auth=true';
properties[2] = 'mail.smtp.username=' + userName;
properties[3] = 'mail.smtp.password=' + passWord;
properties[4] = 'mail.smtp.port=587';
properties[5] = 'mail.smtp.starttls.enable=true';
```

Return Types

Attachment MailMessage

Method Summary

Attachment	createBinaryAttachment(filename, binarydata) Creates a binary attachment object.
Attachment	createBinaryAttachment(filename, binarydata, mimeType) Creates a binary attachment object.
Attachment	createTextAttachment(filename, textdata) Creates a text based attachment objec with the default 'text/plain' mimetype
Attachment	createTextAttachment(filename, textdata, mimeType) Creates a text based attachment object.
String	getLastSendMailExceptionMsg() Get the exception that occurred in the last sendMail attempt (null if no exception occurred).
MailMessage	getMailMessage(binaryblob/string) Helper method, returns MailMessage object from binary or 7bits string.
String[]	getPlainMailAddresses(addressesString) Helper method to only get the plain addresses.
Boolean	isValidEmailAddress(email) Checks whether the given e-mail address is valid or not.
MailMessage[]	receiveMail(username, password, leaveMsgsOnServer) Receive mails from pop3 account.
MailMessage[]	receiveMail(username, password, leaveMsgsOnServer, receiveMode) Receive mails from pop3 account.
MailMessage[]	receiveMail(username, password, leaveMsgsOnServer, receiveMode, onlyReceiveMsgWithSentDate) Receive mails from pop3 account.
MailMessage[]	receiveMail(username, password, leaveMsgsOnServer, receiveMode, onlyReceiveMsgWithSentDate, pop3Host) Receive mails from pop3 account.
MailMessage[]	receiveMail(username, password, leaveMsgsOnServer, receiveMode, onlyReceiveMsgWithSentDate, properties) Receive mails from pop3 account.
Boolean	sendBulkMail(to, from, subject, msgText) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc, attachment) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc, attachment, smtpHost) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc, attachment, overrideProperties) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc, attachments) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc, attachments, smtpHost) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendBulkMail(to, from, subject, msgText, cc, bcc, attachments, overrideProperties) Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Boolean	sendMail(to, from, subject, msgText, cc) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc, attachment) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc, attachment, smtpHost) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc, attachment, overrideProperties) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc, attachments) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc, attachments, smtpHost) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
Boolean	sendMail(to, from, subject, msgText, cc, bcc, attachments, overrideProperties) Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Method Details

createBinaryAttachment

Attachment **createBinaryAttachment** (filename, binarydata)

Creates a binary attachment object.

Parameters

{String} filename
{byte[]} binarydata

Returns

Attachment

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('logo1.gif',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createBinaryAttachment('logo2.gif',plugins.file.readFile('c:/temp
/another_logo.gif'));
var success = plugins.mail.sendMail('to_someone@example.com', 'John Cobb <from_me@example.org>', 'subject',
'msgText',null,null,new Array(attachment1,attachment2));
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

createBinaryAttachment

Attachment **createBinaryAttachment** (filename, binarydata, mimeType)

Creates a binary attachment object.

Parameters

{String} filename
{byte[]} binarydata
{String} mimeType

Returns

Attachment

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('logo1.gif',plugins.file.readFile('c:/temp/a_logo.
gif', 'image/gif'));
var attachment2 = plugins.mail.createBinaryAttachment('logo2.gif',plugins.file.readFile('c:/temp
/another_logo.gif', 'image/gif'));
var success = plugins.mail.sendMail('to_someone@example.com', 'John Cobb <from_me@example.org>', 'subject',
'msgText',null,null,new Array(attachment1,attachment2));
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

createTextAttachment

Attachment **createTextAttachment** (filename, textdata)

Creates a text based attachment objec with the default 'text/plain' mimetype

Parameters

{String} filename
{String} textdata

Returns

Attachment

Sample

```
var attachment = plugins.mail.createTextAttachment('readme.html','<html>bla bla bla');
var success = plugins.mail.sendMail('to_someone@example.com', 'John Cobb <from_me@example.com>', 'subject',
'msgText',null,null,attachment);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

createTextAttachment

Attachment **createTextAttachment** (filename, textdata, mimeType)

Creates a text based attachment object.

Parameters

{String} filename
{String} textdata
{String} mimeType

Returns

Attachment

Sample

```
var attachment = plugins.mail.createTextAttachment('readme.html','<html>bla bla bla', 'text/html');
var success = plugins.mail.sendMail('to_someone@example.com', 'John Cobb <from_me@example.com>', 'subject',
'msgText',null,null,attachment);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

getLastSendMailExceptionMsg

String **getLastSendMailExceptionMsg** ()

Get the exception that occurred in the last sendMail attempt (null if no exception occurred).

Returns

String

Sample

```
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.org', 'John Cobb
<from_me@example.com>', 'subject', 'my message',null,'unnamed@example.com');
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert',plugins.mail.getLastSendMailExceptionMsg(),'OK');
}
```

getMailMessage

MailMessage **getMailMessage** (binaryblob/string)

Helper method, returns MailMessage object from binary or 7bits string.

Parameters

binaryblob/string

Returns

MailMessage

Sample

```
var msg = plugins.mail.getMailMessage(myBlob);
if (msg != null) //if is null error occurred!
{
        application.output(msg.getFromAddresses())
}
```

getPlainMailAddresses

String[] **getPlainMailAddresses** (addressesString)

Helper method to only get the plain addresses.

Parameters

addressesString

Returns

String[]

Sample

```
var plainArray = plugins.mail.getPlainMailAddresses('John Cobb <from_me@example.com>,Pete
Cobb<from_pete@example.com>');
application.output(plainArray[0]) //will return 'from_me@example.com'
```

isValidEmailAddress

Boolean **isValidEmailAddress** (email)

Checks whether the given e-mail address is valid or not.

Parameters

{String} email

Returns

Boolean

Sample

```
plugins.mail.isValidEmailAddress("me@example.com");
```

receiveMail

MailMessage[] **receiveMail** (username, password, leaveMsgsOnServer)

Receive mails from pop3 account.

Parameters

{String} username
{String} password
{Boolean} leaveMsgsOnServer

Returns

MailMessage[]

Sample

```
var msgs = plugins.mail.receiveMail('mylogin', 'secretpass',  true);
if (msgs != null) //if is null error occurred!
{
        for (var i = 0 ; i < msgs.length ; i++)
        {
                var msg = msgs[i]
                application.output(msg.getFromAddresses())
                application.output(msg.getRecipientAddresses())
                application.output(msg.getReplyAddresses())
                application.output(msg.getSentDate())
                application.output(msg.getHeaders())
                application.output(msg.getSubject())
                application.output(msg.getHtmlMsg())
                application.output(msg.getPlainMsg())
                var attachments = msg.getAttachments()
                if (attachments != null)
                {
                        for (var j = 0 ; j < attachments.length ; j++)
                        {
                                var attachment = attachments[j]
                                application.output(attachment.getName())
                                var attachmentDataByteArray = attachment.getData()
                                //write attachmentDataByteArray to a file...
                        }
                }
        }
}
```

receiveMail

MailMessage[] **receiveMail** (username, password, leaveMsgsOnServer, receiveMode)

Receive mails from pop3 account.

Parameters

{String} username
{String} password
{Boolean} leaveMsgsOnServer
{Number} receiveMode

Returns

MailMessage[]

Sample

```
var receiveMode = 1;//0=FULL,1=HEADERS_ONLY,2=NO_ATTACHMENTS
var msgs = plugins.mail.receiveMail('mylogin', 'secretpass',  true,  0);
if (msgs != null) //if is null error occurred!
{
        for (var i = 0 ; i < msgs.length ; i++)
        {
                var msg = msgs[i]
                application.output(msg.getFromAddresses())
                application.output(msg.getRecipientAddresses())
                application.output(msg.getReplyAddresses())
                application.output(msg.getSentDate())
                application.output(msg.getHeaders())
                application.output(msg.getSubject())
        }
}
```

receiveMail

MailMessage[] **receiveMail** (username, password, leaveMsgsOnServer, receiveMode, onlyReceiveMsgWithSentDate)

Receive mails from pop3 account.

Parameters

{String} username
{String} password
{Boolean} leaveMsgsOnServer
{Number} receiveMode
{Date} onlyReceiveMsgWithSentDate

Returns

MailMessage[]

Sample

```
//it is also possible to first receive the headers and later receive a full message with particular
'sentdate'
//var receiveMode = 1;//0=FULL,1=HEADERS_ONLY,2=NO_ATTACHMENTS
var msgs = plugins.mail.receiveMail('mylogin', 'secretpass',  true,  0,
theSentDateObjectFormPreviousHeaderLoading);
if (msgs != null) //if is null error occurred!
{
        for (var i = 0 ; i < msgs.length ; i++)
        {
                var msg = msgs[i]
                application.output(msg.getFromAddresses())
                application.output(msg.getRecipientAddresses())
                application.output(msg.getReplyAddresses())
                application.output(msg.getSentDate())
                application.output(msg.getHeaders())
                application.output(msg.getSubject())
        }
}
```

receiveMail

MailMessage[] **receiveMail** (username, password, leaveMsgsOnServer, receiveMode, onlyReceiveMsgWithSentDate, pop3Host)

Receive mails from pop3 account.

Parameters

{String} username
{String} password
{Boolean} leaveMsgsOnServer
{Number} receiveMode
{Date} onlyReceiveMsgWithSentDate
{String} pop3Host

Returns

MailMessage[]

Sample

```
//it is also possible to first receive the headers and later receive a full message
var receiveMode = 0;//0=FULL,1=HEADERS_ONLY,2=NO_ATTACHMENTS
var pop3Host = 'myserver.com';
var msgs = plugins.mail.receiveMail('mylogin', 'secretpass',  true,  receiveMode,  null, pop3Host);
if (msgs != null) //if is null error occurred!
{
        for (var i = 0 ; i < msgs.length ; i++)
        {
                var msg = msgs[i]
                application.output(msg.getFromAddresses())
                application.output(msg.getRecipientAddresses())
                application.output(msg.getReplyAddresses())
                application.output(msg.getSentDate())
                application.output(msg.getHeaders())
                application.output(msg.getSubject())
                application.output(msg.getHtmlMsg())
                application.output(msg.getPlainMsg())
                var attachments = msg.getAttachments()
                if (attachments != null)
                {
                        for (var j = 0 ; j < attachments.length ; j++)
                        {
                                var attachment = attachments[j]
                                application.output(attachment.getName())
                                var attachmentDataByteArray = attachment.getData()
                                //write attachmentDataByteArray to a file...
                        }
                }
        }
}
```

receiveMail

MailMessage[] **receiveMail** (username, password, leaveMsgsOnServer, receiveMode, onlyReceiveMsgWithSentDate, properties)

Receive mails from pop3 account.

Parameters

{String} username
{String} password
{Boolean} leaveMsgsOnServer
{Number} receiveMode
{Date} onlyReceiveMsgWithSentDate
{String[]} properties

Returns

MailMessage[]

Sample

```
var receiveMode = 1;//0=FULL,1=HEADERS_ONLY,2=NO_ATTACHMENTS

var properties = new Array();
properties[0] = 'mail.pop3.port=995';
properties[1] = 'mail.pop3.ssl.enable=true';
properties[2] = 'mail.pop3.host=myserver.com';
properties[3] = 'mail.pop3.user=user@myserver.com';

var msgs = plugins.mail.receiveMail('mylogin', 'secretpass',  true,  receiveMode,  null, properties);
if (msgs != null) //if is null error occurred!
{
        for (var i = 0 ; i < msgs.length ; i++)
        {
                var msg = msgs[i]
                application.output(msg.getFromAddresses())
                application.output(msg.getRecipientAddresses())
                application.output(msg.getReplyAddresses())
                application.output(msg.getSentDate())
                application.output(msg.getHeaders())
                application.output(msg.getSubject())
        }
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text

Returns

Boolean

Sample

```
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>,replyTo@example.com', 'subject', msgText);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma

Returns

Boolean

Sample

```
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,'cc1@example.com,cc2@example.com');
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma

Returns

Boolean

Sample

```
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'bcc1@example.com');
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc, attachment)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment} attachment - A single attachment

Returns

Boolean

Sample

```
var attachment = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.gif'));
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'bcc1@example.com,bcc2@example.com',attachment);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc, attachment, smtpHost)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment} attachment - A single attachment
{String} smtpHost - The smtp host

Returns

Boolean

Sample

```
var attachment = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.gif'));
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var smtphost = 'myserver.com';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',attachment,smtphost);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc, attachment, overrideProperties)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment} attachment - A single attachment
{String[]} overrideProperties - An array of properties

Returns

Boolean

Sample

```
var attachment = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.gif'));
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
//it is possbile to set all kind of smtp properties
var properties = new Array()
properties[0] = 'mail.smtp.host=myserver.com'
// properties specification can be found at:http://java.sun.com/products/javamail/javadocs/com/sun/mail/smtp
/package-summary.html
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',attachment,properties);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc, attachments)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment[]} attachments - The attachments

Returns

Boolean

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createTextAttachment('embedded','A text attachement');
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'bcc1@example.com,bcc2@example.com',[attachment1,
attachment2]);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc, attachments, smtpHost)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment[]} attachments - The attachments
{String} smtpHost - The smtp host

Returns

Boolean

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createTextAttachment('embedded','A text attachement');
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var smtphost = 'myserver.com';
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',[attachment1,attachement2],smtphost);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendBulkMail

Boolean **sendBulkMail** (to, from, subject, msgText, cc, bcc, attachments, overrideProperties)

Send a bulk mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).
A bulk email makes it possible for one to not receive "out of office" emails back from receiver.

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the bulk mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment[]} attachments - The attachments
{String[]} overrideProperties - An array of properties

Returns

Boolean

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createTextAttachment('embedded','A text attachement');
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
//it is possbile to set all kind of smtp properties
var properties = new Array()
properties[0] = 'mail.smtp.host=myserver.com'
// properties specification can be found at:http://java.sun.com/products/javamail/javadocs/com/sun/mail/smtp
/package-summary.html
var success = plugins.mail.sendBulkMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',[attachment1,attachement2],properties);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send bulk mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text

Returns

Boolean

Sample

```
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>,replyTo@example.com', 'subject', msgText);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma

Returns

Boolean

Sample

```
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,'cc1@example.com,cc2@example.com');
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma

Returns

Boolean

Sample

```
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'bcc1@example.com');
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc, attachment)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment} attachment - A single attachment

Returns

Boolean

Sample

```
var attachment = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.gif'));
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'bcc1@example.com,bcc2@example.com',attachment);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc, attachment, smtpHost)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment} attachment - A single attachment
{String} smtpHost - The smtp host

Returns

Boolean

Sample

```
var attachment = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.gif'));
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var smtphost = 'myserver.com';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',attachment,smtphost);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc, attachment, overrideProperties)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment} attachment - A single attachment
{String[]} overrideProperties - An array of properties

Returns

Boolean

Sample

```
var attachment = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.gif'));
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
//it is possbile to set all kind of smtp properties
var properties = new Array()
properties[0] = 'mail.smtp.host=myserver.com'
// properties specification can be found at:http://java.sun.com/products/javamail/javadocs/com/sun/mail/smtp
/package-summary.html
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',attachment,properties);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc, attachments)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment[]} attachments - The attachments

Returns

Boolean

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createTextAttachment('embedded','A text attachement');
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'bcc1@example.com,bcc2@example.com',[attachment1,
attachment2]);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc, attachments, smtpHost)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment[]} attachments - The attachments
{String} smtpHost - The smtp host

Returns

Boolean

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createTextAttachment('embedded','A text attachement');
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
var smtphost = 'myserver.com';
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',[attachment1,attachement2],smtphost);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```

sendMail

Boolean **sendMail** (to, from, subject, msgText, cc, bcc, attachments, overrideProperties)

Send a mail, if you make the msgText start with <html> the message will be sent in html (and you can use all html formatting).

Parameters

{String} to - A string containing 1 or multiple addresses seperated by a comma.
{String} from - A string containing an address and an optional reply address, seperated by a comma.
{String} subject - The subject of the mail
{String} msgText - The message text
{String} cc - One or more addresses seperated by a comma
{String} bcc - One or more addresses seperated by a comma
{Attachment[]} attachments - The attachments
{String[]} overrideProperties - An array of properties

Returns

Boolean

Sample

```
var attachment1 = plugins.mail.createBinaryAttachment('embedded',plugins.file.readFile('c:/temp/a_logo.
gif'));
var attachment2 = plugins.mail.createTextAttachment('embedded','A text attachement');
var msgText = 'plain msg<html>styled html msg<img src="%%embedded%%"></html>';
//it is possbile to set all kind of smtp properties
var properties = new Array()
properties[0] = 'mail.smtp.host=myserver.com'
// properties specification can be found at:http://java.sun.com/products/javamail/javadocs/com/sun/mail/smtp
/package-summary.html
var success = plugins.mail.sendMail('to_someone@example.com,to_someone_else@example.net', 'John Cobb
<from_me@example.com>', 'subject', msgText,null,'unnamed@example.com',[attachment1,attachement2],properties);
if (!success)
{
        plugins.dialogs.showWarningDialog('Alert','Failed to send mail','OK');
}
```