Mandando um email textual simples

```
import org.apache.commons.mail.*;
...
Email email = new SimpleEmail();
email.setHostName("<enderecoDoServidorDeSMTP>");
email.setSmtpPort(<porta>); // em geral, 465
DefaultAuthenticator autenticador = new DefaultAuthenticator
                                    ("<usuario", "<senha>");
email.setAuthenticator(autenticador);
email.setSSLOnConnect(true);
email.setSubject("<assunto>");
email.setFrom("<emailDoRemetente");
email.addTo("<emailDeUmDosDestinatarios>"); // 1 ou + vezes
email.setMsg("<mensagem>"); // pode conter \n
email.send();
```

Mandando um email textual com anexos locais

```
import org.apache.commons.mail.*;
...

// Cria o anexo de um arquivo que se tem localmente
// Este trecho pode se repetir em caso de mais anexos
EmailAttachment anexo = new EmailAttachment();
anexo.setPath("<caminhoParaArquivoAnexado>");
anexo.setDisposition(EmailAttachment.ATTACHMENT);
anexo.setDescription("<fraseDescrevendoAnexo>");
anexo.setName("<nomeParaAnexo>");

// Cria a mensagem
MultiPartEmail email = new MultiPartEmail();
email.setHostName("<enderecoDoServidorDeSMTP>");
email.setSubject("<assunto>");
email.setFrom("<emailDoRemetente");
email.addTo("<emailDeUmDosDestinatarios>"); // 1 ou + vezes
email.setMsg("<mensagem>"); // pode conter \n

// Adiciona um anexo
email.attach(anexo); // 1 ou + vezes

// Envia o email
email.send();
```

Mandando um email textual com anexos remotos

```
import org.apache.commons.mail.*;
...

// Cria o anexo de um arquivo que se tem localmente
// Este trecho pode se repetir em caso de mais anexos
EmailAttachment anexo = new EmailAttachment();
anexo.setURL(new URL("<enderecoNaInternetDoArquivoAnexado>"));
anexo.setDisposition(EmailAttachment.ATTACHMENT);
anexo.setDescription("<fraseDescrevendoAnexo>");
anexo.setName("<nomeParaAnexo>");

// Cria a mensagem
```

```
MultiPartEmail email = new MultiPartEmail();
email.setHostName("<enderecoDoServidorDeSMTP>");
email.setSubject("<assunto>");
email.setFrom("<emailDoRemetente");
email.addTo("<emailDeUmDosDestinatarios>"); // 1 ou + vezes
email.setMsg("<mensagem>"); // pode conter \n

// Adiciona um anexo
email.attach(anexo); // 1 ou + vezes

// Envia o email
email.send();
```

Mandando um email formatado em HTML com imagem anexada

```
import org.apache.commons.mail.HtmlEmail;
...
// Cria a mensagem
HtmlEmail email = new HtmlEmail();
email.setHostName("<enderecoDoServidorDeSMTP>");
email.setSubject("<assunto>");
email.setFrom("<emailDoRemetente");
email.addTo("<emailDeUmDosDestinatarios>"); // 1 ou + vezes

// Embute a imagem e obtem seu id
URL url = new URL(<enderecoNaInternetDoArquivoAnexado>");
String id = email.embed (url, "<nomeDoAnexo>");

// Compoe a mensagem HTML
email.setHtmlMsg("<html>...<img src=\"cid:"+id+"\">...</html>");

// Especifica um texto a ser mostrada no caso
// do leitor de email nao suportar mensagens em HTML
email.setTextMsg("<texto>");

// Envia o email
email.send();
```

Mandando um email cujo texto está em um arquivo HTML

```
import org.apache.commons.mail.HtmlEmail;
...
// define a URL base para resolver especificacoes
// de localizacao relativas
URL url = new URL("<enderecoNaInternet");

// Especifica o caminho para o arquivo HTML que constitui
// a mensagem
String modelo = "<caminhoParaModelo";

HtmlEmail email = new ImageHtmlEmail();
email.setDataSourceResolver(new DataSourceResolverImpl(url));
email.setHostName("<enderecoDoServidorDeSMTP>");
email.setSubject("<assunto>");
email.setFrom("<emailDoRemetente");
email.addTo("<emailDeUmDosDestinatarios>"); // 1 ou + vezes
```

```
// Indica o modelo a ser usado
email.setHtmlMsg(modelo);

// Especifica um texto a ser mostrada no caso
// do leitor de email nao suportar mensagens em HTML
email.setTextMsg("<texto>");

// envia o email
email.send();

// todas as imagens referenciadas no modelo serao, automaticamente,
// transformadas em imagens inline
```

Lidando com mensagens que retornam e não podem serem devolvidas ao remetente (boa prática)

```
setBounceAddress
("<enderecoDeEmailParaRetornarMensagemQuandoEmailRemetenteInvalido>");
```

Lendo uma mensagem de uma pasta

```
import java.util.*;
import javax.mail.*;
...
Properties propriedades = new Properties();
propriedades.setProperty("mail.store.protocol", "imaps");
Session sessao = Session.getInstance(propriedades, null);
Store deposito = session.getStore();
deposito.connect("<enderecoDoServidorIMAP",
                 "<emailDoDestinatario>",
                 "<senhaDoDestinatario>");

Folder pasta = deposito.getFolder("<nomeDaPasta>");
pasta.open(Folder.READ_ONLY);

Message mensagem =
pasta.getMessage(<numeroDaMensagem>); // vai de 1 até
                                      // pasta.getMessageCount()

System.out.println("DE:" + mensagem.getFrom()[0]);
System.out.println("DATA DE ENVIO: " + mensagem.getSentDate());
System.out.println("ASSUNTO: " + mensagem.getSubject());

Multipart partes = (Multipart) mensagem.getContent();
for (int i=0; i<partes.getCount(); i++)
{
    BodyPart corpo = partes.getBodyPart(i);

    String   tipo  = corpo.getDisposition();
```

```
    if (tipo!=null && tipo.equals(BodyPart.ATTACHMENT))
    {
         DataHandler arquivo = corpo.getDataHandler();
         System.out.println("Anexo : " + arquivo.getName());

     }
     else
         System.out.println("Texto: " + corpo.getContent());
}
```

Descobrindo as pastas que existem na pasta raiz (para subpastas, subsubpastas, etc, usar recursão)

```
import java.util.*;
import javax.mail.*;
...
Properties propriedades = new Properties();
propriedades.setProperty("mail.store.protocol", "imaps");
Session sessao = Session.getInstance(propriedades, null);
Store deposito = session.getStore();
deposito.connect("<enderecoDoServidorIMAP",
                 "<emailDoDestinatario>",
                 "<senhaDoDestinatario>");
Folder raiz = deposito.getDefaultFolder();
Folder[] pastas = raiz.list();
for (Folder pasta : pastas) {
    System.out.println(pasta.getFullName());
```

Apagando uma mensagem de uma pasta

```
import java.util.*;
import javax.mail.*;
...
Properties propriedades = new Properties();
propriedades.setProperty("mail.store.protocol", "imaps");
Session sessao = Session.getInstance(propriedades, null);
Store deposito = session.getStore();
deposito.connect("<enderecoDoServidorIMAP",
                 "<emailDoDestinatario>",
                 "<senhaDoDestinatario>");
Folder pasta = deposito.getFolder("<nomeDaPasta>");
pasta.open(Folder.READ_WRITE);
Message mensagem =
pasta.getMessage(<numeroDaMensagem>); // vai de 1 até
                                      // pasta.getMessageCount()

mensagem.setFlag(Flags.Flag.DELETED, true);
pasta.expunge();
pasta.close();
```

Respondendo uma mensagem de uma pasta

```
import java.util.*;
import javax.mail.*;
...
Properties propriedades = new Properties();
propriedades.setProperty("mail.store.protocol", "imaps");
Session sessao = Session.getInstance(propriedades, null);
Store deposito = session.getStore();
deposito.connect("<enderecoDoServidorIMAP",
                 "<emailDoDestinatario>",
                 "<senhaDoDestinatario>");
Folder pasta = deposito.getFolder("<nomeDaPasta>");
pasta.open(Folder.READ_ONLY);
Message mensagem =
pasta.getMessage(<numeroDaMensagem>); // vai de 1 até
                                      // pasta.getMessageCount()
String textoMensagem = (String)mensagem.getContent();
Message resposta = mensagem.reply();
String textoResposta = textoMensagem.replaceAll("(?m)^", "> ");
// permita que o usuario edite textoResposta
resposta.setText(replyText);
resposta.send();
```

Encaminhando como anexo uma mensagem de uma pasta

```
import java.util.*;
import javax.mail.*;
import org.apache.commons.mail.*;
...
// proceder de uma das formas acima ensinadas para criar a mensagem
// encaminhamento (texto simples, texto com anexo, HTML, etc)
String TextoEncaminhamento;
// permita que o usuario edite textoEncaminhamento
encaminhamento.setText (textoEncaminhamento);

Properties propriedades = new Properties();
propriedades.setProperty("mail.store.protocol", "imaps");
Session sessao = Session.getInstance(propriedades, null);
Store deposito = session.getStore();
deposito.connect("<enderecoDoServidorIMAP",
                 "<emailDoDestinatario>",
                 "<senhaDoDestinatario>");
Folder pasta = deposito.getFolder("<nomeDaPasta>");
pasta.open(Folder.READ_ONLY);
Message mensagem =
```

```
pasta.getMessage(<numeroDaMensagem>); // vai de 1 até
                                      // pasta.getMessageCount()
MimeBodyPart parte = new MimeBodyPart();
parte.setContent(mensagem, "message/rfc822");

encaminhamento.addPart(parte);
encaminhamento.send();
```

Encaminhando no texto uma mensagem de uma pasta

```
import java.util.*;
import javax.mail.*;
import org.apache.commons.mail.*;
...
Properties propriedades = new Properties();
propriedades.setProperty("mail.store.protocol", "imaps");
Session sessao = Session.getInstance(propriedades, null);
Store deposito = session.getStore();
deposito.connect("<enderecoDoServidorIMAP",
                 "<emailDoDestinatario>",
                 "<senhaDoDestinatario>");
Folder pasta = deposito.getFolder("<nomeDaPasta>");
pasta.open(Folder.READ_ONLY);
Message mensagem =
pasta.getMessage(<numeroDaMensagem>); // vai de 1 até
                                      // pasta.getMessageCount()
String textoMensagem       = (String)mensagem.getContent();
String textoEncaminhamento = String.format(
    "\n\n-------- Mensagem Original --------\n" +
    "Assunto: %s\nData de Envio: %s\nDe: %s\nPara: %s\n",
    mensagem.getSubject(),
    mensagem.getSentDate(),
    mensagem.getFrom()[0],
    mensagem (
    mensagem.getRecipients(Message.RecipientType.TO)));
// proceder de uma das formas acima ensinadas para criar a mensagem
// encaminhamento (texto simples, texto com anexo, HTML, etc)
// permita que o usuario edite textoEncaminhado
encaminhamento.setText(textoEncaminhado);
encaminhamento.send();
```