

Customization Guide

iPlanet™ Messenger Express

Release 5.2

February 2002

Copyright © 2002 Sun Microsystems, Inc. All rights reserved.

Sun, Sun Microsystems, the Sun logo, iPlanet, and the logo iPlanet are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. Netscape and the Netscape N logo are registered trademarks of Netscape Communications Corporation in the U.S. and other countries. Other Netscape logos, product names, and service names are also trademarks of Netscape Communications Corporation, which may be registered in other countries. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

Federal Acquisitions: Commercial Software—Government Users Subject to Standard License Terms and Conditions

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation. No part of the product or this document may be reproduced in any form by any means without prior written authorization of Sun Microsystems, Inc. and its licensors, if any.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright © 2002 Sun Microsystems, Inc. Tous droits réservés.

Sun, Sun Microsystems, le logo Sun, iPlanet, et le logo iPlanet sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et d'autres pays. Netscape et le logo Netscape N sont des marques déposées de Netscape Communications Corporation aux Etats-Unis et d'autres pays. Les autres logos, les noms de produit, et les noms de service de Netscape sont des marques déposées de Netscape Communications Corporation dans certains autres pays. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company Ltd.

Le produit décrit dans ce document est distribué selon des conditions de licence qui en restreignent l'utilisation, la copie, la distribution et la décompilation. Aucune partie de ce produit ni de ce document ne peut être reproduite sous quelque forme ou par quelque moyen que ce soit sans l'autorisation écrite préalable de Sun Microsystems, Inc. et, le cas échéant, de ses bailleurs de licence.

CETTE DOCUMENTATION EST FOURNIE "EN L'ÉTAT", ET TOUTES CONDITIONS EXPRESSES OU IMPLICITES, TOUTES REPRÉSENTATIONS ET TOUTES GARANTIES, Y COMPRIS TOUTE GARANTIE IMPLICITE D'APTITUDE À LA VENTE, OU À UN BUT PARTICULIER OU DE NON CONTREFAÇON SONT EXCLUES, EXCEPTÉ DANS LA MESURE OÙ DE TELLES EXCLUSIONS SERAIENT CONTRAIRES À LA LOI.

Contents

About This Guide	5
Who Should Read This Book	5
What You Need to Know	6
How This Book is Organized	6
Document Conventions	7
Monospaced Font	7
Bold Monospaced Font	7
Italicized Font	7
Square or Straight Brackets	7
Command-Line Prompts	7
Where to Find Related Information	8
Where to Find This Book Online	8
Chapter 1 Introduction to Messenger Express and Customization	9
Messenger Express Customization Overview	9
Messenger Express Components	10
Location of Customizable Files	10
Messenger Express Localization	11
Specific Locales	11
Location of Locale-Specific Customizable Files	12
Basic Interfaces and Associated Functions	12
Inbox Screen	13
Message Screen	14
Folders Screen	16
Options Screen	17
Composition Window	18
Message Search Window	20
Chapter 2 Customizing General Features	23
Modifying the Login Screen	23

Modifying Color Sets	27
Modifying the Logo and Link	30
Modifying the Title Graphic and Text	31
Inserting Banners and Links	33
Chapter 3 Customizing User Interface Features	37
Modifying the Main Function Tabs	38
Modifying the Mailbox Tool Bar	40
Modifying the Message List Window	42
Modifying the Message Display Window	44
Modifying the Message Tool Bar	46
Modifying the Message Composition Window	48
Modifying Message Search Window	53
Modifying the Address (Directory Lookup) Window	57
Modifying the Options Window	59
Modifying the Folders Window	61
Chapter 4 Customizing Advanced Features	65
Advanced Customization Overview	65
Messenger Express User Interface Customizable Features	66
Attachments	66
Collect Mail from Another Server	67
Message Composition	67
Folder Management Tab	68
Address Search	68
Mailbox Management Tab	68
Personal Option Management (Options Tab)	69
Return Receipt	69
Customization of Users' Default LDAP Attributes	70
Domain Specific Customization	71
Domain From URL	72
Chapter 5 Managing Authentication to the Messenger Express Service	73
Introduction to Authentication	73
SDK Files and Functions	74
SDK Configuration Initialization	75
SDK Lookup	76
SDK Cleanup	77
Example Deployment	77
Index	79

About This Guide

This guide explains how to customize the look and feel of iPlanet Messenger Express. Although the product architecture permits an almost unlimited customization of the “static” portion of the pages served by the Messenger Express HTTP daemon, this guide focuses on how to perform the most commonly requested customizations. In addition, the customizations have been tied together into an application scenario so that examples, code, screen shots, all relate to one another and provide a common frame of reference.

Topics covered in this chapter include:

- Who Should Read This Book
- What You Need to Know
- How This Book is Organized
- Document Conventions
- Where to Find Related Information
- Where to Find This Book Online

Who Should Read This Book

You should read this book if you are responsible for administering, configuring, and customizing Messenger Express at your site. Developers may also find this guide useful for reference.

What You Need to Know

This book assumes a knowledge of iPlanet Messaging Server software and an understanding of the following:

- JavaScript™
- HTML
- Email applications
- Web site development

How This Book is Organized

This book contains the following chapters:

- About This Guide (this chapter)
- Chapter 1, “Introduction to Messenger Express and Customization”
This chapter provides a high-level overview of how to customize the look and feel of iPlanet Messenger Express.
- Chapter 2, “Customizing General Features”
This chapter explains how to customize the general features of iPlanet Messenger Express.
- Chapter 3, “Customizing User Interface Features”
This chapter explains how to customize the iPlanet Messenger Express user interface.
- Chapter 4, “Customizing Advanced Features”
This chapter discusses advanced customization techniques.
- Chapter 5, “Managing Authentication to the Messenger Express Service”
This chapter describes how to integrate other authentication mechanisms with iPlanet Messenger Express.

Document Conventions

Monospaced Font

`Monospaced font` is used for any text that appears on the computer screen or text that you should type. It is also used for file names, distinguished names, functions, and examples.

Bold Monospaced Font

`Bold monospaced font` is used to represent text within a code example that you should type.

Italicized Font

Italicized font is used to represent text that you enter using information that is unique to your installation (for example, variables). It is used for server paths, names, and account IDs.

Square or Straight Brackets

Square (or straight) brackets [] are used to enclose optional parameters.

Command-Line Prompts

Command-line prompts (for example, % for a C-Shell, or \$ for a Korn or Bourne shell) are not displayed in the examples. Depending on which operating system environment you are using, you will see a variety of different command-line prompts. However, you should enter the command as it appears in the document unless specifically noted otherwise.

Where to Find Related Information

In addition to this guide, iPlanet Messaging Server comes with supplementary information for administrators as well as documentation for end users and developers. Use the following URL to see all the Messaging Server documentation:

<http://docs.iplanet.com/docs/manuals/messaging.html>

Where to Find This Book Online

You can view the online documentation in PDF and HTML formats by pointing your browser to the following URL:

<http://docs.iplanet.com/docs/manuals/messaging.html>

Introduction to Messenger Express and Customization

iPlanet Messenger Express is a web-based electronic mail program that enables end users to access their mailboxes using a browser. Messenger Express clients send mail to a specialized web server that is part of iPlanet Messaging Server. The HTTP service then sends the message to the local Message Transfer Agent (MTA) or to a remote MTA for routing or delivery.

Almost all features of Messenger Express are fully customizable. This guide explains the features that are customizable. Most features can also be customized easily during an upgrade.

This chapter contains the following sections:

- Messenger Express Customization Overview
- Messenger Express Localization
- Basic Interfaces and Associated Functions

Messenger Express Customization Overview

iPlanet Messenger Express lets you rewrite the “static” portion of the pages served by the Messenger Express HTTP daemon to produce a fully customized webmail service. Messenger Express supports both JavaScript and HTML in implementing customization schemes.

Messenger Express Components

iPlanet Messenger Express consists of two components, the client and the server. The client reads and interprets the JavaScript language. The HTTP server understands proprietary protocols that communicate with Messenger Express. The JavaScript files reside on the server and are downloaded to the client. The client extracts data from the JavaScript code to customize Messenger Express functions. All modifications and customizations are done on the server.

HTML files contain both text and markup describing how the text is formatted and handled. Markup is implemented through a set of tags, which specify things like headers, indents, font size, italics and so on. These are largely static tags that deal exclusively with text within the HTML file on the client. However, the HTML also contains dynamic tags, such as JavaScript embedded in the client file, that point to files and functions on the server. The dynamic tags enable the HTML file client to pull in data processed on the server for use in otherwise static Web pages.

HTML files provide the skeleton structure of each of the interfaces, whereas JavaScript files give the specific attributes. Each of the JavaScript main functions are contained within an HTML parent functions. There are also differentiations within the JavaScript files themselves. The file `main.js` primarily controls the layout of the interfaces, whereas the file `i18n.js` controls the text. The `i18n.js` file also can be localized to fit the language of many regions in the world.

The HTML files are the `parent` functions that call the `main` functions in the JavaScript files to initiate actions.

Location of Customizable Files

The Messenger Express JavaScript and HTML files that can be customized reside in the `server-root/html` directory, where `server-root` represents the directory path in which you install the Messaging Server software. The server needs to be restarted for any changes in the HTML and JavaScript files to take effect.

lists the files to be edited to customize Messenger Express, and which part of Messenger Express each file controls.

Table 1-1 Messenger Express Customizable Files

Files	What the File Controls in Messenger Express
<code>main.js</code>	Layout of the UI
<code>lang/i18n.js</code>	Text on the UI
<code>mbox_fs.html</code>	Mailbox management portion of the UI

Table 1-1 Messenger Express Customizable Files (*Continued*)

Files	What the File Controls in Messenger Express
msg_fs.html	Message management portion of the UI
fldr_fs.html	Folder management portion of the UI
opts_fs.html	Option management portion of the UI
comp_fs.html	Message composition
<i>lang/default</i> .html	Login screen
lookup_fs.html	Address search
attach_fs.html	Attachments
collect_fs.html	Collection of mail from another server
receipt_fs.html	Return receipt
subscribe_fs.html	Subscribe Folders
emoticons.html	Alter emoticons

Messenger Express Localization

You can localize any feature of iPlanet Messenger Express. You can create different pages for different languages. When you create language-specific static webmail pages, you group them in subdirectories under the main document directory. The webmail code automatically detects the client's language preference and fetches the webmail pages from the appropriate subdirectory.

When you change common sections of the static webmail pages, you must make the changes on multiple occasions if modifications (for example, to JavaScript behavior) are desired across languages. Conversely, you can make language-specific modifications selectively throughout the application.

Specific Locales

Table 1-2 lists the specific locales (and their abbreviations) that Messenger Express services. The protocol's default language is English.

Table 1-2 Messenger Express Specific Locales

Locale	Abbreviation
English	en

Table 1-2 Messenger Express Specific Locales (*Continued*)

Locale	Abbreviation
Japanese	ja
Spanish	es
French	fr
German	de

Location of Locale-Specific Customizable Files

The localized Messenger Express JavaScript and HTML files reside in the *server-root/html/locale_specific* directory, where *server-root* represents the directory path in which the Messaging Server software is installed.

Basic Interfaces and Associated Functions

This section presents the functions associated with various Messenger Express screens, including:

- Inbox screen
- Message screen
- Folders screen
- Options screen
- Composition window
- Message search window

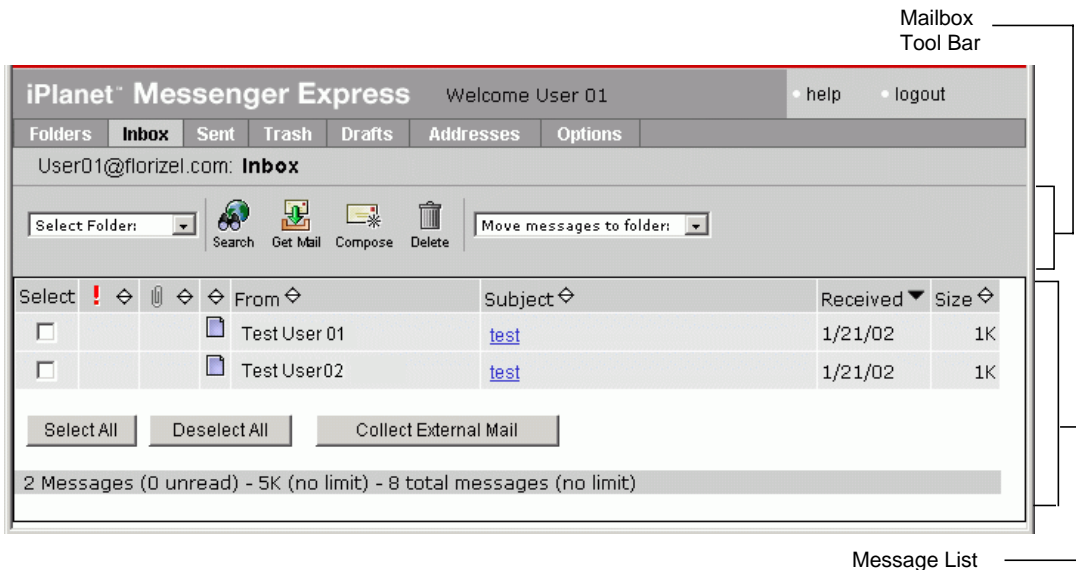
Tables list the functions associated with each button on Messenger Express screens.

The files containing the functions reside in the *server-root/html* directory, where *server-root* represents the directory path in which the Message Server software is installed.

Inbox Screen

The Messenger Express inbox screen, shown in Figure 1-1, enables you to view all messages and its basic features—for example, subject, from, received and size. The inbox screen gets new messages and enables you to search for or delete old messages, as well as move messages into other folders.

Figure 1-1 Messenger Express Inbox Screen



Inbox Screen Functions

Table 1-3 lists the functions needed to customize the inbox screen, including main functions (found in `main.js`) and parent functions (found in `mbox_fs.html`).

Table 1-3 Inbox Screen Functions

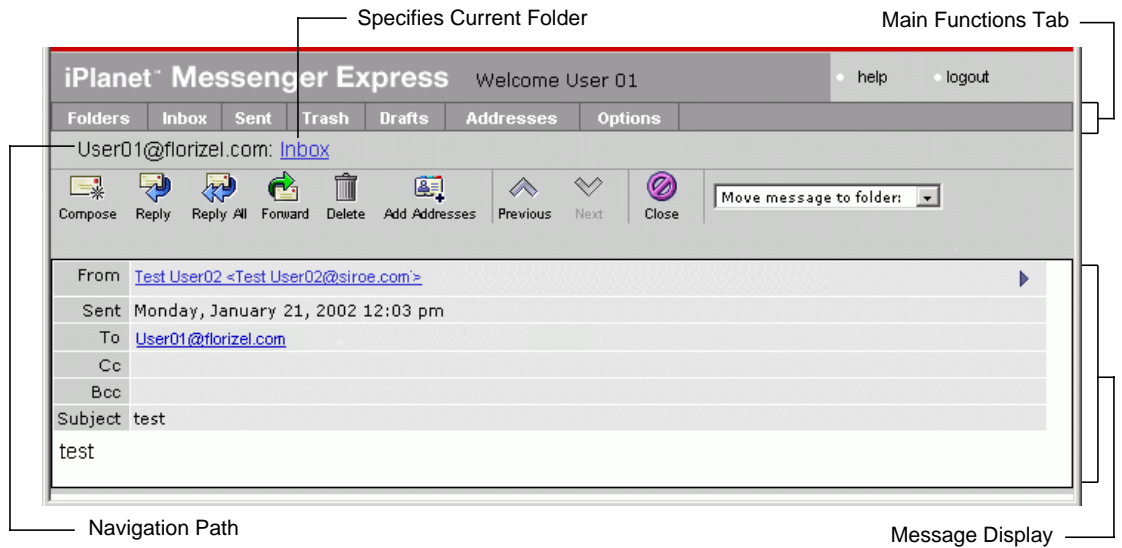
Item	Function
Folders	<code>main.displayFolders()</code>
Inbox	<code>main.displaySpecialMbox('Inbox')</code>
Sent Mail	<code>main.displaySpecialMbox('Sent')</code>
Trash	<code>main.displaySpecialMbox('Trash')</code>
Drafts	<code>main.displaySpecialMbox('Drafts')</code>
Addresses	<code>main.displayPab()</code>

Table 1-3 Inbox Screen Functions *(Continued)*

Item	Function
Options	<code>main.selectOptions()</code>
Help	<code>main.help()</code>
Logout	<code>main.logout()</code>
Get Mail	<code>main.check_mail = 1;</code> <code>main.displaySpecialMbox('Inbox')</code>
Compose	<code>main.compose('new')</code>
Search	<code>parent.srch()</code>
Move Messages to Folder	<code>parent.move()</code>
Delete and Undelete	<code>parent.delmsg()</code>
Collect External Mail	<code>main.collect()</code>

Message Screen

The Messenger Express message screen, shown in Figure 1-2, displays the message selected from the inbox screen. The message screen gives the option of replying to the sender(s), forwarding the message, moving the message to a different folder, or deleting the message. The message screen also enables navigation to the next or previous message.

Figure 1-2 Messenger Express Message Screen

Message Screen Functions

Table 1-4 lists the functions needed to customize the message screen, including main functions (found in `main.js`) and parent functions (found in `msg_fs.html`).

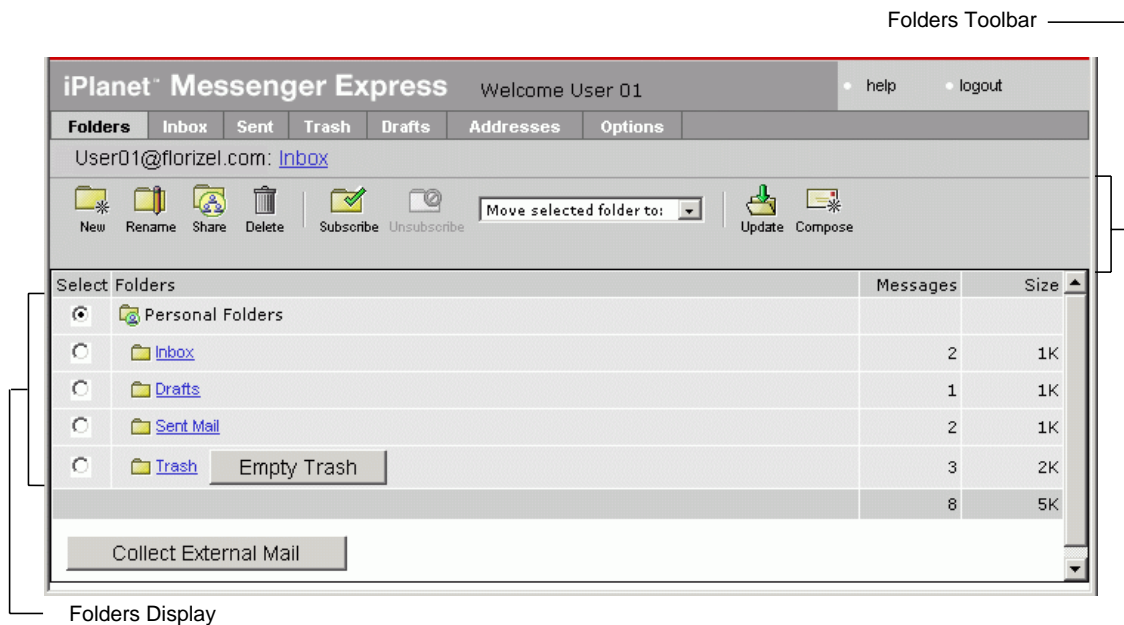
Table 1-4 Message Screen Functions

Item	Function
Compose	<code>main.compose('new')</code>
Reply	<code>main.compose('reply')</code>
Reply All	<code>main.compose('replyall')</code>
Forward	<code>main.compose('forward')</code>
Delete and Undelete	<code>parent.delmsg()</code>
Add Addresses	<code>parent.addAllAddresses()</code>
Previous	<code>parent.prev()</code>
Next	<code>parent.next()</code>
Close	<code>parent.gotofolder()</code>

Folders Screen

The Messenger Express Folders screen, shown in Figure 1-3, displays all folders that can be accessed. The Folders screen lists the number of messages contained and the size of each folder. The Folders screen also enables creating new folders, renaming or deleting old ones, subscribing or unsubscribing shared folder, sharing folder, moving a folder within another folder, updating the inbox, and composing new messages. Like the Inbox screen, the Folders screen also enables collection of external mails.

Figure 1-3 Messenger Express Folders Screen



Folders Screen Functions

Table 1-5 lists the functions needed to customize the folders screen, including main functions (found in `main.js`) and parent functions (found in `fldr_fs.html`).

Table 1-5 Folders Screen Functions

Item	Function
Update	<code>main.refreshFolders()</code>
Compose	<code>main.compose('new')</code>

Table 1-5 Folders Screen Functions (Continued)

Item	Function
New	<code>parent.addFolder()</code>
Rename	<code>parent.renFolder()</code>
Share	<code>parent.setfolder()</code>
Delete	<code>parent.delFolder()</code>
Subscribe	<code>main.subscribeFolder()</code>
Unsubscribe	<code>main.unsubscribeFolder()</code>
Move Folder	<code>parent.moveFolder(options[selectedIndex].value)</code>

Options Screen

The Messenger Express Options screen, shown in Figure 1-4, enables access to the subscriber's account summary, personal information, password, settings, appearance, vacation message, and delegated administrator, all of which can be customized.

Figure 1-4 Messenger Express Options Screen

The screenshot displays the Messenger Express Options screen. On the left is a vertical menu with the following items: Account Summary, Personal Information, Password, Settings, Appearance, Vacation Message, and Delegated Administrator. The 'Password' option is selected, and the main content area shows the 'Password' form. The form includes the following text: 'Use this form to change the password you use to access Messenger Express.' Below this are three numbered steps: '1. Enter your old password:', '2. Enter your new password:', and '3. Confirm your new password:'. Each step has a corresponding text input field. At the bottom right of the form are two buttons: 'Change Password' and 'Reset'.

Options Display

Options Screen Functions

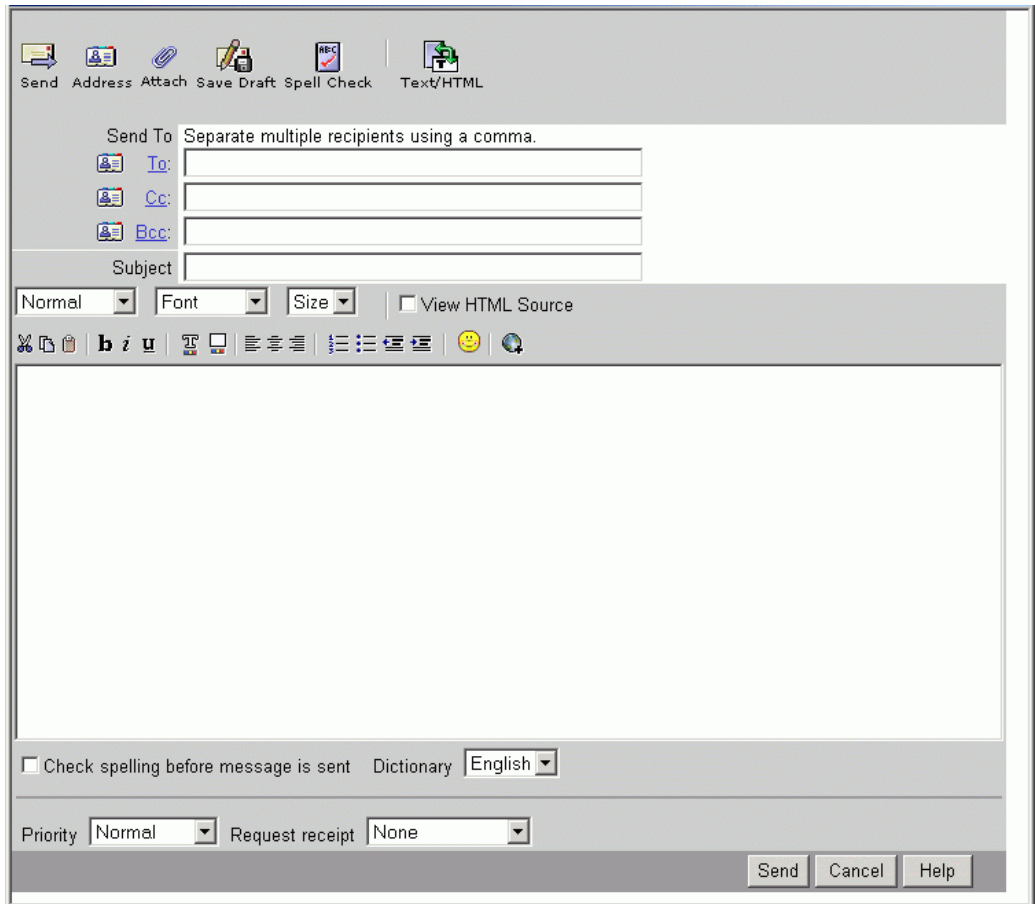
Table 1-6 lists the `parent.` functions (found in `opts_fs.html`) needed to customize the options screen.

Table 1-6 Options Screen Functions

Item	Function
Account Summary	<code>parent.toggle('summary')</code>
Personal Information	<code>parent.toggle('personal')</code>
Password	<code>parent.toggle('password')</code>
Settings	<code>parent.toggle('settings')</code>
Appearance	<code>parent.toggle('appearance')</code>
Vacation Message	<code>parent.toggle('vacation')</code>
Delegated Administrator	<code>parent.toggle('NDA')</code>

Composition Window

The Messenger Express composition window, shown in Figure 1-5, is used primarily to compose a new message. You can also use the window to save a draft or attach a file to the message, look up a recipient in the address book, access the help file, and cancel the composition altogether. Mail recipients can be added in “To”, “Cc”, or “Bcc” fields. You can edit the message in Text or HTML format and also check the spelling. The composition window also enables you to set the mail priority or request for a return receipt.

Figure 1-5 Messenger Express Composition Window

Composition Window Functions

Table 1-7 lists the functions needed to customize the composition window, including main functions (found in `main.js`) and parent functions (found in `comp_fs.html`).

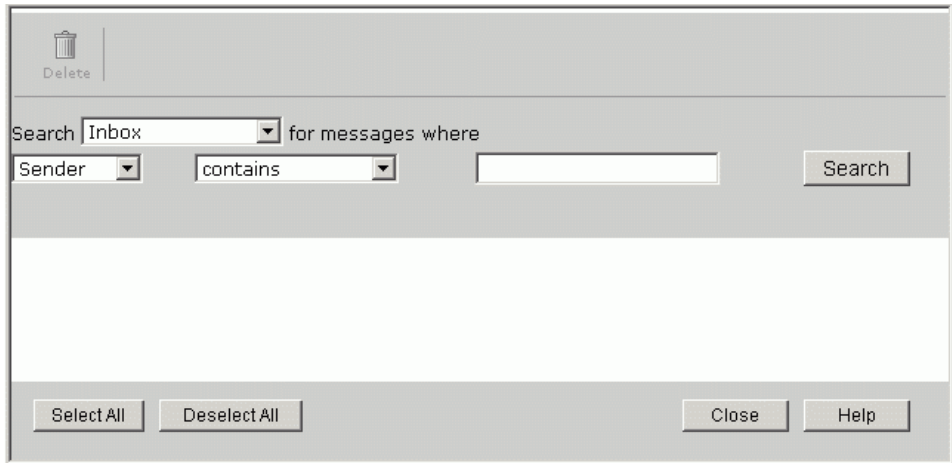
Table 1-7 Composition Window Functions

Item	Function
Send	<code>parent.send('smtp')</code>
Address	<code>parent.lookup()</code>
Attach	<code>main.attach()</code>
Save Draft	<code>parent.send('draft')</code>
Spell Check	<code>parent.spellck()</code>
Help	<code>main.help(1007399)</code>
Cancel	<code>parent.cancel()</code>
To/Cc/Bcc	<code>parent.lookup()</code>
Text/HTML	<code>parent.switchEditor()</code>

Message Search Window

The Messenger Express message search window, shown in Figure 1-6, is primarily used to search messages by entering sender's name, subject, body text, or recipient's name. You can also delete the messages from the search list.

Figure 1-6 Message Search Window



Message Search Window

Customizing General Features

This chapter describes how to customize the general features of iPlanet Messenger Express.

This chapter contains the following sections:

- Modifying the Login Screen
- Modifying Color Sets
- Modifying the Logo and Link
- Modifying the Title Graphic and Text
- Inserting Banners and Links

Modifying the Login Screen

This section describes how to modify the Messenger Express login screen shown in Figure 2-1.

Figure 2-1 Messenger Express Login Screen



You can modify the following on the Messenger Express login screen:

- Replace the logo with a custom graphic
- Change the color scheme
- Replace the service name (for example, Messenger Express)
- Replace images with advertisement banners and links

To Modify the Login Screen

To modify the Login Screen, edit the *lang/default.html* file.

Customize the user interface by editing the body of *lang/default.html*. Functionally, *lang/default.html* contains three forms, two visible and one hidden:

- Username Form (visible)
- Password, Login Button (visible)
- Login Same Window Form (hidden)

The hidden form is the one that is submitted to the server (POST username and password to *login.msc*).

You can also insert a new banner, image or link on the Messenger Express login screen, by editing the *lang/default.html* file.

NOTE *lang* is the language specific file that you need to edit.

Example—Login Screen Modifications

The example shown in Figure 2-2 replaces the iPlanet logo with a custom graphic and adds an advertisement banner with a link.

Figure 2-2 Example Login Screen Modifications



Code Example 2-1 shows the login screen HTML before the edit and shows the changes to be made in the *en/default.html* file to customize the login screen.

Code Example 2-1 Before Altering Login Screen Features

```

.....
<meta http-equiv="Content-Type" content="text/html; ">
<link rel="stylesheet" href="master-style.css" type="text/css">
.....
alt="iPlanet Messaging Server"></td>
.....
<td bgcolor="#cccccc">
.....

```

Code Example 2-1 Before Altering Login Screen Features *(Continued)*

```

<BODY marginwidth="0" marginheight="0" topmargin="0"
leftmargin="0"
rightmargin="0" bottommargin="0" bgcolor="#800000">
....
<td rowspan="4" valign="bottom"></td>
....
<td></td>
....
<td bgcolor="#cccccc">
....

```

Code Example 2-2 After Altering Login Screen Features

```

....
<meta http-equiv="Content-Type" content="text/html; ">
<link rel="stylesheet" href="master-style.css" type="text/css">
....
<BODY marginwidth="0" marginheight="0" topmargin="0"
leftmargin="0"
rightmargin="0" bottommargin="0" bgcolor="#ffffff">
....
<td rowspan="4" valign="bottom"></td>
....
<td><a href="http://www.siroe.com">
</a><BR>

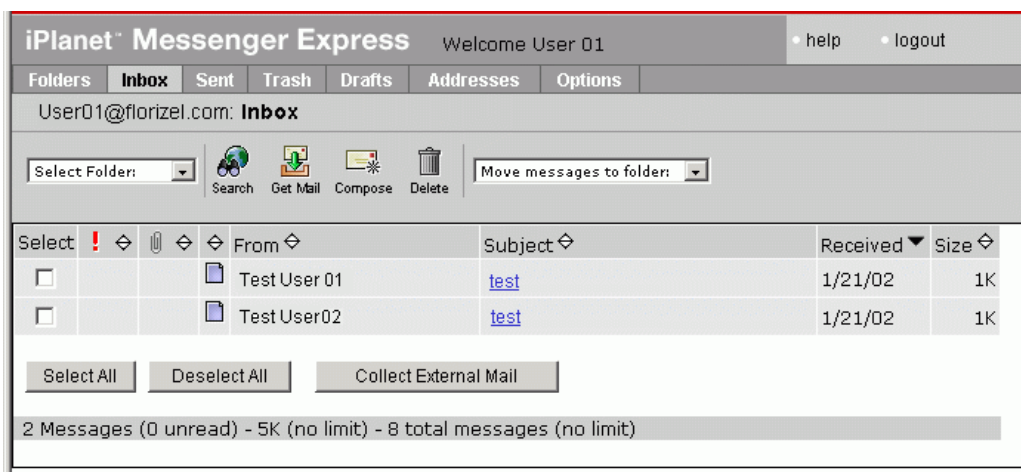
....
<td bgcolor="#C0C0C0">
....

```

Modifying Color Sets

This section describes how to modify the iPlanet Messenger Express user interface color sets shown in Figure 2-3.

Figure 2-3 Messenger Express Color Sets



You can customize the default color sets for the Messenger Express user interface to change items such as the title bar, tab outlines, and column headers.

To Modify Color Sets in the User Interface

To modify color sets in the user interface, edit the `ui[]` array definitions near the top of the `main.js` file.

The function `refreshColorSet()` in `main.js` sets the color scheme of the user interface to color values such as `chromel` and `accent2`. These color values are used by the rest of the display functions in `main.js`.

See `refreshColorSet()` in `main.js` for the translation of the `ui[]` elements to the color values.

The `ui[]` array can have as many rows as desired. Additional color themes are displayed on the preferences page as new rows are defined in `main.js`. The user's JavaScript application will not start when the rows are deleted from the definition script with the user preferences still pointing to a higher color table index than exists in the `ui[]` array.

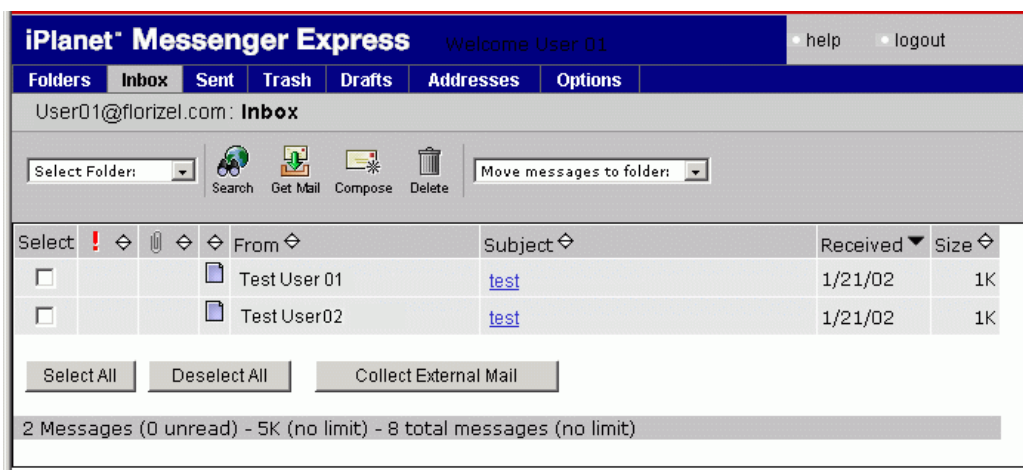
See Table 2-1 for the color index of `ui[]` controls.

Table 2-1 Color Index of `ui[]` Controls

Index	Name	Determines
0	<code>accent0</code>	Not used
1	<code>accent1</code>	Title bar
2	<code>accent2</code>	Not used
3	<code>chrome0</code>	Tab outlines
4	<code>chrome1</code>	Unselected tab background
5	<code>chrome2</code>	Selected tab background, tool bar, column headers, and so on
6	<code>chrome3</code>	Table cell background
7	<code>ch3_img</code>	Not used
8	<code>link0</code>	Unvisited link
9	<code>link1</code>	Visited link (almost always the same as unvisited)
10	<code>link2</code>	Active link
11	<code>positive</code>	Line drawing (white)
12	<code>negative</code>	Line drawing (black)
13	<code>white</code>	Page background color

Example—Color Sets Modifications

The example shown in Figure 2-4 customizes the default color set `ui[0]` to have `accent1` color maroon, `chrome4` color navy blue, and `chrome5` color silver.

Figure 2-4 Example Color Sets Modifications

Code Example 2-3 shows the necessary changes to be made in the file `main.js`.

Code Example 2-3 Altering Color Sets

```
var ui = new Array()

ui[0] = new
array('666699','800000','CCCCFF','666666','000080','C0C0C0',
'E6E6E6','gray90.gif','3333CC','3333CC','333366','FFFFFF',
'000000','FFFFFF','000000')
```

Modifying the Logo and Link

This section describes how to modify the Messenger Express corner logo and link shown in Figure 2-5.

Figure 2-5 Messenger Express Corner Logo and Link



You can modify the following on the Messenger Express corner logo and link:

- Replace logo with custom graphic
- Change destination of link

To Modify the Logo and Link

To modify the logo and link, edit the function `toolFrame()` in the `main.js` file.

Example—Logo Modification

The example shown in Figure 2-6 replaces the iPlanet logo with a custom logo having different dimensions.

Figure 2-6 Example Corner Logo



Code Example 2-5 shows the necessary changes to be made in file `default.html` for altering link.

Code Example 2-4 Altering Logo

```
....  
<table border="0" cellspacing="0" cellpadding="0">  
<tr>  
....
```

Code Example 2-4 Altering Logo

```

.....
<td width="190"></td>
<td rowspan="4" valign="bottom"></td>
.....

```

Code Example 2-5 Altering Link

```

.....
<td colspan="2"><a href="http://www.siroe.com">
onMouseOver="over('bannerlinkone')"
onMouseOut="out('bannerlinkone')"
onClick="loadandswap('bannerlinkone', 'http://www.siroe.com');
return true"> <font size="-1"><span
class="banner-links">siroe.com</span></font></a></td>
</tr>
.....

```

Modifying the Title Graphic and Text

This section describes how to modify the title graphic and the text shown in Figure 2-7.

Figure 2-7 Messenger Express Title Text



You can modify the following on the Messenger Express title graphic and text:

- Replace title graphic with custom graphic
- Change the title text

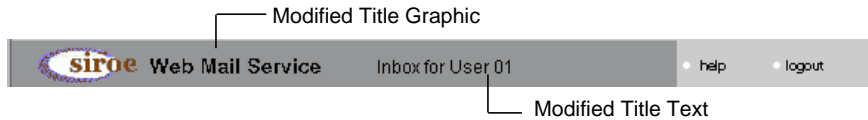
To Modify the Title Graphic and Text

To customize the layout of the title text, edit the function `toolFrame()` in the `main.js` file.

Example—Title Text Modification

The example shown in Figure 2-8 customizes the text to “Inbox for *user*.”

Figure 2-8 Example Title Text Modification



Code Example 2-6 shows the necessary changes to be made in file `main.js` to alter the graphic title.

Code Example 2-6 Altering Graphic Title

```
.....
<TD width=1% nowrap bgcolor="'+ chrome1 +'><NOBR>
<IMG src="imx/Banner.gif width="273" height="27"
alt="'+i18n['iplanet messenger express']+'"
align=texttop border=0 hspace=5 vspace=3
</nobr></TD>
.....
```

Code Example 2-7 shows the necessary changes to be made file `en/i18n.js` to alter the title text.

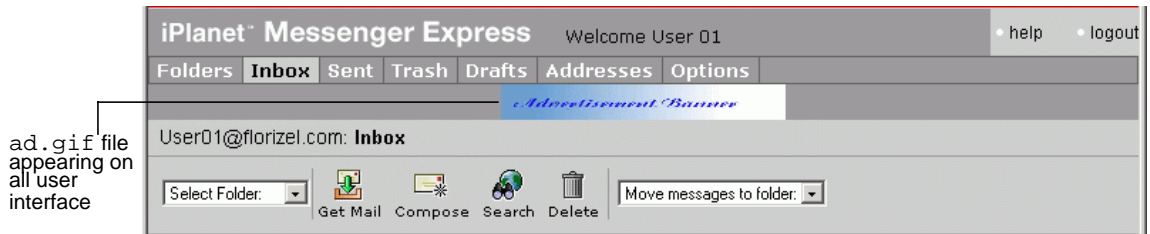
Code Example 2-7 Altering Title Text

```
function i18n_tab_header(user) {
    return '<nobr>&nbsp;&nbsp;&nbsp;Inbox for ' + user
}
```


Inserting Banners and Links

This section describes how to insert advertisement banners and links on the iPlanet Messenger Express user interface shown in Figure 2-9.

Figure 2-9 Example Inserting Common Banner and Link



You can modify the following on the Messenger Express user interface:

- Add a banner and link to appear on all user interface
- Add a banner and link to appear only on certain user interface

To Insert Banner and Link

- Edit the function `toolFrame()` in the `main.js` file to add a common banner and link.
- Edit the function `getToolbar()` and `frameset` height in the files `comp_fs.html`, `fldr_fs.html`, `mbox_fs.html`, `msg_fs.html`, `opts_fs.html`, and `searchmsg_fs.html` to add separate banners and links.

Example—Inserting Banners and Links

The example shown in Figure 2-10 inserts a banner and link in the Messenger Express Inbox screen.

Figure 2-10 Example Inserting a Banner and Link in the Inbox Screen



Code Example 2-8 shows the code before inserting banner and link on all user interface in the `main.js` file.

Code Example 2-8 Before Inserting a Banner and Link

```
function toolFrame() {
  if (isRefreshing())
    return ''
  ....
  tab(i18n['options'], state == 'options', 'selectOptions()') +
  '<td width=50%>' + nbsp + '</td>\n' +
  '</tr></table>\n' +
  '</td>\n' +
  '</tr></table>\n' +
  getBreadCrumb() +
  mailFrame.getToolbar() + '</form>'
}
```

Code Example 2-9 shows how to insert a banner and link to the `main.js` file.

Code Example 2-9 After Inserting a Banner and Link

```
function toolFrame() {
  if (isRefreshing())
    return ''
  ....
  tab(i18n['options'], state == 'options', 'selectOptions()') +
```

Code Example 2-9 After Inserting a Banner and Link (*Continued*)

```
'<td width=50%>' + nbsp + '</td>\n' +
'</tr></table>\n' +
'</td>\n' +
'</tr><tr><td width=100% bgcolor='+main.chromel+'
align=center><a href="http://www.sesta.com">
</a></td></tr></table>\n' + getBreadCrumb() +
mailFrame.getToolbar() + '</form>'
}
```

Code Example 2-10 shows the code before inserting banner and link to the Inbox screen, in `mbox_fs.html` file.

Code Example 2-10 Before Inserting Banner and Link to the Inbox Screen

```
function getToolbar() {
    var s
    var enable = main.sortid.length > 0
    s = '<table border="0" cellpadding="0" cellspacing="0">\n' +
        '<TR><TD colspan=42><IMG src="imx/spacer.gif" width="1"
        height="8"></TD></TR>' +
        '<tr align="center">\n' +
        '<td width="10">&nbsp;</td>\n' + '<td align="center" nowrap>'
    + main.font(1) +
    main.folderSelectionBreadCrumbs('folderList', 'main.selectMbox(op
    tions[selectedIndex].value);selectedIndex=0',
    ....
```

Code Example 2-11 shows how to insert a banner and link to the Inbox screen by editing `mbox_fs.html`. The banner and link are displayed only on the Inbox screen.

Code Example 2-11 After Inserting Banner and Link to the Inbox Screen

```
function getToolbar() {
    var s
    var enable = main.sortid.length > 0
    s = '<table border="0" cellpadding="0" cellspacing="0">\n' +
        '<TR><TD width=100% align=center>' +
        '
        <a href ="http://www.siroe.com"></a></td></tr>' +
        '<tr align="center">\n' +
```

Code Example 2-11 After Inserting Banner and Link to the Inbox Screen *(Continued)*

```
'<td width="10">&nbsp;</td>\n' + '<td align="center" nowrap>'
+ main.font(1) +
main.folderSelectionBreadCrumbs('folderList', 'main.selectMbox(op
tions[selectedIndex].value);selectedIndex=0',
....
```

Customizing User Interface Features

This chapter describes how to customize user interface features of iPlanet Messenger Express.

This chapter contains the following sections:

- Modifying the Main Function Tabs
- Modifying the Mailbox Tool Bar
- Modifying the Message List Window
- Modifying the Message Display Window
- Messenger Express Message Tool Bar
- Modifying the Message Composition Window
- Modifying Message Search Window
- Modifying the Address (Directory Lookup) Window
- Modifying the Options Window

Modifying the Main Function Tabs

This section describes how to modify the Messenger Express main function tabs shown in Figure 3-1.

Figure 3-1 Messenger Express Main Function Tabs, With Default Labels



You can modify the following on the Messenger Express main function tabs:

- Interchange the location of tabs
- Change the text of the tab labels

To Modify the Main Function Tabs

To modify the main function tabs, edit the appropriate files as follows:

- For tab layout, edit the `toolFrame()` function in the `main.js` file.
- For text used in the default tab labels, in the `lang/i18n.js` file, edit the `i18n[]` values for `folders`, `message`, and `options` in the `// Tabs` section, and `help` and `logout` in the `// Tool Bars` section.
- For text of the default folder name labels (including the initially displayed inbox tab label), edit the `fldr[]` values in the `// Localized folder names` section in the `lang/i18n.js` file.

Functionally, the tabs are constructed by the `toolFrame()` function. The `toolFrame()` function calls the `tab()` function in the `main.js` file and specifies the text of the tab label to be displayed.

The following functions, located in `main.js`, handle the default tabs:

- **Folders:** `displayFolders()`
- **Inbox:** `displayMbox()` or `refreshMbox()` (depending on the state selected)
- **Message:** `selectMsg()`
- **Options:** `selectOptions()`
- **Help:** `help()`
- **Logout:** `logout()`

Example—Main Function Tabs Modifications

The example shown in Figure 3-2 moves the Options tab to the right, and changes the text of its tab label to “Preferences.”

Figure 3-2 Example Main Function Tabs Modifications



Code Example 3-1 and Code Example 3-2 show the necessary changes to be made in files `main.js` (layout) and `en/i18n.js` (tab labels).

Code Example 3-1 Altering Function Tabs Layout (`main.js`)

```
function toolFrame() {
  .....
  '<td width=30%>' + nbsp + '</td>\n' +
  tab(i18n['options'], state == 'Options', 'selectOptions()') +
  .....
}
```

Code Example 3-2 Altering Function Tabs Labels (`en/i18n.js`)

```
// Tabs
i18n['folders'] = 'Folders'
i18n['message'] = 'Message'
i18n['pab'] = 'Addresses'
i18n['options'] = 'Preferences'
```

Modifying the Mailbox Tool Bar

This section describes how to modify the Messenger Express mailbox tool bar shown in Figure 3-3.

Figure 3-3 Messenger Express Mailbox Tool Bar



You can modify the following on the Messenger Express mailbox tool bar:

- Change the layout of the mailbox tool bar relative to the rest of the page
- Rearrange the order of tools
- Change the tools text

To Modify the Mailbox Tool Bar

To modify the mailbox tool bar, edit the appropriate files as follows:

- To customize the layout relative to the rest of the page, edit the `toolFrame()` function in the `main.js` file.
- To customize the layout within the tool bar and the associated graphics, edit the `getToolbar()` function in the `mbox_fs.html` file.
- To customize the text associated with the graphics in the Tool Bar, edit the `i18n[]` values `get mail`, `compose`, `search`, `new search`, `file selected message`, `delete`, `undelete`, and `expunge` in the `lang/i18n.js` file.

Functionally, `toolFrame()` in `main.js` calls `getToolbar()` in `mbox_fs.html` to get the HTML code to write out to the page.

The `getToolbar()` function in `mbox_fs.html` assembles the code and assigns the functions to the graphics by calling `toolbar()` in `main.js`, which takes care of items such as colors and text-only versions.

The `getToolbar()` function in `mbox_fs.html` also calls `folderSelection()` in `main.js` to generate the drop-down folder list.

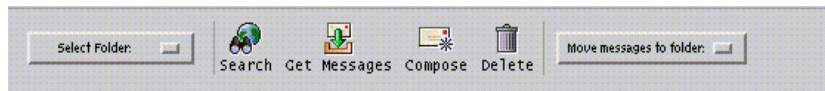
The functions assigned by `getToolbar()` in `mbox_fs.html` that handle the tool clicks are:

- **Get Mail:** `main.refreshMbox()`
- **Compose:** `main.compose("new")`
- **Search:** `parent.srch()`
- **Move Messages to Folder:** `parent.move()`
- **Delete:** `parent.delmsg()`, `parent.undelmsg()`, `parent.exmsg()` (depending on whether the message is in the trash folder or not)

Example—Mailbox Tool Bar Modifications

The example shown in Figure 3-4 makes “Search” as the first tool and changes the text of the “Get Mail” tool to “Get Messages.”

Figure 3-4 Example Mailbox Tool Bar Modifications



Code Example 3-3 and Code Example 3-4 show the necessary changes to be made in files `mbox_fs.html` (layout) and `en/i18n.js` (text).

Code Example 3-3 Altering Tool Bar Layout (`mbox_fs.html`)

```
function getToolbar() {
    ....
    main.WMtoolbar(
        ....
        (main.srch != '' ? i18n['new search'] : i18n['search']),
        'parent.srch()', 'imx/search.gif', 27, 25, true,
        i18n['get mail'], 'main.refreshMbox()', 'imx/pull.gif', 27, 25,
        true, i18n['compose'], 'main.compose("new")', 'imx/compose.gif',
        27, 25, true)
        ....
    }
}
```

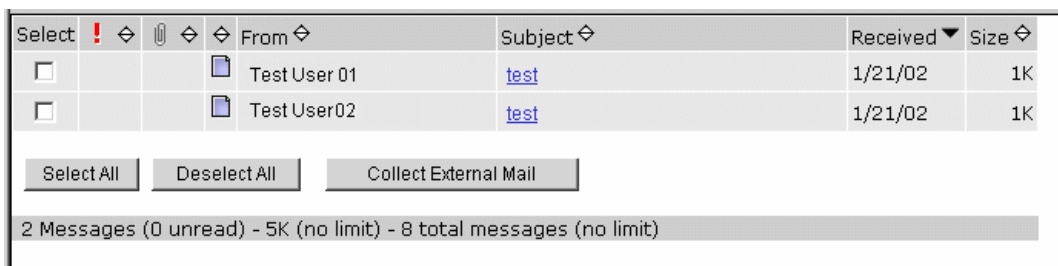
Code Example 3-4 Altering Tool Bar Text (en/i18n.js)

```
// Tool Bars
....
i18n['get mail'] = 'Get Messages'
```

Modifying the Message List Window

This section describes how to modify the Messenger Express message list window shown in Figure 3-5.

Figure 3-5 Messenger Express Message List Window



You can modify the following in the Messenger Express message list window:

- By default change the sorting order
- Change the text of the default column heading
- Change the text on “Collect External Mail” button

To Modify the Message List Window

To modify the message list window, edit the appropriate files as follows:

- To customize how the message list appears, edit the `listFrameHTML()` function in the `mbox_fs.html` file.
- To customize the order in which messages are listed, edit the `defaults[]` values near the top of the `main.js` file.
- To change the text of default column heading, edit the `i18n[]` values for search results, date, from, to, size, and subject in the `lang/i18n.js` file.

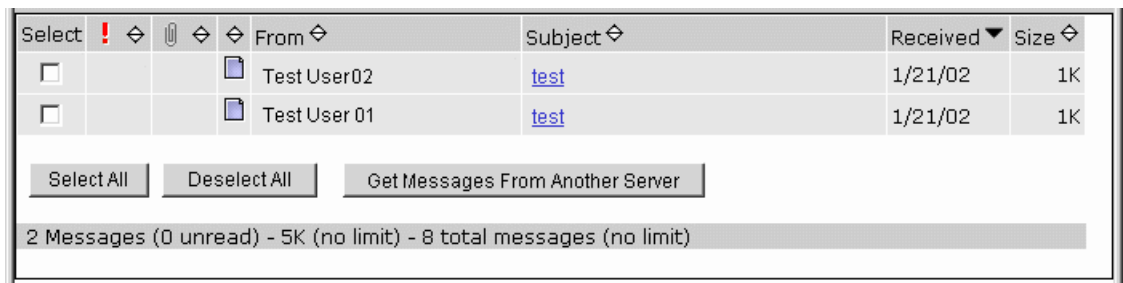
- To change the text on the “Collect External Mail” button, edit `i18n['collect long]` in the `lang/i18n.js` file.

Functionally, `listFrameHTML()` calls `getSortHeader()` in `mbox_fs.html` and assigns column headings with appropriate call to the sorting function `sortMsgs()` in `main.js`. The `listFrameHTML()` function also links the “Collect External Mail” hyperlink to `collect()` in `main.js`.

Example—Message List Window Modifications

The example shown in Figure 3-6 displays the most recently received mails first, and changes the text on “Collect External Mail” button to “Get Messages From Another Server”.

Figure 3-6 Example Message List Window Modifications



Code Example 3-5 and Code Example 3-6 show the necessary changes to be made in files `main.js` and `en/i18n.js`.

Code Example 3-5 Altering Message List Window Layout (`main.js`)

```
var defaults = new Array(
    .....
    'meSortOrder', 'L',
    .....
)
```

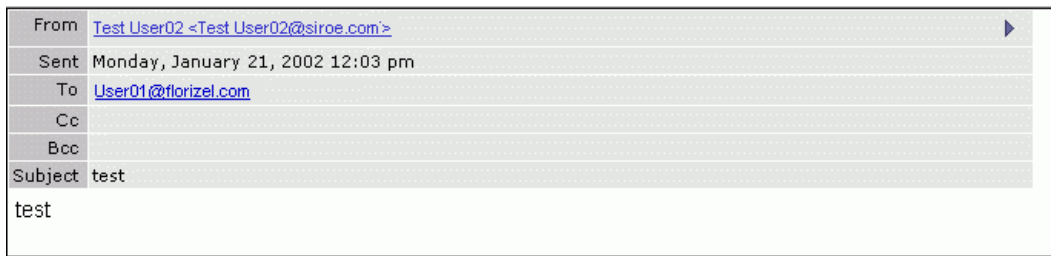
Code Example 3-6 Altering List Window Text (en/i18n.js)

```
// POP3 Collection
....
i18n['collect long'] = 'Get Messages From Another Server'
```

Modifying the Message Display Window

This section describes how to modify the Messenger Express message display window shown in Figure 3-7.

Figure 3-7 Messenger Express Message Display Window



You can modify the following in the Messenger Express message display window:

- Change the message display
- Alter the layout of, window
- Change the text of the fields

To Modify the Message Display Window

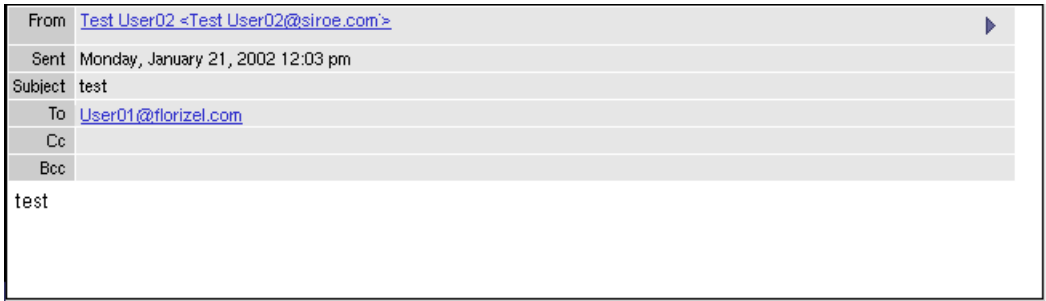
To modify the message display window, edit the appropriate files as follows:

- To customize how the message appears, edit the `listFrameHTML(doc)` function in the `msg_fs.html` file.
- To customize the default text, edit the `i18n[]` values under `// Message Headers` and `// Message` in the `lang/i18n.js` file.
- To customize the display defaults (word wrap), edit the `defaults[]` values in the `main.js` file.

Example—Message Display Window Modifications

The example shown in Figure 3-8 moves “Subject” before “To.”

Figure 3-8 Example Message Display Window Modifications



Code Example 3-7 shows the necessary changes to be made in file `msg_fs.html`.

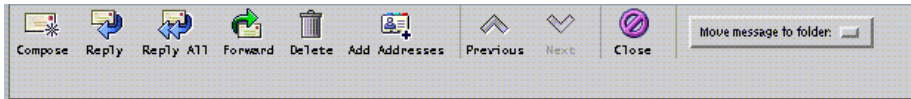
Code Example 3-7 Altering Message Display Window Layout

```
function listFrameHTML(doc) {
  ....
  s += header('from') + header('date') + header('subject')+
  header('to') + header('cc')
  ....
}
```

Modifying the Message Tool Bar

This section describes how to modify the Messenger Express message tool bar shown in Figure 3-9.

Figure 3-9 Messenger Express Message Tool Bar



You can modify the following in the Messenger Express message tool bar:

- Change the layout of the tool bar relative to the rest of the page
- Alter the layout within the tool bar
- Change the text associated with the graphics

To Modify the Message Tool Bar

To modify the message tool bar, edit the appropriate files as follows:

- To customize the layout relative to the rest of the page, edit the `toolFrame()` function in the `main.js` file.
- To customize the layout within the tool bar and the associated graphics, edit the `getToolbar()` function in the `msg_fs.html` file.
- To customize the texts associated with the graphics in the Tool Bar, edit the `i18n[]` values `compose`, `reply`, `reply all`, `forward`, `delete`, `undelete`, `previous`, and `next` in the `lang/i18n.js` file.

Functionally, `getToolbar()` in `msg_fs.html` assembles the code and assigns the functions to the graphics by calling `toolbar()` in `main.js`, which takes care of items such as colors and text-only versions.

The `getToolbar()` function in `msg_fs.html` also calls `folderSelection()` in `main.js` to generate the drop-down folder list.

The functions assigned by `getToolbar()` in `msg_fs.html` to handle the tools are:

- **Compose:** `main.compose('new')`
- **Reply:** `main.compose('reply')`

- **Reply All:** `main.compose('replyall')`
- **Forward:** `main.compose('forward')`
- **Move Messages to Folder:** `parent.move()`
- **Delete and Undelete:** `parent.delmsg()`
- **Add all Addresses:** `parent.addAllAddresses()`
- **Previous:** `parent.prev()`
- **Next:** `parent.next()`
- **Close:** `parent.gotofolder()`

Example—Message Tool Bar Modifications

The example shown in Figure 3-10 moves “Compose” to the right and abbreviates the text from “Previous” to “Prev.”

Figure 3-10 Example Message Tool Bar Modifications



Code Example 3-8 and Code Example 3-9 show the necessary changes to be made in files `msg_fs.html` and `en/i18n.js`.

Code Example 3-8 Altering Message Tool Bar Layout (`msg_fs.html`)

```
function getToolbar() {
var s
...
main.WMtoolbar(
  null,null,'imx/spacer.gif',5,1,false,
  i18n['close'], 'parent.gotofolder()', 'imx/cancel.gif" alt="'
+ i18n['msg close'], 24, 24, true,
  null,null,'imx/spacer.gif',5,1,false,
  i18n['compose'], 'main.compose("new")', 'imx/compose.gif" alt="'
+ i18n['msg compose'], 24, 24, true) +
  <td></td>\n' +
  ....
}
```

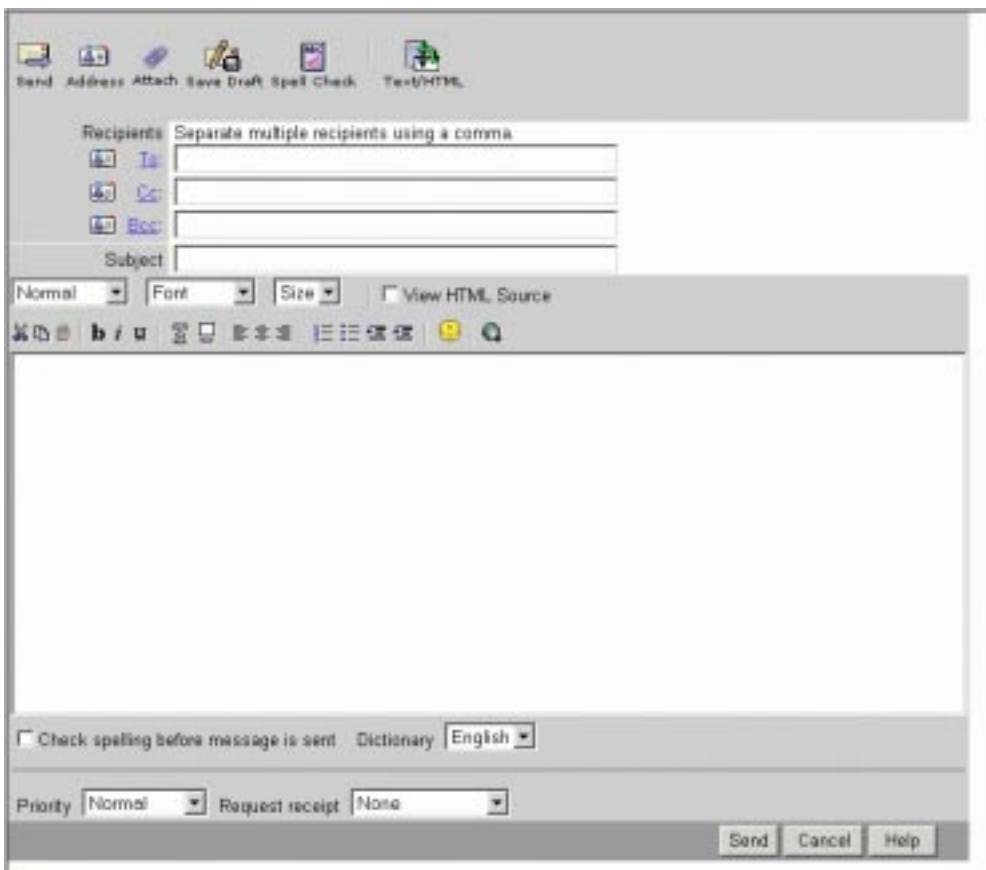
Code Example 3-9 Altering Message Tool Bar Text (en/i18n.js)

```
// Tool Bars
....
i18n['previous'] = 'Prev'
```

Modifying the Message Composition Window

This section describes how to modify the Messenger Express message composition window shown in Figure 3-11.

Figure 3-11 Messenger Express Message Composition Window



You can modify the following in the Messenger Express message composition window:

- Change the location of the tools in the window
- Alter text associated with the tools
- Enable and disable emoticons
- Create your own dictionary

To Modify the Message Composition Window

To modify the message composition window, edit the appropriate files as follows:

- To customize the window, edit the `compFrameHTML()` function in the `comp_fs.html` file.
- To customize the text, edit the values under `// Message Composition and // Tool Bars` in the `lang/i18n.js` file.
- To enable and disable the emoticons, edit the variable `iconHref` in the `main.js` file. By default, the emoticon files are located in `server-root/html/imx` directory.

Functionally, `compFrameHTML()` in `comp_fs.html` assembles the code and assigns the functions to the graphics by calling `WMtoolbar()` in `main.js` which also handles colors and text-only versions. The `compFrameHTML()` function in `comp_fs.html` generates the “To”, “Cc”, and “Bcc” control area by calling `i18n_compose_controls()` in `lang/i18n.js`.

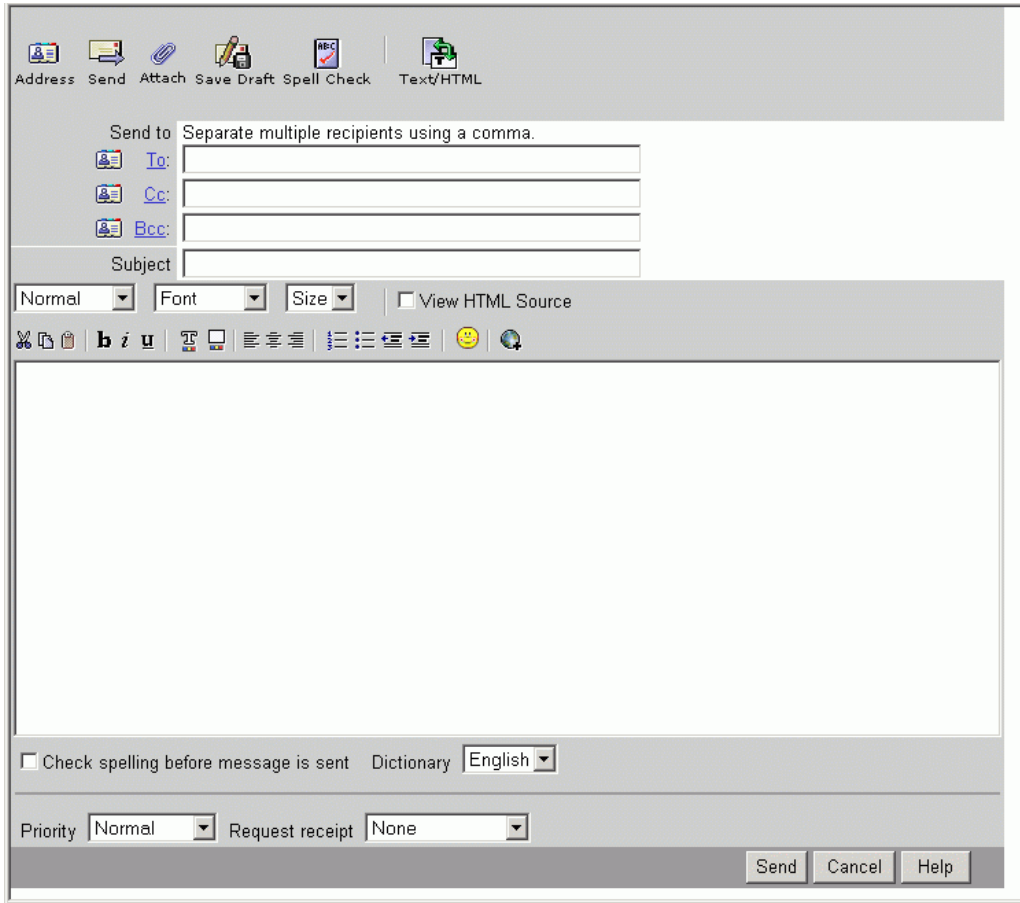
The functions assigned by `compFrameHTML()` in `comp_fs.html` are:

- **Send:** `parent.send('smtp')`
- **Address:** `parent.lookup()`
- **Attach:** `parent.attach()`
- **Save Draft:** `parent.send('draft')`
- **Spell Check:** `parent.spellchk()`
- **Text/HTML:** `parent.switchEditor()`
- **Help:** `main.help(1007399)`
- **Cancel:** `parent.cancel()`

Example—Message Composition Window Modifications

The example shown in Figure 3-12 moves the Address tool first, and changes the text “Recipients” to “Send to.”

Figure 3-12 Example Message Composition Window Modifications



Code Example 3-10 and Code Example 3-11 show the necessary changes to be made in files `comp_fs.html` and `en/i18n.js`.

Code Example 3-10 Altering Composition Window Layout (comp_fs.html)

```
function compFrameHTML() {
    ....
    main.WMtoolbar(i18n['lookup'], 'parent.lookup()',
        'imx/address.gif"
    alt="' + i18n['lookup'], 27, 25, true, i18n['send'],
    'parent.send('smtp')', 'imx/send.gif " alt="'
    + i18n['compose_send'],27, 25, true)
    ...
}
```

Code Example 3-11 Altering Composition Window Text(en/i18n.js)

```
// Message Composition
....
i18n['recipient'] = 'Send To'
```

The emoticons appear on the screen if the Text/HTML option is set to HTML. By default the Text/HTML option is set to Text format.

Code Example 3-12 shows how to edit the `main.js` file to enable emoticons.

Code Example 3-12 Altering Composition Window to Enable Emoticons

```
var iconHREF = 'server-root/imx/'
```

Code Example 3-13 shows how to edit `main.js` file to disable emoticons.

Code Example 3-13 Altering Composition Window to Disable Emoticons

```
var iconHREF = ''
```

To Include Additional Dictionary for Spell Check

1. Get the dictionary file and the affix file for the language you want to add to your dictionary.

The dictionary file contains language-specific vocabulary and the affix file contains grammar rules for the specific language. For information on dictionary and affix files refer to the site:

<http://fmg-www.cs.ucla.edu/fmg-members/geoff/ispell-dictionaries.html>

Messenger Express is shipped with French and English (United States) dictionaries, which are located in the *server-root/dict* directory.

2. Use the `buildhash` utility to create a platform-specific and language-specific hash file from the dictionary and affix files. This hash file is used by the Messenger Express spell checker.

To run the `buildhash` utility, download the `ispell` source files available at the site: <http://www.gnu.org/software/ispell/ispell.html> or, use the `buildhash` utility in the *server-root/dict/bin* directory. The syntax for the `buildhash` utility is:

```
buildhash dictionary_file affix_file language_name.hash
```

The `language_name` in the *language_name.hash* file is the two-letter language code used by Messenger Express (such as: `en` for English, `fr` for French).

To determine your language's two-letter code, enter the command:

```
server-root/msg-instance/configutil | grep local.supportedlanguages
```

NOTE The double-byte character set is not supported by the Messenger Express spell checker.

3. Copy the newly created *language_name.hash* file in the *server-root/dict* directory and restart the `mshttpd` service. When the `mshttpd` service is restarted, the Messenger Express spell checker is enabled.

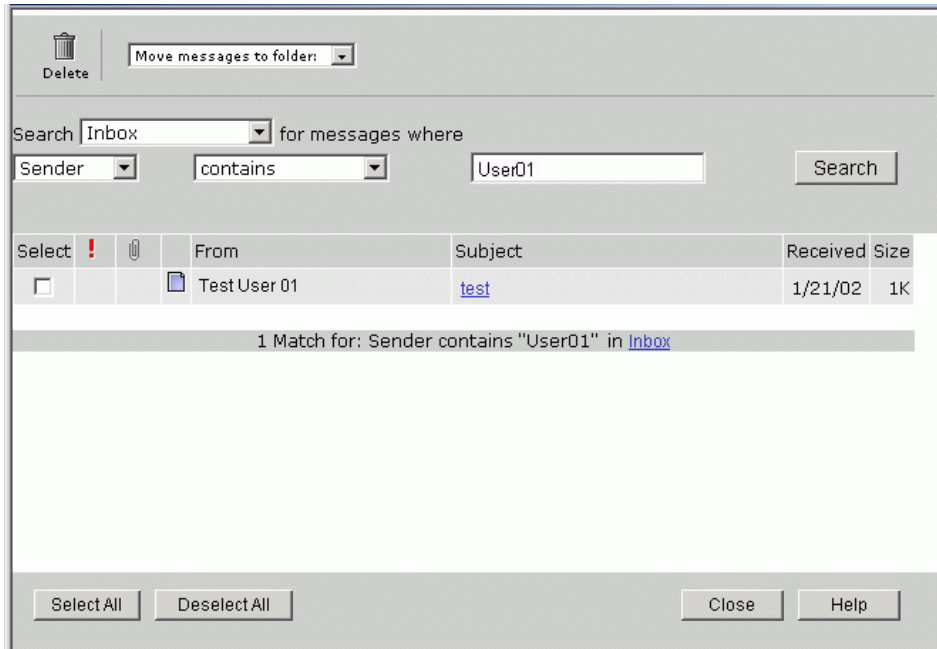
For example, if you want to create a German hash file (`ge.hash`) by using the `buildhash` utility, enter:

```
# cd /usr/iplanet/server5/dict/bin
# ./buildhash german.dico german.aff ge.hash
# cp ge.hash ..
# /usr/iplanet/server5/msg-budgie/start-msg http
```

Modifying Message Search Window

This section describes how to modify the Messenger Express message search window shown in Figure 3-13.

Figure 3-13 Message Search Window



You can modify the following on the Messenger Express message search window:

- Change the layout of the window
- Change the text associated with the tool
- Change the tool bar layout in the message view window

To Modify the Message Search Window

To modify the message search window, edit the appropriate files as follows:

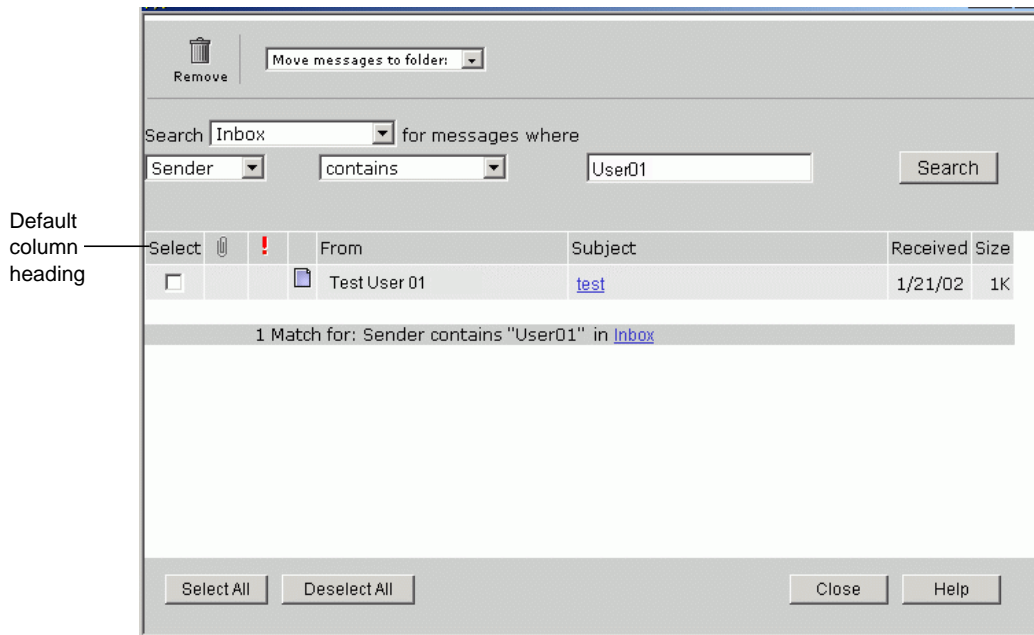
- To alter the layout of the window, edit the `listFrameHTML()` function in the file `/en/searchMessage.html`.

- To change the text associated with the tool, edit the file `en/i18n.js`.
- To alter the tool bar in the message view window, edit the `getToolBar()` function in the `seachmsg_fs.html` file.

Example—Message Search Window Modifications

The example shown in Figure 3-14 changes the text associated with the tool “Delete” to “Remove”. It also shows how to interchange the appearance of the default column headings.

Figure 3-14 Message Search Window With Changes



Code Example 3-14 shows the changes to be made in file `en/i18n.js`, to change the text associated with the tool.

Code Example 3-14 Altering the Tool Text

```
// Tool Bars
i18n['delete'] = 'Remove'
```

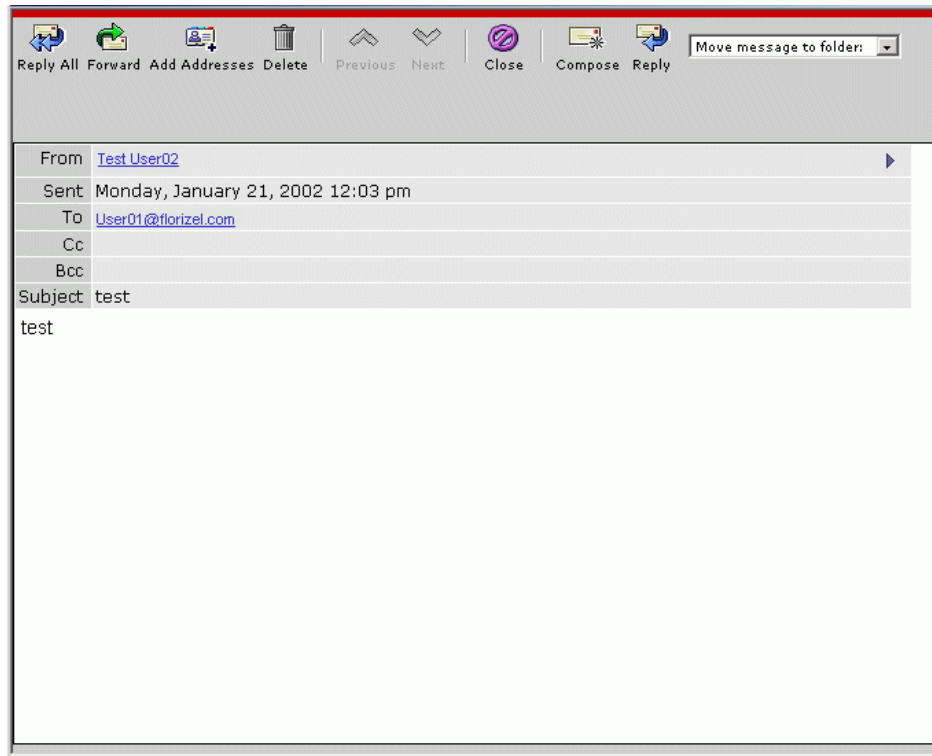
Code Example 3-15 shows the necessary changes to be made in the `searchMessage.html` file.

Code Example 3-15 Interchanging the Order of Appearance of the Default Column Headings

```
function listFrameHTML() {
    var i, msg
    ....
    s += main.tableStart + '<form name="form1"><tr bgcolor=' +
    main.chrome2 + '>\n'
    s += '<td align="center" width=1% nowrap>' + main.font() +
        i18n['selectLabel'] + '</td>\n'
    s += '<td align="center" width=5% nowrap>= 4 || NN > 0 ? ' hspace=2>' : '>') + getSortHeader('') +
        '</td>\n'
    s += '<td align="center" width=1% nowrap>= 4 || NN > 0 ? ' hspace=2>' : '>') + getSortHeader('') +
        '</td>\n'
    s += '<td align="left" width=1% nowrap>' + getSortHeader('') +
        '</td>\n'
    ...
}
```

The example shown in Figure 3-15 alters the tool bar layout in the message view window.

Figure 3-15 Message View Window with the Tool bar Modifications



Code Example 3-16 shows the necessary changes to be made in the file `seachmsg_fs.html` to alter the tool bar layout.

Code Example 3-16 Message View Window Tool Bar Modifications

```
function getToolbar() {
    var s
    ....
    main.WMtoolbar(
        ....
        i18n['close'], 'parent.closeMe()', 'imx/cancel.gif" alt="' +
        i18n['msg close'], 27, 25, true,
        null, null, 'imx/divider.gif', 2, 24, false,
        i18n['compose'], 'main.compose("new")', 'imx/compose.gif" alt="' +
        + i18n['msg compose'], 27, 25, true,
        i18n['reply'], 'main.compose("reply")', 'imx/reply.gif" alt="' +
        i18n['msg reply'], 27, 25, enable) +
        '<td nowrap>' + main.font(1) + main.nbsp +
```

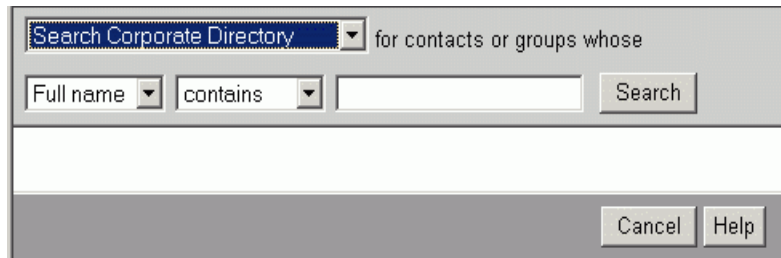

Code Example 3-16 Message View Window Tool Bar Modifications *(Continued)*

```
function getToolBar() {
    main.folderSelection('folderList',
        'parent.move(options[selectedIndex].value);selectedIndex=0',
        i18n['file msg'], false,
        main.mboxFrame.mbox, '', getsharedfolders())+ '</td>' +
    ....
}
```

Modifying the Address (Directory Lookup) Window

This section describes how to modify the Messenger Express address (directory lookup) window shown in Figure 3-16.

Figure 3-16 Messenger Express Address (Directory Lookup) Window



You can modify the following in the address (directory lookup) window:

- Change overall window appearance
- Alter search controls and their text
- Add additional LDAP server search
- Alter all other texts

To Modify the Address (Directory Lookup) Window

To modify the address window, edit the appropriate files as follows:

- To customize the overall window appearance, edit the `getSearchResult()` and `idxHTML()` functions in the `lookup.js` file.
- To customize other text, edit the `// LDAP Lookup` values in the `lang/lookup_fs.html` file.
- To add additional LDAP server search, add new entries specifying the LDAP host followed by the DN as the third parameter, edit the `lang/lookup_fs.html` file.

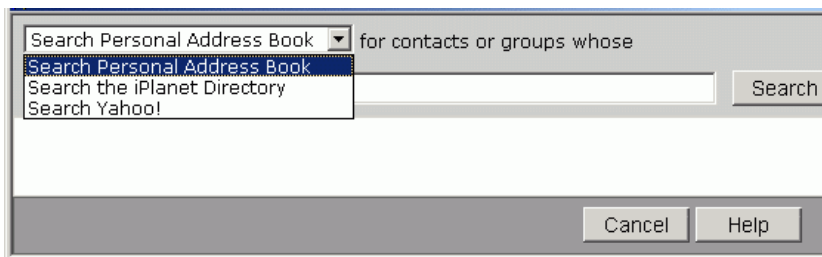
Functionally, `searchFrameHTML()` and `addFrameHTML()` assign the following functions to the buttons:

- Search: `parent.doSearch()`
- Cancel: `parent.cancel()`

Example—Address (Directory Lookup) Window Modifications

The example shown in Figure 3-17 changes “Search Corporate directory” to “Search the iPlanet Directory” and adds the LDAP server search “Search Yahoo!” to the search list.

Figure 3-17 Example Address (Directory Lookup) Window Modifications



Code Example 3-17 shows the necessary changes to be made in `en/lookup_fs.html`.

Code Example 3-17 Altering Address Window Text

```
//Search Control
function s_SearchCtrl() {
....
'<option value="3 200">Search the iPlanet Directory</option>\n' +
....
}
```

Code Example 3-18 shows how to add LDAP server search to the list. The file to edit is `en/lookup_fs.html`.

Code Example 3-18 Adding LDAP Server Search

```
//Search Control
function s_SearchCtrl() {
...
'<option value="2 25 ldap://ldap.yahoo.com/">' +
'Search Yahoo!</option>\n' +
'</select>\n' +
...
}
```

Modifying the Options Window

This section describes how to modify the Messenger Express options window shown in Figure 3-18.

Figure 3-18 Messenger Express Options Window

The screenshot shows a web interface with a left-hand navigation menu and a main content area. The navigation menu includes links for Account Summary, Personal Information, Password, Settings, Appearance, Vacation Message, and Delegated Administrator. The main content area is titled "Password" and contains the instruction: "Use this form to change the password you use to access Messenger Express." Below this instruction are three numbered steps, each with a text input field: "1. Enter your old password:", "2. Enter your new password:", and "3. Confirm your new password:". At the bottom right of the form are two buttons: "Change Password" and "Reset".

You can modify the following on the Messenger Express options window:

- Change the layout of the window
- Change the default text

To Modify the Options Window

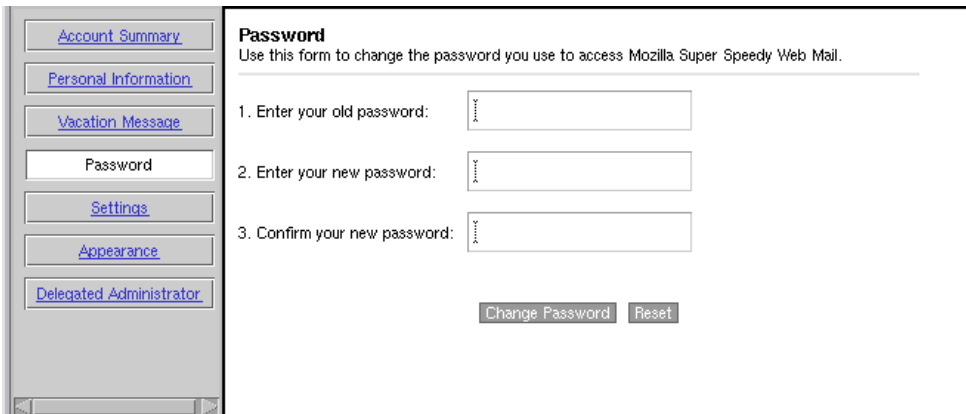
To modify the options window, edit the appropriate files as follows:

- To customize the options and the layout of the options, edit the `toggleFrameHTML()` function in the `opts_fs.html` file.
- To customize default text, edit the `i18n[]` values under `// Options` in the `lang/i18n.js` file.

Example—Options Window Modifications

The example shown in Figure 3-19 moves “Vacation Message” between “Personal Information” and “Password,” and changes the text “Messenger Express” to “Mozilla Super Speedy Web Mail.”

Figure 3-19 Example Options Window Modifications



Code Example 3-19 and Code Example 3-20 show the necessary changes to be made in the files `opts_fs.html` and `en/i18n.js`.

Code Example 3-19 Altering Options Window Layout (`opts_fs.html`)

```
function toggleFrameHTML() {
    ....
    getToggle(main.i18n['personal'], 'personal',
    'javascript:parent.toggle("personal")') +
    getToggle(main.i18n['vacation'], 'vacation',
    'javascript:parent.toggle("vacation")') +
```

Code Example 3-19 Altering Options Window Layout (`opts_fs.html`) (*Continued*)

```

getToggle(main.i18n['password'], 'password',
'javascript:parent.toggle("password")') +
.....
}

```

Code Example 3-20 Altering Options Window Text (`en/i18n.js`)

```

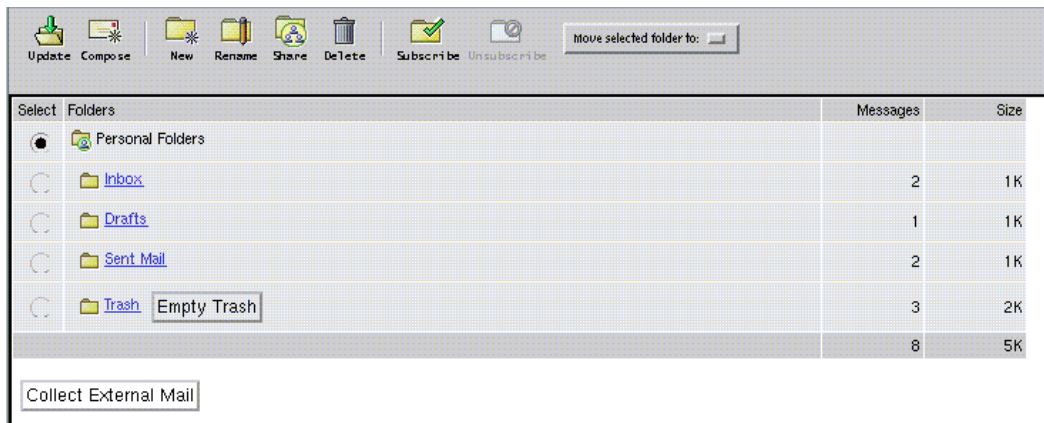
// Options
.....
i18n['passwd exp'] = 'Use this form to change the password you
use to access Mozilla Super Speedy Web Mail.'

```

Modifying the Folders Window

This section describes how to modify the Messenger Express folders window shown in Figure 3-20.

Figure 3-20 Messenger Express Folders Window



You can modify the following in the Messenger Express folders window:

- Change the location of the tools
- Change the text associated with the tools or folders

To Modify the Folders Window

To modify the folders window, edit the appropriate files as follows:

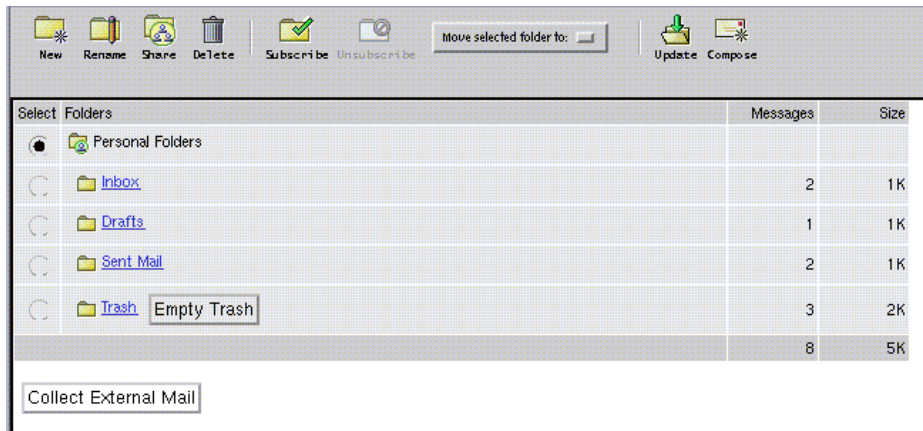
- To rearrange the tools on the folders toolbar, edit the `getToolbar()` function in the `fldr_fs.html` file.
- To customize the column headings, appearance of the buttons, and color of the window, edit the `listFrameHTML()` function in the `fldr_fs.html` file.
- To customize the text of the “Collect External Mail” button, edit `i18n[‘collect long’]` in the `lang/i18n.js` file.
- To customize any other text, edit the `// Folders` section in `lang/i18n.js`.

The functions assigned to the tools and links by `getToolbar()` and `listFrameHTML()` in `fldr_fs.html` are:

- **Update:** `main.refreshFolders()`
- **Compose:** `main.compose(‘new’)`
- **New:** `parent.addFolder()`
- **Rename:** `parent.renFolder()`
- **Share:** `parent.setfolder()`
- **Move Folder:** `parent.moveFolder(options[selectedIndex].value)`
- **Delete:** `parent.delFolder()`
- **Subscribe:** `main.subscribeFolder()`
- **Unsubscribe:** `parent.unsubscribeFolder()`

Example—Folders Window Modifications

The example shown in Figure 3-21 moves “Update” and “Compose” tools to the end of the toolbar.

Figure 3-21 Example Folders Window Modifications

Code Example 3-21 shows the changes to be made in the file `fldr_fs.html`.

Code Example 3-21 Altering Folders Window Layout

```
function getToolbar() {
    ....
    main.WMtoolbar(
        i18n["new folder"], 'parent.addFolder()', 'imx/fldr_new.gif"
        alt="' + i18n['folder new'], 24, 24, true,
        i18n['rename'], 'parent.renFolder()', 'imx/fldr_edit.gif" alt="' +
        i18n['folder rename'], 24, 24, true,
        ....
    main.WMtoolbar(
        null, null, 'imx/spacer.gif', 2, 24, false,
        i18n['update'] ? i18n['update'] : i18n['get mail'],
        'main.refreshFolders()', 'imx/Update_Folder.gif" alt="' +
        i18n['folder update'], 24, 24, true,
        i18n['compose'], 'main.compose("new")', 'imx/compose.gif',
        alt="' + i18n['msg compose'], 24, 24, true
    ....
}
```

Modifying the Folders Window

Customizing Advanced Features

This chapter describes how to perform advanced customizations. It also provides HTML files and their location and details on features that are fully customizable for the Messenger Express user interface.

This chapter contains the following sections:

- Advanced Customization Overview
- Messenger Express User Interface Customizable Features
- Customization of Users' Default LDAP Attributes
- Domain Specific Customization

Advanced Customization Overview

In addition to the Messenger Express features discussed in Chapter 2, “Customizing General Features,” and Chapter 3, “Customizing User Interface Features,” many others are fully customizable. However, to take advantage of these features, an in-depth knowledge of JavaScript is required. In addition, migration problems might be encountered, for example, when attempting to reconfigure JavaScript files.

NOTE This chapter does not provide code samples for the advanced customizations. Also, an advanced knowledge of JavaScript and HTML is assumed.

Messenger Express User Interface Customizable Features

Table 4-1 shows the features of the Messenger Express user interface that are fully customizable.

Table 4-1 Messenger Express User Interface Customizable Features

Features	Files
Attachments	attach_fs.html
Collection of mail from another server	collect_fs.html
Message composition	comp_fs.html
Folder management tab	fldr_fs.html
Addresses search	ldap_fs.html
Mailbox management tab	mbox_fs.html
Message management tab	msg_fs.html
Personalize option management	opts_fs.html
Return receipt	receipt_fs.html
Subscribe Folders	subscribe_fs.html
Search Messages	searchmsg_fs.html
Emoticon	emoticons.html

Attachments

You can modify the following attachments options:

- Browse button
- Attach, Cancel, Help buttons

HTML File Mapping

The HTML file that controls the attachment related features is `attach_fs.html`.

Collect Mail from Another Server

You can modify the following when collecting mail from another server:

- POP server name (text field)
- POP user ID (text field)
- Password (text field)
- Delete messages from server (select button)
- Save to folder (list box)
- Collect, Cancel, and Help (button)

HTML File Mapping

The HTML file that controls the mail collection feature is `collect_fs.html`.

Message Composition

The message composition feature enables basic mail functions. You can modify the following message composition options:

- Compose new message
- Reply to the sender
- Reply to all recipients of the message including the sender
- Forward message to others
- Move message to folder
- Delete message
- Navigate through messages using Prev or Next icon.

HTML File Mapping

The HTML file that controls the message composition feature is `msg_fs.html`.

Folder Management Tab

The folder management tab enables access to server-side folders. You can modify the following folder management tab options:

- Update content of folder
- Create new folder
- Rename folder
- Share folder
- Delete existing folder

HTML File Mapping

The HTML file that controls the folder management tab feature is `fldr_fs.html`.

Address Search

The address search feature enables the management of address search in LDAP directories. You can modify the following address search options:

- Search for people in the selected search directory (list box)
- Insert full name (text field)
- Contain field (text field)
- Search, Close (buttons)
- To, Cc, Bcc (buttons)

HTML File Mapping

The HTML file that controls the address search feature is `lookup_fs.html`.

Mailbox Management Tab

The mailbox management tab enables access to a mailbox. You can modify the following mailbox management tab options:

- Get new message
- Compose new message

- Search for message
- Move message to selected folder
- Delete message
- Undelete message
- Select message
- Select all messages
- Collect external messages

HTML File Mapping

The HTML file that controls the mailbox management tab feature is `mbox_fs.html`.

Personal Option Management (Options Tab)

You can modify the following Options tab options:

- Account summary
- Personal information - change personal information
- Password - change and reset password
- Settings
- Appearance
- Vacation message - set vacation message
- Delegated administrator

HTML File Mapping

The HTML file that controls the options tab features is `opts_fs.html`.

Return Receipt

The return receipt feature enables the management of return receipts.

HTML File Mapping

The file that controls the return receipt feature is `receipt_fs.html`.

Customization of Users' Default LDAP Attributes

The Messenger Express server loads a default set of LDAP attributes for a user at the start of a session. These attributes are as follows:

```
cn
givenName
mail
mailAlternateAddress
mailAutoReplyMode
mailAutoReplySubject
mailAutoReplyText
mailAutoReplyTextInternal
mailAutoReplyTimeout
mailDeliveryOption
mailForwardingAddress
mailQuota,mailMsgQuota
preferredLanguage
sn
uid
vacationEndDate
vacationStartDate
```

You might want to obtain other customized LDAP attributes from the server. For example, an ISP might have a custom LDAP attribute assigned to all users called `myuserclass`. This attribute could denote different types of users that access services, including Messenger Express. Possible values for this attribute are “regular” and “vip”.

Depending on the type of user (that is, the value of the `myuserclass` LDAP attribute), different advertisement are presented to the user when they log into Messenger Express (Messenger Express is customized to display banner advertisement). If the customized client has access to the `myuserclass` LDAP attribute, the type of user can be determined and the relevant banner advertisement for that user type is displayed.

To obtain other customized LDAP attributes from the server, use `configutil` to modify the `service.http.extrauserldapattrs` configuration parameter. The attributes are read-only by default. If the customer wants to modify an attribute using the Messenger Express code, that attribute needs to be marked read-write by appending the suffix `w`.

The example below assumes the customer wants to display banner advertisements depending on the class of the user and that the client program allows the user to edit a link to a home page:

```
configutil -l -o service.http.extrauserldapattrs -v
"myuserclass,homepage:w "
```

Domain Specific Customization

This section describes how to customize the Messenger Express client interface for each domain.

You can perform the following tasks to customize the Messenger Express client interface:

- Create a directory with the domain name under `server-root/html` directory.
- Populate this directory with the customized versions of the files from the original directory hierarchy.

For example, assume that you have a domain called `siroe`. To change the icon for `siroe` domain, add a new icon in the `imx` directory of `siroe.com` and change the reference to it in `main.js` file:

```
html/... // default interface
html/imx/... // default interface
html/en/... // default interface
html/siroe.com/main.js // refers to imx/bottle.gif
html/siroe.com/imx/bottle.gif
```

- After login, the server refers the user agent to pick the file `main.html` which is located in the *domain/lang* directory. The `main.html` file contains relative references to the rest of the interface. The client requests all the files that make the interface in the directory. If these files exist, then they are displayed otherwise there is fallback to the default setup files.

If you have many domains and only a few distinct 'brands' then you can use links to make the server use the correct version:

```
html/... // default interface
html/sesta.com/... // customized interface for brand 1
html/varrius.com -> sesta.com // default interface
```

Domain From URL

The server listens to all IP addresses and can still present a customized interface before login occurs. It does this by looking at the URL and determines if it contains a known domain presenting a per domain login screen.

For example, for the per domain login screen: `http://webmail.sesta.com/`, the server sends the page from the location: `html/sesta.com/en/default.html`.

In this case a user does not have to append `@domain` to the username to login.

Managing Authentication to the Messenger Express Service

This chapter describes how to integrate alternative authentication mechanisms with iPlanet Messenger Express.

This chapter contains the following sections:

- Introduction to Authentication
- SDK Files and Functions
- SDK Configuration Initialization
- SDK Lookup
- SDK Cleanup
- Example Deployment

Introduction to Authentication

Some sites deploying Messenger Express might want to provide an alternative method of authenticating users. Most of these sites have some sort of portal service in which users sign on once at the “front door” to use various services without reauthentication. In this case, the Messenger Express’ login mechanism can recognize that this other service has already performed the authentication and recognizes the user based on credentials provided by the authenticator.

This is known as *proxy authentication*. In most cases, users go to a Web page, type in a user name and password, and then enter the site. One of the services users can access from the site would be the Messenger Express. When users click on the link to open Messenger Express, it is not necessary to ask for a username or password again, because with proxy authentication the user name and password provided at the initial site login is referenced.

The Messenger Express authentication SDK (Software Development Kit) provides the following three components, which can be modified to accept the proxy authentication:

- Initialization
- Lookup
- Cleanup

SDK Files and Functions

To integrate the authentication SDK into the existing code, begin by including the `expapi.h` header file into the calling code and link with the DLL/shared object. On some platforms, the authentication SDK may also require linking with other system libraries.

Table 5-1 shows the contents of `server-root/bin/msg/authsdk` (the install package).

Table 5-1 Contents of `authsdk`

Files	Function
<code>libexpress.so/DLL</code>	The SDK library
<code>cgiauth.c</code>	Source code for sample CGI using the API
<code>expapi.h</code>	Header file for API users
<code>login.html</code>	HTML source for the sample code
<code>nsapiauth.c</code>	Source code for the sample NSAPI plug-in using API
<code>README</code>	A readme documenting the API and use
<code>login.cgi</code>	Compiled CGI file
<code>Makefile.sample</code>	Example makefile to build <code>login.cgi</code>

SDK Configuration Initialization

The following function initializes the SDK configuration information needed when calling other functions:

```
int EXP_Init(
    char *pszLdapHost,
    char *pszLdapMatchAttrib,
    char *pszLdapDN,
    unsigned int iLdapPort,
    char *pszLdapBindUser,
    char *pszLdapBindPass,
    char *pszAdminUser,
    char *pszAdminPassword);
```

Table 5-2 SDK Code Variable Description

SDK Code Variables	Description
pszLdapHost	A null-terminated string containing the host name or IP address of the LDAP server in which to search for users.
pszLdapMatchAttrib	A null-terminated string specifying which LDAP attribute the pszAdminUser parameter should be matched against when searching LDAP. The default is user ID (uid).
pszLdapDN	A null-terminated string specifying the Domain Name (DN) to use when searching for users.
iLdapPort	An integer specifying the port number in which the LDAP server is listening.
pszLdapBindUser	The string specifies the bind DN and password for the directory server. If this is NULL, the SDK attempts to bind as anonymous user.
pszLdapBindUser	The string specifies the bind DN and password for the directory server. If this is NULL, the SDK attempts to bind as anonymous user.
pszAdminUser	The pointer to the string containing the “proxy” username and password is used when connecting to the messaging server. This is not NULL.
pszAdminPass	The pointer to the string containing the “proxy” username and password is used when connecting to the messaging server. This is not NULL.

Upon successful completion, a 0 is returned. If initialization fails, a non-zero number is returned. If failure occurs, `errno` is set to the most appropriate value possible based on what failed (in most cases a system call). These codes then map to standard `errno` values.

SDK Lookup

The following functions are used to generate a session handle for a specific user and client IP address:

```
int EXP_GenerateLoginURL(
    char *pszUser,
    char *pszClientAddress,
    char *pszMailHost,
    char *pszURL);
```

Table 5-3 SDK Code Variable Description

SDK Code Variables	Description
<code>pszUser</code>	A null-terminated string containing the user ID (uid).
<code>pszClientAddress</code>	The string representation of the client's IP address.
<code>pszMailHost</code>	A null-terminated string containing the hostname or IP address of the user's mail server. If the third parameter is NULL, the LDAP server (from <code>EXP_init()</code>) is searched to determine this host. Otherwise, the mail host specified is used.
<code>pszURL</code>	The buffer allocated by the caller function in which URL is returned. The URL can have a maximum of 2048 characters long (including terminating NULL).

The function returns 0 on successful initialization, or a non-zero number on failure of initialization. On failure, `errno` is once again set to the most appropriate value.

The string returned by these functions is a login URL to be used when connecting to Messenger Express. Authentication applications (such as a login CGI) should call these functions after successfully authenticating the user based on the local authentication criteria. A typical CGI would use the resulting string to launch a URL or set a cookie on the client through HTTP headers or JavaScript.

SDK Cleanup

The following function is called to shut down and clean up any resources used by the SDK:

```
int EXP_Shutdown()
```

Typically it is not necessary to call in a simple CGI, but plug-ins using the API might want to reclaim resources and continue running.

The function returns a 0 upon successful initialization or a non-zero number indicating failure, and `errno` is then set to the most appropriate value based upon what failed. These codes also map to standard `errno` values.

The following function returns a `const` pointer to a null-terminated string identifying the version number of the SDK being used:

```
const char*EXP_GetVersion()
```

The value itself should not be used in any dependant manner. In other words, do not code anything that expects this string to be in a certain format, contains a certain value, and so on. This string is only available to provide information to support about which version you are using.

If no version number information is available, the function returns NULL.

The following function can be used to tell the SDK to contact a non-standard port when connecting to the Messenger Express service to generate a session:

```
void EXP_SetHttpPort
(init iHttpPort)
```

By default the SDK contacts the standard HTTP port, 80. This function is not thread safe and sets a global value. If you want to use it in a thread environment, lock the call and call the function `EXP_GenerateLoginURL`.

Example Deployment

The following example explains how the Proxy Authentication can be used. Use this example as a reference point to decide how to use the SDK in your environment.

- To test the Proxy Authentication API, see the `cgiauth.c` file. You need to edit certain `#define` statements appearing at the beginning of the file to suit your configuration:

```
#define HTML_SOURCE_FILE "login.html"
#define BUFFER_SIZE 1024
#define MAIL_SERVER "mail.yourdomain.com"
#define DIRECTORY_SERVER "directory.yourdomain.com"
#define DN "o=yourdomain.com"
#define ADMINNAME "admin"
#define ADMINPASS "admin"
```

- Change the values for `MAIL_SERVER`, `DIRECTORY_SERVER`, `DN`, `ADMINNAME`, and `ADMINPASS` to reflect your configuration. For example, if the mail server is `mail.siroe.com`, the directory server is `ldap.siroe.com`, and the administration username and password are `sysadmin`, then those lines should appear as follows:

```
#define HTML_SOURCE_FILE "login.html"
#define BUFFER_SIZE 1024
#define MAIL_SERVER "mail.siroe.com"
#define DIRECTORY_SERVER "ldap.siroe.com"
#define DN "o=siroe.com"
#define ADMINNAME "sysadmin"
#define ADMINPASS "sysadmin"
```

- Compile the `cgiauth.c` file into an executable CGI file and configure an HTTP server to execute the newly compiled CGI file.

If everything is set up correctly, upon running the script within a web browser, the CGI script presents the user with a simple login screen based on the `login.html` file. If a user enters a valid username and password, the script launches a Messenger Express session without Messenger Express prompting the user to reauthenticate.

Index

A

- account summary 69
- addFrameHTML() function 58
- address search 68
- attach button 66
- attach_fs.html file 66
- attachments 66
- authentication
 - cleanup 77
 - example deployment 77
 - files and functions 74
 - initialization 75
 - introduction 73
 - lookup 76
 - Software Development Kit 74

B

- Bcc button 68
- browse button 66
- buildhash 52

C

- Cancel button 66
- Cc button 68
- CGI 74

- Collect button 67

- collect_fs.html file 66
- collecting mail from another server 67
- comp_fs.html file 66
- compFrameHTML() function 49
- compose("new") function 41
- composition window 18
- composition window functions 20
- customization
 - advanced features 65
 - location of customizable files 10
 - overview 9

D

- Delegated administrator 18
- delegated administrator 69
- delmsg() function 41
- displayFolders() function 38
- displayMbox() function 38

E

- emoticons 49
 - disable 51
- en/i18n.js file 41, 47, 54, 58

F

fldr_fs.html file 62, 66, 68

folder management tab 68

folders screen 16

folders screen functions 16

functions

addFrameHTML() 58

compFrameHTML() 49

compose("new") 41

delmsg() 41

displayFolders() 38

displayMbox() 38

getToolbar() 40, 49

help() 38

listFrameHTML() 42, 58

listFrameHTML(doc) 44

logout() 38

parent.exmsg() 41

parent.move() 41

parent.srch() 41

parent.undelmsg() 41

refreshColorSet() 27

refreshMbox() 41

searchHTML() 58

selectMsg() 38

selectOptions() 38

toggleFrameHTML() 60

toolFrame() 30, 32, 33, 38, 40, 46

G

getToolbar() function 40, 49

H

Help button 66

help() function 38

I

i18n.js file 10, 39

inbox screen 13

inbox screen functions 13

J

JavaScript 9, 65

L

lang/default.html file 24

lang/i18n.js file 38, 40, 42, 44, 46

LDAP directory 68

ldap_fs.html file 58, 66, 68

listFrameHTML() function 42

listFrameHTML(doc) function 44

listHTML() function 58

localization

location of locale-specific customizable files 12
specific locales 11

logout() function 38

M

mailbox management tab 68

main.js file 10, 27, 30, 33, 38, 39, 42, 44, 46

mbox_fs.html file 40, 41, 42, 54, 66, 69

message composition 67

message composition window 48

message display window 44

message list window 42

message screen 14

message screen functions 15

message tool bar 46

Messenger Express
components 10

- introduction to functions 12
- localization 11
- Software Development Kit 74
- user interface customizable features 66

msg_fs.html file 44, 46, 47, 66, 67

O

- options screen 17
- options screen functions 18
- options tab 69
- options window 59
- opts_fs.html file 60, 66, 69

P

- parent.exmsg() function 41
- parent.move() function 41
- parent.srch() function 41
- parent.undelmsg() function 41
- personal information 69
- personal option management 69
- POP server 67
- POP user ID 67
- portal service authentication 73

R

- receipt_fs.html file 66, 69
- refreshColorSet() function 27
- refreshMbox() function 41
- return receipt 69

S

- search 68

- searchHTML() function 58
- selectMsg() function 38
- selectOptions() function 38
- Software Development Kit 74

T

- To button 68
- toggleFrameHTML() function 60
- toolFrame() function 30, 32, 33, 38, 40, 46
- trash folder 41

U

- ui [] array definitions 27
- ui[] controls 28

V

- vacation message 69

