

Customization Guide

iPlanet Messenger Express

Release 5.0

806-4820-01
September 2000

Copyright © 2000 Sun Microsystems, Inc. Some preexisting portions Copyright © 2000 Netscape Communications Corporation. All rights reserved.

Sun, Sun Microsystems, the Sun logo, JavaScript, iPlanet, and the iPlanet log 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.

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 the Sun-Netscape Alliance 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 © 2000 Sun Microsystems, Inc. Pour certaines parties préexistantes, Copyright © 2000 Netscape Communication Corp. Tous droits réservés.

Sun, Sun Microsystems, the Sun logo, JavaScript, iPlanet, et the iPlanet logo sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et d'autre pays. Netscape et the Netscape N logo sont des marques déposées de Netscape Communications Corporation aux Etats-Unis et d'autre 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.

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 l'Alliance Sun-Netscape 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	7
Who Should Read This Book	7
What You Need to Know	8
How This Book Is Organized	8
Document Conventions	9
Monospaced Font	9
Bold Monospaced Font	9
Italicized Font	9
Square or Straight Brackets	9
Command-Line Prompts	9
Where to Find Related Information	10
Where to Find This Book Online	10
Chapter 1 Introduction to Messenger Express and Customization	11
Messenger Express Customization Overview	11
Messenger Express Components	12
Location of Customizable Files	12
Messenger Express Localization	13
Specific Locales	13
Location of Locale-Specific Customizable Files	14
Basic Interfaces and Associated Functions	14
Inbox Screen	14
Inbox Screen Functions	15
Message Screen	16
Message Screen Functions	17
Folders Screen	18
Folders Screen Functions	18
Options Screen	19
Options Screen Functions	20
Composition Window	21
Composition Window Functions	22

Chapter 2 Customizing General Features	23
Modifying the Login Screen	23
Modifications You Can Make to the Login Screen	24
To Modify the Login Screen	24
Example—Login Screen Modifications	25
Modifying Color Sets	27
Modifications You Can Make to the Color Sets	27
To Modify Color Sets in the User Interface	28
Example—Color Sets Modifications	29
Modifying the Corner Logo and Link	30
Modifications You Can Make to the Corner Logo and Link	30
To Modify the Corner Logo and Link	30
Example—Corner Logo and Link Modifications	30
Modifying the Title Text	31
Modifications You Can Make to the Title Text	31
To Modify the Title Text	31
Example—Title Text Modification	32
Modifying the Brand Image	32
Modifications You Can Make to the Brand Image	32
To Modify the Brand Image	33
Example—Brand Image Modification	33
Chapter 3 Customizing User Interface Features	35
Modifying the Main Function Tabs	36
Modifications You Can Make to the Main Function Tabs	36
To Modify the Main Function Tabs	36
Example—Main Function Tabs Modifications	37
Modifying the MailBox Tool Bar	38
Modifications You Can Make to the Mailbox Tool Bar	38
To Modify the Mailbox Tool Bar	38
Example—Mailbox Tool Bar Modifications	39
Modifying the Message List Window	40
Modifications You Can Make to the Message List Window	41
To Modify the Message List Window	41
Example—Message List Window Modifications	41
Modifying the Message Display Window	43
Modifications You Can Make to the Message Display Window	43
To Modify the Message Display Window	43
Example—Message Display Window Modifications	44
Modifying the Message Tool Bar	45
Modifications You Can Make to the Message Tool Bar	45
To Modify the Message Tool Bar	45
Example—Message Tool Bar Modifications	46

Modifying the Message Composition Window	47
Modifications You Can Make to the Message Composition Window	48
To Modify the Message Composition Window	48
Example—Message Composition Window Modifications	49
Modifying the Address (Directory Lookup) Window	51
Modifications You Can Make to the Address (Directory Lookup) Window	51
To Modify the Address (Directory Lookup) Window	52
Example—Address (Directory Lookup) Window Modifications	52
Modifying the Options Window	53
Modifications You Can Make to the Options Window	53
To Modify the Options Window	54
Example—Options Window Modifications	54
Modifying the Folders Window	55
Modifications You Can Make to the Folders Window	56
To Modify the Folders Window	56
Example—Folders Window Modifications	57
Chapter 4 Customizing Advanced Features	59
Advanced Customization Overview	59
Messenger Express User Interface Customizable Features	60
Attachments Options	60
HTML File Mapping	60
Collect Mail from Another Server Options	60
HTML File Mapping	61
Message Composition	61
HTML File Mapping	61
Folder Management Tab	61
HTML File Mapping	62
Address Search	62
HTML File Mapping	62
Mailbox Management Tab	62
HTML File Mapping	63
Personal Option Management (Options Tab)	63
HTML File Mapping	63
Return Receipt	63
HTML File Mapping	63
Chapter 5 Managing Authentication to the Messenger Express Service	65
Introduction to Authentication	65
SDK Files and Functions	66
SDK Configuration Initialization	67
SDK Lookup	68

SDK Cleanup	68
Example Deployment	69
Index	71

About This Guide

This manual explains how to customize the look and feel of iPlanet Messenger Express 5.0. Although the product architecture permits an almost unlimited customization of the “static” portion of the pages served by the Messenger Express HTTP daemon, this manual 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, and so on, 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 iPlanet Messaging Server 5.0 at your site. Developers may also find this guide useful for reference.

What You Need to Know

This book assumes you are knowledgeable in the iPlanet Messaging Server software and that you have 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 shows how to customize the general features of iPlanet Messenger Express.
- Chapter 3, “Customizing User Interface Features”
This chapter shows 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 5.0 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 find *Customization Guide* online in PDF and HTML formats. To find this book, use this URL:

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

Introduction to Messenger Express and Customization

iPlanet Messenger Express 5.0 is a Web-based electronic mail program that lets end users access their mailboxes using a browser running on an Internet-connected computer system using HTTP. Messenger Express clients send mail to a specialized web server that is part of iPlanet Message 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, but for the sake of simplicity, this manual discusses only the basic features. In addition, you can customize most features easily during an upgrade.

This chapter contains these sections:

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

Messenger Express Customization Overview

iPlanet Messenger Express 5.0 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 function. 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 Message Server software.

Table 1-1 lists the files that you edit 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 of the UI
<code>mailbox_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
<code>msg_fs.html</code>	Message management portion of the UI
<code>fldr_fs.html</code>	Folder management portion of the UI
<code>opts_fs.html</code>	Option management portion of the UI
<code>comp_fs.html</code>	Message composition
<code>lang/default.html</code>	Login screen
<code>ldap_fs.html</code>	Address search
<code>attach_fs.html</code>	Attachments
<code>collect_fs.html</code>	Collection of mail from another server
<code>receipt_fs.html</code>	Return receipt

Messenger Express Localization

You can localize any feature of iPlanet Messenger Express. You can create different pages for each language your users speak. 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 webmail pages from the appropriate subdirectory.

When you change common sections of the static webmail pages, you must make the changes multiple times 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
Japanese	ja

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

Locale	Abbreviation
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 you install the Message Server software.

Basic Interfaces and Associated Functions

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

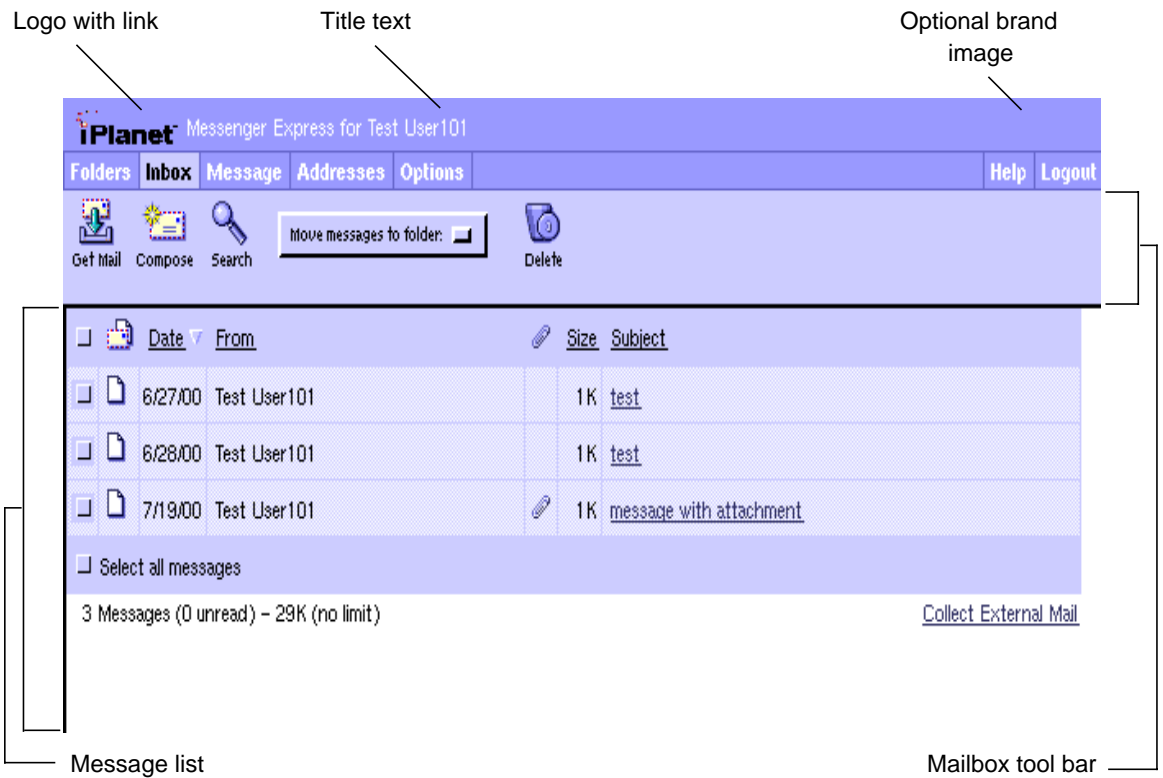
- Inbox screen
- Message screen
- Folders screen
- Options screen
- Composition window

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

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

Inbox Screen

The Messenger Express inbox screen, shown in Figure 1-1 on page 15, enables you to view all your messages, and specifies if they have been read. 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.displayMbox()</code> or <code>refreshMbox()</code>
Message	<code>main.selectMsg()</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.folderRefresh=true;main.refreshMbox()</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 on page 17, displays the message previously 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 direct access 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`).

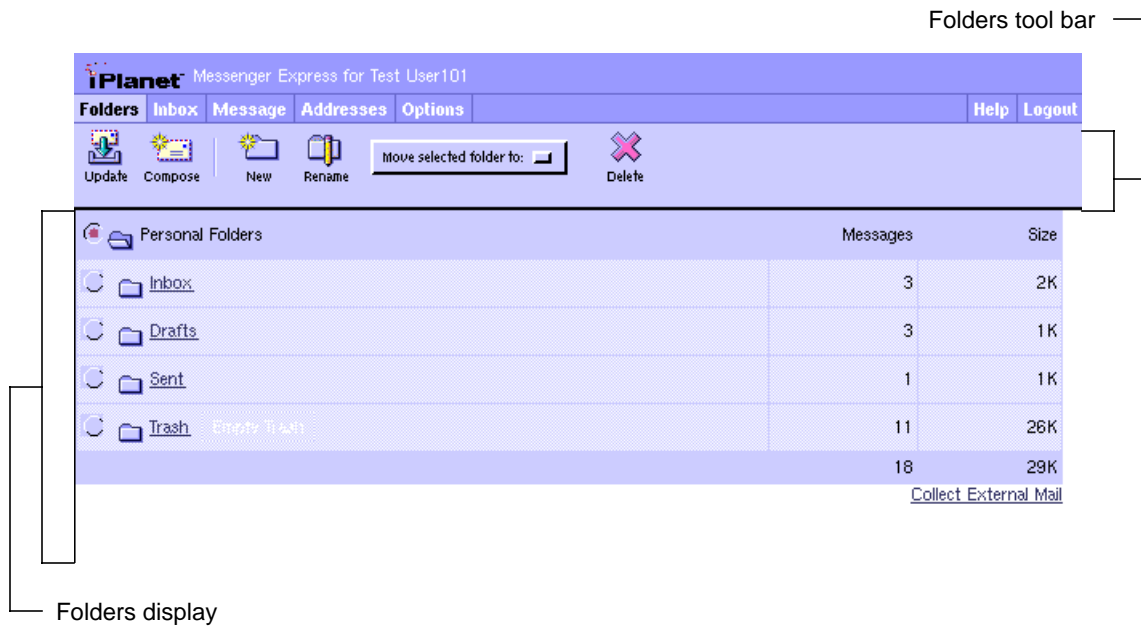
Table 1-4 Message Screen Functions

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>
Move Messages to Folder	<code>parent.move()</code>
Delete and Undelete	<code>parent.delmsg()</code>
Previous	<code>parent.prev()</code>
Next	<code>parent.next()</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, moving a folder within another one, updating the inbox, and composing new messages. Like the inbox screen, the folders screen also has the functionality of collecting external mail.

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>

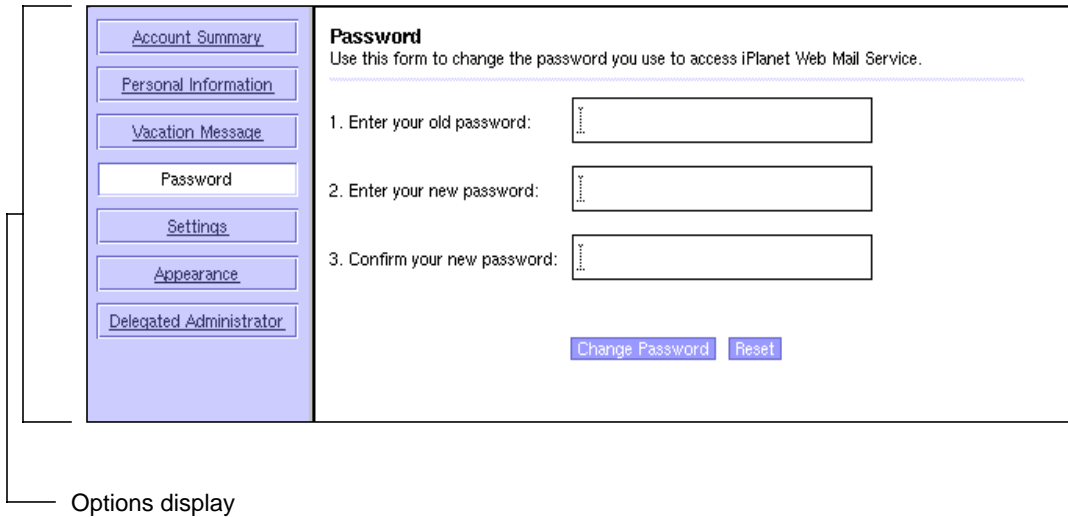
Table 1-5 Folders Screen Functions *(Continued)*

Item	Function
New	<code>parent.addFolder()</code>
Rename	<code>parent.renFolder()</code>
Move Folder	<code>parent.moveFolder(options[selectedIndex].value)</code>
Delete	<code>parent.delFolder()</code>
Select a Folder	<code>parent.select('i')</code>

Options Screen

The Messenger Express options screen, shown in Figure 1-4 on page 20, enables access to the subscriber's account summary, personal information, password, settings, appearance, delegated administrator, and vacation message, all of which are adjustable.

Figure 1-4 Messenger Express Options Screen



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 for composing 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. Recipients can be added in the form of “to,” “cc,” or “bcc,” and senders can be added as well. Furthermore, the composition window enables the option of requesting different mailing priority or a return receipt.

Figure 1-5 Messenger Express Composition Window

The screenshot shows the Messenger Express Composition Window interface. At the top is a light blue toolbar with six icons and labels: Send (envelope), Attach (paperclip), Save Draft (floppy disk), Address (address book), Help (question mark), and Cancel (red X). Below the toolbar are three main input areas: a 'Send To' field with a dropdown arrow and 'As To Cc Bcc' buttons to its right; a 'Subject' field with a dropdown arrow; and a large empty text area for the message body with a dropdown arrow on the left. At the bottom, there are two dropdown menus: 'Priority' set to 'Normal' and 'Request receipt' set to 'None'.

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>
Help	<code>main.help(1007399)</code>
Cancel	<code>parent.cancel()</code>
To/Cc/Bcc	<code>parent.add()</code>

Customizing General Features

This chapter describes how to customize iPlanet Messenger Express 5.0 general features.

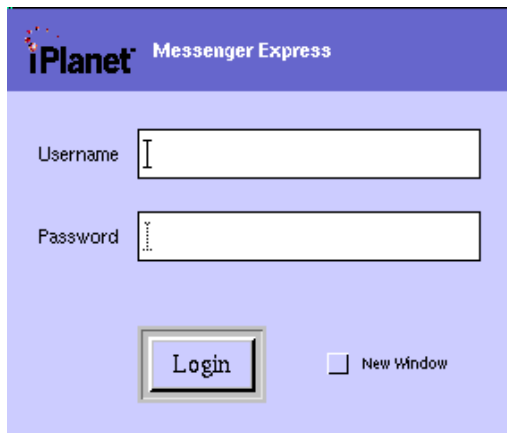
This chapter contains these sections:

- Modifying the Login Screen
- Modifying Color Sets
- Modifying the Corner Logo and Link
- Modifying the Title Text
- Modifying the Brand Image

Modifying the Login Screen

This section describes how to modify the Messenger Express login screen shown in Figure 2-1 on page 24.

Figure 2-1 Messenger Express Login Screen



Modifications You Can Make to the Login Screen

You can perform the following modifications on the Messenger Express login screen:

- Replace the logo and accompanying link with a custom graphic and link
- Change the color scheme
- Replace the service name (for example, Messenger Express)

To Modify the Login Screen

- Edit the `lang/default.html` file.

Customize the look by editing the body of `lang/default.html`. Functionally, `lang/default.html` contains four forms, two visible and two hidden:

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

- Login New Window Form (hidden)

The hidden forms are the only ones that are submitted to the server (POST username and password to `login.msc`).

Submitting the visible forms calls the function `post()`, which calls `login()`, which fills out the hidden forms based on the visible form values and submits the appropriate hidden form based on whether the New Window box is checked. Opening in a new window involves specifying a `TARGET` window name for the server response in the `FORM` tag.

Example—Login Screen Modifications

This example, shown in Figure 2-2, replaces the iPlanet logo with a custom graphic and link, changes the color scheme to maroon and silver, and changes the wording to “Web Mail Service.”

Figure 2-2 Example Login Screen Modifications

The image shows a web login interface. At the top, there is a maroon header bar containing a silver and red logo with the text "siroc" and "Web Mail Service" to its right. Below the header is a light gray area. On the left, the labels "Username" and "Password" are positioned next to their respective input fields. The "Username" field is a simple white box with a cursor. The "Password" field is a white box with a password mask (dots). Below these fields, there is a "Login" button with a 3D effect and a "New Window" checkbox with the label "New Window" to its right.

Code Example 2-1 on page 26 shows the login screen HTML before changes, and Code Example 2-2 on page 26 shows the changes. The file to edit is `en/default.html`.

Code Example 2-1 Before Altering Login Screen Features

```
<body bgcolor="white" link="#666699" vlink="#666699" alink="#CCCCFF">
....
<td bgcolor="#666699">
....
<td width="24"><a href="http://www.iplanet.com/" target="_top">
</a></td>
<td><font face="PrimaSans BT,Verdana,sans-serif" color="white"
size="-1"><font color="#CCCCCC" size="-2"><sup></sup></font><b>
    Messenger Express</b></font></td>
....
<td bgcolor="#CCCCCC">
....
</body>
```

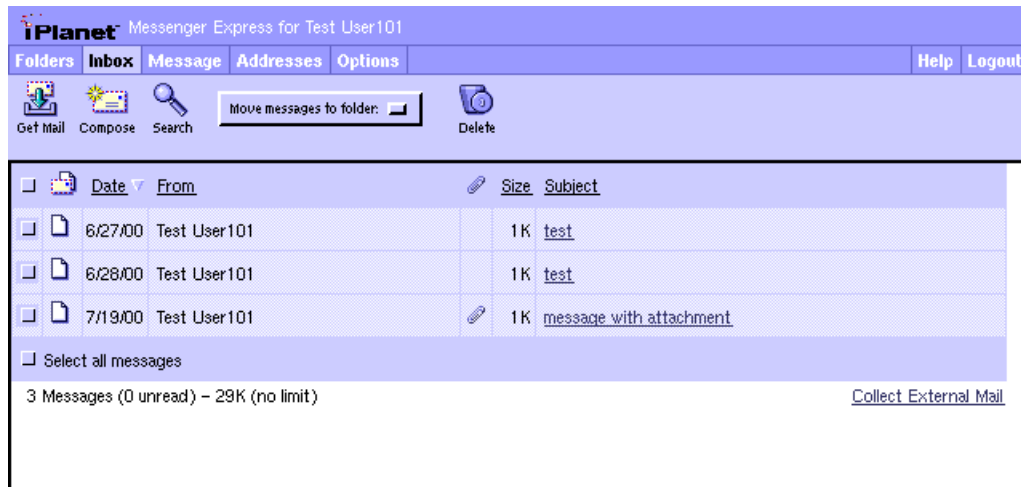
Code Example 2-2 After Altering Login Screen Features

```
<body bgcolor="white" link="#666699" vlink="#666699" alink="#CCCCFF">
.....
<td bgcolor="#800000">
.....
<td width="24"><a href="http://siroe.org" target="_top">
</a></td>
<td><font face="PrimaSans BT,Verdana,sans-serif" color="white"
size="-1"><font color="CCCCCC" size="-2"><sup></sup></font><b>
    Web Mail Service</b></font></td>
.....
<td bgcolor="#C0C0C0">
....
</body>
```

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



Modifications You Can Make to the Color Sets

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

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. This sets color values like `chrome1`, `accent2`, and so on, which 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`. If the rows are deleted from the definition script and the user preferences still point to a higher color table index than exists in the `ui[]` array, the user's JavaScript application will not start.

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

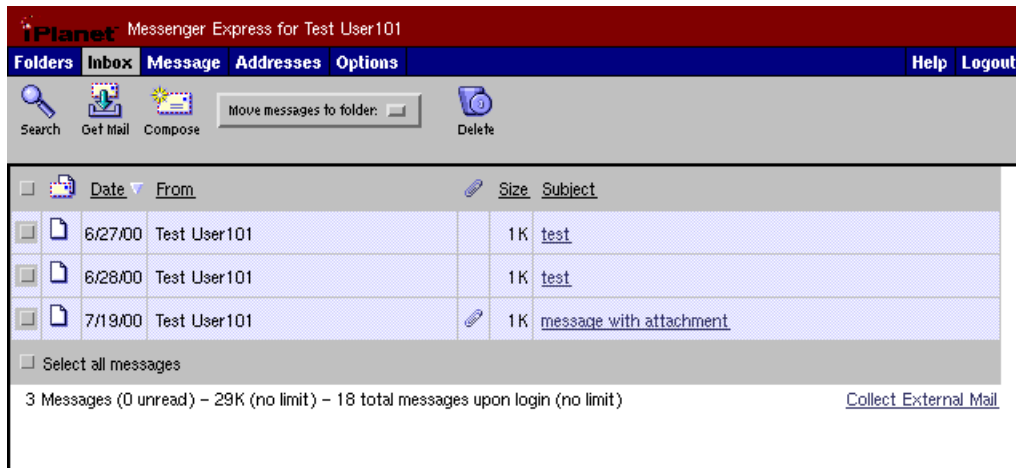
Table 2-1 Color Index of `ui[]` Controls

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

This example, shown in Figure 2-4, customizes the default color set to have `accent1` color maroon, `chrome4` color navy, and `chrome5` color silver.

Figure 2-4 Example Color Sets Modifications



Code Example 2-3 shows the necessary changes. The file to edit is `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 Corner 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



Modifications You Can Make to the Corner Logo and Link

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

- Replace logo with custom graphic
- Change destination of link

To Modify the Corner Logo and Link

- Edit the function `toolFrame()` in the `main.js` file.

Example—Corner Logo and Link Modifications

This example, shown in Figure 2-6, replaces the iPlanet logo with a custom logo having different dimensions and a custom link.

Figure 2-6 Example Corner Log and Link Modification



Code Example 2-4 shows the necessary changes. The file to edit is `main.js`.

Code Example 2-4 Altering Logo and Link

```
function toolFrame() {
    ....
    '<a href="http://www.siroe.org"' +
    ' target="Netscape"></a>' + '</td><td>' +
    ....
}
```

Modifying the Title Text

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

Figure 2-7 Messenger Express Title Text



Modifications You Can Make to the Title Text

You can exchange the Messenger Express title text with the service name.

To Modify the Title Text

- To customize the layout of the title text, edit the function `toolFrame()` in the `main.js` file. To customize the title text itself, edit the function `i18n_tab_header()` in the `lang/i18n.js` file.

In general, you can quickly custom brand the interface by replacing all the “Messenger Express” strings with your custom brand name in the `lang/i18n.js` file.

Example—Title Text Modification

This example, shown in Figure 2-8, customizes the text to say “Web Mail Service for *user*.”

Figure 2-8 Example Title Text Modification



Code Example 2-5 shows the necessary changes. The file to edit is `en/i18n.js`.

Code Example 2-5 Altering Title Text

```
function i18n_tab_header(user) {  
    return '<no>Web Mail Service</no> for ' + user  
}
```

Modifying the Brand Image

This section describes how to modify the brand image shown in Figure 2-9.

Figure 2-9 Messenger Express Brand Image



Blank spacer graphic

Modifications You Can Make to the Brand Image

You can perform the following modifications on the Messenger Express brand image:

- Include a custom brand logo of the server

- Alter the size of the brand image

To Modify the Brand Image

- Edit the `main.js` file as follows:
 - To customize the brand image, near the top of the `main.js` file, edit the values of `brandht` (height), `brandwd` (width), and `brandimg` (image location). The default is a blank 1x16 spacer graphic.
 - To customize how the image is displayed, near the top of the `main.js` file, edit the value of `brand` (the `IMG` tag generated from the other values).
 - To further customize the properties of the brand, edit the function `toolFrame()` in the `main.js` file.

Example—Brand Image Modification

This example, shown in Figure 2-10, customizes Messenger Express with a new brand image.

Figure 2-10 Example Brand Image Modification



Code Example 2-6 shows the necessary changes. The file to edit is `main.js`.

Code Example 2-6 Altering Brand Image

```
// set this to appropriate text and/or image to brand the top frame
// MUST specify height and width for Navigator 3.x to work
var brandht = 45
var brandwd = 45
var brandwd = 45
```


Customizing User Interface Features

This chapter describes how to customize iPlanet Messenger Express 5.0 user interface features.

This chapter contains these sections:

- Modifying the Main Function Tabs
- Modifying the MailBox Tool Bar
- Modifying the Message List Window
- Modifying the Message Display Window
- Modifying the Message Tool Bar
- Modifying the Message Composition Window
- Modifying the Address (Directory Lookup) Window
- Modifying the Options Window
- Modifying the Folders Window

Modifying the Main Function Tabs

This section describes how to modify the Messenger Express main function tabs shown in Figure 3-1 and gives examples of files edited to make the modifications.

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



Modifications You Can Make to the Main Function Tabs

You can perform the following modifications 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

- 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 `toolFrame()` with a call to the function `tab()` in the `main.js` file, specifying the text of the tab label to display, a flag for whether it is currently selected, and the function to call when the tab is clicked.

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

- `Folders: displayFolders()`

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

Example—Main Function Tabs Modifications

This example, shown in Figure 3-2, moves the Options tab to the right side, 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. The files to edit are `main.js` (layout) and `en/i18n.js` (tab labels).

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

```
function toolFrame() {
    .....
    '<td width=50%>' + 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['options'] = 'Preferences'
```

Modifying the MailBox Tool Bar

This section describes how to modify the Messenger Express mailbox tool bar shown in Figure 3-3 gives examples of files edited to make the modifications.

Figure 3-3 Messenger Express Mailbox Tool Bar



Modifications You Can Make to the Mailbox Tool Bar

You can perform the following modifications on the Messenger Express mailbox tool bar:

- Rearrange the order of tools
- Change text of tools

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 words 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:** `refreshMbox()` in `main.js`
- **Compose:** `compose("new")` in `main.js`
- **Search:** `parent.srch()`
- **Move Messages to Folder:** `parent.move()`
- **Delete:** `parent.delmsg()`, `parent.undelmsg()`, `parent.exmsg()` (depends whether in trash folder or not)

Example—Mailbox Tool Bar Modifications

This example, shown in Figure 3-4, makes Search 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 on page 40 and Code Example 3-4 on page 40 show the necessary changes. The files to edit are `mbox_fs.html` (layout) and `en/i18n.js` (wording).

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

```
function getToolbar() {
    ....
    main.toolbar(
        ....
        (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)
        ....
    }
}
```

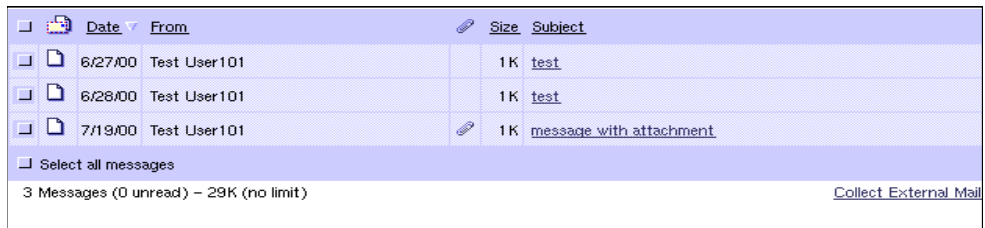
Code Example 3-4 Altering Tool Bar Wording (en/i18n.js)

```
// 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 gives examples of files edited to make the modifications.

Figure 3-5 Messenger Express Message List Window



Modifications You Can Make to the Message List Window

You can perform the following modifications to the Messenger Express message list window:

- Make messages sort differently by default
- Change text for collecting external mail

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 default column heading wording, edit the `i18n[]` values for search results, date, from, to, size, and subject in the `lang/i18n.js` file.
 - To customize the wording of the “Collect External Mail” link, edit `i18n['collect long]` in the `lang/i18n.js` file.
 - To customize the order in which messages are listed, edit the `defaults[]` values near the top of the `main.js` file.

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

Example—Message List Window Modifications

This example, shown in Figure 3-6 on page 42, makes messages by default list “last-in” first, and changes the text for collecting external mail.

Figure 3-6 Example Message List Window Modifications

<input type="checkbox"/>		Date ▲	From		Size	Subject
<input type="checkbox"/>		7/19/00	Test User101		1K	message with attachment
<input type="checkbox"/>		6/28/00	Test User101		1K	test
<input type="checkbox"/>		6/27/00	Test User101		1K	test
<input type="checkbox"/> Select all messages						
3 Messages (0 unread) – 29K (no limit) – 18 total messages upon login (no limit)						Get Messages From Another Server

Code Example 3-5 and Code Example 3-6 show the necessary changes. The files to edit are `main.js` and `en/i18n.js`.

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

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

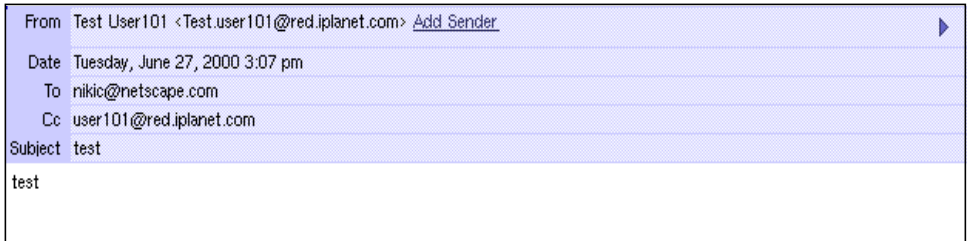
Code Example 3-6 Altering List Window Wording (`en/i18n.js`)

```
// 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 gives examples of files edited to make the modifications.

Figure 3-7 Messenger Express Message Display Window



Modifications You Can Make to the Message Display Window

You can perform the following modifications to the Messenger Express message display window:

- Change how messages are displayed
- Alter the layout of window and messages
- Change the wording

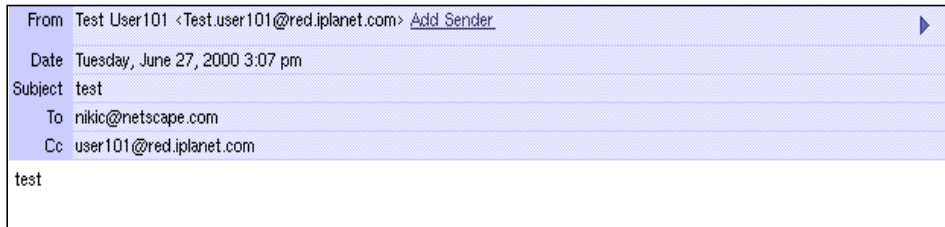
To Modify the Message Display Window

- 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 wording, edit the `i18n[]` values under `// Message Headers` and `// Message` in the `lang/i18n.js` file.
 - To customize the defaults of how the messages are displayed (wordwrap and so on), edit the `defaults[]` values near the top of the `main.js` file.

Example—Message Display Window Modifications

This example, shown in Figure 3-8, moves “Subject” before “To” rather than after “To.”

Figure 3-8 Example Message Display Window Modifications



Code Example 3-7 shows the necessary changes. The file to edit is `msg_fs.html`.

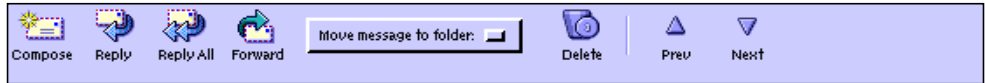
Code Example 3-7 Altering Message Display Window Layout

```
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 gives examples of files edited to make the modifications.

Figure 3-9 Messenger Express Message Tool Bar



Modifications You Can Make to the Message Tool Bar

You can perform the following modifications to 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 words associated with the graphics

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 words associated with the graphics in the Tool Bar, edit the `i18n[]` values `compose`, `reply`, `reply all`, `forward`, `file msg`, `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` that handle the tool clicks are:

- **Compose:** `compose(\`new\`)` in `main.js`
- **Reply:** `compose(\`reply\`)` in `main.js`
- **Reply All:** `compose(\`replyall\`)` in `main.js`
- **Forward:** `compose(\`forward\`)` in `main.js`
- **Move Messages to Folder:** `parent.move()`
- **Delete and Undelete:** `parent.delmsg()`
- **Previous:** `parent.prev()`
- **Next:** `parent.next()`

Example—Message Tool Bar Modifications

This example, shown in Figure 3-10, moves Compose divided to the right side and unabbreviates the text of “Prev” to “Previous.”

Figure 3-10 Example Message Tool Bar Modifications



Code Example 3-8 and Code Example 3-9 show the necessary changes. The files to edit are `msg_fs.html` and `en/i18n.js`.

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

```
function getToolbar() {
  ....
  main.toolbar(
    ....
    i18n['next'], 'parent.next()', n ? 'imx/next-1.gif' : 'imx/next-0.gif', 27,
    25, n,
    null, null, 'imx/divider.gif', 2, 24, false,
    i18n['compose'], 'main.compose("new")', 'imx/compose.gif', 27, 25, true)
    ....
  }
}
```

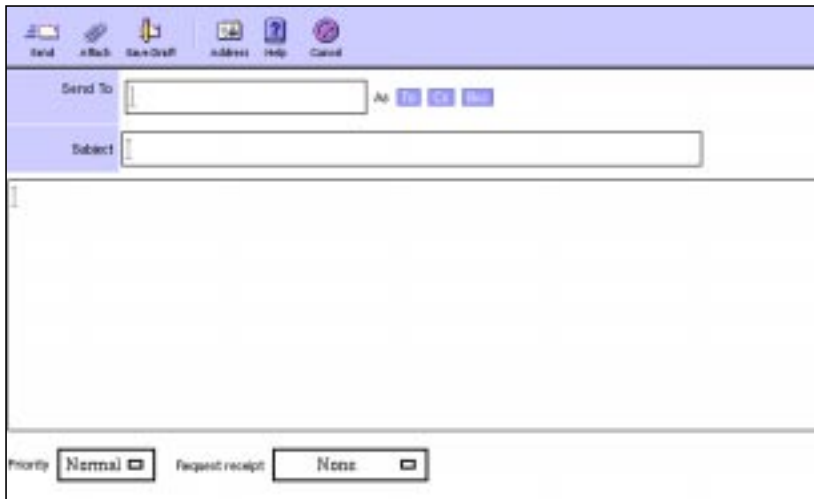
Code Example 3-9 Altering Message Tool Bar Text (`en/i18n.js`)

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

Modifying the Message Composition Window

This section describes how to modify the Messenger Express message composition window shown in Figure 3-11 on page 48 gives examples of files edited to make the modifications.

Figure 3-11 Messenger Express Message Composition Window



Modifications You Can Make to the Message Composition Window

You can perform the following modifications to the Messenger Express message composition window:

- Change location of tools in window
- Alter wording of tools

To Modify the Message Composition Window

- Edit the appropriate files as follows:
 - To customize the tool bar, edit the `getToolbar()` function in the `comp_fs.html` file.
 - To customize anything else in the window, edit the `compFrameHTML()` function in the `comp_fs.html` file.

- To customize any of the wording, edit the values under `// Message Composition` and `// Tool Bars` in the `lang/i18n.js` file.

Functionally, `getToolbar()` in `comp_fs.html` assembles the code and assigns the functions to the graphics by calling `toolbar()` in `main.js`, which takes care of things like colors and text-only versions. The `compFrameHTML()` function in `comp_fs.html` generates the "To/Cc/Bcc" control area by calling `i18n_compose_controls()` in `lang/i18n.js`.

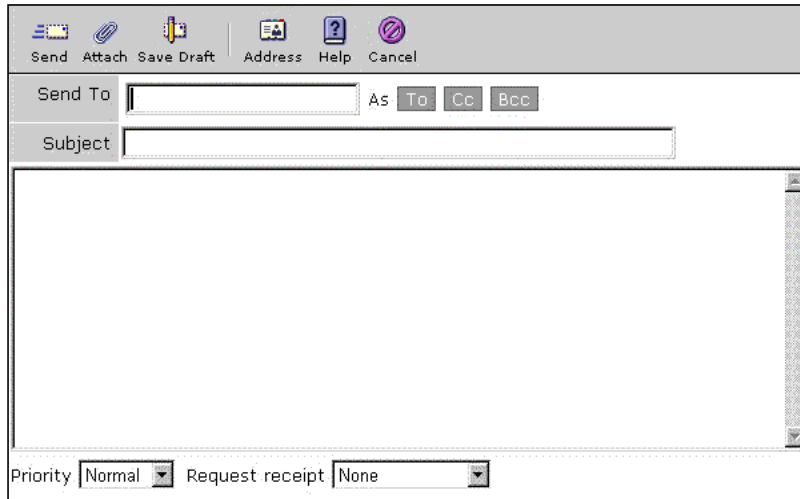
The functions assigned by `getToolbar()` and `compFrameHTML()` in `comp_fs.html` are:

- **Send:** `parent.send('\smtp')`
- **Address:** `parent.lookup()`
- **Attach:** `parent.attach()`
- **Save Draft:** `parent.send('\draft')`
- **Help:** `help(1007399)` in `main.js`.
- **Cancel:** `parent.cancel()`
- **To/Cc/Bcc:** `parent.add()`

Example—Message Composition Window Modifications

This example, shown in Figure 3-12 on page 50, moves the Address tool between divider and Help, and changes the "Add Recipient" wording to "Send To."

Figure 3-12 Example Message Composition Window Modifications



Code Example 3-10 and Code Example 3-11 on page 51 show the necessary changes. The files to edit are `comp_fs.html` and `en/i18n.js`.

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

```
function getToolbar() {
    ....
    i18n['send'], 'parent.send("smtp")', 'imx/send.gif', 27, 25, true)
    if ((!main.cfg['meAutoQuote'] || main.cfg['meAutoQuote'] == 'false') &&
        (compose_type != 'new' && compose_type != 'draft')) {
        s += main.toolbar(
            i18n['quote'], 'parent.quote(true)', 'imx/quote.gif', 29, 25, true)
    }
    s += main.toolbar(
        i18n['attach'], 'parent.attach()', 'imx/attach_comp.gif', 24, 24, true,
        i18n['draft'], 'parent.send("draft")', 'imx/draft.gif', 27, 25, true,
        null, null, 'imx/divider.gif', 2, 24, false,
        i18n['lookup'], 'parent.lookup()', 'imx/address.gif', 27, 25, true,
        i18n['help'], 'main.help(1007399)', 'imx/help.gif', 27, 25, true,
        ....
    }
}
```

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

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

Modifying the Address (Directory Lookup) Window

This section describes how to modify the Messenger Express address (directory lookup) window shown in Figure 3-13 gives examples of files edited to make the modifications.

Figure 3-13 Messenger Express Address (Directory Lookup) Window



Modifications You Can Make to the Address (Directory Lookup) Window

You can perform the following modifications to the Messenger Express address (directory lookup) window:

- Change overall window appearance
- Alter search controls and their wording
- Alter all other wording

To Modify the Address (Directory Lookup) Window

- Edit the appropriate files as follows:
 - To customize overall window appearance, edit the `searchFrameHTML()`, `listFrameHTML()`, and `addFrameHTML()` functions in the `ldap_fs.html` file.
 - To customize the search controls and their wording, edit `i18n_ldap_controls()` in the `lang/i18n.js` file.
 - To customize other wording, edit the `// LDAP Lookup` values in the `lang/i18n.js` file.

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

- Search: `parent.doSearch()`
- Cancel: `parent.cancel()`
- To/Cc/Bcc: `parent.add()`

Example—Address (Directory Lookup) Window Modifications

This example, shown in Figure 3-14, changes “Search the local directory” to “Search the iPlanet directory.”

Figure 3-14 Example Address (Directory Lookup) Window Modifications



Code Example 3-12 shows the necessary changes. The file to edit is `en/i18n.js`.

Code Example 3-12 Altering Address Window Text

```
function i18n_ldap_controls() {
    ....
    '<select name="dir">\n' +
    '<opti'<select name="dir">\n' + on value="3 200">Search the iPlanet
    directory</option>\n' +
    ....
}
```

Modifying the Options Window

This section describes how to modify the Messenger Express options window shown in Figure 3-15 gives examples of files edited to make the modifications.

Figure 3-15 Messenger Express Options Window

Modifications You Can Make to the Options Window

You can perform the following modifications to the Messenger Express options window:

- Change the layout of window and choices
- Alter any of the default wording

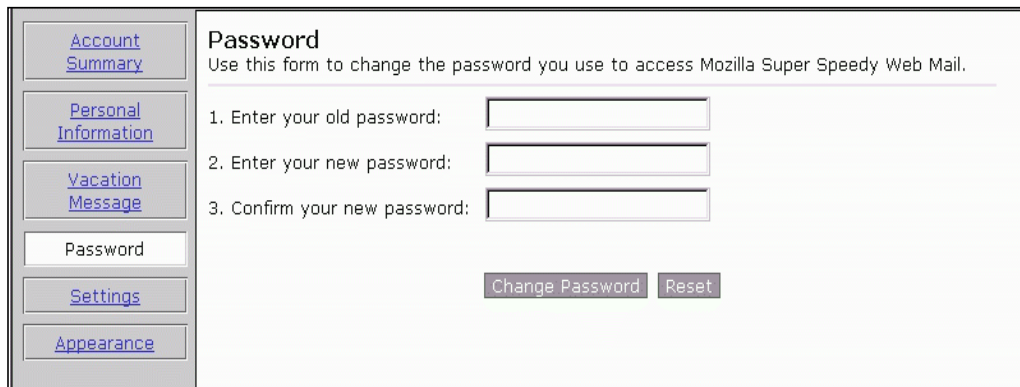
To Modify the Options Window

- Edit the appropriate files as follows:
 - To customize the choices and choice layout, edit the `toggleFrameHTML()` function in the `opts_fs.html` file.
 - To customize any of the default wording, edit the `i18n[]` values under `// Options` in the `lang/i18n.js` file.

Example—Options Window Modifications

This example, shown in Figure 3-16, moves “Vacation Message” between “Personal Information” and “Password,” and changes the Password form to “Mozilla Super Speedy Web Mail” as the text.

Figure 3-16 Example Options Window Modifications



The screenshot shows a web interface with a left sidebar and a main content area. The sidebar contains several menu items: "Account Summary", "Personal Information", "Vacation Message", "Password", "Settings", and "Appearance". The "Vacation Message" item is highlighted. The main content area is titled "Password" and contains the text "Use this form to change the password you use to access Mozilla Super Speedy Web Mail." Below this text 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 of the form are two buttons: "Change Password" and "Reset".

Code Example 3-13 on page 55 and Code Example 3-14 on page 55 show the necessary changes. The files to edit are `opts_fs.html` and `en/i18n.js`.

Code Example 3-13 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")') +
    getToggle(main.i18n['password'], 'password',
    'javascript:parent.toggle("password")') +
    ....
}
```

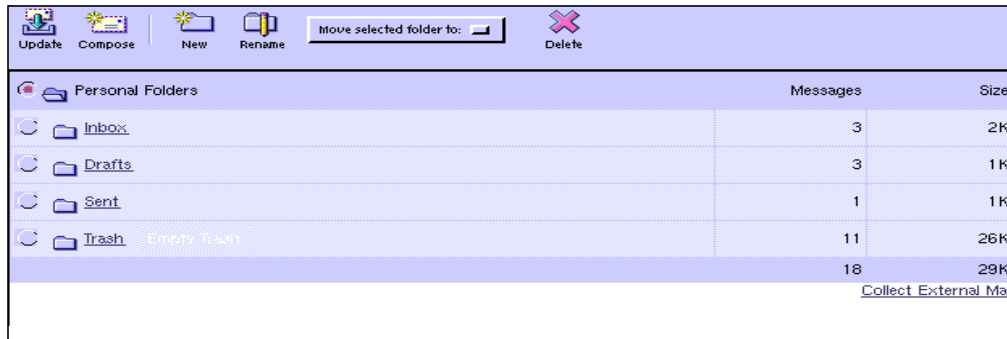
Code Example 3-14 Altering Options Window Text (en/i18n.js)

```
// 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 3-17 on page 56 gives examples of files edited to make the modifications.

Figure 3-17 Messenger Express Folders Window



Modifications You Can Make to the Folders Window

You can perform the following modifications to the Messenger Express folders window:

- Change the locations of the tools
- Alter the wording of the tools or folders

To Modify the Folders Window

- Edit the appropriate files as follows:
 - To customize the folders toolbar, edit the `getToolbar()` function in the `fldr_fs.html` file.
 - To customize the rest of the window, edit the `listFrameHTML()` function in the `fldr_fs.html` file.
 - To customize the wording of the “Collect External Mail” hyperlink, edit `i18n[‘collect long]` in the `lang/i18n.js` file.
 - To customize any other wording, edit the `// Folders` section in `lang/i18n.js`.

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

- **Update:** `refreshFolders()` in `main.js`
- **Compose:** `compose('\new')` in `main.js`
- **New:** `parent.addFolder()`
- **Rename:** `parent.renFolder()`
- **Move Folder:** `parent.moveFolder(options[selectedIndex].value)`
- **Delete:** `parent.delFolder()`
- **Select a Folder:** `parent.select('i')`

Example—Folders Window Modifications

This example, shown in Figure 3-18 on page 57, moves “Update” and “Compose” tools divided off to the end of the toolbar.

Figure 3-18 Example Folders Window Modifications



Code Example 3-15 on page 58 shows the necessary changes. The file to edit is `fldr_fs.html`.

Code Example 3-15 Altering Folders Window Layout

```
function getToolbar() {
    ....
    main.toolbar(
        i18n["new folder"], 'parent.addFolder()', 'imx/fldr_new.gif', 27, 25, true,
        i18n['rename'], 'parent.renFolder()', 'imx/fldr_edit.gif', 27, 25, true) +
    ....
    main.toolbar(
        i18n['delete'], 'parent.delFolder()', 'imx/delete.gif', 27, 25, true,
        null, null, 'imx/divider.gif', 2, 24, false,
        i18n['update'] ? i18n['update'] : i18n['get mail'],
    'main.refreshFolders()', 'imx/pull.gif', 27, 25, true,
        i18n['compose'], 'main.compose("new")', 'imx/compose.gif', 27, 25, true)
    ....
}
```

Customizing Advanced Features

To enable you to perform advanced customizations, this chapter provides the HTML files and interface points, which are fully customizable, for the Messenger Express 5.0 user interface.

This chapter contains these sections:

- Advanced Customization Overview
- Messenger Express User Interface Customizable Features

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, a much more substantial knowledge of JavaScript is necessary. 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	<code>attach_fs.html</code>
Collection of mail from another server	<code>collect_fs.html</code>
Message composition	<code>comp_fs.html</code>
Folder management tab	<code>fldr_fs.html</code>
Address search	<code>ldap_fs.html</code>
Mailbox management tab	<code>mbox_fs.html</code>
Message management tab	<code>msg_fs.html</code>
Personalize option management	<code>opts_fs.html</code>
Return receipt	<code>receipt_fs.html</code>

Attachments Options

You can modify the following attachments options:

- Browse button
- Attach, cancel, help (buttons)

HTML File Mapping

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

Collect Mail from Another Server Options

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 collect-mail-from-another-server 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 (Prev/Next)

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
- Move mail to selected 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 `ldap_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
- Expunge 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`.

Managing Authentication to the Messenger Express Service

This chapter describes how to integrate alternative authentication mechanisms with iPlanet Messenger Express 5.0.

This chapter contains these 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 5.0 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” and then are able to use various services without reauthenticating. In this case, Messenger Express 5.0’s login mechanism can recognize that this other service has already performed the authentication and recognize the user based on credentials provided by the authenticator.

In basic terms this is referred to as *proxy authentication*. In most cases, users will go to a Web page, type in a user name and password, and then enter the site. One of the services users are able to access from the site would be 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 this service the user name and password the users provided at the initial site login would be referred to instead.

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

- Initialization
- Lookup
- Cleanup

SDK Files and Functions

To integrate the authentication SDK into 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
README	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);
```

The `pszLdapHost` is a null terminated string containing the host name or IP address of the LDAP server in which to search for users.

The `pszLdapMatchAttrib` is a null terminated string specifying which LDAP attribute the `pszAdminUser` parameter should be matched against when searching LDAP. The default is user ID (uid).

The `pszLdapDN` is another null terminated string specifying the DN to use when searching for users.

The `iLdapPort` is an integer specifying the port number in which the LDAP server is listening.

The `pszLdapBindUser` and `pszLdapBindPass` are strings specifying the bind DN and password for the directory server. If NULL, the SDK will attempt to bind as an anonymous user.

The `pszAdminUser` and `pszAdminPass` are pointers to strings containing the “proxy” user name and password to use when connecting to the messaging server. These cannot be NULL.

Upon completion of the initialization, either a 0 will be returned if the initialization was successful or a non-zero number if the initialization failed. 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)
```

The `pszUser` is a null terminated string containing the user ID (uid).

The `pszClientAddress` is the string representation of the client's IP address.

The `pszMailHost` is 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 `EXP_init()`) will be searched to determine this host. Otherwise, the mail host specified will be used.

The fourth parameter, found in `pszURL`, is a pointer to a buffer which must be allocated by the caller for the function to return the URL in. The URL will be, at most, 2048 characters long (including terminating NULL).

The function returns a 0 upon success or a non-zero number upon failure. On failure, `errno` is once again set to the most appropriate value possible.

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 via 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's 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, don't 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 will return `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(
    int iHttpPort)
```

By default the SDK will contact 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, you'll need to lock around this call and the `EXP_GenerateLoginURL` call.

Example Deployment

To test the Proxy Authentication API, look at the `cgiauth.c` file. Near the beginning of the file, there are some `#define` statements that have to be edited in order to work with 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.mcom.com`, the directory server is `ldap.mcom.com`, and the administration username and password are `sysadmin`, then those lines should look like the following:

Example Deployment

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

From here, compile the `cgiauth.c` file into an executable CGI file. Next, 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 will present the user with a simple login screen based on the `login.html` file. If a valid user name and password are entered into the form, the script launches a Messenger Express session without Messenger Express prompting the user to reauthenticate.

This is only one example of how proxy authentication could be used. Use this example as a reference point in deciding how to use the SDK in your environment.

Index

A

- account summary, 63
- addFrameHTML() function, 52
- address (directory lookup) window, 51
- address search, 62
- attach button, 60
- attach_fs.html file, 60
- attachments, 60
- authentication
 - cleanup, 68
 - example deployment, 69
 - files and functions, 66
 - initialization, 67
 - introduction, 65
 - lookup, 68
 - Software Development Kit, 66

B

- bcc button, 62
- brand image, 32
- browse button, 60

C

- cancel button, 60
- cc button, 62

CGI, 66

- collect button, 61
- collect_fs.html file, 60
- collecting mail from another server, 60
- color sets, 27
- comp_fs.html file, 48, 60
- compFrameHTML() function, 49
- compose("new") function, 39
- composition window, 21
- composition window functions, 22
- corner logo and link, 30
- customization
 - advanced features, 59
 - location of customizable files, 12
 - overview, 11

D

- delegated administrator, 20, 63
- delmsg() function, 39
- displayFolders() function, 36
- displayMbox() function, 37

E

- en/i18n.js file, 39, 47, 53

F

`fldr_fs.html` file, 56, 60, 62

folder management tab, 61

folders screen, 18

folders screen functions, 18

folders window, 56

functions

`addFrameHTML()`, 52

`compFrameHTML()`, 49

`compose("new")`, 39

`delmsg()`, 39

`displayFolders()`, 36

`displayMbox()`, 37

`getToolbar()`, 38, 49

`help()`, 37

`i18n_ldap_controls()`, 52

`i18n_tab_header()`, 31

`listFrameHTML()`, 41, 52

`listFrameHTML(doc)`, 43

`logout()`, 37

`parent.exmsg()`, 39

`parent.move()`, 39

`parent.srch()`, 39

`parent.undelmsg()`, 39

`refreshColorSet()`, 28

`refreshMbox()`, 37, 39

`searchHTML()`, 52

`selectMsg()`, 37

`selectOptions()`, 37

`toggleFrameHTML()`, 54

`toolFrame()`, 30, 31, 33, 36, 38, 45

G

`getToolbar()` function, 38, 49

H

help button, 60

`help()` function, 37

I

`i18n.js` file, 12, 37

`i18n_ldap_controls()` function, 52

`i18n_tab_header()` function, 31

inbox screen, 14

inbox screen functions, 15

J

JavaScript, 11, 59

L

`lang/default.html` file, 24

`lang/i18n.js` file, 31, 36, 39, 41, 43, 46, 49, 52

LDAP directory, 62

`ldap_fs.html` file, 52, 60, 62

`listFrameHTML()` function, 41

`listFrameHTML(doc)` function, 43

`listHTML()` function, 52

localization

location of locale-specific customizable files, 14

specific locales, 13

login screen, 24

`logout()` function, 37

M

mailbox management tab, 62

mailbox tool bar, 38

main function tabs, 36

`main.js` file, 12, 28, 30, 33, 36, 37, 41, 43, 45

`mbox_fs.html` file, 38, 39, 41, 60, 63

message composition, 61

message composition window, 47

message composition window options, 48

message display window, 43

message list window, 40, 41
message screen, 16
message screen functions, 17
message tool bar, 45
message tool bar options, 45
Messenger Express
 components, 12
 introduction to functions, 14
 localization, 13
 Software Development Kit, 66
 user interface customizable features, 60
msg_fs.html file, 43, 45, 46, 47, 60, 61

O

options screen, 19
options screen functions, 20
options tab, 63
options window, 53
opts_fs.html file, 54, 60, 63

P

parent.exmsg() function, 39
parent.move() function, 39
parent.srch() function, 39
parent.undelmsg() function, 39
personal information, 63
personal option management, 63
POP server, 60
POP user ID, 61
portal service authentication, 65
proxy authentication, 66

R

receipt_fs.html file, 60, 63

refreshColorSet() function, 28
refreshMbox() function, 37, 39
return receipt, 63

S

search, 62
searchHTML() function, 52
selectMsg() function, 37
selectOptions() function, 37
Software Development Kit, 66

T

title text, 31
to button, 62
toggleFrameHTML() function, 54
toolFrame() function, 30, 31, 33, 36, 38, 45
trash folder, 39

U

ui [] array definitions, 28
ui[] controls, 28

V

vacation message, 63

