



Sun Java™ System

Communications Express 6 Customization Guide

2005Q1

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No:819-0116-10

Copyright © 2005 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

THIS PRODUCT CONTAINS CONFIDENTIAL INFORMATION AND TRADE SECRETS OF SUN MICROSYSTEMS, INC. USE, DISCLOSURE OR REPRODUCTION IS PROHIBITED WITHOUT THE PRIOR EXPRESS WRITTEN PERMISSION OF SUN MICROSYSTEMS, INC.

U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

This distribution may include materials developed by third parties.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Java, Solaris, JDK, Java Naming and Directory Interface, JavaMail, JavaHelp, J2SE, iPlanet, the Duke logo, the Java Coffee Cup logo, the Solaris logo, the SunTone Certified logo and the Sun ONE logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon architecture developed by Sun Microsystems, Inc.

Legato and the Legato logo are registered trademarks, and Legato NetWorker, are trademarks or registered trademarks of Legato Systems, Inc. The Netscape Communications Corp logo is a trademark or registered trademark of Netscape Communications Corporation.

The OPEN LOOK and Sun(TM) Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

Products covered by and information contained in this service manual are controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

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 © 2005 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés.

Sun Microsystems, Inc. détient les droits de propriété intellectuels relatifs à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et ce sans limitation, ces droits de propriété intellectuelle peuvent inclure un ou plus des brevets américains listés à l'adresse <http://www.sun.com/patents> et un ou les brevets supplémentaires ou les applications de brevet en attente aux Etats - Unis et dans les autres pays.

CE PRODUIT CONTIENT DES INFORMATIONS CONFIDENTIELLES ET DES SECRETS COMMERCIAUX DE SUN MICROSYSTEMS, INC. SON UTILISATION, SA DIVULGATION ET SA REPRODUCTION SONT INTERDITES SANS L'AUTORISATION EXPRESSE, ECRITE ET PREALABLE DE SUN MICROSYSTEMS, INC.

Cette distribution peut comprendre des composants développés par des tierces parties.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Java, Solaris, JDK, Java Naming and Directory Interface, JavaMail, JavaHelp, J2SE, iPlanet, le logo Duke, le logo Java Coffee Cup, le logo Solaris, le logo SunTone Certified et le logo Sun[tm] ONE sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

Legato, le logo Legato, et Legato NetWorker sont des marques de fabrique ou des marques déposées de Legato Systems, Inc. Le logo Netscape Communications Corp est une marque de fabrique ou une marque déposée de Netscape Communications Corporation.

L'interface d'utilisation graphique OPEN LOOK et Sun(TM) a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui, en outre, se conforment aux licences écrites de Sun.

Les produits qui font l'objet de ce manuel d'entretien et les informations qu'il contient sont régis par la législation américaine en matière de contrôle des exportations et peuvent être soumis au droit d'autres pays dans le domaine des exportations et importations. Les utilisations finales, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes biologiques et chimiques ou du nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers des pays sous embargo des Etats-Unis, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exclusive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la législation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement désignés, sont rigoureusement interdites.

LA DOCUMENTATION EST FOURNIE "EN L'ETAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.

Contents

List of Figures	5
List of Tables	7
Preface	9
Who Should Read This Book	9
What You Need to Know	10
How This Book is Organized	10
Monospaced Font	11
Bold Monospaced Font	11
Italicized Font	11
Where to Find Related Information	12
Where to Find This Book Online	12
Chapter 1 Introduction to Communications Express and Customization	13
Chapter 2 Customizing General Features in Communications Express	15
Guidelines to be Followed Before the Customization Process	15
List of Customizable and Configurable Files for Calendar and Address Book	16
Customizing Images	17
Customizing Text	18
Customizing the Skin	19
Creating a new skin	19
Modifying a skin	19
Customizing the Login Screen for the English Locale	20
Replacing the Logo With a Custom Graphic	20
Example—Logo Modification	20

Changing the Title Text	21
Customizing Banner	21
Customizing Application Bar	22
Chapter 3 Customizing General Features in Calendar	27
Customizing Icons	27
Customizing the Order of Calendar Toolbar Icons	28
Customizing the Toolbar Images	35
Customizing Icons	35
Customizing Events/Task Related Informative Icons	36
Customizing the Error Icon for Full Page Errors	37
Customizing Icons in Pop ups	37
Customizing Search for Events	37
Customizing the Text That Appears Besides the Search Events Text Box	38
Customizing Views	38
Customizing Tasks Search View	39
Customizing the Column Order in Tasks Printable View	39
Customizing Tasks Error View	43
Customizing Events view	43
Customizing Tasks View	44
Customizing Pop ups	44
Customizing Full Page Error	44
Customizing the Layout for Full Page Errors	45
Chapter 4 Customizing General Features in Mail	47
Defining Your Customization Requirements	47
Customizing the Skin to Modify Global Look and Feel	47
Customizing Text	48
Customizing Mail by Domain	48
Customizing Mail Icons	48
Adding Extra Features	48
Understanding the File Layout in Mail	49
Understanding Basic Interfaces in Mail	50
Inbox Screen	50
Message Screen	51
Folders Screen	51
Composition Window	52
Message Search Window	53
Chapter 5 Customizing User Interface Features in Mail	55
Customizing the Mailbox Tool Bar	55
Example—Mailbox Tool Bar Modifications	56

Customizing Message Listing	57
Customizing the Message Display Window	58
Modify the Message Display Window to Display User Defined Header Fields	59
Example—Customizing the Message Display Window to Display User Defined Header Fields	59
Customizing the Message Tool Bar	60
Customizing the Message Composition Window	60
Aligning the User Interface from Right to Left(For Arabic only)	63
Chapter 6 Customizing Advanced Features in Mail	65
Understanding Advanced Customization	65
List of files to use for customizing UI in Mail	66
Attachments	66
HTML File Mapping	66
Collect Mail from Another Server	66
HTML File Mapping	67
Message Composition	67
HTML File Mapping	67
Mailbox Management Tab	67
HTML File Mapping	68
HTML File Mapping	68
Return Receipt	68
HTML File Mapping	68
Understanding Shared Folders	68
Customizing the Default LDAP Attributes for Users	69
Chapter 7 Customizing General Features in Address Book	71
Customizing Address Book Icons and Labels	72
Customizing Icons	72
Customizing Labels	73
Customizing Style Sheets	74
What can be Customized in Style Sheets	75
Customizing the Main Search Page	76
Customizing New/Edit Contact Window	77
Customizing Phone	79
Customizing E-mail	82
Customizing Dates	82
Customizing Web Addresses	84
Customizing IM Nicknames	84
Customizing View Contact Window	86
Customizing New Group or Edit Group Window	86
Customizing Group Details	87

Customizing the View Group Window	88
Customizing Printable Window	88
Customizing the Import/Export Address Book Window	88
Chapter 8 Customizing Communications Express for a Specific Domain	91
Customizing Default User Preferences for a Specific Domain	91
Customizing Theme for a Specific Domain and Locale	92
Customizing Mail for a Specific Domain	93
Domain From URL	94
Customizing Address Book for a Specific Domain and a Locale	94
Chapter 9 Localizing Communications Express	97
Communications Express Localization	97
Localizing Calendar and Options	97
Localizing Date/Time Formats for Calendar	98
Localizing for languages that use characters outside the ASCII range	99
Localizing Mail	99
To Add an Additional Locale	101
Localizing Address Book	102
Customizing Address Book for a Specific Locale	103
Glossary	105
Index	107

List of Figures

Figure 2-1	Communications Express Login Screen	20
Figure 2-2	Communications Express Corner Logo and Link	20
Figure 2-3	Example Corner Logo	21
Figure 2-4	Login Screen with Customized Logo	21
Figure 2-5	Communications Express Title Text	21
Figure 2-6	Left Banner Image Before Customization	21
Figure 2-7	Customized Left Banner Image	22
Figure 2-8	Right Banner Image Before Customization	22
Figure 2-9	Customized Right Banner Image	22
Figure 2-10	Default application tab order	22
Figure 2-11	Changing the application tab order to display Address Book, Mail, Calendar and Options	24
Figure 3-1	Order of Calendar Toolbar Icons	28
Figure 3-2	Changed order of the Toolbar Icons	31
Figure 3-3	Customizing Search for Events	37
Figure 4-1	Communications Express Inbox Screen	51
Figure 4-2	Communications Express Message Screen	51
Figure 4-3	Folders Screen	52
Figure 4-4	Communications Express Composition Window	53
Figure 4-5	Message Search Window	53
Figure 5-1	Communications Express Mailbox Tool Bar	55
Figure 5-2	Communications Express Message List Window	57
Figure 5-3	Communications Express Message Display Window	58
Figure 5-4	Communications Express Message Tool Bar	60
Figure 5-5	Communications Express Message Composition Window	61

Figure 7-1	Customizing Phone	79
Figure 7-2	Customizing E-mail	82
Figure 7-3	Customizing the Import/Export Address Book Window	88

List of Tables

Table 0-1	Organization of the Sun Java System Communications Express Customization Guide 10	
Table 2-1	List of Customizable files for Calendar and Address Book	16
Table 2-2	Location of the Common images in <code><uwc-images-dir></code>	17
Table 3-1	Customizing the Calendar Toolbar Icons	35
Table 3-2	Customizing Icons	36
Table 3-3	Customizing Task Related Informative Icons	36
Table 3-4	Customizing Error Icon for Full Page Errors	37
Table 3-5	Customizing Icons in Pop ups	37
Table 3-6	Customizing Pop ups	44
Table 4-1	JavaScript files that are used in Mail	49
Table 6-1	Communications Express User Interface Customizable Features	66
Table 6-2	Customizable LDAP attributes	69
Table 7-1	Commonly used templates in <code>search-template.xml</code> and <code>search.xml</code>	77
Table 7-2	Templates applicable for New/Edit Contact Window	78
Table 7-3	Templates defined for various sections in the New/Edit group	87
Table 8-1	Directory Structure for the Domain <code>siroe.com</code>	93
Table 8-2	Linking Multiple Domains to few distinct brands	94
Table 9-1	Localizing Date/Time Formats for Calendar	98
Table 9-2	Localization Rules	99
Table 9-3	Communications Express Specific Locales	100

Preface

This guide explains how to customize the look and feel of Sun Java™ System Communications Express. Although the product architecture permits an almost unlimited customization, 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](#)
- [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 Sun Java System Communications Express at your site. Developers may also find this guide useful for reference.

What You Need to Know

This book assumes a knowledge of Sun Java System Messaging Server software and an understanding of the following:

- JavaScript™
- HTML
- .jsp
- Email applications
- Web development

How This Book is Organized

Organization of the Sun Java System Communications Express Customization Guide

Table 0-1 Organization of the Sun Java System Communications Express Customization Guide

Chapter	Description
This Chapter	Describes the audience, requirements, organization, document conventions, and related information
Chapter 1, “Introduction to Communications Express and Customization”	This chapter provides a high-level overview on how to customize the look and feel of Sun Java System Communications Express.
Chapter 2, “Customizing General Features in Communications Express”	This chapter explains how to customize the general features in Sun Java System Communications Express.
Chapter 3, “Customizing General Features in Calendar”	This chapter explains how to customize the Calendar component in Sun Java System Communications Express.
Chapter 4, “Customizing General Features in Mail”	This chapter explains how to customize the Mail component in Sun Java System Communications Express
Chapter 5, “Customizing User Interface Features in Mail”	This chapter explains how to customize the User Interface Features of the Mail component in Sun Java System Communications Express
Chapter 6, “Customizing Advanced Features in Mail”	This chapter explains how to customize the Calendar component in Sun Java System Communications Express

Table 0-1 Organization of the Sun Java System Communications Express Customization Guide

Chapter	Description
Chapter 8, “Customizing Options”	This chapter discusses advanced customization techniques for the mail component.
Chapter 7, “Customizing General Features in Address Book”	This chapter explains how to customize the Address Book component in Sun Java System Communications Express
Chapter 9, “Localizing Communications Express”	This chapter explains how to localize Mail, Calendar and Address Book in Sun Java System Communications Express

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. For example, you might see something like this:

```
./installer
```

In this example, `./installer` is what you would type at the command line.

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.

For example, throughout this document you will see path references of the form:

```
msg_svr_base/...
```

The Messaging Server Base (*msg_svr_base*) represents the directory path in which you install the server. The default value of the *msg_svr_base* is `/opt/SUNWmsgsr.`

Italicized font is also used for variables within the synopsis of a command line utility. For example, the synopsis for the `commadmin admin remove` command is:

```
commadmin admin remove -D login -l userid -n domain -w password [-d domain]  
[-h] [-i inputfile] [-p port] [-X host] [-s] [-v]
```

In this example, the italicized words are arguments for their associated option. For example, in the `-w password` option, you would substitute the Administrator's password for *password* when you enter the `commadmin admin remove` command.

Where to Find Related Information

In addition to this guide, Sun Java Systems Communications Express comes with supplementary information for administrators as well as documentation for end users and developers. Use the following URL to see all the Communications Express documentation:

<http://docs.sun.com/db/prod/>

Where to Find This Book Online

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

<http://docs.sun.com/db/prod/>

Introduction to Communications Express and Customization

Sun Java™ System Communications Express 6 2004Q2 provides an integrated web-based communication and collaboration client that caters to the needs of Internet Service Providers, Enterprises, and Original Equipment Manufacturers.

As a web-based client, the Communications Express depends on a web container for access and a browser for presentation. The Communications Express consists of three client modules - Calendar, Address Book and Mail. The Calendar and Address Book client modules are deployed as a single application on the web container and are collectively referred to as the Unified Web Client.

The Calendar component of Communications Express uses `.jsp` pages as the presentation layer. These Java Server Pages pages can be customized to suit the requirements of the client.

The Address Book component of Communications Express uses XML/XSL files that contain XSL tags, static HTML and `.js` scripts. The XSL and JavaScript code are used to display dynamic data. These XSL files can be edited for customizing the Address Book component.

The Mail component of Communications Express uses the JavaScript language that is read and interpreted by the client. The JavaScript files are located on the server and downloaded to the client. The client extracts data from the JavaScript code to customize Communications Express functions. All modifications and customizations are done on the server.

Most of the features in Communications Express are fully customizable. Most features can also be customized easily during an upgrade. This guide explains the important features that are customizable.

Customizing General Features in Communications Express

This chapter describes how to customize some of the common elements in Communications Express. It also provides information related to `.jsp` files and their location along with details on the common features that are customizable in Communications Express.

This chapter provides details on

- [Guidelines to be Followed Before the Customization Process](#)
- [List of Customizable and Configurable Files for Calendar and Address Book](#)
- [Customizing Images](#)
- [Customizing Text](#)
- [Customizing the Skin](#)
- [Customizing the Login Screen for the English Locale](#)

Guidelines to be Followed Before the Customization Process

Before you start customizing the application, it is important to understand how to edit and deploy the `.jsp` files. The guidelines for customizing `.jsp` files are as follows.

1. Do not edit `.jsp` files directly in your production web server. Instead, make and test changes in a development environment.

2. Make a backup copy of the `.jsp` file.
3. Edit the `.jsp` file with any HTML or text editor. Typically HTML editors do not work well for `.jsp` files because of the amount of embedded Java code. In general when editing a `.jsp` file, be careful not to edit Java code.
4. After completing your edits, you can optionally compile the `.jsp` file. Because the `.jsp` file contains both HTML and Java code, this compilation ensures that you have not introduced syntax errors in the `.jsp` file.
5. Restart the web container for the changes to take effect.

List of Customizable and Configurable Files for Calendar and Address Book

The following list of files are customizable and configurable for Calendar and Address Book. It is assumed here that the customization is performed before configuring Communications Express. The directories mentioned are relative to `<install root>/lib/config-templates`. If configured, the files are available in the location `/var/opt/sunWuwc`

Table 2-1 List of Customizable files for Calendar and Address Book

File Location	Customizable Files	Name used in the document
WEB-INF/config	All the configuration files	<code><config-dir></code>
WEB-INF/domain	All <code>i18n.properties</code> files. <code>i18n</code> files are used to internationalize the interface for multiple languages and/or countries	<code><domain-dir></code>
WEB-INF/skin	<code>themes.properties</code> file	<code><skin-dir></code>
WEB-INF/ui/html/abs	All the <code>.html</code> , <code>.xml</code> and <code>.xsl</code> files	<code><ab-ui-dir></code>
absimg	All files such as <code>.gif</code> , <code>.css</code> , <code>.html</code> files	<code><ab-images-dir></code>
uwc/css	All stylesheets(<code>.css</code> files)	<code><css-dir></code>
uwc/images	All image files(<code>*.gif</code> , <code>*.jpg</code>)	<code><uwc-images-dir></code>
uwc/js	<code>.jsp</code> files	<code><js-dir></code>
uwc/calclient	<code>.jsp</code> files	<code><calclient-dir></code>
uwc/mailclient	<code>.jsp</code> files	<code><mailclient-dir></code>
uwc/common	<code>.jsp</code> files	<code><common-dir></code>

Customizing Images

The calendar images can be customized by specifying the location for these icons in `<skin-dir>/themes.properties`.

To customize the icons, edit the `themes.properties` file to change the names of properties to contain a value that points to the location of the new images. A default value for image location will be used if the corresponding key is not defined in the `theme.properties` file.

Table 2-2 lists the common images that are located in `<uwc-images-dir>`.

Table 2-2 Location of the Common images in `<uwc-images-dir>`

Image/Icon	Property Name	Value
Banner Image	<code>uwc-common-bannerImage</code>	<code>Masthead_s1UMC.gif</code>
Right Branding Image	<code>uwc-common-RightBrandingImage</code>	<code>logo_sun.gif</code>
Warning Image	<code>uwc-common-WarningImage</code>	<code>Warning_Large.gif</code>
Search Image	<code>uwc-common-SearchImage</code>	<code>LrlSearch_1_wo.gif</code>
Printable Image	<code>uwc-common-PrintableImage</code>	<code>LrlPrintable_1_wo.gif</code>
Import Export Image	<code>uwc-common-ImportExportImage</code>	<code>LrlImpExp_1_wo.gif</code>
Sort down and non selected image	<code>uwc-common-Sort-Down-NonSelected-image</code>	<code>sort_down_nonsel_che.gif</code>
Sort down and selected image	<code>uwc-common-Sort-Down-Selected-image</code>	<code>sort_down_sel.gif</code>
Sort up and non selected image	<code>uwc-common-Sort-Up-NonSelected-image</code>	<code>sort_up_nonsel.gif</code>
Sort up and selected image	<code>uwc-common-Sort-Up-Selected-image</code>	<code>sort_up_sel.gif</code>
Anchor Image	<code>uwc-common-To-Anchor-image</code>	<code>to_anchor.gif</code>

Table 2-2 Location of the Common images in `<uwc-images-dir>`

Image/Icon	Property Name	Value
Subscribed Image	<code>uwc-common-Subscribed-image</code>	<code>LrlSub_1.gif</code>
Required Image	<code>uwc-common-Required-image</code>	<code>required.gif</code>
Error Message Icon	<code>uwc-common-Error-Message-icon</code>	<code>Error_Large.gif</code>
Warning Message Icon	<code>uwc-common-Warning-Message-icon</code>	<code>Warning_Large.gif</code>
Question Message Icon	<code>uwc-common-Question-Message-icon</code>	<code>Question_Large.gif</code>
Information Message Icon	<code>uwc-common-Info-Message-icon</code>	<code>Info_Large.gif</code>

Customizing Text

Communications Express allows you to customize text that is displayed on web pages. This text is both customizable and localizable. You can also configure text on a per domain basis. To customize text, you need to change the `i18n.properties` file that is deployed in `<domain-dir>/domain/en`. The `i18n.properties` file in `<domain-dir>/domain/en` is the default file. If the deployment is configured to work with multiple domains and multiple locales, then you need to create a directory by that domain name under `<domain-dir>` appropriately.

To customize the text of a domain called `example.com`, perform the following steps.

1. Create `<domain-dir>/domain/example.com`
2. Copy `<domain-dir>/domain/en/i18n.properties` to `<domain-dir>/domain/example.com/en`. This would be the default path of the `i18n.properties` for users in `example.com` domain.
3. Create directories for each supported locale such as `en_US`, `ja`, `de_FR` under that domain, copy `i18n.properties` to the domain, and change the locations accordingly

Customizing the Skin

Creating a new skin

1. The `<skin-dir>` directory contains subdirectories, which correspond to each skin defined for Communications Express.
2. For example, to create a skin called "christmas". create a "christmas" directory under the `<skin-dir>` directory
3. Copy `themes.properties` under `<skin-dir>` directory to `<skin-dir>/christmas.themes.properties` file that defines the various parameters of the skin
4. Change the locations in `themes.properties` to point to the customized style sheets and images.
5. For help on various paramaters in this file, refer to to Appendix D of the Communications Express Administration Guide. Look for "themes.properties" under the "Configuration Parameters Reference" section.
6. To configure the `christmas` theme go to `<domain-dir>` and set `uwc-user-attr-sunUCTheme=christmas` in the `uwcdomainconfig.properties` file.
7. Restart the web server to apply changes.

Modifying a skin

1. The `<skin-dir>` directory contains subdirectories, which correspond to each skin defined for Communications Express.
2. Go to the skin directory that you want to update. For example, to update the skin called "christmas". change to `<skin-dir>/christmas` directory.
3. The `<skin-dir>/christmas` contains a file called `themes.properties`. This file defines the various parameters of the skin.
4. Update the required parameter(s) in the `themes.properties` file.
5. For help on various paramaters in this file, refer to Appendix D of the Communications Express Administration Guide. Look for "themes.properties" under the "Configuration Parameters Reference" section.
6. Restart the web server to apply changes

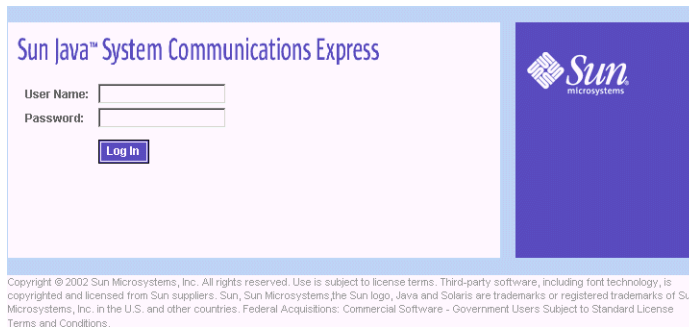
Customizing the Login Screen for the English Locale

This section describes how to modify the Communications Express Login screen shown in [Figure 2-1](#).

You can replace the logo with a custom graphic on the Communications Express Login screen:

To modify the Communications Express Login Screen, you need to edit `<skin-dir>/themes.properties` and the relevant `.jsp` file is `login.jsp`.

Figure 2-1 Communications Express Login Screen.



Replacing the Logo With a Custom Graphic

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

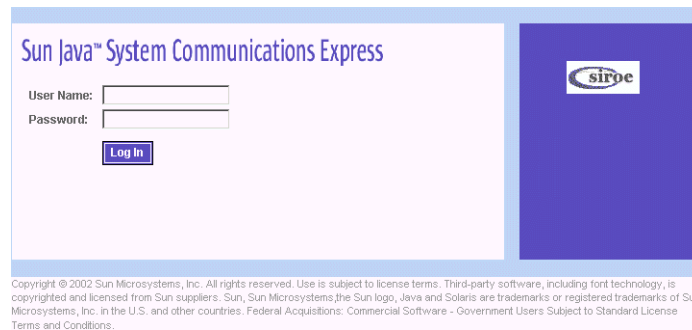
- Replace the logo with a custom graphic
- Change the destination of link

Figure 2-2 Communications Express Corner Logo and Link



Example—Logo Modification

The example shown in [Figure 2-3](#) replaces the Sun logo with a custom logo having different dimensions.

Figure 2-3 Example Corner Logo**Figure 2-4** Login Screen with Customized Logo

Changing the Title Text

Figure 2-5 Communications Express Title Text

You can also do the following:

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

Customizing Banner

You can modify the left image and the right image in the banner. These images show the banner before and after customization.

Figure 2-6 Left Banner Image Before Customization



Figure 2-7 Customized Left Banner Image



Figure 2-8 Right Banner Image Before Customization



Figure 2-9 Customized Right Banner Image



Customizing Application Bar

You can modify the application bar by editing the `CalApplBarNormal.jsp` file located at `<calclient-dir>/CalApplBarNormal.jsp`

By default, the order in which the application tab appear, provided all services are enabled, is 'Mail', 'Calendar', 'Address Book' and 'Options'. This order can be changed to suit your requirements.

Figure 2-10 Default application tab order



Code Example 2-1 shows the default code that displays the default application tab order .

Code Example 2-1 Code for customizing the Application tab order

```

<td class="TablGutter">
  <table border="0" cellspacing="0" cellpadding="0" class="TablTbl">
    <tr>
      <td colspan="12"></td>
    </tr>
    <tr>
      <jato:content name="MailTab">
        <td class="TablGutter" colspan="3"></td>
      </jato:content>
        <td rowspan="3" class="TablSel">
          <div class="TablSel"><span class="TablLblSel" title="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-Calendar",
"Calendar") %>"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-Calendar", "Calendar") %></span></div>
        </td>
        <td rowspan="3" class="TablSel"></td>
        <td class="TablGutter" colspan="7"></td>
      </tr>
      <tr>
        <jato:content name="MailTab">
          <td class="TablNotSel">
            <div class="TablNotSel"><a href="<%= getContextURI(request)
%>/mailclient/Mail" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Mail", "Mail") %>" class="TablLblNormal"
onmouseover="status='Mail'; return true;" onmouseout="status='';"><%=
getLocalizedLabel(session, "uwc-calclient-toolbar-Mail", "Mail")
%></a></div>
          </td>
          <td class="TablNotSel"></td>
          <td class="TablGutter"></td>
        </jato:content>
        <td class="TablGutter"></td>
        <td class="TablNotSel" nowrap>

```

Code Example 2-1 Code for customizing the Application tab order

```

<td class="TablGutter">
    <div class="TablNotSel"><a href="<%= getContextURI(request)
%>/abclient/AddressBook" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-AddressBook", "Address Book") %>"
class="TablLblNormal" onmouseover="status='Address Book'; return true;"
onmouseout="status='';"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-AddressBook", "Address Book") %></a></div>
    </td>
    <td class="TablNotSel"></td>
    <td class="TablGutter"></td>
    <td class="TablNotSel">
        <div class="TablNotSel"><a href="<%= getContextURI(request)
%>/base/CalendarPreferences" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Options", "Options") %>"
class="TablLblNormal" onmouseover="status='Options'; return true;"
onmouseout="status='';"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-Options", "Options") %></a></div>
        </td>
    <td class="TablNotSel"></td>
    <td class="TablGutter"></td>
</tr>
<tr>
    <td colspan="3" class="TablGutter"></td>
    <td colspan="7" class="TablGutter"></td>
</tr>
</table>
</td>

```

The application tab order can be customized to display in the order, Address Book, Mail, Calendar and Options.

Figure 2-11 Changing the application tab order to display Address Book, Mail, Calendar and Options



To change the order of Application tabs for example 'Address Book', 'Mail', 'Calendar', 'Options' re-arrange the code shown in [Code Example 2-1](#) to the following:

Code Example 2-2 Customizing the order of the application tabs

```

<td class="TablGutter">
<table border="0" cellspacing="0" cellpadding="0" class="TablTbl">
<tr>
<td colspan="12"></td>
</tr>
<tr>
<td class="TablGutter"></td>
<td class="TablNotSel" nowrap>
<div class="TablNotSel"><a href="<%= getContextURI(request)
%>/abclient/AddressBook" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-AddressBook", "Address Book") %>"
class="TablLblNormal" onmouseover="status='Address Book'; return true;"
onmouseout="status='';"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-AddressBook", "Address Book") %></a></div>
</td>
<jato:content name="MailTab">
<td class="TablNotSel">
<div class="TablNotSel"><a href="<%= getContextURI(request)
%>/mailclient/Mail" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Mail", "Mail") %>" class="TablLblNormal"
onmouseover="status='Mail'; return true;" onmouseout="status='';"><%=
getLocalizedLabel(session, "uwc-calclient-toolbar-Mail", "Mail")
%></a></div>
</td>
<td class="TablNotSel"></td>
<td class="TablGutter"></td>
</jato:content>
<jato:content name="MailTab">
<td class="TablGutter" colspan="3"></td>
</jato:content>
<td rowspan="3" class="TablSel">
<div class="TablSel"><span class="TablLblSel" title="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-Calendar",
"Calendar") %>"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-Calendar", "Calendar") %></span></div>
</td>
<td rowspan="3" class="TablSel"></td>

```

Code Example 2-2 Customizing the order of the application tabs

```

<td class="TablGutter">
    <td class="TablGutter" colspan="7"></td>
</tr>
<tr>
    <td class="TablNotSel"></td>
    <td class="TablGutter"></td>
    <td class="TablNotSel">
        <div class="TablNotSel"><a href="<%= getContextURI(request)
%>/base/CalendarPreferences" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Options", "Options") %>"
class="TablLblNormal" onmouseover="status='Options'; return true;"
onmouseout="status='';"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-Options", "Options") %></a></div>
        </td>
    <td class="TablNotSel"></td>
    <td class="TablGutter"></td>
</tr>
<tr>
    <td colspan="3" class="TablGutter"></td>
    <td colspan="7" class="TablGutter"></td>
</tr>
</table>
</td>

```

Customizing General Features in Calendar

The look and feel of the calendar is provided by a set of style sheets, images and localizable strings. These strings pertain to the text that is displayed on the pages. The Code examples show how to modify icons, text and stylesheets for Calendar.

NOTE This chapter does not provide code samples for all customization scenarios in Calendar. The code samples have been provided only for illustrative purposes. This cannot be used directly for customization.

These are some common customization scenarios.

- [Customizing Icons](#)
- [Customizing Views](#)
- [Customizing Pop ups](#)
- [Customizing Full Page Error](#)

Customizing Icons

- [Customizing the Order of Calendar Toolbar Icons](#)
- [Customizing Icons](#)

- [Customizing the Toolbar Images](#)
- [Customizing Events/Task Related Informative Icons](#)
- [Customizing the Error Icon for Full Page Errors](#)
- [Customizing Icons in Pop ups](#)
- [Customizing Search for Events](#)

Customizing the Order of Calendar Toolbar Icons

The file to be modified to change the calendar toolbar icon order is `<calclient-dir>/CalToolBarNormal.jsp`

[Figure 3-1](#) shows the default order of the Calendar Toolbar Icons.

Figure 3-1 Order of Calendar Toolbar Icons



[Code Example 3-1](#) shows the default code used to display the Calendar Toolbar Icons in the order, New Event, New Task, Check Availability, Search for Calendar, Printable and Import/Export.

Code Example 3-1 Default code that displays the default Calendar Toolbar Icons

```
<div class="Toolbar">
<a name="toolbar"></a>
<script language="JavaScript" src="../uwc/js/define_images_cal.js">
</script>
<table border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td nowrap><a href="javascript:void(0)" accesskey="e"
onmouseover="over('newEvent')" onmouseout="out('newEvent')"
onClick="openWinAutoHeight('<jato:text name="NewEventUrl"
```

Code Example 3-1 Default code that displays the default Calendar Toolbar Icons

```

<div class="Toolbar">
escape="false"/>', 'eventWin', 'scrollbars=yes,resizable=yes,width=700');
return false;">" width="24" height="24"
align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewEvent", "New Event") %>" alt="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-NewEvent", "New
Event") %>"></a>
  <a href="javascript:void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewEvent", "New Event") %>" class="ToolLbl"
onClick="openWinAutoHeight('<jato:text name="NewEventUrl"
escape="false"/>', 'eventWin', 'scrollbars=yes,resizable=yes,width=700');
return false;"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-NewEvent", "New Event") %></a></td>
New Task
  <td class="ToolbarItem" nowrap><a href="javascript:void(0)"
accesskey="t" onMouseOver="over('newTask')" onMouseOut="out('newTask')"
onClick="openWinAutoHeight('<jato:text name="NewTaskUrl"
escape="false"/>', 'task', 'scrollbars=yes,resizable=yes,width=700'); return
false;">"
width="24" height="24" align="absmiddle" border="0" title="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-NewTask", "New
Task") %>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewTask", "New Task") %>"></a>
  <a href="javascript:void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewTask", "New Task") %>" class="ToolLbl"
onClick="openWinAutoHeight('<jato:text name="NewTaskUrl"
escape="false"/>', 'task', 'scrollbars=yes,resizable=yes,width=700'); return
false;"><%= getLocalizedLabel(session, "uwc-calclient-toolbar-NewTask",
"New Task") %></a></td>
Check Availability
  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
accesskey="a" onMouseOver="over('checkAvail')"
onMouseOut="out('checkAvail')" onClick="openWinAutoHeight('<jato:text
name="AvailabilityUrl"
escape="false"/>', 'availability', 'scrollbars=yes,resizable=yes,width=800');
return false;">" width="24" height="24"
align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-CheckAvailability", "Check Availability")
%>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-CheckAvailability", "Check Availability")
%>"></a>

```

Code Example 3-1 Default code that displays the default Calendar Toolbar Icons

```

<div class="Toolbar">
  <a href="javascript: void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-CheckAvailability", "Check Availability")
%"> class="ToolLbl" onClick="openWinAutoHeight('<jato:text
name="AvailabilityUrl"
escape="false"/>', 'availability', 'scrollbars=yes,resizable=yes,width=800');
return false;"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-CheckAvailability", "Check Availability")
%"></a></td>
Search for Calendar
<!-- Show SearchCalendar if and only if the parent view is not manage
calendars -->
<jato:content name="ShowSearchCalendars">

  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
accesskey="s" onMouseOver="over('search')" onMouseOut="out('search')"
onClick="openSearchForCalendarsPopup(); return false;">" width="24" height="24"
align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-SearchCalendar", "Search for Calendar") %>"
alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-SearchCalendar", "Search for Calendar")
%">"></a>
  <a href="javascript: void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-SearchCalendar", "Search for Calendar") %>"
class="ToolLbl" onClick="openSearchForCalendarsPopup(); return false;"><%=
getLocalizedLabel(session, "uwc-calclient-toolbar-SearchCalendar", "Search
for Calendar") %></a></td>

</jato:content>
Printable
<!-- Show Printable f and only if the parent view is not invitations -->
<jato:content name="ShowPrintable">

  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
onClick="openWinAutoHeight('<jato:text name="PrintUrl"
escape="false"/>', 'print', 'menubar=yes,scrollbars=yes,resizable=yes,width=7
00,height=650'); return false;" accesskey="p" class="ToolLbl"
onMouseOver="over('printable')" onMouseOut="out('printable')">"
width="24" height="24" align="absmiddle" border="0" title="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-Printable",
"Display a Printable Page") %>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Printable", "Display a Printable Page")
%">"></a>

```


Code Example 3-1 Default code that displays the default Calendar Toolbar Icons

```

<div class="Toolbar">
  <a href="javascript:void(0)" onClick="openWinAutoHeight('<jato:text
name="PrintUrl"
escape="false"/>', 'print', 'menubar=yes,scrollbars=yes,resizable=yes,width=7
00,height=650'); return false;" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Printable", "Display a Printable Page") %>"
class="ToolLbl"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-Printable", "Printable") %></a></td>

</jato:content>
Import/Export
  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
accesskey="i" onMouseOver="over('importExport')"
onMouseOut="out('importExport')" onClick="openWinAutoHeight('<jato:text
name="ImportExportUrl"
escape="false"/>', 'importexport', 'scrollbars=yes,resizable=yes,width=700');
return false;">" width="24" height="24"
align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-ImportExport", "Import and Export Calendar")
%>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-ImportExport", "Import and Export Calendar")
%>"></a>
  <a href="javascript: void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-ImportExport", "Import and Export Calendar")
%>" class="ToolLbl" onClick="openWinAutoHeight('<jato:text
name="ImportExportUrl"
escape="false"/>', 'importexport', 'scrollbars=yes,resizable=yes,width=700');
return false;"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-ImportExport", "Import/Export") %></a>
  </td>
</tr>
</table>
</div>

```

Figure 3-2 Changed order of the Toolbar Icons

Code Example 3-2 shows the edited code that changes the order of icons in calendar tool bar to - New Task, New Event, Printable, Check Availability, Import/Export, Search for Calendar

:

Code Example 3-2 Customizing the toolbar icon order

```

<div class="Toolbar">

<a name="toolbar"></a>

<script language="JavaScript" src="../uwc/js/define_images_cal.js">
</script>
<table border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td class="ToolbarItem" nowrap><a href="javascript:void(0)"
accesskey="t" onMouseOver="over('newTask')" onMouseOut="out('newTask')"
onClick="openWinAutoHeight('<jato:text name="NewTaskUrl"
escape="false"/>', 'task', 'scrollbars=yes,resizable=yes,width=700'); return
false;">"
width="24" height="24" align="absmiddle" border="0" title="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-NewTask", "New
Task") %>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewTask", "New Task") %>"></a>
      <a href="javascript:void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewTask", "New Task") %>" class="ToolLbl"
onClick="openWinAutoHeight('<jato:text name="NewTaskUrl"
escape="false"/>', 'task', 'scrollbars=yes,resizable=yes,width=700'); return
false;"><%= getLocalizedLabel(session, "uwc-calclient-toolbar-NewTask",
"New Task") %></a></td>
    <td class="ToolbarItem" nowrap><a href="javascript:void(0)" accesskey="e"
onMouseOver="over('newEvent')" onMouseOut="out('newEvent')"
onClick="openWinAutoHeight('<jato:text name="NewEventUrl"
escape="false"/>', 'eventWin', 'scrollbars=yes,resizable=yes,width=700');
return false;">" width="24" height="24"
align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewEvent", "New Event") %>" alt="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-NewEvent", "New
Event") %>"></a>
      <a href="javascript:void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-NewEvent", "New Event") %>" class="ToolLbl"
onClick="openWinAutoHeight('<jato:text name="NewEventUrl"
escape="false"/>', 'eventWin', 'scrollbars=yes,resizable=yes,width=700');
return false;"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-NewEvent", "New Event") %></a></td>
    <td class="Printable"
    <!-- Show Printable f and only if the parent view is not invitations -->
    <jato:content name="ShowPrintable">

```

Code Example 3-2 Customizing the toolbar icon order (*Continued*)

```

<div class="Toolbar">
  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
onClick="openWinAutoHeight(' <jato:text name="PrintUrl"
escape="false"/>', 'print', 'menubar=yes,scrollbars=yes,resizable=yes,width=7
00,height=650'); return false;" accesskey="p" class="ToolLbl"
onMouseOver="over('printable')" onMouseOut="out('printable')">"
width="24" height="24" align="absmiddle" border="0" title="<%=
getLocalizedLabel(session, "uwc-calclient-toolbar-tooltip-Printable",
"Display a Printable Page") %>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Printable", "Display a Printable Page")
%>"></a>
  <a href="javascript:void(0)" onClick="openWinAutoHeight(' <jato:text
name="PrintUrl"
escape="false"/>', 'print', 'menubar=yes,scrollbars=yes,resizable=yes,width=7
00,height=650'); return false;" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-Printable", "Display a Printable Page") %>"
class="ToolLbl"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-Printable", "Printable") %></a></td>

</jato:content>
Check Availability
  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
accesskey="a" onMouseOver="over('checkAvail')"
onMouseOut="out('checkAvail')" onClick="openWinAutoHeight(' <jato:text
name="AvailabilityUrl"
escape="false"/>', 'availability', 'scrollbars=yes,resizable=yes,width=800');
return false;">" width="24" height="24"
align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-CheckAvailability", "Check Availability")
%>" alt="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-CheckAvailability", "Check Availability")
%>"></a>
  <a href="javascript: void(0)" title="<%= getLocalizedLabel(session,
"uwc-calclient-toolbar-tooltip-CheckAvailability", "Check Availability")
%>" class="ToolLbl" onClick="openWinAutoHeight(' <jato:text
name="AvailabilityUrl"
escape="false"/>', 'availability', 'scrollbars=yes,resizable=yes,width=800');
return false;"><%= getLocalizedLabel(session,
"uwc-calclient-toolbar-CheckAvailability", "Check Availability")
%></a></td>
Import/Export

```

Code Example 3-2 Customizing the toolbar icon order (*Continued*)

```

<div class="Toolbar">
  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
  accesskey="i" onMouseOver="over('importExport')"
  onMouseOut="out('importExport')" onClick="openWinAutoHeight('<jato:text
  name="ImportExportUrl"
  escape="false"/>', 'importexport', 'scrollbars=yes,resizable=yes,width=700');
  return false;">" width="24" height="24"
  align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-tooltip-ImportExport", "Import and Export Calendar")
  %>" alt="<%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-tooltip-ImportExport", "Import and Export Calendar")
  %>"></a>
  <a href="javascript: void(0)" title="<%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-tooltip-ImportExport", "Import and Export Calendar")
  %>" class="ToolLbl" onClick="openWinAutoHeight('<jato:text
  name="ImportExportUrl"
  escape="false"/>', 'importexport', 'scrollbars=yes,resizable=yes,width=700');
  return false;"><%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-ImportExport", "Import/Export") %></a>
  </td>
  Search for Calendar
  <!-- Show SearchCalendar if and only if the parent view is not manage
  calendars -->
  <jato:content name="ShowSearchCalendars">

  <td class="ToolbarItem" nowrap><a href="javascript: void(0)"
  accesskey="s" onMouseOver="over('search')" onMouseOut="out('search')"
  onClick="openSearchForCalendarsPopup(); return false;">" width="24" height="24"
  align="absmiddle" border="0" title="<%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-tooltip-SearchCalendar", "Search for Calendar") %>"
  alt="<%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-tooltip-SearchCalendar", "Search for Calendar")
  %>"></a>
  <a href="javascript: void(0)" title="<%= getLocalizedLabel(session,
  "uwc-calclient-toolbar-tooltip-SearchCalendar", "Search for Calendar") %>"
  class="ToolLbl" onClick="openSearchForCalendarsPopup(); return false;"><%=
  getLocalizedLabel(session, "uwc-calclient-toolbar-SearchCalendar", "Search
  for Calendar") %></a></td>

  </jato:content>
</tr>
</table>
</div>

```

Customizing the Toolbar Images

The property key names that define the location of the toolbar images are mentioned in [Table 3-1](#)

Table 3-1 Customizing the Calendar Toolbar Icons

Icon	Property Name	Value
New Event icon.	uwc-calclient-NewEventImage	LrlNewEvent_1_wo.gif,
New Task Icon	uwc-calclient-NewTaskImage	LrlNewTask_1_wo.gif
Check Availability Icon	uwc-calclient-CheckAvailabilityImage	LrlCheckAvail_1_wo.gif
Search Calendar Icon	uwc-calclient-SearchImage	LrlSearch_1_wo.gif
Printable Icon	uwc-calclient-PrintableImage	LrlPrintable_1_wo.gif
Import/Export Icon	uwc-calclient-ImportExportImage	LrlImpExp_1_wo.gif

Customizing Icons

To customize the icons in Calendar views, you need to modify the `themes.properties` file located at `<skin-dir>`.

You can customize toolbars for all the main calendar views such as the following.

- Day view
- Week view
- Month view
- Year view
- Events view
- Invitations view
- Tasks view
- Manage calendar view

[Table 3-2](#) shows the list of customizable icons in Communications Express.

Table 3-2 Customizing Icons

Image	Property name	Value
Forward	uwc-calclient-Forward-Arrow-image	forward.gif
Backward	uwc-calclient-Back-Arrow-image	backward.gif
Select all	uwc-calclient-Select-All-image	check_all.gif
Unselect all	uwc-calclient-Deselect-All-image	uncheck_all.gif
Sort down non selected image	uwc-common-Sort-Down-NonSelected-image	sort_down_nonsel_che.gif
Sort down selected image	uwc-common-Sort-Down-Selected-image	sort_down_sel.gif
Sort up non selected image	uwc-common-Sort-Up-NonSelected-image	sort_up_nonsel.gif
Sort up selected image	uwc-common-Sort-Up-Selected-image	sort_up_sel.gif

Customizing Events/Task Related Informative Icons

The property key names that define the location of the task related informative icons mentioned in [Table 3-3](#)

You can customize informative icons or images such as the ones used for recurring, reminding, and public tasks for tasks and events.

Table 3-3 Customizing Task Related Informative Icons

Icon	Property Name	Value
Notify Image	uwc-calclient-calclient-NotifyImage,	LrlNotify_1.gif
Recurring Image	uwc-calclient-RecurringImage	LrlRecur_1.gif
Public Image	uwc-calclient-PublicImage	LrlPrvPub_1.gif

Customizing the Error Icon for Full Page Errors

[Table 3-4](#) shows the customizable error icons for Full Page Errors

Table 3-4 Customizing Error Icon for Full Page Errors

Icon	Property Name	Value
Error Icon	uwc-calclient-Error-Message-icon	Error_Large.gif

Customizing Icons in Pop ups

[Code Example 3-5](#) shows the customizable icons located in `uwc/calclient/SearchCalendarData.jsp`.

Table 3-5 Customizing Icons in Pop ups

Icon	Property Name	Value
select_all	uwc-calclient-Select-All -image	check_all.gif
unselect_all	uwc-calclient-Deselect-A ll-image	uncheck_all.gif
subscribed icon	uwc-calclient-Subscribed -image	LrlSub_1.gif

Customizing Search for Events

The File name including path is `<calclient-dir>/CalSearchBarPagelet.jsp`.

- Communications Express allows you to customize Search for events label as shown in [Figure 3-3](#)

Figure 3-3 Customizing Search for Events

Search for Events:

This is applicable for the following views

- Day view
- Week view

- Month view
- Year view
- Events manager.
- You can change the text that appears besides the search events text box.
- You can also customize the overall layout.

Customizing the Text That Appears Besides the Search Events Text Box

To change the introductory text that is displayed adjacent to the search events text box, change the code in the resource bundle of `il8n.properties`.

[Code Example 3-3](#) shows the code before modifying the resource bundle

Code Example 3-3 Code before modifying the resource bundle

```
uwc-calclient-toolbar-SearchForEvents=Search for Events:
```

[Code Example 3-4](#) shows the code after modifying the resource bundle.

Code Example 3-4 Code after modifying the resource bundle

```
uwc-calclient-toolbar-SearchForEvents=<Customized Text>
```

Customizing Views

These examples show how to modify views

- [Customizing Tasks Search View](#)
- [Customizing the Column Order in Tasks Printable View](#)
- [Customizing Tasks Error View](#)
- [Customizing Events view](#)
- [Customizing Tasks View](#)

Customizing Tasks Search View

- The `.jsp` file name including path that pertains to tasks search view is `<calclient-dir>/tasksSearch.jsp`
- The common components present in `tasksSearch.jsp` are: Banner, Application bar, calendar tool bar.
- In the `tasksSearch.jsp` you can customize
 - The overall layout and look and feel.
 - Icons or images such as `select_all`, `unselect_all`, `sort_up`, `sort_down`, `sort_selected_up`, `sort_selected_down` gifs.
 - Task related informative icons or images such as `recurring gif`, `reminding gif`, `public task gifs`.
 - The table column ordering.
 - The edit task and view task popup window sizes and styles.

Customizing the Column Order in Tasks Printable View

The file pertaining to customizing the column order in tasks is `<calclient-dir>/tasksPrint.jsp`

The common components present are Banner and Application bar.

You can customize the table column ordering.

The `tasksPrint.jsp` is located at `<calclient-dir>/tasksPrint.jsp`

Code Example 3-5 shows the code used to customize Tasks print view in a table with the columns in the order - 'Priority', 'Title', 'Due Date', 'Status', 'Type'

:

Code Example 3-5 Default code that displays the table column order

```

        <th class="TblThCl1" scope="col" nowrap><strong><%=
getLocalizedLabel(session, "uwc-calclient-tasks-Priority", "Priority")
%></strong>
        </th>
        <th width="35%" scope="col" nowrap><strong><%=
getLocalizedLabel(session, "uwc-calclient-tasks-Title", "Title")
%></strong></th>

```


Code Example 3-5 Default code that displays the table column order

```

        "
width="12" height="12" border="0" hspace="2" title="<%=
getLocalizedLabel(session, "uwc-calclient-tasks-Public", "Public") %>"
alt="<%= getLocalizedLabel(session, "uwc-calclient-tasks-Public", "Public")
%>">
        </jato:content>
        <jato:content name="IsExternal">
            <br>
            [<jato:text name="OwnerName"/>&nbsp;<jato:text
name="CalendarName"/>]
        </jato:content>
    </td>
</jato:content>
    <td <jato:content name="NormalColumnStyle"/><jato:text
name="DueDate"/>&nbsp;<jato:text name="DueTime"/></td>
        <td <jato:content name="NormalColumnStyle"/><jato:text
name="PercentStatusAsText" escape="false"/></td>
        <td <jato:content name="NormalColumnStyle"/><jato:text
name="Category" escape="false"/></td>
    </tr>
</jato:tiledView>
</jato:content>

```

Code Example 3-6 shows the code required to change the table column ordering so that it displays in the order title, due date, status, priority and type

Code Example 3-6 Changed table column ordering

```

        <tr>
            <th class="TblThCl1" scope="col" nowrap><strong><%=
getLocalizedLabel(session, "uwc-calclient-tasks-Priority", "Title")
%></strong>
            </th>
            <th width="35%" scope="col" nowrap><strong><%=
getLocalizedLabel(session, "uwc-calclient-tasks-Title", "Due Date")
%></strong></th>
            <th scope="col" nowrap><strong><%= getLocalizedLabel(session,
"uwc-calclient-tasks-Status", "Status") %></strong></th>
            <th scope="col" nowrap><strong><%= getLocalizedLabel(session,
"uwc-calclient-tasks-DueDate", "Priority") %></strong></th>
            <th scope="col" nowrap><strong><%= getLocalizedLabel(session,
"uwc-calclient-tasks-Type", "Type") %></strong></th>
        </tr>

        <jato:content name="ZeroTasks">
            <tr>

```

Code Example 3-6 Changed table column ordering

```

        <tr>
            <td class="TblTdCl1Lst" colspan="6"><%=
getLocalizedLabel(session, "uwc-calclient-tasks-NoTasksToDisplay-message",
"This view does not have any tasks to display. Choose another filter or
search for tasks.") %></td>
        </tr>
    </jato:content>
    <jato:content name="NonZeroTasks">
        <jato:tiledView name="TasksTiledView"
type="com.sun.uwc.calclient.TasksTileView">
            <jato:hidden name="TaskCalID"/>
            <jato:hidden name="TaskUID"/>
            <jato:hidden name="TaskRID"/>
            <tr>
                <jato:content name="IsConfidential">
                    <td <jato:content name="TitleColumnStyle"/><%=
getLocalizedLabel(session, "il8nModel", "uwc-calclient-view-Busy", "Busy")
%></td>
                </jato:content>
                <jato:content name="IsNotConfidential">
                    <td <jato:content name="FirstColumnStyle"/>
align="center"><strong><jato:text name="Title"/></strong><br>
                    <jato:content name="HasAlarm">
                        "
width="12" height="12" border="0" hspace="2" align="absmiddle" alt="<%=
getLocalizedLabel(session, "uwc-calclient-tasks-Notify", "Notify") %>"
title="<%= getLocalizedLabel(session, "uwc-calclient-tasks-Notify",
"Notify") %>">
                    </jato:content>
                    <jato:content name="IsRecurring">
                        "
width="12" height="12" border="0" align="absmiddle" hspace="2" title="<%=
getLocalizedLabel(session, "uwc-calclient-tasks-Recurring", "Recurring")
%>" alt="<%= getLocalizedLabel(session, "uwc-calclient-tasks-Recurring",
"Recurring") %>">
                    </jato:content>
                    <jato:content name="IsPublic">
                        "
width="12" height="12" border="0" hspace="2" title="<%=
getLocalizedLabel(session, "uwc-calclient-tasks-Public", "Public") %>"
alt="<%= getLocalizedLabel(session, "uwc-calclient-tasks-Public", "Public")
%>">
                    </jato:content>
                    <jato:content name="IsExternal">
                        <br>
                        [<jato:text name="OwnerName"/>&nbsp;  <jato:text
name="CalendarName"/>]
                    </jato:content>

```

Code Example 3-6 Changed table column ordering

```

        <tr>
            </td>
        </jato:content>
        <td <jato:content name="NormalColumnStyle"/><jato:text
name="DueDate"/>&nbsp;<jato:text name="DueTime"/></td>
            <td <jato:content name="NormalColumnStyle"/><jato:text
name="PercentStatusAsText" escape="false"/></td>
            <td <jato:content name="NormalColumnStyle"/><jato:text
name="Priority" escape="false"/></td>
            <td <jato:content name="NormalColumnStyle"/><jato:text
name="Category" escape="false"/></td>
        </tr>
    </jato:tableView>

```

Customizing Tasks Error View

To modify the tasks error view, you need to edit the file
<calclient-dir>/tasksError.jsp

Customizing Events view

- To modify the events view, edit the following .jsp file.

<calclient-dir>/EventsManager.jsp

- To modify the search events view, edit the following .jsp file.

<calclient-dir>/eventsSearch.jsp

- To modify the anonymous events view, edit the following .jsp file.

<calclient-dir>/EventsManagerAnon.jsp

- To modify the anonymous search events view, edit the following .jsp file.

<calclient-dir>/eventsSearchAnon.jsp

- To modify the anonymous Error view, edit the following .jsp file.

<calclient-dir>/eventsanonError.jsp

To modify the printable images view, edit the following image. You can change the image and then replace the changed image in the root folder.

uwc-common-PrintableImage=../uwc/images/Lr1Printable_1_wo.gif

Customizing Tasks View

To modify the tasks view, you need to modify the file `<calclient-dir>/tasks.jsp`.

Customizing Pop ups

[Table 3-6](#) shows the pop up names and the relevant .jsp files to be modified for customization.

Table 3-6 Customizing Pop ups

Pop up Name	jsp files to be modified
New and Edit Event Pop up	<code><calclient-dir>/EditEvent.jsp</code> <code><calclient-dir>/EditEventData.jsp</code> <code><js-dir>/DeleteEventConfirmRepeatSelection.jsp</code> <code><js-dir>/SaveEventConfirmRepeatSelection.jsp</code> <code><js-dir>/EditEventButtons.jsp</code> <code><js-dir>/NewEventButtons.jsp</code>
View Calendar Pop up	<code><calclient-dir>/ViewCalendar.jsp</code> <code><calclient-dir>/ViewCalendarData.jsp</code> <code><js-dir>/ViewCalendarButtons.jsp</code>
New and Edit Calendar Group Pop up	<code><calclient-dir>/EditCalGroup.jsp</code> <code><calclient-dir>/EditCalGroupData.jsp</code> <code><js-dir>/EditCalGroupButtons.jsp</code> <code><js-dir>/NewCalGroupButtons.jsp</code>
Subscribe to Calendars Pop up and Search for Calendars Pop up	<code><calclient-dir>/SearchCalendar.jsp</code> <code><calclient-dir>/SearchCalendarData.jsp</code> <code><calclient-dir>/SearchCalendarInitial.jsp</code> <code><js-dir>/SearchCalendarButtons.jsp</code> <code><js-dir>/SearchNViewCalendarButtons.jsp</code>

Customizing Full Page Error

To modify the Error page, you need to edit the file `<common-dir>/FullPageError.jsp`

You can customize the error page for calendar views such as

- Day view
- Week view
- Month view
- Year view
- Events manager
- Invitations view
- Tasks view
- Manager calendar view.

Customizing the Layout for Full Page Errors

Full page error appears for variety of reasons. This page displays the following information:

- Full Page Error Summary
- Full Page Error Sub-Title
- Full Page Error Description
 - FullPageErrorSummary refers to Full Page Error Summary
 - FullPageErrorSubTitle refers to Full Page Error Sub-Title
 - FullPageErrorDescription refers to Full Page Error Description

These keys are replaced with the actual information provided by the server depending on the type or cause of error.

To customize the information that is displayed for each of these keys:

Edit the `i18n.properties` file, look for the following property names and change the value of the property as follows:

```
<per-view-key>-FullPageError-<ReasonForError>-summary=<Summary to be displayed>
```

```
<per-view-key>-FullPageError-<ReasonForError>-subtitle=<Sub Title to be displayed>
```

`<per-view-key>-FullPageError-<ReasonForError>-description=<Description to be displayed>`

For example,

```
calclient-error-dayviewview-FullPageError-CalendarNotAvailable-summary=
Calendar Not Available
```

```
calclient-error-dayviewview-FullPageError-CalendarNotAvailable-subtitle
=Could Not Display View
```

```
calclient-error-dayview-FullPageError-CalendarNotAvailable-description
=The selected calendar was either deleted or does not exist or you do not have
permissions to view it. Select another calendar.
```


Customizing General Features in Mail

This chapter includes information on the customizable features in the Mail component of Communications Express. This chapter has the following sections.

- [Defining Your Customization Requirements](#)
- [Understanding the File Layout in Mail](#)
- [Understanding Basic Interfaces in Mail](#)

Defining Your Customization Requirements

Before you customize mail for Communications Express, you need to decide your customization goals. There are several common customization scenarios.

- [Customizing the Skin to Modify Global Look and Feel](#)
- [Customizing Text](#)
- [Customizing Mail by Domain](#)
- [Customizing Mail Icons](#)
- [Adding Extra Features](#)

Customizing the Skin to Modify Global Look and Feel

To customize the skin, you need to understand the structure of the underlying Cascading Style Sheet (CSS). The Skin color, font style, or font size can be globally changed in the CSS file.

Customizing Text

Since the mail client supports internationalization, most of the text used in mail are referenced through a special array named `i18n`. The strings are saved in `xx/i18n.js`, where `xx` is the locale name for the specific language. To replace the text with a desired text, you need to look up the string from `xx/i18n.js` and modify it. The default encoding for all files is UTF-8, and you need to make sure that you use the right one, especially when the text contains non-Latin characters.

Customizing Mail by Domain

Mail supports customization specific for each domain. A user who belongs to a specific domain will be directed to a different view (skin) on signing in. Refer the chapter [Chapter 8, “Customizing Communications Express for a Specific Domain,”](#) for more information on customizing mail by domain.

Customizing Mail Icons

All Image files are stored in `img/` directory. You can look up the image file name from html files and replace the URL with the required one.

Adding Extra Features

For functionalities that are not supported by the current mail component, you can create a new function using the `listFrameHTML()` function in the corresponding `_lr_fs.html`. You can also add new functions to be called by `listFrameHTML()`. The new content will then be inserted in the mail application. If the feature to be added involves more technical aspects that are not documented here, contact Sun Professional Service for assistance.

Understanding the File Layout in Mail

In mail, the static files are HTML and JavaScript files. These files have `.html` and `.js` extensions. These files are transmitted to the browser and used to retrieve dynamic contents from the server using a proprietary HTTP protocol. The mail server provides user interfaces for the stand-alone Messenger Express and the integrated mail component of Communication Express. The JavaScript files are shared by both these interfaces, and in general provide functionalities for the application.

Files that end with `_fs_lr.html` are used exclusively in the integrated mail component of the Communication Express. This section provides a basic overview on the various files used in the integrated mail component of Communication Express.

The CSS files that are used in mail are:

- `mail_css_ie5win.css`
- `mail_css_ns6up.css`
- `master-style.css`
- `master-style_ie5up.css`
- `master-style_ns4sol.css`
- `master-style_ns4win.css`
- `master-style_ns6up.css`

[Table 4-1](#) shows the different JavaScript files that are used in Mail.

Table 4-1 JavaScript files that are used in Mail

JavaScript file Name	Application
<code>browser.js</code>	Detects the browser to select which <code>.css</code> file should be used
<code>mailui.js</code>	Maintains the common UI codes shared by messaging listing and Search message result listing
<code>setdomain.js</code>	Checks for Domain security
<code>main.js</code>	Maintains the Initialization and globally shared functions
<code>util.js</code>	Utility functions
<code>add dtree.js</code>	UI Codes for folder tree

The frame work related files are,

- `en/frame.html`
- `en/mail.html`, which is the first file loaded by the browser.

The Mail page is configured to display the Mail page. It creates a number of frames, including several invisible frames. These invisible frames are used to pull the dynamic contents as JavaScript objects from the server. Once downloaded, the hidden frame can invoke the specific callback function in `frameset HTML (_lr_fs.html)` to render the updated dynamic contents.

The Mail component can be perceived as a dynamic client-side HTML engine. There are not many standard HTML statements in `_lr_fs.html` files. It is the `ListFrameHTML()` function in `_lr_fs.html` files which is responsible for the HTML output. This action takes place when `ListFrameHTML()` accesses the JavaScript objects from the hidden frames such as `cfgFrame`, `msgFrame`, and `mboxFrame` to generate the dynamic HTML page.

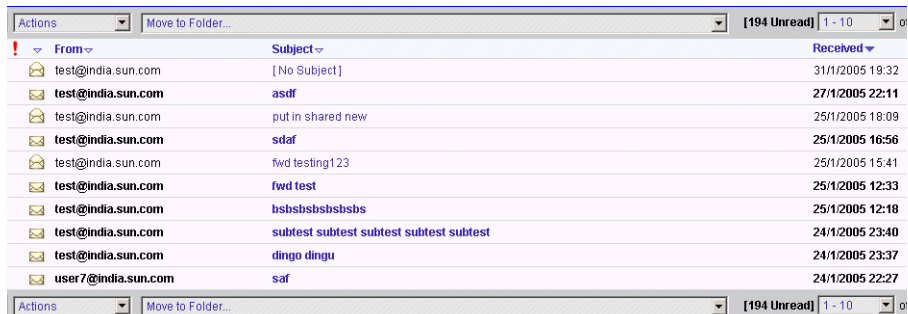
Understanding Basic Interfaces in Mail

This section provides information on the various Mail screens in Communications Express, including:

- Message Listing screen
- Message screen
- Folder Management screen
- Composition window
- Message Search window

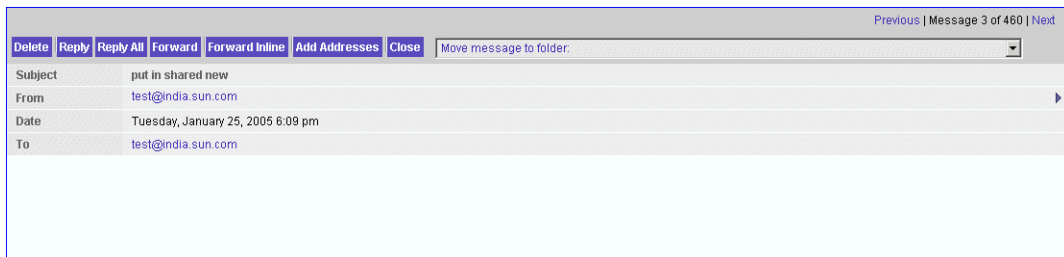
Inbox Screen

The Inbox screen, shown in [Figure 4-1](#), 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 4-1 Communications Express Inbox Screen

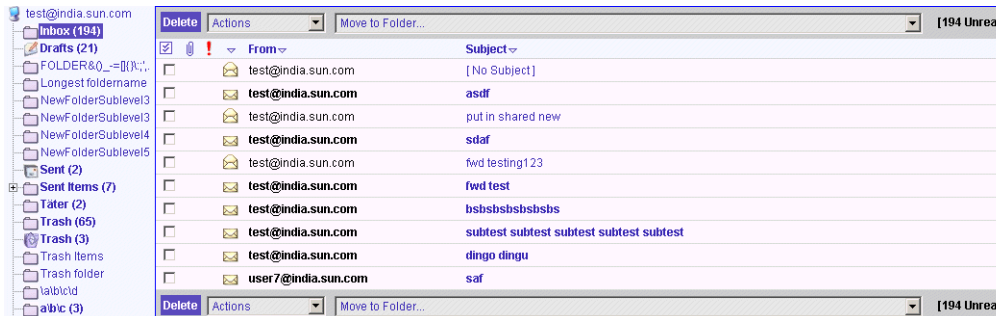
Message Screen

The Message screen, shown in [Figure 4-2](#), displays the message selected from the Inbox screen. The Message screen gives the option of replying to the sender, 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 4-2 Communications Express Message Screen

Folders Screen

The Folders screen, shown in [Figure 4-3](#), 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 4-3 Folders Screen

Composition Window

The Composition window, shown in [Figure 4-4](#), 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 if you are using Internet Explorer and this feature is not supported in Netscape Navigator. In the Composition window you can also check the spelling. The Composition window also enables you to set the mail priority or request for a return receipt.

Figure 4-4 Communications Express Composition Window

Message Search Window

The Communications Express Message Search window, shown in [Figure 4-5](#), 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 4-5 Message Search Window

Customizing User Interface Features in Mail

This chapter describes how to customize user interface features of the Mail component in Communications Express.

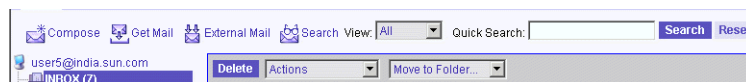
This chapter contains the following sections:

- [Customizing the Mailbox Tool Bar](#)
- [Customizing Message Listing](#)
- [Customizing the Message Display Window](#)
- [Customizing the Message Composition Window](#)
- [Aligning the User Interface from Right to Left\(For Arabic only\)](#)

Customizing the Mailbox Tool Bar

This section describes how to modify the mailbox tool bar shown in [Figure 5-1](#).

Figure 5-1 Communications Express Mailbox Tool Bar



You can modify the following on the 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

Customizing the Mailbox Tool Bar

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

- To customize the layout within the tool bar and the associated graphics, edit the `getToolbar()` function in the `mbox_fs_lr.html` file.
- To customize the text associated with the graphics in the Tool Bar, edit the `il8n[]` values for get mail, compose, new search, file selected message, delete, undelete, and expunge in the `lang/il8n.js` file

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

- Get Mail: `main.refreshMbox()`
- Compose: `main.compose("new")`
- Search: `srch()`

Example—Mailbox Tool Bar Modifications

Code Example 5-1 shows the necessary edits to be made in the file `mbox_fs_lr.html` to display “Search” as the first tool and changes the text of “Get Mail” tool to “Get Messages.”

Code Example 5-1 Altered Tool Bar Layout (`mbox_fs_lr.html`)

```
function getToolbar() {
    if (!main.loaded) return '';
    return nWMtoolbar(il8n['mbox search'], 'srch()',
        'search', 'imx/LrlSearchMsg_wo_1.gif', 'imx/LrlSearchMsg_1.gif') +
        nWMtoolbar(il8n['compose'], 'main.compose(\'new\')',
        'compose', 'imx/LrlNewMsg_wo_1.gif', 'imx/LrlNewMsg_1.gif') +
        nWMtoolbar(il8n['get mail'], 'main.check_mail = 1;
    main.displaySpecialMbox(\'Inbox\')',
        'getmail', 'imx/LrlGetMail_wo_1.gif', 'imx/LrlGetMail_1.gif')+
        nWMtoolbar(il8n['collect long'], 'main.collect()',
        'collect', 'imx/LrlColExtMail_wo_1.gif',
        'imx/LrlColExtMail_1.gif')+
}
```

Code Example 5-2 shows the necessary changes to be made in file `en/il8n.js` (text)

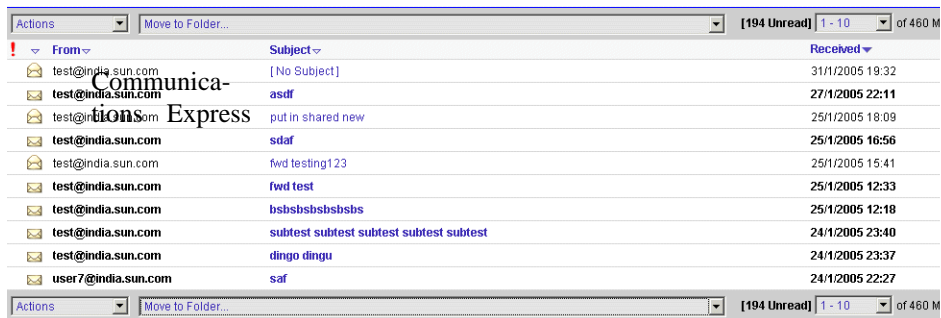
Code Example 5-2 Altering Tool Bar Text (en/i18n.js)

```
i18n['get mail'] = 'Get Messages'
```

Customizing Message Listing

This section describes how to modify the Message List window shown in [Figure 5-2](#).

Figure 5-2 Communications Express Message List Window



You can modify the following in the 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_lr.html` 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, the `listFrameHTML()` function links the “Collect External Mail” hyperlink to `collect()` in `main.js`. An example of such a linking is shown in - Message List Window Modifications

Example - Message Listing Modifications

To display the most recently received mails first, and change the text on “Collect External Mail” button to “Get Messages From Another Server”, follow the code example provided in [Code Example 5-3](#). This shows the necessary changes to be made in file `en/i18n.js`.

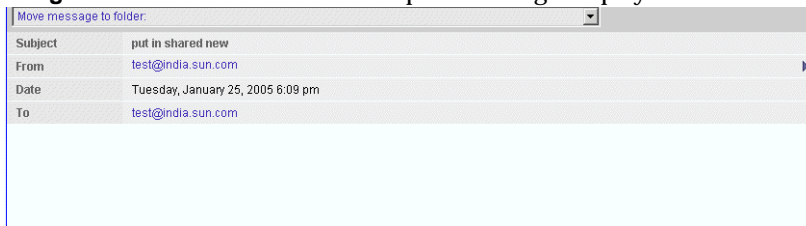
Code Example 5-3 Altering List Window Text (`en/i18n.js`)

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

Customizing the Message Display Window

This section describes how to modify the Message Display window shown in [Figure 5-3](#).

Figure 5-3 Communications Express Message Display Window



You can modify the following in the Message Display window:

- Change the message display
- Alter the layout of the window
- Change the text of the fields
- Display user-defined header 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_lr.html` file.
- To customize the default text, edit the `i18n[]` values under Message Headers and Message in the `lang/i18n.js` file.

Modify the Message Display Window to Display User Defined Header Fields

This section describes how to add and display user-defined header fields in the Message Display window.

To Display User Defined Header Fields

Edit the `listFrameHTML(doc)` function in the `msg_fs_lr.html` file.

Example—Customizing the Message Display Window to Display User Defined Header Fields

This example shows how to display the user defined field *X-document-id* in the Message Display Window.

[Code Example 5-4](#) shows the edits made to the function `listFrameHTML()` in the `msg_fs_lr.html` file.

Code Example 5-4 Changes made to the `msg_fs_lr.html` file

```
function listFrameHTML(doc) {
  ...
  var hdrstr = getHeaderStr(main.msgFrame.hdr[0], 'X-document-id')
  if(hdrstr != '') {
    s += <tr><td nowrap align=right valign=top width=5%' + main.base_line + '
    bgcolor=' + main.chrome2 + '>' + main.font() + html('X-document-id') + nbsp
    + '</td>\n<td ' + main.cellBgString + '>' + extra + main.font() + hdrstr +
    '</td></tr>\n'
  }
  ...
}
```

Customizing the Message Tool Bar

This section describes how to customize the message tool bar

Figure 5-4 Communications Express Message Tool Bar



You can modify the following in the message tool bar:

- Change the layout of the toolbar 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 within the tool bar, edit the `actionBar()` function in the `msg_fs_lr.html` file.
- To customize the texts associated with the graphics in the Tool Bar, edit the `i18n[]` values for `compose`, `reply`, `reply all`, `forward`, `delete`, `undelete`, `previous`, and `next` in the `lang/i18n.js` file.

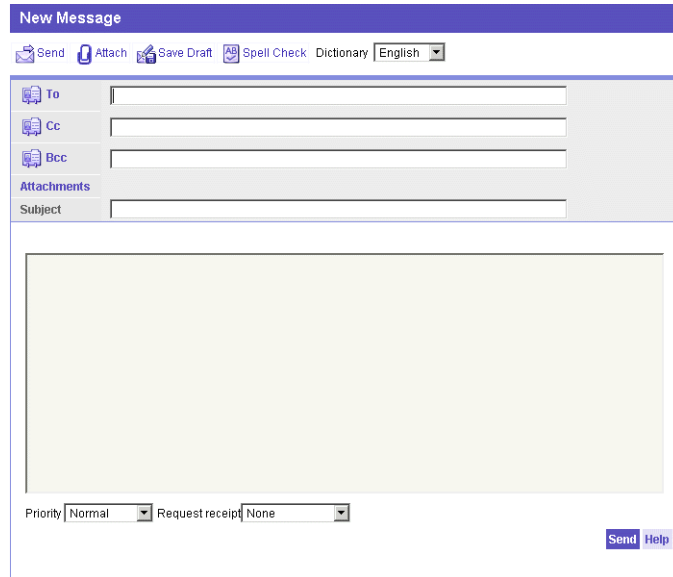
[Code Example 5-5](#) shows the necessary changes to be made in the file `en/i18n.js`.

Code Example 5-5 Altering Message Tool Bar Text (`en/i18n.js`)

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

Customizing the Message Composition Window

This section describes how to modify the Message Composition window

Figure 5-5 Communications Express Message Composition Window

You can modify the following in the 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_lr.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 `msg_svr_base/html/imx` directory.

Functionally, `compFrameHTML()` in `comp_fs_lr.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_lr.html` generates the “To”, “Cc”, and “Bcc” control area by calling `i18n_compose_controls()` in `lang/i18n.js`.

[Code Example 5-6](#) shows the necessary changes to be made in the file `en/i18n.js` for changing the text “Recipients” to “Send to”.

Code Example 5-6 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 5-7](#) shows how to edit the `main.js` file to enable emoticons.

Code Example 5-7 Altering Composition Window to Enable Emoticons

```
var iconHREF = 'msg_svr_base/imx/'
```

[Code Example 5-8](#) shows how to edit `main.js` file to disable emoticons.

Code Example 5-8 Altering Composition Window to Disable Emoticons

```
var iconHREF = ''
```


Aligning the User Interface from Right to Left(For Arabic only)

By default, the Menu tabs are aligned from left to right. If you have selected Arabic as the preferred language, then you need to customize Communications Express to align the Menu tabs from right to left.

To Align the User Interface From Right to Left

- To align the User Interface from right to left you need to remove a line from the `ar/i18n.js` file.

[Code Example 5-9](#) shows the line that needs to be added to the `ar/i18n.js` file.

Code Example 5-9 Aligning the User Interface from right to left

```
....  
i18n['dir'] = 'ltr'  
....
```

Aligning the User Interface from Right to Left(For Arabic only)

Customizing Advanced Features in Mail

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

This chapter contains the following sections:

- [Understanding Advanced Customization](#)
- [List of files to use for customizing UI in Mail](#)
- [Understanding Shared Folders](#)
- [Customizing the Default LDAP Attributes for Users](#)

Understanding Advanced Customization

In addition to the features mentioned in [Chapter 5, “Customizing User Interface Features in Mail”](#), many other features are fully customizable. However, to take advantage of these features, an in-depth knowledge of JavaScript is required. You might also face migration problems when attempting to reconfigure JavaScript files.

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

List of files to use for customizing UI in Mail

Table 6-1 lists the UI related FrameSet files of Communications Express user interface that are fully customizable.

Table 6-1 Communications Express User Interface Customizable Features

Features	Files
Adds attachment for upload (popup window)	attach_fs_lr.html
Collects pop email (popup window)	collect_fs_lr.html
New message composition	comp_fs_lr.html
Folder viewing and management	fldr_fs_lr.html
Message listing for a selected folder	mbox_fs_lr.html
Message viewing	msg_fs_lr.html
Receipt notification (popup window)	receipt_fs_lr.html
Search message (popup window)	searchmsg_fs_lr.html
Sets folder sharing permission (popup window)	setpermission_fs_lr.html
Search Results	srchresults_fs_lr.html
Subscribe to	subscribe_fs_lr.html

Attachments

You can modify the following attachments options in mail.

- Browse button
- Attach, Cancel, Help buttons

HTML File Mapping

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

Collect Mail from Another Server

You can modify the following labels that appear 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_lr.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_lr.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_lr.html`.

HTML File Mapping

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

Understanding Shared Folders

A shared folder is a folder in which you can store messages that can be accessed by other users. Communications Express manages access to the shared folders by granting others access to the shared folders.

Sharing folders is a two-step process that involves sharing and subscribing:

1. A user shares a folder, specifying who has permissions to the folder.
2. The users who were given permissions to that folder then subscribe to it.

For more information on Shared Folders, see *Sun Java System Communications Express Online Help*, that can be accessed by clicking the Help on Sun Java System Communications Express.

Customizing the Default LDAP Attributes for Users

The Communications Express Server loads a default set of LDAP attributes for a user at the start of a session. These attributes are provided in [Table 6-2](#).

Table 6-2 Customizable LDAP attributes

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

For more information on the attributes required or allowed by LDAP object classes for Communications Express, see Chapter 3 Attributes in the *Sun One Messaging and Collaboration Schema Reference Manual* at:

<http://docs.sun.com/db/prod/slmsgsrv#hic>

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 Communications Express. Possible values for this attribute are “regular” and “vip.”

NOTE Before adding custom attributes to the directory, ensure that the directory schema supports these new attributes. The global set of schema for Directory Server can be found in the entry named `cn=schema`.

For more information on extending directory schema, see *Sun ONE Directory Server Deployment Guide* at:

<http://docs.sun.com/db/prod/s1dirsrv#hic>

After extending the directory schema, you can add the new attribute to a User or a group entry. To add custom attribute to as User entry you need to modify the entry in `servletsconf` section using the `ldapmodify` command. For more information on `ldapmodify` command, see *Sun ONE Directory Server Configuration, Command, and File Reference* at:

<http://docs.sun.com/db/prod/s1dirsrv#hic>

Depending on the type of user (that is, the value of the *myuserclass LDAP* attribute), different advertisements are presented to the user when they log into Communications Express (Communications 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 user wants to modify an attribute using the Communications Express code, the attribute needs to be marked read-write by appending the suffix.

The following example assumes the user 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:

Code Example 6-1 Display of banner advertisements depending on the user class

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


Customizing General Features in Address Book

The Address Book contains many customizable and localizable items that can be changed to create a different look and feel. These customizable items include icons, style sheets, and labels. You can change the position of certain items and also customize captions, headings, button labels, alert and inline messages.

The rendering of Address Book UI is handled by XML/XSL files that contain XSL tags, static HTML and JavaScripts. The XSL and JavaScript code are used to display dynamic data and perform actions or form handlings. The XSL code in most of the pages is structured into templates which helps make a page modular for various components. All these files are located under the `<uwcinstall>/WEB-INF/ui/html/abs` directory. This chapter contains information on customizing the following in Address Book.

- [Customizing Address Book Icons and Labels](#)
- [Customizing Style Sheets](#)
- [Customizing the Main Search Page](#)
- [Customizing New/Edit Contact Window](#)
- [Customizing New Group or Edit Group Window](#)
- [Customizing Printable Window](#)
- [Customizing the Import/Export Address Book Window](#)

Customizing Address Book Icons and Labels

Customizing Icons

All the icons that are used for display in Address Book are defined under two files namely,

- `search-images.xml` - This file contains most of the icon references that are used in the search result table.
- `commonimages.xml` - This file contains common icon file references.

All Address Book related images are located under `<uwcinstall>/absimx`. `Search-images.xml` and `commonimages.xml` refer to images in this directory. To customize, you can add or replace the required images in this directory and then change the corresponding references in these files.

These are two examples on how you can modify the icons.

Example 1 - To change the icon for Printable as displayed in the toolbar of main page, you need to do the following.

Old Icon: Printable (File name: - `LrlPrintable_1_wo.gif`)

New Icon: New Printable (File name - `ico_print.gif`)

In `commonimages.xml`, the reference is defined as,

```
<xsl:variable name="print_page.gif" select="'../absimx/LrlPrintable_1_wo.gif'"/>
```

To change the icon file reference,

Add `ico_print.gif` file into `<uwcinstall>/absimx`.

Change the reference in `xsl:variable` tag for `print_page.gif` from

```
'../absimx/LrlPrintable_1_wo.gif' to '../absimx/ico_print.gif '
```

Example 2 - To add a banner image called `AddressBook_banner.gif` you need to do the following.

- Add `AddressBook_banner.gif` file into `<uwcinstall>/absimx`.
- Add a reference for the image in `commonimages.xml` by adding the following code.

```
<xsl:variable name="AddressBook_banner.gif"  
select="'../absimx/AddressBook_banner.gif'"/>
```

Then add the image in the required position in the desired `.xml` file as

```

```

Customizing Labels

All captions and displayable static items that are displayed in the Address Book can be customized. All the labels are defined in the `dictionary-<lang>.xml` file which is located under `<uwcinstall>/WEB-INF/ui/html/abs` directory. This XML file contains definitions in the following form

```
<word key="_Message">Message Actual Text</word>
```

The key is specified in the XSL file which is used to obtain the value of the `<word>` XML node. This value is eventually displayed during the rendering of XSL into HTML. In the XSL file, the key is specified in the `<xsl:text>` tag.

To customize a particular caption or a static label, change the text corresponding to desired key. To add new text,

- Add new `<word>` tag with the appropriate key and the prefix `'_'`
- Add the `<xsl:text>` tag with the key as the tag value in the XSL file.

The following example changes the label for 'New Contact' link on the Main page toolbar to 'Add Contact'. In the example 'en' is assumed to be the language in which the label is displayed.

In `search.xml`, the label is coded as following:

Code Example 7-1 Customization of the label

```
<a class="ToolLbl" href="javascript:void(0)">
  <xsl:attribute name="onClick">
    <xsl:text>javascript:openWinAutoHeight('addcontact-main.xml?bookid=</xsl:te
xt>
  <xsl:value-of select="$selectedBook/entry/@entryID"/>
    <xsl:text>', 'NewContact',
  'scrollbars=yes,resizable=yes,width=700');</xsl:text>
  </xsl:attribute>
  <xsl:text>_New Contact</xsl:text>      -----> Label Definition
</a>
```

[Code Example 7-2](#) shows the code before modifying the default definition in `dictionary-en.xml`

Code Example 7-2 Code before Modification of the Default Definition in Dictionary -en-xml

```
<word key="_View Calendar...">View Calendar...</word>
```

[Table 7-3](#) shows the code before modifying the default definition in dictionary -en-xml

Code Example 7-3 Customized Default Definition in Dictionary -en-xml

```
<word key="_View Calendar...">View Schedule...</word>
```

Customizing Style Sheets

In Address Book, style sheets play a significant role in the manner in which the UI is rendered in terms of look and feel as many items that are displayed in Address Book are associated with style sheets. Hence, the look and feel of Address Book pages and pop ups can be customized by modifying these style sheets.

These style sheets are defined in a CSS file which is included using the *<link>* HTML tag in the HTML *<head>* section of XSL files. These are referenced by assigning a 'class' attribute of HTML tags.

[Code Example 7-4](#) shows the class attribute using HTML tags.

Code Example 7-4 Class attribute of HTML tags

```
<td class="TdActTdLst">.....</td>
```

[Code Example 7-5](#) shows the class attribute using XSL tags.

Code Example 7-5 Class attribute of XSL tags

```
<xsl:variable name="css-class" select="'TdActTdLst'"/>
....
<td>
  <xsl:attribute name="class">
    <xsl:value-of select="$css-class"/>
  </xsl:attribute>
  ...
</td>
```

The Address Book has two separate CSS files for IE and Netscape. These files exist under `<uwcinstall>/absimx` and are named as,

- `ab_css_ie5win.css` for IE
- `ab_css_ns6up.css` for Netscape

What can be Customized in Style Sheets

- You can change the definition of an existing style sheet. In this case, you need not change the class attribute reference.
- You can add new style sheet definitions and change the class attribute reference for the desired HTML tag in XSL files.

1. Example of changing the definitions of an existing style sheet:

The definition of a primary button display is 'Btn1' which has the following definition in the CSS file.

[Code Example 7-6](#) shows the code before changing the definition of a primary button display 'Btn1'

Code Example 7-6 Before changing the definition of a primary button display 'Btn1'

```
.Btn1 {background:#594fbf;color:#FFF;font-weight:bold;padding:2px
0px;margin:0px 0px 1px 0px;border:0px none #000; }
```

[Code Example 7-7](#) shows how the class is used in the XSL file for the 'Close' button.

Code Example 7-7 Class used in the XSL file for the 'Close' button

```
<input class="Btn1" id="close" onblur="if (this.disabled==0)
this.className='Btn1'" onmouseover="if (this.disabled==0)
this.className='Btn1Hov'" onfocus="if (this.disabled==0)
this.className='Btn1Hov'" onclick="parent.close()" tabIndex="1"
onmouseout="if (this.disabled==0) this.className='Btn1'" type="button"
name="close" value="Close">
```

[Code Example 7-8](#) shows the code after changing the definition of a primary button display.

Code Example 7-8 After changing the definition of a primary button display 'Btn1'

```
.Btn1 {background:#ffff;color:#000;font-weight:bold;padding:2px 0px;margin:0px 1px 1px 0px;border:0px none #000; }
```

2. Example of adding new style sheet definitions and change the class attribute reference for the desired HTML tag in XSL files.

You may add a different class definition altogether in the CSS file. In this case, the newly defined class needs to be assigned to the XSL file.

```
.EMPBtn { font-family: Arial, Helvetica, sans-serif; text-decoration none; font-weight:bold; color: #594fbf; }
```

In XSL file, the corresponding change would be

Code Example 7-9 After changing class attribute reference for the desired HTML tag

```
<input class="EMPBtn" id="close"onblur="if(this.disabled==0)
this.className='Btn1'"onmouseover="if(this.disabled==0)
this.className='Btn1Hov'"onfocus="if(this.disabled==0)
this.className='Btn1Hov'"onclick="parent.close()"tabIndex="1" onmouseout="if
(this.disabled==0)
this.className='Btn1'"type="button"name="close"value="Close">
```

Customizing the Main Search Page

The search page can be divided into following sections

- Tabs
- Toolbar Pane
- Search Pane
- Action Row
- Search Result Table
- Quick Add Pane

Each of these sections are defined in templates which exist either under search.xml or search-template.xml. The search-template.xml file contains commonly used templates shared by the Main Search Page, Search Address Book popup and Add Addresses popup.

These sections can be reorganized by modifying their respective positions.

Table 7-1 shows the commonly used templates for search-template.xml and search.xml

Table 7-1 Commonly used templates in search-template.xml and search.xml

Sections	Template Name	File
Tabs	search-template-bookbar	search-template.xml
Toolbar	search-template-toolbar	search.xml
Search Pane	search-template-searchbar	search-template.xml
Action Row	search-template-actionbar	search.xml
Search Result Table	search-template-searchresult	search-template.xml
Quick Add	quick-add	search.xml

Search Result Table

This templates that refer to Search Result Table are,

- **commonattributes-list.xml**. The templates defined in this file are used in the display of header captions in result table in a particular layout.
- **entries-list.xml**. The templates defined in this file are used to display the actual data such as email, displayname, web address etc and so on in a particular format.

To customize the Search Main Page, Search Address Book popup, Add Addresses and Printable views, you need to change the layout information in these templates.

Customizing New/Edit Contact Window

The files pertaining to the new contact or edit contact window are,

- **New contact** - addcontact.xml
- **Edit contact** - editcontact.xml

Common code shared among new and edit window can be accessed from `contact-common-editcontact.xml`

Table 7-2 shows the templates that are defined for the sections in New Contact or Edit Contact Window

Table 7-2 Templates applicable for New/Edit Contact Window

Sections	Template Name	File
Skip Links	-	addcontact.xml and editcontact.xml
Name and Company	nameAndCompany	common-editcontact.xml
Phone	phone	common-editcontact.xml
Email	email	common-editcontact.xml
Addresses	address	common-editcontact.xml
IM Nicknames	IM	common-editcontact.xml
Web Addresses	onlineInfos	common-editcontact.xml
Notes	notes	common-editcontact.xml

You can customize the window by changing labels or reordering the listed sections in any desired order.

In `addcontact.xml` the '*edit-abperson*' template calls the templates.

Code Example 7-10 shows the code that displays Address before Online information.

Code Example 7-10 Code that displays Address displayed before online information

```
<xsl:call-template name="address">
  <xsl:with-param name="abperson" select="$abperson"/>
</xsl:call-template>
```


Code Example 7-10 Code that displays Address displayed before online information

```
<xsl:call-template name="address">
  <xsl:call-template name="onlineInfos">
    <xsl:with-param name="abperson" select="$abperson" />
  </xsl:call-template>
</xsl:call-template>
```

Code Example 7-11 shows how to display Online information before Address.

Code Example 7-11 Code that displays Online information before Address

```
<xsl:call-template name="onlineInfos">
  <xsl:with-param name="abperson" select="$abperson" />
</xsl:call-template>

<xsl:call-template name="address">
  <xsl:with-param name="abperson" select="$abperson" />
</xsl:call-template>
```

Customizing Phone

Address Book allows you to set five types of phones which are defined in phoneEmailAndIM. The five definitions of 'phone' template are called and the priority value is passed to them when the values are displayed in the drop-down list. You can change their respective priority orders.

Figure 7-1 Customizing Phone

Phone

Phone (1): Work <input type="text"/>	Phone (2): Home <input type="text"/>
Phone (3): Mobile <input type="text"/>	Phone (4): Pager <input type="text"/>
Phone (5): Fax <input type="text"/>	

[Back to Top](#)

Code Example 7-12 shows the default code where Work is displayed as priority 1 and home is displayed as priority 2.

Code Example 7-12 Default code wherein Work phone is displayed as priority 1 and home phone as priority 2

```
<option value="work">
  <xsl:choose>
    <xsl:when test="$abperson">
      <xsl:choose>
        <xsl:when test="$abperson/phone[@priority = $priority]">
          <xsl:if test="$abperson/phone[@priority = $priority]/@type
= 'work'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:when>
        <xsl:otherwise>
          <xsl:if test="$priority = '1'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:when>
    <xsl:otherwise>
      <xsl:if test="$priority = '1'">
        <xsl:attribute name="selected"/>
      </xsl:if>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text>_Work</xsl:text>
</option>
<option value="home">
  <xsl:choose>
    <xsl:when test="$abperson">
      <xsl:choose>
        <xsl:when test="$abperson/phone[@priority = $priority]">
          <xsl:if test="$abperson/phone[@priority = $priority]/@type
= 'home'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:when>
        <xsl:otherwise>
          <xsl:if test="$priority = '2'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:when>
    <xsl:otherwise>
      <xsl:if test="$priority = '2'">
        <xsl:attribute name="selected"/>
      </xsl:if>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text>_Home</xsl:text>
</option>
```

Code Example 7-13 shows how to change the code so that it displays Home phone as priority 1 and Work phone as priority 2.

Code Example 7-13 Code that displays Home as priority 1 and Work as priority 2

```
<option value="work">
  <xsl:choose>
    <xsl:when test="$abperson">
      <xsl:choose>
        <xsl:when test="$abperson/phone[@priority = $priority]">
          <xsl:if test="$abperson/phone[@priority = $priority]/@type
= 'work'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:when>
        <xsl:otherwise>
          <xsl:if test="$priority = '2'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:when>
    <xsl:otherwise>
      <xsl:if test="$priority = '2'">
        <xsl:attribute name="selected"/>
      </xsl:if>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text>_Work</xsl:text>
</option>
<option value="home">
  <xsl:choose>
    <xsl:when test="$abperson">
      <xsl:choose>
        <xsl:when test="$abperson/phone[@priority = $priority]">
          <xsl:if test="$abperson/phone[@priority = $priority]/@type
= 'home'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:when>
        <xsl:otherwise>
          <xsl:if test="$priority = '1'">
            <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:when>
    <xsl:otherwise>
      <xsl:if test="$priority = '1'">
        <xsl:attribute name="selected"/>
      </xsl:if>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text>_Home</xsl:text>
</option>
```

Customizing E-mail

Address Book allows to set three types of E-mail which are defined in `phoneEmailAndIM`. The three definitions of 'email' templates are called and the priority values are passed to them. You can change their priority orders.

Figure 7-2 Customizing E-mail

Email

Email (1): Work <input type="text"/>	Email (3): Pager <input type="text"/>
Email (2): Home <input type="text"/>	

[Back to Top](#)

Customizing Dates

Address Book allows you to set Birthday and Anniversary dates. These two dates are defined in the 'importantDates' template. This template passes all relevant information to the 'date' template.

You may customize the labels, change layout such as size of fields for the dates or their respective positions.

Code Example 7-14 Default code that shows horizontally aligned fields and displays Birthday as the first field and Anniversary as the second

```
<tr>
  <xsl:call-template name="date">
    <xsl:with-param name="abperson" select="$abperson"/>
    <xsl:with-param name="type" select="'birthday'"/>
    <xsl:with-param name="index" select="'1'"/>
    <xsl:with-param name="dateFormat" select="$dateFormat"/>
    <xsl:with-param name="width" select="40"/>
    <xsl:with-param name="fieldName" >
      <xsl:value-of select="$fieldPrefix" />
      <xsl:text>dateOfBirth</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="label">
      <xsl:text>_Birthday</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="tabindex" select="'52'"/>
  </xsl:call-template>

  <xsl:call-template name="date">
    <xsl:with-param name="abperson" select="$abperson"/>
    <xsl:with-param name="type" select="'anniversary'"/>
    <xsl:with-param name="index" select="'2'"/>
    <xsl:with-param name="dateFormat" select="$dateFormat"/>
```

Code Example 7-14 Default code that shows horizontally aligned fields and displays Birthday as the first field and Anniversary as the second

```
<tr>
  <xsl:with-param name="width" select="65"/>
  <xsl:with-param name="fieldName" >
    <xsl:value-of select="$fieldPrefix" />
    <xsl:text>anniversary</xsl:text>
  </xsl:with-param>
  <xsl:with-param name="label">
    <xsl:text>_Anniversary</xsl:text>
  </xsl:with-param>
  <xsl:with-param name="tabindex" select="'55'"/>
</xsl:call-template>
</tr>
```

Code Example 7-15 shows how to display the fields in a vertical layout and also rearranges Anniversary as the first date and Birthday as second

Code Example 7-15 Code that displays the field in a vertical layout and also rearranges Anniversary as the first date and Birthday as second

```
<tr>
  <xsl:call-template name="date">
    <xsl:with-param name="abperson" select="$abperson"/>
    <xsl:with-param name="type" select="'anniversary'"/>
    <xsl:with-param name="index" select="'1'"/>
    <xsl:with-param name="dateFormat" select="$dateFormat"/>
    <xsl:with-param name="width" select="65"/>
    <xsl:with-param name="fieldName" >
      <xsl:value-of select="$fieldPrefix" />
      <xsl:text>anniversary</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="label">
      <xsl:text>_Anniversary</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="tabindex" select="'52'"/>
  </xsl:call-template>
</tr>
<tr>
  <xsl:call-template name="date">
    <xsl:with-param name="abperson" select="$abperson"/>
    <xsl:with-param name="type" select="'birthday'"/>
    <xsl:with-param name="index" select="'2'"/>
    <xsl:with-param name="dateFormat" select="$dateFormat"/>
    <xsl:with-param name="width" select="40"/>
    <xsl:with-param name="fieldName" >
      <xsl:value-of select="$fieldPrefix" />
      <xsl:text>dateOfBirth</xsl:text>
    </xsl:with-param>
    <xsl:with-param name="label">
      <xsl:text>_Birthday</xsl:text>
    </xsl:with-param>
  </xsl:call-template>
</tr>
```

Code Example 7-15 Code that displays the field in a vertical layout and also rearranges Anniversary as the first date and Birthday as second

```
<tr>
  </xsl:with-param>
  <xsl:with-param name="tabindex" select="'55'"/>
</xsl:call-template>
```

Customizing Web Addresses

Address Book allows you to set four web addresses for work, home, calendar and freebusy. These addresses are defined in the *'onlineInfos'* template. The *'onlineInfos'* template calls the following:

- *'weburl'* that passes the priority for ordered display
- *'calendar'* with *type="calendar"* for calendar url and *type="freebusy"* for *freebusyurl"*.

You may change the order in which these addresses similar to the examples illustrated above.

Customizing IM Nicknames

Address Book allows you to set two IM nicknames. The list of IM services that is supported by Address Book are listed in the drop-down menus. These two IM options are defined in *'phoneEmailAndIM'* template which calls the *'im'* template passing the relevant information.

- You may change the order of IMs by changing their priorities as illustrated in previous examples.
- You may also change the default selected services. The default services available in the drop down list are *'SunONE'* and *'AIM'*. The code checks for the passed priority and marks the appropriate option in the drop down as selected. You may shuffle the priority value in the code to select a different service. This code exists in the *'im'* template.

Code Example 7-16 shows the default code that displays SunONE as the default selected service

Code Example 7-16 Default code that displays SunONE as the default service

```
<option value="SunONE">
  <xsl:choose>
    <xsl:when test="$abperson and $abperson/im[@priority=$priority]">
      <xsl:if test="$abperson/im[@priority=$priority]/@service =
' SunONE' ">
```

Code Example 7-16 Default code that displays SunONE as the default service

```
<option value="SunONE">
  <xsl:attribute name="selected"/>
  </xsl:if>
</xsl:when>
<xsl:otherwise>
  <xsl:if test="$priority= '1'"> ---- This value is passed by
'phoneEmailAndIM' template to have this option default selected
  <xsl:attribute name="selected"/>
  </xsl:if>
  </xsl:otherwise>
</xsl:choose>
<xsl:text>iPlanet</xsl:text>
</option>
.....
<option value="Yahoo">
  <xsl:choose>
    <xsl:when test="$abperson and $abperson/im[@priority=$priority]">
      <xsl:if test="$abperson/im[@priority=$priority]/@service =
'Yahoo'">
        <xsl:attribute name="selected"/>
        </xsl:if>
      </xsl:when>
      <xsl:otherwise>
        <xsl:if test="$priority = '3'">
          <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:otherwise>
      </xsl:choose>
      <xsl:text>Yahoo</xsl:text>
    </option>
```

Code Example 7-17 shows the modified code that displays Yahoo as the default Service

Code Example 7-17 Modified code that displays Yahoo as the default Service

```
<option value="SunONE">
  <xsl:choose>
    <xsl:when test="$abperson and $abperson/im[@priority=$priority]">
      <xsl:if test="$abperson/im[@priority=$priority]/@service =
'SunONE'">
        <xsl:attribute name="selected"/>
        </xsl:if>
      </xsl:when>
      <xsl:otherwise>
        <xsl:if test="$priority= '3'"> ---- This value is passed by
'phoneEmailAndIM' template to have this option default selected
          <xsl:attribute name="selected"/>
          </xsl:if>
        </xsl:otherwise>
      </xsl:choose>
```

Code Example 7-17 Modified code that displays Yahoo as the default Service

```
<option value="SunONE">
    <xsl:text>iPlanet</xsl:text>
</option>
.....
<option value="Yahoo">
    <xsl:choose>
        <xsl:when test="$abperson and $abperson/im[@priority=$priority]">
            <xsl:if test="$abperson/im[@priority=$priority]/@service =
'Yahoo'">
                <xsl:attribute name="selected"/>
            </xsl:if>
        </xsl:when>
        <xsl:otherwise>
            <xsl:if test="$priority = '1'">
                <xsl:attribute name="selected"/>
            </xsl:if>
        </xsl:otherwise>
    </xsl:choose>
    <xsl:text>Yahoo</xsl:text>
</option>
```

Customizing View Contact Window

The View contact window displays the sections in the same order as defined in new/edit contact window. You may customize this window by modifying labels, reordering sections, modifying layout, alignment in the page etc. The file to be changed for this window is 'viewcontact.xml'

Customizing New Group or Edit Group Window

The files pertaining to the New Group or Edit Group window are,

- New Group - addgroup.xml
- Edit Group - editgroup.xml

The common code shared among new and edit contact are stored in common-editgroup.xml

Table 7-3 lists the templates defined for the various sections in New Group or Edit group:

Code Example 7-18 Code to be used for adding the E-mail address which is supported for a group

```
<tr>
  <td width="11%">&nbsp;</td>
  <td width="6%">&nbsp;</td>
</tr>
```

Customizing the View Group Window

The View Group window displays the sections in the same order as defined in New Group or Edit Group window. You may customize this window by modifying labels, reordering sections, modifying layout, their alignment in the page etc. The file to be changed for this window is 'viewgroup.xml'

Customizing Printable Window

The file for this window is 'abprnlist.xml'. The window displays each contact/group in card format. You can customize the page by changing the layout or changing the order of fields and stylesheet items displayed in the page.

Customizing the Import/Export Address Book Window

The file that is used for modifying the import/export window is 'importexport.xml'. The page displays Import and Export sections.

Figure 7-3 Customizing the Import/Export Address Book Window

Import and Export Address Book

*Indicates a required field

Import

* Import from File:

Import Format:

Export

Export Format:

You can customize the page for

- Labels
- Swapping the Import and Export sections
- Change the format options displayed in drop-downs.
- Layout

For example, let us consider modifying the listed format options. The format options listed are,

- Netscape Communication Address Book (LDIF)
- Microsoft Outlook CSV
- SunONE Address Book CSV
- vCard

[Code Example 7-19](#) shows the code before modifying the options list

Code Example 7-19 Code before modifying the options list

```
<td class="PropInput" width="80%">  
  
<select name="importformat" tabindex="2">  
  <option value="ldif"><xsl:text>_Netscape Ldif</xsl:text></option>  
  <option value="csvus"><xsl:text>_Outlook CSV</xsl:text></option>  
  <option value="iabs"><xsl:text>_SunONE CSV</xsl:text></option>  
  <option value="vcard"><xsl:text>_vCard</xsl:text></option>  
</select>  
</td>
```

The options list can be changed to appear in the following order in drop down list.

- SunONE Address Book CSV
- Microsoft Outlook CSV
- Netscape Communication Address Book (LDIF)
- vCard

[Code Example 7-20](#) shows the code after modifying the options list

Code Example 7-20 Code after modifying the options list

```
<td class="PropInput" width="80%">  
<select name="importformat" tabindex="2">  
  <option value="iabs"><xsl:text>_SunONE CSV</xsl:text></option>  
  <option value="csvus"><xsl:text>_Outlook CSV</xsl:text></option>  
  <option value="ldif"><xsl:text>_Netscape Ldif</xsl:text></option>  
  <option value="vcard"><xsl:text>_vCard</xsl:text></option>  
</select>  
</td>
```

Customizing Communications Express for a Specific Domain

Communications Express allows you to customize all the three components namely Mail, Calendar and Address Book for a specific domain. The sections contained in this chapter are,

- [Customizing Default User Preferences for a Specific Domain](#)
- [Customizing Theme for a Specific Domain and Locale](#)
- [Customizing Mail for a Specific Domain](#)
- [Customizing Address Book for a Specific Domain and a Locale](#)

Customizing Default User Preferences for a Specific Domain

The default user preferences that will be used for a first-time user can be configured on a per-domain basis. The default user preferences can be configured in the `uwcdomainconfig.properties` file.

To configure default user preferences for a specific domain, for example, `siroe.com`, you need to do the following:

1. Create a directory with the name same as the domain name in `<domain-dir>` directory. For example, create `<domain-dir>/siroe.com` directory.
2. Copy `uwcdomainconfig.properties` file from `<domain-dir>` directory to the newly created directory.

3. Make the required changes to `uwcdomainconfig.properties` file in `<domain-dir>/siroe.com` directory. You can configure the default user preferences for global, calendar and address book.
4. Restart the web server to apply changes

Customizing Theme for a Specific Domain and Locale

You can configure themes on a per-domain basis.

NOTE Currently Communications Express does not support customization of themes on a per-user basis

To configure a theme for a specific domain, for example, `siroe.com` and a locale `ja` and `fr`, you need to do the following.

1. Create a directory with the name same as the domain name in `<domain-dir>` directory. For example, create `<domain-dir>/siroe.com` directory
2. Copy `uwcdomainconfig.properties` file from `<domain-dir>` to the `<domain-dir>/siroe.com` directory.
3. Modify the parameter `uwc-user-attr-sunUCTheme` to the required theme name in `uwcdomainconfig.properties` file in `<domain-dir>/siroe.com` directory. For example, to configure a theme "christmas" set `uwc-user-attr-sunUCTheme=christmas`. To configure a theme called "christmasJA" for japanese locale, set `uwc-user-attr-sunUCTheme-ja=christmasJA`. To configure a theme called "christmasFR" for french locale, set `uwc-user-attr-sunUCTheme-fr=christmasFR`. The parameter "uwc-user-attr-sunUCTheme" is used for default locale. It is important that though the theme name could be anything, the property name should end with corresponding locale as mentioned in the example.
4. Create the directories `<skin-dir>/christmasJA`, `<skin-dir>/christmasFR` and `<skin-dir>/christmas`.
5. Copy `themes.properties` from `<skin-dir>/` to `<skin-dir>/christmasJA`, `<skin-dir>/christmasFR` and `<skin-dir>/christmas`

6. Edit the corresponding `themes.properties` to point to the customized style sheets and images.
7. For help on various parameters in this file, refer to the Appendix D of the Communications Express Administration Guide. Look for “`themes.properties`” under the “Configuration Parameters Reference” section.
8. Restart the web server to apply changes.

Customizing Mail for a Specific Domain

This section describes how to customize the mail interface for each domain.

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

- Create a directory for mail with the domain name under `msg_svr_base/config/html` directory.
- Populate this directory with the customized versions of the files from the default directory hierarchy.

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

Table 8-1 Directory Structure for the Domain `siroe.com`

<code>html/...</code>	<code>// default interface</code>
<code>html/imx/...</code>	<code>// default interface</code>
<code>html/en/...</code>	<code>// default interface</code>
<code>html/siroe.com/main.js</code>	<code>// refers to imx/bottle.gif</code>
<code>html/siroe.com/imc/bottle.gif</code>	<code>// refers to imx/bottle.gif</code>

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

If you have many domains and only a few distinct ‘brands’ then you can use links to make the server point to the correct brand

Table 8-2 Linking Multiple Domains to few distinct brands

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

Domain From URL

The server listens to all IP addresses and presents a customized interface before the authentication occurs. The server does this by looking at the URL and determines if it contains a known domain and presents the per domain Login screen for the domain.

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

In this case a user does not have to suffix *@domain* to the username to login.

For more information on Customizing the Mail Component, refer the Sun Java System Messenger Express Customization Guide.

Customizing Address Book for a Specific Domain and a Locale

In address book, all UI related source files (xml/xsl) are located under `<uwcinstall>/WEB-INF/ui/html/abs` directory by default. To customize the UI for a domain, follow these steps.

1. Create a directory with a domain name, (e.g. `sireo.com`) under `<uwcinstall>/WEB-INF/ui/html`.
2. Copy all files from `<uwcinstall>/WEB-INF/ui/html/abs` to `<uwcinstall>/WEB-INF/ui/html/<domain>`.
3. Make the required changes in XSL files as explained in the previous sections.

Although all images are located under `<uwcinstall>/absimx` by default, you may create a directory with a domain name (e.g. `<uwcinstall>/absimx/sireo.com`) under it and include the icons that are customizable under the newly created domain directory. Also ensure that the path reference to the newly created directory is updated in `<uwcinstall>/WEB-INF/ui/html/<domain> search-images.xsl` and `commonimages.xsl`.

Localizing Communications Express

Communications Express Localization

Communications Express allows you to localize Calendar, Address Book and Options as per your requirements.

- [Localizing Calendar and Options](#)
- [Localizing for languages that use characters outside the ASCII range](#)
- [Localizing Mail](#)
- [Localizing Address Book](#)

Localizing Calendar and Options

To localize text, you need to change the `i18n.properties` file that is located in `<domain-dir>/domain/<locale>` directory. If the deployment is configured to work with multiple domains and multiple locales, then a directory by that domain name is created under `<domain-dir>/domain`

For example, if there is a domain called `example.com`, perform the following steps.

- Create `<domain-dir>/domain/example.com`
- Copy `<domain-dir>/domain/<locale>/i18n.properties` to `<domain-dir>/domain/example.com/<locale>` directory.
- Create directories for each supported locale (`en_US`, `ja`, `de_FR` etc) under that domain, copy `i18n.properties` to the domain, and change it accordingly

Localizing Date/Time Formats for Calendar

The localization of the Date/Time formats for Calendar should be performed by constructing locale-specific patterns for certain symbols, delimiters and hard-coded strings that are enclosed in single quotes. These symbols are converted into locale specific strings by Communications Express

[Table 9-1](#) contains the information on localizing Date/Time Formats for Calendar

Table 9-1 Localizing Date/Time Formats for Calendar

Symbol	Meaning	Presentation	Example
y	year	Number	1996
M	month in year	Text&Number	July & 07
d	day in month	Number	10
h	hour in am/pm (1-12)	Number	12
H	hour in day (0-23)	Number	0
m	minute in hour	Number	30
s	second in minute	Number	55
S	millisecond	Number	978
E	day in week	Text	Tuesday
D	day in year	Number	189
F	day of week in month	Number	2 (2nd Wed in July)
w	week in year	Number	27
W	week in month	Number	2
a	am/pm marker	Text	PM
z	time zone	Text	Pacific Standard Time
'	escape for text	Delimiter	(none)
'	single quote	Literal	'

Characters that are in the above table are treated as quoted text. These characters appear in the formatted text even if they are not enclosed within single quotes.

The number of symbol letters you specify also determines the format. For example, if the "zz" pattern results in "PDT," then the "zzzz" pattern generates "Pacific Daylight Time." [Table 9-2](#) summarizes these rules.

Table 9-2 Localization Rules

Presentation	Number of Symbols	Result
Text	1-3	'abbreviated form, if one exists
Text	>=4	full form
Number	minimum number of digits is required	shorter number are padded with zeros(for year, if the count of 'y' is 2, then the year is truncated to 2 digits)
Text & Number	1-2	text form
Text & Number	3	number form

Localizing for languages that use characters outside the ASCII range

You may want to use the 'native2ascii' utility provided by JDK to convert non-ascii characters into escaped unicode values. This utility is available at `<jdk-install-dir>/bin`

To use the 'native2ascii' utility, enter non-ascii characters for the corresponding values in the properties file and save it.

Use the command, `native2ascii -encoding <native_encoding> <src_filename> <target_filename>`

where,

`native_encoding` represents the encoding used to create the source file, `src_filename` represents the source file and `target_filename` represents the `i18n.properties` file.

Localizing Mail

Communications Express allows you to localize any feature of Mail. Different pages can be created for different languages. When you create language-specific static pages, you need to group them in subdirectories under the main document directory. The code automatically detects the client's language preference and fetches the pages from the appropriate subdirectory.

When you change common sections for static pages, you must make the changes in multiple locale (languages) directories, if modifications are desired across languages (for example, if changes are required to modify JavaScript behavior). Conversely, you can also make language-specific modifications selectively throughout the application.

Specific Locales

Table 9-3 lists the specific locales and their abbreviations that are supported by Communications Express services. The default language is English.

Table 9-3 Communications Express Specific Locales

Locale	Abbreviation
English	en
Japanese	ja
Spanish	es
French	fr
German	de
Russian	ru
Arabic	ar

Code Example 9-1 shows the changes to be made to the `ru/i18n.js` file for modifying the charset in the `i18n.resource` file.

Code Example 9-1 Modifying the charset in the i18n Resource File

```
// I18N Resource file
var i18n = new Array()
var fldr = new Array()
// DO NOT TRANSLATE AS STRINGS--JUST VALUES
i18n['fontface'] = 'PrimaSans BT,Verdana,sans-serif'
i18n['fontface1'] = i18n['fontface']
i18n['fontface2'] = 'Times New Roman,Times,serif'
i18n['fontface3'] = 'Courier New,Courier,mono'
i18n['nbsp'] = '&nbsp;'
```

Communications Express allows you to customize text that is displayed on web pages. This text is both customizable and localizable. The system also allows you to configure text on a per domain basis

To Add an Additional Locale

To add an additional locale, perform the following steps.

- Edit the `uwcdomainconfig.properties` file. This file is available in the `<uwc-deploy-dir>/WEB-INF/domain directory>`
- Add a keyword for the new locale that needs to be added. For example, if you need to add a new locale, “Afrikaans”

Code Example 9-2 Adding a keyword for the new locale

```
supportedLanguages=en;es;de;fr;ja;ko;zh-CN;zh-TW;Afrikaans
```

- Add a localized string for the locale name in different locales. This could be done by adding a new line in `i18n.properties` file of each locale. For example, let us add a key in `i18n.properties` file for en locale.

Code Example 9-3 Adding a localized string for the locale name

```
uwc-common-options-preferredLanguage-Afrikaans=Afrkaans Locale
```

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>

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 Communications 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 `msg_svr_base/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 Communications Express (such as: `en` for English, `fr` for French).

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

```
msg_svr_base/msg-instance/configutil | grep local.supportedlanguages
```

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

3. Copy the newly created `language_name.hash` file in the `msg_svr_base/dict` directory and restart the `mshttpd` service. When the `mshttpd` service is restarted, the Communications Express spell checker is enabled.

The following example shows how to create a German hash file (`ge.hash`) by using the `buildhash` utility.

4. Creating a German hash file by using the `buildhash` utility.

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

,where 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

Localizing Address Book

To localize address book, the xml file that needs to be modified is the dictionary-`<lang>.xml` file that is located under `<uwcinstalldir>/WEB-INF/ui/html/abs` directory.

Customizing Address Book for a Specific Locale

You can customize the Address Book UI either for a specific locale and not under a specific domain and also for a locale under a specific domain.

Customizing the Address Book UI for a locale and not under a specific domain

1. Create a directory with the locale name(e.g., ja under `<uwcinstall>/WEB-INF/ui/html/abs`
2. Copy all files from `<uwc-deploy>/WEB-INF/ui/html/abs` to `<uwc-deploy>/WEB-INF/ui/html/abs/<locale>`
3. Make the required changes in XSL files as explained in the previous sections.

Customizing the Address Book for a locale under a specific domain

1. Create a directory with `<domain>/<locale>` (e.g., `siroe.com/ja`) under `<uwcinstall>/WEB-INF/ui/html`
2. Copy all files from `<uwcinstall>/WEB-INF/ui/html/abs` to `.../abs/<locale>`
3. Make the required changes in `.xsl` files as explained in the previous sections.

Glossary

Refer to the *Java Enterprise System Glossary* (<http://docs.sun.com/doc/816-6873>) for a complete list of terms that are used in this documentation set.

Index

SYMBOLS

16

NUMERICS

49897

Head2-Procedure

Localizing Calendar and Options 97

A

absimx 16

attach button 66

attach_fs.html file 66

attachments 66

B

browse button 66

browser.js 49

buildhash 101

C

Cancel button 66

Collect button 67

collect_fs.html file 66

collecting mail from another server 66

comp_fs.html file 66

comp_fs_lr.html 61

composition window 52

composition window functions 53

Customizing General Features in Address Bo 71

Customizing General Features in Calendar 27

E

emoticons

disable 62

F

fldr_fs.html file 66

folders screen 51

H

Help button 66

I

i18n.js file 102

inbox screen 50

J

JavaScript 65

L

lang/i18n.js 59, 60, 61

ldap_fs.html file 66

listFrameHTML(doc) 59

localization

 specific locales 100

Localizing Calendar and Options 97

Localizing Communications Express 97

M

mailbox management tab 67

mailui.js 49

main.js 49, 62

mbox_fs.html file 56, 66, 68

message composition 67

message composition window 60

message display window 58

message list window 57

message screen 51

message tool bar 60

Messenger Express

 user interface customizable features 66

Modifying the Address Book Tool Bar 72

Modifying the Mailbox Tool Bar 72

msg_fs.html file 59, 60, 66, 67

msg_fs_lr.html 59

msg_svr_base/html/imx 61

O

opts_fs.html file 66, 68

P

POP server 67

POP user ID 67

R

receipt_fs.html file 66, 68

return receipt 68

S

setdomain.js 49

U

util.js 49

uwc/.js 16

uwc/calclient 16

uwc/common 16

uwc/css 16

uwc/images 16

uwc/mailclient 16

- uwc-common-bannerImage 17
- uwc-common-Error-Message-icon 18
- uwc-common-ImportExportImage 17
- uwc-common-Info-Message-icon 18
- uwc-common-PrintableImage 17
- uwc-common-Question-Message-icon 18
- uwc-common-Required-image 18
- uwc-common-RightBrandingImage 17
- uwc-common-SearchImage 17
- uwc-common-Sort-Down-NonSelected-image 17
- uwc-common-Sort-Down-Selected-image 17
- uwc-common-Sort-Up-NonSelected-image 17
- uwc-common-Sort-Up-Selected-image 17
- uwc-common-Subscribed-image 18
- uwc-common-To-Anchor-image 17
- uwc-common-WarningImage 17
- uwc-common-Warning-Message-icon 18

W

- WEB-INF/config 16
- WEB-INF/domain 16
- WEB-INF/skin 16
- WEB-INF/ui/html/abs 16

