



SUGARCRMTM
COMMERCIAL OPEN SOURCE

Sugar Community Preview Guide

Community Edition

Version 5.1 Beta 1

Sugar Community Preview-Community Edition
Version 5.1 Beta 1, 2008

Copyright © 2004-2008 SugarCRM Inc.

www.sugarcrm.com

This document is subject to change without notice.

License

This work is licensed under the Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 License ("License"). To view a copy of this license, visit <http://www.creativecommons.org/licenses/by-nc-nd/3.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Disclaimer

Your Warranty, Limitations of liability and Indemnity are expressly stated in the License. Please refer to the License for the specific language governing these rights and limitations

Trademarks

All SugarCRM logos in this document are registered trademarks of SugarCRM Inc. See the SugarCRM trademark policies at <http://www.sugarcrm.com/crm/open-source/trademark-information.html> for more information on how SugarCRM trademarks can be used.

Overview

Welcome and thank you for participating in the SugarCRM Community Preview process.

This document describes the release of Sugar 5.1 Beta for the Community Preview scheduled to begin in April.

The URL of the Community Preview version of Sugar Suite 5.1 Beta is:

https://www.sugarcrm.com/crm/Sugar5.1.0_Beta_CP

This preview site is accessible to all preview participants and thus is a shared site. Accordingly, do not store any confidential data on this site. Also, SugarCRM periodically refreshes the data on the site so changes you make will be erased.

We will work as quickly as possible to incorporate your feedback into the development process.

You can provide feedback through the SugarCRM Bug Portal or through a private forum called “Community Preview” at <http://forums.sugarcrm.com>.

The SugarCRM Development Team will be closely monitoring these channels as the process is underway.

Please review the Known Issues list, which is provided later in this document prior to submitting bugs. You can provide feedback through the SugarCRM Bug Portal, accessible through <http://bugs.sugarcrm.com>. Use “5.1.0 Beta” to designate new bugs found in this release.

You can log in to the Community Preview version by using the logins available on Sugar demo systems:

The users (and passwords) in demo data for Sugar Suite are as follows:

jim/jim (User “jim” with password of “jim” (no quotes)).

chris/chris

max/max

sally/sally

sarah/sarah

will/will

New and Enhanced Features in 5.1

Sugar 5.1 has implemented many enhancements to improve the application, including the Silent Upgrader.

New and enhanced features in Sugar 5.1 are as follows:

- Enhanced import functionality to improve performance and handle bigger import files; support to import data for all Sugar modules.
- Ability to allow Sugar Email client to access only some folders on your external Email server.

- Ability to create relationships between modules in Module Builder and Studio.
- Ability to create a field with a Link or IFrame datatype in Studio.
- Ability to lock down the Upgrade Wizard for complete control over your Sugar instances.
- Ability to lock down the Module Loader to ensure that modules are uploaded into Sugar only from a location of your choice.
- A new Logger to create log files.
- Ability to configure default permissions on Linux for control over ownership and accessibility to Sugar files.

Enhanced Import Functionality

You can now import data for the Bug Tracker. Also, you can enable the import functionality for any custom module that you build in Module Builder.

Other enhancements are as follows:

- When you map fields between the import file and the Sugar database, you can save the mappings for future use. You can also publish them to enable other users to use them.

-- Do not map this field --	lead_id	
-- Do not map this field --	account_name	
Campaign ID	campaign_id	<input type="text"/>
Email	email_address	<input type="text"/>
Assigned to	assigned_user_name	<input type="text"/> <input type="button" value="Select"/> <input type="button" value="Clear"/>
-- Do not map this field --	team_name	

- You can specify qualifiers such as double quotes and single quotes, if any, that enclose data fields in the import file.

- You have the option of importing data to create new records and to update existing records.
- If data is not imported successfully, Sugar logs the error messages in a .csv file and displays a link to the file.

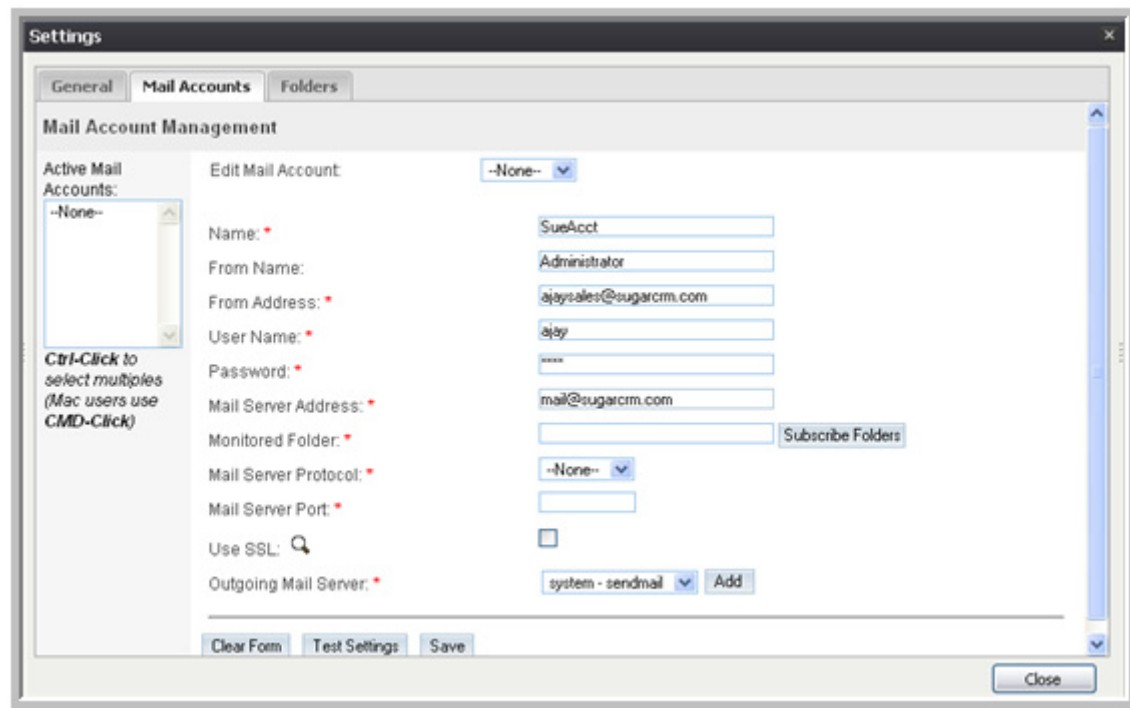
Limiting Access to Folders on External Email Accounts

With the IMAP protocol, you can limit access to specific folders on your external account.

To limit access to Email folders

- In the Email Settings window, select the Mail Accounts Tab.

2. In the Monitored Folder field, click **Subscribe Folders** and select the folders.



Creating Relationships Between Modules

You can create relationships between modules in Module Builder as well as Studio.

In Module Builder, you can create relationships between two undeployed modules, and between one undeployed module and one deployed module. However, you cannot create a relationship between two deployed modules.

In Studio, you can create relationships only between deployed modules. After you deploy a relationship in Studio, you cannot change it.

A relationship can be one of three types:

- One-to-one: You can also create a relationship between a module and itself. In this case, the relationship becomes a parent-child relationship.
- One-to-many: In this relationship, records in the primary module can have relationships with many records in the related module.
- Many-to-many. In this relationship, records in both the primary module and the related module can have relationships with multiple records in each module.

To create a relationship

1. Select the module in Module Builder or Studio

2. On the module's page, click **View Relationships**.

The screenshot shows the 'Module Builder' interface for a module named 'm1'. The breadcrumb path is 'Module Builder > p1 > m1'. At the top, there are buttons: 'Save', 'Duplicate', 'View Fields', 'View Relationships', 'View Layouts', and 'Delete'. Below these, the 'Package' is 'p1'. The 'Module Name' and 'Label' are both 'm1'. There are checkboxes for 'Importing' (unchecked), 'Team Security' (checked), and 'Navigation Tab' (checked). The 'Type' is 'basic', represented by a blue cube icon.

3. On the Relationships page, click **Add Relationship** and define the relationship in the Edit Relationship tab.

The screenshot shows the 'Edit Relationship' dialog box. It has tabs for 'm1 Relationships' and 'Edit Relationship'. At the top are buttons: 'Cancel', 'Save', and 'Delete'. Below these are three columns: 'Primary Module', 'Type', and 'Related Module'. The 'Primary Module' is 'p1_m1', the 'Type' is 'Many To Many', and the 'Related Module' is 'Activities'. Below the 'Primary Module' column is a 'Subpanel from Activities:' dropdown set to 'Default'. Below the 'Related Module' column is a 'Subpanel from p1_m1:' dropdown set to 'default'. At the bottom right is an 'Optional Condition:' section with a 'Value:' label and an empty text box.

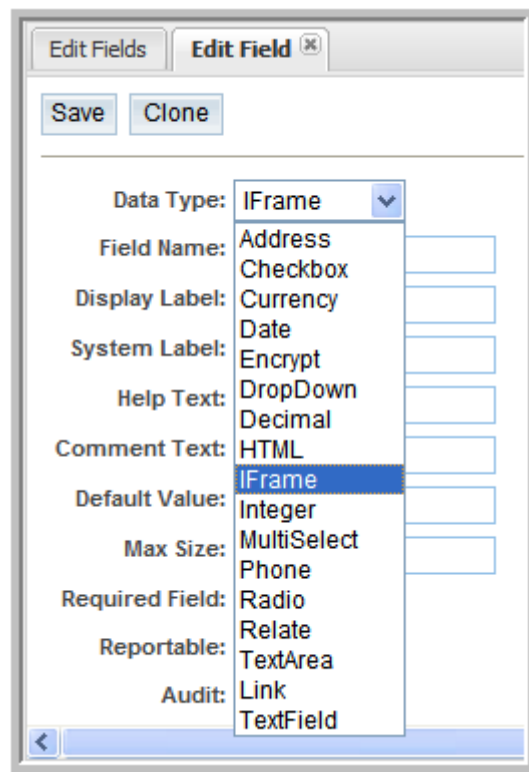
Embedding Links and IFrames in a Layout

A Link field allows you to store a URL in a record such as a customer's Website or a link to a related internal or external system. The URL can either be entered as a normal field in an Edit View, or it can be dynamically generated based on other fields in the record.

You can also embed a view of the Website itself in the layout rather than as a link by using the IFrame field. IFrames support regular URLs as well as generated URLs.

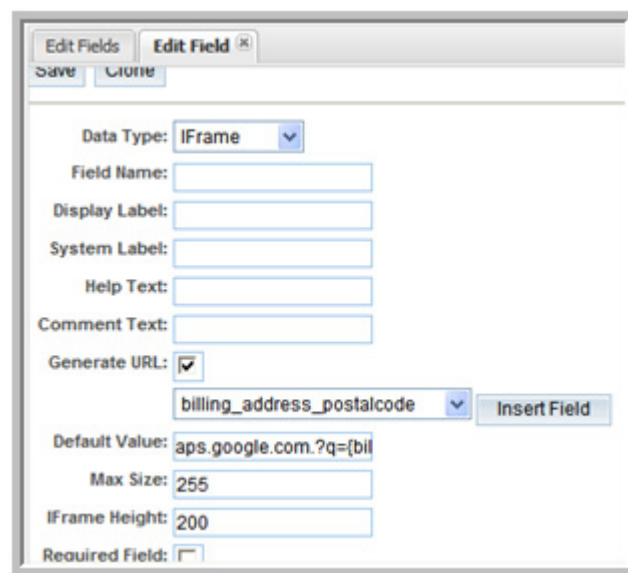
To embed a URL

1. To embed a manually generated URL, from the Data Type drop-down list, select **IFrame** or **Link**.



The screenshot shows the 'Edit Field' dialog box with the 'Data Type' dropdown menu open. The menu lists various data types: Address, Checkbox, Currency, Date, Encrypt, DropDown, Decimal, HTML, IFrame (highlighted), Integer, MultiSelect, Phone, Radio, Relate, TextArea, Link, and TextField. The 'IFrame' option is currently selected.

2. To embed a dynamically generated URL, click the **Generate URL** checkbox and insert the base URL into the Default Value field.
3. Select a field you wish to include in the URL from the dropdown and click **Insert Field** to append it to the base URL.



The screenshot shows the 'Edit Field' dialog box with the 'Generate URL' checkbox checked. The 'Default Value' field contains the URL 'aps.google.com.?q={bil'. The 'billing_address_postalcode' field is selected in the dropdown menu, and the 'Insert Field' button is visible. The 'Max Size' is set to 255 and the 'IFrame Height' is set to 200.

Locking Down the Upgrade Wizard

If you are managing multiple instances of the Sugar application and you want to maintain complete control over the instances, you can lock down the Upgrade Wizard to ensure that no user with administrative privileges can upgrade any of them.

To lock down functions on the Administration page

1. Navigate to the *config.php* file in the Sugar root directory.
2. Set the following parameter to `true` as as below:
`$sugar_config['admin_access_control']=true`
3. Save the file.

Locking Down the Module Loader

In order to ensure that users with administrative privileges do not load sub-standard modules into your Sugar application, you can lock down the Module Loader and direct them to load modules from a location of your choice.

To lock down the Module Loader

1. Navigate to the *config.php* file in the Sugar root directory.
2. Set the following parameter to `true` as as below:
`$sugar_config['file_access_control']=true`
3. Save the file.

Using a Slave Database

To enhance performance of the Reports module, you can configure a slave database to share the load with the master Sugar database.

To configure a slave database

1. Navigate to the *config.php* file in the Sugar root directory.
2. Set the following parameters as follows and save the file:
`$sugar_config['db']['reports'] = array`
 where **db** could also be an IP address.
`('db_host_name' => 'localhost',`
`'db_user_name' => 'readonly',`
 where **db_user_name** is name of a user with read-only privileges.
`'db_password' => 'mypassword',`
 where **mypassword** is the read-only user's password.
`'db_name' => 'reports_slave',`
 where **db_name** is the database with which Sugar is connecting
`'db_type' => 'mysql',`
`);`

Setting Logger Levels

Sugar now uses a custom Logger to create log files. By default, the logs are written to *sugarcrm.log* in the Sugar root directory.

When you upgrade to Sugar 5.1, Sugar automatically parses your Logger settings from the *log4.php* properties file of your previous Sugar version and populates the Logger Settings sub-panel with the information.

The logging levels are as follows:

- **debug**: Logs events that would help to debug the application.
- **info**: Logs informational messages.
- **warn**: Logs potentially harmful events.
- **error**: Logs error events in the application.
- **fatal**: Logs severe error events that leads the application to abort. This is the default level.
- **security**: Logs events that may compromise the security of the application.
- **off**: The logger will not log any events.

When you specify a logging level, the system will create log files for the specified level as well as higher levels. For example, if you specify 'Error', the system creates log files for 'error', 'fatal', and 'security'.

To set logger levels

1. On the Administration Home page, click **System Settings** in the System sub-panel and enter the necessary information in the Logger Settings sub-panel.

Configuring Default Permissions for Sugar Files on Linux

If you are running Sugar on Linux platform, you can control ownership and accessibility to all Sugar files and folders by configuring default user and group permissions.

The following is an example of setting Read, Write, and Execute permissions for the Apache user and the Apache group on Centos operating system:

```
'default_permissions' => array (  
    'mode' => 0755,  
    'chown' => 'apache',  
    'chgrp' => 'apache',  
) ,
```