

Enhanced Studio for SugarCRM

Administration Manual

Release 1.0

Enhanced Studio Administration Manual
Release 1.0, 2008
Copyright © 2008 Patrizio Gelosi, All Rights Reserved
This document is subject to change without notice.

Disclaimer

The Enhanced Studio software and all related documents are distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied.

Contents

Disclaimer	2
Contents	3
Preface	4
Intended Audience.....	4
Overview.....	4
Compatibility	5
Feature Matrix	6
System Administration	7
Download	7
Install.....	7
Uninstall / Disable	7
SugarCRM Upgrade.....	7
Enhanced Usage of SugarCRM Studio tool	8
How to create a "Hello World" Code-type field.	8
Advanced Code-type field management.	10
How to modify an existing Code-type field.....	13
Enhanced Usage of Module Builder tool.....	14
How to create a new Code-type field.	14
How to modify an existing Code-type field.....	15
Advanced Code-type field management.	16
Technical Notes.....	18
General.....	18
Code Preservation (NEW).....	18
Samples / Scenarios	19
1. Progressive Page ID Code.	19
2. Opportunities warning on amount	19
3. Google Map automatic link	20
4. SugarCRM standard text field	22
5. Image field.....	22
6. Generic AJAX Call.	24
7. SugarCRM DB / External DB Query.....	25
7.1 Contact List in Account Module	25
Webography	27
Enhanced Studio on SugarForge	27
Enhanced Studio on SugarExchange.....	27
Enhanced Studio on Sugar Forums.....	27

Preface

Enhanced Studio is a new powerful tool for SugarCRM Administrators. It aims to provide the SugarCRM **Studio** tool and the **Module Builder** with new features to meet any customization requirement. With the fundamental aid of Enhanced Studio, **SugarCRM can be easily customized without touching any line of SugarCRM code.**

Intended Audience

Enhanced Studio, as well as this Guide, is targeted to a wide range of audience. A non-developer Administrator will find easy-following procedures applied to many scenarios at the **Sample Section** of the present Document or on the web (to this end the present document is accompanied by a rich **Webography**). On the other side, a developer will be pleased to explore the Enhanced Studio capabilities and fit them to his specific demands : from the Database to the graphic interface design.

Overview

Enhanced Studio is available in two versions :

- Enhanced Studio **DEMO** , with all the functionalities available on SugarCRM Studio
- Enhanced Studio **FULL** , with all the functionalities available on SugarCRM Studio and Module Builder.

Enhanced Studio may be used for the widest range of purposes. **The Administrator can add any element he desires to all the Views available in SugarCRM (List View, Edit View, Quick Create View, Detail View, Search View and Dashlet View) of any SugarCRM Module or custom Module.**

An element might be one of the followings :

- Images
- Buttons
- AJAX-based fields doing any action
- AJAX-based autocomplete fields
- Fields automatically evaluated from others
- Embedded elements (Flash, Java, etc.)

...and whatever the Admin has in mind (there's virtually no limitation)

These elements, that are basically SugarCRM fields, can be **Static** or **Dynamic** at Admin's choice.

Static fields are mainly graphic elements (like Images, Buttons, etc.), while dynamic fields are elements that can store/retrieve values to/from Database, like any other SugarCRM field does.

Admin can freely customize the Database to store the data of the dynamic fields : the only limitation to the DB field creation is obviously the syntax of the DB Engine used.

After deployed, Dynamic and Static field values are available to the Admin to be :

- Created
- Viewed in a List
- Viewed in Detail
- Edited
- Searched for
- Ordered by (**NEW in the 2.2 release**)
- Added to a Dashlet
- Imported from a file

A field can be given a different behaviour for each SugarCRM View. For instance a standard SugarCRM text field can be built using Enhanced Studio : the value is simply dumped on the page when List-viewed or Detail-viewed whereas it can be edited in an input field when Edit-viewed (see Sample #4).

Compatibility

Enhanced Studio is currently compatible with **all SugarCRM 5 Versions / Editions**.

The last Enhanced Studio releases for each SugarCRM version can be found in the following table :

SugarCRM Versions / Editions	Enhanced Studio Releases	Enhanced Studio available Versions
SugarCRM Version 5.0 CE / PRO / ENT Editions (all patches)	Enhanced Studio 2.0	DEMO & FULL
SugarCRM Version 5.1 CE / PRO / ENT Editions (all patches)	Enhanced Studio 2.2	DEMO & FULL

Feature Matrix

The following table contains the information about which features are present in each Enhanced Studio Release / Version of the Compatibility table.

Enhanced Studio Rel. / Ver. Features	Enhanced Studio 2.0 DEMO (SugarCRM 5.0)	Enhanced Studio 2.0 FULL (SugarCRM 5.0)	Enhanced Studio 2.2 DEMO (SugarCRM 5.1)	Enhanced Studio 2.2 FULL (SugarCRM 5.1)
SugarCRM Studio Code type support	✓	✓	✓	✓
Module Builder Code type support	-	✓	-	✓
Database management (type, default, required, ...) for Code Dynamic DB-types	✓	✓	✓	✓
QuickCreate View Code type evaluation	✓	✓	-	✓
Detail View Code type evaluation	✓	✓	✓	✓
Edit View Code type evaluation	✓	✓	✓	✓
List View Code type evaluation	✓	✓	✓	✓
Basic/Advanced Search Code type support for Dynamic DB-types	✓	✓	✓	✓
Dashlets List / Search Views Code type evaluation	-	-	✓	✓
Importable Code Dynamic DB-types	-	-	✓	✓
Sortable Code Dynamic DB-types in List View	-	-	✓	✓
Advanced code-preservation during installation for Logic Hooks (see Technical Notes section for more details)	-	-	✓	✓

System Administration

Download

The last releases of Enhanced Studio **DEMO** can be downloaded from the Enhanced Studio Project download page on SugarForge, following this link

http://www.sugarforge.org/frs/?group_id=580

The last **FULL** versions can be purchased and then downloaded from SugarExchange project's page at this link

http://www.sugarexchange.com/product_details.php?product=580

Install

To install the package log in as admin user, then upload the package and install it using the "Admin" → "Module Loader" tool.

Please read carefully the Licenses (for both DEMO and FULL versions) before installing the package.

- License for **DEMO** version :
http://www.sugarforge.org/frs/download.php/4640/DEMO_VERSION_LICENSE.txt
- License for **FULL** version :
http://www.sugarforge.org/frs/download.php/4339/FULL_VERSION_LICENSE.txt

Uninstall / Disable

Once logged as admin user, uninstall / disable the package using the "Admin" → "Module Loader" tool.

SugarCRM Upgrade

To safely upgrade SugarCRM with Enhanced Studio installed :

- 1- Uninstall Enhanced Studio.
- 2- Install the SugarCRM upgrade.
- 3- Re-install Enhanced Studio.

Enhanced Usage of SugarCRM Studio tool

After having installed Enhanced Studio, the Admin finds a new field type (“**Code**”) in the SugarCRM Studio field list. This is the basic element from which to start to customize SugarCRM. The first step is to create a Code-type field in any SugarCRM module; then the field is available to be added to any View where it is required to appear.

How to create a “Hello World” Code-type field.

1. In the “Admin” → “Studio” tool, click the “Fields” Icon of the module to customize (“Cases” in the example in Figure 1).

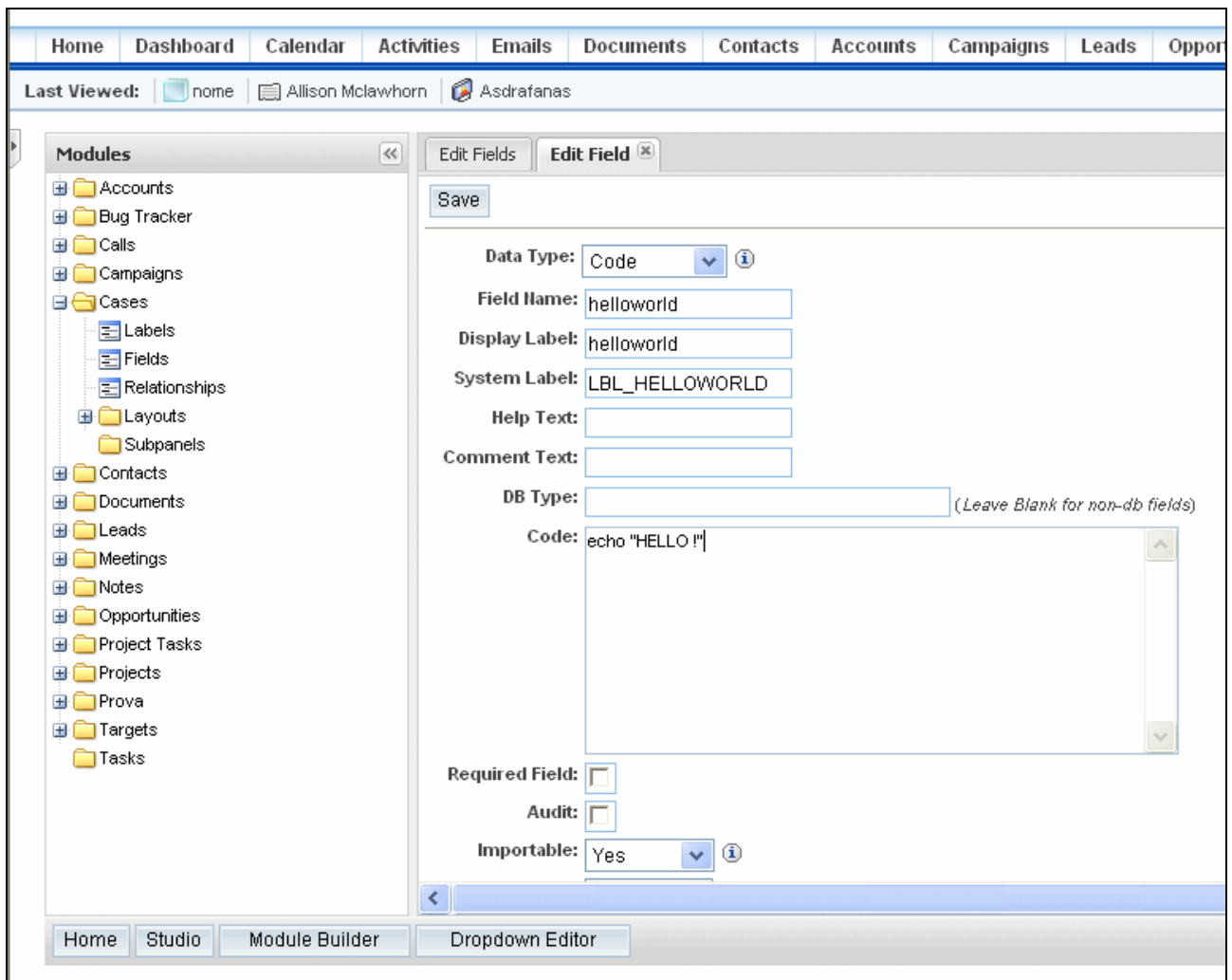


Figure 1

2. Select "Code" in the "Data Type" dropdown at the top of the field page. A mask like the one in Figure 1 will appear.

The mask contains all the standard SugarCRM type options, plus :

- The "**DB Type**" input field
- The "**Code**" textarea

The "DB Type" allows the admin to enter **the portion of the query string that creates/modifies the field in the Database custom table of the module** (as specified in the next paragraph).

To create a **static** Code field (with no DB value to store, as explained in the Overview section), the "DB Type" input field must be left blank, as specified in the note positioned on the right of the input field.

Otherwise a **dynamic** Code field will be created, whose type in SugarCRM Database is just the one entered in the "DB Type" input field.

The "Code" textarea permits to enter the php code that will be evaled at run-time each time the field is accessed for being viewed or edited.

A "Hello world !" sample can be found in the Figure 1 : a static Code field named "helloworld" (that will be automatically renamed into "helloworld_c" by SugarCRM Studio) is being creating through the simple Code

```
echo "HELLO !";
```

No change involves the Database, the only action performed as the field is viewed, listed or edited (depending on the Layout it is put into) is to write the string "HELLO !".

3. Save the field
4. Add the new Code field to the desired Layout.

Available Layouts are "EditView", "DetailView", "ListView", "QuickCreate", "Basic Search" and "Advanced Search".

WARNING

Only **dynamic** fields should be added to the Search Layouts since **static** fields are not created in the Database and so they cannot be searched for.

If the "helloworld" field of the example is added, for instance, to the "ListView" Layout, the Cases Module looks like the one in the Figure 2.

Cases: Home Print ? Help

Basic Search | Advanced Search

Number Subject Account Name

Only my items

Search | Saved Searches

Case List

Select Selected: 0 (1 - 20 of 50)

<input type="checkbox"/>	Num.	Subject	Account Name	helloworld	Priority	Status	Assigned to
<input type="checkbox"/>	1	Having Trouble Plugging It In	Doggie Diner Co Ltd 418071	HELLO!	Medium	Pending Input	sarah
<input type="checkbox"/>	2	Having Trouble Plugging It In	Dirt Mining Ltd 401795	HELLO!	Medium	Assigned	chris
<input type="checkbox"/>	3	Having Trouble Plugging It In	Rhyme & Reason Inc 193967	HELLO!	Low	New	sally
<input type="checkbox"/>	4	System is Performing Too Fast	Nimble Technologies Inc 884020	HELLO!	Medium	New	max
<input type="checkbox"/>	5	Need to Purchase Additional Licenses	Slender Broadband Inc 656789	HELLO!	Low	Closed	will
<input type="checkbox"/>	6	Warning message when using the wrong browser	TJ O'Rourke Inc 339632	HELLO!	High	Assigned	sarah
<input type="checkbox"/>	7	System is Performing Too Fast	COMPLETE HLDNG 465534	HELLO!	Low	Pending Input	sally
<input type="checkbox"/>	8	System is Performing Too Fast	Tri-State Medical Corp 344389	HELLO!	Low	Rejected	will

Figure 2

Advanced Code-type field management.

As explained in the previous paragraph, when a Code field type is selected in SugarCRM Studio a Code specific mask appears. The mask has the standard input fields in common with other SugarCRM field types, and in addition it has two specific fields :

1. DB Type.

The "DB Type" value switches the kind of Code field that is to be generated :

1.1. Static (Non-DB) field.

To select a static field just leave the "DB Type" entry blank.

This kind of field is only evaled at run-time and does not need to be stored/retrieved to/from DB, so **when a static field is created no field is added to the Database.**

It can be used for many kinds of elements that do not need to store a value to the Database, for example buttons that do any action or fields containing operations on the content of other fields, that need to be evaled each time they are viewed.

All the Code fields of the Enhanced Studio releases prior to 2.0 belong to this category.

An example a bit more useful than the "Helloworld" is the Sample #3 (Google Map automatic link) reported in the Samples Section.

1.2. Dynamic field.

From the Enhanced Studio 2.0 release, "Code" fields can store/retrieve values of any type to/from Database, exactly like other SugarCRM fields.

The php code assigned to the field is also evaled at run-time and the field value retrieved from the DB can be used from inside the php code itself (a standard SugarCRM Text field is reproduced in the Sample #4 to show this working).

The Admin can enter the **SQL creation string** right into the "DB Type" entry. The creation string is used to create a field on a custom DB-table that is bound to the Code field itself. The syntax must follow the one of the DB Engine in use, otherwise the creation / modification procedure fails and the field is not correctly saved.

For instance a variable-length string field having the Max Length equal to 255 can be created, on a MySQL DB Engine, through the following "DB Type" entry

```
VARCHAR (255)
```

To create an "Integer" DB field that is always valued and whose default value is "0", a valid entry for a MySQL DB Engine is the following :

```
INT (11) NOT NULL DEFAULT 0
```

2. Code entry.

The content of the "Code" entry is a php code which is evaled at run-time when the field is list-viewed, detail-viewed or edited.

Using php, any operation can be performed, including to echo to the screen any **HTML** or any **JavaScript** Code (even using **AJAX** functions).

Dynamic field value can be referred to as

```
$bean->field
```

in the php code. For example the first name of a contact can be easily dumped through a Code field in Contacts Module with this simple php code :

```
echo $bean->first_name;
```

The code to refer to a custom "try" field is (note that the "_c" prefix is automatically added to the custom field name by SugarCRM Studio) :

```
echo $bean->try_c;
```

Here's a complete list of the variables that can be accessed from inside the Code :

- `$bean` (whose possible changes are kept after the termination of the Code)
- `$event` (containing useful information about the specific view)
- `$args`
- `$GLOBALS`

The exact content of the variables should need an in-depth examination that involves SugarCRM application structure and would go beyond the aim of this Guide. However, the following is a useful suggestion to explore the content of these variables while they are run-time evaled, and generally to **Debug** the Code. Simply add the following debugging line to the Code:

```
error_log(print_r(<variable>, true), 3, "<log_file>");
```

Where:

- <variable> is the variable to watch, chosen among the previous list
- <log_file> is the path to a writable log file (for example "/tmp/watch.txt")

Then, once accessed to a View containing the Code field, the content of the variables can be found in the specified log file.

WARNING

Please remember to delete the debugging line from the code after finished, or it will carry on appending data to the log file every time the field is accessed.

This Guide cannot go into the specific code language syntax and the SugarCRM application structure, but it provides an exhaustive Sample section with a wealth of scenarios.

After created, fields can be added to almost any Layout and almost any operation can be performed on them (depending on their types) as described in the following table.

Field DB-Type Operation	Static	Dynamic ("Text" Type)	Dynamic (Other Types)
Quick Create	✓	✓	✓
Detail View	✓	✓	✓
Edit	✓	✓	✓
List View	✓	✓	✓
Basic/Advanced Search	-	✓	✓
Dashlet View	✓	✓	✓
Import	-	✓	✓
Order By (List View)	-	-	✓

Numeric Code DB fields are also recognized by the Enhanced Search plugin (<http://www.sugarforge.org/projects/enhancedsearch/>) and can be searched for using the Enhanced Search numeric operators.

How to modify an existing Code-type field.

Both Static and Dynamic Code fields can be modified after they have been deployed through the SugarCRM Studio tool according to the Admin's preferences.

The only important limitation comes with the Database behaviour and is resumed in the following warning.

WARNING

The DB-Type of a Code field must not be changed to an incompatible type field (i.e. from "text" to "integer") when the related Database field contains any not-empty value (that might also happen when no row is shown in SugarCRM, as rows are logically but not physically deleted).

This obviously leads to an error if such DB values cannot be automatically converted by the DB Engines.

So it is allowed to change DB-type even after the field has been created, but the related DB-field must be manually emptied in the Database if the specific DB type conversion is not supported by the DB Engine.

Moreover, if a Dynamic Code field is changed to a Static field after it's been created in SugarCRM Studio, the related DB field is not automatically removed from the Sugar tables, and should be manually deleted in order to save Database space.

Enhanced Studio is a very powerful tool that allows deep after-deployment changes : this feature fully balances the precautions expressed in this paragraph.

Enhanced Usage of Module Builder tool

After having installed Enhanced Studio **FULL** version, the Admin finds a new field type ("**Code**") in the SugarCRM Module Builder field list. This is the basic element from which starting to customize any user defined module in SugarCRM. The first step is creating a Code-type field in any user-generated module; then the field can be added to any View where it is required to appear.

How to create a new Code-type field.

1. In the "Admin" → "Module Builder" tool, click the "Fields" Icon of any user-generated module to customize.

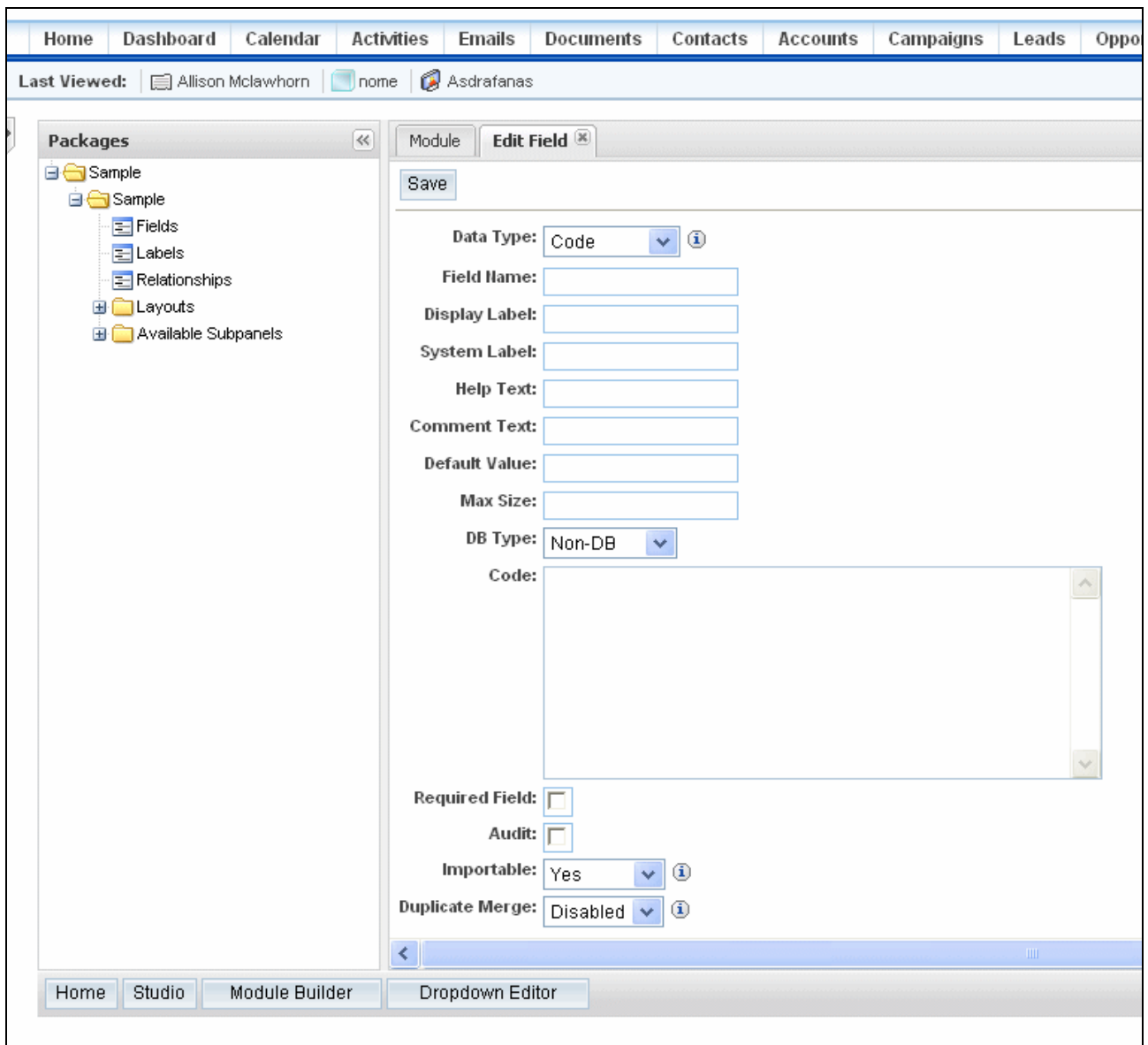


Figure 3

2. Select "Code" in the "Data Type" dropdown at the top of the field page. A mask like the one in Figure 3 will appear.

In addition to the standard input fields, there are four Code-type specific :

- The "**Default Value**" input field
- The "**Max Size**" input field
- The "**DB Type**" selection
- The "**Code**" textarea

The "Code" field works exactly like the Studio one : the php code entered is run-time evaled during the field access.

The "DB Type" is different from the Studio's as it forces the Admin's choice to a list of SugarCRM types mapped by SugarCRM into DB specific types (see the Advanced management paragraph of this Section for a detailed description of how they are mapped in the different DB Engines).

Given that the "DB type" field does not support the entry of further parameters, like the equivalent field in Studio does, two input fields has been added to this mask to specify the "Default Value" and the "Max Size", that is the max size in characters for a field in the Database.

3. Save the field.
4. Add the new Code field to the desired Layout.

Available Layouts are "EditView", "DetailView", "ListView", "QuickCreate", "Dashlets", "Basic Search" and "Advanced Search".

WARNING

Only **dynamic** fields should be added to the Search Layouts since **static** fields are not created in the Database and so they cannot be searched for.

5. Deploy the package containing the module.

How to modify an existing Code-type field.

Both Static and Dynamic Code fields can be modified according to your preferences using the SugarCRM Module Builder tool, with a procedure similar to the creation :

1. Edit the Code field according to your preferences.
2. Save the field.

3. Deploy the package.

As for the rest, please refer to the corresponding paragraph of the Enhanced SugarCRM Studio Usage section.

Advanced Code-type field management.

As explained in the previous paragraph, when a Code field type is selected in SugarCRM Module Builder, a Code specific mask appears. The mask has the standard input fields in common with other SugarCRM field types, and in addition it has four specific fields :

1. DB Type.

“DB Type” is used to switch between **Static** and **Dynamic** fields, like in Studio : to create a static field, the “Non-DB” option has to be selected, otherwise a dynamic field will be created.

As for static fields, the same concepts expressed for SugarCRM Studio hold true. Dynamic fields are created in a different way according to the “DB type” selected and the DB Engine in use, as detailed in the following table

DB-Type	MySQL	MsSQL	Oracle
Non-DB	-	-	-
Int	Int	Int	number
Double	Double	Float	number(30,10)
Float	Float	Float	number(30,6)
Uint	int unsigned	Int	number(15)
Ulong	bigint unsigned	Int	number(38)
Long	Bigint	bigint	number(38)
Short	smallint	smallint	number(3)
Varchar	varchar	varchar	varchar2
Text	text	text	clob
Longtext	longtext	text	clob
Date	date	datetime	date
Enum	varchar	varchar	varchar2(255)
Relate	varchar	varchar	varchar2
Multienum	text	text	clob
Html	text	text	clob
Datetime	datetime	datetime	date
Time	time	datetime	date
Bool	bool	bit	number(1)
Tinyint	tinyint	tinyint	number(3)
Char	char	char	char
Blob	blob	image	varchar2(36)
Longblob	longblob	Image	blob
Currency	decimal(26,6)	Decimal	blob
Decimal	decimal	Decimal	number(26,6)
Decimal2	decimal	Decimal	number(20,2)
Id	char(36)	varchar(36)	number(30,6)

2. **Code entry.**

As for the Code entry, refer to the SugarCRM Studio corresponding paragraph. The only relevant difference from Studio is that when referring to a Dynamic field value as "\$bean-><field_name>" the "_c" postfix is not automatically added. For example if the field's name is "try", its value can be easily dumped through this php code :

```
echo $bean->try;
```

3. **Default Value entry.**

A default value can be entered for the Dynamic field, according to the "DB Type" selected (the Default Value is ignored for Static fields). This entry is disabled for "text", "longtext", "multienum" , "blob" and "longblob" DB-Types to follow a constraint of DB Engines.

Default value is ignored if the "Required Field" checkbox is not ticked.

4. **Max Size entry**

A max character length can be assigned to the Dynamic fields to optimize Database disk space usage. The Max Size is ignored for Static fields and for those fields having a fixed conversion Size by default (see the conversion "DB Type" table above).

Technical Notes

General

One of the primary targets of the Enhanced Studio project is to minimize the impact on SugarCRM original code. The main reason is to avoid problems with Enhanced Studio functionalities when upgrading the SugarCRM main application.

The use of **Logic Hooks** is along these lines.

A new extension has been released in Enhanced Studio 2.2 : it permits code-preservation of some SugarCRM files without overwriting them as it would happen in a standard package installation.

It is described in the following paragraph.

Code Preservation (NEW)

When Enhanced Studio is installed, a new extension comes in use to preserve user-defined logic hook settings.

Before the 2.2 release, the "<sugardir>/custom/modules/logic_hooks.php" setting file was simply overwritten during Enhanced Studio installation, so the SugarCRM developers might have lost some customizations.

From the 2.2 release the Code Preservation extension automatically distinguishes between the Enhanced Studio settings and the other settings in the "logic_hooks" file. It replaces the old Enhanced Studio code with the new release one (surrounded by a comment tag containing the Enhanced Studio Release / Version info) and preserves the original code while copying the contents into the new file version.

The extension usage is also very useful on other files that should have been overwritten by Enhanced Studio and might have been affected by the risk of being changed in future SugarCRM patches (and every time it happens, a new compatible Enhanced Studio patch might be needed).

Samples / Scenarios

1. Progressive Page ID Code.

Scenario : add a progressive ID to the page screen of the List View of any module.

Procedure :

1. Create a new Code field with the values below in the module where to add the counter :

DB Type : *leave it blank*

CODE :

```
$GLOBALS['count_cc']++;  
echo '<div align="right">'.$GLOBALS['count_cc'].'</div>';
```

2. Add it to Layouts → ListView.

2. Opportunities warning on amount

Scenario : add a warning field that shows different images depending on the amount value. It shows "Thumb down" for less than 20,000 \$ amount, "Thumb up" for more than 60,000 \$.

Procedure :

1. Create a new Code field in the Opportunities module with the values below

DB Type : *leave it blank*

CODE :

```
if ($bean->amount_usdollar > 60000) echo "<image  
src='http://www.sugarcrm.com/forums/images/icons/icon14.gif'>";  
elseif ($bean->amount_usdollar < 20000) echo "<image  
src='http://www.sugarcrm.com/forums/images/icons/icon13.gif'>";
```

2. Add it to Layouts → ListView.

The List View should appear like shown in the Figure 4

Opportunity List						
Select ▾	Delete	Export	Merge Duplicates	Selected: 0		
<input type="checkbox"/>	Name ⇅	Account Name ⇅	Sales Stage	Amount ⇅	warning1	
<input type="checkbox"/>	SEA REGION S A 69708 - 1000 units	SEA REGION S A 69708	Prospecting	€75.000,00	👍	
<input type="checkbox"/>	EEE Endowments LTD 534404 - 1000 units	EEE Endowments LTD 534404	Negotiation/Review	€75.000,00	👍	
<input type="checkbox"/>	NW Bridge Construction 590016 - 1000 units	NW Bridge Construction 590016	Closed Lost	€10.000,00	👎	
<input type="checkbox"/>	MTM Investment Bank F S B 961149 - 1000 units	MTM Investment Bank F S B 961149	Qualification	€75.000,00	👍	
<input type="checkbox"/>	NW Capital Corp 846231 - 1000 units	NW Capital Corp 846231	Prospecting	€50.000,00		
<input type="checkbox"/>	Jungle Man Inc 276547 - 1000 units	Jungle Man Inc 276547	Closed Won	€10.000,00	👎	
<input type="checkbox"/>	MMM Mortuary Corp 51712 - 1000 units	MMM Mortuary Corp 51712	Needs Analysis	€50.000,00		
<input type="checkbox"/>	5D Invest A/S 776587 - 1000 units	5D Invest A/S 776587	Id. Decision Makers	€10.000,00	👎	

Figure 4

3. Google Map automatic link

Scenario : add a link to the Google Map of the customer address on Contacts / Accounts module.

Procedure for Accounts :

1. Create a new Code field in the Accounts module with the values below

DB Type : *leave it blank*

CODE :

```
$search_string = urlencode("$bean->billing_address_street
$bean->billing_address_city - $bean->billing_address_country");

echo '<a
href="http://maps.google.it/maps?f=q&hl=en&geocode=&q=' .
$search_string . '&ie=UTF8&z=12&iwloc=addr"
style="color:#0000FF;text-align:left" target="_blank">Click
here to view the map</a>';
```

2. Add it to Layouts → DetailView or EditView.

Procedure for Contacts :

1. Create a new Code field in the Contacts module with the values below

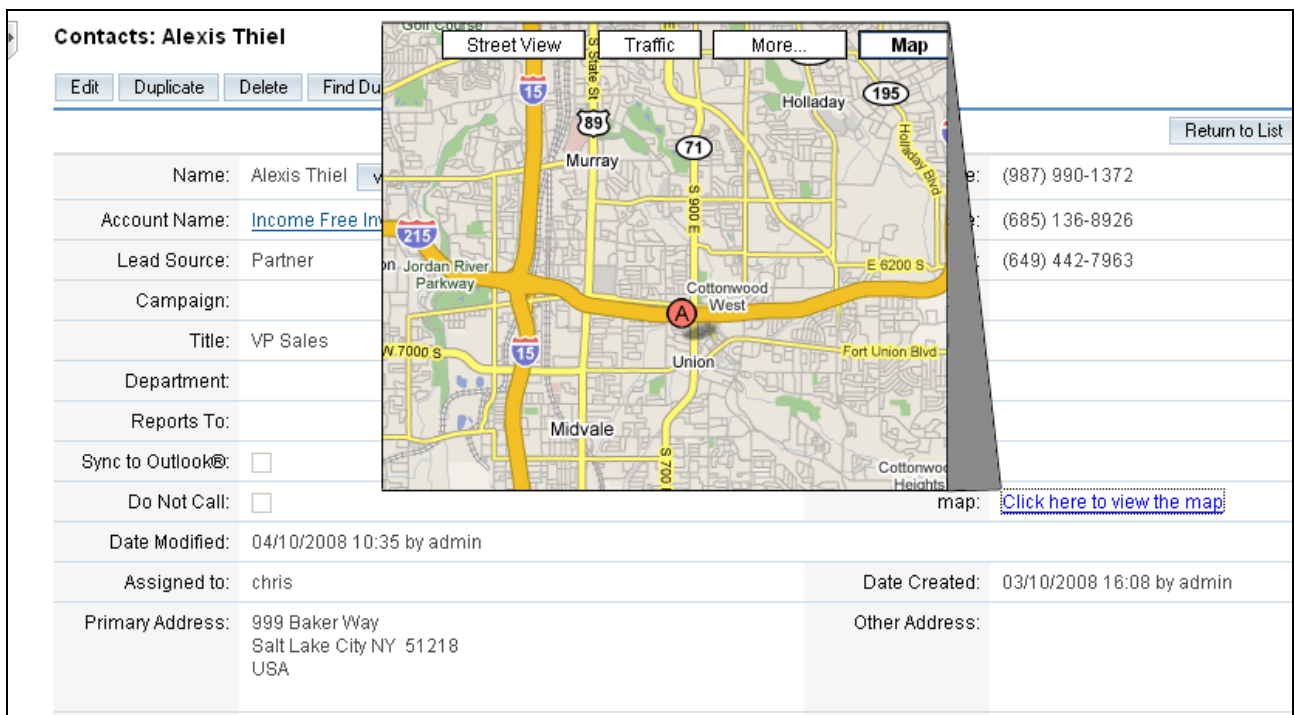
DB Type : *leave it blank*

CODE :

```
$search_string = urlencode("$bean->primary_address_street  
$bean->primary_address_city - $bean->primary_address_country");  
  
echo '<a  
href="http://maps.google.it/maps?f=q&hl=en&geocode=&q=' .  
$search_string . '&ie=UTF8&z=12&iwloc=addr"  
style="color:#0000FF;text-align:left" target="_blank">Click  
here to view the map</a>';
```

2. Add it to Layouts → DetailView or EditView.

The Detail View for a contact should appear like the one shown in the Figure 5



The screenshot displays the 'Contacts: Alexis Thiel' detail view. On the left, there is a list of fields including Name, Account Name, Lead Source, Campaign, Title, Department, Reports To, Sync to Outlook, Do Not Call, Date Modified, Assigned to, and Primary Address. The Primary Address is 999 Baker Way, Salt Lake City NY 51218, USA. On the right, there is a list of phone numbers. A map of Salt Lake City is overlaid on the contact information, showing a red pin at the location of the contact. The map includes labels for 'Street View', 'Traffic', 'More...', and 'Map'. A link 'Click here to view the map' is visible at the bottom right of the map area.

Figure 5

4. SugarCRM standard text field

Scenario : reproduce the standard SugarCRM text field.

The effect is the same of adding ad SugarCRM text field, but it can be used as a base to develop new features (such as AJAX spellcheck, auto-complete or validation Javascript for instance).

Procedure :

1. Create a new Code field in any module with the values below

DB Type :

TEXT

Code :

```
if ($GLOBALS['app']->controller->action == 'EditView')
    echo '
<input type="text" value="'. $bean-><field_name>.'" size="30"
id="<field_name>" name="<field_name>"/>';
else
    echo $bean-><field name>;
```

where the string "<field_name>" in the Code must be replaced with the Field Name (with the "_c" postfix if using SugarCRM Studio).

2. Add it to any Layout

The result is the same as a standard SugarCRM text field were used.

5. Image field.

Scenario : add an image field with a web link.

Procedure :

1. Create a new Code field in any module with the values below

Field Name : web_image

Display Label : web image

System Label : LBL_WEB_IMAGE

DB Type :

VARCHAR(255)

Code :

```
if ($GLOBALS['app']->controller->action == 'EditView')
    echo '
    (URL : )<input type="text" title="" value="'. $bean->web_image_c.'"
    maxlength="255" size="50" id="web_image_c"
    name="web_image_c"/><br/>';

if ($bean->web_image_c)
    echo '<br>';
```

2. Add it to Layouts → DetailView or EditView.

A sample of Edit View for the field is shown in the Figure 6



Figure 6

B. Generic AJAX Call.

Scenario : perform any AJAX call and dump results on the screen without reloading the page.

CODE :

```
echo '  
<script>  
function ajax_action () {  
  
    var callback = {  
        success: function(o) {  
            ... write here your action ...  
        }  
    }  
  
    var connectionObject = YAHOO.util.Connect.asyncRequest  
("GET", ... write here your URI ..., callback);  
}  
</script>  
<input type="button" value="ajax1" onclick="ajax_action();">';
```

The AJAX URI might call for instance a .php file written by the Admin that performs any action, with GET parameters passed as input and the result echoed and returned by the "success" function. For example the result can be Javascript alerted through the following callback:

```
var callback = {  
    success: function(o) {  
        alert(o.responseText)  
    }  
}
```

7. SugarCRM DB / External DB Query

Scenario : connect to SugarCRM DB or any external DB, execute a query (possibly containing any data of the current field through the "\$bean" variable) and echo the results. MySQL DB Engine is assumed to be used in this and the next sample.

CODE :

```
global $connection_code_field;

if (!isset($connection_code_field)) {
    $connection_code_field = mysql_connect("<db_host>",
"<db_user>", "<db_password>");
    mysql_select_db("<db_name>", $connection_code_field);
}
$sql = "<db_query>";

$res = mysql_query($sql, $connection_code_field);
if (mysql_num_rows($res))
    $value = mysql_result($res, <id_field>);

echo $value;
```

where <db_host>, <db_user>, <db_password>, <db_name>, <db_query> and <id_field> must be properly set.

7.1 Contact List in Account Module

Scenario : add a list of clickable related contacts for each account.

Procedure :

2. Create a new Code field in the Accounts module with the values below

DB Type : *leave it blank*

Code :

```
global $connection_code_field;

if (!isset($connection_code_field)) {
    $connection_code_field = mysql_connect("<db_host>", "<db_user>",
"<db_password>");
    mysql_select_db("<db_name>", $connection_code_field);
}
$sql = "SELECT ac.contact_id as id, CONCAT(c.first_name, ' ',
c.last_name) as name FROM contacts c JOIN accounts_contacts ac ON
c.id = ac.contact_id WHERE ac.account_id = '$bean->id'";
```

```

$res = mysql_query($sql, $connection_code_field);
if (mysql_num_rows($res)) {
    while($value = mysql_fetch_array($res))
        $values[] = "<a
href='index.php?module=Contacts&return_module=Accounts&action=Detail
View&record=" . $value['id'] . "'>" . $value['name'] . "</a>";
    echo join($values, '<br/>');
}

```

where <db_host>, <db_user>, <db_password> and <db_name> must be properly set.

3. Add it to Layouts → DetailView or ListView.

A sample of List View for the field is shown in the Figure 7

Account List					
Select	Delete	Export	Merge Duplicates	Compose Email	Selected: 0
Account Name	City	Type	contactslist	User	
<input type="checkbox"/> Dirt Mining Ltd 401795	Los Angeles	Customer	Priscilla Meldrum Jaime Risley Demarcus McCreary Constance Ludwig Moises Cash Reyes Scalf Wilton Waites	chris	<input type="checkbox"/>
<input type="checkbox"/> Jungle Man Inc 338625	Salt Lake City	Customer	Kristi Alcaraz Ambrose Ladner Damion Knights Bud Goudeau Lionel Alford Imelda Karpinski Bennett Arline	chris	<input type="checkbox"/>
<input type="checkbox"/> C Nelson Inc 762191	Kansas City	Customer	Marvin Berta Adelaide Kalman Jed Winbush	chris	<input type="checkbox"/>
<input type="checkbox"/> Jungle Man Inc 276547	San Francisco	Customer	Eddie Swims Rosemarie Trew Jocelyn Watford	chris	<input type="checkbox"/>
<input type="checkbox"/> Income Free Investing LP 400013	Alabama	Customer	Charles Work Irving Rome Winnie Boggs Virginia Winkel Rita Unger Sid Somerville Alexis Thiel	chris	<input type="checkbox"/>

Figure 7

Webography

Enhanced Studio on SugarForge

The official Enhanced Studio page on SugarForge can be found at http://www.sugarforge.org/frs/?group_id=580

The history of all the releases is available at the download section, as well as a dedicated Forum Area, Documentation, Reviews and more.

Enhanced Studio on SugarExchange

Visit the Enhanced Studio page on the official Marketplace of Sugar Products at http://www.sugarexchange.com/product_details.php?product=580

Before purchasing the product, **please read carefully the EULA license** : http://www.sugarforge.org/frs/download.php/4339/FULL_VERSION_LICENSE.txt

Enhanced Studio on Sugar Forums

(<http://www.sugarcrm.com/forums/>)

Here's a collection of the most suggestive posts related to Enhanced Studio

- (ENG) Enhanced Studio : A new tool to customize Sugar
<http://www.sugarcrm.com/forums/showthread.php?t=33140>
- (ENG) Enhanced Studio 2.1 available
<http://www.sugarcrm.com/forums/showthread.php?t=37595>
- (ENG) How to get the user logged??
<http://www.sugarcrm.com/forums/showthread.php?t=38258>
- (ENG) [Tutorial] Howto use ajax with sugar
<http://www.sugarcrm.com/forums/showthread.php?t=36226>
- (ESP) Usando ajax con sugar
<http://www.sugarcrm.com/forums/showthread.php?t=36182>
- (ENG) Evaluate Fields from other
<http://www.sugarcrm.com/forums/showthread.php?t=38053>
- (POR) Adicionar campo com imagem
<http://www.sugarcrm.com/forums/showthread.php?t=37994>

- (ENG) Zend_Framework Thoughts?
<http://www.sugarcrm.com/forums/showthread.php?t=37262>
- (ENG) Show iframe with URL parameters on Account DetailView page?
<http://www.sugarcrm.com/forums/showthread.php?t=35318>
- (ENG) Relate and Auto populate fields
<http://www.sugarcrm.com/forums/showthread.php?t=35905>
- (ENG) Maps for Contacts / Accounts addresses with Enhanced Studio
<http://www.sugarcrm.com/forums/showthread.php?t=33792>
- (ENG) Create DB storable custom fields with Enhanced Studio 2.0
<http://www.sugarcrm.com/forums/showthread.php?t=34736>
- (ENG) manually remove enhanced search/studio
<http://www.sugarcrm.com/forums/showthread.php?t=34991>
- (ENG) Add a contact field with tasks count
<http://www.sugarcrm.com/forums/showthread.php?t=34833>
- (DEU) add contact field to activities schedule call
<http://www.sugarcrm.com/forums/showthread.php?t=34500>
- (ENG) Enhanced Studio - losing subpanels and fields in details view
<http://www.sugarcrm.com/forums/showthread.php?t=34573>
- (ENG) custom field of html type with dynamic querystring???
<http://www.sugarcrm.com/forums/showthread.php?t=33745>
- (ENG) enhanced studio and external Databases
<http://www.sugarcrm.com/forums/showthread.php?t=34550>
- (ENG) Recommend any free modules/dashlets?
<http://www.sugarcrm.com/forums/showthread.php?t=34395>
- (DEU) Datensatz ID im Detail View
<http://www.sugarcrm.com/forums/showthread.php?t=33954>
- (ENG) Not sure what this is called...
<http://www.sugarcrm.com/forums/showthread.php?t=33779>
- (ENG) customize the way a field is displayed
<http://www.sugarcrm.com/forums/showthread.php?t=33791>
- (ENG) Custom fields and cache problems
<http://www.sugarcrm.com/forums/showthread.php?t=33734>