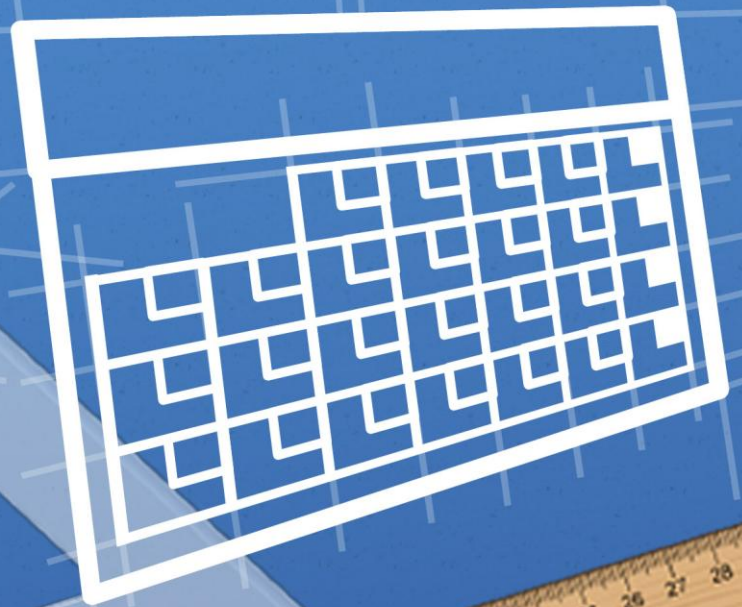
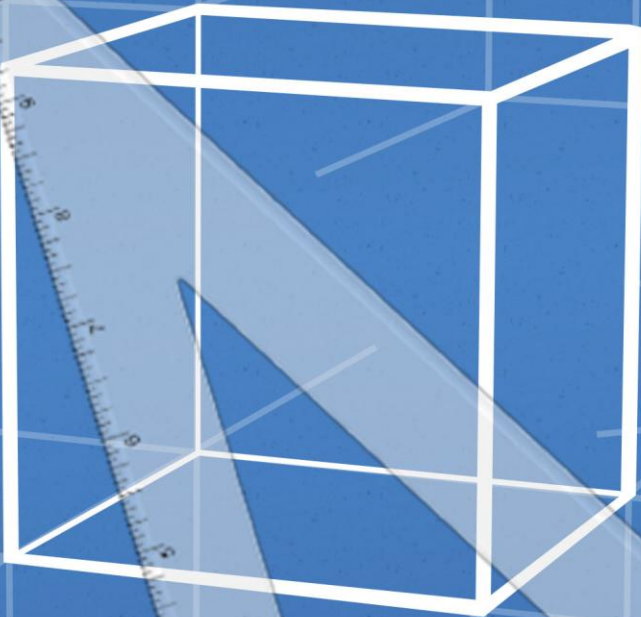


Advanced Datetime Manual

1.0



dispage

Advanced Datetime Manual
Release 1.0.00 – Edition I, 2010
Copyright © 2008 - 2010 Dispage - Patrizio Gelosi, All Rights Reserved
This document is subject to change without notice.

Disclaimer

The Advanced Datetime software and all related documents are distributed on an “**AS IS**” basis, **WITHOUT WARRANTY OF ANY KIND**, either express or implied.

Contents

Preface	4
Overview	5
Compatibility.....	6
Supported Languages.....	6
Feature Matrix	7
System Administration.....	8
AdvancedDatetime fields.....	9
AdvancedDatetime Sub-Types	10
Search for an AdvancedDatetime value	11
AdvancedDatetime calculated fields	11
AdvancedDatetimeOperations Method Reference	14
AdvancedDatetime Method Parameters Table	18
Webography.....	19
Advanced Datetime on Dispage website	19
Advanced Datetime on SugarForge	19

Preface

Advanced Datetime is an Enhanced Studio sub-extension to manage Datetime fields. Used in conjunction with Enhanced Studio, allows to perform any customization involving time measures.

Preliminary reading:

Enhanced Studio Manual

<http://www.dispage.com/support/documentation/enhanced-studio-manual>

Overview

Advanced Datetime extension adds to SugarCRM a new field type: **AdvancedDatetime**. The new AdvancedDatetime fields are more versatile than the default SugarCRM ones, since any date/time measure and dimension can be managed.

Additionally, **Enhanced Studio's Code fields** can be used to perform any kind of calculation among AdvancedDatetime fields thanks to the **AdvancedDatetimeOperations** library, included in Advanced Datetime extension.

Advanced Datetime field type had been integrated in Enhanced Studio releases prior to 3.1.04 with the name of **DatetimeCombo type**.

From Enhanced Studio 3.1.04, it has been splitted into a separated extension and renamed into "AdvancedDatetime" because from SugarCRM™ rel. 6.0 DatetimeCombo fields have been improved in a way that would cause conflicts with the DatetimeCombo fields supported by Enhanced Studio.

Compatibility

Advanced Datetime is currently compatible with all **SugarCRM 5.1, 5.2, 5.5 and 6.0 Versions (with the exclusion of the betas) – all Editions.**

The last Advanced Datetime releases for each SugarCRM version can be found in the following table:

SugarCRM Versions	SugarCRM Editions	Releases
SugarCRM 5.1.x SugarCRM 5.2.x	CE / PRO / ENT	Enhanced Studio 3.0 (DatetimeCombo)
SugarCRM 5.5.x SugarCRM 6.0.x	CE / PRO / ENT	Advanced Datetime 1.0

Table 1

More detailed and up-to-date info on SugarCRM compatibility can be found at

http://www.dispage.com/products/enhanced-studio#tech_info

Supported Languages

The following languages are currently supported by Advanced Datetime extension:

- English
- Spanish
- Italian
- Dutch
- German
- Brazilian Portuguese

Feature Matrix

The following table reports which features are present in each Release and Version of Enhanced Studio / Advanced Datetime.

Enhanced Studio / Advanced Datetime Rel. / Ver. Features	Enhanced Studio 3.0 DEMO	Enhanced Studio 3.0 FULL	Advanced Datetime 1.0 FULL
DatetimeCombo / AdvancedDatetime type field	✓	✓	✓
DatetimeCombo / AdvancedDatetime Functions Library	-	✓	✓
Report Generation Supported for AdvancedDatetime fields	-	-	✓
Exportable / Importable Advanced DateTime field customizations	✓	✓	✓

Table 2

System Administration

Advanced Datetime is managed as a **Dispage Extension** through the **Dispage Extension Manager Tool** (<http://www.dispage.com/index.php/products/extension-manager>).

Advanced Datetime can be now seamlessly managed as a Mozilla Firefox Add-on, with additional info on SugarCRM compatibility versions, supported languages, expire dates and whatever is useful to schedule SugarCRM and Advanced Datetime upgrades.

The procedures to **install**, **uninstall** and **upgrade** Advanced Datetime are explained in the **Extension Install Guide** downloadable from the following link:

http://www.sugarforge.org/frs/download.php/6509/Generic_Extension_Install.1.2.pdf

AdvancedDatetime fields

An **AdvancedDatetime** field has the creation mask shown in [Figure 1](#).

The screenshot shows the SugarCRM Administration interface. The top navigation bar includes tabs for Home, Accounts, Contacts, Opportunities, Leads, Activities, and Administration. Below the navigation bar, there's a 'Last Viewed' section with icons for A.D. Importing, 5D Investments, 360 Vacations, 360 Vacations, and Kin. The main content area is divided into two panes. The left pane, titled 'Modules', shows a tree view of modules including Accounts, Labels, Fields, Relationships, Layouts, Subpanels, bas, Bug Tracker, cal, Calls, Campaigns, Cases, Contacts, Documents, and Leads. The right pane, titled 'Edit Fields', contains a form for editing a field. The form has a 'Data Type' dropdown set to 'AdvancedDateTime'. Below it are text fields for 'Field Name' (closed), 'Display Label' (Closed), 'System Label' (LBL_CLOSED), 'Help Text', and 'Comment Text'. There's a 'DateTime Type' dropdown set to 'Date / Time', which is currently open, showing options: 'Date / Time', 'Time', and 'Length Time'. Below this is a 'Default Value' dropdown set to 'Time'. Other fields include 'Mass Update' (set to 'Length Time'), 'Required Field' (checkbox), 'Audit' (checkbox), 'Importable' (dropdown set to 'Yes'), and 'Duplicate Merge' (dropdown set to 'Disabled').

Figure 1

AdvancedDatetime is a new field type that helps the Administrator to meet all the requirements on managing date and time dimensions.

An AdvancedDatetime field can be created both through Studio or Module Builder by selecting *AdvancedDatetime* in the “**Data Type**” selector. A screen similar to the SugarCRM DateTime type appears. Plus, the key-field “**Date / Time Type**” is above the “*Default Value*” entry, and allows the Administrator to select an AdvancedDatetime sub-type among the followings:

- Date / Time
- Time
- LengthTime

The meaning of sub-types is explained in the following paragraph.

AdvancedDatetime Sub-Types

1. Date / Time Sub-Type

Date / Time sub-type allows the user to enter in the same field a date and a time values using the format chosen in the SugarCRM options (see "[AdvancedDatetime Values](#)" Sub-Paragraph).

For instance an AdvancedDatetime field in the Edit View looks like the one in [Figure 2](#), with the default SugarCRM date / time format.

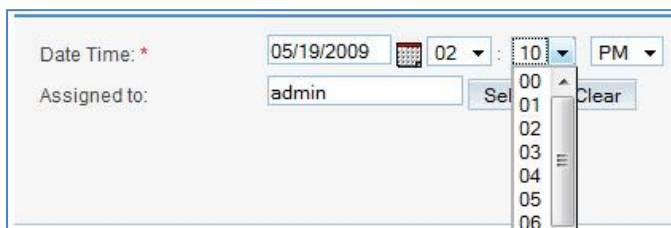


Figure 2

2. Time Sub-Type

The **Time** sub-type is a new sub-type that manages only the time of the day, without specifying a date.

3. LengthTime Sub-Type

LengthTime sub-type is a new sub-type that allows managing a time measure. Compared to the Time sub-type it does not have the limitation of the 24 hours upperbound and can be useful to store time measures across multiple days.

A detail of the LengthTime EditView from an example with an input value of 100 minutes and 16 seconds can be found in [Figure 3](#).

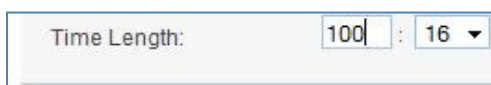


Figure 3

Search for an AdvancedDatetime value

AdvancedDatetime Values can be search for through the Advanced Search Tab.

To achieve this, the last version of the **Enhanced Search extension** must be installed (more details about Enhanced Search can be found at <http://www.dispage.com/products/enhanced-search>).

This functionality is restricted to the **Date / Time** Sub-Type only.

AdvancedDatetime calculated fields

Advanced Datetime provides a way to manipulate AdvancedDatetime fields from inside the PHP Code of a Code type field of Enhanced Studio. The Administrator is then allowed to:

- manage AdvancedDatetime fields in the *\$bean* variable (the AdvancedDatetime fields of the module that the Code field is placed in).
- create new DateTime values.
- calculate other DateTime values from the existing ones.

As parameters of object's methods, DateTime values can be referred to in two ways :

- By specifying the field's name
- By specifying its value

1. AdvancedDatetime Values

When a parameter is passed by specifying the value, the rules explained in this sub-paragraph are followed.

Advanced Datetime uses the date/time format that has been selected in the specific SugarCRM user preferences for both the input and output of the AdvancedDatetime values. This implies that, for example, if the date format "m/d/Y" and the time format "h:i A" have been chosen ¹, a valid DateTime value can be "03/13/2009 11:05 AM". The previous rule applies to **DateTime** and **Time** sub-types.

¹ The syntax of the format used in this Manual is the one of the PHP *date* function that can be found at the PHP official website <http://www.php.net/manual/en/function.date.php>

On the other hand, the **LengthTime** sub-type fields **do not depend on the user preferences** and can have one of the following formats:

- $N_h:i:s$
- $N_i:s$
- N_s

Where N_h , N_i and N_s are number of hours, minutes and seconds, without the 24 / 60 upperbounds.

Thus valid LengthTime values can be either "10:59:59", "999:12:02", "103:32" or "110200", but **NOT** "11:60:60".

2. Initialization

To access the methods of the **AdvancedDatetimeOperations** class, the Code field must contain the following PHP code:

```
global $dtcm;
require_once('custom/include/AdvancedDatetime/AdvancedDatetime.php');
$dtcm = new AdvancedDatetimeOperations($bean);
```

\$dtcm is an instance of a class that provides a user-friendly programming interface that helps to perform the operations detailed below.

3. Access to the AdvancedDatetime values

To get the value of an existing AdvancedDatetime Field of the module, the method **getValue** can be used. For example if the field's name is "Arrival", the code to display the value of the field is:

```
echo $dtcm->getValue('Arrival');
```

If a AdvancedDatetime field of that name does not exist in the current module, an error value is returned:

```
NO DATE / TIME VALUE FOUND !
```

To set the value of a `AdvancedDatetime` Field of the module, or to create a new `AdvancedDatetime` variable, the method **setValue** must be used. For example to change the value of the existing variable “Arrival” (that is supposed to be of sub-type `Time` in this example) to “01/21/2009 20:15”² the code is:

```
$dtcm->setValue('Arrival', '01/21/2009 20:15');
```

Whereas to create a new `TimeLength` “Duration” variable of 132 min. and 15 sec. it is:

```
$dtcm->setValue('Duration', '132:15', 'lengthtime');
```

The last parameter 'lengthtime' could have been omitted in this case, and the sub-type would have been correctly assigned by the method:

```
$dtcm->setValue('Duration', '132:15');  
/* Same result as the example before */
```

4. Calculate values from the existing ones

To perform calculations from the existing `AdvancedDatetime` fields, a set of methods is available in the `AdvancedDatetimeOperation` class.

To add a fixed value to a `DateTime` field, the **addDate** method can be used. For example if the existing field is named “MileStone” and 1 day is to be added to it and echoed to screen, the following code can be used :

```
$dtcm->setValue('AddingValue', '24:00:00');  
$result = $dtcm->addDate('MileStone', 'AddingValue');  
echo $dtcm->getValue($result);
```

Please note that *\$result* does not contain the value itself, but the name of a new `AdvancedDatetime` variable that is automatically created and must be accessed to through the *getValue* method.

The previous code can be simplified by creating the `AdvancedDatetime` variables automatically:

```
echo $dtcm->getValue($dtcm->addDate('MileStone', '24:00'));  
/* Same result as the example before */
```

² All the examples hereafter assume the date format “m/d/Y” and the time format “H:i” (PHP *date* function format)

This is possible thanks to the capability to correctly guess the AdvancedDatetime sub-type (using the rules described in the Sub-Paragraph “[AdvancedDatetime Values](#)”).

Nested calculations are also allowed. For example, the difference between two DateTime variables (in the example *FadeIn* and *FadeOut*) can be added to a third DateTime (in the example *Ciak*).

```
echo $dtcm->getValue($dtcm->addDate(  
    'Ciak',  
    $dtcm->subDate('FadeOut','FadeIn')  
));
```

The following paragraph contains a complete reference on AdvancedDatetime methods.

AdvancedDatetimeOperations Method Reference

The following is a list of the available operators.

1. **getValue**

Synopsis:

field_value \$dtcm->getValue(*field_name*)

Where :

- *field_value* is the value of the field named *field_name*.
- *field_name* is the name of the field to get the value.

Description:

Returns the value associated to the specified *field_name*. It can be used to retrieve the values of a AdvancedDatetime field of the module, or either a calculated field that has been automatically generated by a AdvancedDatetimeOperation method.

If no AdvancedDatetime field is associated to *field_name*, the following error value is returned:

```
NO DATE / TIME VALUE FOUND !
```

2. setValue

Synopsis:

```
void $dtcm->setValue (field_name, field_value[, field_sub_type])
```

Where :

- *field_name* is the name of the field to set the value.
- *field_value* is the value to set into *field_name*.
- *field_sub_type* (optional) is the sub-type of the field ('datetime', 'time' or 'lengthtime').

Description:

The method can be used either to create a new AdvancedDatetime field named *field_name* with the initial value set to *field_value*, or to modify the value of an existing field.

The *field_sub_type* parameter is the sub-type of the field: it can be used to force the class to assign the sub-type to the field. If it is not specified, the application guesses the sub-type from the format of the initial value. If it cannot retrieve the sub-type from the format, the following error value is returned:

NO DATE / TIME VALUE FOUND !

3. addDate

Synopsis:

```
field_name $dtcm->addDate (dtc_value, dtc_length )
field_name $dtcm->addDate (dtc_value, array(dtc_length_1..dtc_length_n) )
```

Where :

- *field_name* is the name of the field where the result is put.
- *dtc_value* is the field to sum the value to. It can be either a string containing the name of the field or the value itself.
- *dtc_length* (*dtc_length_i*) is the *LengthTime* field to sum to *dtc_value*.

Description:

Adds a value or a list of values of sub-type *LenghtTime* (possibly derived by a previous calculation) to any AdvancedDatetime field.

If the second parameter is not of sub-type *LenghtTime*, the following error value is returned:

DATE / TIME TYPE MISMATCH ERROR !

4. **subDate**

Synopsis:

field_name \$dtcm->subDate (*dtc_value1*, *dtc_value2*)

Where :

- *field_name* is the name of the field where the result is put.
- *dtc_value1* is the field to subtract the value from. It can be either a string containing the name of the field or the value itself.
- *dtc_value2* contains the value to subtract. It can be either a string containing the name of the field or the value itself.

Description:

Subtracts the value of the *dtc_value2* (of any AdvancedDatetime sub-type) from the *dtc_value1*.

5. **floorDate**

Synopsis:

field_name \$dtcm->addDate (*dtc_value*, *dtc_length*)

Where :

- *field_name* is the name of the field where the result is put.
- *dtc_value* is the field to round down to the nearest multiple of *dtc_length*.
- *dtc_length* contains the value that *dtc_value* has to be scaled to.

Description:

Rounds down *dtc_value* to the highest multiple of *dtc_length* less than *dtc_value*. For example if *dtc_value* is equal to "21/12/2009 17:14" and *dtc_length* is "05:00", it returns "21/12/2009 17:10".

If the second parameter is not of sub-type *LenghtTime*, the following error value is returned:

DATE / TIME TYPE MISMATCH ERROR !

6. ceilDate

Synopsis:

field_name \$dtcm->addDate (*dtc_value*, *dtc_length*)

Where :

- *field_name* is the name of the field where the result is put.
- *dtc_value* is the field rounds down to the nearest multiple of *dtc_length*.
- *dtc_length* contains the value that *dtc_value* has to be scaled to.

Description:

Rounds up the *dtc_value* to the lowest multiple of *dtc_length* greater than *dtc_value*. For example if *dtc_value* is "21/12/2009 14:03" and *dtc_length* is "30:00", it returns "21/12/2009 14:30".

If the second parameter is not of sub-type *LenghtTime*, the following error value is returned:

DATE / TIME TYPE MISMATCH ERROR !

AdvancedDatetime Method Parameters Table

The following table shows the sub-types of the return value for each combination of input value of each method.

Field #2		DateTime	Time	LengthTime
Method	Field #1			
addDate	DateTime	Type M. Error	Type M. Error	DateTime
	Time	Type M. Error	Type M. Error	Time
	LengthTime	Type M. Error	Type M. Error	LengthTime
subDate	DateTime	LengthTime	LengthTime	LengthTime
	Time	LengthTime	LengthTime	LengthTime
	LengthTime	LengthTime	LengthTime	LengthTime
floorDate	DateTime	Type M. Error	Type M. Error	DateTime
	Time	Type M. Error	Type M. Error	Time
	LengthTime	Type M. Error	Type M. Error	LengthTime
ceilDate	DateTime	Type M. Error	Type M. Error	DateTime
	Time	Type M. Error	Type M. Error	Time
	LengthTime	Type M. Error	Type M. Error	LengthTime

Table 3

Webography

Advanced Datetime on Dispage website

Advanced Datetime extension is currently hosted in Enhanced Studio 3.1 homepage at Dispage Website:

<http://www.dispage.com/index.php/products/enhanced-studio>

(New) A useful wiki section can be found at:

http://www.dispage.com/wiki/Advanced_Datetime_Type

Advanced Datetime on SugarForge

The official Advanced Datetime page on SugarForge can be found in the Enhanced Studio project's page

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

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

