

Calendar Template Manual

1.0



dispage

Calendar Template Manual
Release 1.0.01 – Edition I, 2010
Copyright © 2008 - 2010 Dispage - Patrizio Gelosi, All Rights Reserved
This document is subject to change without notice.

Disclaimer

The Calendar Template 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
Calendar Template	9
How to create a Custom Calendar Module.....	10
Usage of a Custom Calendar Module	12
Creation of Aggregates and Sub-Aggregates	13
Samples / Scenarios	17
1. Simple Invoice Calendar Module	17
2. Work Timesheet Calendar Module	20
Webography.....	25
Calendar Template on Dispage website	25
Calendar Template on SugarForge.....	25

Preface

Calendar Template is an Enhanced Studio sub-extension to create custom Calendar modules. Used in conjunction with Enhanced Studio, allows to perform any customization involving time measures.

Preliminary readings:

- **Enhanced Studio Manual**

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

- **Advanced Datetime Manual**

<http://www.dispage.com/support/documentation/advanced-datetime-manual>

Overview

Calendar Template extension adds to SugarCRM Module Builder a new template to create custom modules that combines the functionalities of SugarCRM Calendar Module and the customization capabilities of Module Builder.

Additionally, a wide range of **Time Aggregate Measures**, calculated from the module's field values, can be displayed in the views of Custom Calendar Modules.

Calendar Template is available in Enhanced Studio from 3.0 release; from Enhanced Studio 3.1 release it has been splitted into a separate extension.

Compatibility

Calendar Template 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 Calendar Template 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
SugarCRM 5.5.x SugarCRM 6.0.x	CE / PRO / ENT	Calendar Template 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 Calendar Template 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 / Calendar Template.

Enhanced Studio / Calendar Template Rel. / Ver.	Enhanced Studio 3.0 DEMO	Enhanced Studio 3.0 FULL	Calendar Template 1.0 FULL
Features			
Calendar Template support	✓	✓	✓
Time Aggregate Measures for Custom Calendar Modules	-	✓	✓
Exportable / Importable Custom Calendar Modules	✓	✓	✓

Table 2

System Administration

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

Calendar Template 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 Calendar Template upgrades.

The procedures to **install**, **uninstall** and **upgrade** Calendar Template 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

Calendar Template

Calendar Template allows to create modules having the same look of the SugarCRM Calendar module, but may be customized at will.

The Administrator can now create a Custom Calendar Module (CCM) with any fields, included AdvancedDatetime and Code fields that allow calculation on the time dimensions and measures used by the module itself.

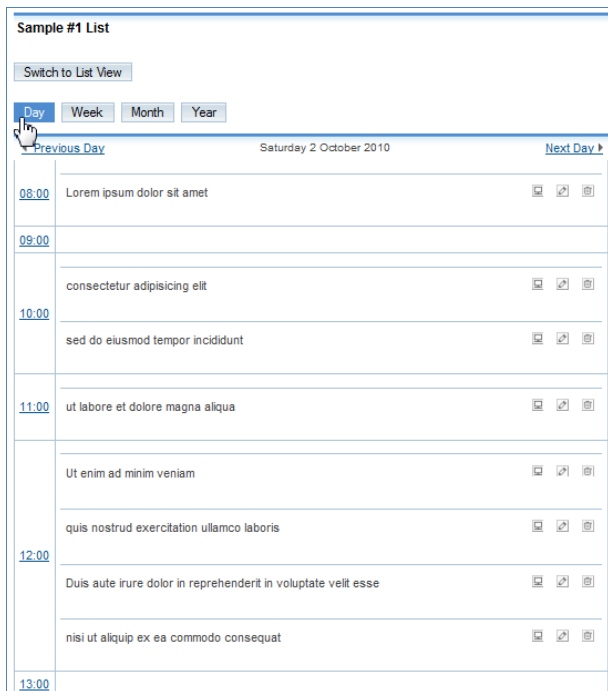


Figure 1

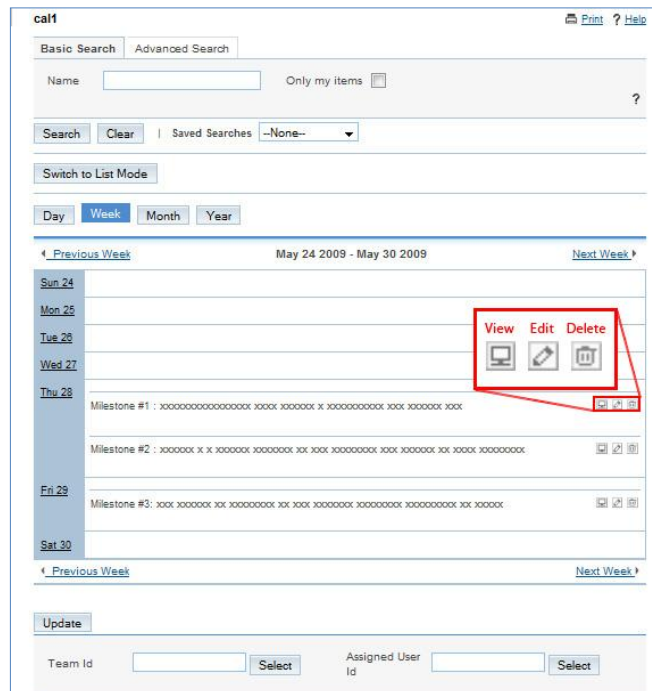


Figure 2

Custom Module, once deployed, is available both in the **Classic SugarCRM Mode** and in the **Calendar Mode**, which lets the user create/view/edit/delete the elements from the common calendar views:

- Day ([Figure 1](#))
- Week ([Figure 2](#))
- Month ([Figure 4](#))
- Year ([Figure 5](#))

Also, the **Aggregate Measure** functionality is provided.

Through aggregates, Administrator can define special Code fields in CCM that can be executed and viewed on each specific Calendar view.

Each view has a main class of **Total Aggregate Measures** that is shown at its bottom (**Figure 4**). The same figure shows how the Week, Month and Year views support special **sub-aggregates**.

The complete set of Aggregates and Sub-Aggregates available for each view can be found in the following table.

Aggregate \ View	Day	Week	Month	Year
Day	✓	✓	✓	-
Week	-	✓	✓	-
Month	-	-	✓	✓
Year	-	-	-	✓

Table 3

How to create a Custom Calendar Module

Calendar Template is simply another Module Template choice appearing at the Template selection during the Module Creation process.

Thus, to create a new CCM, the Calendar Template must be selected as shown in the **Figure 3**. The new module has all the fields of a basic SugarCRM module, and in addition the **"date_start"** default field (an AdvancedDatetime field of *Date/ Time* sub-type labelled *"Date Time"*) which is the

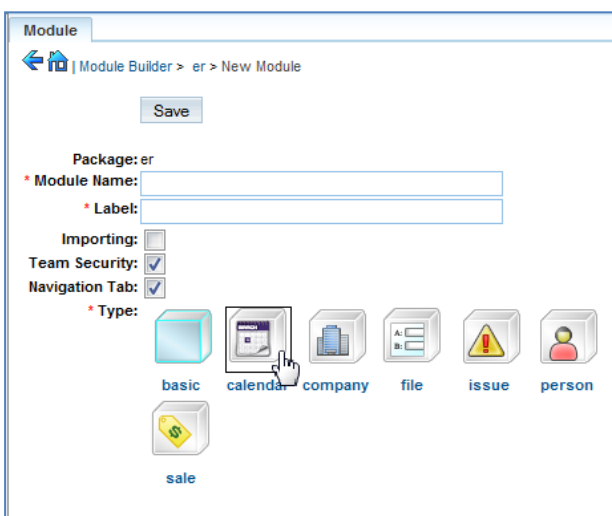


Figure 3

key-dimension used to place the element in any Calendar view and to aggregate it properly.

From now on, the module is managed exactly like the other SugarCRM modules: fields can be created / modified / deleted.

AdvancedDatetime and Code fields are particularly useful to CCMs, as is shown later and in the Samples **Simple Invoice Calendar Module** and **Work Timesheet Calendar Module**.

The only relevant difference from a SugarCRM Module is that **the Fields added in the ListView are also shown in the Element's cell of the Calendar View**.

After that all the customizations are performed, the CCM can be **Deployed**, **Published** and **Exported** like any other SugarCRM module and it is ready to be used.

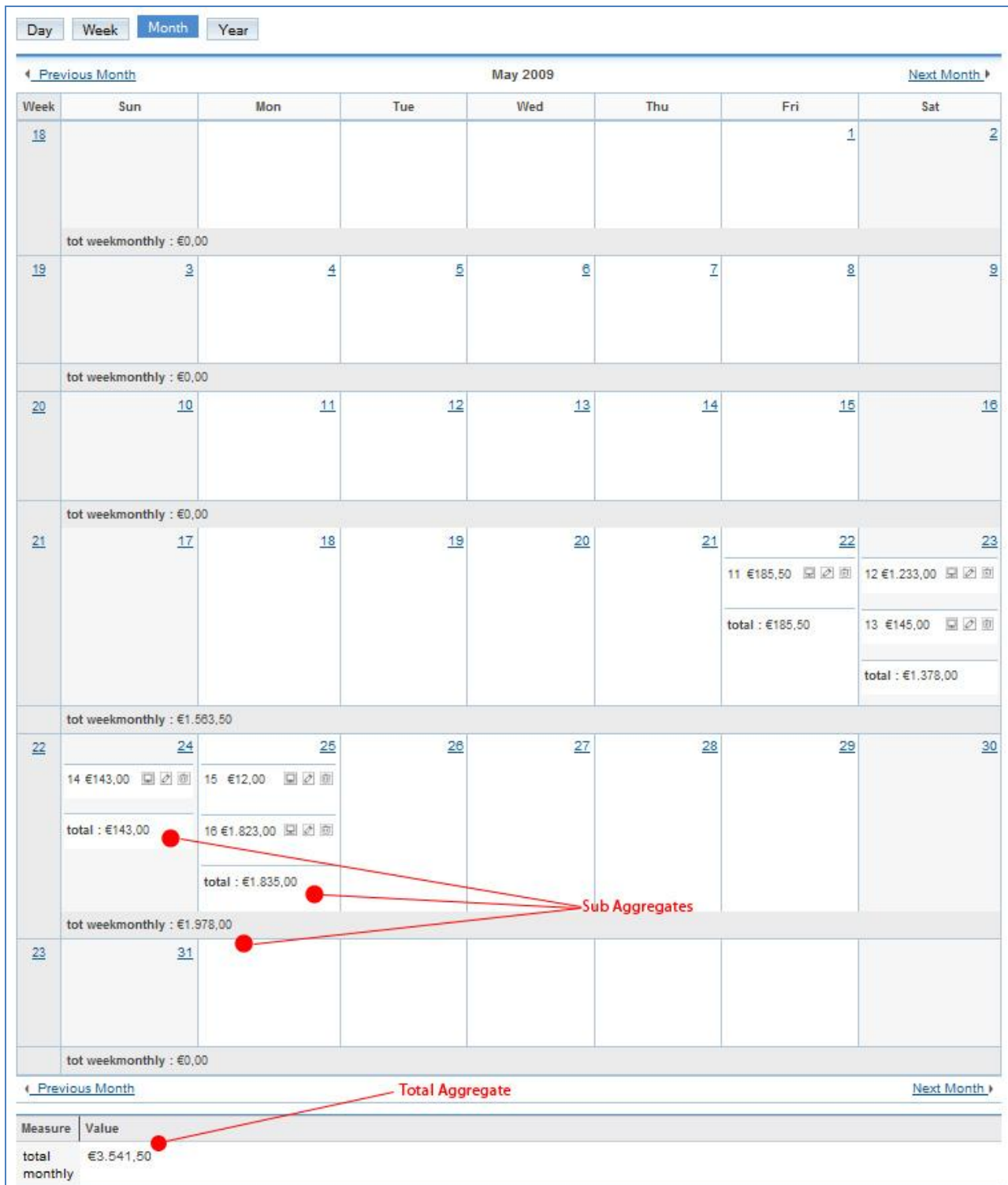


Figure 4

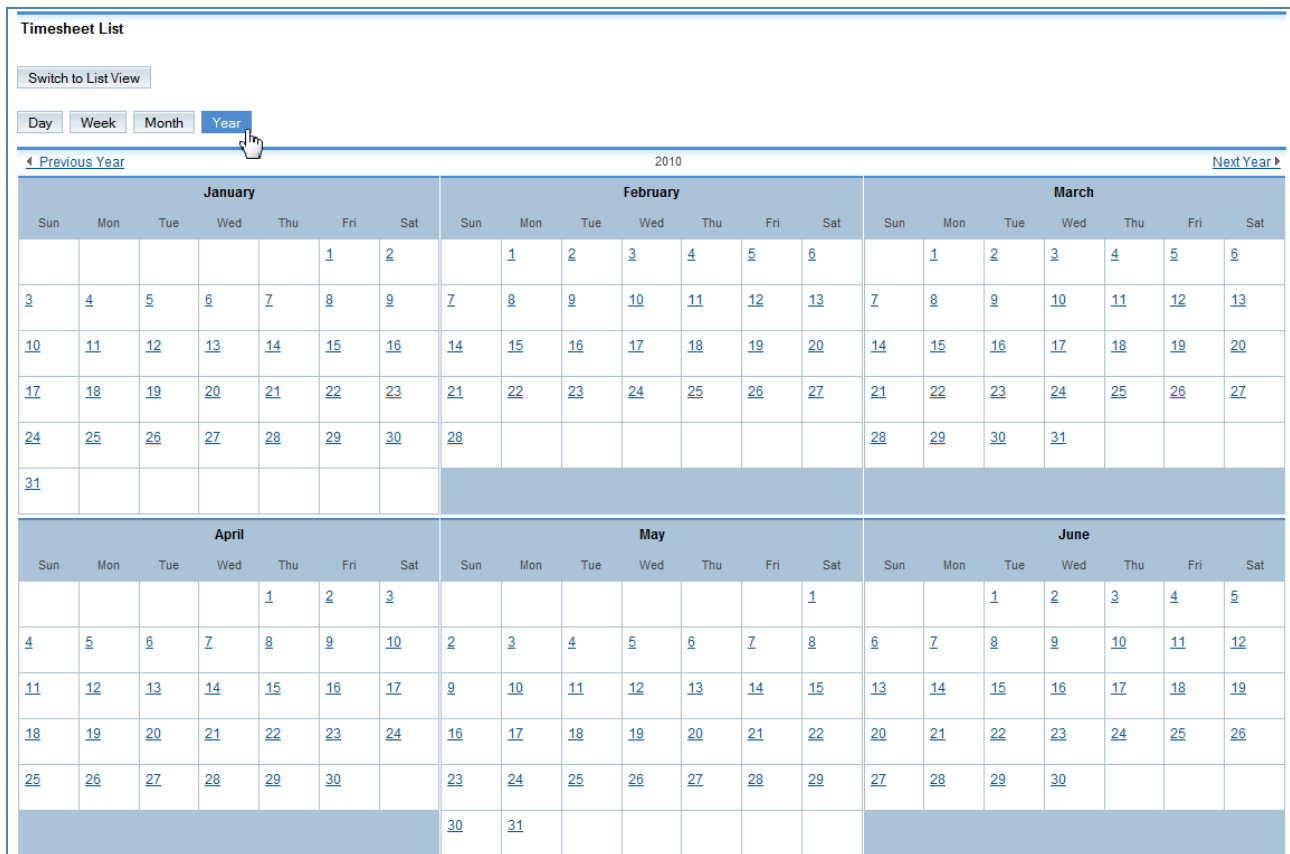


Figure 5

Usage of a Custom Calendar Module

Custom Calendar Modules (CCM) bring together the key points of default SugarCRM Modules and the SugarCRM Calendar Module.

Calendar Views of CCM, shown in [Figure 1](#), [Figure 2](#), [Figure 4](#) and [Figure 5](#), are described in detail below.

1. Like a default SugarCRM Module, CCMs have the **Basic / Advanced Search Tabs** in their header ([Figure 2](#)) to allow the user to filter the elements of the Calendar Main Panel.
2. Under the tabs, the button **“Switch to List Mode”** ([Figure 2](#)) can be found. Clicking it, user can view the module as a default SugarCRM one. This feature might come in use if there are elements scattered in a wide range of time and need to be collected in a single view (the default SugarCRM ListView).

3. Below is the Main Calendar Panel. User can easily navigate it by changing the time detail level (for example clicking the **Day**, **Week**, **Month** or **Year** button or clicking the link to a specific week in the left column of the Month View) or by moving the timeslice showed (clicking the “**Previous/Next Day/Week/Month/Year**” link).
Each element can be **viewed / modified / deleted** by pressing the icons placed on the right of the Elements cell (see [Figure 2](#)). The View/Edit/Deletion process is the same of the default Sugar Modules.
New elements can be **created** with the Create-Field procedure of all the SugarCRM Modules or by clicking the link to a specific Hour in the Day view.
4. Under the Calendar Panel, the global **Aggregate Measures** added by the Administrator to the specific Time detail level (as explained in the next paragraph) can be found. Other **Sub-Aggregate Measures** can also be found at the bottom of each element’s cell in the Main Calendar Panel as shown in [Figure 4](#).
5. At the bottom of the window, the default Module’s panel for **Mass Updating**.

Creation of Aggregates and Sub-Aggregates

Aggregates and **Sub-Aggregates** are Code fields whose name follows a particular syntax.

Administrator can add these fields to the CCM through Module Builder and, after having deployed the Module, Aggregates are placed in the Views exactly at the level specified in the field name. In the calendar View, the label of the Code field appears followed by the Aggregate Value, which is the result value calculated in the PHP Code execution. Of Course being it a Code field it can also be used for advanced customizations: for example to append Buttons, Images and other elements to the bottom of the Module.

The same Code Aggregate field can also be used in more than one Calendar View.

WARNING

Aggregate and Sub-Aggregates Code fields **do not need to be added to any Layout** to be viewed.

1. Name Syntax for Aggregates and Sub-Aggregates

To create an **Aggregate** for a specific Calendar View, the name of the field **must simply contain the adverb of time related to that View**, in English language, separated from the rest of the string by an underscore (_).

For example a 'total' field that should appear at the bottom of month views could be named

```
total_monthly
```

One field can also be viewed in multiple calendar views. To achieve this, the adverbs of time related to all the views must be inserted in the field name, using the underscore as separator. For example an "income" field can be required in the Week, Month and Year view: the name of the field would be

```
income_weekly_monthly_yearly
```

The syntax for the **Sub-Aggregates** is similar: before the adverb of the main aggregate, telling the application which view the field must appear in, an additional word selected among "**day**", "**week**" or "**month**" must be placed without the underscore separator. This word specifies which sub-dimension the Sub-Aggregate have to be placed in.

The syntax of an Aggregate field's name has the following definition:

```
chunk0[_chunk1[_chunk2 ... [_chunkn]]]
```

where

```
chunki = <Name of the field>
        | /* OR */
        [day|week|month] (daily|weekly|monthly|yearly)
```

For example, if the Administrator wants to create a Code field named "amount" which has to be viewed as Aggregate in the Day view and as a Day Sub-Aggregate in the Week and Month views, the name to be assigned in Module Builder is

```
amount_daily_dayweekly_daymonthly
```

2. PHP Code Content for Aggregates and Sub-Aggregates

Working Examples on how to manage Aggregates and Sub-Aggregates can be found in Sample Section [Samples / Scenarios](#).

To calculate the Aggregates and Sub-Aggregates values, two methods can be followed.

2.1. Calculate the value using *\$bean* variable

The *\$bean* variable is available from inside the Code fields in CCM, same as for the default ones.

In the examples hereafter we assume that the aggregation function is the SUM.

As for the **dayweekly** and **daymonthly** Sub-Aggregates, all the values of the day are available in the *sugar_bean* field of the objects collection of the *\$bean* array.

For example a “dayweek” Sub-Aggregate of a field named “total” can be evaled with a Code like the following:

```
foreach($bean as $a) {
    $sum += $a->sugar_bean->total;
}
```

Others Aggregates and Sub-Aggregates have more levels of values to be summed, stored in the field *acts_arr*.

For example to calculate the total in the monthly Aggregate of the previous example, the Code is:

```
foreach($bean as $v) {
    if (isset($v->acts_arr)) {
        foreach ($v->acts_arr as $b) {
            foreach ($b as $a) {
                $total += $a->sugar_bean->total;
            }
        }
    }
}
```

2.2. Sum the values calculated in other Fields/Sub-Aggregates

The method described in the previous paragraph permits to calculate the Aggregates and Sub-Aggregates each time it's needed without collecting the partial results.

To save calculation time the partial Sub-Aggregates results might be saved in GLOBALS variables, and then summed in higher-level Aggregates.

Carrying on with the “total” example, if the day and week Sub-Aggregates are required in the Month view, the following procedure may be followed.

1. Create a *partial1_daymonthly* Sub-Aggregate as described in the previous paragraph, for the total amount of each day of the month.

The final value has to be both echoed and stored in GLOBALS variable:

```
global $tot_week, $tot;
$sum = 0;
foreach($bean as $a) {
    $sum += $a->sugar_bean->total;
}
if (!isset($tot_week)) $tot_week = 0;
$tot_week += $sum;
$tot += $sum;
echo $sum
```

2. Create a *partial2_weekmonthly* Sub-Aggregate that simply echoes the partial value calculated and reset it:

```
global $tot_week;
echo $tot_week;
$tot_week = 0;
```

WARNING

The procedure explained in this paragraph has a key-point: it needs **that all the fields are evaled in the exact order explained** to work properly.

Generally speaking, the eval order must be the following:

1. first, the Low-level Sub-Aggregates
2. then, the High-level Aggregates
3. finally, the Total Aggregate

To have the expected execution order followed, **the Sub-Aggregates and Aggregates fields must be created in the exact order they have to be calculated.**

Samples / Scenarios

1. Simple Invoice Calendar Module

Scenario: build a simple Module that allows to store simple Invoice (with only a number and an amount) with Day / Week / Month Aggregates and Sub-Aggregates.

Procedure:

- i. Create a new package and a new CCM in module Builder as Explained in the paragraph “[How to create a Custom Calendar Module](#)”
- ii. Add the following fields **in this exact order** (Logic Hook is assumed to be **Default**):

ii.1. **Data Type:** TextField
Field Name: invoice_id

ii.2. **Data Type:** Currency
Field Name: total

ii.3. **Data Type:** Code
Field Name: tot_daymonthly_dayweekly
DB Type: Non-DB
Code:

```
global $amount_week, $file_tot;
if (isset($bean->date_start)) return;
$sum = 0;
foreach($bean as $a) {
    $sum += $a->sugar_bean->total;
}
if (!isset($amount_week)) $amount_week = 0;
$amount_week += $sum;
$file_tot += $sum;
echo currency_format_number($sum)
```

ii.4. **Data Type:** Code

Field Name: tot_weekmonthly

DB Type: Non-DB

Code:

```
global $amount_week;
echo currency_format_number($amount_week);
$amount_week = 0;
```

ii.5. **Data Type:** Code

Field Name: tot_monthly

DB Type: Non-DB

Code:

```
foreach($bean as $v) {
    if (isset($v->acts_arr)) {
        foreach ($v->acts_arr as $b) {
            foreach ($b as $a) {
                $total += $a->sugar_bean->total;
            }
        }
    }
}
```

iii. Add the fields *invoice_id* and *total* to the following Layouts:

- Editview
- DetailView
- ListView

iv. Deploy the package.

The result is an Invoice Module with partial and total amounts.

A Week view can be seen in [Figure 6](#) and a Month view in [Figure 4](#).

Invoices
[Print](#)
[Help](#)

Basic Search

Advanced Search

Name
Only my items ☐
?

Search

Clear

Saved Searches
--None--

Switch to List Mode










Day

Week

Month

Year

[Previous Week](#)
May 24 2009 - May 30 2009
[Next Week](#)

<u>Sun 24</u>	14	€143,00	  
total : €143,00			
<u>Mon 25</u>	15	€12,00	  
	16	€1.823,00	  
total : €1.835,00			
<u>Tue 26</u>			
<u>Wed 27</u>			
<u>Thu 28</u>			
<u>Fri 29</u>			
<u>Sat 30</u>			

[Previous Week](#)
[Next Week](#)

Figure 6

2. Work Timesheet Calendar Module

A. Module Deployment

Scenario: build a Basic Timesheet Management Module for the Employees of a Company. The module automatically calculates and shows the **Worked Time Amount** for each Employee or group of them, grouped for Day, Week and Month.

Procedure:

i. Create a new package and a new CCM in module Builder as explained in the paragraph “[How to create a Custom Calendar Module](#)”

ii. Add the following fields in this exact order (Logic Hook is assumed to be **Default** for all):

ii.1. **Data Type:** DropDown

Field Name: direction

Drop Down List:

Name: direction_list

Values: Entry, Exit

ii.2. **Data Type:** Code

Field Name: amount_daily

DB Type: Non-DB

Code:

```
require_once('custom/include/AdvancedDatetime/AdvancedDatetime.php');
$dtcm = new AdvancedDatetimeOperations($bean);
$sum = '00:00';
foreach($bean as $v) {
    if (isset($v->acts_arr)) {
        foreach ($v->acts_arr as $a) {
            if ($a[0]->sugar_bean->direction == 'entry') {
                $entry = $a[0]->sugar_bean->date_start;
            }
            elseif ($a[0]->sugar_bean->direction == 'exit') {
                if ($entry) {
                    $sum = $dtcm->addDate(
                        $dtcm->subDate($a[0]->sugar_bean->date_start, $entry),
                        $sum
                    );
                }
                $entry = '';
            }
        }
    }
}
echo $dtcm->getValue($sum);
```

ii.3. Data Type: Code

Field Name: amount_weekly_monthly

DB Type: Non-DB

Code:

```
global $dtcm, $time_amount;
require_once('custom/include/AdvancedDatetime/AdvancedDatetime.php');
$dtcm = new AdvancedDatetimeOperations($bean);
$time_amount = '00:00';
foreach($bean as $v) {
    if (isset($v->acts_arr)) {
        foreach ($v->acts_arr as $b) {
            foreach ($b as $a) {
                if ($a->sugar_bean->direction == 'entry') {
                    $entry = $a->sugar_bean->date_start;
                }
                elseif ($a->sugar_bean->direction == 'exit'){
                    if ($entry) {
                        $time_amount = $dtcm->addDate(
                            $dtcm->subDate($a->sugar_bean->date_start, $entry),
                            $time_amount
                        );
                    }
                    $entry = '';
                }
            }
        }
    }
}
echo $dtcm->getValue($time_amount);
```

ii.4. Data Type: Code

Field Name: amount_daymonthly_dayweekly

DB Type: Non-DB

Code:

```
global $dtcm, $amount_week;
if (isset($bean->date_start)) return;
$sum = '00:00';
foreach($bean as $a) {
    if ($a->sugar_bean->direction == 'entry') {
        $entry = $a->sugar_bean->date_start;
    }
    elseif ($a->sugar_bean->direction == 'exit') {
        if ($entry) {
            $sum = $dtcm->addDate(
                $dtcm->subDate($a->sugar_bean->date_start, $entry),
                $sum
            );
        }
        $entry = '';
    }
}
if (!isset($amount_week)) $amount_week = '00:00';
$amount_week = $dtcm->addDate($sum, $amount_week);
echo $dtcm->getValue($sum)
```

ii.5. Data Type: Code

Field Name: amount_weekmonthly

DB Type: Non-DB

Code:

```
global $dtcm, $amount_week;  
if (isset($bean->date_start)) return;  
if (!isset($amount_week))  
    echo '00:00';  
else  
    echo $dtcm->getValue($amount_week);  
$dtcm->setValue($amount_week, '00:00');
```

iii. Add the fields “*direction*” and “*assigned_user_id*” (“*Assigned to*” or “*User*”) to the Layouts:

- Editview
- DetailView
- ListView
- SearchView

iv. Optionally add the date_start field (“Date Time”) to the ListView Layout to view the exact Time entry/exit in the sheet.

v. Deploy the package.

B. Module Usage

The Timesheet Module that has been deployed provides the **basic functionalities of a Work Timesheet management**:

- i. An element of the Work Timesheet module is an Entry or Exit of a SugarCRM user. For example if Sarah has started to work at 9:00AM of 05/18/2009 and stopped at 12:00PM, then from 2:02PM to 6:57PM, four elements must be created in the Timesheet module (one for each entry/exit)

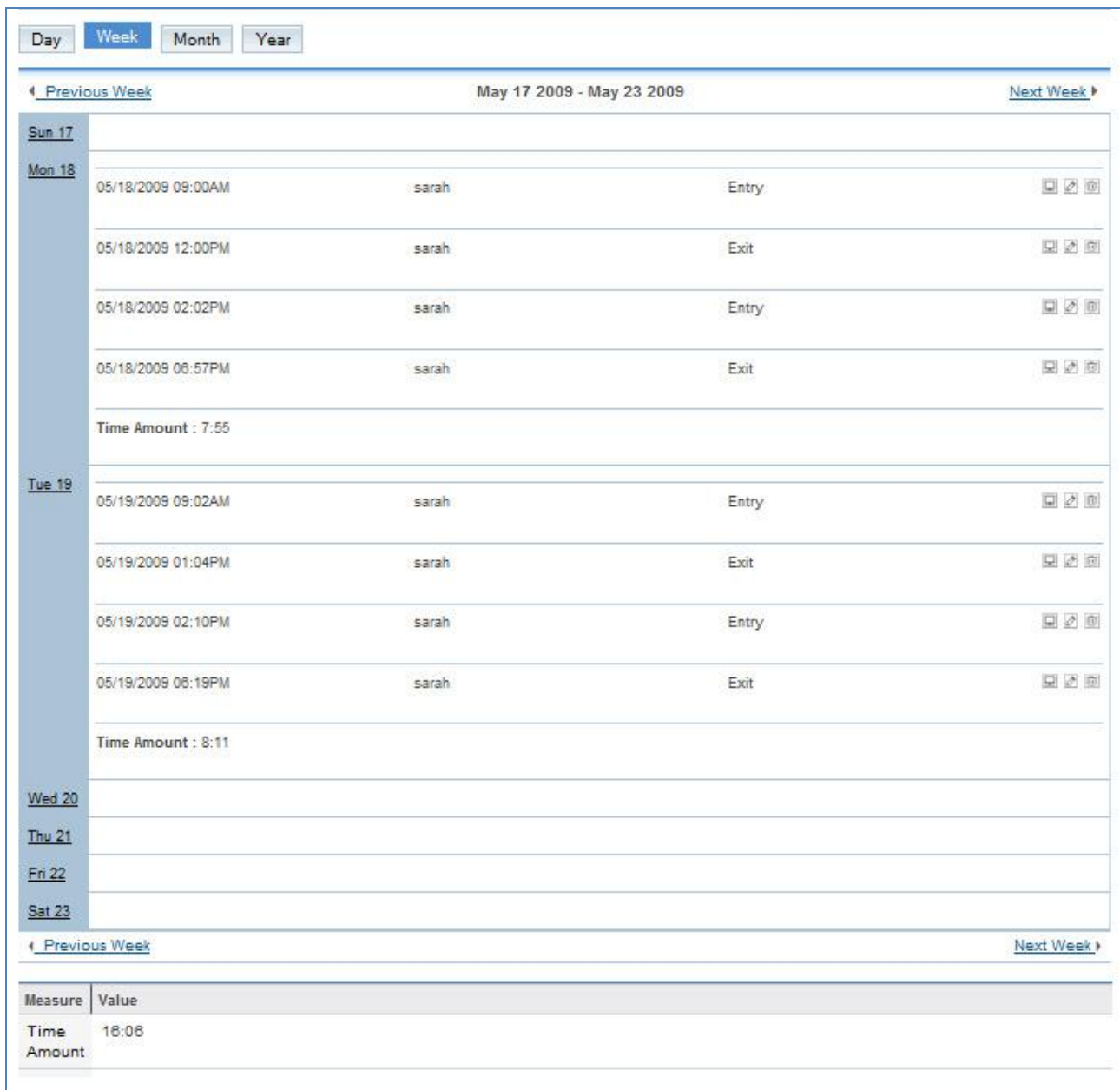


Figure 7

- ii. The Worked Time Amount is available as Aggregate and Sub-Aggregate in the Day, Week and Month views.

The **Week view** of the Timesheet Module would be something like what shown in **Figure 7**. In the figure both the Daily Sub-Aggregates and the Weekly Aggregate are correctly reported. The **Month view**, reporting the Daily and Weekly Sub-Aggregates and Monthly Aggregate, can be seen in **Figure 8**.

























Time Amount : 00:00							
21	17	18	19	20	21	22	23
		05/18/2009 sarah Entry   09:00AM	05/19/2009 sarah Entry   09:02AM				
		05/18/2009 sarah Exit   12:00PM	05/19/2009 sarah Exit   01:04PM				
		05/18/2009 sarah Entry   02:02PM	05/19/2009 sarah Entry   02:10PM				
		05/18/2009 sarah Exit   06:57PM	05/19/2009 sarah Exit   06:19PM				
		Time Amount : 7:55	Time Amount : 8:11				
Time Amount : 16:06							
22	24	25	26	27	28	29	30
		05/25/2009 sarah Entry   09:03AM			05/28/2009 sarah Entry   11:09AM		
		05/25/2009 sarah Exit   01:15PM			05/28/2009 sarah Exit   04:44PM		
		Time Amount : 4:12			Time Amount : 5:35		
Time Amount : 9:47							
23	31						
Time Amount : 00:00							
Previous Month				Next Month			
Measure	Value						
Time Amount	25:53						

Figure 8

- iii. The Worked Time Amount can be calculated for a single Employee only, for a group of them or for all of them (total Worked Time Amount). To achieve it, the Basic/Advanced Search "Assigned to" filter can be used.

Webography

Calendar Template on Dispage website

Calendar Template 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/Calendar_Template

Calendar Template on SugarForge

The official Calendar Template 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.

