oCal v2.0 & jsoCal v1.1

by Brainy Data Limited

About the software

oCal & jsoCal are enhanced calendar components that provide alternatives to the calendar component provided by Omnis Software. The main difference is that it allows you to attach a list of events and view these events in various display modes, such as single day, week, two weeks, day plan, the original month view or annual view.

Both oCal & jsoCal implement drag & drop for moving and the sizing of events and the appearance is highly configurable. In addition, the desktop version allows the dropping of external Omnis objects onto the calendar. This is currently not supported in the JS-client version.

Essentially then, oCal and jsoCal are list controls where each event displayed relates to a row in the list. Selecting an event will select the appropriate line in the list and generate an event. The supported drag&drop features directly manipulate the associated list data and when doing so will generate additional events providing details of the changes that were made by the user.

Latest Changes

For a list of the latest changes please see the section “oCal: Changes History” and “jsoCal: Changes History” towards the end of this chapter.

Installing the Software

The following is a brief description of the various component parts of this software and how to install them.

oCal Desktop Control

As a general rule, the downloaded folder containing the software will be organised so that you may follow these generic steps.

There are a number of components to install and the component names vary between platforms. It typically contains folders for different versions of studio, i.e. *studio_810, studio_1000, studio_1010 and studio_1020*, referring to Studio versions 8.1.0, 10.0, 10.1 and 10.2 respectively.

1. Open the appropriate Omnis Studio folder. Always use the latest version that is not later in version than your version of Omnis Studio. For example, for Studio 8.1.7 you would use components from the *studio_810* folder and **not** the *studio_1000* folder.

The Omnis Studio folder will contain an additional folder called *XCOMP*. This may or may not be suffixed with a three letter platform identifier, i.e. *_mac, _win* or *_lin*.

2. Copy the components from inside XCOMP to your Omnis installation. You will find identical named folders inside the Omnis application support folder (macOS) or

executable folder (winOS). On Mac OSX you may need to create this folder inside the *~/Library/Application Support/Omnis/Omnis Studio {version}/* folder. On windows, you will find the XCOMP folder alongside the Omnis executable.

You can open the example library directly from the o*Cal* folder.

jsoCal JS-Client/Server

The jsoCal component consists of the following.

1. JSON control description:
   This provides the IDE interface for designing the control. Copy the contents of the folder *INTO html/controls* to the "controls" folder inside the Omnis "hmtl" support folder.
2. CSS:
   This file specifies the appearance of jsoCal. Copy the contents of the folder *INTO html/css* to the folder "css" located within the Omnis "hmtl" support folder.
3. Javascript:
   This file contains the control's main source code. Copy the contents of the folder *INTO html/scripts* to the folder "scripts" located within the Omnis "hmtl" support folder.
4. Strings:
   This file contains the control's main text strings that have been exported from the string table editor. So you do not have to export them yourself, copy the contents of the folder *INTO html/strings* to the folder "strings" located within the Omnis "hmtl" support folder.
5. ctl_jsocal:
   This folder contains the images for the control. Copy the folder inside *INTO html/images* into the folder "images" located within the Omnis "hmtl" support folder.

Once you have installed the above files and folders in the correct locations, please update the file jsctempl.htm inside the Omnis "hmtl" support folder by copying the lines with the comment "JSOCAL" from the supplied jsctempl.htm file to yours. The line referencing "fontawesome.com" is only required to run the examples. Some of the event data embeds SVG icons.

Please read the sections *Known Problems* and *Important Differences* below before running the examples. Once you have read them, run the examples which contain a remote form for the purpose of testing and familiarising yourself with the control.

jsoCal Known Problems and Limitations

This section lists all the problems we could think of at the time of writing this. There may be other problems, that we have missed.

First Install (IMPORTANT)

When installing jsoCal in a new Omnis tree and when running the example for the first time, there appears to be an issue with some of the jsoCal properties in the example remote form. All or some of the properties that have constants applied become corrupt and link to constants not

associated with jsoCal (this has been reported to Omnis, case ST/EC/1728). This causes the example not to function correctly. We have remedied this by checking and correcting them in the example's Startup library, but just in case, before testing the remote form, you should check the following properties in the property inspector and make sure they have the correct constants applied as specified below:

$digestoptions	kJSOCalDigestOptCountEvents
$firstday	kJSOcalMonday
$viewmode	kJSOcalViewDay
$viewtype	kJSOCalViewTypeTimed
$shownavbuttons	kJSOCalNavAll (new in version 1.1)

List View

This view does not deal with multi-selection lists at this stage. We are open to implementing this in a future release should the demand exist.

Sizing and Moving Events

Events can be moved and sized in the day view and to some extent in the month view. Events can also be sized in the group view, but they cannot be moved between groups (layers). Which layers an event belongs to is best decided using an interface like the one provided in the example.

Touch Screens

We have not tested jsoCal with touch screens. We suspect that sizing events may not work using the touch screen, but they can be resized using your own interface to change the start and end date/times of the selected event.

Important Differences (between oCal and jsoCal)

Although we tried to create the Java script calendar control in the image of its desktop ancestor, some minor functional differences were unavoidable.

Multi-day events

Events that cross days are now always displayed in the all-day pane when the calendar is displaying the day view mode.

List handling

In jsoCal changes made to an event via the control interface, i.e. resizing an event, are directly applied to the Omnis list line associated with the event. The desktop version required the developer to make the appropriate changes in response to the appropriate events. This is not required in the js client version.

View Modes

Setting the view is now achieved via two properties, $viewmode and $viewtype.

Appearance

The jsoCal appearance is now mostly specified via CSS. However, each event (in the Omnis list) may also specify CSS background specifically for the event. See $columnbackgroundcss in the chapter “jsoCal Reference”.

The layout for the content for events is specified via templates. See the properties $templatelist and $columntemplate in the property inspector. The $columntemplate property specifies the column name in the Omnis list that provides the template name or raw HTML for each event. The property $templatelist takes a list of xhtml and css templates for the events, based on view settings or directly linked to an event via the $columntemplate property. The events list may also provide CSS background properties for each event in its list via the a column specified by $columnbackgroundcss.

Deploying your software

Please refer to the license agreement for rules on deployment.

Documentation

This documentation describes the functionality provided by the external. As a minimum you should read the chapters “Feature Overview” and “Designing oCal” or “Designing jsoCal”.

Table of Contents

oCal: Changes History

Version	Description
2.0.0	New properties: - $viewmode replaces $dayview. The calendar control now provides a number of new viewing modes. - $dayviewalldayheight specifies height of all-day panel in pixels. - $dayviewalldayheightmax specifies the maximum height for the all-day pane. - $dayviewheadingusedaycolors if true, the day header colours are matched with the day colours. - $templatetimelines for specifying one or more time lines. Other Improvements: - Visual indicators for events that are not in view. - The nowrap tag for display templates now causes ellipses to be displayed when the text does not fit horizontally.
1.7.0	New version numbers: OCal version 1.7.0 supercedes version 10.6.x. Please read the release notes for a detailed explanation.
10.6.0	New properties: $isoweektext - If specified, OCal will display the ISO 8601 week number in both the day and month views.
10.5.5	New properties: $repeatcontent - if enabled content is displayed repeatedly when the event wraps in day view or month views.
10.5.0	New properties: $holidaylist - list for specifying public holidays. $vertscroll - property for showing/hiding the vertical scroll bar (day view only). $vscroll- specifies the current vertical scroll position (day view only). evScroll - new event message. This event is send when the control is scrolled vertically. $dayviewtimecolumnwidth - width of the time column in pixels $makedatareference() - static library method for creating a special list reference for use with OCal. $disablescreenupdates() - static library method for disabling screen updates. $enablescreenupdates() - static library method for enabling screen updates. $dayviewnowrap - Property for turning off wrapping of text within an event box. $template... - set of new properties for controlling event content, time column and scales. Please refer to the new chapter "oCal Templates" for a detailed description of this new feature.

10.4.0	New properties: $ddallowotherfields - If true, calendar will allow drops from other fields of any type (default is false) $ddhiliteallday : If true, all-day cells are enabled and highlighted during drag & drop $ddhiliteworking : If true, working hours time slots are enabled and highlighted during drag & drop $ddhilitenonworking : If true, non-working hours time slots are enabled and highlighted during drag & drop $mousedate - returns the date and time under the mouse, adjusted according to $dayviewsnapminutes $mouseallday - returns true if the mouse is over the all-day pane Page 2 out of 3 $mouseevent - returns the line number of the event in the list currently under the mouse, zero if the mouse is not over an event
10.3.0	New properties: $columnallday : if set calendar allows all-day events $noeventborder : removes the extra border when showing event boxes $dayviewdividercolor: fill colour for all-day / day-view divider $dayviewtimecolumncolor : fill colour for time column in day view $dayviewtimecolumntextcolor :text colour for time column in day view $dayviewdeadcolumncolor : fill colour for dead area above scroll bar New event: evDeleteEvent : sent to field when user presses the delete/backspace key New event parameters: pAllDay : used with evCreateEvent, if true event was created in all-day pane
10.2.0	New properties $dayviewminutescale and $dayviewtimescale. New property $weekenddays. New property $dayviewvposminutes.

jsoCal: Changes History

Version	Description
1.1.0.0	Navigational arrows: We have introduced two new properties ($shownavbuttons and $navbuttonstext) and CSS for navigation buttons The property $eventslist has been made obsolete and you should now exclusively use $dataname. The jsoCal images have been moved into a folder named ctl_jsocal (previously located in the root of the Studio images folder). For other changes in this version please refer to the release notes.

Feature Overview

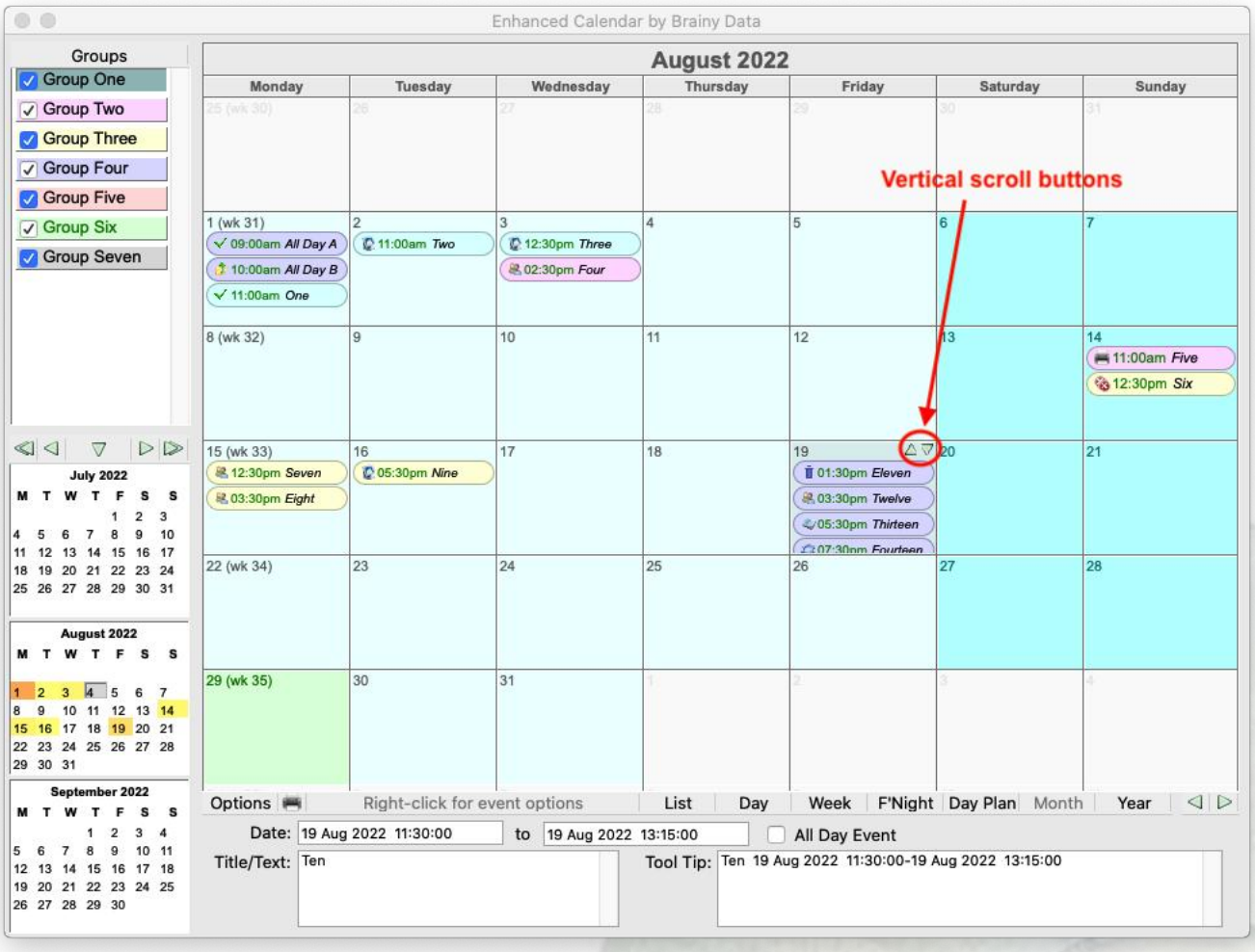
This section provides a quick overview of the main features of oCal and jsoCal. In many ways both oCal and jsoCal's main features are similar but also different due to the different nature of the two platforms. Therefore, the main features for both products will be listed separately.

oCal Desktop Main Features

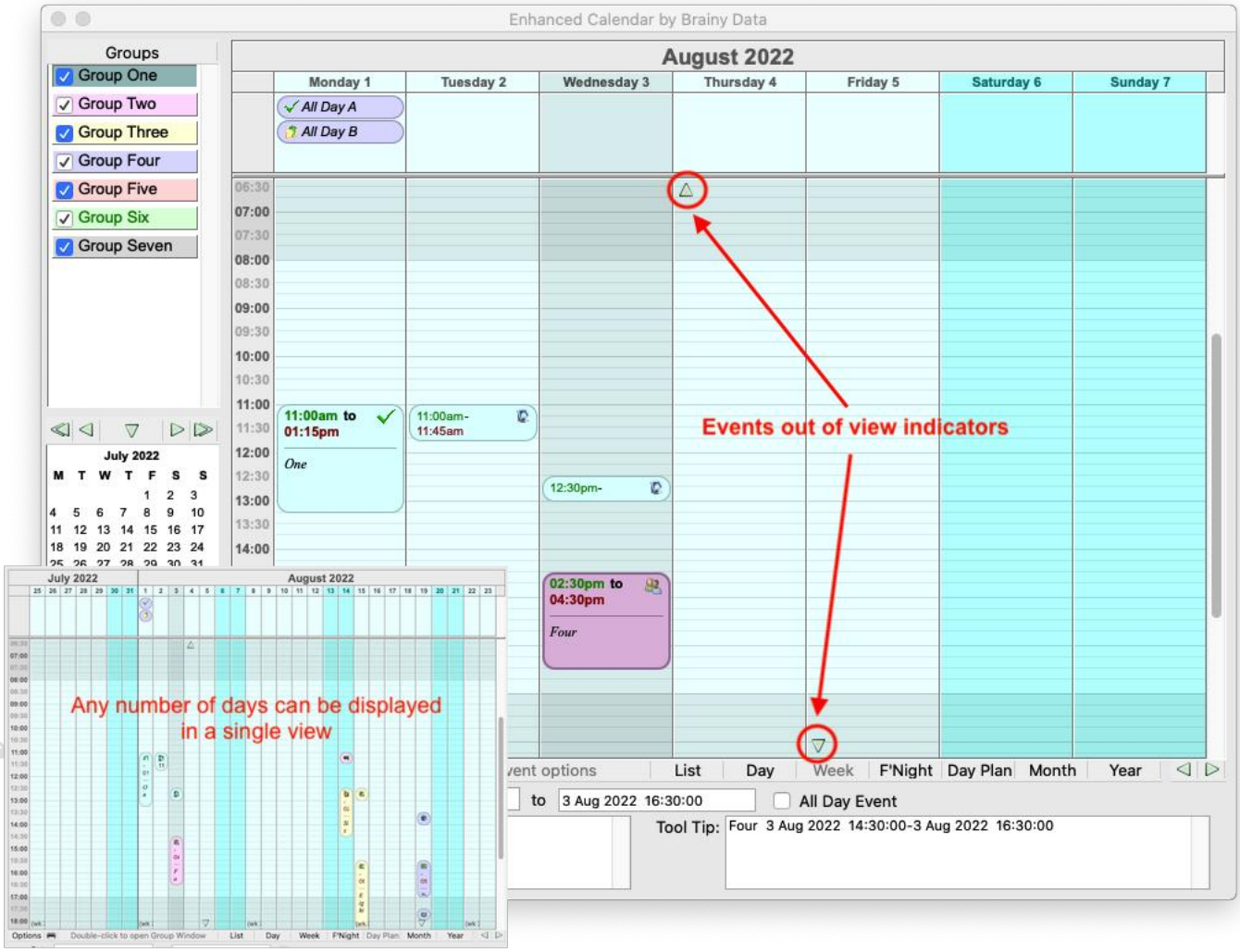
The enhanced calendar allows you to display and manipulate a list of events. The list can come from any source but must provide columns with start date/time and optionally an end date/time for each event. You can additionally provide columns for event colours, icons, text, layers, etc. The Chapter Designing oCal describes these features in more detail.

Views

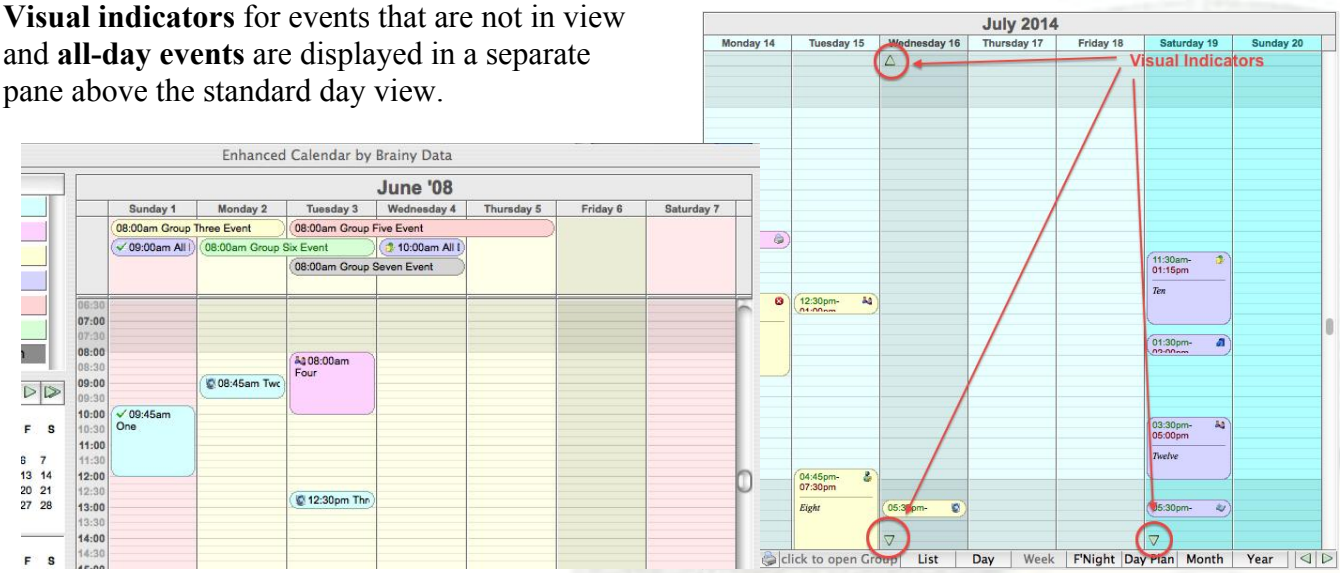
The oCal **month view** displaying events for an entire month.



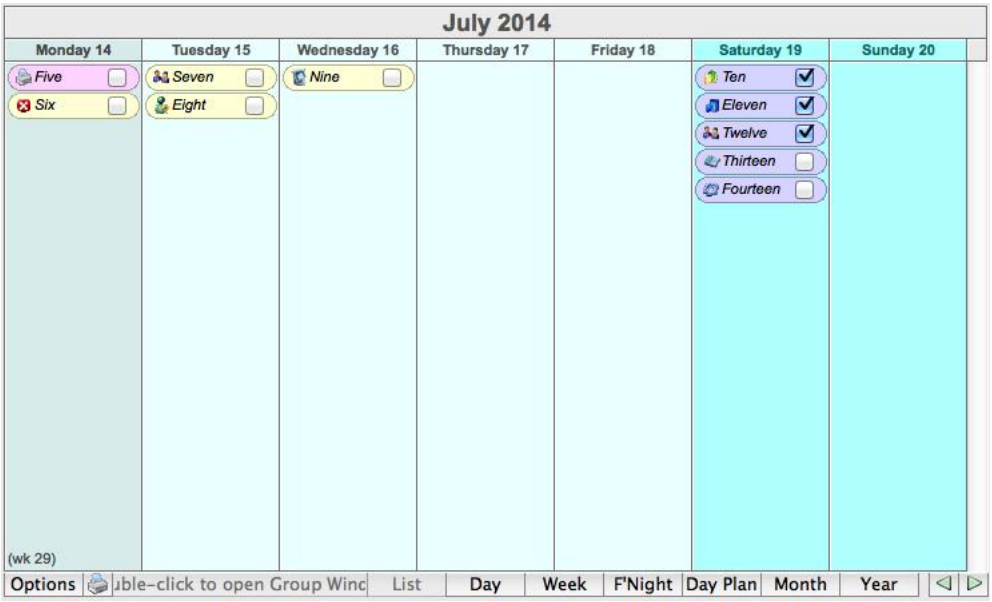
The oCal **day view** mode can display any number of days from 1 day to an entire year (if the screen size will allow it).



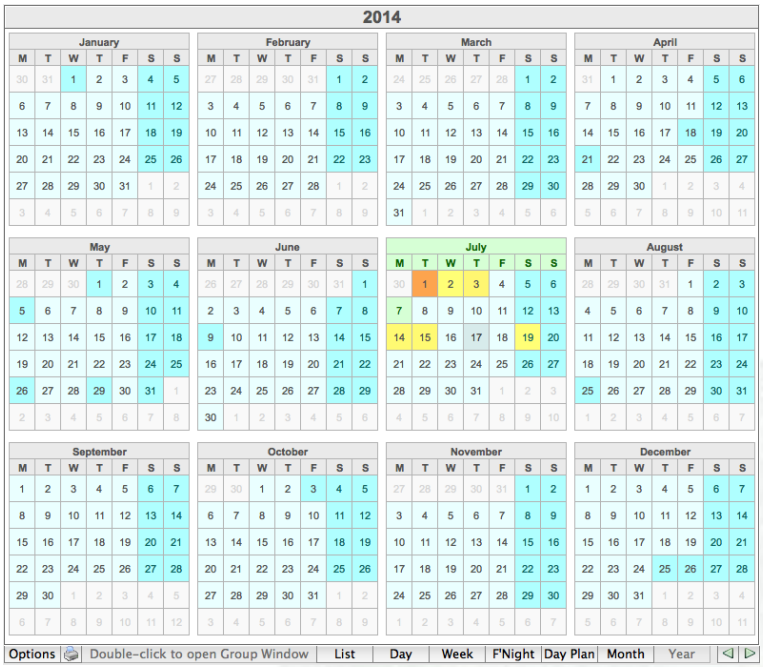
Visual indicators for events that are not in view and **all-day events** are displayed in a separate pane above the standard day view.



The **list view** displays all events in list form and can display check boxes for simulating to do lists.

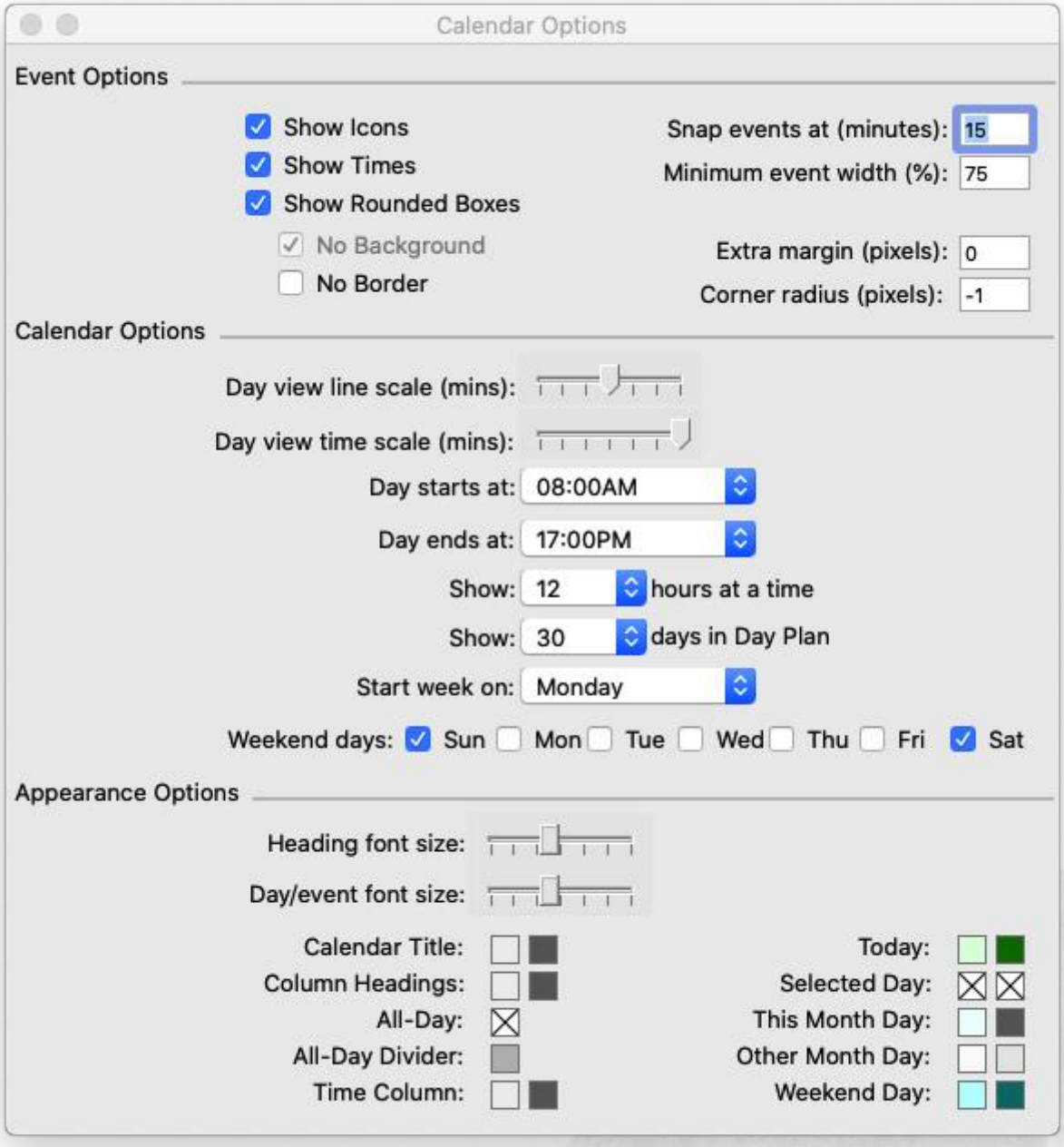


Month and Year **digest views** can outline busy days using definable colour ranges.

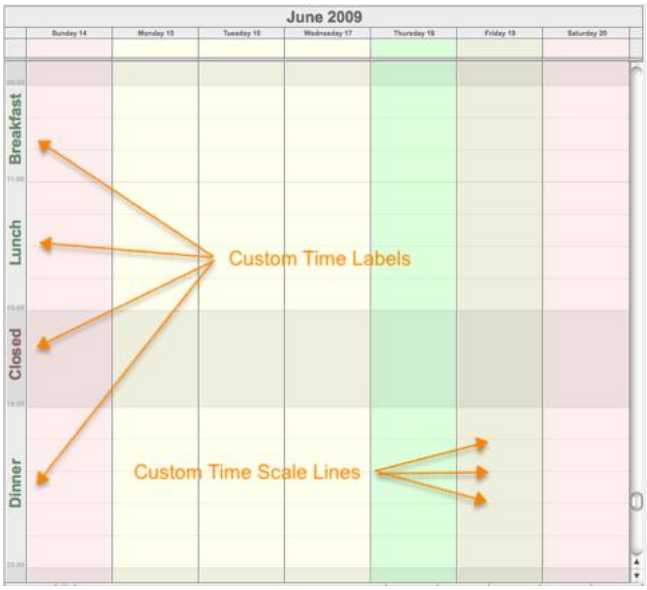


Appearance

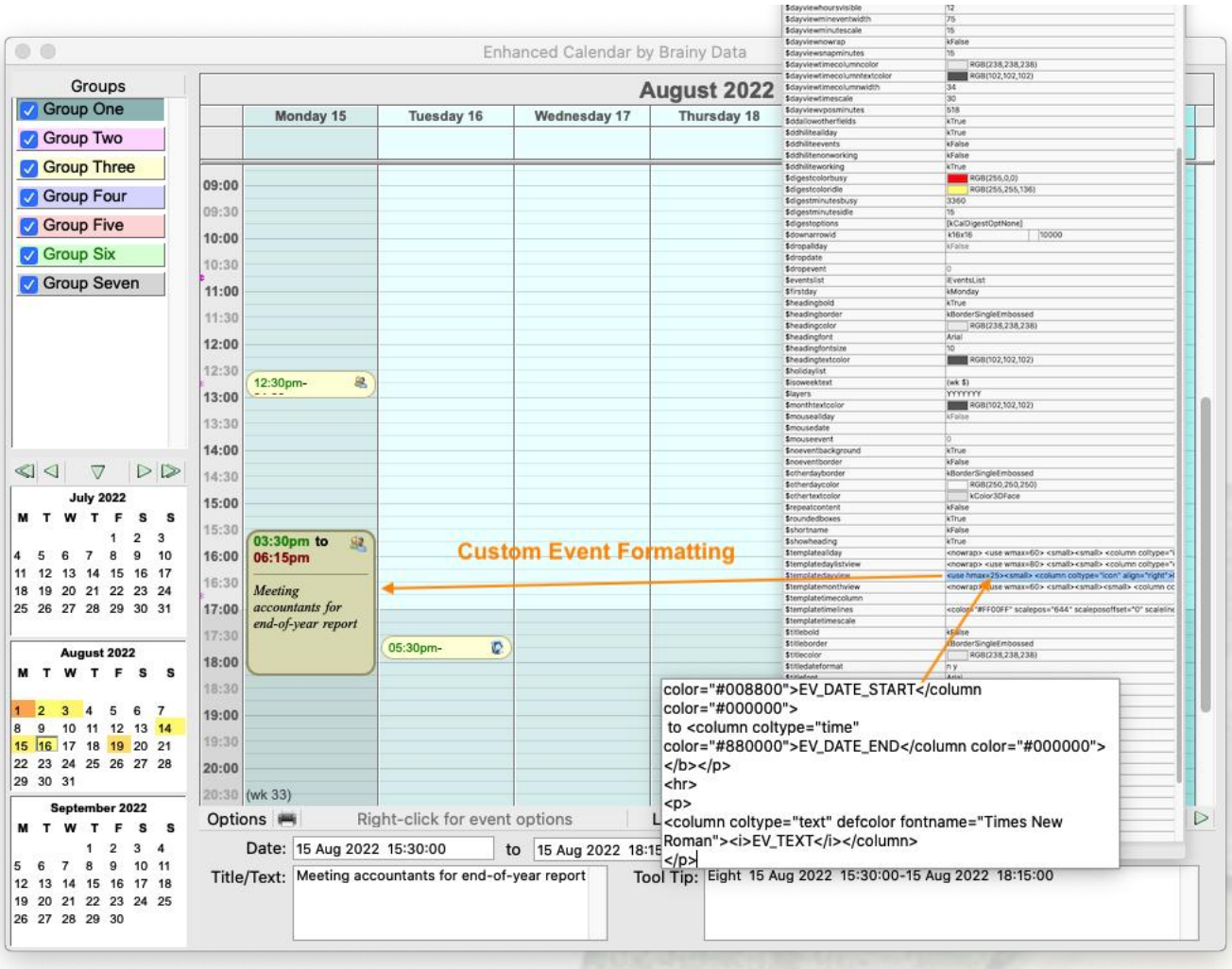
oCal Desktop implements numerous appearance and functional properties for providing a rich interface within Omnis that allows users to control every aspect of oCal.



oCal templates provide additional control over the appearance of every aspect of oCal, such as the scale of the time lines or the time line labels or the event boxes themselves. Different templates for the events can be provided for the different views and areas, i.e. the all-day pane, the main day view, the list view and the month view. An event template consists of HTML style tags to control the flow and appearance of the event box content. These HTML style tags, although similar to HTML tags, are a subset of HTML tags with some custom tags that have been specifically created for oCal to achieve certain features not available with standard HTML.



For full details please read the chapter "oCal Event Templates".

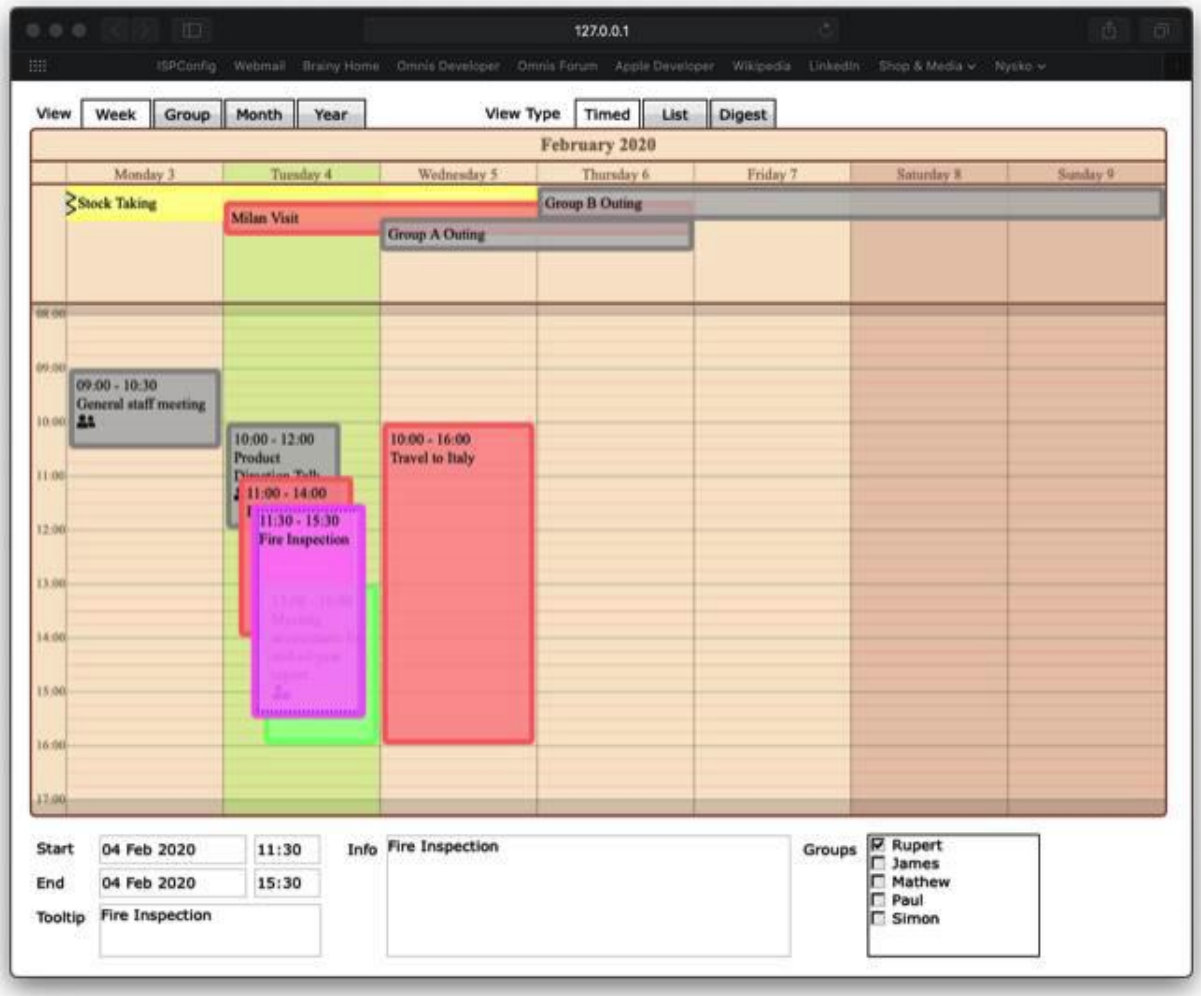


jsoCal Client Main Features

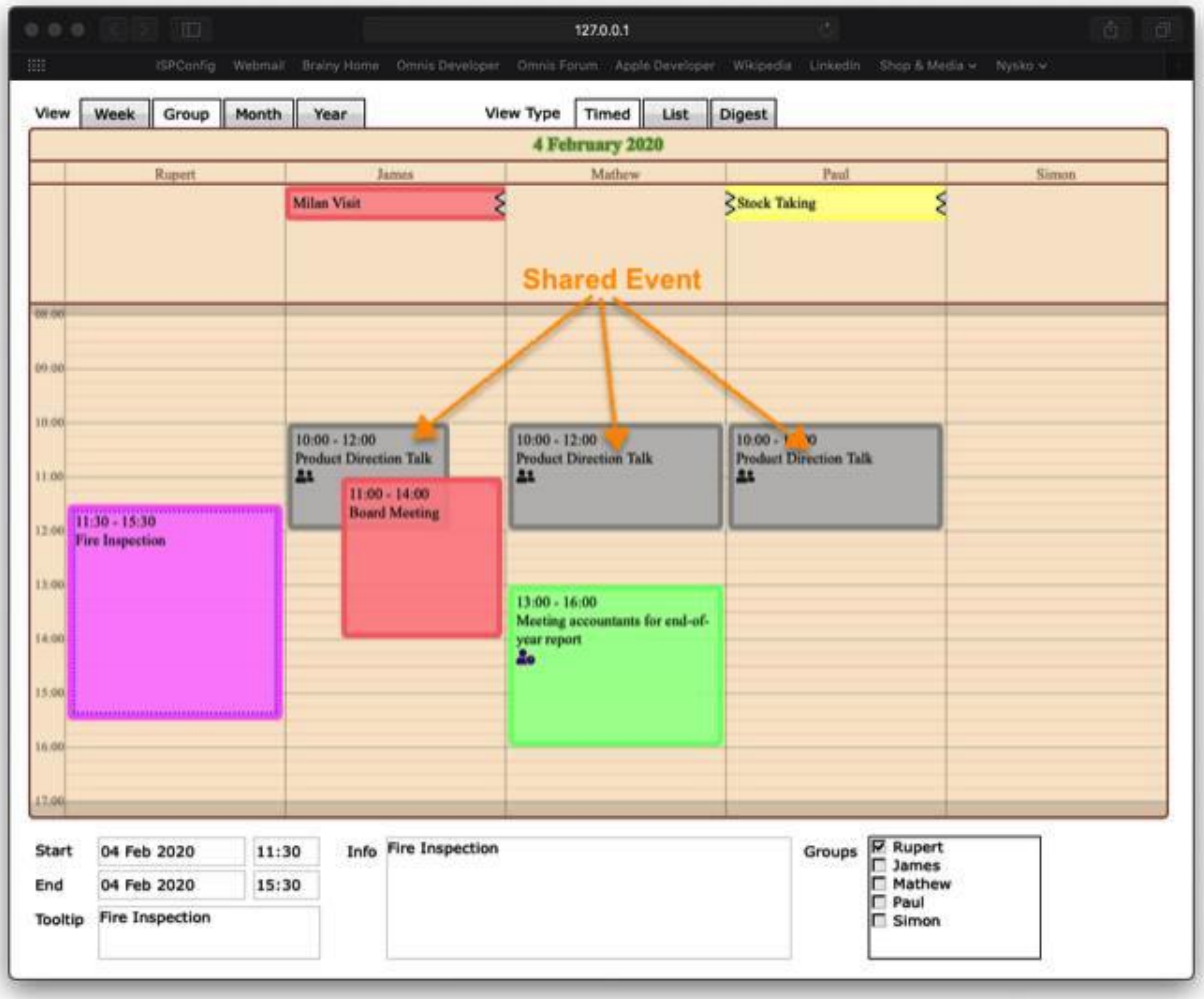
Just as its desktop counterpart, the JS-Client version of the calendar provides numerous views and extensive customization features. In the main, the views are similar to those provided by oCal with some minor differences and a proper group view that is merely simulated in the desktop version. However, the main difference is that much of the appearance of jsoCal is driven by CSS version 3 alongside the property driven event templates that utilize HTML5. Given the scope of CSS and HTML, the scope of customization in jsoCal is increased exponentially. The Chapter Designing jsoCal describes these features in more detail.

Views

The **week/day view** can display events in two different modes. The **timed mode** displays events across the configurable vertical time scale on each day. The **list mode** will list the events without overlapping them, displaying the earliest event of the day at the top of the column. Multi-day events are displayed in a separate **multi-day pane** above the standard day view columns. The week view can also be configured to show fewer or more days, i.e. just a single day, or 14 days, etc. See property $dayviewcolcount for valid assignments.

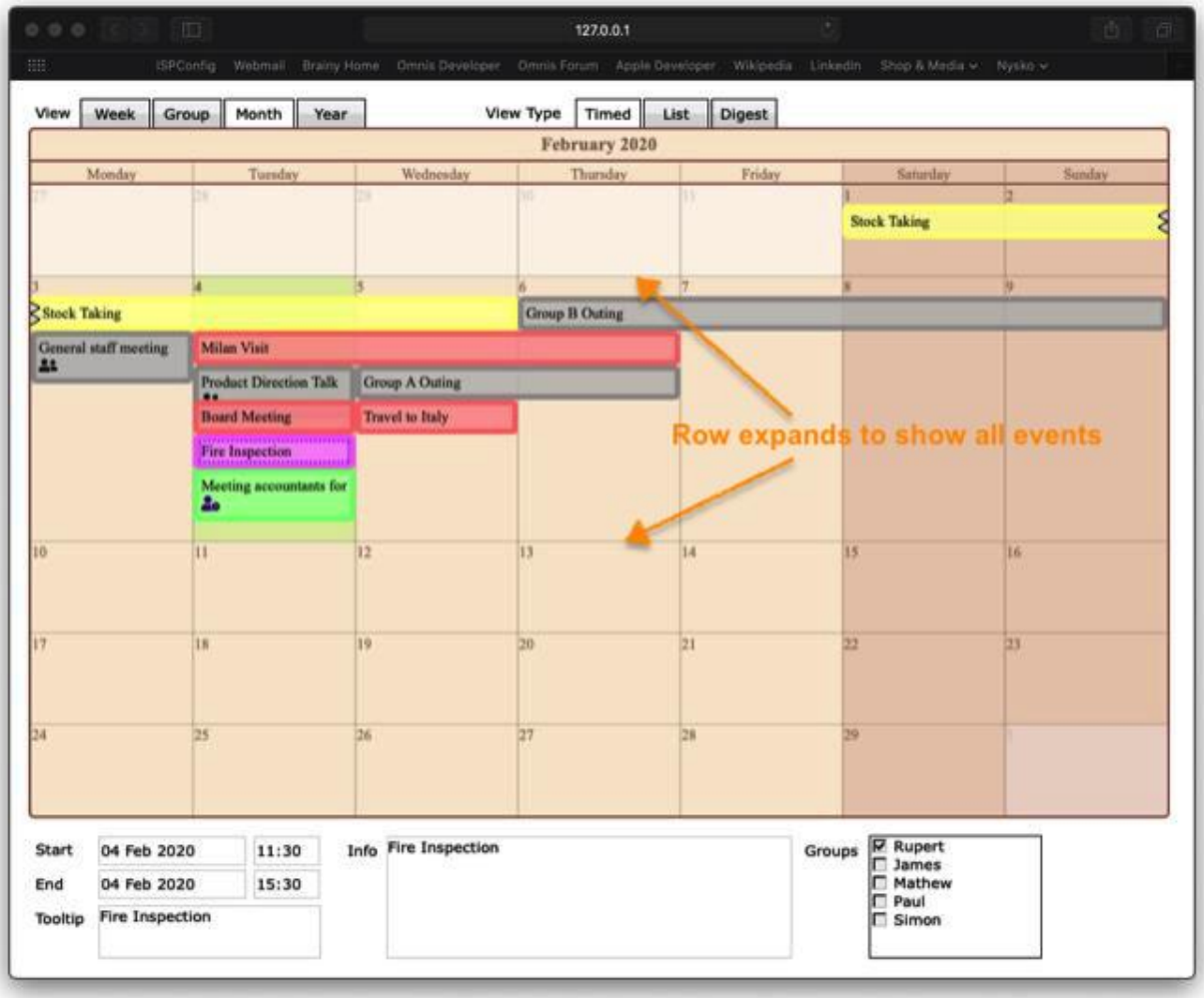


The **group view** allows the display of the events for a single day within configurable group columns. A scheduled event may belong to one or more groups. Theoretically, there is no limit to the number of layers, so you can display as many groups at the same time as the monitor will allow. However, jsoCal makes it easy to show or hide ranges of groups by assigning a sequence of YN states. As with the week view, events can be displayed in timed or list mode. Events that are shared amongst several groups will all select when one is clicked.

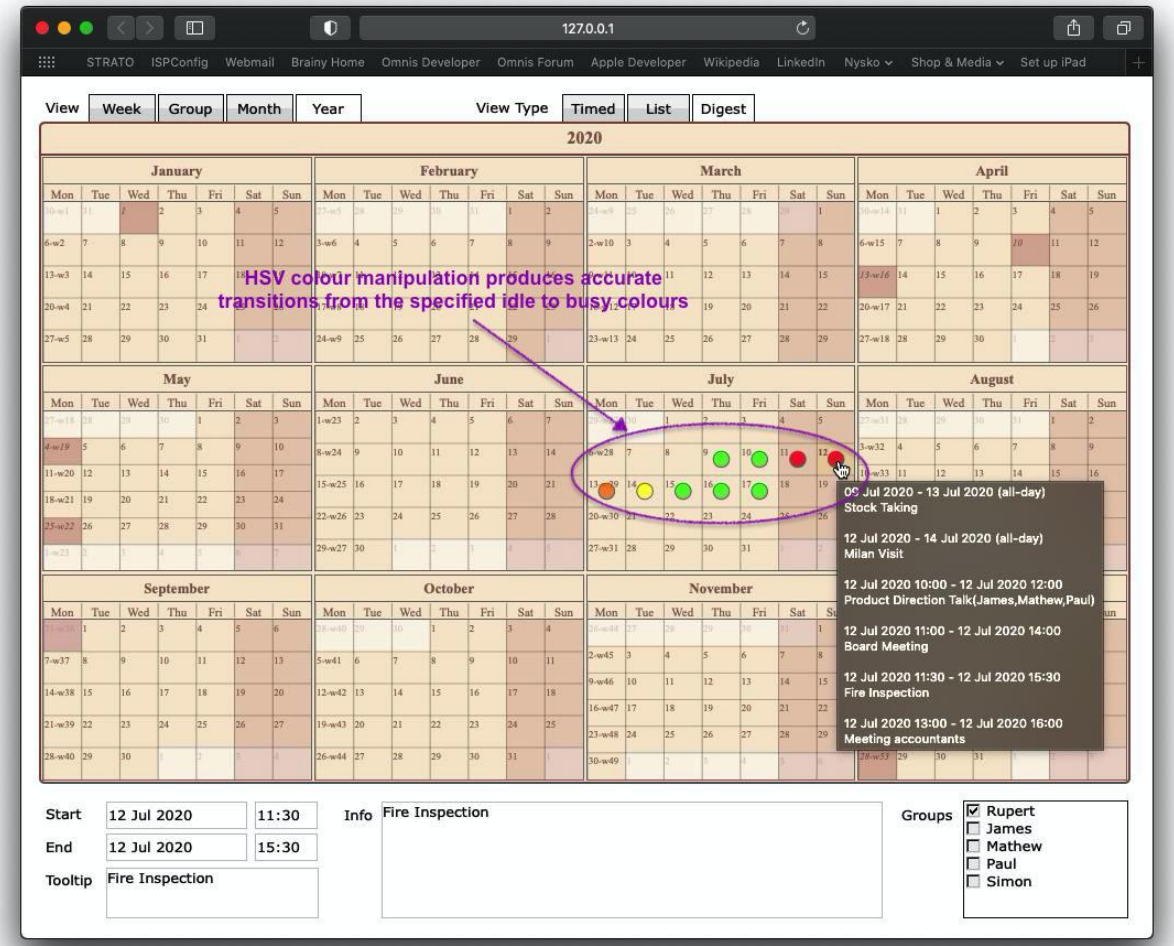


Both the **week/day view and group view** allow the dragging and resizing of events using the mouse. Of course, events can also be repositioned using date and time fields provided by your interface.

The **month view** displays the calendar using a traditional month view layout where the month is divided into rows of weeks. In this mode events are displayed in list mode only. A calendar row will expand to make room so all events in a day are visible. If need be, jsoCal will add a scroll bar should there not be enough room to display all rows of the month. The month view is also capable of displaying the ISO week numbers.



The **annual digest view** displays an entire year of events using colours to highlight busy or less busy days. The calculations and colours used are highly configurable. The jsoCal control uses HSV colour manipulation techniques to produce the smooth shades that morph the idle colour with the busy colour. Hovering over a digest bullet will pop-up a tool-tip displaying a configurable summary of all the events of that day.



Appearance

The calendar appearance, with the exception of the main colours and border style which are controlled by Omnis properties, is entirely controlled by configurable CSS. To manipulate individual elements within the control, the provided CSS file “ctl_ocal.css” can be edited. For example, to change the appearance for the time lines, simply edit the CSS for “.jsocal .horzDividerLineMajor” and “.jsocal .horzDividerLineMinor”

```
.jsocal .horzDividerLineMajor { /* seperates hours */
  stroke: black;
  stroke-width:1;
  stroke-linecap:butt;
  stroke-opacity: 0.2;
}
.jsocal .horzDividerLineMinor { /* seperates $dayviewtimescale minutes */
  stroke: red;
  stroke-width:1;
  stroke-linecap:butt;
  stroke-opacity: 0.1;
}
```

For a detailed description of CSS styles used by jsoCal, please read the chapter Designing jsoCal.

Individual events may also specify their own unique style settings via the event list as well as specifying custom templates for custom HTML and CSS layouts based on the current view rather than current event. Different templates for the events can be provided for the different views and view types, either individually to set the template for a specific viewtype (”Timed”, “List”, “Digest”) across all view modes, or a specific view mode (“DayView”, “GroupView”, “MonthView”, “YearView”) across all view types. Templates can also be specified for a specific view type within a specific view such as “DayView.AllDay”, “GroupView.Timed” or “MonthView.Digest”).

oCal…Designing OCal

Introduction

This chapter gives a brief description of the most important aspects of the enhanced calendar control. For a more detailed description of the external components please read the chapter oCal Reference.

Essentially, the enhanced calendar allows you to display and manipulate a list of events. The list can come from any source but must provide columns with start date/time and optionally an end date/time for each event. You can additionally provide columns for event colours, icons, text, layers, etc.

Important: Your event list must be sorted at all times in ascending order based on the start date/time column of your list.

Basic OCal interface

The ocal interface allows the display of events in the traditional month and day views as well as list and digest modes. The day view is capable of displaying any number of days from just 1 day to 7 days for a week, 31 days for a month, 90 days for a quarter of a year or as many days as is physically possible to fit on the screen.

The Event List

OCal directly interacts with the list of events that you assign to it via the $eventslist property. When a user clicks an event box, OCal automatically updates the selection states and current line of your list. However, it is up to you to implement code to manipulate your event list when you receive the appropriate events, i.e. when the user moved or resized an event, pressed the delete key, or dropped objects or data from outside the calendar control. Nothing will happen unless you say so.

Important: When assigning a list to $eventslist, the list must be an instance, task or class variable of type list and belong to the window/sub-window that owns the OCal control. You cannot specify an item reference variable because of a data handling limitation in the external interface. However, many people have expressed a need to pass references to list into a window for editing so we have implemented a solution in version 10.5.0. We have provided a new function called $makedatareference() available under the 'Functions' tab in the Catalog. This function can be used in the class instance that owns the master list to produce a text based reference to the list that can be used with $eventslist.

Event List Columns

As well as specifying the list of events, OCal also needs to know the names of the columns that contain the information that OCal requires to display events. You can tell OCal about your list columns via the set of $column... properties. At a minimum, you must provide a column for $columndate and $columntime. If your list contains just one column with combined date and time, you may assign the same column name to both properties. If your events have a duration you must also provide a name for $columnenddate and $columnendtime.

In addition you can specify columns for the event fill color, text color, icon, etc. OCal will use this to format and display information in the event box. Version 10.5.0 introduces a new way to specify content in events using templates. Please refer to the chapter oCal Event Templates.

Important: At all times you must ensure that your list is sorted in ascending order by the starting date and time. If the list is not sorted correctly when you issue a redraw, events may not be displayed correctly.

Drag and Drop

Users can interact with events by dragging them to move or resize them. When this happens you will receive appropriate events in the $event method. OCal provides a set of properties who's names begin with $dd..., that give you control over what users can and cannot do. They are mostly concerned with what can be dropped where.

Layers

OCal provides the ability to group your events in up to 255 layers. Each event in your list can belong to one or more of these layers. You will need to provide a column in your list that specifies the layers string for each event. The layers basically consists of a series of 'Y' and 'N' characters. For example, the string "YYYNNY" tells OCal that this event belongs to layer 1,2,3 and 6. You can tell OCal to show just one, several or all layers at the same time by calculating the property $layers with a similar string. This makes it very easy to show and hide individual groups of events.

Translation

In the examples folder is a string table file called jsocal.tsv. You can edit this file with the Omnis string table editor for the purpose of translating the strings used by jsoCal. Once they are translated, you must export the string table to javascript and place it in the folder "strings" inside the Omnis "html" support folder.

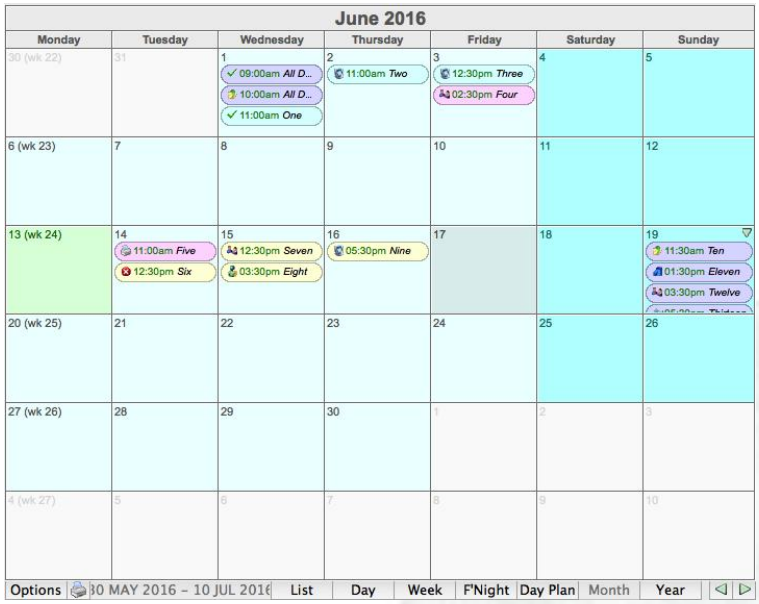
View Modes

From version 2 onwards, besides the traditional Month and Day viewing modes, the calendar can display events in a list view and show monthly or yearly digest information. The desired viewing mode is selected by assigning one of the kCalViewMode... constants to the property $viewmode.

kCalViewModeMonth	Traditional month view (default)
kCalViewModeDayNormal	Traditional day view with vertical time scales
kCalViewModeDayList	List based day view without vertical time scales New
kCalViewModeMonthDigest	Shows events in month view as shades of color ranging from idle to busy (see $digest... properties) New
kCalViewModeYearDigest	Shows events in year view as shades of color ranging from idle to busy (see $digest... properties) New

kCalViewModeMonth

The traditional month view mode displays events one month at a time showing seven days along the horizontal axes and the weeks of the months along the vertical axes. Within this mode, the event display can be customised using the $templatemonthview property. There are numerous other appearance properties that can alter the appearance of the month view all of which are documented in the chapter “External Component Reference”.



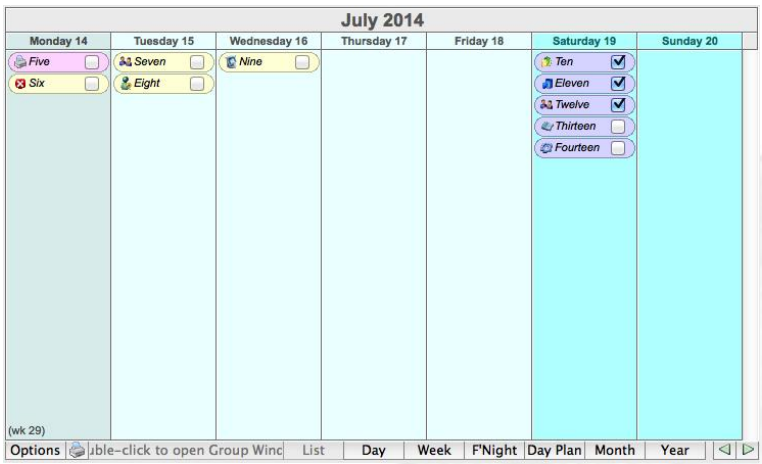
kCalViewModeDayNormal

The traditional day view mode can displays events along a vertical time scale with a varied number days along the horizontal axes controlled by $dayviewdaycount. Within this mode, the

event display can be customised using the $templatedayview property and the vertical time scale appearance can be controlled by the $templatetimecolumn and $templatetimescale properties. There are numerous other appearance properties that can alter the appearance of the day view all of which are documented in the chapter “External Component Reference”.

kCalViewModeDayList

The new list view mode is similar to the day view display mode but it does not display a vertical time scale. This view simply lists the events vertically only showing a single line of content for each event. The content can be controlled by specifying a template for the $templatedaylistview property. A new template tag has been added so an active check-box can be displayed within the event allowing users to tick events/to-do items as they are completed.

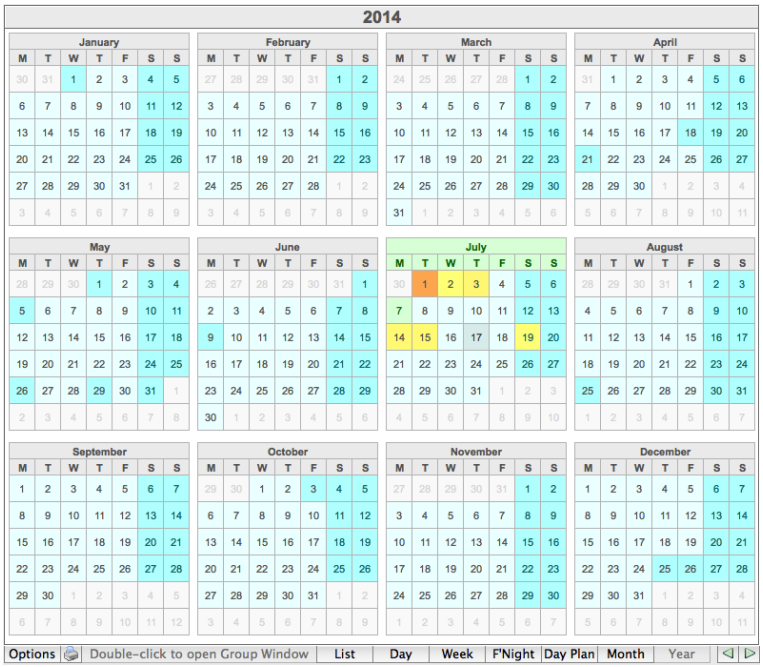


When a user clicks a check-box, the event evCheckboxClick is sent to the calendars $event method. The event parameter pColumnNumber specifies the list column that the check-box is associated with, if a column was specified in the display template.

Note: Only one check-box is supported.

kCalViewModeMonthDigest & kCalViewModeYearDigest

The new month digest view is like the standard month view except events are displayed in digest mode (the background colour of each day changes according to how busy the day is). The new year view displays an entire year by displaying 12 small month views in digest mode.



The digest colours and how they are calculated is controlled by a range of digest properties, $digestcoloridle, $digestcolorbusy, $digestminutesidle, $digestminutesbusy, $digestoptions.

Further Reading

The Brainy Data support website lists a number of technical notes relating to OCal and other software. You should always read these notes before you begin developing our software.

The chapter Event Templates explains in detail how to compose your own templates for the visual interface of OCal.

oCal…Event Templates

oCal Version 1.5 introduced templates for greater control over the display of content in an event box, and customize the day view time column and scales. The template properties $templateallday, $templatedayview, $templatemonthview and $templatedaylistview control the appearance of event content, making it possible to display multiple icons in any order, text using multiple fonts and styles, and to have different content depending on the size of the event box. The template properties $templatetimecolumn and $templatetimescale control the day view time column and scales allowing one to place scale lines at any desired position and paint any text in the time column using horizontal or vertical text directions.

Template Syntax

The syntax of a template is in many ways very similar to simple HTML or XML in as far as that you specify tags within the '<' and '>' characters and actual content anywhere else. Some tags may have a start and end tag with content in between, i.e. <column> and </column>.

You may specify multiple tags within the same set of '<' and '>' characters.

Some tags may have parameters. Parameters are specified by the parameter name, the assignment character followed by the value in double quotes, i.e. coltype="text".

The syntax of a template is very strict and errors in the syntax will result in the template not to paint correctly if at all.

Not all tags are supported in every template, although including them does not cause an error. For example, when specifying a template for custom day view scales, the only valid tags are *scaledark*, *scalelight*, *scalepos*, *scaleline* and *scalefilldark*. Each tag description will specify in which templates the tag is supported.

Decision and Data Tags

This section lists all tags related to data handling and decision making during paint.

<use>

Syntax: **<use>***event_content***</use>**

Example: <use wmax=20>...</use>
<use hmax=15>...</use>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The use tag is used to decide if the content within the start and end use tags is to be displayed based on the current width or height of the event box.

You may not specify both wmax and hmax for the same use clause. A template may contain multiple use clauses followed by default content without a use clause. Only the first use clause that fits the criteria will be used and no further clauses are processed once one is found.

Parameters	Description
wmax	content is displayed if the event box is n pixels or less in width
hmax	content is displayed if the event box is n pixels or less in height

<column>

Syntax: **<column>***column_name***</column>**

Example: <column coltype="text" nowrap>EV_TEXT</nowrap /column>
<column coltype="icon48" align="right">EV_ICON_ID</column>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The column tag allows one to specify a column from the event list for displaying content. The coltype parameter tells OCal what type of column this is, text, time or icon. For icons one can also specify icon16, icon32 or icon48, to specify the icon size to be used. In addition Icons can also be right aligned.

Parameters	Description
coltype	Specifies the column type, either "text", "time", "checkbox", "icon", "icon16", "icon32" or "icon48".
align	Specifies the horizontal alignment "left", "center" or "right".

<calculation>

Syntax: **<calculation>***omnis_calculation***</calculation>**

Example: <calculation calctype="text" nowrap>cap(iEventsList.EV_TEXT)</nowrap /calculation>
<calculation calctype="text">jst("T: H:M", iEventsList.EV_DATE_START)</calculation>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The calculation tag allows you to specify any valid Omnis calculation that is within the context of the OCal window. The calculation may refer to a column in the event list by specifying the instance list name and column name, omitting the line number. OCal will automatically set the correct line number prior to evaluating the calculation.

Just as with the column tag above, the calculation tag has a parameter to specify the result type of the calculation, so OCal knows how to paint the result.

Important Note: Calculations are slower than the more direct column tags and overuse of calculations may effect performance.

Parameters	Description
calctype	Specifies the calculation result type, either "text", "time", "checkbox", "icon", "icon16", "icon32" or "icon48".

Paragraph and Style Tags and Parameters

This section lists all the tags for handling paragraphs and text styles.

* *

Syntax: ** ***content*** **

Example: ** **<column>CUST_NAME</column>** **

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The paragraph tag is optional. The only time you need to use it is if you want some extra spacing between two sets of content. OCal will automatically add a third of the current font size as a gap between two paragraphs.

Parameters	Description
none	

*
*

Syntax: **
**

Example: <column>CUST_NAME</column>
<column>CUST_ADDRESS</column>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

Insert a new line.

Parameters	Description
none	

,,<u>

Syntax: *****content*****, *****content*****, **<u>***content***</u>**

Example: <column>CUST_NAME</column>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

Paints text in between tags in bold, italic or underline.

Parameters	Description
none	

<small>,<big>

Syntax: **<small>***content***</small>**, **<big>***content***</big>**

Example: <small><small><column>CUST_NAME</column></small></small>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

Paints text in between tags 1 point smaller or bigger. You may use multiple small or big tags to reduce or increase the font size by several points.

Parameters	Description
none	

<nowrap>

Syntax: **<nowrap>***content***</nowrap>**

Example: <calculation calctype="text" nowrap>cap(iEventsList.EV_TEXT)</nowrap /calculation>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

All content in between the nowrap tag will prevent wrapping and only specific break or paragraph tags will insert line feeds. If text does not fit on the nowrap line it is displayed with ellipses.

Parameters	Description
none	

<color>

Syntax: **<color="#RRGGBB">***content***</tag>**

Example: <color="#0000FF"><column>CUST_NAME</column><color="#000000">

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The color tag changes the current text color. The color change applies to the remainder of the content. The color value can be specified as a 6 digit hexadecimal value in the format RRGGBB.

Parameters	Description
self	the color value in hexadecimal

<fontname>, <fontsize>

Syntax: **<fontname="*font_name*" fontsize="*font_size*">**

Example: <fontname="Arial" fontsize="10"><column>CUST_NAME</column><fontname="Times New Roman" fontsize="12">

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The font name tag changes the current font face. The font change applies to the remainder of the content. The font name must be one of the font names that Omnis supports. For a complete list of font names use the Omnis font function FontOps.$winlistfonts()

Parameters	Description
self	the font name

<align>

Syntax: **<align="*alignment*">**

Example: <column align="left">CUST__NAME</column>

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview, $templatetimecolumn

The align tag changes the current text or icon alignment. When used with $templateallday, $templatedayview, $templatemonthview, the only supported option is “right” when used with icons. Changing the text alignment in event boxes is not supported.

When used with $templatetimecolumn, all options are supported. The up and down alignment will draw the text vertically, centering the text at the current scale position.

Parameters	Description
self	the alignment, one of "left", "center", "right", "up" or "down".

<hr>

Syntax: **<hr color="#RRGGBB">**

Example: Text Above<hr color="#FF0000">Text Below

Templates: $templateallday, $templatedayview, $templatemonthview, $templatedaylistview

The horizontal ruler tag will draw a horizontal single pixel line, allowing the clear separation of content. By default the line will take on the color of the event box border. This color can be overridden by specifying a color within the ruler tag

Parameters	Description
color	the line color

Scale Tags

The following are specific tags supported when painting custom day view time column and scales.

<scalelight><scaledark>

Syntax: **<scaledark>***dark_scale_lines***<scalelight>***light_scale_lines*

Example: <scaledark><scalepos="480" scaleline><scalelight><scalepos="540" scaleline>

Templates: $templatetimecolumn, $templatetimescale

Selects the light or dark font or pen for painting the lighter or darker scale text or lines.

Parameters	Description
none	

<scalepos>

Syntax: **<scalepos="minutes">**

Example: <scaledark><scalepos="480" scaleline><scalelight><scalepos="540" scaleline>

Templates: $templatetimecolumn, $templatetimescale

Sets the current scale position by specifying the time of day in minutes. For example 8am would be specified as 480 (8 x 60), and 5pm would be 1020 (17 x 60)

Parameters	Description
self	the vertical position in minutes

<scaleline>

Syntax: **<scaleline>**

Example: <scaledark><scalepos="480" scaleline><scalelight><scalepos="540" scaleline>

Templates: $templatetimecolumn, $templatetimescale

Instructs OCal to paint a scale line at the current position using the selected pen.

Parameters	Description
none	

<scalefilldark>

Syntax: **<scalefilldark>**

Example: <scalepos=”720” scalepos=”900” scalefilldark> // fill the area between 12pm and 3pm

Templates: $templatetimecolumn, $templatetimescale

Instructs OCal to fill the background of the time column or day view area with the non-working hour color. The fill is positioned between the last two scalepos coordinates.

Parameters	Description
none	

Sample Event Templates

The following are the templates used in the OCal example library. Explanations of each line is given in read between the lines of the template text but do not form a valid part of the template text it self.

Month View Template ($templatemonthview)

The month view template specifies four “use” clauses for when the width of the event is less or equal to 60, 80, 100, or 130 pixels respectively. For the smaller use clauses less data is displayed using a smaller font size with the use of the <small> tag.

```
;; for month views, event content is not wrapped so we use the <nowrap> tag for all content
<nowrap>

;; when the event width is 60 pixels or less, we display content three point sizes smaller, and we display an icon,
;; start time in green and the event text in italic.
<use wmax=60>
<small><small><small>
<column coltype="icon">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column>
<column coltype="text" color="#000000"><i>EV_TEXT</i></column>
</use>

;; when the event width is 80 pixels or less, we display content two point sizes smaller, and we display an icon,
;; start time in green and the event text in italic.
<use wmax=80>
<small><small>
<column coltype="icon">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column>
<column coltype="text" color="#000000"><i>EV_TEXT</i></column>
</use>

;; when the event width is 100 pixels or less, we display content one point size smaller, and we display an icon,
;; start time in green and the event text in italic.
<use wmax=100>
<small>
<column coltype="icon">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column>
<column coltype="text" color="#000000"><i>EV_TEXT</i></column>
</use>

;; when the event width is 130 pixels or less, we display content at the default size, and we display an icon,
;; start time in green and the event text in italic.
<use wmax=130>
<column coltype="icon">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column>
<column coltype="text" color="#000000"><i>EV_TEXT</i></column>
</use>

;; when the event width is wider than 130 pixels, we display content at the default size, and we display an icon,
;; start time in green, end time in red and the event text in italic.
<column coltype="icon">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column><color="#000000">-
<column coltype="time" color="#880000">EV_DATE_END</column>:
<column coltype="text" color="#000000"><small><i>EV_TEXT</i></small></column>
```

All-Day View Template ($templateallday)

The all-day view template specifies just one "use" clause for when the width of the event is less or equal to 60 pixels.

```
;; for all-day views, event content is not wrapped so we use the <nowrap> tag for all content
<nowrap>

;; when the event width is 60 pixels or less, we display content two point sizes smaller, and we display an icon,
;; and the event text in italic.
<use wmax=60>
<small><small>
<column coltype="icon">EV_ICON_ID</column>
<column coltype="text"><i>EV_TEXT</i></column>
</use>

;; when the event width is wider than 60 pixels, we display content at the default size, and we display an icon
;; and the event text in italic.
<column coltype="icon">EV_ICON_ID</column>
<column coltype="text"><i>EV_TEXT</i></column>
```

Day View Template ($templatedayview)

The day view template specifies the "use" clauses for when the hight is 25 pixels or less and the width is 90 pixels or less, respectively.

```
;; when the event height is 25 pixels or less, we display content one point size smaller, and we display a right-
aligned
;; icon, the start time in green, the end time in red and the event text in Italic and Times New Roman.
<use hmax=25><small>
<column coltype="icon" align="right">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column color="#000000">-
<column coltype="time" color="#880000">EV_DATE_END</column color="#000000"><br>
<column coltype="text" color="#000000" fontname="Times New Roman"><i>EV_TEXT</i></column>
</small></use>

;; when the event width is 90 pixels or less, we display content one point size smaller, and we display a right-aligned
;; icon, the start time in green, the end time in red and the event text in Italic and Times New Roman.
<use wmax=90><small>
<column coltype="icon" align="right">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column color="#000000">-
<column coltype="time" color="#880000">EV_DATE_END</column color="#000000"><br>
<column coltype="text" color="#000000" fontname="Times New Roman"><i>EV_TEXT</i></column>
</small></use>

;; for all other event sizes we use the default font size and we display a 16 pixel right-aligned icon, start time in
green,
;; end time in red, followed by a new paragraph with the event text in Italic and Times New Roman
<p><b>
<column coltype="icon16" align="right">EV_ICON_ID</column>
<column coltype="time" color="#008800">EV_DATE_START</column color="#000000">
 to <column coltype="time" color="#880000">EV_DATE_END</column color="#000000">
</b></p>
<p>
<column coltype="text" color="#000000" fontname="Times New Roman"><i>EV_TEXT</i></column>
</p>
```

List Day View Template ($templatedaylistview)

Our example list template displays a non-wrapping line containing a checkbox, icon and text.

```
;; specify no wrapping, causes text to display ellipses if it does not fit
<nowrap>
;; use smaller text if we have 80 pixels or less in width
<use wmax=80>
<small><small>
<column coltype="checkbox" align="right">EV_DONE</column>
<column coltype="icon" align="left">EV_ICON_ID</column>
<column coltype="text"><i>EV_TEXT</i></column>
</use>
;; standard view when we have more than 80 pixels
<column coltype="checkbox" align="right">EV_DONE</column>
<column coltype="icon" align="left">EV_ICON_ID</column>
<column coltype="text"><i>EV_TEXT</i></column>
```

Sample Time Column and Scale Templates

The following sample templates demonstrate what can be achieved when applying templates to both the time column and day view scale. The Image at the end shows the visual result of the two templates.

Time Column Template ($templatetimecolumn)

This sample displays e mixture of times painted horizontally and text denoting different parts of the day, painted vertically.

```
;; we pretend to be a restaurant and break up the day into portions relevant to the establishment.
;;
;; first we fill an area between 3pm and 6pm with the non-working hour fill, as this is the time when we are closed
<scalepos="900" scalepos="1080" scalefilldark>

;; next we specify the times that separate the parts of the day. We right justify the times within the time column and
;; paint them using the default light font
<align="right" scalelight>
<scalepos="480">08:00
<scalepos="660">11:00
<scalepos="900">15:00
<scalepos="1080">18:00
<scalepos="1380">23:00

;; next we paint portions of the day when the restaurant is closed in a subtle red color
;; for this we now use a larger Arial font and we change the painting direction to up
<fontname="Arial" fontsize="18" align="up">
<color="#886666">
<scalepos="990">Closed

;; the portions of the day when food is served are painted in a subtle green
<color="#668866">
<scalepos="570">Breakfast
<scalepos="780">Lunch
<scalepos="1230">Dinner
```

Scale Template ($templatetimescale)

This template is used to specify the position of the horizontal scale lines in the day view. Typically it is synchronized in some form with the content of the time column.

```
;; first we fill an area between 3pm and 6pm with the non-working hour fill, as this is the time when we are closed
<scalepos="900" scalepos="1080" scalefilldark>

;; now we paint the darker scale lines at the time positions that separate the different parts of the day
;; (Breakfast, Lunch, etc)
<scaledark>
<scalepos="480" scaleline>
<scalepos="660" scaleline>
<scalepos="900" scaleline>
<scalepos="1080" scaleline>
<scalepos="1380" scaleline>

;; in between the darker scales we paint lighter scales on every hour
<scalelight>
<scalepos="540" scaleline>
<scalepos="600" scaleline>
```

```
<scalepos="720" scaleline>
<scalepos="780" scaleline>
<scalepos="840" scaleline>
<scalepos="960" scaleline>
<scalepos="1020" scaleline>
<scalepos="1140" scaleline>
<scalepos="1200" scaleline>
<scalepos="1260" scaleline>
<scalepos="1320" scaleline>
```

The final result is shown in the image below.

Timelines Template ($templatetimelines)

The $templatetimelines property specifies the template for time lines that are drawn across all days or specified days within the day view display mode. The following template tags are supported.

scalepos: specifies the position in minutes within the day view

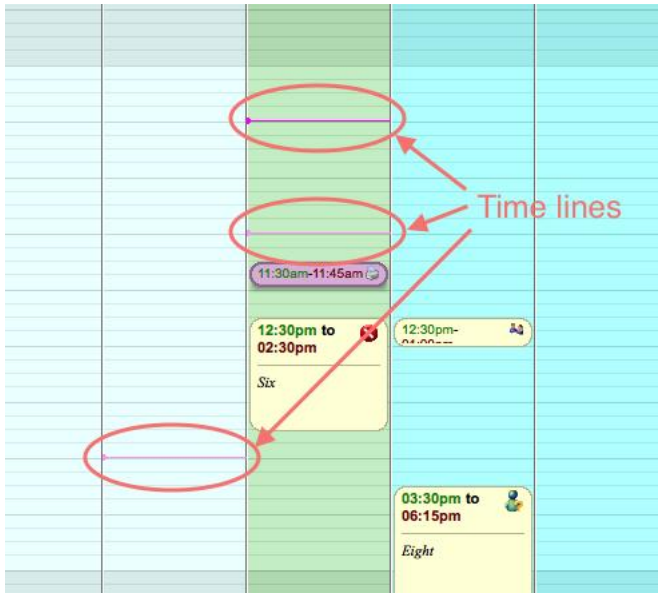
scaleposoffset: specifies the offset in minutes from scalepos. If this tag is not specified, scaleline instruction will paint a line across all displayed days. If scaleposoffset specifies zero, a scaleline instruction will draw a line within todays box only. If scaleposoffset specifies a negative or positive offset, the next scaleline instruction draws a line offset by the specified minutes within the box of the appropriate day.

scaleline: draws a time line at positions calculated from scalepos and scaleposoffset.

color: changes the color for the next scaleline instruction

Example: The following template would produce the time lines displayed in the image.

```
<color="#FF00FF" scalepos="540" scaleposoffset="0" scaleline>
<color="#FFAAFF" scaleposoffset="-1080" scaleline>
<scaleposoffset="+120" scaleline>
```



The bolder time line is displayed at 9am in the todays box. The second time line is displayed 18 hours earlier and the third time line is displayed 2 hours later.

Implementation notes: In the examples we use a timer object (oCalCurrentTime) that runs once every minute which updates $templatetimelines with the current time.

oCal…Reference

Introduction

This chapter lists all the enhanced calendar properties, methods and events as provided by the external component. The OCal component library supplies a window control for use on your windows, a remote form control for use in remote forms for the web-client, and a report object for printing calendars.

Contents

Properties	
Property	**Description**
$allowchange	If true, the user can change the current date
$ampmstring	String that specifies the am/pm characters separated by a '/'.
$column...	$columndate, $columnenddate, $columntime, $columnendtime, $columnallday, $columnicon, $columntext, $columntextcolor, $columnbackcolor, $columntooltip, $columnlayers - all these properties specify the names of the columns in your list that provide the information. This gives you great flexibility in how you organize your list/smart list.
$currday	The calendar's current date
$currdaycolor	The fill color of the current day
$currdayicon	The icon used to represent the current day (month view only)
$currdaymode	The border style for the current date
$currdaytextcolor	The color of the current date
$daycolor	The color of other then the current, weekend, and other months days
$dayfont	The font for the days in this month
$dayfontsize	The font size for the days in this month
$daymode	The border style for the days in this month
$daytooltips	If true,a separate tool-tip is required for each day. See also $tooltipday.
$dayview (v2.0)	Obsolete in version 2. Please see $viewmode.
$dayviewalldayheight (v2.0)	User defined height for the all-day panel. If set to zero, the panel will size as required, if set to a positive value the height of the panel is fixed to that number of pixels and a scroll bar will be provided if required.
$dayviewalldayheightmax (v2.0)	Maximum height off all day pane in percent (10% to 90%, default 40%). A scroll bar will be shown if not all all-day events can be shown in the provided space. See also $dayviewalldayheight.
$dayviewcancreate	If true, the user can create events in day view mode by clicking and dragging across the calendar background. Feedback will be given during dragging. See evCreateEvent.

$dayviewdate	Date of first day displayed in day view mode. Prior to switching on $dayview, calculating this property to #NULL will use $currday and $dayviewdaycount and $firstday to calculate a sensible $dayviewdate. How it is calculated depends on the setting of $dayviewdaycount; 1 $dayviewdate will be set to $currday. 7 $dayviewdate will be set to $firstday in the week specified by $currday. 14 $dayviewdate will be set to $firstday in a two week period specified by $currday. 21 $dayviewdate will be set to $firstday in a three week period 28 $dayviewdate will be set to $firstday in a four week period specified by $currday 31 $dayviewdate will be set to the first of the month specified by $currday, and only the days of that month are shown. For all other settings, $dayviewdate is calculated so that $currday is centered. See also $firstday.
$dayviewdaycount	The number of days displayed in day mode.
$dayviewdayend	Number of minutes into the day when working hours end, 1020 minutes = 1700 hours. Non-working hours are displayed darker.
$dayviewdaystart	Number of minutes into the day when working hours begin, 540 minutes = 0900 hours. Non-working hours are displayed darker.
$dayviewdeadcolumncolor (v1.3)	Fill color for the left and right dead columns in all day panel.
$dayviewdividercolor (v1.3)	Fill color for the all-day / day-view divider.
$dayviewheadingusedaycolors (v2.0)	If true, day headings are rendered using the colours of the day as used in the client area.
$dayviewhoursvisible	Number of hours displayed in day mode.
$dayviewmineventwidth	The minimum event width in percent of the day column width, when two or more events overlap in time.
$dayviewminutescale (v1.2)	If non zero, additional horizontal minute scale lines are drawn at the specified interval. The valid range is 1 to 30. A value of 0 is the default and draws a scale at 30 minutes to every hour. See also $dayviewtimescale.

$dayviewnowrap (v1.5)	If kTrue, text in an event box is not wrapped. Only a CR character will create a new line.
$dayviewvposminutes (v1.2)	The vertical scroll bar position in minutes from mid-night onwards. This property can be used to position the vertical offset in the report object to simulate the current view in the window calendar component.
$dayviewsnapminutes	The number of minutes events are snapped to when dragging or sizing events. If it is set to 15, events are sized or moved in increments/decrements of 15 minutes.
$dayviewtimecolumncolor (v1.3)	Fill color for the time column in day view.
$dayviewtimecolumntextcolor (v1.3)	Text color for the time column in day view.
$dayviewtimecolumnwidth (v1.5)	Width of the time column in day view. Set it to -1 to hide the time column, set it to zero to let the control size the column, set to > 0 to fix the width of the time column. When reading this property at runtime, it returns the actual width of the time column in pixels.
$dayviewtimescale (v1.2)	If non zero, additional times are displayed in the time column at the specified minute interval. The valid range is 1 to 30. A value of zero is the default and only draws times on the hour. See also $dayviewminutescale.
$ddallowotherfields (v1.4)	If true, the calendar control will allow drops during drag & drop from other fields of any type (default is false). You can still implement evCanDrop in the usual way.
$ddhiliteallday (v1.4)	If true, all-day cells in the day view mode are enabled and highlighted during drag & drop. (default is true)
$ddhiliteworking (v1.4)	If true, working hours time slots in the day view mode are enabled and highlighted during drag & drop. (default is true)
$ddhilitenonworking (v1.4)	If true, non-working hours time slots in the day view mode are enabled and highlighted during drag & drop. (default is true)
$digestcoloridle (v2.0)	Color representing idle range as described by $digestminutesidle.
$digestcolorbusy (v2.0)	Color representing idle range as described by $digestminutesbusy.
$digestminutesidle (v2.0)	Minimum number of minutes required (or events when $digestoptions specifies kCalDigestOptCountEvents) before the idle colour is shown.

$digestminutesbusy (v2.0)	Minimum number of minutes required (or events when $digestoptions specifies kCalDigestOptCountEvents) before the busy colour is shown.
$digestoptions (v2.0)	specifies options for how to collect digest data. One of the following kCalDigestOptNone: No digest options specified kCalDigestOptCountEvents: If specified digest view will count events rather than minutes occupied by events kCalDigestOptIgnoreWorkHours: If specified, events falling partially or completly outside working hours will be included in the digest view in their entirety
$downarrowid	This property allows you to specify icons for the up and down arrows to scroll through events in a single day (Month view only) if there is not enough room to display them all.
$dropdate	The date on which the drop occurred. Set when you receive an evDrop message.
$eventslist	This property takes a list name with the following possible columns; combined Date/Time or separate Date and Time columns, combined End Date/Time or separate End Date and End Time columns, Icon, Display Text, Display Text Color, Display Back Color, Tool-tip text, Event layers (groups this event belongs to).
$firstday	The first day the calendar will show. See also $dayviewdate.
$headingbold	If true, the column headers are drawn in bold
$headingcolor	The fill color for the column headers
$headingfont	The font for the column headers
$headingfontsize	The font size for the column headers
$headingmode	The border style for the column headers
$headingstyle	Alternative text style for the column headers
$headingtextcolor	The text color of the calendar column headers

$holidaylist (v1.5.0)	Specifies a list of public holidays. The list must contain the following columns in the given order: Date: The day, month and year of the special day Fill Color: The background fill color for that day. If kColorDefault, the fill-color is not altered. Text Color: The text color for that day Text: The text to be displayed in the bottom left corner of that day, i.e. the name of the public holiday or any other suitable description. Tooltip: (optional) If specified, it provides the text for a tooltip. One can specify multiple lines of text by separating text with a chr(13) character.
$isoweektext (v1.6.0)	If specified, OCal will display the ISO 8601 week number in both the day and month views. The week number is always displayed in the cells for each Monday. When specifying the text, including a '$' character indicates the point where OCal will insert the week number. If t he '$' character is not present, the week number is appended.
$layers	This property can be set to only show events belonging to the specified groups (layers). It works like a bit array. An event can belong to all your groups or just some or one. The $layers property can be set to show all groups, just some or just one. It is specified as a string of 'Y' for show events and 'N' for hide events. If your calendar supports 7 groups your string may look like "YYNNNYY" which means show groups 1,2,6 and 7.
$monthtextcolor	The color of the days in this month
$mouseallday (v1.4)	Returns true if the mouse is over the all-day pane in the day view.
$mousedate (v1.4)	Returns the date and time under the mouse. The time is adjusted according to $dayviewsnapminutes.
$mouseevent (v1.4)	Returns the line number of the event in the list currently under the mouse. Zero is returned if the mouse is not over an event.
$noeventbackground	If true and $roundedboxes is false, events will ignore event background colors.

$noeventborder (v1.3)	If true, rounded boxes have no extra border around the content making them slightly smaller in appearance.
$otherdaycolor	The fill color of the days in previous and next month
$otherdaymode	The border style for the previous and next months days
$othertextcolor	The text color of the days in previous and next month
$repeatcontent	If true, content is repeated when the event wraps in the day view or month views.
$roundedboxes	If true, events are drawn inside a box with rounded corners. A bit like events in iCal on Mac OSX.
$shortname	If true,the days are drawn using a short name
$showheading	If true,days of the week are shown
$templateallday (v1.5)	Template for rendering content for events in the all-day pane. Please refer to the section Event Templates for a full description.
$templatedayview (v1.5)	Template for rendering content for events in the day view. Please refer to the chapter Event Templates for a full description.
$templatemonthview (v1.5)	Template for rendering content for events in the month view. Please refer to the chapter Event Templates for a full description.
$templatetimecolumn (v1.5)	Template for rendering content in the time column of the day view. Please refer to the chapter Event Templates for a full description.
$templatetimelines (v2.0)	Specifies a template for rendering time lines in the day view. Please refer to the chapter Event Templates for a full description.
$templatetimescale (v1.5)	Template for rendering scale lines in the day view. Please refer to the chapter Event Templates for a full description.
$titlebold	If true, the calendar title is drawn in bold
$titlecolor	The fill color for the calendar title
$titledateformat	The date format used when evaluating the text for the calendar title, i.e. "n y" Shows "January 2006"
$titlefont	The font used for the calendar title
$titlefontsize	The font size for the calendar title
$titlemode	The border style for the calendar title.
$titlestyle	Alternative text style for the calendar title, no need to specify font, font size, etc

$titletext	The calendar title. The '$' character specifies the insertion point for the calendar date with the formatting as specified by $titledateformat
$titletextcolor	The text color of the calendar title
$todaybold	If true, todays date is drawn in bold
$todayscolor	The fill color of today's date
$todaystextcolor	The text color of today's date
$tooltipday	The date to use for the current tool-tip; only significant when $daytooltips is kTrue, and a tool-tip is being generated.
$uparrowid	see $downarrowid.
$viewmode (v2.0)	Specifies the current view mode. Please also see the chapter "oCal…Designing oCal" kCalViewModeMonth Traditional month view (default) kCalViewModeDayNormal Traditional day view with vertical time scales kCalViewModeDayList List based day view without vertical time scales New kCalViewModeMonthDigest Shows events in month view as shades of color ranging from idle to busy (see $digest... properties) New kCalViewModeYearDigest Shows events in year view as shades of color ranging from idle to busy (see $digest... properties) New
$vertscroll (v1.5.0)	Hides or shows the vertical scroll bar in day view.
$vscroll (v1.5.0)	The current vertical scroll position in day view. See also evScroll.
$weekendcolor	The background fill color for Saturdays and Sundays.
$weekenddays (v1.2)	A 7 character mask of 1s and 0s that specify the days that make up the weekend. The first character in the mask is the Sunday, and the seventh character is the Saturday. The default mask for western countries is "1000001".
$weekendtextcolor	The background text color for Saturdays and Sundays.

Events

evCanDrop

Standard Omnis event. Send during dragging of events.

Standard parameters

evClick

This event is generated when the user has clicked on an event. The list lines would have been appropriately selected or de-selected. Shift clicking will select a range of events, ctrl-clicking will add the event to the selection.

This event has no parameters

evCreateEvent

This event is generated when $dayviewcancreate is enabled, and the user clicks and drags the calendar background in day view mode to create an event.

Event Parameter	Description
pStartDate	The start date & time for the new event snapped to $dayviewsnapminutes.
pEndDate	The end date & time for the new event snapped to $dayviewsnapminutes.
pAllDay (v1.3)	If true, the event was created in the all-day pane of the day view.

evDateChange

Sent to the control when the current date is changed.

Event Parameter	Description
pCurrentDate	The new current date.

evDateDClick

Sent to the control when the user double clicks on a date.

Event Parameter	Description
pCurrentDate	The date & time clicked on snapped to $dayviewsnapminutes.

evDateRClick

This event is generated when the user has right clicked on an event or the calendar background.

Event Parameter	Description
pCurrentDate	The date & time clicked on snapped to $dayviewsnapminutes.

evDeleteEvent

Version: 1.3

This event is generated when the user presses the delete or backspace keys. The default action should be to delete all selected events in the event list.

This event has no parameters

evDoubleClick

This event is generated when the user has double-clicked on an event. The list lines would have been appropriately selected or de-selected.

This event has no parameters

evDragFinished

Send when drag & drop has finished

This event has no parameters

evDrop

Standard Omnis event. Send when an event was dropped.

Standard parameters plus $dropdate is set to the date and time where the user let go. The time part of $dropdate will be snapped to $dayviewsnapminutes.

evMonthReset

Sent to the control when the month is changed.

Event Parameter	Description
pCurrentDate	The new current date.

evResizeEventStart

This event is generated when the user has changed the start date/time by dragging the top edge of an event.

This event has no parameters, but $dropdate is set to the date and time where the user let go. The time part of $dropdate will be snapped to $dayviewsnapminutes.

evResizeEventEnd

This event is generated when the user has changed the end date/time by dragging the bottom edge of an event.

This event has no parameters, but $dropdate is set to the date and time where the user let go. The time part of $dropdate will be snapped to $dayviewsnapminutes.

evScroll Version: v1.5 This event is generated when the control is scrolled by the user.
This event has no parameters, but $vscroll is set to the current vertical scroll position.

Methods

$clearicons()

Syntax: *OCalObjectRef*.$clearicons()

Version: 1.0

Sets the icon for the specified day.

Parameter	Description
returns	n/a

$getdayicon()

Syntax: *OCalObjectRef*.$getdayicon(iDay)

Version: 1.0

Gets the icon ID for the specified day.

Parameter	Description
iDay	The date of the current month.
returns	The icon ID

$setdayicon()

Syntax: *OCalObjectRef*.$setdayicon(iDay,iDayIcon)

Version: 1.0

Sets the icon for the specified day.

Parameter	Description
iDay	The date of the current month.
iDayIcon	The icon ID.
returns	n/a

Static Methods

$disablescreenupdates()

Syntax: Calendar Library.$disablescreenupdates()

Version: 1.5

This method can be used to temporarily disable all screen updates to prevent flashing and increase performance when assigning numerous properties to OCal. This call must be balanced with a call to $enablescreenupdates().

Parameter	Description
returns 1 if successful	

$enablescreenupdates()

Syntax: Calendar Library.$enablescreenupdates()

Version: 1.5

Call this method if you have previously disabled screen updates and you have finished modifying the control. OCal will update all affected controls.

Parameter	Description
returns 1 if successful	

$makedatareference()

Syntax: Calendar Library.$makedatareference(cVariableName,&cOutReference)

Version: 1.5

Creates a special reference to an Omnis list containing events for display in OCal. The reference that is returned can be used with $eventslist in any OCal window control even if the list belongs to another class. WARNING. If the data list is destroyed and a OCal control still has a reference to that list, Omnis may crash.

Parameter	Description
cVaraiableName	Name of the Omnis list.
cOutReference	A text based reference is returned in this parameter.
returns 1 if successful	

jsoCal…Designing jsoCal

Introduction

This chapter provides a brief description of the most important aspects of the jsoCal calendar control and their intended design. For a more detailed description of the properties and events please read the chapter “JSON Control Reference”.

Essentially, the jsoCal calendar allows you to display and manipulate a list of events. The list can come from any source but must provide columns with start date/time and optionally an end date/time for each event. You can additionally provide columns for event colors, icons, text, layers, etc.

Important: Your event list must be sorted at all times in ascending order based on the start date/time column of your list.

Events

The jsoCal interface allows the display of events in the traditional time based month and day views as well as in list and digest modes (the latter applies to month and year view only). The day view is capable of displaying any number of days from just 1 day to 7 days for a week, 31 days for a month, 90 days for a quarter of a year or as many days as is physically possible to fit on the screen.

The Event List

jsoCal directly interacts with the list of events that you assign to it via the $eventslist property. When a user clicks an event box, jsoCal automatically updates the current line of your list.When the event box is moved or resized, jsoCal will automatically update the list and generate appropriate events.

Important: When assigning a list to $eventslist, the list must be an instance variable of type list and belong to the remote form that owns the jsoCal control. You cannot specify an item reference variable because of a data handling limitation in the web client.

Event List Columns

As well as specifying the list of events, jsoCal also needs to know the names of the columns that contain the information that jsoCal requires to display events. You can tell jsoCal about your list columns via the set of $column... properties. At a minimum, you must provide a column for $columnstartdate and $columnstarttime. If your list contains just one column with combined date and time, you may assign the same column name to both properties. If your events have a duration you must also provide a name for $columnenddate and $columnendtime.

In addition you can specify columns for the event text, tool tips, event layout/templates and CSS. jsoCal will use the data from these columns to format and display information in the event box. For more details about event templates and CSS, refer to the chapters jsoCal Event Templates and jsoCal CSS.

Important: At all times you must ensure that your list is sorted in ascending order by the starting date and time. If the list is not sorted correctly, events may not be displayed correctly.

Drag and Drop

Users can interact with events by dragging them to move or resize them. When this happens you will receive appropriate events in the $event method.

Layers

jsoCal provides the ability to group your events in layers. There is no theoretical limit to the number of layers. Each event in your list can belong to one or more of these layers. You will need to provide a column in your list that specifies the layers for each event (see $columnlayers). Layers are specified with a series of ‘Y’ and ‘n’ characters. For example, the string “YYYnnY” tells jsoCal that this event belongs to layer 1,2,3 and 6 respectively. You can tell jsoCal to show just one, several or all layers at the same time by setting the property $layers with a similar string. This makes it very easy to show and hide events belonging to individual or a set of layers. How you group layers and what they mean is entirely up to you. The traditional oCal control and

examples used layers to represent calendar groups, such as ‘Home’, ‘Work’, etc.

Views

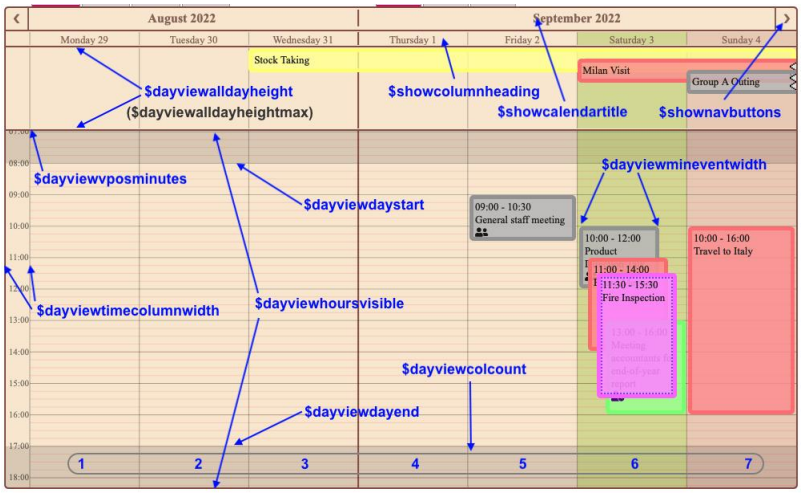
The jsoCal control supports four different view modes and three different view types for displaying events. This sub-section documents each view and their supported types and layout in some detail. Each description of a view and type is accompanied by a graphical example of the layout and how the layout is affected by the properties relevant to the view and mode being discussed. The styling of the various calendar elements is not described here. For details about styling, please refer to the chapter “jsoCal CSS”.

Switching modes and types is achieved by setting the properties $viewmode and $viewtype respectively.

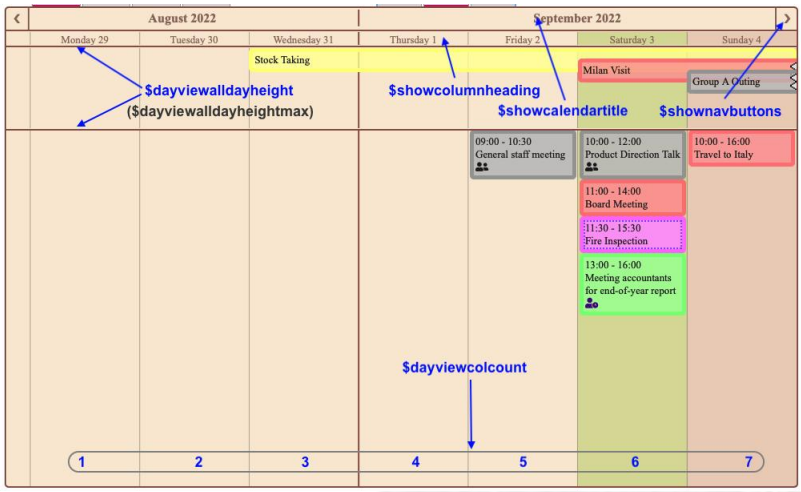
kJSOCalViewDay

The kJSOCalViewDay mode supports two types of display, kJSOCalViewTypeTimed and kJSOCalViewTypeList. There are a number of properties that provide some control over these view modes and types. Their names are typically prefixed with $dayview.. and $show..., more details for which can be found in the chapter jsoCal Reference. Which events are visible will depend on the $layers property and the strings in the layers column (specified by $columnlayers) in your data list.

kJSOCalViewTypeTime:



kJSOCalViewTypeList:

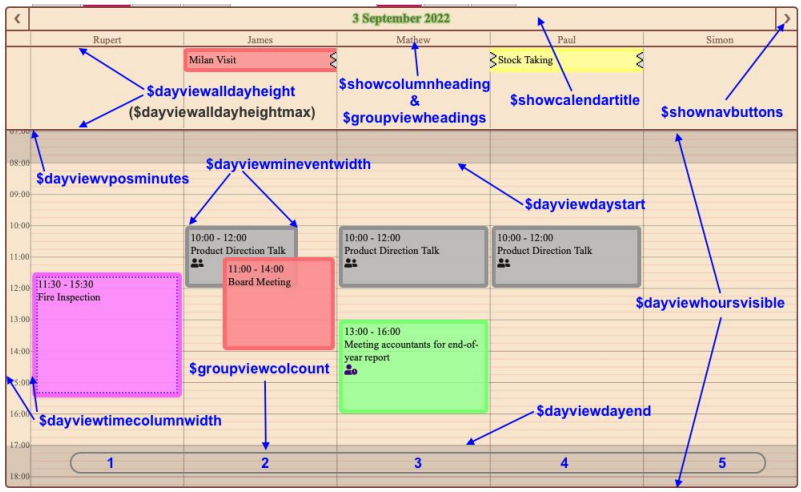


kJSOCalViewGroup

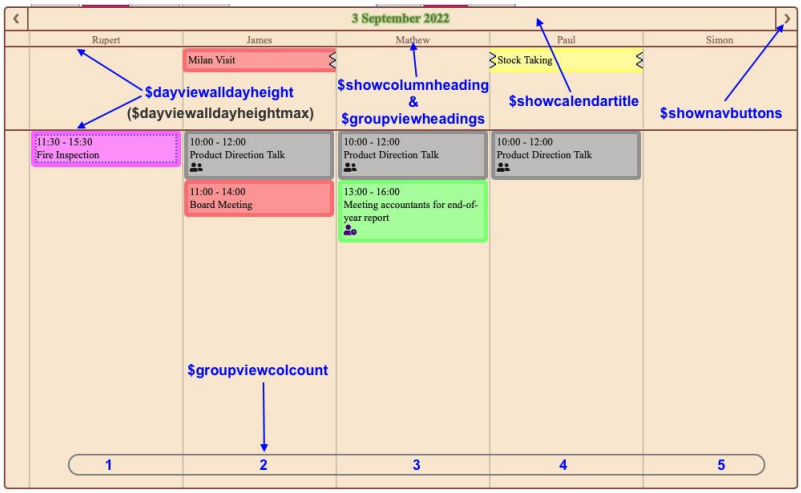
The group view is very similar to the day view in terms of layout, except that it displays columns of your own groupings using the layers feature. Many of the $dayview properties also apply to this view with a few different properties that apply only to the group view. These additional properties are prefixed with $group..., details for which can be found in the chapter "jsoCal Reference".

The property $groupviewlayersarray determines which events appear in which group columns. For example, if we assigned "Ynnnn,nYnnn,nnYnn,nnnYn,nnnnY", events that specified 'Y' for layer 1 would be shown in column 1, events that specified 'Y' for layer 2 would be shown in column 2, and so on. Some events may be displayed in more than one group. The provided examples use layers to group events to individuals. The group view then displays the events belonging to different individuals in different columns of the view. When an event is assigned to multiple individuals as in the sample event "Product Direction Talk" (see image below), the event will be shown in the columns of all the individuals that share the event. The shared event in question specifies "nYYYn" for its layers.

kJSOCalViewTypeTimed:



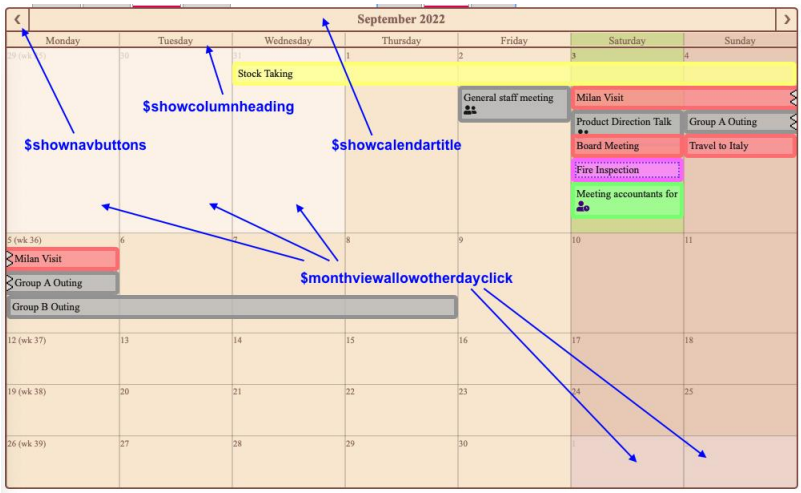
kJSOCalViewTypeList:



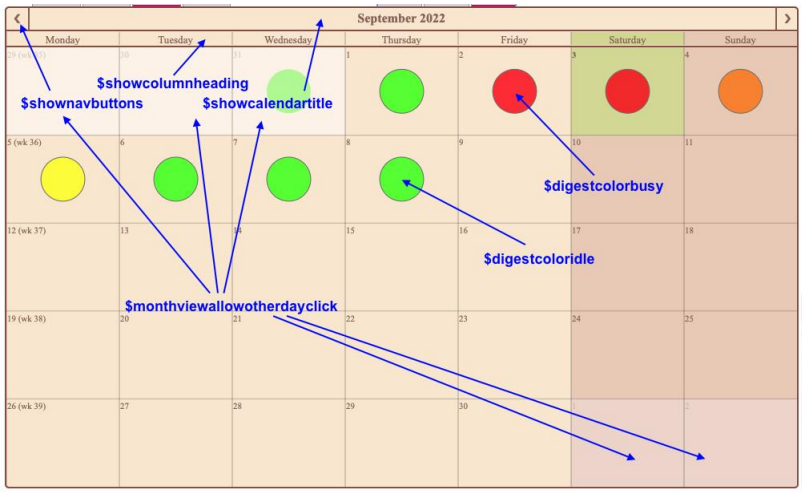
kJSOCalViewMonth

The traditional month view displays events one month at a time showing seven days along the horizontal axes and the weeks of the month along the vertical axes. Other than the header and title related properties, there are no specific month view properties that manipulate the layout. The only functional property is $monthviewallowotherdayclick, which enables or disables clicks on fringe days that belong to the previous or next month for moving back or forward a month. The month view supports both kJSOCalViewTypeList and kJSOCalViewTypeDigest view types. Assigning the kJSOCalViewTypeTimed type is meaningless within the month view and will revert to the event list view.

kJSOCalViewTypeList:



kJSOCalViewTypeDigest:

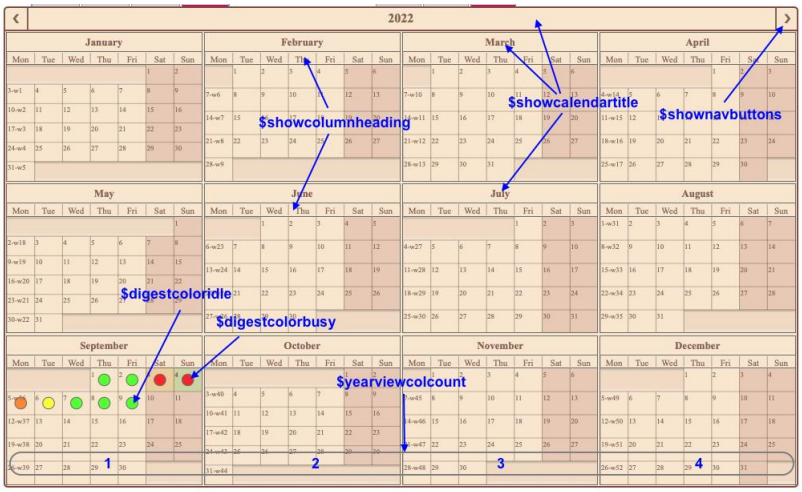


The digest colors and how they are calculated is controlled by a range of digest properties, $digestcoloridle, $digestcolorbusy, $digestminutesidle, $digestminutesbusy, $digestoptions.

kJSOCalViewYear

The only valid view type for the year view is kJSOCalViewTypeDigest. Effectively, the year view displays twelve month views arranged in a grid that is controlled by the property $yearviewcolcount (valid settings are: 0,1,2,3,4*,6 or 12, 0 = auto size). When zero is specified, jsoCal attempts to choose the most appropriate column/row count combination for the width and height of the calendar.

kJSOCalViewTypeDigest:



The digest colors and how they are calculated is controlled by a range of digest properties, $digestcoloridle, $digestcolorbusy, $digestminutesidle, $digestminutesbusy, $digestoptions. jsoCal uses HSV color manipulation techniques to produce smooth color transitions from between the specified idle and busy colors.

Further Reading

The Brainy Data support website lists a number of technical notes relating to developing our software. You should always read these notes before you begin developing our software and keep an eye on technical notes as they appear.

The chapter jsoCal Event Templates explains in detail how to compose your own templates for the visual display of events and the chapter jsoCal CSS introduces the styling of various calendar areas and views using CSS styles in the supplied cascading style sheet *ctl_ocal.css*.

The technical document TN0022 explains about our component version numbers in more detail.

jsoCal…Event Templates

The implementation of templates in jsoCal differs from that in oCal desktop. Consequently, if you have become familiar with the oCal desktop ways of templating, you are well advised to study this chapter to fully understand the differences. One essential difference is that you will be using standard XHTML and CSS, instead of just a subset of elements that look a bit like HTML.

In jsoCal, there are three ways of providing HTML and background CSS to lay out and style events (**Note** that the backgrounds are rendered using SVG so the CSS properties must relate to standard SVG style properties).

1. The event list (specified by $dataname) can contain two columns, one for the XHTML and one for the CSS that will be used to present this particular event. Note, the HTML must begin with an opening tag marker character '<', for jsoCal to identify the content of this column as raw XHTML data.The names of the two columns are specified by the properties $columntemplate and $columnbackgroundcss.
2. The column $columntemplate, instead of providing XHTML directly which is indicated by a starting'<' character, could instead point to a named template in a template list that is specified by the property $templatelist. Thus the specific event would be laid out and styled by the XHTML and CSS in that template. This makes it easier for events to share a common template.
3. The third option is to have specially named templates provided by the template list, that provides layout and style information based on the different view modes and types supported by jsoOcal.

The first and second choices are somewhat self-explanatory, but it is the third option that requires further explanation.

The Template List

The template list is a three column list that specifies a template name, an XHTML template for the event content and the CSS for the event background.

The template name can be one of the following:

- An arbitrary name that may be referred to by one or more events via the data list column specified by $columntemplate.
- The view name, one of "DayView", "GroupView", "MonthView" or "YearView". If specified, it sets the default template for a specific view.
- The view type name, one of "AllDay", "Timed", "List", "Digest". If specified, it sets the default template for a specific view type.
- The view type name in conjunction with the view name separated by a dot. For example "DayView.AllDay". If specified, it sets the default template for a specific view type in a specific view.

Embedding Data

The XHTML part of the template can refer to data columns in your event data list using the following syntax:

$$fmt(column_name,format_string)$$

when the data is to be formatted using Omnis formatting strings, or

$$column_name$$

when the data is not to be formatted.

The jsoCal code will look out for these embedded references within the HTML and replace it with the data from the list. The format strings are standard Omnis JS Client data formatting strings as documented at

https://omnis.net/developers/resources/onlinedocs/index.jsp?detail=JavaScriptSDK/01overview.html

You will need to scroll about a third of the way down to the sections on “Date Formatting” and “Number Formatting”.

Note: Text data in your Omnis list that is to be displayed in an event may also contain HTML.

Template Priority

When using view based templates they have specific priorities. Put simply, the view mode overrides view type and view mode.type overrides all others.

For example: you could specify a template specifically for the view type kJSOCalViewTypeList for all view modes by creating a template named “List”. You could also create a template specifically for the group’s list view using the name “GroupView.List”. Consequently, if you have two templates named “GroupView” and “List”, the former overrides the latter in the list based group view. If you then add a third template called “GroupView.List”, this overrides the former two within the list based group view. The full list of possible view based templates and their priority is as follows (grouped according to order of priority):

Template Name	**(Priority)**	**Description**
DayView.AllDay	(1)	Applies to all-day events in any day view
GroupView.AllDay		Applies to all-day events in any group view
DayView.Timed	(2)	Applies to all events in timed day view
DayView.List		Applies to all events in list day view
GroupView.Timed		Applies to all events in timed group view
GroupView.List		Applies to all events in list group view
MonthView.List		Applies to all events in list month view

DayView	(3)	Applies to all events in any day view
GroupView		Applies to all events in any group view
MonthView		Applies to all events in any month view
AllDay	(4)	Applies to all all-day events
Timed		Applies to all events displayed in timed view type
List		Applies to all events displayed in list view type

The following template names are currently meaningless either because the view mode.type combination is not fully supported, or the view/type displays digest information only:

DayView.Digest, GroupView.Digest, MonthView.Timed, MonthView.Digest, YearView, YearView.Timed, YearView.List, YearView.Digest, Digest.

Theoretically, one could create templates for unsupported mode/type combos and jsoCal will use these templates if the combo is selected using the $view... properties. However, how events are displayed within this mode is unpredictable as the code may not fully cater for invalid display options.

Default Templates and tooltips

The following are the default templates used by jsoCal when no alternatives are specified.

- all-day events and events in month view

```
<p style="margin:0">$$Text$$</p>
# $$Text$$ is the place holder for the text from the column specified by $columntext
```

- all other events

```
<p style="margin:0">$$fmt(StartTime,DefTimeFormat)$$ -
   $$fmt(EndTime,DefTimeFormat)$$<br>$$"Text$$</p>"
# StartTime and EndTime refer to the columns specified by $columnstarttime and $columnendtime
# respectively, and DefTimeFormat refers to the format string specified by the jsoCal string table entry
# with the same name.
```

When no template can be found for an event, jsoCal also generates default tooltip text that it will use for the title property of the event's div box.

- all-day events

```
$$fmt(StartDate,DefDateFormat)$$ - $$fmt(EndDate,DefDateFormat)$$ (all-day)
# StartDate and EndDate refer to the columns specified by $columnstartdate and $columnenddate
# respectively, and DefDateFormat refers to the format string specified by the jsoCal string table entry
# with the same name.
```

- all other events

```
$$fmt(StartDate,DefDateFormat)$$ $$fmt(StartTime,DefTimeFormat)$$ -
   $$fmt(EndDate,DefDateFormat)$$ $$fmt(EndTime,DefTimeFormat)$$
```

If your event list specifies tooltip text via the $columntooltip property, the above tooltip text is appended to your tooltip text following a line break. If you intent to specify your own templates,

your tooltip text in the event list should incorporate any data references as shown above.

jsoCal…CSS

While the standard Omnis appearance and text properties control the overall appearance of the calendar, every styling aspect of the jsoCal views and events is built around the use of CSS provided by a single file. By default, the cascading styles within this file inherit the appearance as set by the Omnis properties and merely modify various aspects using a heavy helping of transparency. For example, Omnis may set the overall background color, but the jsoCal style for a weekend day modifies the background using a red fill and low opacity as shown in the image below, thus merely touching up the inherited background color with a hint of red.

```
.jsocal .backgroundWE { /* svg background for weekend days */
  stroke:none; /* no border */
  fill:rgb(128,0,0);
  fill-opacity:0.1;
}
```

Pretty much all of the CSS styles are designed in this way while taking full advantage of the cascading feature of CSS with the root class name called ".jsocal". The various style classes within the file are organised into functional groups which are identified by their cascaded names.

The "SVG Divider Lines" are used to divide the days, month, hours and minute intervals in the various views and specify the stroke features of these lines:

.jsocal .vertDividerLineMajor { /* used as month divider */
.jsocal .vertDividerLineMinor { /* used as day divider */
.jsocal .horzDividerLineMajor { /* seperates hours */
.jsocal .horzDividerLineMinor { /* seperates $dayviewtimescale minutes */

Example:

```
.jsocal .vertDividerLineMajor { /* used as month divider */
  stroke:inherit;
  stroke-width:2;
  stroke-linecap:butt;
  stroke-opacity: 1;
}
```

The "SVG Backgrounds" provide the styles for the various backgrounds for the days in the various views and mainly specify the fill color and opacity:

.jsocal .background { /* svg background for all days (used as place holders) */
.jsocal .backgroundWE { /* svg background for weekend days */
.jsocal .backgroundHO { /* svg background for holidays */
.jsocal .backgroundDIS { /* svg background for disabled days (typically other month days in year view) */
.jsocal .backgroundSHDT { /* svg background top shadow for disabled days (typically other month days in year view) */
.jsocal .backgroundSHDB { /* svg background bottom shadow for disabled days (typically other month days in year view) */

Example:

```
.jsocal .backgroundWE { /* svg background for weekend days */
  stroke:none; /* no border */
  fill:rgb(128,0,0);
  fill-opacity:0.1;
}
```

The “SVG Text” group of styles specify the text attributes of the various areas of the calendar views:

.jsocal .titleBox .textWE { /* svg title text for weekend days in group view */
.jsocal .titleBox .textTD { /* svg title text for today day in group view */
.jsocal .monthViewBox text { /* svg date text for normal days in month view */
.jsocal .monthViewBox .textHO { /* svg date text for holidays in month view */
.jsocal .monthViewBox .textWE { /* svg date text for weekend days in month view */
.jsocal .monthViewBox .textTD { /* svg date text for today day in month view */
.jsocal .monthViewBox .textCD { /* svg date text for current/selected day in month view */
.jsocal .monthViewBox .textOM { /* svg text for dates of other month days in month view */

Example:

```
.jsocal .titleBox .textWE { /* svg title text for weekend days in group view */
  stroke: rgb(128,0,0); /* we frame our text red to indicate it is a weekend day */
  stroke-width:3px;
  stroke-opacity:0.25;
  fill:inherit;
}
```

The “DIV box styles” group provides the styles for the div boxes of the major areas of the calendar:

.jsocal .titleBox { /* div styles for title box */
.jsocal .headerBox { /* div styles for dayview column headers box */
.jsocal .allDayViewBox { /* div styles for dayview all-day events box */
.jsocal .dayViewSubBox { /* div styles for day view events box covering 24 hour range */
.jsocal .dayViewBox { /* div styles for day view scroll box showing visible hour range */
.jsocal .monthViewBox { /* div styles for month view showing all days of a month in a grid */
.jsocal .yearViewMonthBox .monthViewBox { /* div styles for month view showing all days of a month in a grid
.jsocal .yearViewBox { /* div styles for year view showing all months of a year in a grid */
.jsocal .yearViewMonthBox { /* div styles for month cell within year view */
.jsocal .overlayBox { /* div styles for day view overlay box which mainly displays details of holidays */

Example:

```
.jsocal .monthViewBox { /* div styles for month view showing all days of a month in a grid */
  position:relative;
  border: none;
  overflow-x:hidden;
  overflow-y:scroll;
  width: 100%;
  height: auto;
  flex-grow:1; /* this view occupies the space that remains after all other areas (i.e. title
}
```

The “DIV Event Styles” group provides the box styles for the events within different major areas of the calendar. There are two styles, eventBox and eventBoxSelected that cascade from the major area styles. In addition there are two additional styles that deal with boxes around event content:

.jsocal .allDayViewBox .eventBox { /* div styles for individual event box within the all-day view */
.jsocal .allDayViewBox .eventBoxSelected { /* div styles for individual event box within the all-day view */
.jsocal .dayViewBox .eventBox { /* div styles for individual event box within the all-day view */
.jsocal .dayViewBox .eventBoxSelected { /* div styles for individual event box within the day view */
.jsocal .monthViewBox .eventBox { /* div styles for individual event box within the all-day view */
.jsocal .monthViewBox .eventBoxSelected { /* div styles for individual event box within the day view */
.jsocal .eventContentSelected { /* div styles for when event is selected */
.jsocal .eventContent { /* div styles for when event is not selected */

Example:
```
.jsocal .eventContentSelected {
  position:relative;
  width:100%;
  height:100%;
  border:2px dotted rgba(0,0,255,0.5);
  border-radius:none;
  padding:0;
  background-color: inherit;
  opacity:1;
  box-sizing:border-box;
  cursor:inherit;
  /*background: linear-gradient(0deg, rgba(0,0,0,0.1) 0%, rgba(255,255,255,0.1) 100%);*/
}
```

The "SVG Event Drag Bar Styles" group are responsible for animating the drag bars of an event box when the mouse hovers over or activates the drag bar:

.jsocal .eventDragBar { /* svg line styles for event box drag bars (inactive) */
.jsocal .eventDragBar:hover,.jsocal .eventDragBar:active { /* svg line styles for event box drag bars (when active) */
.jsocal .eventDragBar.top,.jsocal .eventDragBar.bottom { /* svg line styles for event box drag bars (vertical sizing and moving)
.jsocal .eventDragBar.left,.jsocal .eventDragBar.right { /* svg line styles for event box drag bars (horizontal sizing and moving)
.jsocal .eventDragBar.all { /* svg line styles for event box drag bars (moving all directions) */
.jsocal .eventDragBar.vert { /* svg line styles for event box drag bars (moving vertical only) */
.jsocal .eventDragBar.horz { /* svg line styles for event box drag bars (moving horizontal only) */

Example:
```
.jsocal .eventDragBar { /* svg line styles for event box drag bars (inactive)
  pointer-events: auto;
  stroke: rgba(0,0,0,0);
  stroke-width:5;
  stroke-linecap:round;
  background-color: rgba(0, 0, 0, 0);
  position:absolute;
  width:5px;
  height:5px;
}
.jsocal .eventDragBar:hover,.jsocal .eventDragBar:active { /* svg line styles
  stroke: rgba(0,0,0,0.25);
  background-color: rgba(0,0,0,0.25);
}
```

The “SVG Event Navigate Button Styles” group provides the styles for the version 1.1.0.0 title bar navigation buttons:

.jsocal .navigateButton { /* style for inactive navigation button */
.jsocal .navigateButton:hover,.jsocal .navigateButton:active { /* style for active navigation button */
.jsocal .navigateButton.prev { /* style for left navigation button */
.jsocal .navigateButton.next { /* style for right navigation button */

Example:

```
.jsocal .navigateButton { /* style for inactive navigation button */
  pointer-events: auto;
  background-color: rgba(0, 0, 0, 0);
  position:absolute;
  height:100%;
  top:0px;
  font-size:inherit;
  font-weight:normal;
  font-style: normal;
  text-align: center;
  border-style: solid;
}
```

The “Day View Text Styles” group provides the styles for the background text in the day view, namely the time column and the holiday descriptions:

.jsocal .dayViewBox text.timeColumn {
.jsocal .dayViewBox text.holidays {

jsoCal…JSON Control Reference

Introduction

This chapter lists all the properties, methods and events as provided by the JSON control. The jsoCal JSON control provides the IDE interface for designing a remote form control for use in browsers. Some of these properties will be similar to those in the oCal desktop control, but there will also be some new properties, methods and events, some will be missing and some will differ functionally. Hence the decision was made to keep the desktop and JSON control reference separate.

Contents

Properties	
Property	**Description**
$allowchange	If true,the user can change the current date by clicking on different days in the applicable views.
$ampmstring	String that specifies the am/pm characters separated by a '/'
$column...	The $column... properties tell jsoCal in which columns of the $eventlist to find the relevant data. Not all columns are required, but as a minimum the start date and time and the event description must be specified for the calendar to display meaningful events. Below is the entire list of all supported columns and their associated $column... property names:
	$columnstartdate: The name of the list column that specifies the event's date (required).
	$columnstarttime: The name of the list column that specifies the event's time (required).
	$columnenddate: The name of the list column that specifies the event's end date (optional)
	$columnendtime: The name of the list column that specifies the event's end time (optional)
	$columnallday: The name of the list column that specifies the event's all-day state (optional) Note: events that cross day boundaries are automatically considered all-day events without consulting this flag.
	$columnlayers: The name of the list column that specifies the layers string for this event, i.e which layer(s) the event belongs to (optional). This property can be populated with a series of 'Y' and 'n' characters, indicating which layer or layers the event belongs to. See $layers for more details.
	$columntext: The name of the list column that specifies the events main text (optional)
	$columntooltip: The name of the list column that specifies the events tooltip text (optional)

	$columntemplate: The name of the template for the event or raw HTML beginning with the '<' character if the event has a unique custom template (optional). Read the chapter "jsoCal Event templates" for more details.
	$columnbackgroundcss: SVG style for rendering background (overrides class settings in css style sheet)
$currentdate	The calendar's current date as a string (leave empty to select today during construct)
$dataname (Omnis Property)	This is the controls data bound property that specifies the name of the list that contains the event data for the events to be displayed. The $column... properties tell jsoOcal which of the data columns contain specific data that jsoCal requires. Making changes to the events list associated with this property should trigger the Omnis web client to redraw the calendar control.
$dayviewalldayheight	All-day panel height, zero = auto-size, >0 = fixed height in pixels. See also $dayviewalldayheightmax.
$dayviewalldayheightmax	Maximum height of the all day pane in percent (10% to 90%, default 40%). See also $dayviewalldayheight.
$dayviewcolcount	Number of days displayed in day view. Supported values are 1, 7, 14, 21, 28, 29, 30 and 31.
$dayviewdayend	Time in hours (24hour clock) when day ends. Working hours and non-working are drawn using different CSS styles for the background.
$dayviewdaystart	Time in hours (24hour clock) when day starts. Working hours and non-working are drawn using different CSS styles for the background.
$dayviewhoursvisible	The number of hours that are visibly displayed in day view.
$dayviewmineventwidth	The minimum width of an event in percent of the day column width. Default is 50.
$dayviewsnapminutes	The minutes of the hour at which events are snapped during drag & drop. Default is 15.
$dayviewtimecolumnwidth	Time column width, zero = auto-size, -1 = hide time column, >0 = fixed width in pixels.
$dayviewtimescale	If non-zero, additional time values are displayed at specified minute intervals. Valid range is 0 - 30. Default is 30.

$dayviewvposminutes	The vertical scroolbar position in minutes from mid-night onwards
$digestcolorbusy	Color representing busy range as described by $digestminutesbusy.
$digestcoloridle	Color representing idle range as described by $digestminutesidle.
$digestminutesbusy	Minimum number of minutes required (or events when $digestoptions specifies kJSOCalDigestOptCountEvents) before the busy color is shown.
$digestminutesidle	Maximum number of minutes required (or events when $digestoptions specifies kJSOCalDigestOptCountEvents) before the idle color is shown.
$digestoptions	Specifies options for how to collect digest data (one of the kJSOCalDigestOpt... constants): kJSOCalDigestOptNone: No digest options specified, no digest info will be displayed. kJSOCalDigestOptCountEvents: If specified digest view will count events rather than minutes occupied by events kJSOCalDigestOptIgnoreWorkHours: If specified, events falling partially or completely outside working hours will be included in the digest view in their entirety. kJSOCalDigestOptIgnoreLayers: If specified, events belonging to multiple layers are only counted once. kJSOCalDigestOptCountEventsAndIgnoreLayers: If specified digest view will count events rather than minutes occupied by events (multi-layer events are only counted once) kJSOCalDigestOptIgnoreWorkHoursAndLayers: If specified, events falling partially or completly outside working hours will be included in the digest view in their entirety (multi-layer events are only counted once)
$eventslist	This property is obsolete as of version 1.1.0.0. The property $dataname should be used to specify the list that provides the calendar events.

$firstday	First day of the week, typically Sunday or Monday (used in day and month view). Use the constants kJSOCalSunday through kJSOCalSaturday.
$groupviewcolcount	Number of groups displayed in group view.
$groupviewheadings	String that specifies the comma list of group view headings. Default is "Group 1,Group 2,Group 3,Group 4,Group 5"
$groupviewlayersarray	String that specifies the comma list of layers to be shown for multiple groups within the group view, a 'Y' at a character position means show the events that specify a 'Y' at the same pos in the list column specified by $columnlayers (the last character in the string specifies the state for all remaining layers). Default is "Ynnnn,nYnnn,nnYnn,nnnYn,nnnnY"
$headershortname	If non-zero, the days are drawn using a short name constructed from the first n characters of the jsoCal string table's full day names.
$holidaylist	Dataname of list that specifies the calendar holidays. The list must specify two columns. The first column is the date and the second the description. The jsoCal CSS file specifies the background and text styles for displaying holiday days. See the chapter "Designing jsoCal" for more details.
$isoweektext	If set, the ISO week number is displayed with the specified text in the month view, use '$' for ISO week place holder.
$layers	String that specifies the layers to be shown, a 'Y' at a character position means show the events that specify a 'Y' at the same pos in the list column specified by $columnlayers. Note: The last character in a layers string specifies the state for all remaining layers.
$monthviewallowotherdayclick	If true,the user can click other month days to change to the previous or next month
$navbuttonstext	The html for both prev & next navigation buttons seperated by a '~' character (i.e. '❮~❯' or '~'). Any HTML and style will be valid but some type of elements may not work as expected.
$showcalendartitle	Show the calendar title. What is shown will depend on the current $viewmode.
$showcolumnheading	Show the calendar day headings for the month and day view and the months in the year view.

$shownavbuttons	Show navigation buttons in calendar title if the title is visible. Navigation buttons will advance the calendar date $currentdate by a day, week, month etc, depending on the current view settings. Specify one of the kJSOCalNav... constants kJSOCalNavNone: Do not show any navigation buttons. kJSOCalNavPrev: Show previous day/week/month/year button. kJSOCalNavNext: Show next day/week/month/year button. kJSOCalNavAll: Show both navigation buttons.
$templatelist	Dataname of list that specifies custom templates and background CSS. The columns are: TemplateName: The name of the template either specifying a view mode and/or type, or a custom name which can be referred to directly by one or more events. ContentTemplate: The XHTML for displaying an event's content. BackgroundCSS: The CSS for specifying an event's background style. Please read the chapter "jsoCal Event Templates" for more details.
$titletext	The calendar title. The '$' character specifies the insertion point for the calendar date. Default is "$"
$viewmode	Specifies the current calendar display mode, one of the kJSOCalView... constants. kJSOCalViewDay: Standard day view, $dayviewcolcount specifies number of days to display horizontally kJSOCalViewGroup: Group view, $dayviewcolcount specifies number of groups to display horizontally kJSOCalViewMonth: Standard month view kJSOCalViewYear: Standard year view, displays 12 month views in grid specified $yearviewcolcount. This property is used in conjunction with $viewtype.

$viewtype	Specifies how events are displayed within the current view mode, one of the kJSOCalViewType... constants. kJSOCalViewTypeTimed: Standard display type for the day and group views. It is currently meaningless when applied to the month and year view. kJSOCalViewTypeList: Events are listed vertically, in order of occurrence within the day, group and month views. It is currently meaningless when applied to the year view. kJSOCalViewTypeDigest: Events are displayed in digest format. See $digestoptions for more details.
$weekenddays	A seven character mask that specifies the weekend days. The first character maps to Sunday and the last character to Saturday. A 'W' character specifies a weekend day. Default is “WnnnnnW”.
$yearviewcolcount	Number of months displayed horizontally within the year view (valid settings are: 0,1,2,3,4*,6 or 12, 0 = auto size)

Events

evClick

This event is generated when the user has clicked on an event or the calendar background. The list lines would have been appropriately selected or de-selected.

pLineNumber	The line number of the event in the list that was clicked or zero when clicking the calendar background

evDateChange

Sent when the current date is changed.

Event Parameter	Description
pCurrentDate	The new current date.
pAllDay	If true, change occured because of a click in the all-day event area.

evDateDClick

Sent when the user double clicks on a day.

Event Parameter	Description
pCurrentDate	The date of the clicked day
pAllDay	If true, click occured in all day-event area.
pGroupLayers	The layers string of the group column that was clicked.

evDateRClick

Sent when the user right clicks on a day.

Event Parameter	Description
pCurrentDate	The date of the clicked day
pAllDay	If true, click occured in all day-event area.
pGroupLayers	The layers string of the group column that was clicked.

evDoubleClick

This event is generated when the user has double-clicked on an event or the calendar background. The list lines would have been appropriately selected or de-selected.

This event has no parameters

evMoveEvent

Sent when an event has been dragged to a different date/time

This event has no parameters
evResizeEventStart Sent when the user has dragged the start date/time of an event.
This event has no parameters, but the current line in the list can be queried.
evResizeEventEnd Sent when the user has dragged the end date/time of an event.
This event has no parameters, but the current line in the list can be queried.