

What's New in Omnis Studio 11.2

Omnis Software

August 2024

65-082024-01

The software this document describes is furnished under a license agreement. The software may be used or copied only in accordance with the terms of the agreement. Names of persons, corporations, or products used in the tutorials and examples of this manual are fictitious. No part of this publication may be reproduced, transmitted, stored in a retrieval system or translated into any language in any form by any means without the written permission of Omnis Software.

© Omnis Software, and its licensors 2024. All rights reserved.

Portions © Copyright Microsoft Corporation.

Regular expressions Copyright (c) 1986,1993,1995 University of Toronto.

© 1999-2024 The Apache Software Foundation. All rights reserved.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

Specifically, this product uses Json-smart published under Apache License 2.0

(<http://www.apache.org/licenses/LICENSE-2.0>)

© 2001-2024 Python Software Foundation; All Rights Reserved.

The iOS application wrapper uses UIKeyChainStore created by <http://kishikawakatsumi.com> and governed by the MIT license.

Omnis® and Omnis Studio® are registered trademarks of Omnis Software.

Microsoft, MS, MS-DOS, Visual Basic, Windows, Windows Vista, Windows Mobile, Win32, Win32s are registered trademarks, and Windows NT, Visual C++ are trademarks of Microsoft Corporation in the US and other countries.

Apple, the Apple logo, Mac OS, Macintosh, iPhone, and iPod touch are registered trademarks and iPad is a trademark of Apple, Inc.

IBM, DB2, and INFORMIX are registered trademarks of International Business Machines Corporation.

ICU is Copyright © 1995-2024 International Business Machines Corporation and others.

UNIX is a registered trademark in the US and other countries exclusively licensed by X/Open Company Ltd.

Portions Copyright (c) 1996-2024, The PostgreSQL Global Development Group

Portions Copyright (c) 1994, The Regents of the University of California

Oracle, Java, and MySQL are registered trademarks of Oracle Corporation and/or its affiliates

SYBASE, Net-Library, Open Client, DB-Library and CT-Library are registered trademarks of Sybase Inc.

Acrobat is a registered trademark of Adobe Systems, Inc.

CodeWarrior is a trademark of Metrowerks, Inc.

This software is based in part on ChartDirector, copyright Advanced Software Engineering (www.advsofteng.com).

This software is based in part on the work of the Independent JPEG Group.

This software is based in part on the work of the FreeType Team.

Other products mentioned are trademarks or registered trademarks of their corporations.

Table of Contents

ABOUT THIS MANUAL.....	4
SOFTWARE SUPPORT, COMPATIBILITY AND CONVERSION ISSUES	5
<i>Serial Numbers and Licensing</i>	5
<i>Library and Datafile Conversion</i>	5
<i>Studio 11.2</i>	5
<i>macOS Tree Restructure</i>	5
<i>Subform Client Commands</i>	5
WHAT'S NEW IN OMNIS STUDIO 11.2 REVISION 38542.....	6
JAVASCRIPT REMOTE FORMS.....	6
<i>Subformset Panel</i>	6
<i>Construct Row Variable</i>	9
<i>HTTP Server</i>	9
APPLICATION SETUP & DEPLOYMENT	9
<i>macOS Tree Restructure</i>	9
<i>Notarization</i>	10
OMNIS ENVIRONMENT	10
<i>Studio Now Sample Libraries</i>	10
OMNIS LIBRARIES	11
<i>Recent Libraries</i>	11
REPORT PROGRAMMING.....	11
<i>Enterable Report Fields</i>	11
WINDOW COMPONENTS	11
<i>Tab Strip</i>	11
<i>OBrowser Configuration</i>	12
OMNIS VCS	12
<i>Update from VCS</i>	12
<i>Update Local Library</i>	12
FUNCTIONS	12
<i>idletime()</i>	12
WHAT'S NEW IN OMNIS STUDIO 11.2 REVISION 38349.....	13
JAVASCRIPT REMOTE FORMS.....	13
<i>Remote Form Properties</i>	13
OMNIS ENVIRONMENT	13
<i>Omnis Configuration</i>	13
<i>SQL Query Builder</i>	13
OMNIS VCS	14
<i>VCS Privileges</i>	14
OW3 WORKER OBJECTS.....	14
<i>OAuth2 Worker Object</i>	14
FUNCTIONS	14
<i>rcredit</i>	14
<i>bool()</i>	15
<i>bitclear()</i>	15
<i>replace() and replaceall()</i>	15
<i>FileOps Workers</i>	16
APPENDIX	17
MACOS TREE RESTRUCTURE	17
<i>Online Documentation changes</i>	17

About This Manual

This document describes the new features and enhancements in **Omnis Studio 11.2** Revision 38542.

See the Studio Now tab in the Studio Browser for details of bug fixes in this revision.

See the **Install.txt** file to find out System Requirements for running the Development and Server versions of Omnis Studio 11.2.

NOTE: Where a new feature or an enhancement relates to an Enhancement Request or Customer reported fault, the fault reference is included to enable you to track your own ERs and reported faults.

Software Support, Compatibility and Conversion Issues

Serial Numbers and Licensing

You will require a new serial number to run Omnis Studio 11.2. Contact your local sales office to buy a license or obtain an upgrade serial number under your current support program; go to the Contacts page on the Omnis website: www.omnis.net

Library and Datafile Conversion

Converting 10.x Libraries

All Omnis Studio 10 or earlier libraries need to be converted to run in Omnis Studio 11.1 or 11.0. ONCE A STUDIO 10.0, 10.1 or 10.2 LIBRARY HAS BEEN OPENED WITH OMNIS STUDIO 11.x IT CANNOT BE OPENED WITH STUDIO 10.x – THE CONVERSION PROCESS IS IRREVERSIBLE.

Converting 8.x or earlier Libraries

OMNIS STUDIO 11.1 or 11.0 WILL CONVERT EXISTING VERSION 8.1.X, 8.0.X, 6.1.X, 6.0.X AND 5.X LIBRARIES – THE CONVERSION PROCESS IS IRREVERSIBLE.

***DISCLAIMER:** OMNIS SOFTWARE LTD. DISCLAIMS ANY RESPONSIBILITY FOR, OR LIABILITY RELATED TO, SOFTWARE OBTAINED THROUGH ANY CHANNEL. IN NO EVENT WILL OMNIS SOFTWARE BE LIABLE FOR ANY INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES HOWEVER THEY MAY ARISE AND EVEN IF WE HAVE BEEN PREVIOUSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.*

Studio 11.2

The following section contains issues regarding software support, compatibility and conversion in **Omnis Studio Now 11.2**.

macOS Tree Restructure

On macOS the application tree has been restructured to adhere to the Apple best practice guidelines. See the notes under 'Application Setup & Deployment' for more information.

Subform Client Commands

The subform set/dialog/palette parameters string now expects a comma as the separator regardless of \$prefs.\$language. (Revision 38303, ST/JS/3528)

A fault in the subform set/dialog/palette client commands (e.g. \$clientcommand('subformdialogshow',...)) meant that comma parameter separators were failing when the language in \$prefs.\$language was set to anything other than English. This has been fixed and using a comma as a parameter separator in the subform set/dialog/palette client commands now works as expected. Therefore, if you have created a workaround for this issue, your code may not work now and if this is the case it will need to be amended.

What's New in Omnis Studio 11.2 Revision 38542

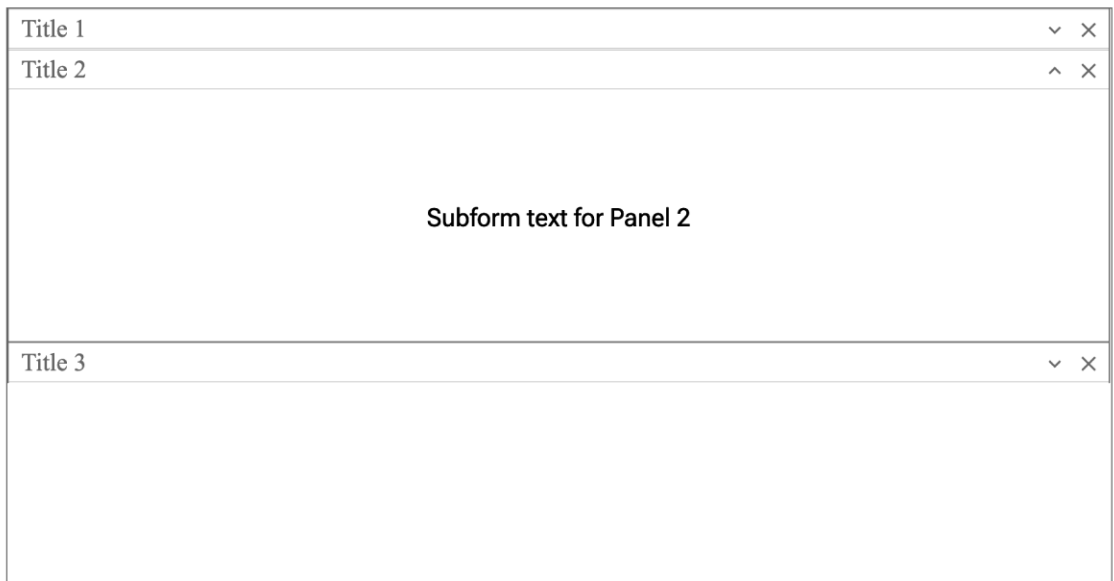
The following enhancements have been added to **Omnis Studio 11.2 Revision 38542**. See the Studio Now panel in the Studio Browser for information about faults fixed in this release.

JavaScript Remote Forms

Subformset Panel

The Subformset Panel is a new control that allows you to display a number of subforms in a vertical panel. (Revision 38468, ST/JS/3467)

The **Subformset Panel** control allows a set of subforms to be displayed as a group of expandible and collapsible panels within its border. Each panel consists of a title bar, with optional expand/collapse and close buttons, and an associated subform arranged vertically within the control.



The end user can expand and collapse each panel to view or hide the content of its subform by clicking on the minimize icon (drop arrow) or by double-clicking the title bar, and panels can be removed by clicking on the close icon. Individual panels can be dragged by the user to change their order on the screen.

Creating a Subformset Panel

To create a Subformset Panel and setup its panels, set the \$panelcount property under the Panel tab in the Property Manager to the number of panels you want to include in the control. To setup each panel, set the \$currentpanel property to each panel in turn and for each panel enter the name of a subform class (remote form) in the \$panelclassname property.

To ensure all the panels stretch to fit the width of the control, you can set \$parentwidth to kTrue; this overrides widths set for individual panels in \$panelwidth.

You need to set \$minbutton to kTrue to add a button to expand and collapse the content of each panel. If you want the end user to be able to remove subforms, you can add a close button to each panel by setting \$closebutton to kTrue.

General Properties

The Subformset Panel Control has some properties that are generic to all the panels, including:

Property	Description
\$closebuttoniconid	if selected, the default close button icon will be replaced with the selected icon
\$font	the font of the title text of each panel title bar
\$fontsize	the font size of the title text of each panel title bar. The size of the button icons in the title bar will be proportional to the size of the title text
\$minbuttoniconid	if selected, the default minimize / restore button icon will be replaced with the selected icon
\$restoreiconid	if selected, and the \$minbuttoniconid property is also selected, the minimize icon will be replaced by the restore icon when a panel is collapsed
\$textcolor	the text color for the title text of all the panels. This value can be overridden for individual panels by means of the \$paneltextcolor property in the Panels tab
\$titlebackcolor	the default background color for the title text of all the panels. This value can be overridden for individual panels by means of the \$paneltitlecolor property in the Panels tab
\$titlebarheight	overrides the default titlebar height of 24 pixels (0 in the Property Manager)
\$tooltipsactive	if true, any tooltips which have been defined for individual panels will be displayed when the panel titlebar is hovered

The following properties in the Action tab in the Property Manager determine the behavior of the panels:

Property	Description
\$closebutton	if true a close button is shown on the panel title. If the user clicks the close button the panel and subform are removed.
\$escctoclose	if true the panels in the set can be closed by pressing the Escape key.
\$minbutton	if true a minimize button is shown on the panel title. The user can flip the expanded / collapsed status of the panel by clicking on the button. Alternatively, the user can double-click on the titlebar to achieve the same functionality.
\$minbuttonistitle	if true the minimize button icon is removed and the whole title bar becomes the minimize button.
\$openmax	if true \$panelheight is ignored and open panels expand to the full height of the control. The title bars of any collapsed panels will not show until the open panel has been collapsed.
\$openmin	if true the subform panels in the set are initially opened in the minimized state.

Property	Description
\$parentwidth	if true the individual \$panelwidth for each panel is overridden, and its width is set to fill the width of the control.
\$preventdrag	if true the user will not be able to drag the panels to re-order them.
\$scrollable	if true allows subforms to scroll. Only affects non-responsive subforms as responsive subforms are scrollable by default.
\$singleopen	if true the panels are initially displayed with the first panel only expanded (\$openmin must be set to false).

Panel Properties

The 'Panel' tab in the Property Manager contains properties that apply to the panel specified in \$currentpanel. First set \$panelcount to specify the number of panels in the control.

Property	Description
\$currentpanel	panel specific properties are assigned to the current panel; set this in design mode to assign panel properties
\$movepanel	allows you to move a panel in design mode; the current panel moves to the position specified by the number entered and the panels after this are bumped down
\$panelclassname	the classname of the subform (remote form)
\$panelcount	the number of panels in the control
\$panelheight	the height of the subform panel when it is in the expanded state
\$panelid	an optional unique ID character string used to identify the panel
\$panelsubformparams	an optional comma-separated list of parameters that can be passed to a subform
\$paneltitle	the title text of the panel
\$paneltitletooltip	an optional character string displayed when the title bar is hovered. The \$tooltipsactive property must be set to true
\$panelwidth	the width of the subform panel if \$parentwidth is set to false

Colors and Icons

The colors of the panel title text and panel title background are determined by the general \$textcolor and \$titlebackcolor properties. To assign colors to individual panels, the colors can be overridden by the specific color properties \$paneltextcolor and \$paneltitlecolor in the 'Panel' tab.

The default minimize and close button icons can be overridden by selecting an icon in the \$minbuttoniconid and \$closebuttoniconid properties.

Passing Parameters

The Subformset Panel Control has the facility to pass a set of parameters to its subforms. The \$panelsubformparams property is a comma-separated list of parameters that can be passed to each subform.

Events

The Subformset Panel control reports the following events. The **evPanelOpened** event reports the panel that has been expanded from the collapsed state, while the **evPanelCollapsed** event reports the panel that has been collapsed from the open

state. The **evPanelRemoved** event reports the panel that has been removed from the set.

For each of these events the **pPanelID** parameter returns the unique ID of the panel, and the **pPanelPosition** parameter returns the position of the panel in the set.

Methods

The Subformset Panel control has the following methods:

- ❑ **\$addpanel**(cPanelID, cPaneltitle, cPanelclassname, cPanelsubformparams, iPaneltextcolor, iPaneltitlecolor)
adds a new panel at the end of the subform set.
- ❑ **\$removepanel**(cPanelID)
removes the panel denoted by cPanelID.
- ❑ **\$openpanel**(cPanelID)
opens the panel denoted by cPanelID.
- ❑ **\$collapsepanel**(cPanelID)
collapses the panel denoted by cPanelID.

The methods that require cPanelID use the IDs assigned to each panel in \$panelid.

In addition, the Subformset Panel control has the \$showpanel() method which can be used when the control is enabled as a Side Panel (\$sidepanel is kTrue).

Construct Row Variable

A new column named **cookies** has been added to the construct row variable that contains the cookies available in the client browser. (Revision 38472, ST/TC/043)

The \$construct row parameter in a remote task now has an additional column named cookies, containing a row with each column matching a name-value pair for each cookie terminated by a semi-colon. For example, cookie1=value; cookie2=value;

HTTP Server

The HTTP Server built into Omnis now supports IPv6. (Revision 38474, ST/WT/1881)

The HTTP Server in Omnis Studio allows you to test remote forms in your web and mobile applications. The HTTP Server on macOS already supports IPv6, but IPv6 support has been added for Omnis running on Windows and Linux.

There is a new item "IPv" in the 'server' group in the Omnis Configuration file (config.json). The values can be 4 for IPv4, 6 for IPv6, and 64 for a dual-stack IPv6 socket, that is, an IPv6 socket which has support for IPv4 on the same socket. By default, IPv4 will be used.

Application Setup & Deployment

macOS Tree Restructure

On macOS the application tree has been restructured to adhere to the Apple best practice guidelines. This will improve verification speeds when Omnis first starts up. (Revision 38518, ST/IN/281)

As a consequence of the macOS tree restructuring, you should be aware that many files have changed location which may have an effect on the functioning of your application or the deployment process you use. You should read the following sections for details of the changes and make any necessary adjustments to your application or deployment setup.

Only the main core Mach-O binary executable is now inside the Contents/MacOS folder.

Any external component previously existing in the external, xcomp and jscomp folders is now placed in the **Contents/PlugIns** folder. An external component will be identified by its file suffix which will be one of u_external, u_xcomp or u_webdesign. The wesecure and weshared bundles are also now placed into PlugIns.

Third-party components should be placed in the Contents/PlugIns folder if part of the application bundle.

Components can still be placed in the legacy xcomp and jscomp folders within a user's application data folder as well as the PlugIns folder. Where there is a component with the same name in the PlugIns folder of the users application data this will be loaded instead of the duplicate in the legacy xcomp or jscomp folder.

The Contents/Helpers directory now contains any binary which supplements the use of Omnis Studio and is not directly loaded. This will contain binaries for nodejs (the node packages and other files remain in Resources), crashpad, Omnis load sharing and web server plug-ins.

All other non-binary files which were previously present in Contents/MacOS have been moved into Contents/Resources. The original custom directory names will persist, e.g. /Contents/Resources/firstruninstall

DAMs

The DAM components now locate their client libraries from a set of predetermined paths which are the recommended locations for installing or bundling third party libraries.

For more details see Tech note: [TNSQ0043 – Loading External macOS Libraries](#)

See the **Appendix** at the end of this document for specific changes in the online docs regarding the macOS tree restructuring.

Notarization

The **Notarize** option in the Deployment tool has been updated and now uses the notarytool command to notarize your application. (Revision 38542, ST/AD/326)

Older versions of the Deployment tool use the altool command, but this will no longer work since Apple has removed support for altool. You can however notarize the bundle built with the Deployment tool using the notarytool command as follows:

```
xcrun notarytool submit <path-to>/<app-name>.dmg --apple-id <developer-email>
--team-id <team-id> --password <app-specific-password> --verbose -wait
```

Omnis Environment

Studio Now Sample Libraries

There is a new 'Studio Now' filter in the **Samples** section in the Hub to show sample libraries that have been added to the latest Studio Now release. (Revision 38502, ST/BR/463)

Omnis Libraries

Recent Libraries

You can now remove a library from the **Recent Project Libraries** list in the Studio Browser by right-clicking on the item in the list and selecting **Remove from list**. (Revision 38463, ST/BR/460)

There is also a new option 'Add Samples to 'Recent Project Libraries' in the IDE Options in the Hub to control whether the sample libraries (available under the Samples option in the Hub) are added to the **Recent Project Libraries** list. This is disabled by default so sample libraries are not added to the recent library list.

Report Programming

Enterable Report Fields

Report entry fields have a new property **\$enterable** allowing the end user to enter and save text in a PDF report. (Revision 38452, ST/EC/1849)

You can allow the end user to enter and save text in a field in a PDF report by setting its **\$enterable** property to **kTrue**. When the end user opens the PDF report, the field is active and enterable and once text has been entered the PDF can be saved. Note this only applies to PDFs that have been generated using the Omnis PDF Device.

Note that if the enterable report field is empty when it is printed, and the **\$nolineifempty** property is set to **kTrue**, then the whole line including the enterable field will not be included in the report. In this case, it is advisable to set **\$nolineifempty** for enterable report fields to **kFalse** (which is the default).

The **PRObjectStruct** has also been updated with a new **mEnterable** member which external component developers could use to make their own enterable fields.

Window Components

Tab Strip

The scroll speed of a vertical Tab Strip has changed so by default, the control uses the number of tabs and the size of the control to determine the scroll speed, plus there are some new config items to control the scroll speed and behavior. (Revision 38500, ST/WO/2836)

You can control the scroll speed using the new **tabstripSpeedToScroll** item in the 'defaults' section of config.json. The default is 0 which uses the new scroll speed, or you can enter the number of milliseconds you would like the scroll animation to use; 600ms was the default in previous versions.

You can control whether the control scrolls when clicked using the new **tabstripScrollsOnClick** item in the 'defaults' section of config.json. When set to false (the default), the control scrolls as the pointer enters the scroll buttons. When set to true the scroll buttons must be clicked to scroll the control. In this case, the Tab Strip will scroll an average tab height per click and is not animated.

OBrowser Configuration

You can now add cefSwitches with values to the “obrowser” item in the Omnis configuration file, edited using the Configuration Editor. (Revision 38461, ST/EC/1895)

You can define cefSwitches with values in the config.json. For example, to enable web socket connections from any origin, add the following:

```
"cefSwitches": [
    "remote-allow-origins=* "
],
```

Omnis VCS

Update from VCS

The **Update from VCS** option is now available when you right-click on individual or selected classes and folders in the Studio Browser. (Revision 38480, ST/VC/829)

A new context menu option **Update from VCS** is available at class level in the Studio Browser. Note the existing Update from VCS hyperlink option checks the whole library. Therefore, if you only want to update the selected class(es) or folder(s), you can use the new context menu option.

Update Local Library

The **Update Local Library** option is now available in the VCS to update the local library relating to the selected project. (Revision 38498, ST/VC/817)

A new hyperlink and context menu option **Update Local Library** allows you to update the local library with latest changes from the current, selected project. It is the equivalent of the 'Update from VCS' option except that it pushes any updates to the relevant local library.

Functions

idletime()

There is a new function **idletime()** that returns the elapsed time in milliseconds since the keyboard or mouse was used. (Revision 38495, ST/FU/897)

Function group	Execute on client	Platform(s)
Date and Time	NO	All

Syntax

idletime()

Description

Returns the number of milliseconds elapsed since the mouse or keyboard was last used.

What's New in Omnis Studio 11.2 Revision 38349

The following enhancements have been added to **Omnis Studio 11.2 Revision 38349**. See the Studio Now panel in the Studio Browser for information about faults fixed in this release.

JavaScript Remote Forms

Remote Form Properties

The minimum value of **\$layoutminheight** has been reduced to 10 (the previous minimum value was 100). (Revision 38276, ST/LR/057)

The value range of **\$layoutminheight** is now 10 to 32000 inclusive. Note it is set to 0 by default (as in previous versions), which means the layout height is 2 pixels below the lower edge of the bottom-most component on the remote form.

Omnis Environment

Omnis Configuration

The **\$getConfigjson()** and **\$setConfigjson()** methods (**\$root.\$prefs**) have an additional optional boolean parameter **bBaseConfig**, to control whether the methods operate on the **userconfig.json** (the default) or the **config.json** configuration file. (Revision 38344, ST/FU/896)

The syntax for the updated methods is:

- ❑ **\$getConfigjson([bBaseConfig=kFalse])**
Returns **userconfig.json** as a row when **bBaseConfig** is set to **kFalse** or is omitted (or empty if **userconfig.json** could not be parsed). If **bBaseConfig** is **kTrue**, returns **config.json** instead.
- ❑ **\$setConfigjson(wConfigJson[, bBaseConfig=kFalse])**
Sets **userconfig.json** to the supplied row when **bBaseConfig** is set to **kFalse** or is omitted. If **bBaseConfig** is **kTrue**, sets **config.json** instead.

By default the **\$get** and **\$set** methods will operate on **userconfig.json** when **bBaseConfig** is set to **kFalse** or is omitted. You can operate on the base **config.json** by setting **bBaseConfig** to **kTrue**; however you should avoid editing **config.json** directly, so in general you should update **userconfig.json**.

Note that *userconfig.json overrides settings in config.json*, so if a setting is present in both **config.json** and **userconfig.json**, the **userconfig.json** value will be used.

SQL Query Builder

A **Query Builder Manager** has been added to the SQL Query Builder toolbar and the query time has been added to the Results tab. (Revision 38339, ST/SS/448)

The new Query Builder Manager lists the queries you have saved in the Query Builder and is accessed via a new button next to the list of queries in the Query Builder toolbar. In the Query Builder Manager, you can search for queries using the Find button, and you can amend a query's description.

When you Run a query, the time taken to run the query is now shown in the status bar of the Results window.

Omnis VCS

VCS Privileges

The Privileges window in the VCS has been updated and now allows you to set privileges for all the classes in a folder or for a whole project. (Revision 38290, ST/VC/818)

The **Privileges** window in the Omnis VCS now displays a tree of classes, including all classes within a folder. Classes or folders can be selected or deselected individually. There is also a new second pane which lists all the privileges that have been set for the current project. In addition, it is now possible to set privileges *at a project level*, with new hyperlink and context menu options to initiate this. Note that when these options are used, the tree only shows the project name, not the individual classes within it.

OW3 Worker Objects

OAUTH2 Worker Object

The pRow in the \$completed callback method for the OAUTH2 Worker Object has a new column called *state*, which contains the value you set up in the \$oauth2state property. (Revision 38315, ST/EC/1889)

The *state* column contains the value you set up in the \$oauth2state property. This helps you to identify which user the callback belongs to. For example, where multiple OAuth2 workers are used to authenticate multiple mobile devices, you could set a UUID in the \$oauth2state property and keep a list of username:uuid until the \$completed callback, where you can match up the token received.

Functions

rcedit

Some new static methods have been added to the **rcedit** external component to return various information about files (note rcredit is available on Windows only). (Revision 38316, ST/EC/1891)

- ❑ **\$getapplicationmanifest**(cFileName)
Returns the manifest from the exe/dll in cFileName.
- ❑ **\$getfileversion**(cFileName)
Returns the file version from the exe/dll in cFileName.
- ❑ **\$getproductversion**(cFileName)
Returns the product version from the exe/dll in cFileName.
- ❑ **\$getresourcestring**(cFileName, iKey)
Returns the resource string with key iKey from the exe/dll in cFileName. You need an integer to identify the string. These can be inspected with tools such as Resource Hacker.
- ❑ **\$getversionstring**(cFileName, cKey)
Returns the version string with key cKey from the exe/dll in cFileName. The key could be some of the standard Windows PE keys, e.g.:
Comments
CompanyName

FileDescription
 FileVersion
 InternalName
 LegalCopyright
 LegalTrademarks
 OriginalFilename
 PrivateBuild
 ProductName
 ProductVersion
 SpecialBuild

bool()

There is a new function **bool()** which converts a value to a Boolean. (Revision 38265, ST/FU/891)

Function group	Execute on client	Platform(s)
General	NO	All

Syntax

bool(*pValue*)

Description

Converts *pValue* to Boolean. In effect, it returns the ‘truthiness’ of a value.

Example

```
Do bool(#NULL)          ## would return kFalse

If kTrue|bool(#NULL)   ## Would resolve to true
...
End If

If bool(lObjRef)       ## Would resolve to true if lObjRef was set to an
instance, and false if unset
...
End If
```

bitclear()

The *lastBitNumber* parameter of the *bitclear()* function is now optional. If *lastBitNumber* is omitted, the value in *firstBitNumber* is used allowing you to clear a single bit. (Revision 38248, ST/FU/893)

Syntax

bitclear(*binary*,*firstbitnumber*[,*lastbitnumber=firstbitnumber*])

You should note that bit numbers are *zero-indexed* and bit 0 is the most significant bit.

replace() and replaceall()

The *replace()* and *replaceall()* functions now have an optional fourth parameter to control whether a case-sensitive or case-insensitive replace is performed. (Revision 38261, ST/FU/892)

Syntax

replace(*source-string*,*target-string*,*replacement-string*[,*case-sensitive=kTrue*])

replaceall(*source-string*,*target-string*,*replacement-string*[,*case-sensitive=kTrue*])

If *case-sensitive* is `kTrue` (the default), or is omitted, a case-sensitive replace is performed. If passed as `kFalse`, a case-insensitive replace is performed.

FileOps Workers

Tag parameter

The **FileOps Workers** (Copy, Delete, Move) now have an extra last parameter `vTag` added to their `$init` methods. This works exactly as the tag parameter for the JavaScript and Python Workers. (Revision 38270, ST/EC/1885)

If supplied, the `vTag` parameter is some data that is passed to `$completed` in the column `__tag` of the row parameter. This can be used for example to identify the caller when the worker object is shared by several instances.

Thread Lock

The lock on the main thread caused when cancelling a FileOps worker has been removed. (Revision 38285, ST/EC/1886)

The lock on the main thread when a FileOps worker is cancelled, that is, either directly through a call to `$cancel`, or through the FileOps worker variable going out of scope, has been removed. However, you should note that the FileOps worker will still execute in the background even when cancelled, as it did before in previous versions.

Appendix

The following changes have been made to the Omnis Online docs with regards to the macOS Tree Restructuring in Omnis Studio 11.2 Revision 38542. This may assist you in making any necessary adjustments to your application or deployment setup on macOS.

macOS Tree Restructure

Online Documentation changes

Creating Web & Mobile Apps

Chapter 2 - JavaScript Remote Forms


PDF Printing

Supporting Files

The **blue** text was replaced by the **red** text.

The PDF Device component is available for Windows and macOS and **is located in the 'xcomp' folder and is loaded automatically.**

is loaded automatically. It is located in the 'xcomp' folder on Windows and the 'PlugIns' folder on macOS.

Updated doc: 

Chapter 7 - Deploying your Web & Mobile Apps

Setting up the Omnis App Server

Server Logging

logcomp is the name of the logging component to use, that is, "logToFile" which **references the logtofile.dll component in the logcomp folder** of the Studio tree.

references the logtofile component located in the logcomp folder in the Windows and Linux Studio tree and the PlugIns folder in the macOS tree.

Updated doc: 


Omnis Programming

Chapter 10 - Report Programming

HTML Report Device

The Omnis Studio Print Manager API has been made public, allowing you to create your own custom printing devices as external components and place them [in the XCOMP folder](#).

[in the external component folder \(PlugIns on macOS and XCOMP on other platforms\)](#).

Updated doc: 


Chapter 17 - Deployment

Deployment Tool

macOS

The second screen allows you to specify the bundle's startup folder, [iconsets \(for the library\)](#), [xcomp and icons folders](#), as well as the option to pre-serialise the bundle or add a custom read/write directory.

[iconsets \(for the library\), PlugIns and icons folders](#),

Updated doc: 

The label on the 3rd entry field should be Additions to /PlugIns.


Code Signing Omnis

[An application can only be signed if its code portion remains unchanged. For the Omnis application, the code portion is located in the Omnis package, e.g.:](#)

[The Studio application will only retain a valid signature if the contents of the signed application package remain unchanged, i.e. altering this folder will break the signature,](#)

The code text in the following box should be replaced with this:

[Omnis\ Studio\ 11\ x64.app/Contents](#)

Updated doc: 

Firstruninstall and Application Support Folders

To do this, when Omnis starts up it will check for the existence of a folder [called 'firstruninstall' in the macOS folder](#) in the Omnis package.

[called 'firstruninstall' in the Resources folder](#)

Updated doc: 

Updating Components


Either type of component can be placed in:

`x64/PlugIns/` `~/Library/Application Support/ Omnis\Omnis\ Studio\ 11\`

Where there is a component with the same name in PlugIns this will be loaded instead of the duplicate in the legacy folder.


*Insert the **above** before this existing line:*

If the required folder does not exist it can be created by the user.

Updated doc: 


Patching a signed tree

Components can be patched without re-signing into the `xcomp` and `jscomp` folders of the user data location, e.g.:
`xcomp, jscomp and PlugIns folders`

Updated doc: 

Update Manifest Files

When Omnis starts, it reads the contents of the 'version' file in the root of its installation files, that is `'/Application/Omnis Studio 11.app/Contents/MacOS/version'` on macOS
which is located at `'/Application/Omnis Studio 11.app/Contents/Resources/version'` on macOS


Updated doc: 

Update on macOS

When Omnis starts it will read an integer deployment version number from a file called "version" in the Omnis application's macOS folder:
`Resources folder`

The code text in the following box should be replaced with this:


`/Applications/Omnis\ Studio\
11.2.app/Contents/Resources/version`

Updated doc: 

The updates are specified in a set of files which should be placed in a folder called “manifest” within the Omnis application's [macOS folder](#).
Resources folder

The code text in the following box should be replaced with this:

```
/Applications/Omnis\ Studio\  
11.0.1.app/Contents/Resources/manifest/23071
```

Updated doc:  (Scroll to view update)

Extending Omnis

Chapter 7 - OW3 Worker Objects JavaScript Worker Object

Npm is provided alongside Node.js in Omnis Studio. *Insert after:* **On Windows and Linux the nodejs folder is installed within clientserver\server. On macOS the nodejs folder is installed within Resources and the node executable is installed in Helpers.**

To launch npm you can run index.js inside the npm folder, e.g.

```
./node npm/index.js
```

On Windows and Linux to launch npm you can run index.js inside the npm folder, e.g.

```
./node npm/index.js
```

On macOS if inside the npm folder then use the relative path to the node binary, e.g.

```
../../Helpers/node npm/index.js
```

Updated doc: 

Omnis Studio External Components


Chapter 1 - Omnis External Components Creating your own External Components

Components in Omnis

Loading Components

On all platforms, external components are loaded from the relevant [sub-directory \(xcomps, jscomps and logcomps folders\)](#) of both the Omnis data folder and the Omnis program / Application folder.

[sub-directory \(PlugIns, xcomps, jscomps and logcomps folders\)](#) of

Updated doc: 

Getting Started with Generic

Testing the Generic Component

To use the component, place a copy in the [XCOMP folder of the Studio tree](#).

[XCOMP folder within the Studio tree on Windows and the PlugIns folder within the Studio tree on macOS.](#)

Updated doc:  (scroll to view update)

Debugging on macOS/Linux

Change the target modules command line for macOS:
target modules add /Applications/Omnis\ Studio\
11.app/Contents/MacOS/xcomp/myxcomp.u_xcomp/Contents/MacOS/myxcomp --
symfile /Users/user/Documents/

OmnisSDKBuild/DebugSymbols/xcomp/myxcomp.u_xcomp.dSYM
[/MacOS/PlugIns/myxcomp.u_xcomp/](#)

Updated doc:  (scroll further to view update)

Extending Generic

After building the generic2 component close Omnis if it is still running, and move the component into [your XCOMP folder](#).
[your XCOMP or PlugIns folder](#)

Updated doc:  (scroll to view update)

Chapter 14—DAM API Reference

Developer Guide

Building the DAM

Mac OSX

To debug, the DAM needs to be built into the [Omnis.app/Contents/MacOS/xcomp](#) folder.

[Omnis.app/Contents/MacOS/PlugIns](#)


Updated doc: 

If the component fails to load on starting Omnis, you can verify the integrity of the component by navigating to

[Omnis.app/Contents/MacOS/xcomp](#),


[Omnis.app/Contents/MacOS/PlugIns](#),

Replace the screenshot showing the Contents folder.

Updated doc: 

Note the resource files which should be copied into the component package during the build process. [Localizable.strings](#) and [xcomp.rsrc](#) are generated by the Omnis resource compiler from the [.RC](#) files. [xcomp.rsrc](#) in particular must be present in order for Omnis to recognise the package as an Omnis external component.

[Localizable.strings](#) is generated by the Omnis resource compiler from the [.RC](#) files. This file must be present for Omnis to recognise the package as an Omnis external component.

Updated doc:  (scroll to view update)

Omnis Resource Compiler Mac OSX

Replace the code text in the box:

`gXcomp: 1`
`COCOA VERSION 1.0CountResources('OCTY')`
`= 64`

with:

`gXcomp: 1`
`COCOA VERSION 2.0`


Updated doc:  (scroll further to view update)

Base classes

tqfDAMbaseObj

Remove these methods:

```
tqfDAMbaseObj::setEnvFile()
tqfDAMbaseObj::getEnvFile()
```


Updated doc: 

JavaScript Component SDK Tutorial

Building Generic C++ Component Mac

You then need to copy the built component (jsgeneric.u_webdesign) into the **jscomp** folder in the Omnis app package. into the **PlugIns** folder

For example, if you want to place the component directly into your **jscomp PlugIns** folder, you need to set the Installation Directory to **"/Applications/Omnis.app/Contents/MacOS/jscomp"**. **"/Applications/Omnis.app/Contents/MacOS/PlugIns"**.

Updated doc:  (scroll to Mac section)

Debugging Debugging a Component Mac

Omnis app. Set the **Debug** value to the full path to the **jscomp** folder in your

Omnis app. Set the **Debug** value to the full path to the **PlugIns** folder in your

Replace the screenshot.

Updated doc:  (scroll to view update)