ODC 2025

Omnis Developer Conference

omnis studio

Level Up Your Development:
New Features in Omnis Studio 11.2 that Raise the Bar

Andreas H. Pfeiffer

JavaScript Remote Forms

Construct Row Variable

The $construct row variable in a remote task now has an additional column named headers, containing a row with each column matching a name-value pair for each HTTP header received from the client.

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, for example cookie1=hello will result in a row with the column name 'cookie' and 'hello' as the value.

Copying Data to the Clipboard

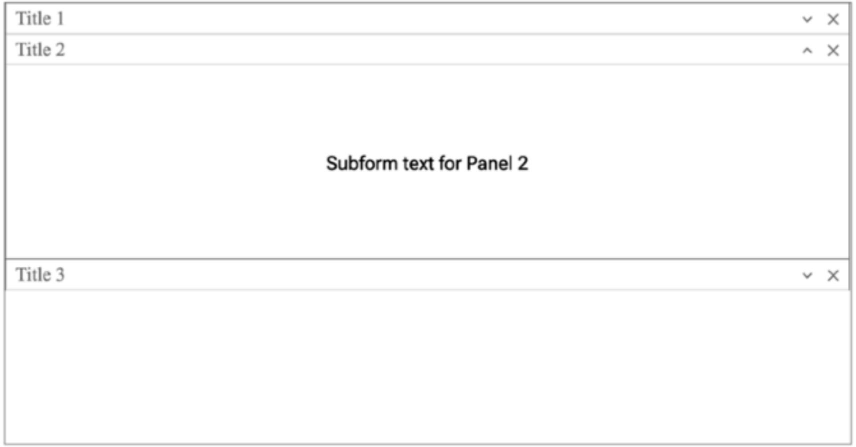
- $copytoclipboard(vDataToCopy[,cMIMEType='text/plain',bDataIsBinary=kFalse])

```
On evClick
   Do $cinst.$copytoclipboard(iText)

On evClick
   Do $cinst.$copytoclipboard(iText,"text/html")

On evClick
   Do $cinst.$copytoclipboard(iImageData,"image/png",kTrue)
```

Subformset Panels

Works similar like a subform control.

- $panelcount
- $currentpanel
- $panelclassname
- $closebutton
- $esctoclose
- $openmax
- $panelsubformparams



Customizing Keyboard Shortcuts

You can now intercept and modify keyboard input in remote forms or controls using custom event handlers.

$keyboardshortcut (class or field)

pSystemKey, pCharacterKey, pModifiers, pFocusedControl

```
If pCharacterKey="s"&lControlKey=kTrue
  Do $cinst.$save()
End If
```

Reading and Copying Data from the Clipboard

$readclipboard

remote form method allows you to read the last item of data on the end user’s clipboard, which in turn calls the

$onclipboardread remote form

method, passing it three parameters: pSuccess, pPlainText, pExtraData

pExtraData is a row containing any other MIME types associated with the data (for example

‘text/html’, ‘image/png’).

Return Methods

When calling a server-executed method from the client, a promise will be returned, unless there is a matching ..._return method, in which case it will call back to the return method when complete.

The return value from the server method will be passed through as a parameter in the Promise's "then" function. This allows you to chain your asynchronous logic in the same method, maintaining context and continue to use variables defined in the outer method. For example:

```
Do $cinst.$myServerMethod() Returns lPromise
JavaScript: lPromise.then( (returnVal) => {
JavaScript: lRetVal = returnVal; // Assign the parameter to a local variable that Omnis code knows about
Do $cinst.$showmessage(lRetVal,"Server Method Complete")
JavaScript: });
```

Push Notifications - openpush

The 'openpush' client command (executed using $clientcommand) has two new optional row() parameters iMaxTries and cRetryCodes.

- **iMaxTries**

The max number of times a request will automatically be tried, if it returns a http status code matched by cRetryCodes. -1 Means unlimited. Default is 5.

- **cRetryCodes**

HTTP status codes to treat as 'temporary errors'. These will be retried automatically, up to iMaxTries times before reporting an error. This is a string containing comma-separated codes and/or code ranges, e.g. "404,500-599" means any responses with http status code 404 or anywhere between 500 and 599 will automatically be retried. Default is 500-599.

Push Notifications - $pushclosed

There is a new client-executed method $pushclosed which can be implemented for remote forms and is called (on all forms) if a push connection closes.

Return kTrue from the method to prevent the default behavior.

$pushclosed has two parameters:

- **pErrorCode**

Non-zero values indicate it was closed due to an error.

0 means it was closed by calling the 'closepush' $clientcommand.

> 1 means pErrorCode is the http status code which caused the error. An error message will be displayed, unless you return kTrue to prevent the default behavior.

-1 means a transport error, i.e. the request could not be delivered.

- **pWillRestart**

If true, the default behaviour will be to restart the push connection. Return kTrue to prevent this.

Subform Palettes - Palette Arrow

When using the "subformpaletteshow" client command to open a subform palette, you can use the kSFSPaletteHideArrow flag in the iPositionFlags row parameter to hide the arrow normally shown on the palette window.

Subform Palettes - Control Name

If the control name given in the cControl parameter of the "subformpaletteshow" client command is not found (or you omit the name), the palette dialog will be centered in the

browser window, and the arrow is not displayed since there is no control for it to point at.

Subform Palettes - Cut out

A cutout has been added around the target for palette dialogs when overlay is shown.

If you pass the **bShowOverlay** parameter as kTrue to the "subformpaletteshow" client command ($clientcommand), then if it points at a control (or a sub-element of the control), a cutout will be made in the overlay around that element, to highlight it.

Date Parsing

The Date Parser in Omnis tries to convert a string representation of a date or date/time value to an Omnis Date/time format. Date string parsing has been improved and now falls back to Short date/time formats.

The Short date/time format depends on the browser locale, so for GB English it is D/M/y, and for US English it is M/D/y. Therefore, an end user could enter "3/5" into any Date or Date/time field, and it might be converted to "3 May 2024" or “March 5 2024”

Overriding the Browser History

New form methods let you interact with the browser's history, adding navigation events triggered by Back or Forward button clicks of the browser.

$pushhistorystate(wRow, [bReplaceState=kFalse])

$applynewhistory

receives the row wRow from $pushhistorystate as a parameter.

JavaScript Remote Components

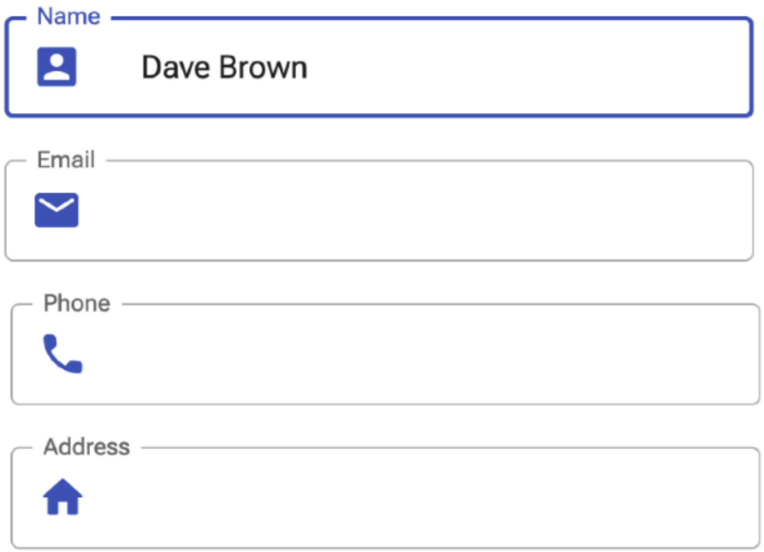
JS Entry Field Border Icons

$bordericonstyle

$setbordericonstyle

evBorderIconClicked

Field border icons can be added to Entry fields (as well as Combo boxes and Drop lists) using the **$bordericonstyle** property, which only applies when the field border style in $effect is set to kJSborderPlain.

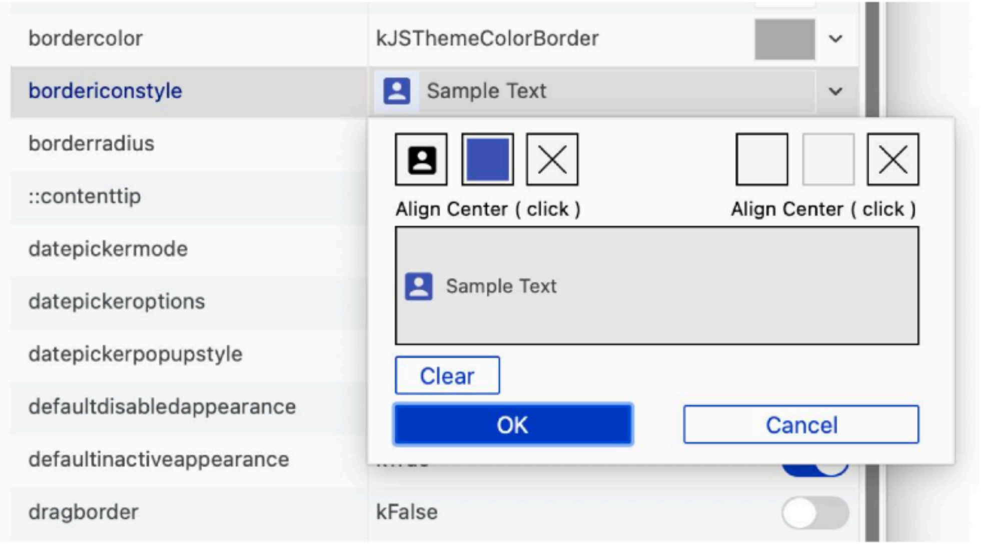


Name: Dave Brown

JS Entry Field Border Icons

The single property stores the settings for the left and right icons, which you can specify in the Property Manager.

You can enable the evBorderIconClicked event to make the border icons active. The event has the parameter pLeftBorderIcon. If true, the left border icon was clicked, or if false, the right border icon was clicked.



bordercolor	kJSThemeColorBorder
bordericonstyle	Sample Text
borderradius	
::contenttip	
datepickermode	
datepickeroptions	
datepickerpopupstyle	
defaultdisabledappearance	
defaultinactiveappearance	
dragborder	kFalse

JS Data Grid - $filteronkeypress

If **$filteronkeypress** is set to true, the grid is filtered on every keypress

(the default, to maintain behavior from previous versions).

If false, filtering takes place when the Enter key is pressed or when the focus leaves the field.

JS Data Grid - Grid Cell Tooltips

You can now display tooltips for individual cells in a Data Grid.

The **$gettooltip()** client-executed method is called when a grid cell is hovered. The returned string is displayed as the tooltip for the cell. The method has the following parameters pHorzCell, pVertCell, pDataColumnName to identify which cell has been hovered over.

JS Data Grid - Drop events

The evDrop, evCanDrop and evWillDrop events have a new parameter pColNum to report the column number for the dropped data.

The Data Grid control has a new parameter, pColNum, representing the column number of the cell in which the data will be dropped. The parameter is available for the evDrop, evCanDrop and evWillDrop events.

JS Data Grid - Pick Lists

The $picklistrowheight property has been added to the Data grid control to allow you to set the row height in pick lists.

You could set this to the same value as the row height for the data grid, e.g. 44 which is the default grid row height.

The pick list row height will not be less than the font height.

Datagrid - $enablecell()

The **$enablecell** method has been added to the Data Grid control to prevent specific cells from being edited.

The $enablecell(pRow,pCol,pDataColumnName) method is called before a cell is made editable, so return kFalse to disable editing for the cell.

This method can be client or server executed.

JS Data Grid - Date Picker

If either of the Data Grid properties $editdatetext or $columneditdatetext is true, the Date picker button is no longer treated as a separate tab stop. You can open the Date picker using Alt + Down arrow, or clicking on it.

If you want the old behavior, where the Date picker is a separate tab stop, you can set the JavaScript variable 'alwaysTabToDatePicker' to true on the data grid. For example:

```
Calculate lDataGrid as $cinst.$objs.datagrid
JavaScript:lDataGrid.alwaysTabToDatePicker = true;
```

Native List - $extrascrollheight

The **$extrascrollheight** property specifies the extra scroll height added to the end of a Native List.

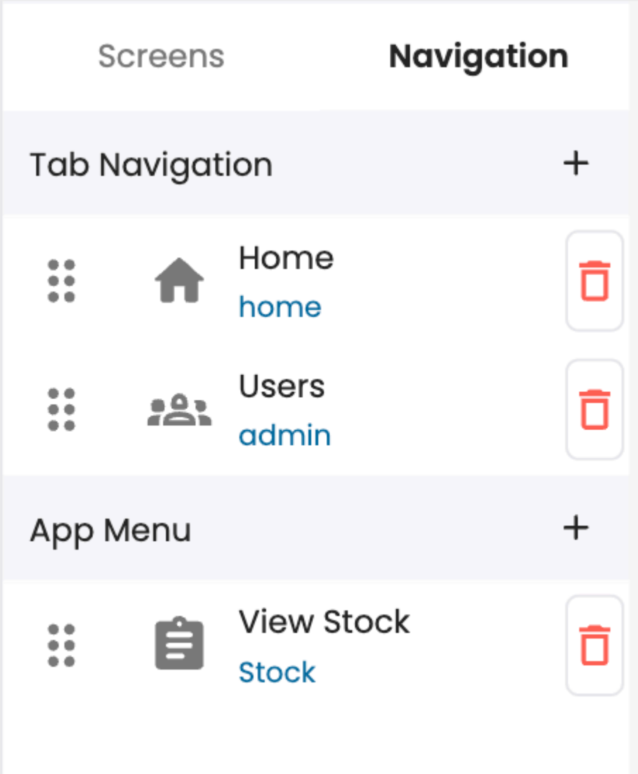
This could be useful to allow space for a Floating Action Button to be displayed over the top of a Native List, for example.

Native List - Buttons

You can now add buttons to the group headings in Native Lists, plus button accessories can now take an icon (or image) URL as their content.

You can set **$groupaccessorytypecol** to kJSNativeListAccessoryTypeButton to add a button to the group heading (no other accessory type is currently supported).

$groupaccessorycontentcol can be set to text or an icon/image URL, the image being detected by the URL ending with a .png/.jpg/.jpeg/.svg extension.



Native List - Buttons

When a group heading accessory is clicked, evClick will be fired.

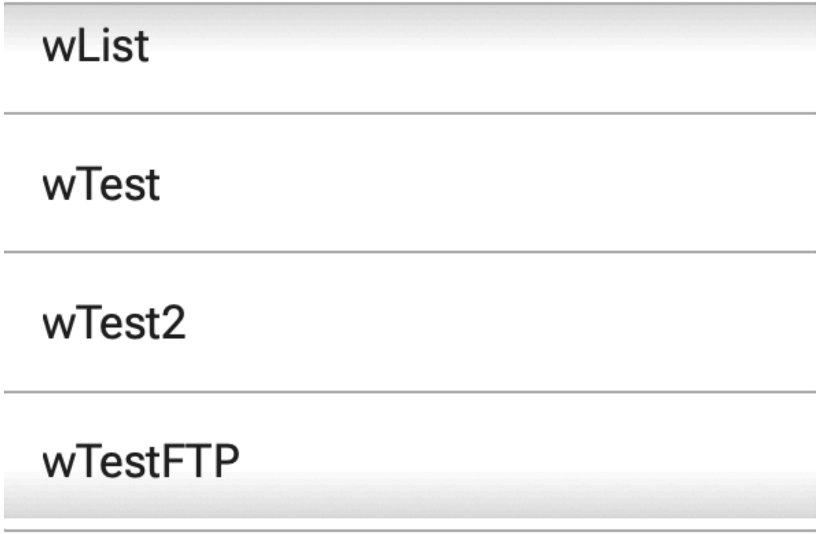
The **pWhat** parameter will be **kJSNativeListPartGroupAccessory**, and **pGroup** will be the group number.

Scroll Shadows for lists

Edge shadows now highlight scrollable content in lists, grids, containers, and other controls.

$scrollshadowwidth

$scrollshadowcolor



wList

wTest

wTest2

wTestFTP

Toolbar Control back button

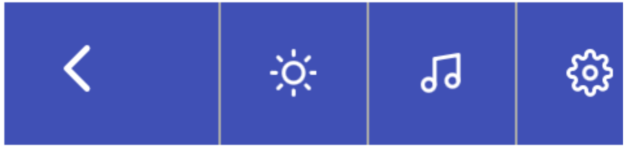
You can now add a 'back' button to the left side of a JS Toolbar control by setting

$backbutton to kTrue. In this case, if the side menu is visible ($sidemenu is kTrue), the hamburger menu is replaced by the back button. When the back button is visible and is clicked, the new evBackClick event is triggered.

$backbutton = kTrue

$event

On evBackClick



JS Tab Bar Control - $currenttab

The **$currenttab** property of the Tab Bar control can now be set to 0.

Which has the effect of deselecting all tabs.

Html Link Control - Link URL

If the $linkurl, $linkurlname, or $linkedobject properties are **empty**, the URL for the Html Link control defaults to the text from $text or $dataname, provided the text has a valid

URL format (http://... or https://...).

Events

The default event behaviour for evClick in the Html Link control can now be overridden in the $event method using the Quit event handler (Discard event) command.

Date Pickers

Date picker popups now use the client locale's default for the first day of week.

Date pickers for Entry fields and Data Grids (not the Date picker control) will now use the standard first day of the week from the client browser's locale when in the calendar view.

In previous versions the first day of the week defaulted to Sunday.

Navigation Menu Object

Two new properties **$::iconcolor** and **$cascadeiconcolor** have been added to the Navigation Menu Control to allow you set the color of the SVG icons in the control. This enhancement applies to the JavaScript (Remote form) and Window class Nav Menu controls.

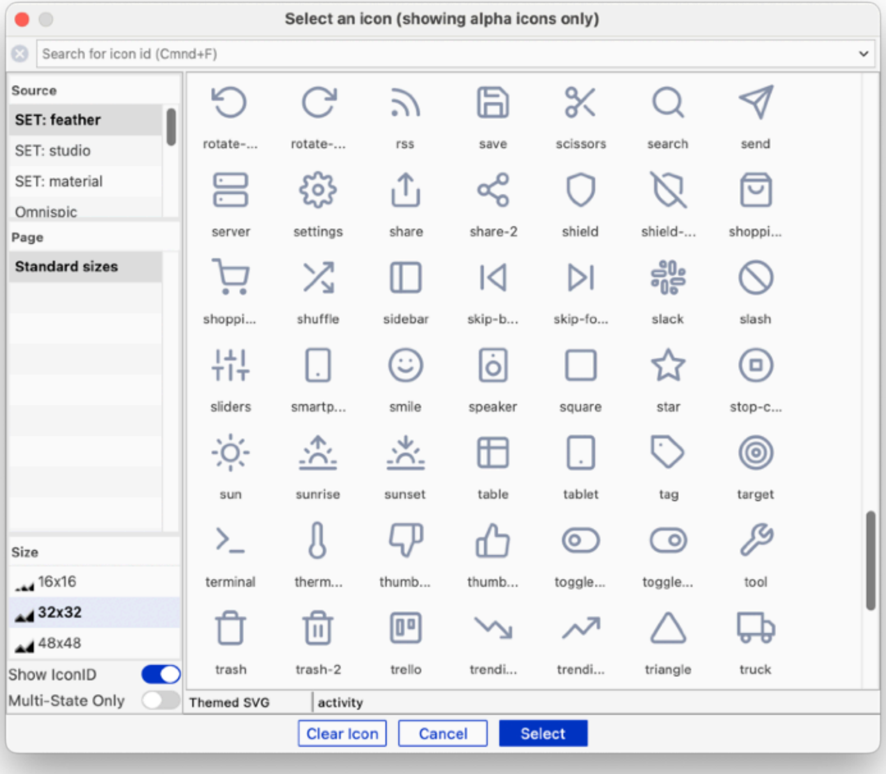
The **$::iconcolor** property specifies the color of the SVG icon on the top level menu specified in **$verticalcascadeiconid**.

The **$cascadeiconcolor** property specifies the color of the SVG icons on the cascaded menus, specified in **$horizontalcascadeiconid**, **$closeboxiconid** and **$hotcloseboxiconid**.

In the JS client component, kColorDefault conforms to the default text color rules, while in the Window class control kColorDefault causes the icon colors to follow the text color.

Component Icons new Feather Icon Set

Add "feather" to the **$iconset** property of your library.



Select an icon (showing alpha icons only)

Search for icon id (Cmnd+F)

Source: SET: feather, SET: studio, SET: material, Omnispic

Page: Standard sizes

Size: 16x16, 32x32, 48x48

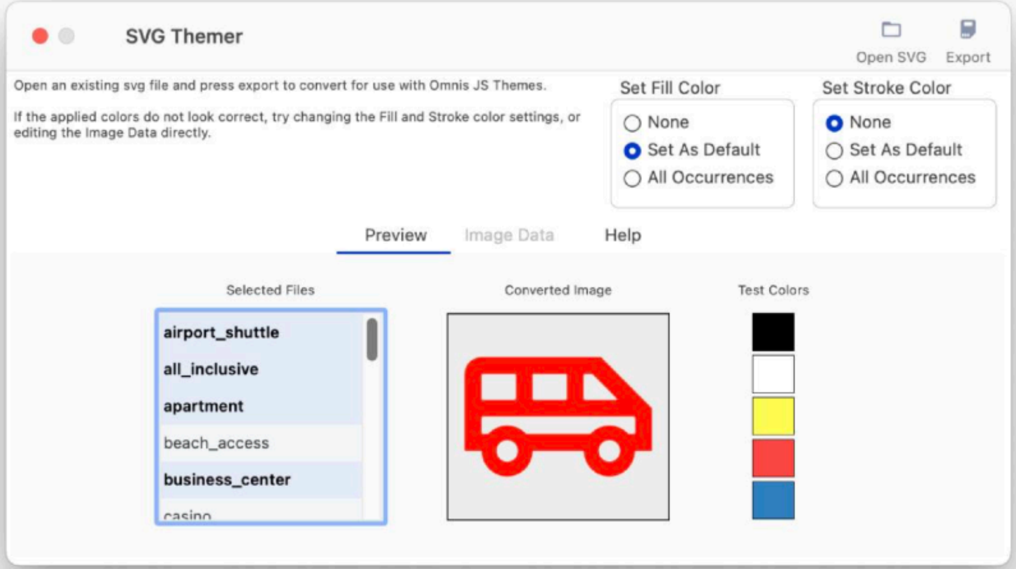
Show IconID: on

Multi-State Only: off

Themed SVG | activity

SVG Themer Tool batch theming

You can theme and export a batch of SVG icons using the SVG Themer tool.



Open an existing svg file and press export to convert for use with Omnis JS Themes.

If the applied colors do not look correct, try changing the Fill and Stroke color settings, or editing the Image Data directly.

Set Fill Color

○ None
● Set As Default
○ All Occurrences

Set Stroke Color

● None
○ Set As Default
○ All Occurrences

Selected Files

- airport_shuttle
- all_inclusive
- apartment
- beach_access
- business_center
- casino

Deploying Your Apps

Server Configuration - discardRequestsContaining

The **discardRequestsContaining** item has been added to the ‘server’ section of the Omnis Configuration file.

The **discardRequestsContaining** item is an array of strings, where each string will be matched against a requested URL by the Omnis server and if it matches, the request is discarded. By default, the array contains the string:

```
"discardRequestsContaining": [
"../"
]
```

This means that any requests received by the Omnis Server that contain ../ anywhere in the URL path will be discarded and an error returned to the client.

Firstruninstall

A hidden option has been added to disable the firstruninstall folder from copying over if the data folder already exists on Windows or macOS.

Omnis now checks for a file named "**disable_fri.txt**" in the root of the data folder, and if this file exists, Omnis does not copy files from the firstruninstall folder. Omnis does not read the file's contents, just checks if "**disable_fri.txt**" exists.

Window Components

Entry Fields and Combo Boxes

The properties **$labelcolor**, **$labelfontsize**, and **$labelhottextcolor** have been added to Entry fields and Combo boxes to control the text color and size of the floating content tip labels.

By default, these map to a font size of 0 and kColorDefault to maintain behaviour in previous versions where the color of the label text is the same as the field text and the font size is 80% of the edit field.

Masked Entry Field - $passwordchar

The **$passwordchar** property has been added to the Masked Entry Field.

You can use '*' in $passwordchar in a Masked Entry Field to replace characters with asterisks to hide the field content.

Pushbuttons

The style of pushbuttons set to kSystemButton in $buttonstyle has changed on macOS.

For headed lists and tree lists, $buttonstyle is the style of the header button, either kSystemButton or kUserButton. On macOS, the kSystemButton button style now uses the style of a native system pushbutton with a flexible height. If the height exceeds that of a regular size button the button keeps the same style.

In previous versions, the behavior was to use the square system style when the height was taller than a regular size button. To enable the legacy behavior, set the new $squaresystembuttons root preference to kTrue.

Headed List - $showcolumnslinesifempty

A new property **$showcolumnslinesifempty** has been added to Headed Lists to show column lines when the list is empty.

This only applies when $showcolumnlines is also kTrue.

oBrowser - $allowscrollzoom

The $allowscrollzoom property allows you to toggle the scroll zoom functionality within a browser window inside OBrowser.

When set to kTrue (the default), the end user can zoom in and out of the browser content by holding down the Ctrl key while scrolling the mouse wheel.

If $allowscrollzoom is set to false, this zoom functionality is disabled.

Border Icons

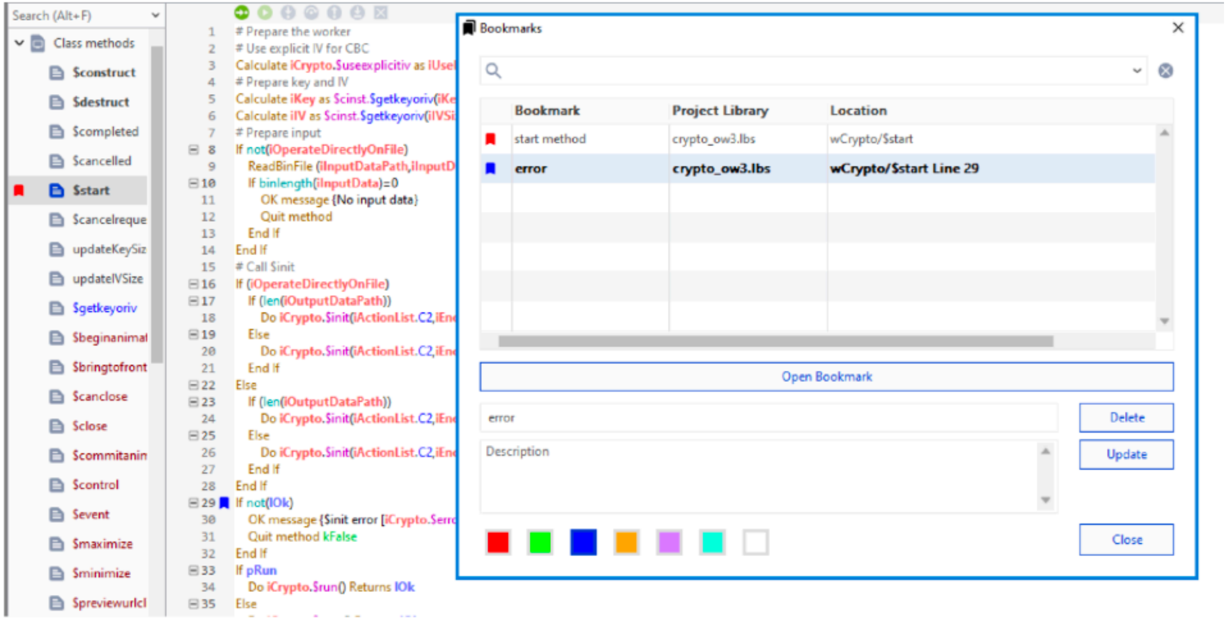
Border icons now respond to clicks. This applies to controls that support $bordericonstyle including Edit fields, Token edit fields, Combo boxes, and Drop lists

```
On evBorderIconClicked
  If pLeftBorderIcon
    Popup menu MyOptionsMenu
  Else
    Send to trace log Right Icon Clicked
  End If
```

Method Editor

Method Bookmarks

Add bookmarks to methods or lines to easily mark and find important code.



```
1  # Prepare the worker
2  # Use explicit IV for CBC
3  Calculate iCrypto.$useexplicitiv as iUsel
4  # Prepare key and IV
5  Calculate iKey as $cinst.$getkeyoriv(iKe
6  Calculate iIV as $cinst.$getkeyoriv(iIVSi
7  # Prepare input
8  If not(iOperateDirectlyOnFile)
9     ReadBinFile (iInputDataPath,iInputD
10    If binlength(iInputData)=0
11       OK message {No input data}
12       Quit method
13    End If
14 End If
15 # Call $init
16 If (iOperateDirectlyOnFile)
17    If (len(iOutputDataPath))
18       Do iCrypto.$init(iActionList.C2,iEn
19    Else
20       Do iCrypto.$init(iActionList.C2,iEn
21    End If
22 Else
23    If (len(iOutputDataPath))
24       Do iCrypto.$init(iActionList.C2,iEn
25    Else
26       Do iCrypto.$init(iActionList.C2,iEn
27    End If
28 End If
29 If not(lOk)
30    OK message {$init error [iCrypto.$erro
31    Quit method kFalse
32 End If
33 If pRun
34    Do iCrypto.$run() Returns lOk
35 Else
```

	Bookmark	Project Library	Location
🟥	start method	crypto_ow3.lbs	wCrypto/$start
🟦	**error**	**crypto_ow3.lbs**	**wCrypto/$start Line 29**

error

🟥 🟩 🟦 🟧 🟪 ▨ □

Libraries and Classes

Exporting Libraries to JSON

The following VCS-related properties are excluded from the JSON export when the

fullexportimport option (part of the $exportimportjsonoptions root preference) is false:

- Library properties:

$vcsbuilddate, $vcsbuildersname, $vcsbuildnotes

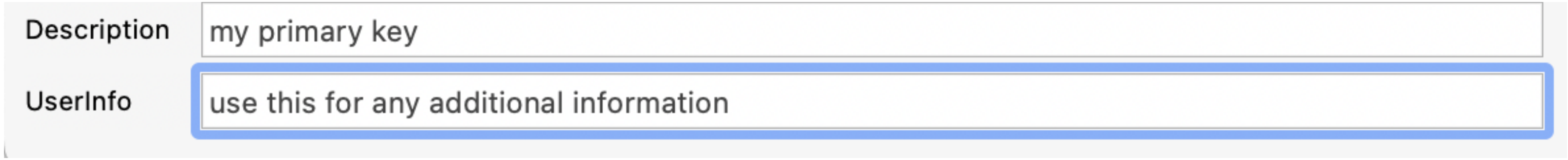
- Class properties:

$vcsrevision, $showascheckedout

File Classes

The file class editor now includes userinfo for each field in the file class which can be used to store data of any type for the field.

This maps to fileclass.field.$userinfo.



Description: my primary key

UserInfo: use this for any additional information

Reports

Enterable Report Fields

Set a field’s **$enterable** property to kTrue to let users enter and save text in a PDF report.

This works only with PDFs generated by the Omnis PDF Device.

$setdocinfo

The **$setdocinfo** method now allows you to set the producer and creator of the output PDF file.

This works only with PDFs generated by the Omnis PDF Device.

```
$setdocinfo(cAuthor[,cTitle='',cSubject='',cKeywords='',cLanguage='en-GB',cProducer='Omnis Studio PDF Device',cCreator='Omnis Studio'])
```

cProducer
is the software that generated the PDF, the default is Omnis Studio PDF Device.
cCreator
is the software that created the content of the PDF, the default is Omnis Studio.

$embedfile() and $embeddata()

Two new functions have been added to the Omnis PDF Device to allow you to embed files and data into PDF files.

This works only with PDFs generated by the Omnis PDF Device.

Omnis VCS API

VCS API - $getapiobject()

The way you access the VCS API has changed. You should now use the **$getapiobject** method to access the VCS API methods.

To call the VCS API methods you first need to use the **$getapiobject()** method to return an object reference to the VCS API, for example:
Do $root.$modes.$getapiobject('VCS') Returns iAPIObjRef

Once the object reference has been generated, all calls to the API can be made using this reference, allowing a simpler syntax for method calls.
For example, to call the $logon method, you can use:
Do iAPIObjRef.$logon(cUSERNAME, cPASSWORD, nTokenTime, cToken, cErrors, rSessionOrSessionPoolRef, cDBName) Returns bStatus

VCS API - $buildProject

$buildProject VCS API method now allows classes to be locked or excluded in the build.

The $buildProject method has two new list parameters, **excludeClassList** and **lockClassesList**.

Both are defined with **cClassName** and **cClassType** (e,g. wTest, kWindow) and contain a list of classes to either exclude from the build or to be built locked.

OW3 Worker Objects

LDAP Worker Object

The $init method in the LDAP worker can now accept the cUser and cPassword parameters as either Character or Binary; this enhancement is to accommodate names with umlauted characters.

When Character is used, Omnis sends cUser and cPassword as UTF-8. When Binary is used, Omnis sends cUser and cPassword using whatever encoding is required by the server, e.g. CP1252 is required by some servers.

HTTP Worker - Using AI in Omnis

You can access AI models via their APIs using the Omnis HTTP Worker.

An example application called HTTP AI has been added to the Samples section of the Hub in the Studio Browser to demonstrate how you can access various AI models.

HASH Worker - $initsignature

The $initsignature method has a new optional parameter iSignatureType to allow you to select the RSA or ECDSA signature type.

In addition, the bBlind parameter is now set to kTrue by default.

OAUTH2 Worker - $pkceentropy

The **$pkceentropy** property has been added to the OAUTH2 Worker to control the code verifier length for PKCE.

The **$pkceentropy** property controls the code verifier length for PKCE and takes an integer value between 43 (the default) and 128.

Functions

Searching Lists - bSetCurLineWhenNotSelecting

$search(calculation [,bFromStart=kTrue, bOnlySelected=kFalse, bSelectMatches=kTrue, bDeselectNonMatches=kTrue, **bSetCurLineWhenNotSelecting**=kTrue])

When **bSetCurLineWhenNotSelecting** is passed as **false** (the default is kTrue), and the select/deselect options are false, the search is completed, the found line number is returned but the found line is not selected.

FileOps

$copy**() and **$move()

- work for files and folders.
- include new parameters to create missing paths, replace existing files and merge content.

$joinpath(cSegment, ...)

- Joins all given segments together using the platform-specific separator as delimiter.

FileOps.$writefile - bCreateParentDirectories

A new parameter **bCreateParentDirectories** has been added to the FileOps.$writefile function to create the directories in the path.

```
FileOps.$writefile(cFilePath,vVariable [,iEnc=kUniTypeUTF8, bBOM=kTrue,
bReplace=kTrue, bCreateParentDirectories=kFalse])
```

bool()

There is a new function **bool()** which converts a value to a Boolean.

```
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
```

idletime()

There is a new function **idletime()** that returns the elapsed time in milliseconds since the keyboard or mouse was used.

coalesce() and coalesceempty()

coalesce()

- Returns the first non-NULL parameter from a list of values.

$coalesceempty()

- Returns the first non-NULL and non-Empty parameter from a list of values.

OREGEX

OREGEX.$replace(cSource, cTarget, cReplace [, iFlags, &cErrorText]))

- Searches for a match in cSource using pattern in cTarget and replaces the matched substring with cReplace.

OREGEX.$replaceall(cSource, cTarget, cReplace [, iFlags, &cErrorText]))

- Returns the first non-NULL and non-Empty parameter from a list of values.

The flags can be summed together, e.g. kORegExCaseInsensitive+kORegExMatchNotNull.

idletime()

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

JSON Object Methods

Some of the JSON Object methods have a new parameter to ensure missing objects are created automatically.

$setstring(), $setbool(), $setinteger(), $setfloat(), and $setobject().

syntax for $setstring() is now:

- $setstring(cMember,cString[,bAddMissingMembers=kFalse])

Sets specified member to JSON string with value cString. When

bAddMissingMembers is passed as kTrue, creates the missing members (objects

only), defaults to kFalse. Returns true for success.

rcedit

New static methods in the rcedit component (Windows only) return various file information.

- **$getapplicationmanifest**(cFileName)
  Returns the manifest from the exe/dll in cFileName.
- **$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.
- **$getversionstring**(cFileName, cKey)
  Returns the version string with key cKey from the exe/dll in cFileName.

There are many more small enhancements ..