APL64 2025.0.1 Update

Table of Contents

Overview

This document describes the enhancements and bug fixes that are present in the APL64 2025.0.1 Update.

Interpreter Modifications

Interpreter Performance Optimization (Pat: Please review)

- Internally, APL statements are "compiled" the first time they're executed and emitted as arrays of fundamental operations (xcodes). This compilation step is the reason an APL statement may run noticeably slower the first time it is executed. This update has eliminated this compilation step for numerous operators that were especially expensive to compile such as +.×, ^.=, +/, +\, ×/, ×\, ^/, ^\ etc. for most common datatype and shape combinations. Over 1000 such operators were precompiled (refer to the document APL64_Precompiled_Scex_Expressions.pdf for the complete list) by pre-compiling them, and including them in the APL64 installer. This pre-compilation may result in faster performance for some applications.
- This update optimized the way the main interpreter loop dispatches execution of each "xcode" operation that makes up the compiled form of an APL program. It also reduces the cost of pushing and popping SI levels as functions are called and return. The combination of these two optimizations may yield performance improvements for some applications.

Colossal File System: Significant performance improvements reading components

This is accomplished when a colossal file is tied exclusively (open mode = 0+16 or 2+16) in this update. Refer to the user documentation of the new open modes for ⎕cftie for additional information.

Colossal File System: New File Open Modes

⎕cfstie now supports file tying modes like the modes of ⎕xntie. This allows files to be explicitly opened in read-only mode, providing shared, read-only access by multiple users or processes. Refer to the user documentation of the new open modes for ⎕cftie for additional information.

XML Serialization Exception Message Format

If an XML serialization exception occurs, the formatting of a multi-line exception message is improved.

⎕XL: New Actions to Read and Write CSV-format Files

The ⎕XL SaveWorkSheetAsCsv and ToAplFromCsv actions have been implemented to read and write 'comma-separated' format files. An option is provided to specify the field separator.

⎕FnEdHist System Function

Function editing history can be captured as an APL nested array or an XML document using the new ⎕FnEdHist system function. Refer to the function editing change records features described below for additional information.

⎕NewLine and ⎕SetNewLines System Functions

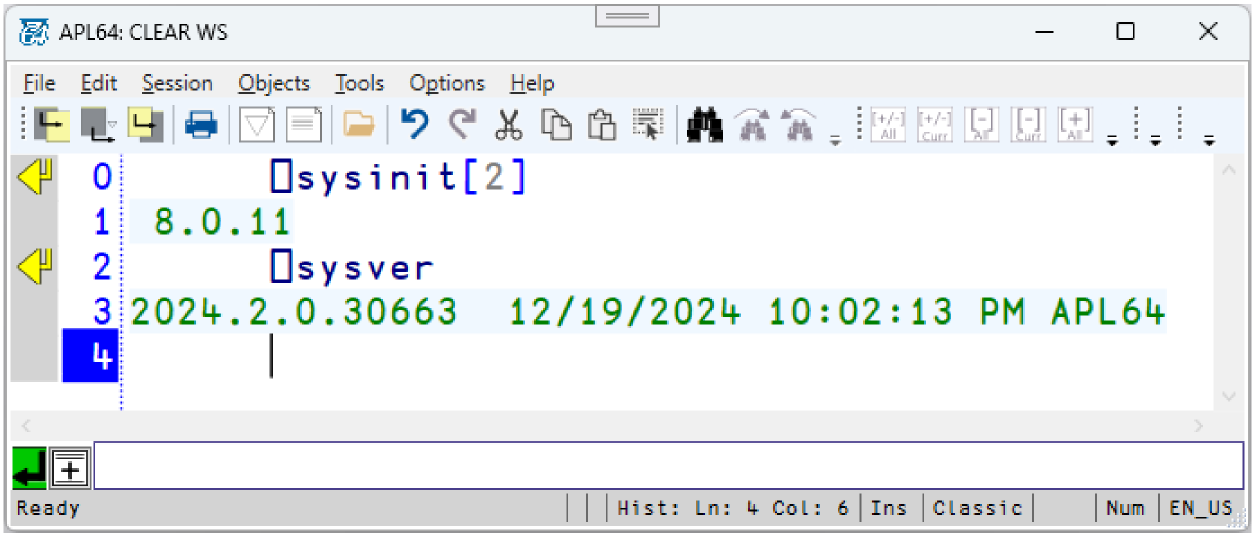
These new system functions are useful when an APL64 cross-platform component generates multi-line text, so that line termination character(s) which are appropriate for the operating system are used.

⎕LCASE and ⎕UCASE system functions updated to support Unicode characters

These system functions now support Unicode characters which are not included in ⎕av.

⎕SYSINIT[2] (2nd element) returns the .Net Version targeted by APL64 instance

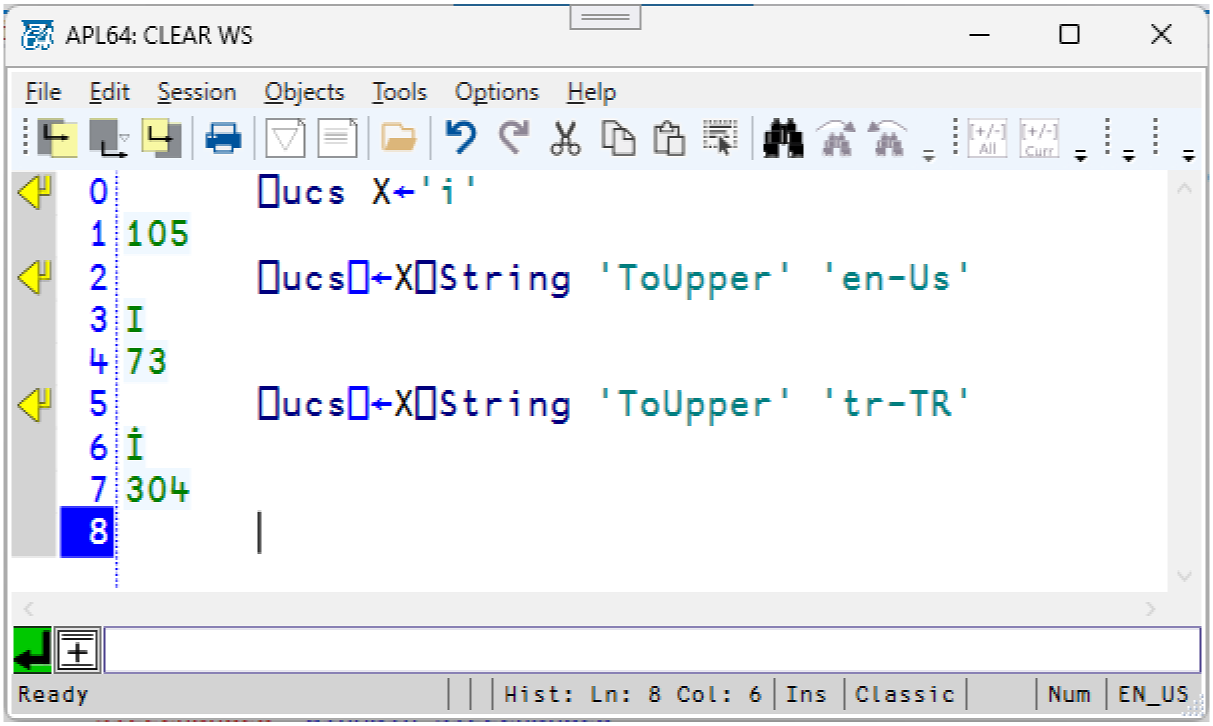
This information is useful when an APL64 cross-platform component is incorporated into a larger .Net application, or when using the ⎕cse system function. Refer to [Environment.Version Property (System) | Microsoft Learn] for additional information.



```
      ⎕sysinit[2]
 8.0.11
      ⎕sysver
2024.2.0.30663  12/19/2024 10:02:13 PM APL64
```

⎕STRING 'ToLower' and 'ToUpper' actions enhanced to support 'culture'

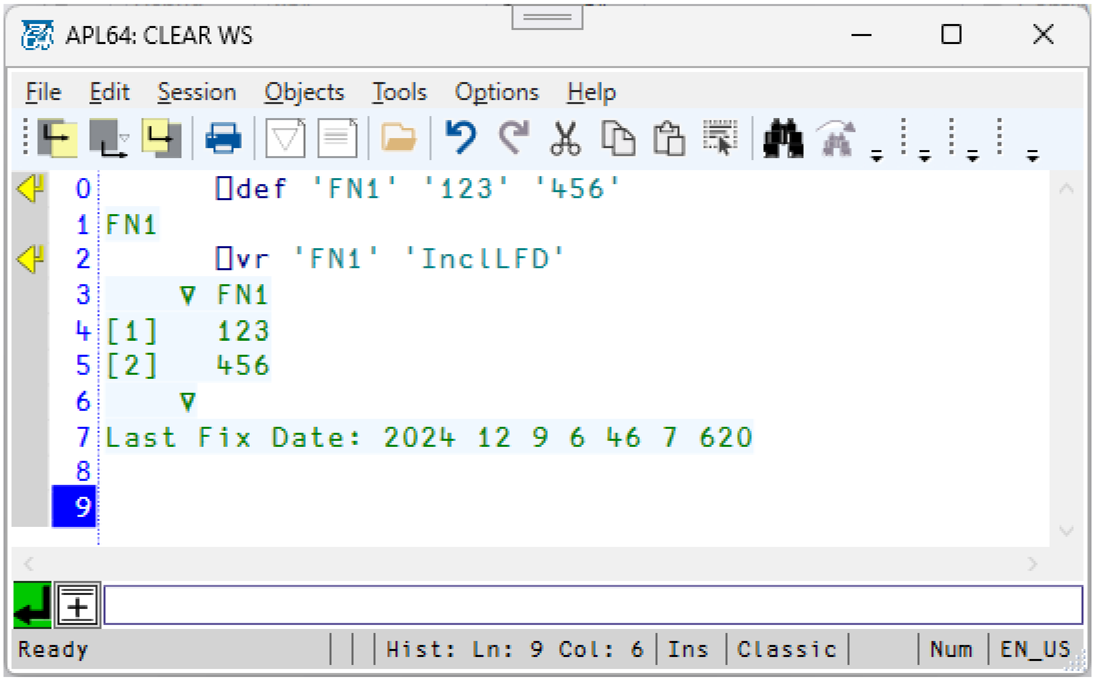
⎕string 'ToLower' and 'ToUpper' actions have been enhanced to provide an optional argument specifying the 'culture' to use for these actions. Use the new ⎕string 'GetCultureInfo' action to obtain the appropriate .Net culture specification, e.g. 'en-US', 'de-DE'. E.g.,


APL64: CLEAR WS
File Edit Session Objects Tools Options Help
Ready | Hist: Ln: 8 Col: 6 | Ins | Classic | Num | EN_US

```
      ⎕ucs X←'i'
105
      ⎕ucs⎕←X⎕String 'ToUpper' 'en-Us'
I
73
      ⎕ucs⎕←X⎕String 'ToUpper' 'tr-TR'
İ
304
```

⎕VR system function enhanced to specify the format of the result

⎕vr system function has been enhanced to provide an optional right argument to specify the format of the result. Without the optional right argument (⎕vr 'myFn') the traditional format is the result. The 'InclLFD' optional right argument (e.g., ⎕ vr 'myFn' 'InclLFD') includes the 'last fix date' timestamp) of the function in the result. The 'xml' and 'json' optional right arguments (e.g, ⎕ vr 'myFn' 'xml' or e.g, ⎕ vr 'myFn' 'json') include the function editing history in the result of ⎕ vr. The ⎕def system function accepts the three new formats of ⎕ vr 'myFn'.

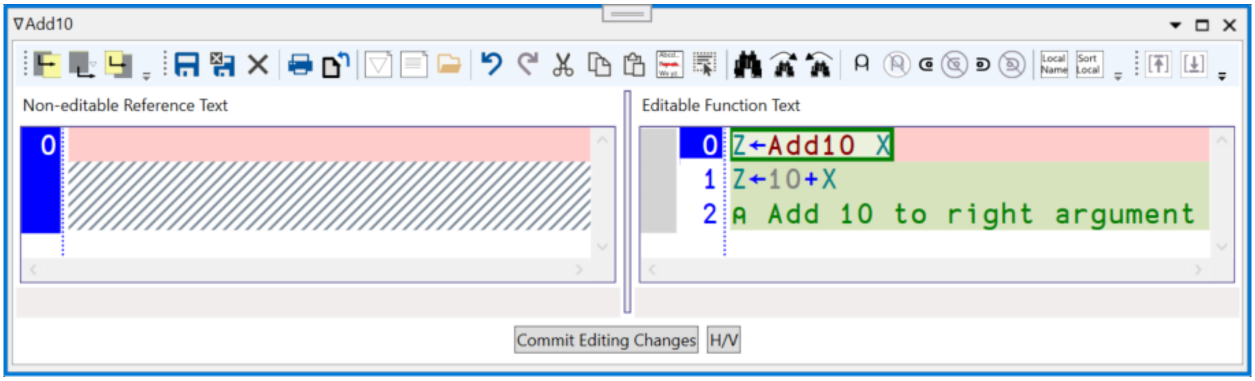

APL64: CLEAR WS
File Edit Session Objects Tools Options Help
Ready | Hist: Ln: 9 Col: 6 | Ins | Classic | Num | EN_US

```
      ⎕def 'FN1' '123' '456'
FN1
      ⎕vr 'FN1' 'InclLFD'
    ∇ FN1
[1]   123
[2]   456
    ∇
Last Fix Date: 2024 12 9 6 46 7 620
```

APL64 Developer Version GUI Modifications

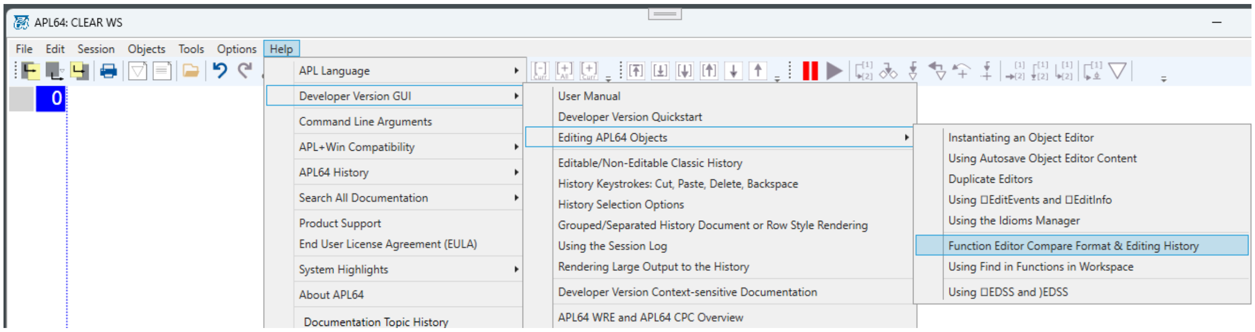
New Function Editor Compare Format

The function editor compare format illustrates a reference text (left) and the current edited text (right). Editing changes are identified graphically using background colors, and rectangles. The traditional format of the function editor has not been modified.



```
Z←Add10 X
Z←10+X
⍝ Add 10 to right argument
```

To learn more about the new function editor compare format access the user documentation here:


APL64: CLEAR WS
File
Edit
Session
Objects
Tools
Options
Help
APL Language
Developer Version GUI
Command Line Arguments
APL+Win Compatibility
APL64 History
Search All Documentation
Product Support
End User License Agreement (EULA)
System Highlights
About APL64
Documentation Topic History
User Manual
Developer Version Quickstart
Editing APL64 Objects
Editable/Non-Editable Classic History
History Keystrokes: Cut, Paste, Delete, Backspace
History Selection Options
Grouped/Separated History Document or Row Style Rendering
Using the Session Log
Rendering Large Output to the History
Developer Version Context-sensitive Documentation
APL64 WRE and APL64 CPC Overview
Instantiating an Object Editor
Using Autosave Object Editor Content
Duplicate Editors
Using ⎕EditEvents and ⎕EditInfo
Using the Idioms Manager
Function Editor Compare Format & Editing History
Using Find in Functions in Workspace
Using ⎕EDSS and)EDSS

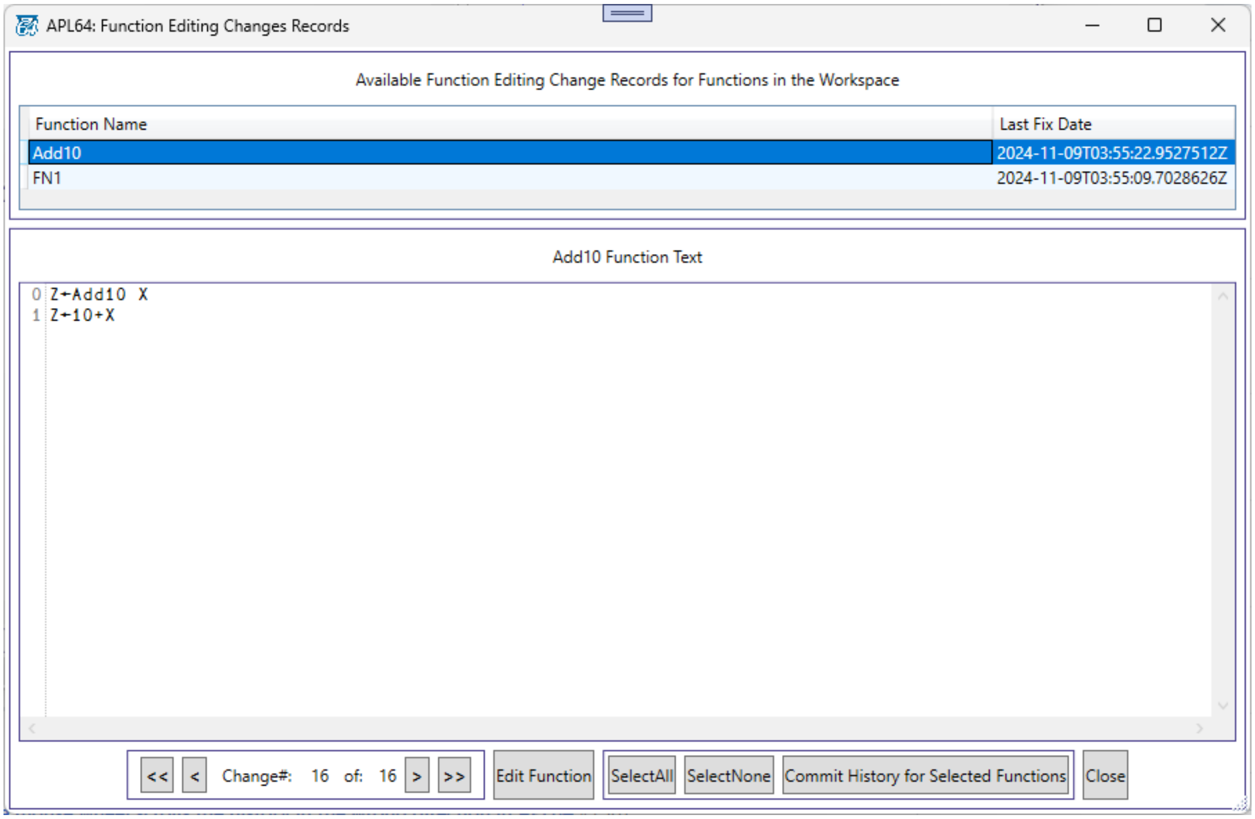
New Function Editing Changes Records

An option has been implemented to record function editing changes within the function definition so that they may be undone/redone in subsequent function editing instances, even in subsequent APL64 developer version instances. Select this option using the **Objects | Function Editor | Function Editing Change Records | Consider Function Editing Change Records** check box.

Function editing change records are graphically shown when the new function editor compare format is selected.

Function editing change records for user-defined functions in the workspace may be reviewed using the **Objects | Function Editor | Function Editing Change Records | Review Function Editing Change Record** dialog.

Refer to the **Help | Editing APL64 Objects | Function Editor Compare Format & Editing History** document for details.



APL64: Function Editing Changes Records

Available Function Editing Change Records for Functions in the Workspace

Function Name	Last Fix Date
Add10	2024-11-09T03:55:22.9527512Z
FN1	2024-11-09T03:55:09.7028626Z

Add10 Function Text

```
Z←Add10 X
Z←10+X
```

Change#: 16 of: 16

<<
<
>
>>
Edit Function
SelectAll
SelectNone
Commit History for Selected Functions
Close

Documentation of the Options | Keyboard Definition Dialog Updated

Refer to the **Help | Developer Version GUI | Keyboard Definition** document for details.

Find/Replace Tool Updates

1. The Find/Replace tool now reopens in the same location it was closed. This behavior applies between text panes in the same APL64 developer version instance, and between APL64 developer version instances
2. The Find action will now select the next found instance, if any, and put the keyboard focus on the searched text so that the user may conveniently enter a replacement text.

Set Initial Caret Marker Position when Editing a New Function

This is configurable in the **Objects | Function Editor | Caret At End of Header For New Function** menu item.

WRE and CPC Utility

APL64 provides two runtime options. The APL64 Windows Runtime Executable (WRE) option is analogous to the traditional APL+Win Windows desktop runtime. The APL64 Cross-platform Component (CPC) option creates a .Net assembly which can be incorporated into a larger .Net application that can be deployed to the Windows, Android, Linux and ios environments.

New APL64 WRE and Overview documentation

Read this user documentation to find out about the APL64 runtime options. Refer to **Help | Developer Version GUI | Cross-platform Component | APL64 WRE and APL64 CPC Overview** for more information.

Update to APL64 CPC in a REST Service Example (APL64CpcRestSvc.zip)

This example shows how to use an APL64 cross-platform component (CPC) to support the server side of a REST web service. See **Help | Developer Version GUI | Cross-platform Component | Cross-platform Component in a REST Service** for more information.

Update to APL64 CPC in WPF example targets .Net 8 Windows Presentation Foundation (WPF)

This example shows how to use an APL64 cross-platform component (CPC) to support calculations and algorithms exposed by an open-source, Windows Presentation Foundation (WPF) desktop GUI. Refer to **Help | Developer Version GUI | Cross-platform Component | Cross-platform Component in a WPF Application** for more information.

Update to APL64 CPC in Excel example targets .Net 8 Windows Presentation Foundation (WPF)

This example shows how expose an APL64 cross-platform component (CPC) as a custom Excel function. Refer to **Help | Developer Version GUI | Cross-platform Component | Cross-platform Component in Excel** for more information.

New APL64 CPC using .NET MAUI example

The APL64 CPC using .NET MAUI example, illustrates how APL64 programmer-defined functions can be included in a cross-platform GUI. Only one source code base is required to create a GUI for the Windows, Android and Apple operating system environments. Refer to **Help | Developer Version GUI | Cross-platform Component | Cross-platform and GUI using MAUI** for more information.

New APL64 CPC in an Azure Function example

The APL64 CPC in an Azure Function application example illustrates how to use APL64 to create a 'server-less' application on the Microsoft Azure cloud platform which does not require a ’24-7’ web server and accessible on-demand from a client browser. Refer to **Help | Developer Version GUI | Cross-platform Component | Cross-platform in Azure On-Deman Function** for more information.

Line Termination for Exception Messages

The line termination character(s) in a multi-line APL64 interpreter exception message will be determined by the host operating system environment.

Miscellaneous

APL64 system and installer updated to support .Net 8

APL64 has been updated to Microsoft .Net 8. The APL64 installer for the developer version requires that the target workstation have the appropriate Microsoft .Net 8 SDK and Windows desktop runtime installed.

:IFTEST control structure can be closed with either :ENDIF or :ENDIFTEST or :END:

The :ENDIFTEST and :END options are now documented in the Control Structure and User Manual documents.

⎕XL,)EDSS and ⎕EDSS: SpreadsheetGear Nuget package updated from v9.2.44 to 9.3.23.

⎕SQLite: Microsoft.Data.Sqlite client software Nuget package updated from v8.0.6 to v8.0.8

Pdf document generation component updated: ceTe v12.23.0 => 12.25.0. This component supports the display of user documentation and printing in the APL64 developer version.

Bug Fixes

SYSTEM ERROR:Exception when executing ⎕xl 'ToAPL' action with an invalid worksheet

An unanticipated exception occurred when the ⎕xl 'ToApl' action was used and the user-specified Excel worksheet in the target workbook did not exist.

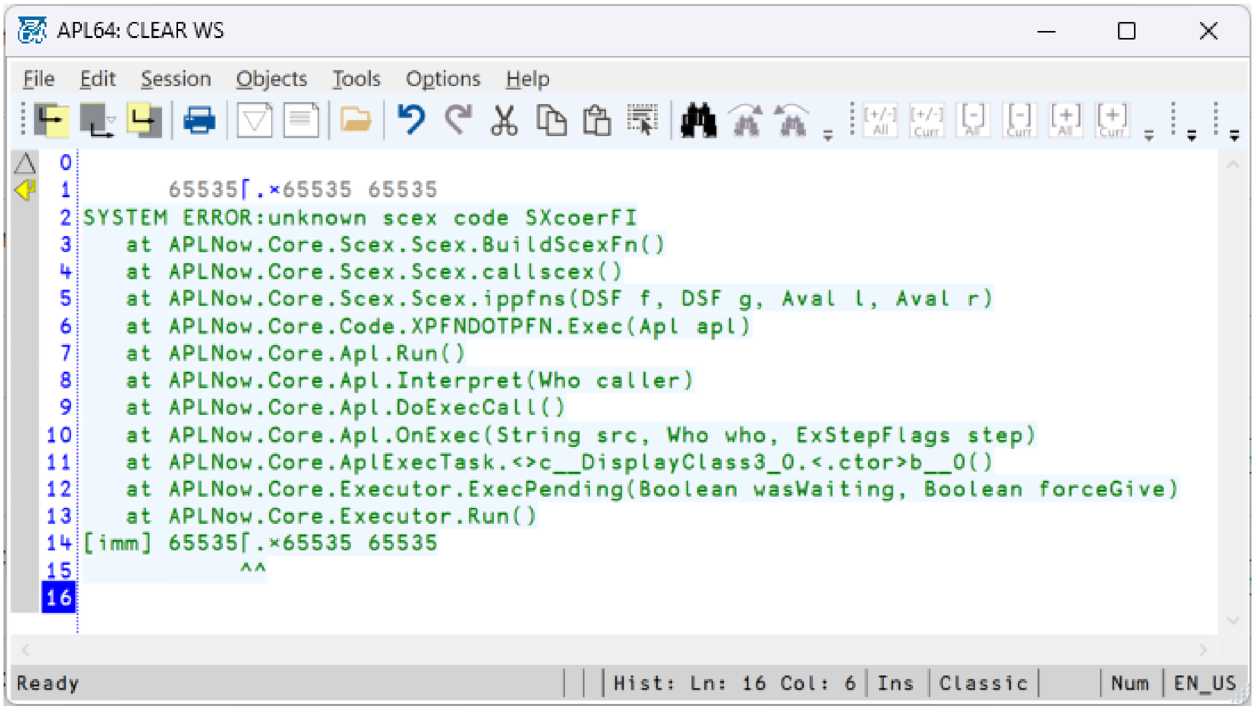
SYSTEM ERROR:Exception when executing ⎕xl 'SortSelectedRange' action on the same worksheet

An unhandled exception occurred when the ⎕XL 'SortSelectedRange' action was used multiple times on the same Excel worksheet and the 'lock' on the target Excel worksheet was not properly released.

Inner product function produced SYSTEM ERROR:unknown scex code SXcoerFI

The inner product function did not properly address when integer arguments overflowed to a double result type.

E.g.,

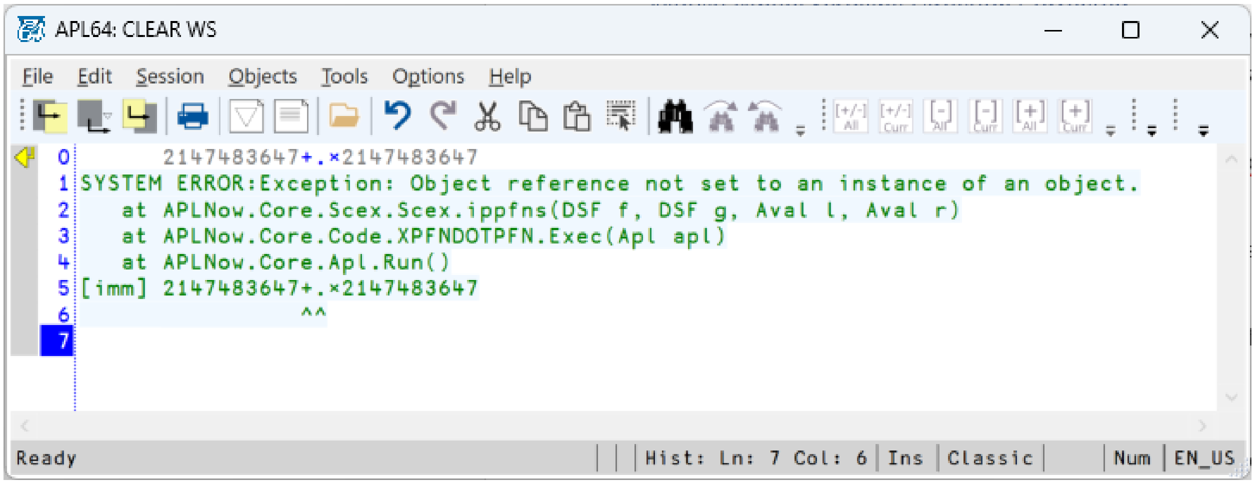


```
      65535⌈.×65535 65535
SYSTEM ERROR:unknown scex code SXcoerFI
   at APLNow.Core.Scex.Scex.BuildScexFn()
   at APLNow.Core.Scex.Scex.callscex()
   at APLNow.Core.Scex.Scex.ippfns(DSF f, DSF g, Aval l, Aval r)
   at APLNow.Core.Code.XPFNDOTPFN.Exec(Apl apl)
   at APLNow.Core.Apl.Run()
   at APLNow.Core.Apl.Interpret(Who caller)
   at APLNow.Core.Apl.DoExecCall()
   at APLNow.Core.Apl.OnExec(String src, Who who, ExStepFlags step)
   at APLNow.Core.AplExecTask.<>c__DisplayClass3_0.<.ctor>b__0()
   at APLNow.Core.Executor.ExecPending(Boolean wasWaiting, Boolean forceGive)
   at APLNow.Core.Executor.Run()
[imm] 65535⌈.×65535 65535
           ^^
```

Inner product function produced Null Reference Exception

The inner product function did not properly address the overflow state that resulted in a Null Reference Exception.

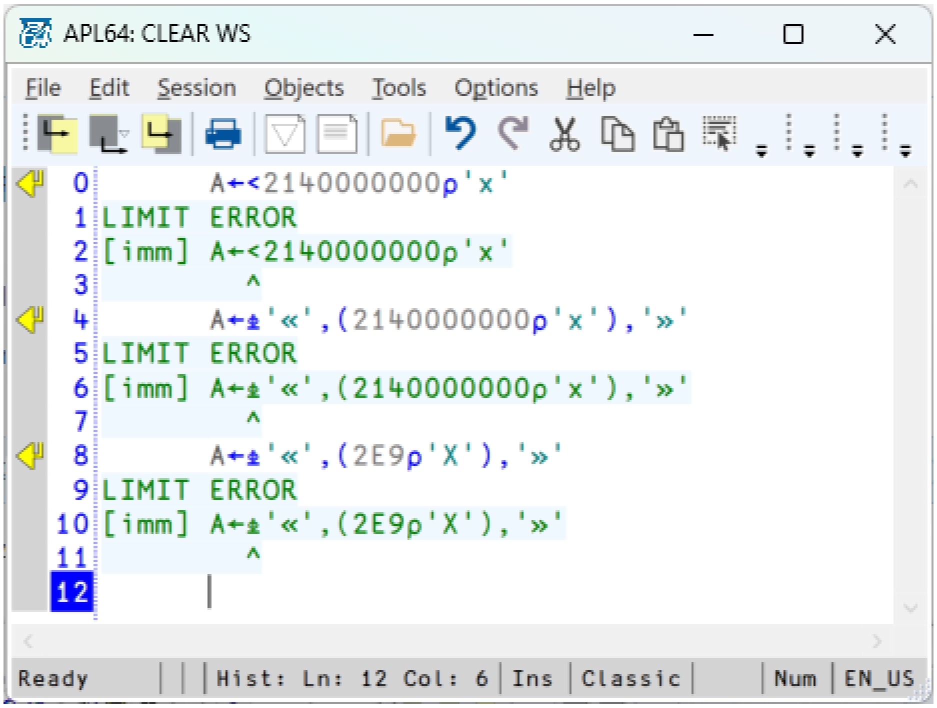
E.g.,



```
      2147483647+.×2147483647
SYSTEM ERROR:Exception: Object reference not set to an instance of an object.
   at APLNow.Core.Scex.Scex.ippfns(DSF f, DSF g, Aval l, Aval r)
   at APLNow.Core.Code.XPFNDOTPFN.Exec(Apl apl)
   at APLNow.Core.Apl.Run()
[imm] 2147483647+.×2147483647
                ^^
```

A LIMIT ERROR didn't result when a string was larger than the system defined maximum length

The examples below now correctly report a LIMIT ERROR:

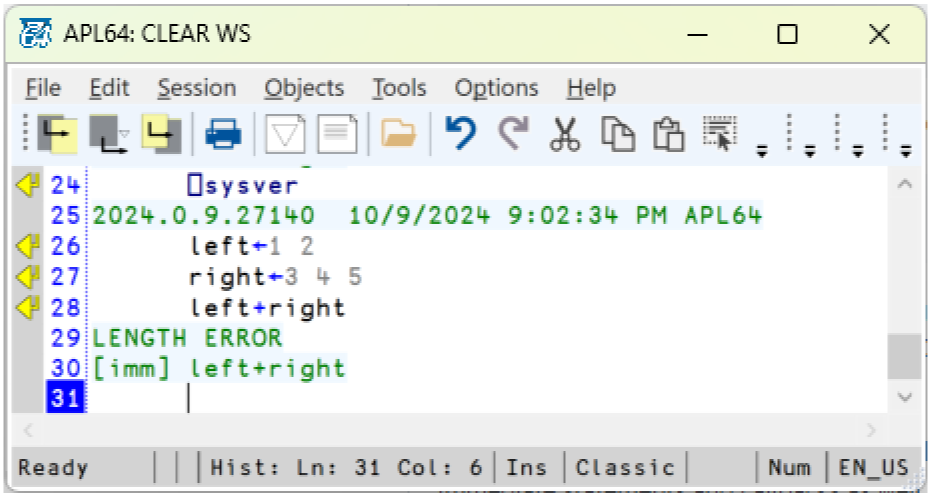


```
      A←<2140000000⍴'x'
LIMIT ERROR
[imm] A←<2140000000⍴'x'
          ^
      A←⍎'«',(2140000000⍴'x'),'»'
LIMIT ERROR
[imm] A←⍎'«',(2140000000⍴'x'),'»'
          ^
      A←⍎'«',(2E9⍴'X'),'»'
LIMIT ERROR
[imm] A←⍎'«',(2E9⍴'X'),'»'
          ^
```

Caret marker did not display for APL statements containing any variable or functions

A statement such as: left + right

where left is 1 2 and right is 3 4 5, displayed a LENGTH ERROR but not a caret under the plus function in the result, as expected. E.g.,



```
      ⎕sysver
2024.0.9.27140  10/9/2024 9:02:34 PM APL64
      left←1 2
      right←3 4 5
      left+right
LENGTH ERROR
[imm] left+right
```

Caret marker not displayed in the error message executing :CHOOSE

The caret marker did not display in the error message when executing :CHOOSE with an index out of range and assigning the result to a variable.

Mouse Wheel Scrolling Direction Corrected

The mouse wheel did not operate as expected when the Microsoft Windows option: **Mouse Properties | Wheel | Vertical Scrolling | One screen at a time** option is checked.

APL64 Interpreter Closed Unexpectedly

APL64 exited unexpectedly for the command: X←'abc',⎕tcnl,'pqrst' ⋄ ⎕edss 'X'. ⎕edss and)edss now handle certain non-printing character glyphs the same as Microsoft Excel.

Find/Replace Dialog Focus

The focus sometimes incorrectly appeared first in the replacement text field of the Replace dialog.

The 'Selection' Scope option in the Find/Replace tool to find within a selection corrected

- This option is now always visible in the Find/Replace dialog.
- When checked, this 'Selection' Scope applies only when there is a non-empty selection in the find/replace target.
- If the 'Selection' Scope option is checked, and no selection exists, the Scope will be the current text line.
- Any selection must be made after the 'Selection' Scope option is checked.

A single replace action in the Replace tool could skip a row

The Replace action has been corrected so that after a replace action, the next potential replace location, if any, is not skipped.

Changing Editors Pane Format Layout

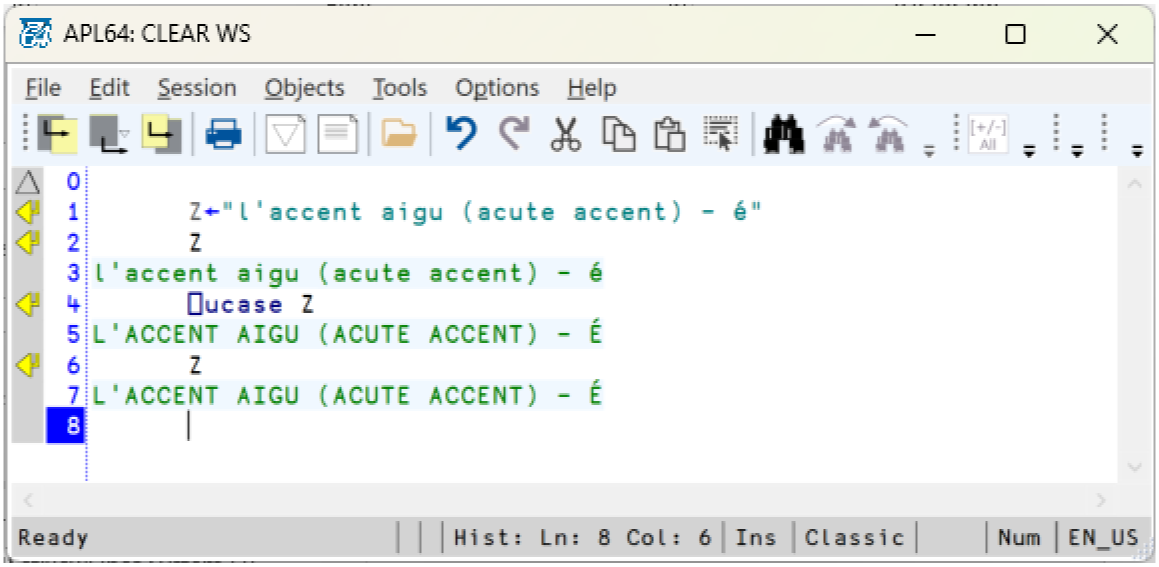
A blank area appeared to the left of the object editors after selecting the Editors Tiled Horizontally or Vertically option.

Duplicate)ed … In History

When selecting a function to edit in the Editor Autosave Options dialog, and closing the dialog, a duplicate ')ed …' line occurred in the history pane.

⎕ucase and ⎕lcase on a variable with Unicode dash and foreign accented characters changed the variable

E.g.,



```
      Z←"l'accent aigu (acute accent) - é"
      Z
l'accent aigu (acute accent) - é
      ⎕ucase Z
L'ACCENT AIGU (ACUTE ACCENT) - É
      Z
L'ACCENT AIGU (ACUTE ACCENT) - É
```

APL64 exited when the EDSS worksheet dialog was resized to its minimum size

⎕edss and)edss behavior when the dialog is resized is now the same as Microsoft Excel.