DYALOG

FinnAPL

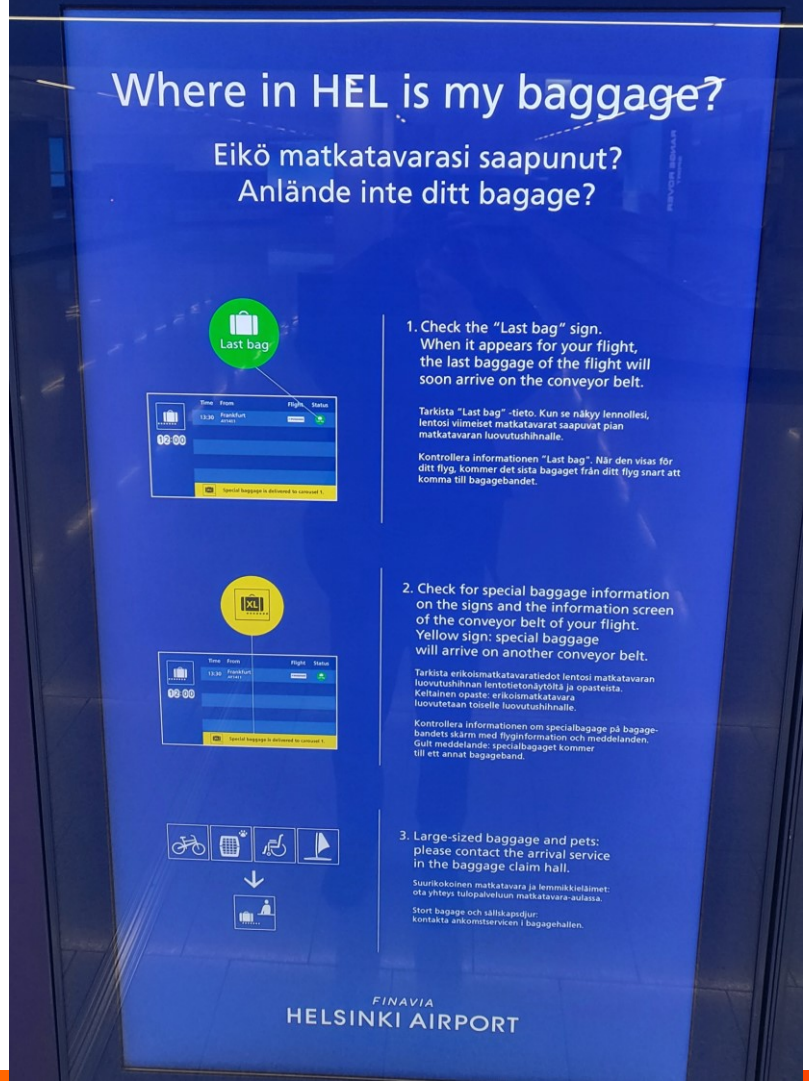
FinnAPL Spring Meeting, Solo Sokos Hotel Helsinki, May14th 2024

News From Dyalog

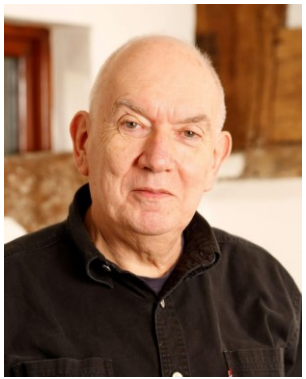
Morten Kromberg, CTO

I love Finnish Humour!

(my bags were on carousel #2)

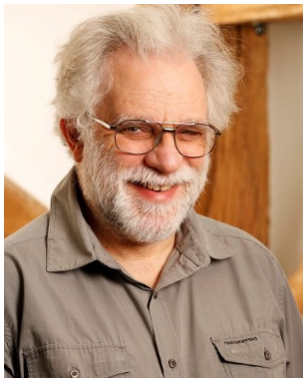


Changing of The Guard



John Scholes (1948-2019)

Roger Hui (1953-2021)



Geoff Streeter (retired 2023)

Pete Donnelly
(retiring – finally - 2024)

And of course...

Gitte Christensen

The CEO 2005-2023 has retired!

Under Gitte's Reign ...

- Headcount and Turnover ~~Quadrupled~~ Quintupled
  - Headcount now ~26 full time equivalents
  - Steadily increasing R&D budget
  - First generation of interpreter developers retired;
    New Team installed
  - Largest sustained investment in APL technology in history

Under Gitte's Reign ...

- Trading Profit in every year since 2005. Sources:
  - Growing partnerships with major customers
  - Increasing share of APL market
  - A small number of completely new APL systems
- Constantly working towards positioning APL as a modern tool for prototyping and rapid application development

Financial Status

Owners	2008	2018	Present
SimCorp (Denmark)	32.33%	40%	24%
APL Italiana (Italy)	32.33%		
Management & Employees	35.33%	60%	76%

- Ownership
  - 24% owned by SimCorp (which acquired APL Italiana, and was then itself acquired by Deutsche Börse)
  - 76% owned by management and employees
- Revenue steadily increasing:

5-year period	Growth
2017-2023	35%
2012-2017	8%
2007-2012	57%

Our 2nd Employee to Retire...

Gitte Christensen

The CEO has retired!

Kirstine Kromberg

Long Live the New CEO!

What about Morten?

- I have been working on a plan to make myself redundant since about 2020, and plan to complete the task by ~2030
- After that my plan is to write tools and documentation
- … like my predecessor

After Saturday's Bike Ride

Peter Donnelly

Dyalog – The Next Generation

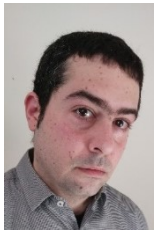
2010-2021





2022



2023





2024

2023 Summer Interns


Planned Hires in 2024

- Karl Holt (Aarhus): APL Consultant
- Brandon Wilson (Japan): Static Analysis Project
- Probably one more APL Consultant / Toolsmith
- Probably a JavaScript Developer (RIDE / VS Code Plugin + EWC)

Finding The Next Generation of Users

Aaron Hsu
Compiler
Research

Jesús López
Metallurgist
Uni Delft

Adam Brudzewsky
Here, There and
Everywhere

Rich Park
Media, Training
and "Outreach"

Stefan Kruger
Technical
Writer &
Pythonista

Mike Mingard
Web Page
Media Production
"Branding"

Introduction

Stefan Kruger

> A language that doesn't affect the way you think about programming is not worth knowing. –*Alan Perlis*

Who is this for?

I wrote this to be the book I would have wanted to read when I started to learn APL. An introduction to APL for an experienced practitioner from a different programming language or two. We all learn in different ways, and I prefer the fundamental concepts laid bare first, and then learn by example.

I came to APL after discovering a file of solutions to the Advent of Code 2015 challenge in K, an APL derivative. That's around 100 lines of actual code, and whilst I didn't understand any of it, I kept looking at it, trying to

Jesús Galan López

Google Scholar SIGN IN

Jesus Galan

TU Delft
Verified email at tudelft.nl - Homepage

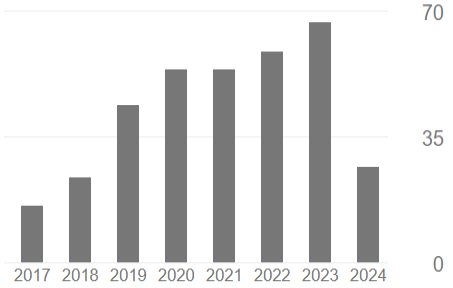
Materials Science Mechanical Engineering Computational Materials Sc…

GET MY OWN PROFILE

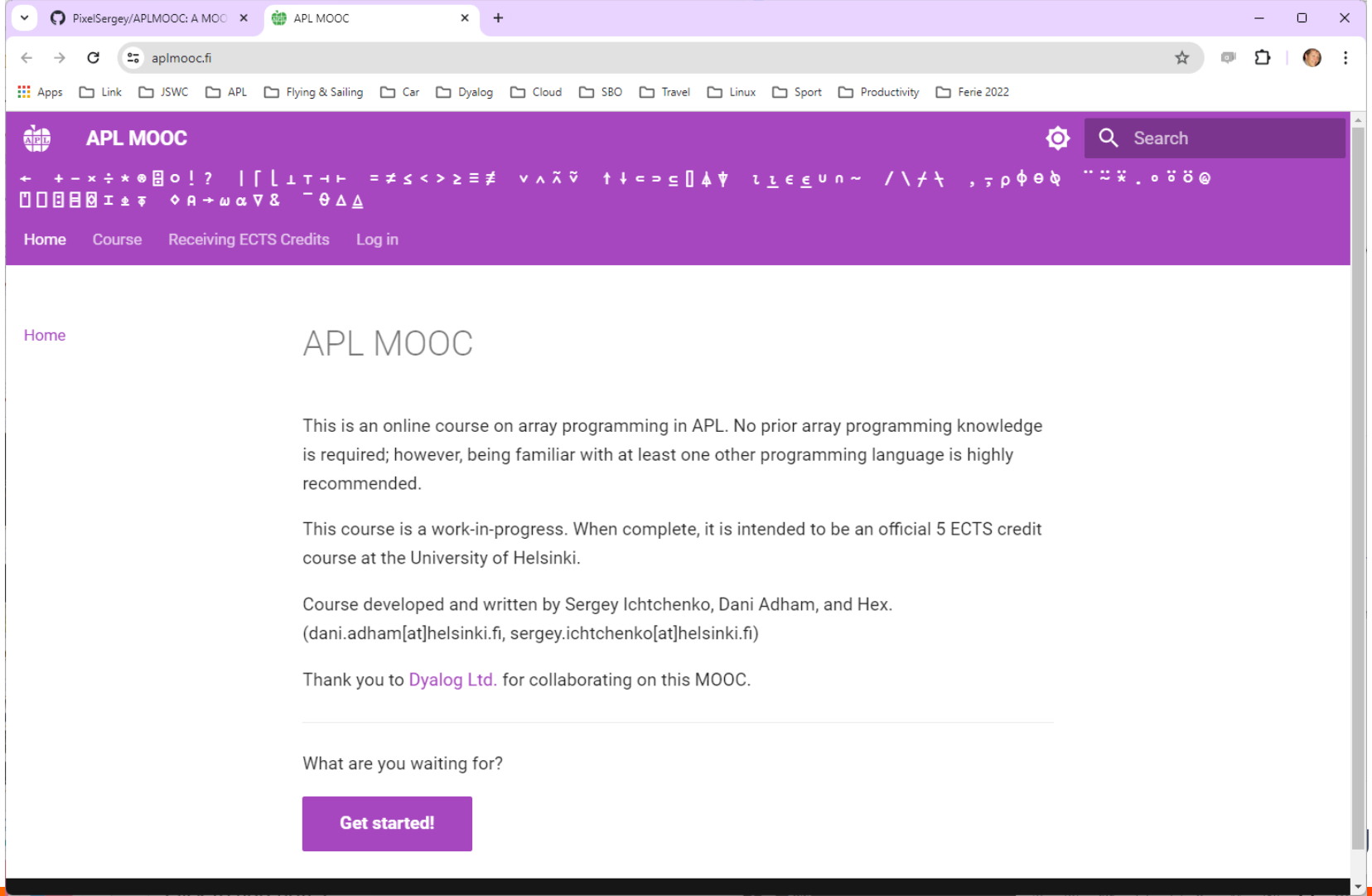
TITLE	CITED BY	YEAR
Advanced high strength steels for automotive industry J Galán, L Samek, P Verleysen, K Verbeken, Y Houbaert Revista de metalurgia 48 (2), 118	195	2012
Analysis of ESAFORM 2021 cup drawing benchmark of an Al alloy, critical factors for accuracy and efficiency of FE simulations AM Habraken, TA Aksen, JL Alves, RL Amaral, E Betaieb, N Chandola, ... International Journal of Material Forming 15 (5), 61	32	2022
Thermal effects during tensile deformation of Ti-6Al-4V at different strain rates J Galán, P Verleysen, J Degrieck Strain 49 (4), 354-365	26	2013
Effect of fatigue damage on static and dynamic tensile behaviour of electro-discharge machined Ti-6Al-4V JG Lopez, P Verleysen, J Degrieck Fatigue & Fracture of Engineering Materials & Structures 35 (12), 1120-1132	23	2012

Cited by VIEW ALL

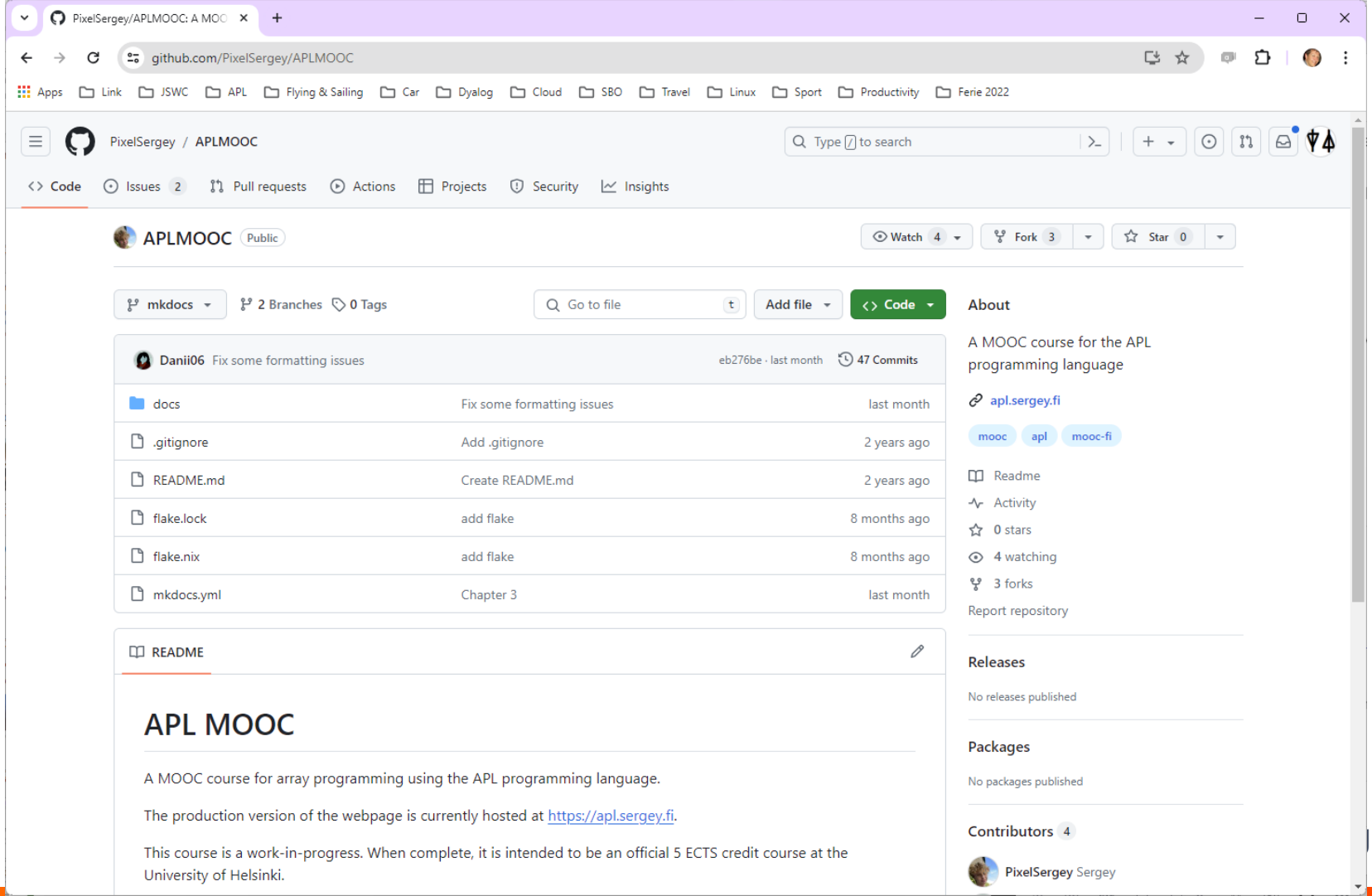
	All	Since 2019
Citations	421	306
h-index	10	9
i10-index	10	8



Public access VIEW ALL


PixelSergey/APLMOOC: A MOC
APL MOOC
aplmooc.fi
Apps
Link
JSWC
APL
Flying & Sailing
Car
Dyalog
Cloud
SBO
Travel
Linux
Sport
Productivity
Ferie 2022
APL MOOC
Search
Home
Course
Receiving ECTS Credits
Log in
Home
APL MOOC
This is an online course on array programming in APL. No prior array programming knowledge is required; however, being familiar with at least one other programming language is highly recommended.
This course is a work-in-progress. When complete, it is intended to be an official 5 ECTS credit course at the University of Helsinki.
Course developed and written by Sergey Ichtchenko, Dani Adham, and Hex. (dani.adham[at]helsinki.fi, sergey.ichtchenko[at]helsinki.fi)
Thank you to Dyalog Ltd. for collaborating on this MOOC.
What are you waiting for?
Get started!

'24


PixelSergey/APLMOOC: A MOO
github.com/PixelSergey/APLMOOC
Apps Link JSWC APL Flying & Sailing Car Dyalog Cloud SBO Travel Linux Sport Productivity Ferie 2022
PixelSergey / APLMOOC
Type / to search
Code Issues 2 Pull requests Actions Projects Security Insights
APLMOOC Public
Watch 4 Fork 3 Star 0
mkdocs 2 Branches 0 Tags
Go to file t
Add file
Code
Danii06 Fix some formatting issues
eb276be · last month
47 Commits
docs Fix some formatting issues last month
.gitignore Add .gitignore 2 years ago
README.md Create README.md 2 years ago
flake.lock add flake 8 months ago
flake.nix add flake 8 months ago
mkdocs.yml Chapter 3 last month
README
APL MOOC
A MOOC course for array programming using the APL programming language.
The production version of the webpage is currently hosted at https://apl.sergey.fi.
This course is a work-in-progress. When complete, it is intended to be an official 5 ECTS credit course at the University of Helsinki.
About
A MOOC course for the APL programming language
apl.sergey.fi
mooc apl mooc-fi
Readme
Activity
0 stars
4 watching
3 forks
Report repository
Releases
No releases published
Packages
No packages published
Contributors 4
PixelSergey Sergey

'24

The APL Forge - APL Forge

forge.dyalog.com

New Chrome available

Apps Link JSWC APL Flying & Sailing Car Dyalog Cloud SBO Travel Linux Sport Productivity Ferie 2022

About the APL Forge | Eligibility/Judging Criteria | Submissions | FAQs | Contact Us | Legal

SUBMIT YOUR PROJECT

Welcome to the APL Forge

Shaping Tomorrow with APL

Do you have a problem and are looking for the right language in which to solve it? Do you have an idea for a library that you think other APL users might benefit from? Do you have an APL-based application that you want to share? The APL Forge is where we reward you for using Dyalog APL to solve problems and develop libraries, applications, and tools.

MORE DETAILS

Mike Mingard

FinnAPL Spring '24

Speakers | LambdaConf

lambdaconf.us/speakers

Apps Link JSWC APL Flying & Sailing Car Dyalog Cloud SBO Travel Linux Sport Productivity Ferie 2022

Schedule Speakers Sponsorship Hackathon Workshops Buy tickets

LambdaConf speakers

Apply to be a speaker!

Aaron Hsu

Adam Fraser

Adam McCullough

Afsal Thaj

Ziverge

Speakers | LambdaConf

lambdaconf.us/speakers

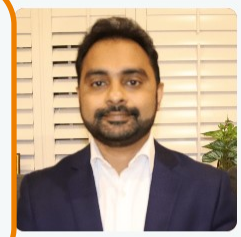
Apps Link JSWC APL Flying & Sailing Car Dyalog Cloud SBO Travel Linux Sport Productivity Ferie 2022

Joseph Guadagno

RT2

Josh David

Dyalog



Jubin Thomas

Signet Jewelers

Julia Belyakova

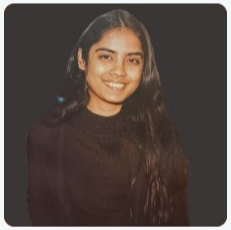
Purdue University

Liam Fitzgerald

Urbit Foundation

Luke Champine

The Sia Foundation

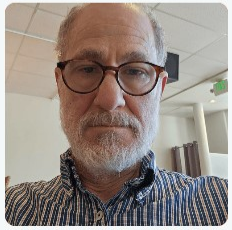


Madhu Patel

Adobe

Martin Förtsch

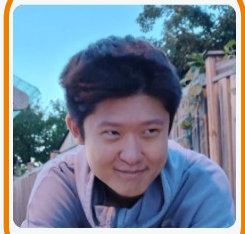
TNG Technology Consulting



Matthew Fuchs

Max Demoulin

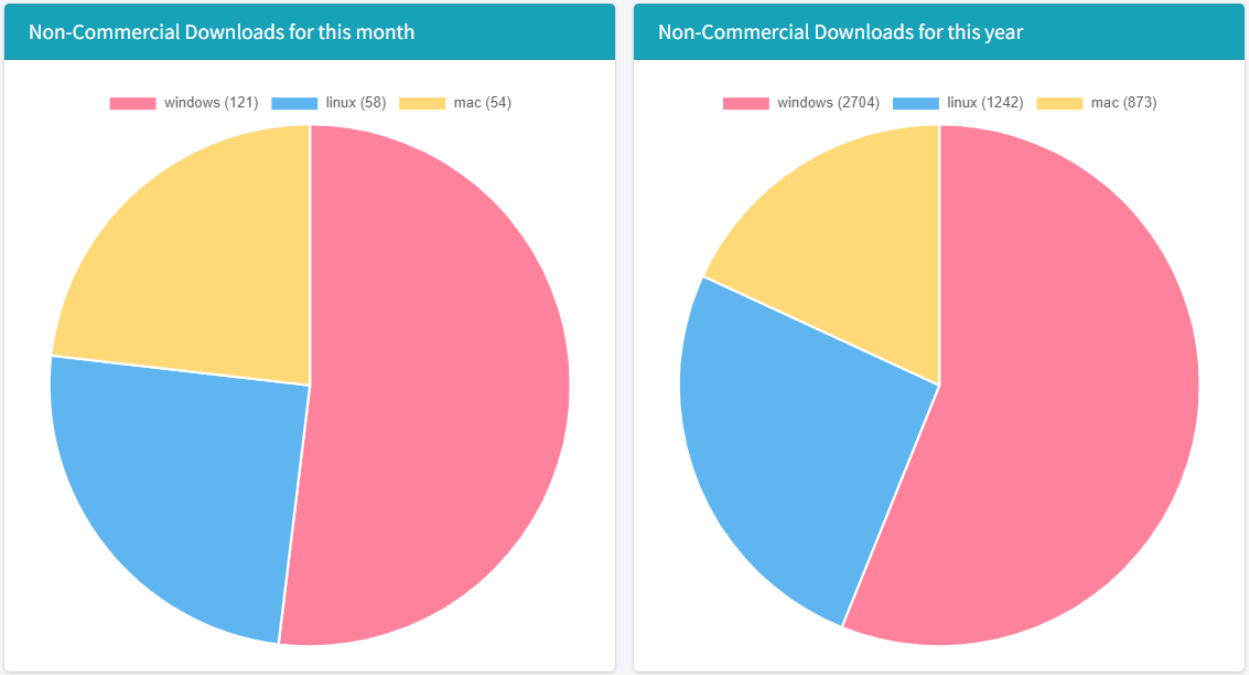
Max Gfeller



Max Sun

Non-Commercial Downloads (snapshot September 18th, 2023)

= ~7,000 / year



README.md

quAPL

Santiago Núñez-Corrales, PhD (nunezco2@illinois.edu)

Marcos Frenkel (marcosf2@illinois.edu)

Bruno de Abreu (babreu@illinois.edu)

National Center for Supercomputing Applications

Description

quAPL is an APL implementation of a collection of primitives that can be used to simulate a quantum computer. The code is inteded to be experimental, and will continue to be expanded upwards to contain composable primitives leading to the vision of an Instruction Set Architecture, as well as downwards to replace core entities with calls to existing quantum implementations, ideally using OpenQASM.

Why APL

- APL naturally captures the effect of operators over vectors in normed, finite-dimensional Hilbert spaces representing quantum states
- APL's array-based syntax allows quantum programmers to focus on the semantics of qubit operators while removing concerns about implementation details unrelated to quantum computing
- APL's ability to optimize operations across multiple data types
- APL's native handling of complex numbers

No releases published

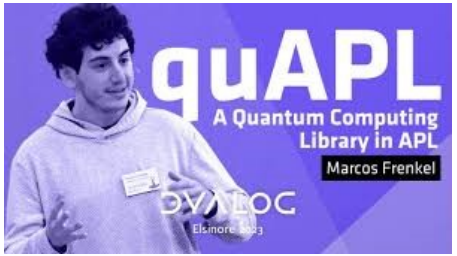
Packages

No packages published

Contributors 2

nunezco2 Santiago Nunez-Corrales

marcosfrenkel



Features of Version 19.0 (March 24)

- Platform Support / Distribution
  - 64-bit ARM support
    - New Macs, Pi 4&5, AWS Graviton
  - Enhanced .NET Bridge
    - Framework vs new .NET versions
  - Bound executables on all platforms
- Building Production Systems
  - Token range reservation
  - WS FULL handling
  - NCOPY/NMOVE callbacks
- Developer Productivity / IDE
  - Source "as typed" by default
  - Multi-line input on by default
  - HTMLRenderer updates
  - Link 4.0: Support for simple text data
  - HttpCommand client, Jarvis web service
- Installing & Managing APL
  - Multiple session files
  - Health Monitor

Service Orientation

A rapidly increasing proportion of new APL code is delivered as services

- ***Jarvis*** wraps APL code as HTTP/JSON or RESTful services on any platform
  - https://github.com/dyalog/jarvis
- Off-the-shelf docker containers containing Dyalog APL (optionally with Jarvis)
- ***HttpCommand*** is our HTTP client

Source Code Management

- **Link** 4.0 was included with v19.0. Highlights include:
  - Link a single namespace or class file
  - Default to current namespace if no namespace specified
  - Configuration files
  - Support for simple text vectors, vectors of text vectors, and character matrices in simple text files (rather than using array notation)
- The **Cider** project manager and the **Tatin** package manager will be bundled with v19.0
- Dyalog is starting to distribute tools as **Tatin** packages

Link (source)

Cider (projects)

Tatin (packages)

aplteam-SMTP	SMTP client for sending emails from within Dyalog APL	2	github.com
aplteam-Snippets	Manage APL-code snippets	1	github.com
aplteam-Tester2	Dyalog APL test framework	1	github.com
aplteam-WindowsEventLog	Tools to read from and write to the Windows Event Log	1	github.com
aplteam-WinReg	Tools for dealing with the Windows Registry	1	github.com
aplteam-WinRegSimple	Limited set of tools for dealing with the Windows Registry	1	github.com
aplteam-WinSCP_NET	Interface between Dyalog and the WINSCP .NET DLL	1	github.com
aplteam-WinSys	Windows-only OS calls and system infos	1	github.com
aplteam-ZipArchive	Zipping and unzipping with .NET on Windows and zip/unzip on other platforms	2	github.com
davin-DateTime	Easy calculations with dates	1	github.com
davin-FilePlus	Extend component files to use anything for a component number	1	github.com
davin-SQLFns	Easily create text SQL commands for use with any SQL program interface	1	github.com
davin-Tester	Simplified function-level testing of programs	1	github.com
dyalog-HttpCommand	Utility to execute HTTP requests		
dyalog-Jarvis	JSON and REST Web Service Framework		
dyalog-NuGet	Use NuGet packages from Dyalog APL		

Coming soon:
dyalog-APLProcess
dyalog-OpenAI

The NuGet Package...

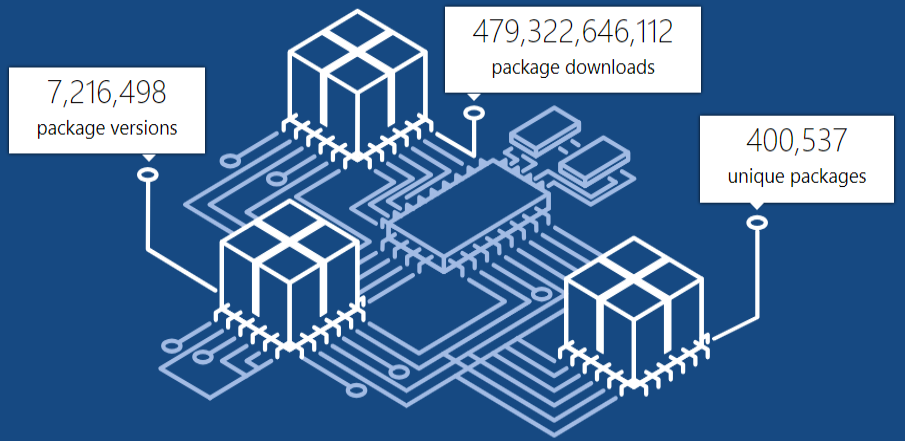
A Tatin package giving access to another package manager ☺

NuGet is the .NET Package Manager.

nuget Packages Upload Statistics Documentation Downloads Blog Sign in

Create .NET apps faster with NuGet

Search for packages...



What is NuGet?

NuGet is the package manager for .NET. The NuGet client tools provide the ability to produce and consume packages. The NuGet Gallery is the central package repository used by all package authors and consumers.

Frameworks

Include compatible frameworks

Framework Filter Mode ALL ANY

- .NET
- .NET Core
- .NET Standard
- .NET Framework

Package type

- All types
- Dependency
- .NET tool
- Template

Options

Include prerelease

Apply Reset

There are 373,610 packages

Sort by Relevance

Newtonsoft.Json

by: dotnetfoundation jamesnk newtonsoft

.NET 5.0 .NET Core 1.0 .NET Standard 1.0 .NET Framework 2.0

4,599,211,003 total downloads last updated 08/03/2023 Latest version: 13.0.3

json

Json.NET is a popular high-performance JSON framework for .NET

Microsoft.Extensions.DependencyInjection

by: aspnet dotnetframework Microsoft

.NET 5.0 .NET Core 2.0 .NET Standard 2.0 .NET Framework 4.6.1

3,270,537,332 total downloads last updated 5 days ago Latest version: 9.0.0-preview.3.24172.9

Default implementation of dependency injection for Microsoft.Extensions.DependencyInjection.

Microsoft.Extensions.Logging

by: aspnet dotnetframework Microsoft

.NET 5.0 .NET Core 2.0 .NET Standard 2.0 .NET Framework 4.6.1

3,046,756,343 total downloads last updated 5 days ago Latest version: 9.0.0-preview.3.24172.9

Logging infrastructure default implementation for Microsoft.Extensions.Logging.

System.Text.Json

by: dotnetframework Microsoft

.NET 5.0 .NET Core 2.0 .NET Standard 2.0 .NET Framework 4.6.1

2,074,912,442 total downloads last updated 5 days ago Latest version: 9.0.0-preview.3.24172.9

Provides high-performance and low-allocating types that serialize objects to JavaScript Object Notation (JSON) text and deserialize JSON text to objects, with UTF-8 support built-in. Also provides types to read... More information

nuget Packages Upload Statistics Documentation Downloads Blog Sign in

mail

Frameworks

Include compatible frameworks

Framework Filter Mode ALL ANY

- .NET
- .NET Core
- .NET Standard
- .NET Framework

Package type

- All types
- Dependency
- .NET tool
- Template

Options

Include prerelease

Apply Reset

1,819 packages returned for mail

Sort by Relevance

MailKit by: jstedfast

.NET 5.0 .NET Core 2.0 .NET Standard 2.0 .NET Framework 4.6.1

108,379,307 total downloads last updated 3 days ago Latest version: 4.5.0

smtp pop3 imap mime security arc dkim smime s/mime openpgp More tags

MailKit is an Open Source cross-platform .NET mail-client library that is based on MimeKit and optimized for mobile devices. Features include: * HTTP, Socks4, Socks4a and Socks5 proxy support. * SASL... More information

SendGrid by: twilio-api

.NET 5.0 .NET Core 1.0 .NET Standard 1.3 .NET Framework 4.0

112,588,456 total downloads last updated 14 days ago Latest version: 9.29.3

Twilio SendGrid Email Mail Microsoft Azure Transactional .NET Core

C# client library and examples for using Twilio SendGrid API's to send mail and access Web API v3 endpoints with .NET Standard 1.3 and .NET Core support.

MailChimp.Net.V3 by: brandonseydel

.NET 5.0 .NET Core 1.0 .NET Standard 1.3 .NET Framework 4.5

4,222,840 total downloads last updated 15/05/2023 Latest version: 5.5.0

MailChimp Mail Chimp 3.0 v3.0 MailChimp.Net.V3 MailChimpv3.0 MailChimpv3 MailChimp3

A .NET Wrapper for Mail Chimp v3.0 API

NETCore.MailKit by: jos_kl

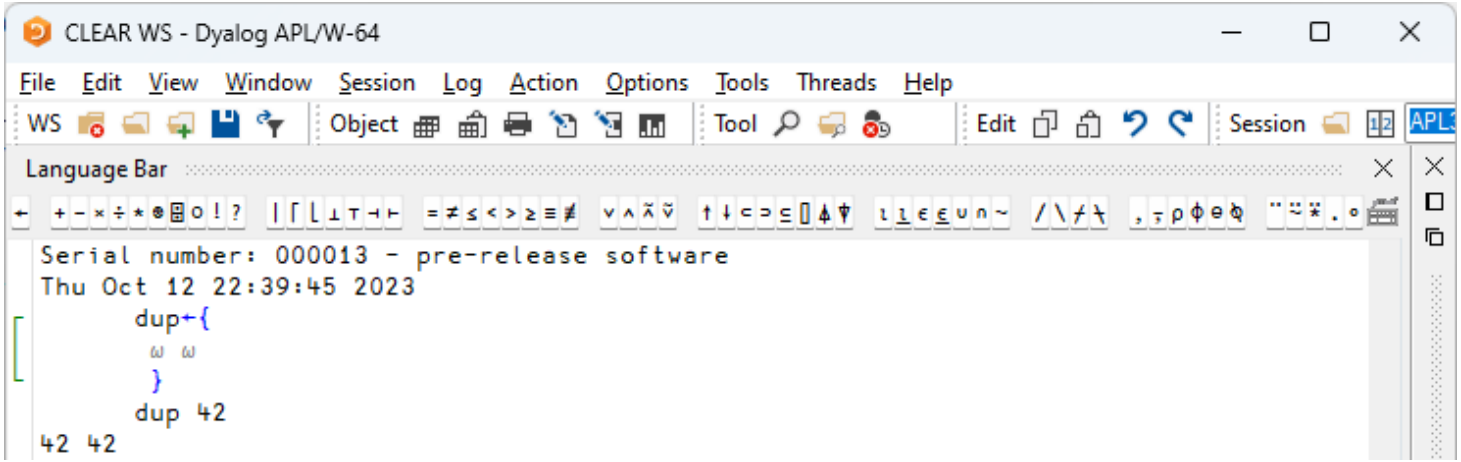
.NET 5.0 .NET Core 3.0 .NET Standard 2.1

The NuGet Tool...

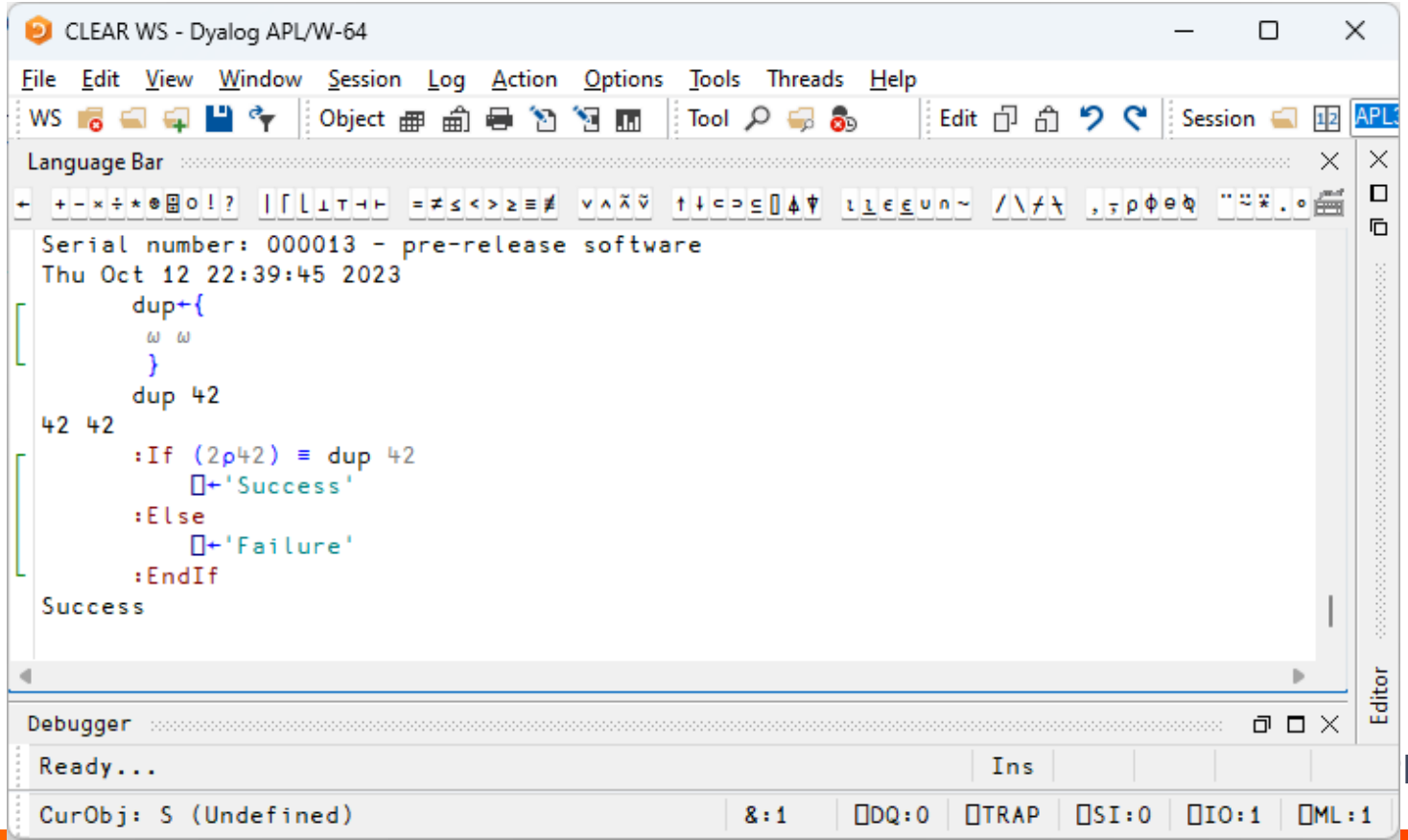
Let's try MailKit....

```
      ]tatin.loadpackages dyalog-nuget
1 package loaded into #

      ]link.import c:\devt\nuget\APLSource\Tests
Imported: #.Tests ← c:\devt\nuget\APLSource\Tests

      Tests.test 'mailkit' ⍝ Trace me
```

Multi-line Input



Multi-line Input



[Microsoft].NET History

- .NET has been around for 20+ years. The old "Framework" is being replaced by an open source, cross-platform .NET, initially known as ".NET Core".

Name	Platforms	Version Numbers
Microsoft.NET Framework	Windows	1 2 3 4.0 4.8.1
".NET Core"	Windows Linux macOS	1 2 3
".NET"	Windows Linux macOS	5.0 6.0 7.0 8.0

- Dyalog v9.5 added a bridge to the Framework in 2002
- Dyalog v18.0 added a bridge to .NET Core 3.0, v18.2 targeted 3.1
- Dyalog v19.0 supports 6.0 - 8.0 (the default, Long Term Support version)

v19.0 .NET Bridge

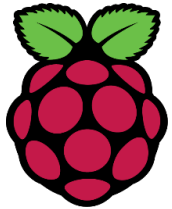
- Adds support for .NET 6, 7, 8 ...
  - Tested with 6.0 & 8.0
  - Configured for 8.0 by default
  - Also full support for 4.8 (aka ".NET Framework") (but not both at the same time)
- Can export APL code as .NET assemblies
  - Will allow embedding APL code in .NET frameworks like ASP.NET Core, etc

.NET 8.0 is the Long Term Support version (LTS)

Arm64

64-bit ARM chips are everywhere:

- M1, M2, M3 Macs
- Raspberry Pi – 64 Bit
- Amazon Web Services "Graviton" (cheaper cloud computing)
- Windows ARM64 in 2025/26 (v21?)



Health Monitor

Experimental TCP-based monitor:

- Interpreter can broadcast regular updates on (e.g.):
  - CPU consumption, Memory statistics
  -)SI stack and Error information
- Notifications on
  - untrapped errors
  - ws compaction
- Last successfully executed line of code
- Details of how to connect Remote IDE if it is possible

Health Monitor Example


```
["PollFacts",{"Facts":["AccountInformation","Workspace","ThreadCount"],"Interval":5000,"UID":"1 1"}]
```

```
["Facts",
{"Facts": [ {
  "ID": 2, "Name": "AccountInformation",
  "Value": {
  "ComputeTime": 438,
  "ConnectTime": 46973,
  "KeyingTime": 0,
  "UserIdentification": 0
}},{
  "ID": 3, "Name": "Workspace",
  "Value": {
  "Allocation": 33882064,
  "AllocationHWM": 33882064,
  "Available": 2144207480,
  "Compactions": 2,
  "FreePockets": 186682,
  "GarbageCollections": 0,
  "GarbagePockets": 0,
  "Sediment": 2120,
  "Used": 3276168,
  "UsedPockets": 23209,
  "WSID": "CLEAR WS"
}},{
  "ID": 6, "Name": "ThreadCount",
  "Value": {
  "Suspended": 1,
  "Total": 2
}}
],
"Interval": 5000,
"UID": "1 1"
}]
```

Health Monitor Example


```
["PollFacts",{"Facts":["AccountInformation","Workspace","ThreadCount"],"Interval":5000,"UID":"1 1"}]
```

```
["Facts",
{"Facts": [ {
  "ID": 2, "Name": "AccountInformation",
  "Value": {
  "ComputeTime": 438,
  "ConnectTime": 46973,
  "KeyingTime": 0,
  "UserIdentification": 0
}},{
  "ID": 3, "Name": "Workspace",
  "Value": {
  "Allocation": 33882064,
  "AllocationHWM": 33882064,
  "Available": 2144207480,
  "Compactions": 2,
  "FreePockets": 186682,
  "GarbageCollections": 0,
  "GarbagePockets": 0,
  "Sediment": 2120,
  "Used": 3276168,
  "UsedPockets": 23209,
  "WSID": "CLEAR WS"
}},{
  "ID": 6, "Name": "ThreadCount",
  "Value": {
  "Suspended": 1,
  "Total": 2
}}
],
"Interval": 5000,
"UID": "1 1"
}]
```

Health Monitor Example


```
["PollFacts",{"Facts":["AccountInformation","Workspace","ThreadCount"],"Interval":5000,"UID":"1 1"}]
```



```
["Facts",
{"Facts": [ {
  "ID": 2, "Name": "AccountInformation",
  "Value": {
  "ComputeTime": 438,
  "ConnectTime": 46973,
  "KeyingTime": 0,
  "UserIdentification": 0
}},{
  "ID": 3, "Name": "Workspace",
  "Value": {
  "Allocation": 33882064,
  "AllocationHWM": 33882064,
  "Available": 2144207480,
  "Compactions": 2,
  "FreePockets": 186682,
  "GarbageCollections": 0,
  "GarbagePockets": 0,
  "Sediment": 2120,
  "Used": 3276168,
  "UsedPockets": 23209,
  "WSID": "CLEAR WS"
}},{
  "ID": 6, "Name": "ThreadCount",
  "Value": {
  "Suspended": 1,
  "Total": 2
}}
],
"Interval": 5000,
"UID": "1 1"
}]
```

Health Monitor Example


```
["PollFacts",{"Facts":["AccountInformation","Workspace","ThreadCount"],"Interval":5000,"UID":"1 1"}]
```



```
["Facts",
{"Facts": [ {
  "ID": 2, "Name": "AccountInformation",
  "Value": {
  "ComputeTime": 438,
  "ConnectTime": 46973,
  "KeyingTime": 0,
  "UserIdentification": 0
}},{
  "ID": 3, "Name": "Workspace",
  "Value": {
  "Allocation": 33882064,
  "AllocationHWM": 33882064,
  "Available": 2144207480,
  "Compactions": 2,
  "FreePockets": 186682,
  "GarbageCollections": 0,
  "GarbagePockets": 0,
  "Sediment": 2120,
  "Used": 3276168,
  "UsedPockets": 23209,
  "WSID": "CLEAR WS"
}},{
  "ID": 6, "Name": "ThreadCount",
  "Value": {
  "Suspended": 1,
  "Total": 2
}}
],
"Interval": 5000,
"UID": "1 1"
}]
```

Health Monitor Example


```
["PollFacts",{"Facts":["AccountInformation","Workspace","ThreadCount"],"Interval":5000,"UID":"1 1"}]
```

```
["Facts",
{"Facts": [ {
  "ID": 2, "Name": "AccountInformation",
  "Value": {
  "ComputeTime": 438,
  "ConnectTime": 46973,
  "KeyingTime": 0,
  "UserIdentification": 0
}},{
  "ID": 3, "Name": "Workspace",
  "Value": {
  "Allocation": 33882064,
  "AllocationHWM": 33882064,
  "Available": 2144207480,
  "Compactions": 2,
  "FreePockets": 186682,
  "GarbageCollections": 0,
  "GarbagePockets": 0,
  "Sediment": 2120,
  "Used": 3276168,
  "UsedPockets": 23209,
  "WSID": "CLEAR WS"
}},{
  "ID": 6, "Name": "ThreadCount",
  "Value": {
  "Suspended": 1,
  "Total": 2
}}
],
"Interval": 5000,
"UID": "1 1"
}]
```

Health Monitor Example

```
["PollFacts",{"Facts":["AccountInformation","Workspace","ThreadCount"],"Interval":5000,"UID":"1 1"}]
```


```
["Facts",
{"Facts": [ {
  "ID": 2, "Name": "AccountInformation",
  "Value": {
  "ComputeTime": 438,
  "ConnectTime": 46973,
  "KeyingTime": 0,
  "UserIdentification": 0
}},{
  "ID": 3, "Name": "Workspace",
  "Value": {
  "Allocation": 33882064,
  "AllocationHWM": 33882064,
  "Available": 2144207480,
  "Compactions": 2,
  "FreePockets": 186682,
  "GarbageCollections": 0,
  "GarbagePockets": 0,
  "Sediment": 2120,
  "Used": 3276168,
  "UsedPockets": 23209,
  "WSID": "CLEAR WS"
}},{
  "ID": 6, "Name": "ThreadCount",
  "Value": {
  "Suspended": 1,
  "Total": 2
}}
],
"Interval": 5000,
"UID": "1 1"
}]
```

Token Allocation

EWC needs up to `MAXSESSIONS` tokens for threads doing DQ or NQ:

```
allocateTokens;z;t;i

t←'EWC Synchronisation' ⍝ Name of range we will use

:Trap 2 ⍝ ⎕TALLOC will SYNTAX ERROR in v18.2
   z←↑⎕TALLOC 0                                ⍝ List all allocated ranges
   :If (≠z)≥i←z[;2]⍳⊂t                         ⍝ Do we already have a range?
      TOKENRANGE←z[i;1]                        ⍝ Yes: use it
   :Else
      TOKENRANGE←⎕TALLOC 1 t                   ⍝ No: Allocate a range
   :EndIf
   TOKENSTEP←10*-⌈10⍟MAXSESSIONS+1             ⍝ e.g. step by 0.001 if MAXSESSIONS is 100
   (WG_TOKENBASE DQ_TOKENBASE)←TOKENRANGE+0.1 0.2 ⍝ Start at n.101 and n.2.01
:Else ⍝ Version 18.2 or earlier
   TOKENSTEP←1                ⍝ Step by 1, and separate the ranges out
   (WG_TOKENBASE DQ_TOKENBASE)←(0 1×1+MAXSESSIONS)+30⊥⎕A⍳'EWC' ⍝ Think of a number
:EndTrap
```

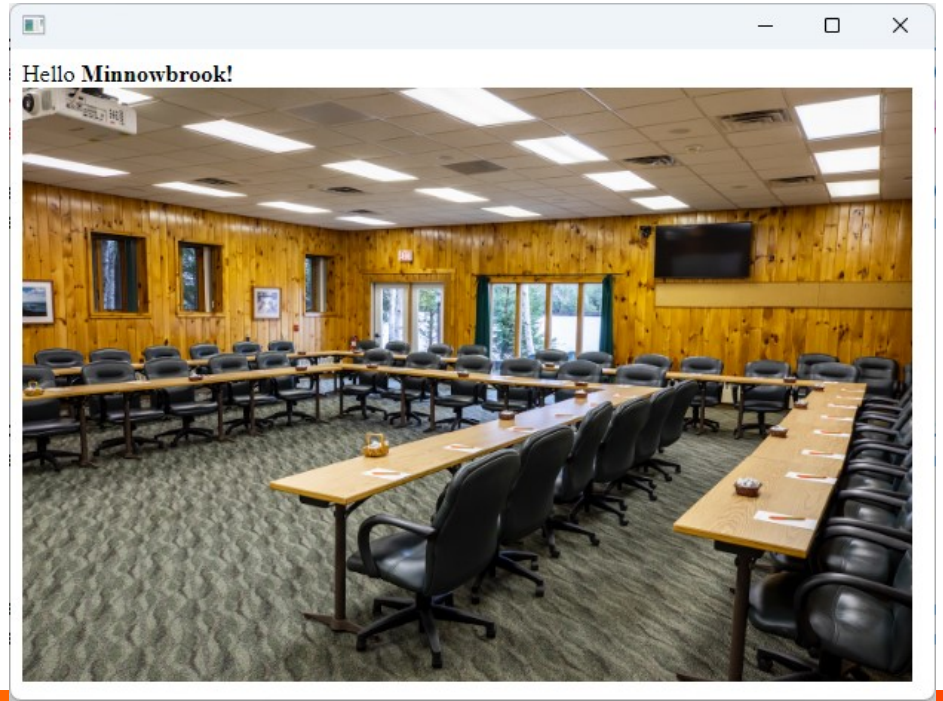
HTMLRenderer updates

New features include:

- AllowContextMenu
- Get/SetZoomLevel
- IsLoading + LoadEnd event

HTMLRenderer – what's that?

```
pic←'https://minnowbrook.org/wp-content/uploads/2022/04/
                    Screen-Shot-2022-04-18-at-2.24.30-PM.png'
'MyForm' ⎕WC 'HTMLRenderer' ('Hello <b>Minnowbrook!</b><br/>
                    <img src="',pic,'" height=400 width=600>')
```



Chromium Embedded Framework (CEF)

HTMLRenderer updates

New features include:

- AllowContextMenu
- Get/SetZoomLevel
- IsLoading + LoadEnd event

Version 19.0 (March 2024)

- Platform Support / Distribution
  - 64-bit ARM support
    - New Macs, Pi 4&5, AWS Graviton
  - Enhanced .NET Bridge
    - Framework vs new .NET versions
  - Bound executables on all platforms
- Building Production Systems
  - Token range reservation
  - WS FULL handling
  - NCOPY/NMOVE callbacks
- Developer Productivity / IDE
  - Source "as typed" by default
  - Multi-line input on by default
  - HTMLRenderer updates
  - Link 4.0: Support for simple text data
  - HttpCommand client, Jarvis web service
- Installing & Managing APL
  - Multiple session files
  - Health Monitor

Sketch of Version 20.0 (Q2/2025)

Ongoing

- Resume Optimisation Work
- .NET Bridge enhancements
  - Support "Generic" methods & classes
- More HTMLRenderer improvements
- Health Monitor
- Script Engine Support

Next Set of Projects

- Relax Interpreter Limits
- Open-Source Plugin Architecture
- Set and Get values w/out Execute
- Array Notation
- Token-by-token Debugging
- New "Shell" System Command
- JavaScript emulation of □WC

Relax Interpreter Limits

We are going to spend the time relaxing limitations in the interpreter, like...

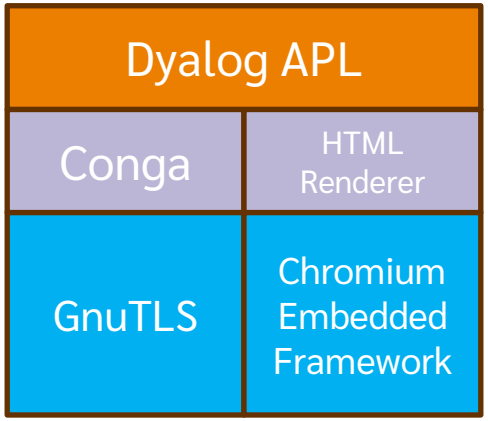
- Max Rank (15)
- # of lines in a function (9,999)
- # tokens in a line (2*15)
- Number of token types (needed to add more language structure)
  - The primitives in Adam Brudzewsky's presentation "Filling the Core Language Gaps" at Dyalog'22 will have to wait a year

Relax Interpreter Limits – Why Now?

- Problem: internal representation of arrays has not changed for decades
  - Different versions of Dyalog APL can share component files
- We must have a solution that avoids "big bang" data conversion events
  - Different versions of APL on client and server exchange arrays in binary format
  - Retrieve archived component files and use immediately
- Now, because
  - We need more token types for new primitives, new control structures, array notation, ...
  - Migrants have functions with >9,999 lines ☺

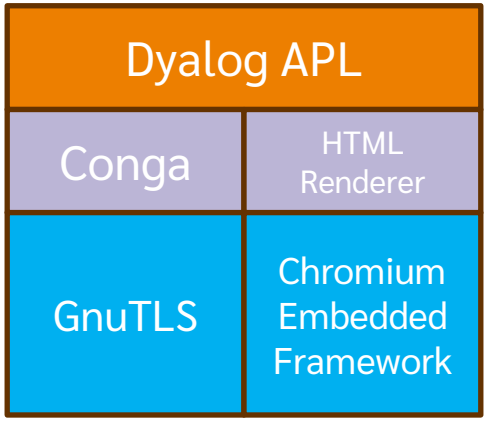
An Open [Source] Plugin Architecture

- A modern replacement for Auxiliary Processors
  - Uses Direct Workspace Access (DWA) for high performance
  - Accessible via ⎕WC / ⎕NEW
- Make parts of the interpreter open source, initially
  - HTMLRenderer
  - Conga (our TCP platform)
  - Cryptographic Library
- These are all interfaces to open source components



Plugin Architecture Benefits

- Allow users to move faster and prototype enhancements rather than wait for Dyalog
  - For example, new versions of Chromium or OpenSSL
- Make the Dyalog community more inclusive by allowing users to contribute
- Users can develop completely new extensions and easily share them with others
- Makes our use of encryption more transparent and verifyable
  - Easier to comply with FOSS licence constraints



Health Monitor

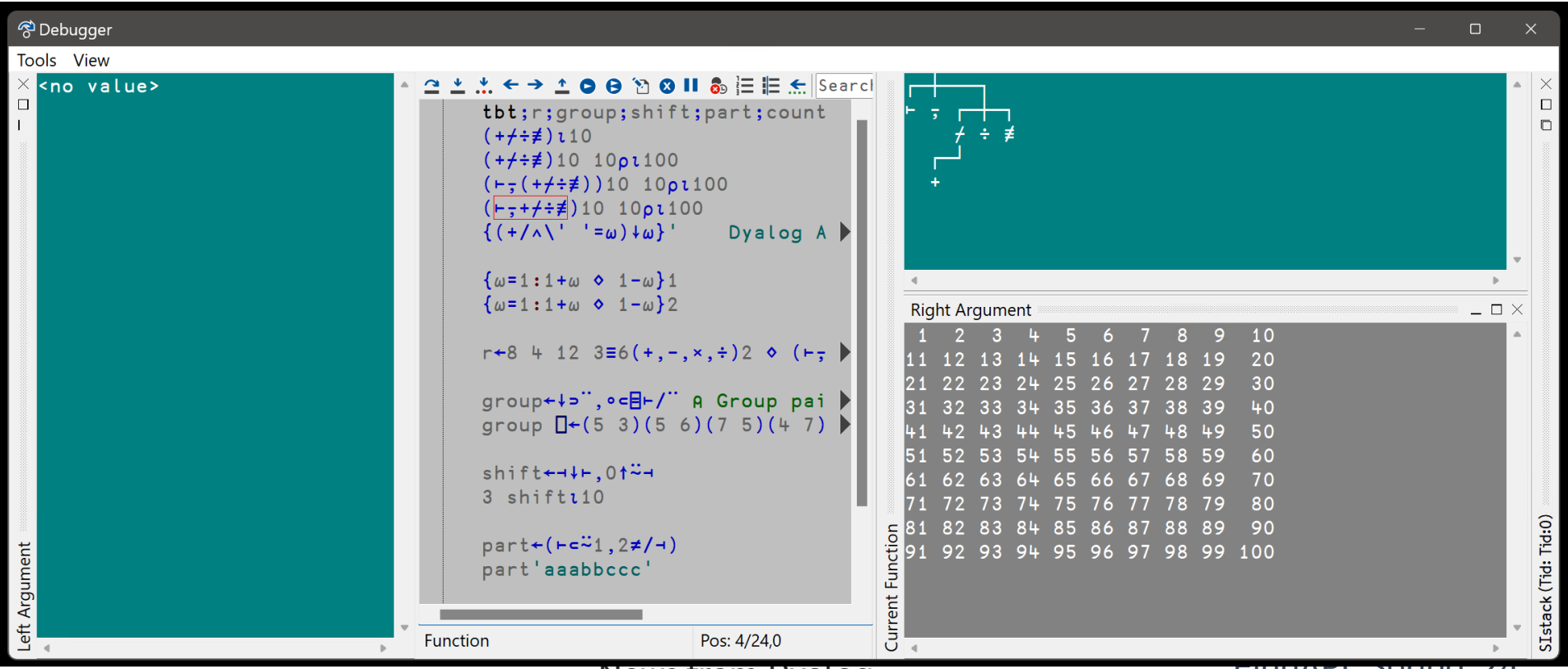
Version 19.0 contains a prototype. Ideas for v20.0 include:

- Complete feature to find last known location of a "hanging" interpreter
- Sending signals to interrupt or terminate tasks
- Discoverability: allow APL process to broadcast services that it provides
- Switch `⎕PROFILE` on and off; collect data
- Possibly add OpenTelemetry data feed

Token-by-token Debugging

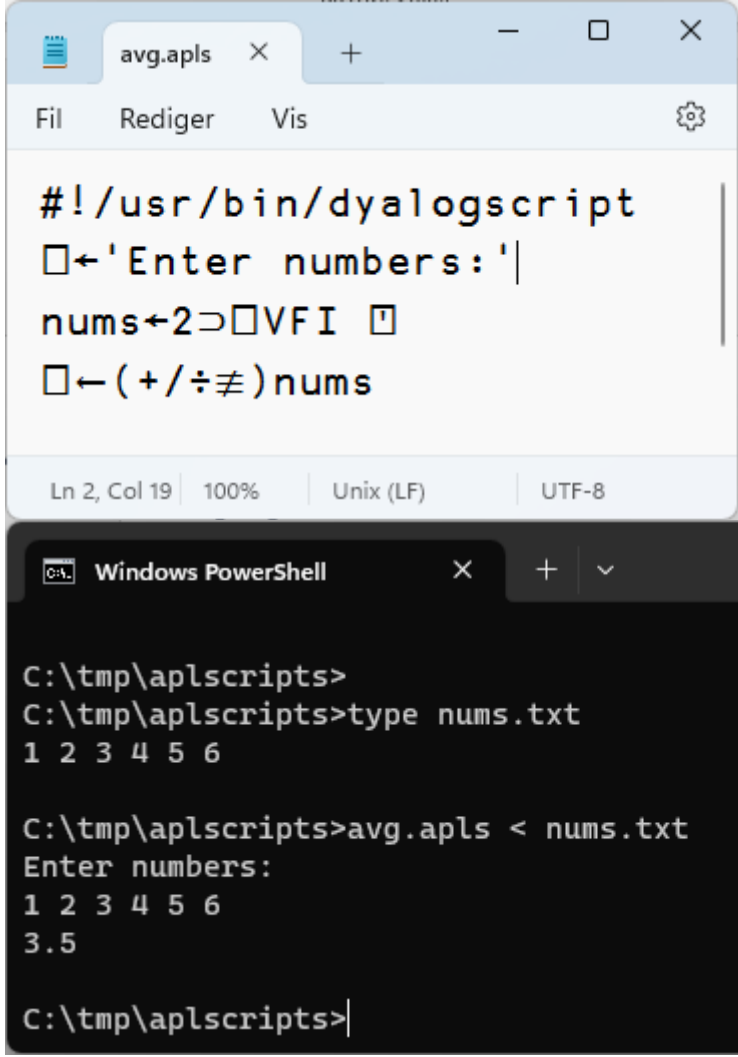
- See John's presentations at Dyalog'22 – and Dyalog'23

Token-by-token Debugging



Script-Engine Support

- #! (hash bang) scripting
- We think the script engine is critical for attracting new users
- Also desired by CI engineers
- Still a bit of a prototype in v19.0
  - Will be hardened for v20.0
  - Need to be able to debug scripts via RIDE



⎕SHELL to ~~replace~~ complement ⎕SH

Invoke OS commands from APL

- Interruptible
- Optionally return data as an asynchronous Stream
- Manage stdin, stdout & stderr independently
- Handle variety of data encodings

HTMLRenderer Enhancements

Suggested:

- File Upload
- Modal HTMLRenderer Windows on all platforms
- More control over "Chrome"
- Other changes driven by EWC project

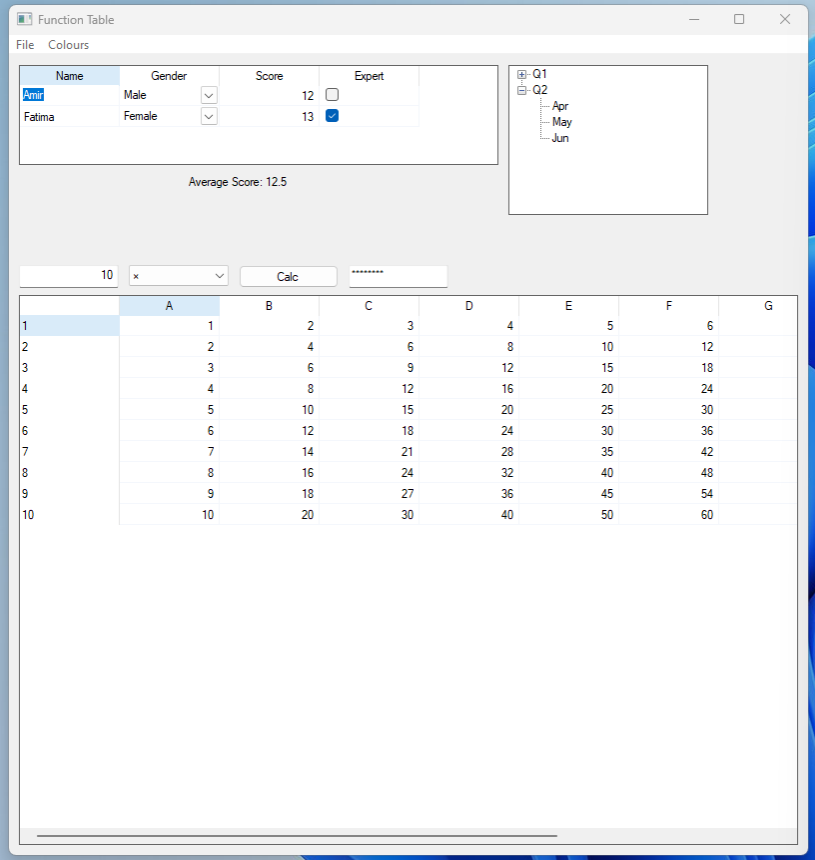
Potential v20.0 Projects

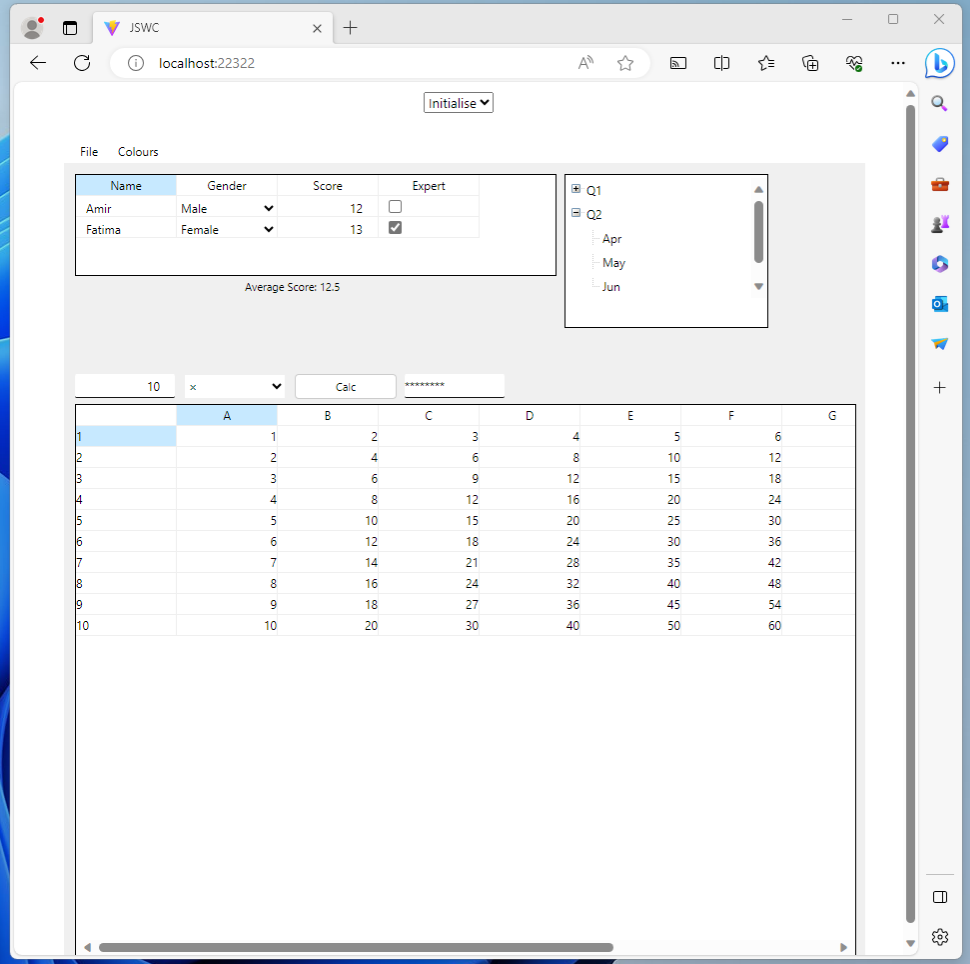
- Actually make use of relaxed limits to increase max rank to 63
- .NET Bridge Enhancements
- HMON Enhancements
- Conga: Initialize by default, add HTTP2
- Plus, outside v20.0 as such
  - Kafka Support
  - Migrations

Easy Web Creator - EWC

- A JavaScript emulation of our Win32 layer (⎕WC, ⎕WG, ⎕WS ...)
- Still often referred to as JSWC (JavaScript Window Create)
- Separate presentation later

Easy Web Creator – EWC
(JavaScript implementation of WC)





.NET Bridge Enhancements

- The v19.0 bridge to .NET 6/7/8 is roughly on par with the Framework bridge
- From v20.0, the new bridge may move ahead, adding:
  - Generics
  - Delegates
  - Async
- Some of these will only be designed / modelled in v20

Static Analysis of APL Code

- Static Analysis of application code is seen as a required "best practice" by some corporations
- We are building a prototype of a tool which will
  - Detect vulnerabilities and other bad practices
  - "Lint" APL Code
  - Compute readability and other metrics
- This tool will initially be licensed separately
- A free "community edition" may follow

Sketch of Version 20.0 (Q2/2025)

Ongoing

- Resume Optimisation Work
- .NET Bridge enhancements
  - Support "Generic" methods & classes
- More HTMLRenderer improvements
- Health Monitor
- Script Engine Support

Next Set of Projects

- Relax Interpreter Limits
- Open-Source Plugin Architecture *
- Set and Get values w/out Execute *
- Array Notation *
- Token-by-token Debugging
- New "Shell" System Command *
- JavaScript emulation of `⎕WC` *

* = Detailed presentation to come

Migrants

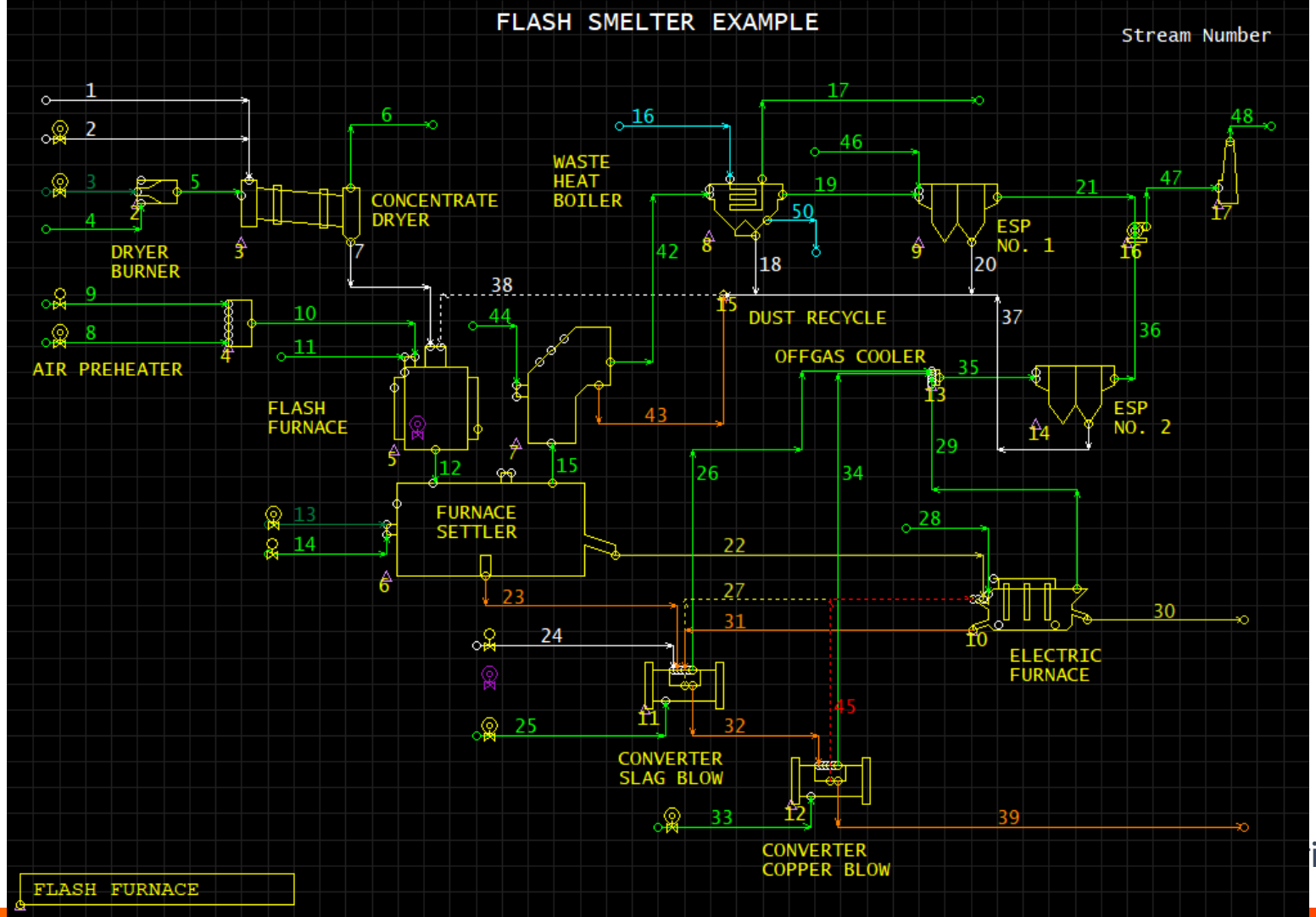
- Every couple of weeks, another company calls to talk about migration from APL2 or APL+Win
- 37 people attended DYNA in New York City a month ago, where migration was the main focus
- We have decided to drop some planned 2024 projects and invest in tools to help migrants
- All resulting tools will be placed in the public domain

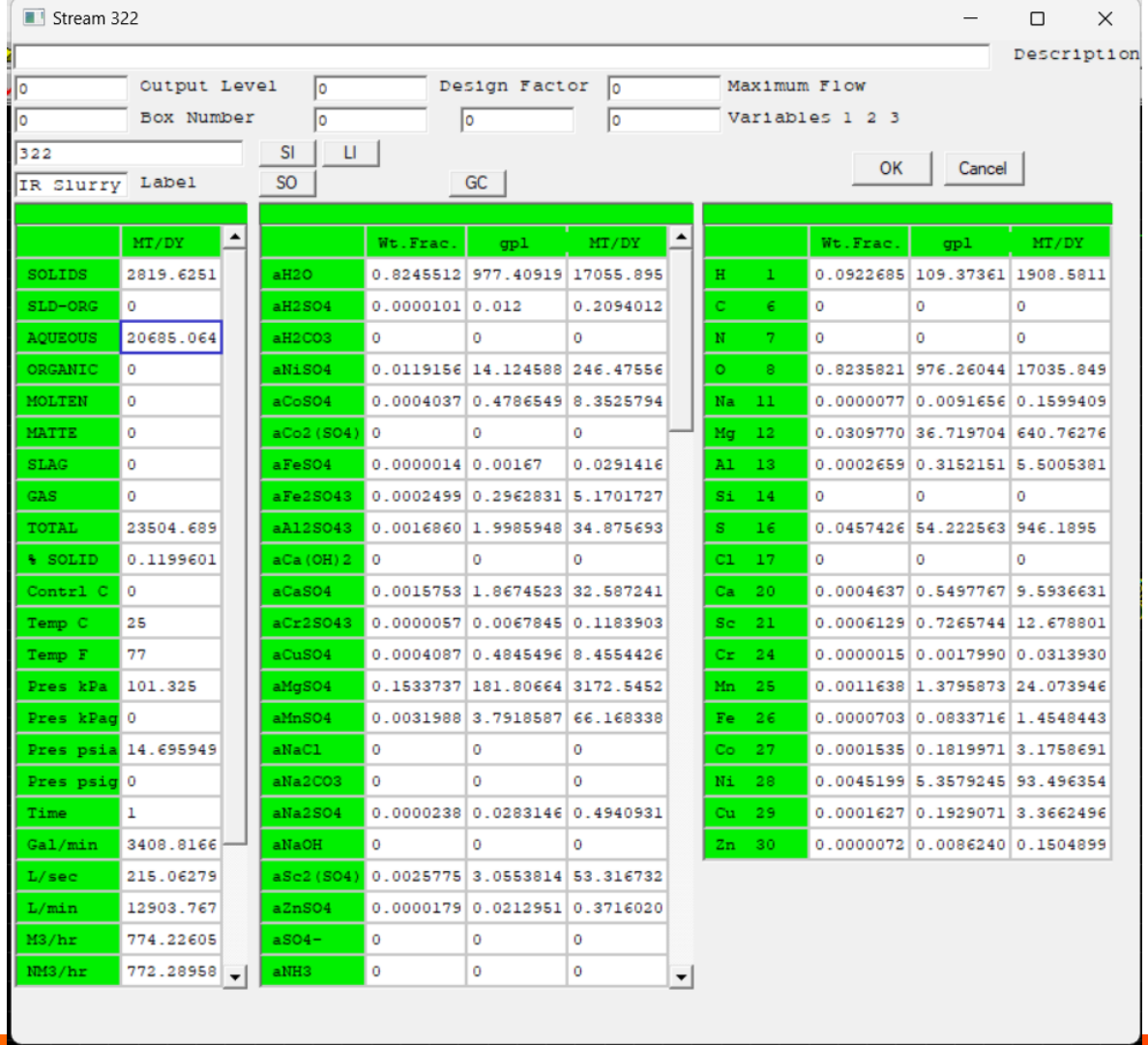
Recent / Active APL2 Migrations

- Insurance company
  - No UI, manipulates text and Excel files
  - Handled by European Consulting Partner
- Sandvik (Sweden) – in progress: Mainframe APL2 direct to Docker Containers and HTML/svg (and Windows desktop)
  - Handled by Tiamatica in Malmö (Gilgamesh Athoraya)
- BIG Jewellers: Windows
  - Handled by Mark Wolfson himself "with a little help"
- Two more under discussion
  - (Germany, Canada)

Recent / Active APL+Win Migrations

- Two European Insurance companies
  - One with GUI, completely rewritten in Dyalog APL, the other a pure service converted to Jarvis in Linux containers
  - Handled by a European consulting partner
- METSIM® - in progress
  - Migration being handled by Dyalog
  - Will be used to develop tools to automate migration, including the Graphical User Interface
- Meeting one more potential migrant next week


FLASH SMELTER EXAMPLE
Stream Number
CONCENTRATE DRYER
DRYER BURNER
WASTE HEAT BOILER
ESP NO. 1
ESP NO. 2
AIR PREHEATER
FLASH FURNACE
DUST RECYCLE
OFFGAS COOLER
FURNACE SETTLER
ELECTRIC FURNACE
CONVERTER SLAG BLOW
CONVERTER COPPER BLOW
FLASH FURNACE



	MT/DY
SOLIDS	2819.6251
SLD-ORG	0
AQUEOUS	20685.064
ORGANIC	0
MOLTEN	0
MATTE	0
SLAG	0
GAS	0
TOTAL	23504.689
% SOLID	0.1199601
Contrl C	0
Temp C	25
Temp F	77
Pres kPa	101.325
Pres kPag	0
Pres psia	14.695949
Pres psig	0
Time	1
Gal/min	3408.8166
L/sec	215.06279
L/min	12903.767
M3/hr	774.22605
NM3/hr	772.28958

	Wt.Frac.	gpl	MT/DY
aH2O	0.8245512	977.40919	17055.895
aH2SO4	0.0000101	0.012	0.2094012
aH2CO3	0	0	0
aNiSO4	0.0119156	14.124588	246.47556
aCoSO4	0.0004037	0.4786549	8.3525794
aCo2(SO4)	0	0	0
aFeSO4	0.0000014	0.00167	0.0291416
aFe2SO43	0.0002499	0.2962831	5.1701727
aAl2SO43	0.0016860	1.9985948	34.875693
aCa(OH)2	0	0	0
aCaSO4	0.0015753	1.8674523	32.587241
aCr2SO43	0.0000057	0.0067845	0.1183903
aCuSO4	0.0004087	0.4845496	8.4554426
aMgSO4	0.1533737	181.80664	3172.5452
aMnSO4	0.0031988	3.7918587	66.168338
aNaCl	0	0	0
aNa2CO3	0	0	0
aNa2SO4	0.0000238	0.0283146	0.4940931
aNaOH	0	0	0
aSc2(SO4)	0.0025775	3.0553814	53.316732
aZnSO4	0.0000179	0.0212951	0.3716020
aSO4-	0	0	0
aNH3	0	0	0

	Wt.Frac.	gpl	MT/DY
H 1	0.0922685	109.37361	1908.5811
C 6	0	0	0
N 7	0	0	0
O 8	0.8235821	976.26044	17035.849
Na 11	0.0000077	0.0091656	0.1599409
Mg 12	0.0309770	36.719704	640.76276
Al 13	0.0002659	0.3152151	5.5005381
Si 14	0	0	0
S 16	0.0457426	54.222563	946.1895
Cl 17	0	0	0
Ca 20	0.0004637	0.5497767	9.5936631
Sc 21	0.0006129	0.7265744	12.678801
Cr 24	0.0000015	0.0017990	0.0313930
Mn 25	0.0011638	1.3795873	24.073946
Fe 26	0.0000703	0.0833716	1.4548443
Co 27	0.0001535	0.1819971	3.1758691
Ni 28	0.0045199	5.3579245	93.496354
Cu 29	0.0001627	0.1929071	3.3662496
Zn 30	0.0000072	0.0086240	0.1504899

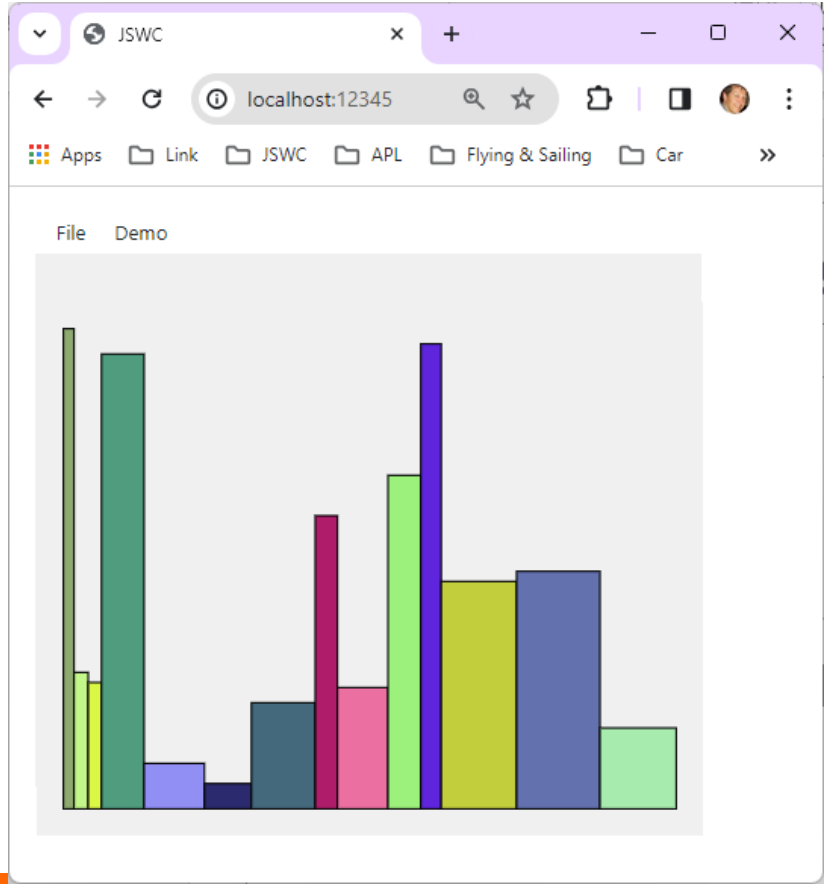
Status

- Dyalog has been contracted to port the METSIM® application
- Hired one new APL developer, thinking about another
- We will have ~1.5-2 full time equivalent resources working on migration tools until further notice

- **All the resulting tools and documentation will be free and open source**

APL+Win "DEMODRAW"

- Under Dyalog APL
- In a Browser
- With no code changes ☺



Demos?

- OpenAI
- EWC