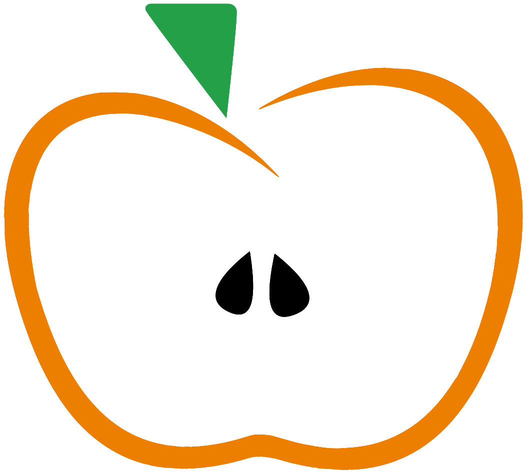
DYALOG

APL Seeds 2022

Welcome



Gitte Christensen

Gitte Christensen, MD Dyalog Ltd
Cand. Scient. Biology

What is APL?

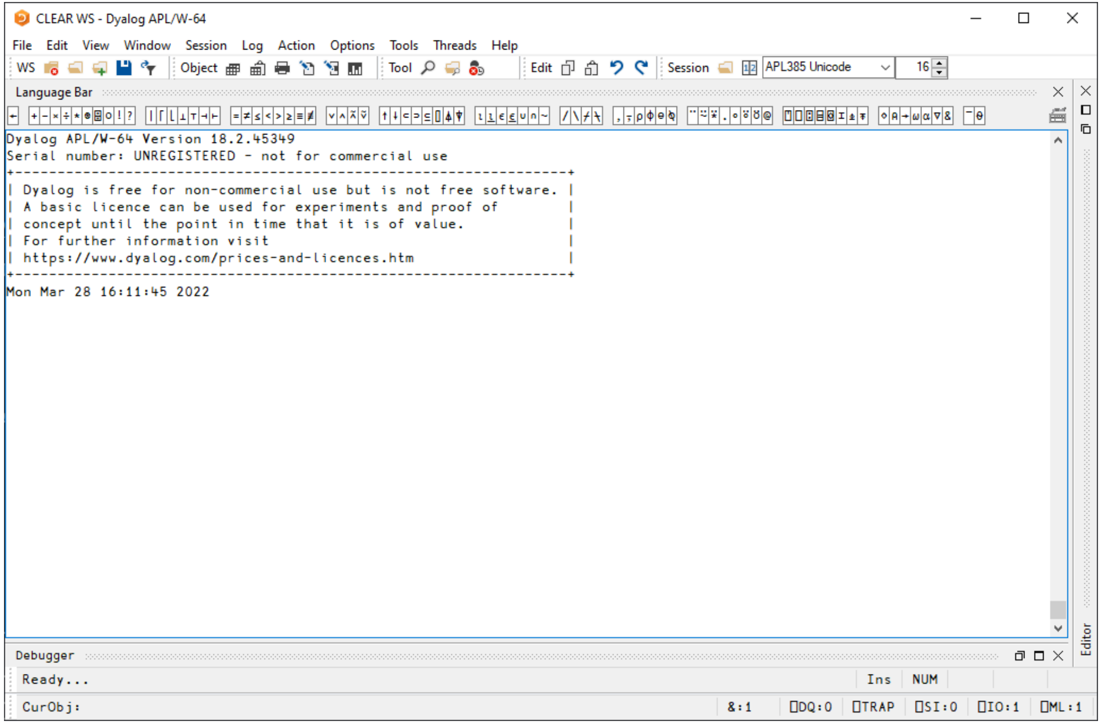
An executable notation and **a tool of thought** which continues **to enable people** with good ideas **to bring** those **ideas to life** with computers.

```
      1 2 3 + 4 5 6
5 7 9
```

Dyalog Ltd

- Dyalog interpreter and application development platform
- Dedicated to the evolution and promotion of APL
- Bringing the benefits of APL to a wider audience

Dyalog Interpreter and IDE



```
Dyalog APL/W-64 Version 18.2.45349
Serial number: UNREGISTERED - not for commercial use
+-----------------------------------------------------------------+
| Dyalog is free for non-commercial use but is not free software. |
| A basic licence can be used for experiments and proof of        |
| concept until the point in time that it is of value.            |
| For further information visit                                   |
| https://www.dyalog.com/prices-and-licences.htm                  |
+-----------------------------------------------------------------+
Mon Mar 28 16:11:45 2022
```

APL Core with Many Tools

Functional programming	{α+ω}
Object-oriented programming	⎕NEW
.NET	⎕USING
Convert between formats	⎕JSON, ⎕CSV, ⎕XML
Datetimes	⎕DT
Regular expressions	⎕R, ⎕S
Graphics	⎕CY'sharpplot'
GUI	⎕WC, github/dyalog/DUI
SQL (ODBC)	⎕CY'sqapl'
TCP/IP	⎕CY'conga'
Parallel Computing	⎕CY'isolate'

New: import directly from the web

```
]Get github.com/Dyalog/Jarvis/blob/master/Source/Jarvis.dyalog
```

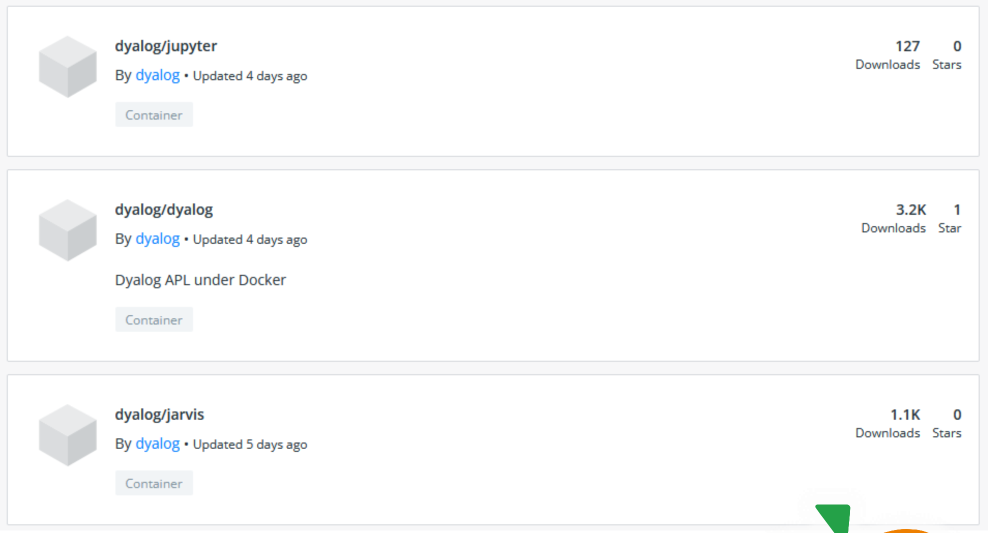
New Basic License

- From Dyalog v18.2, the Non-commercial licence is replaced by a Basic licence
- A Basic Licence is a **free licence** that allows APL users to have a copy of the latest Dyalog technology **for personal or non-commercial use** and experimentation.
- Allows **distribution of Dyalog along with your work** under the terms of the Royalty-Based Run-Time Licence, which will apply as the default run-time licence.
  - Fee is 2% of gross APL-based revenue
  - No fee if revenue < GBP 5,000 in a calendar year
  - Multiple alternative commercial license schemes are available

Ways to Distribute your APL App

- Executable desktop applications
- Web services
- Docker containers

hub.docker.com/u/dyalog

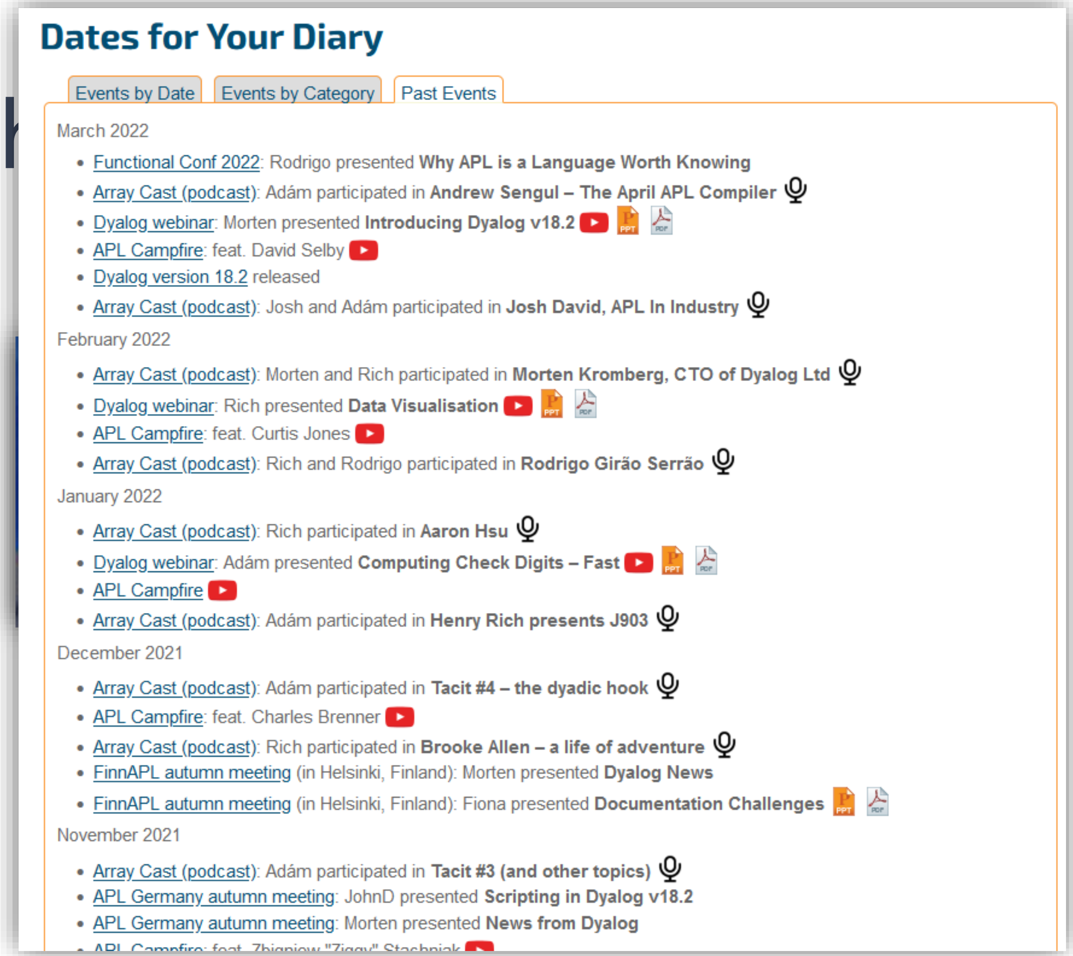


dyalog/jupyter
By dyalog • Updated 4 days ago
Container
127 Downloads 0 Stars

dyalog/dyalog
By dyalog • Updated 4 days ago
Dyalog APL under Docker
Container
3.2K Downloads 1 Star

dyalog/jarvis
By dyalog • Updated 5 days ago
Container
1.1K Downloads 0 Stars

Outreach



Dates for Your Diary

March 2022

- Functional Conf 2022: Rodrigo presented **Why APL is a Language Worth Knowing**
- Array Cast (podcast): Adám participated in **Andrew Sengul – The April APL Compiler**
- Dyalog webinar: Morten presented **Introducing Dyalog v18.2**
- APL Campfire: feat. David Selby
- Dyalog version 18.2 released
- Array Cast (podcast): Josh and Adám participated in **Josh David, APL In Industry**

February 2022

- Array Cast (podcast): Morten and Rich participated in **Morten Kromberg, CTO of Dyalog Ltd**
- Dyalog webinar: Rich presented **Data Visualisation**
- APL Campfire: feat. Curtis Jones
- Array Cast (podcast): Rich and Rodrigo participated in **Rodrigo Girão Serrão**

January 2022

- Array Cast (podcast): Rich participated in **Aaron Hsu**
- Dyalog webinar: Adám presented **Computing Check Digits – Fast**
- APL Campfire
- Array Cast (podcast): Adám participated in **Henry Rich presents J903**

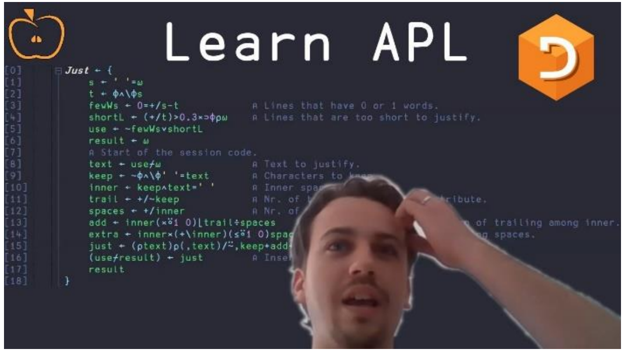
December 2021

- Array Cast (podcast): Adám participated in **Tacit #4 – the dyadic hook**
- APL Campfire: feat. Charles Brenner
- Array Cast (podcast): Rich participated in **Brooke Allen – a life of adventure**
- FinnAPL autumn meeting (in Helsinki, Finland): Morten presented **Dyalog News**
- FinnAPL autumn meeting (in Helsinki, Finland): Fiona presented **Documentation Challenges**

November 2021

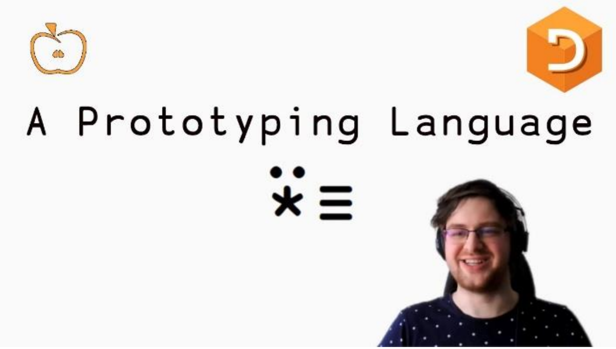
- Array Cast (podcast): Adám participated in **Tacit #3 (and other topics)**
- APL Germany autumn meeting: JohnD presented **Scripting in Dyalog v18.2**
- APL Germany autumn meeting: Morten presented **News from Dyalog**
- APL Campfire: feat. Zbigniew "Ziggy" Stachniak

Dyalog.tv/APLSeeds21



Learn APL

```
Just ← {
    s ← ' '=⍵
    t ← ⌽∧\⌽s
    fewWs ← 0=+/s-t              ⍝ Lines that have 0 or 1 words.
    shortL ← (+/t)>0.3×⊃⌽⍴⍵      ⍝ Lines that are too short to justify.
    use ← ~fewWs∨shortL
    result ← ⍵
    ⍝ Start of the session code.
    text ← use⌿⍵                 ⍝ Text to justify.
    keep ← ~⌽∧\⌽' '=text         ⍝ Characters to
    inner ← keep∧text=' '        ⍝ Inner spa
    trail ← +/~keep              ⍝ Nr. of                tribute.
    spaces ← +/inner             ⍝ Nr. of
    add ← inner(×⍤1 0)⌊trail÷spaces                      of trailing among inner.
    extra ← inner×(+\inner)(≤⍤1 0)spac                   g spaces.
    just ← (⍴text)⍴(,text)/⍨,keep+add
    (use⌿result) ← just          ⍝ Inse
    result
}
```



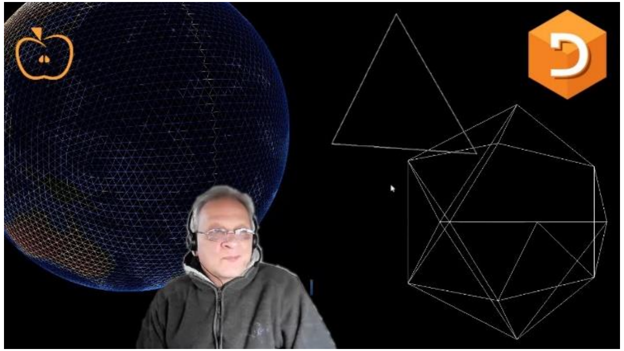
A Prototyping Language

⍣≡

ALG⌽RITHMS AS A
T⌽⌽L ⌽F TH⌽UGHT

Conor Hoekstra

code_report



So who uses APL?

Production
Management
Finance
Medicine
Science
Simulation
Computer Science

People solving new problems, or solving them in a new way

People solving problems or manipulating data in environments where the conditions are constantly changing

So who uses APL?

Asset Management

- SimCorp (DK & IT)
- Tegra118 (US)

Business Intelligence

- KCI Corp (US)
- Carlisle Group
- IBM Cognos Planning

Manufacturing

- Just-in-Time Production Planning (Auto)
- Propeller Design (Marine Engines)

Energy

- Refinery Optimisation
- Product Design

Medicine

- Medical Records
- Pharmaceutical Production Modeling

Gaming

- Stormwind (Finland)

APL Seeds '22

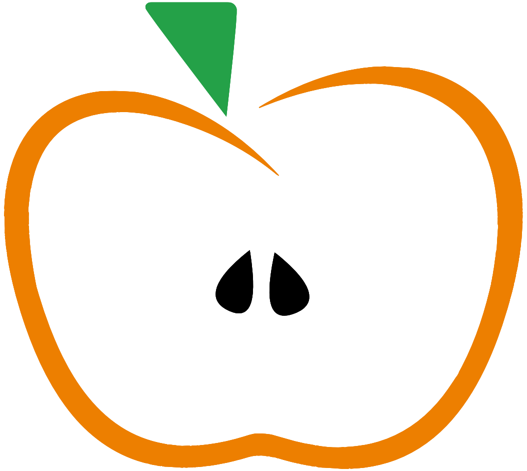
We hope you enjoy this event

DYALOG

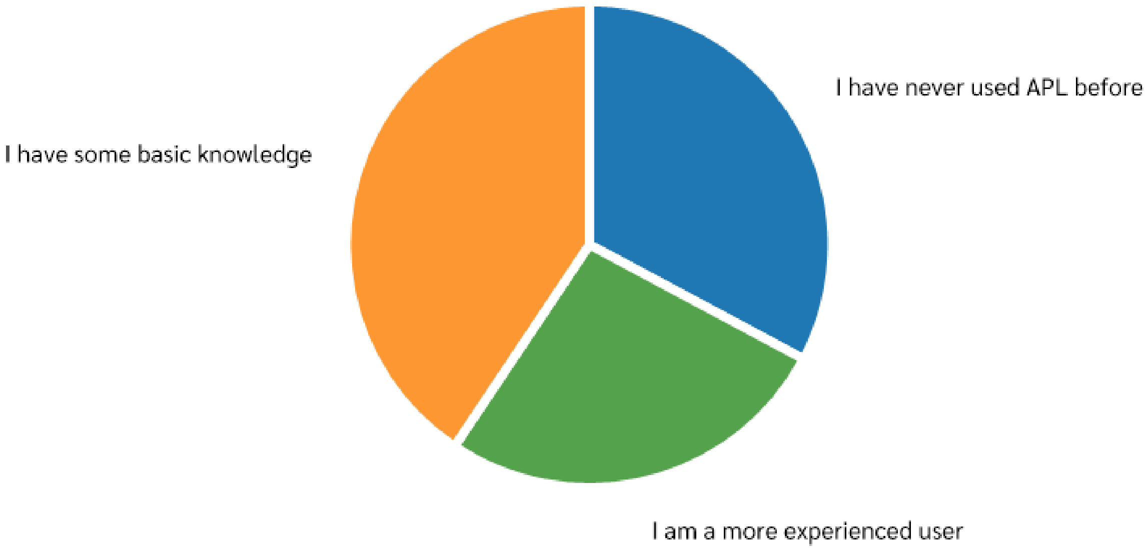
APL Seeds 2022

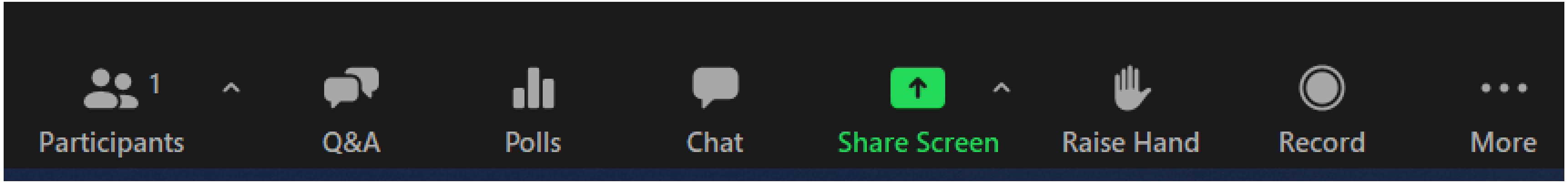
Growing APLers

Rich Park



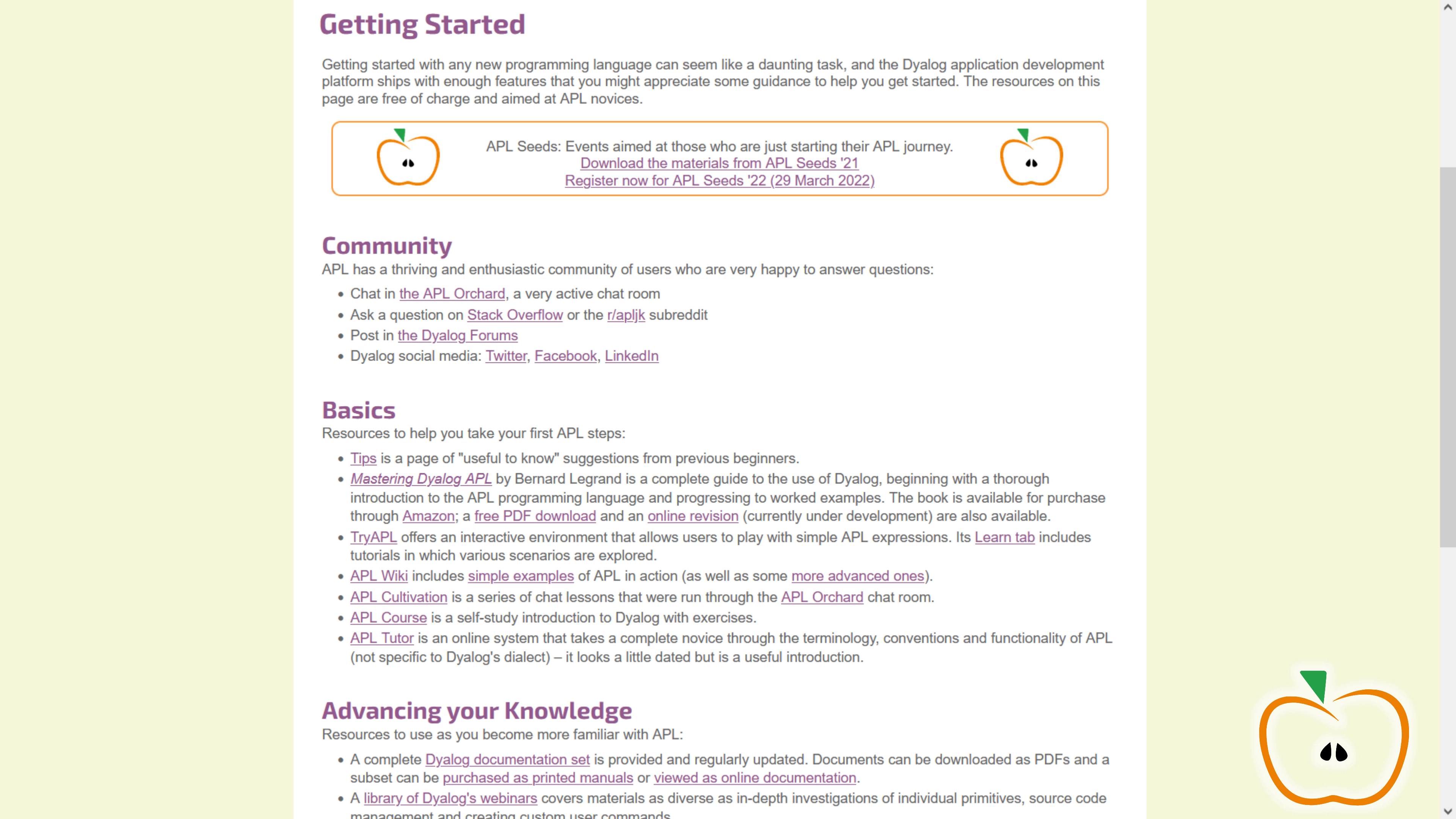
Planting Seeds




1
Participants
Q&A
Polls
Chat
Share Screen
Raise Hand
Record
More

Learning APL

dyalog.com/getting-started.htm



Getting Started

Getting started with any new programming language can seem like a daunting task, and the Dyalog application development platform ships with enough features that you might appreciate some guidance to help you get started. The resources on this page are free of charge and aimed at APL novices.

APL Seeds: Events aimed at those who are just starting their APL journey.
Download the materials from APL Seeds '21
Register now for APL Seeds '22 (29 March 2022)

Community

APL has a thriving and enthusiastic community of users who are very happy to answer questions:

- Chat in the APL Orchard, a very active chat room
- Ask a question on Stack Overflow or the r/apljk subreddit
- Post in the Dyalog Forums
- Dyalog social media: Twitter, Facebook, LinkedIn

Basics

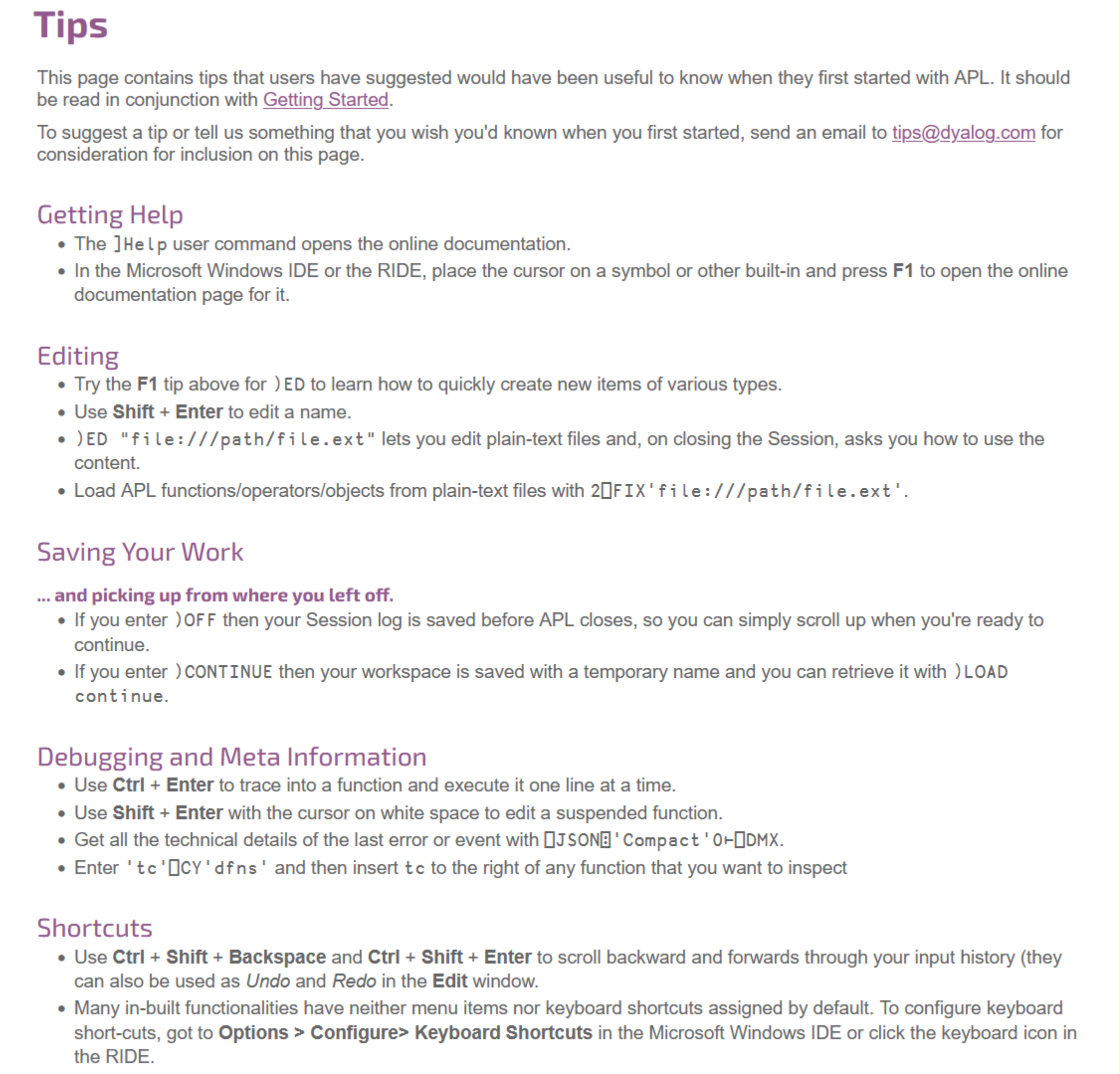
Resources to help you take your first APL steps:

- Tips is a page of "useful to know" suggestions from previous beginners.
- *Mastering Dyalog APL* by Bernard Legrand is a complete guide to the use of Dyalog, beginning with a thorough introduction to the APL programming language and progressing to worked examples. The book is available for purchase through Amazon; a free PDF download and an online revision (currently under development) are also available.
- TryAPL offers an interactive environment that allows users to play with simple APL expressions. Its Learn tab includes tutorials in which various scenarios are explored.
- APL Wiki includes simple examples of APL in action (as well as some more advanced ones).
- APL Cultivation is a series of chat lessons that were run through the APL Orchard chat room.
- APL Course is a self-study introduction to Dyalog with exercises.
- APL Tutor is an online system that takes a complete novice through the terminology, conventions and functionality of APL (not specific to Dyalog's dialect) – it looks a little dated but is a useful introduction.

Advancing your Knowledge

Resources to use as you become more familiar with APL:

- A complete Dyalog documentation set is provided and regularly updated. Documents can be downloaded as PDFs and a subset can be purchased as printed manuals or viewed as online documentation.
- A library of Dyalog's webinars covers materials as diverse as in-depth investigations of individual primitives, source code management and creating custom user commands.



Tips

This page contains tips that users have suggested would have been useful to know when they first started with APL. It should be read in conjunction with Getting Started.

To suggest a tip or tell us something that you wish you'd known when you first started, send an email to tips@dyalog.com for consideration for inclusion on this page.

Getting Help

- The `]Help` user command opens the online documentation.
- In the Microsoft Windows IDE or the RIDE, place the cursor on a symbol or other built-in and press **F1** to open the online documentation page for it.

Editing

- Try the **F1** tip above for `)ED` to learn how to quickly create new items of various types.
- Use **Shift** + **Enter** to edit a name.
- `)ED "file:///path/file.ext"` lets you edit plain-text files and, on closing the Session, asks you how to use the content.
- Load APL functions/operators/objects from plain-text files with `2⎕FIX'file:///path/file.ext'`.

Saving Your Work

... and picking up from where you left off.

- If you enter `)OFF` then your Session log is saved before APL closes, so you can simply scroll up when you're ready to continue.
- If you enter `)CONTINUE` then your workspace is saved with a temporary name and you can retrieve it with `)LOAD continue`.

Debugging and Meta Information

- Use **Ctrl** + **Enter** to trace into a function and execute it one line at a time.
- Use **Shift** + **Enter** with the cursor on white space to edit a suspended function.
- Get all the technical details of the last error or event with `⎕JSON⍠'Compact'0⊢⎕DMX`.
- Enter `'tc'⎕CY'dfns'` and then insert `tc` to the right of any function that you want to inspect

Shortcuts

- Use **Ctrl** + **Shift** + **Backspace** and **Ctrl** + **Shift** + **Enter** to scroll backward and forwards through your input history (they can also be used as *Undo* and *Redo* in the **Edit** window.
- Many in-built functionalities have neither menu items nor keyboard shortcuts assigned by default. To configure keyboard short-cuts, got to **Options > Configure> Keyboard Shortcuts** in the Microsoft Windows IDE or click the keyboard icon in the RIDE.

Dyalog APL - Vocabulary

· Function | Monadic Operator | Dyadic Operator | Array | Multi-Role | Control · X, Y: Arrays f, g, h: Functions

Plus	+	Conjugate	Minus	−	Negate	Times	×	Direction	Divide	÷	Reciprocal
Maximum	⌈	Ceiling	Minimum	⌊	Floor	Power	*	Exponential	Logarithm	⍟	Natural Log
Circular	○	Pi Times	Binomial	!	Factorial	Deal	?	Roll	Residue	|	Magnitude
And/LCM	∧		Or/GCD	∨		Nand	⍲		Nor	⍱	
Equal	=		Not Equal	≠	Unique mask	Match	≡	Depth	Not Match	≢	Tally
Less Than	<		Less Or Equal	≤		Greater Or Equal	≥		Greater Than	>	
Index	⌷	Materialise	Take	↑	Mix	Drop	↓	Split	Reshape	⍴	Shape
Partitioned Enclose	⊂	Enclose	Pick	⊃	First	Left	⊣	Same	Right	⊢	Same
Partition	⊆	Nest	Union	∪	Unique	Intersection	∩		Without	~	Not
Indices Of	⍳	Indices	Interval Index	⍸	Where	Member Of	∊	Enlist	Find	⍷	
Replicate	/		Expand	\		Replicate First	⌿		Expand First	⍀	
Rotate	⌽	Reverse	Rotate First	⊖	Reverse First	Catenate	,	Ravel	Catenate First	⍪	Table
Reorder Axes	⍉	Transpose	Decode	⊥		Encode	⊤		Matrix Divide	⌹	Matrix Inverse
Namespace Execute	⍎	Execute	Specified Format	⍕	Format	Collated Grade By	⍋	Grade up	Collated Grade By	⍒	Grade down

Each	f¨	Each	Constant	X⍨	Constant	Swap	f⍨	Self	System	X⌶	System
N-Wise Reduce	f/	Reduce	N-Wise Reduce First	f⌿	Reduce First		f\	Scan		f⍀	Scan First
Key	f⌸	Index Key	Spawn	f&	Spawn	Axis	f[X]	Axis	Outer Product	∘.g	
Until	f⍣g	Until	Repeat	f⍣Y	Repeat				Variant	f⍠Y	Variant
Beside	f∘g	Beside		f∘Y	Bind		X∘g	Bind		f⌺Y	Stencil
Atop	f⍤g	Atop	Rank	f⍤Y	Rank				Amends	@	Amend
Over	f⍥g	Over							Inner Product	f.g	

→	Abort	⋄	Statement Separator	:	Guard	::	Error Guard
{…}	Dfn	{⍺⍺}	Monadic Dop	{⍵⍵}	Dyadic Dop	⍬	Empty Numeric Vector
⍺	Left Argument	∇	Function Self	⍵	Right Argument	#	Root
⍺⍺	Left Operand	∇∇	Operator Self	⍵⍵	Right Operand	##	Parent
←	Assign	⍝	Comment	⎕	Evaluated Input/Stdout	⍞	Text Input/Stderr
(f g h)	Fork	(X g h)	Fork	(g h)	Atop	¯	Negative

awagga.github.io/dyalog/voc

Learning APL

dyalog.com/getting-started.htm
apl.wiki/learning_resources
mastering.dyalog.com
tryapl.org
dyalog.tv
New: xpqz.github.io/learnapl

The APL Way

> Every reader should ask himself periodically "Toward what end, toward what end?"—but do not ask it too often lest you pass up the fun of programming for the constipation of bittersweet philosophy.
> *–Alan Perlis*

Up until now, we've skirted around one of the main advantages of APL – array-oriented, or *data-parallel* programming. This feels awkward and unnatural at first, but finding data-parallel approaches to problems is a skill that makes for efficient solutions in other languages, too, not just APL, and libraries such as Python's NumPy encourages such solutions (it was inspired by APL, by the way).

Note

In this chapter, we'll be making some comparisons between data-parallel APL and "loop & branch" implementations in Python. We chose Python because its syntax is clean and understandable by a large proportion of programmers from other languages, too. In case it's not immediately obvious, no effort has been made to find optimal Python solutions here; indeed, quite the opposite. View the Python examples as pseudocode illustrations of the algorithms, and yes, we're fully aware that one can string together elegant, efficient Python solutions using iterator algebra and comprehensions.

A few pointers – Richard Park gave a series of webinars on Thinking in APL that you should check out, and Adám Brudzewski gave several interactive Cultivations dedicated to the topic, Lesson 39 - Array programming techniques and Lesson 42 - Array coding style in depth, too.

```
⎕IO ← 0
]box on -s=min
]rows on
assert←{α←'assertion failure' ⋄ 0∊ω:α ⎕signal 8 ⋄ shy←0}
```

Click to show

Loamy Earth

dyalog.com/getting-started.htm
apl.wiki/learning_resources
mastering.dyalog.com
tryapl.org
dyalog.tv

New: xpqz.github.io/learnapl
Coming soon: tutorial.dyalog.com revamp

▼ Table of Contents

APL TUTOR

Table of Contents
Watch Intro Video
DYALOG

Loamy Earth

dyalog.com/getting-started.htm
apl.wiki/learning_resources
mastering.dyalog.com
tryapl.org
dyalog.tv
New: xpqz.github.io/learnapl
Coming soon: tutorial.dyalog.com revamp
course.dyalog.com

Problem Set 1

1. A Mathematical Notation

   Use APL to evaluate the following

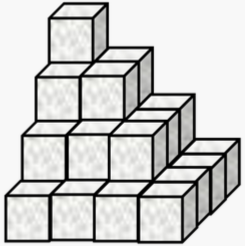
   a. $\prod_{n=1}^{12} n$ (multiply together the first twelve integers)
   b. $\sum_{n=1}^{17} n^2$ (add together the first seventeen squared integers)
   c. $\sum_{n=1}^{100} 2n$ (add together the first one hundred positive even integers)
   d. $\sum_{n=1}^{100} 2n - 1$ (add together the first one hundred odd integers)
   e. In traditional mathematical notation (TMN), the following equation equals `0`, why does the following return `70`?

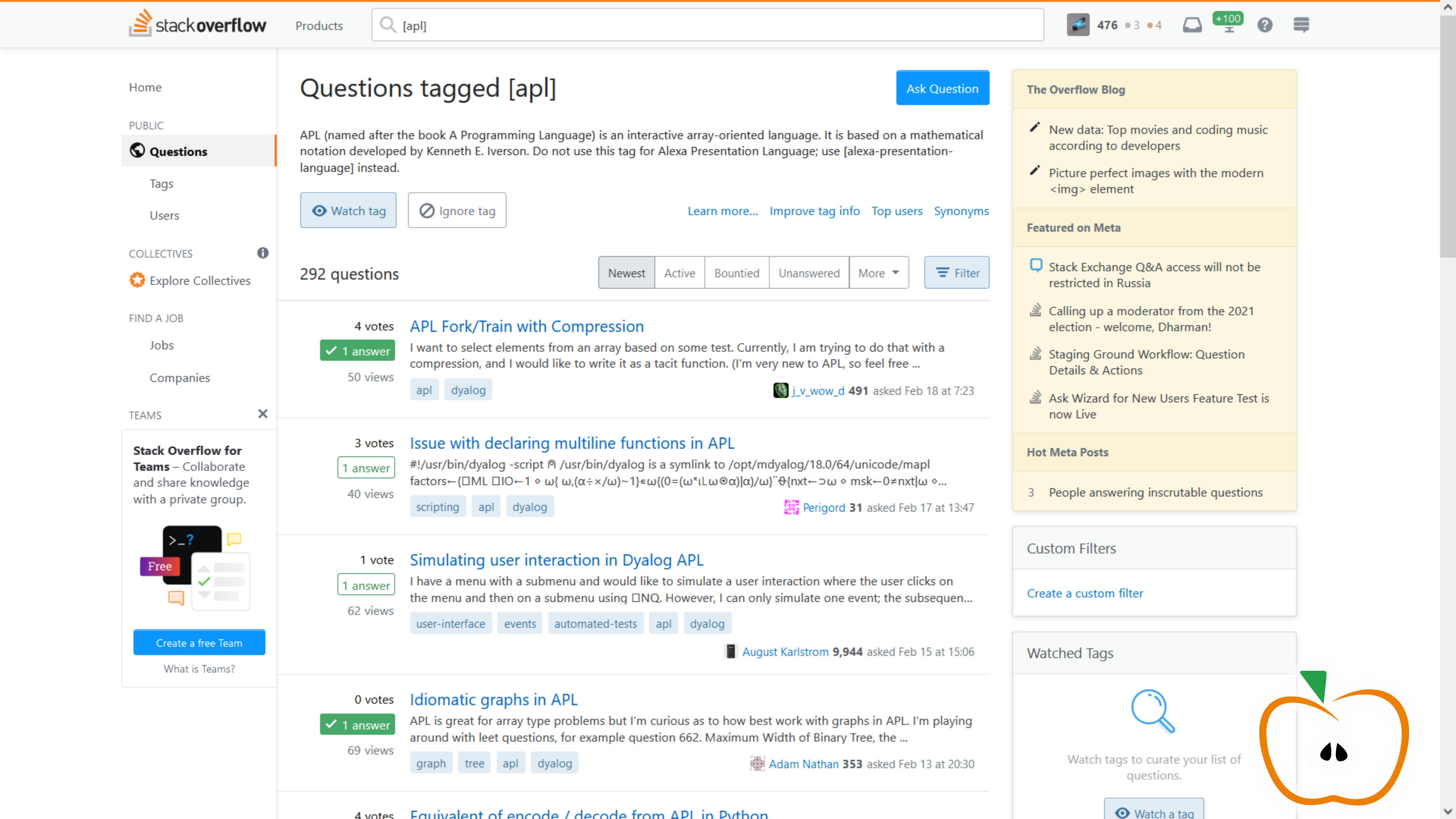
```
          84 - 12 - 1 - 13 - 28 - 9 - 6 - 15
70
```

Answers

2. Pyramid Schemes

   a. Sugar cubes are stacked in an arrangement as shown by **Figure 1**.





Stack Overflow for Teams – Collaborate and share knowledge with a private group.

Questions tagged [apl]

APL (named after the book A Programming Language) is an interactive array-oriented language. It is based on a mathematical notation developed by Kenneth E. Iverson. Do not use this tag for Alexa Presentation Language; use [alexa-presentation-language] instead.

292 questions

4 votes
✓ 1 answer
50 views

APL Fork/Train with Compression

I want to select elements from an array based on some test. Currently, I am trying to do that with a compression, and I would like to write it as a tacit function. (I'm very new to APL, so feel free ...

apl dyalog

j_v_wow_d **491** asked Feb 18 at 7:23

3 votes
1 answer
40 views

Issue with declaring multiline functions in APL

#!/usr/bin/dyalog -script ⍝ /usr/bin/dyalog is a symlink to /opt/mdyalog/18.0/64/unicode/mapl factors←{⎕ML ⎕IO←1 ⋄ ω{ ω,(α÷×/ω)~1}∊ω{(0=(ω*ι⌊ω⍟α)|α)/ω}¨θ{nxt←⊃ω ⋄ msk←0≠nxt|ω ⋄...

scripting apl dyalog

Perigord **31** asked Feb 17 at 13:47

1 vote
1 answer
62 views

Simulating user interaction in Dyalog APL

I have a menu with a submenu and would like to simulate a user interaction where the user clicks on the menu and then on a submenu using ⎕NQ. However, I can only simulate one event; the subsequen...

user-interface events automated-tests apl dyalog

August Karlstrom **9,944** asked Feb 15 at 15:06

0 votes
✓ 1 answer
69 views

Idiomatic graphs in APL

APL is great for array type problems but I'm curious as to how best work with graphs in APL. I'm playing around with leet questions, for example question 662. Maximum Width of Binary Tree, the ...

graph tree apl dyalog

Adam Nathan **353** asked Feb 13 at 20:30

4 votes

Equivalent of encode / decode from APL in Python

The Overflow Blog

- New data: Top movies and coding music according to developers
- Picture perfect images with the modern <img> element

Featured on Meta

- Stack Exchange Q&A access will not be restricted in Russia
- Calling up a moderator from the 2021 election - welcome, Dharman!
- Staging Ground Workflow: Question Details & Actions
- Ask Wizard for New Users Feature Test is now Live

Hot Meta Posts

3 People answering inscrutable questions

Custom Filters

Create a custom filter

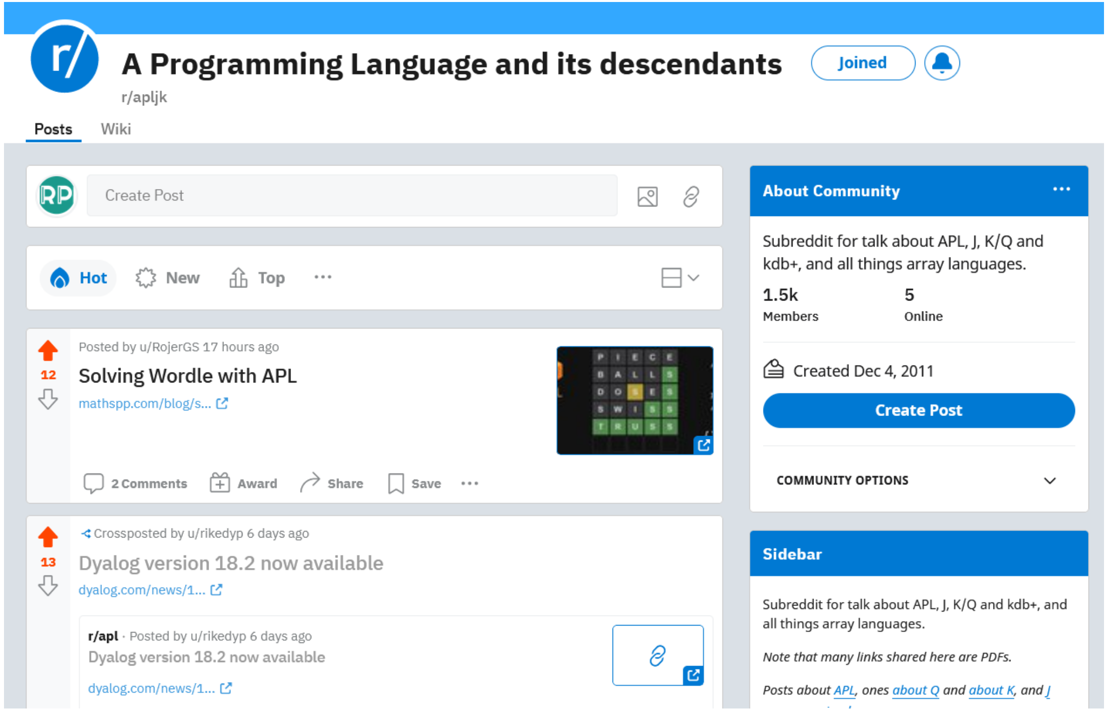
Watched Tags

Watch tags to curate your list of questions.

The APL Farm on Discord

discord.gg/cYbMMw5D

reddit.com/r/apljk



A Programming Language and its descendants

r/apljk

Posted by u/RojerGS 17 hours ago

12

Solving Wordle with APL

mathspp.com/blog/s...

Crossposted by u/rikedyp 6 days ago

13

Dyalog version 18.2 now available

dyalog.com/news/1...

r/apl · Posted by u/rikedyp 6 days ago

Dyalog version 18.2 now available

dyalog.com/news/1...

About Community

Subreddit for talk about APL, J, K/Q and kdb+, and all things array languages.

1.5k Members

5 Online

Created Dec 4, 2011

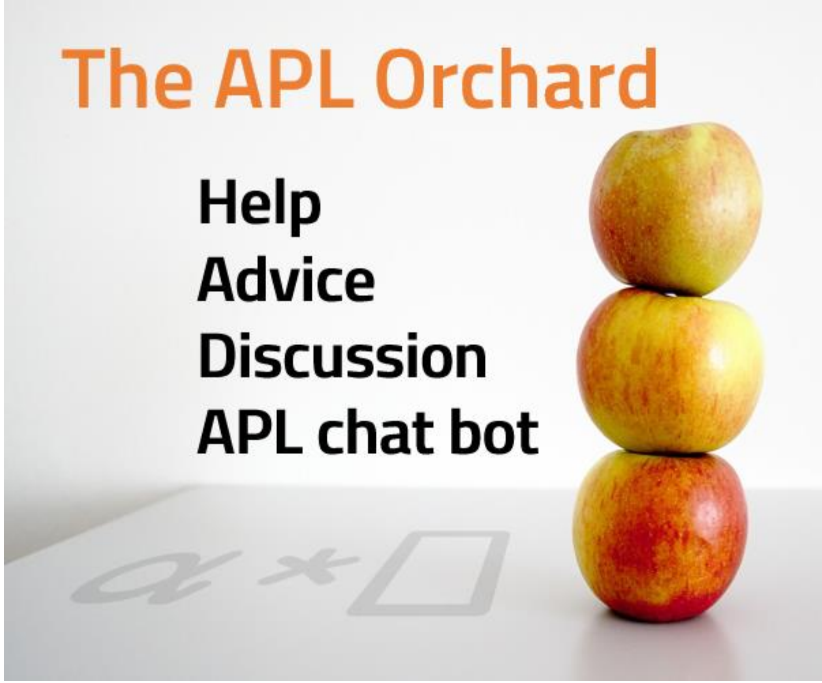
Sidebar

Subreddit for talk about APL, J, K/Q and kdb+, and all things array languages.

Note that many links shared here are PDFs.

Posts about APL, ones about Q and about K, and J

apl.chat



The APL Orchard

Help
Advice
Discussion
APL chat bot

New: Weekly APL Quest!

Fridays at 15:00 UTC

APL Problem Solving Competition

- Cash Prizes for Students (Total USD 6,500)
- Win free registration for Dyalog '22 Portugal
- Spread the word for a chance to win Referral Awards
- Meaty Problems

Start now at	contest.dyalog.com
Submit by	Friday 29 July 23:59 BST

Today's Programme

13:30	Easy to Learn, Worth Mastering **Rich Park**
14:10	What's a k-mer? **Stefan Kruger**
15:00	April: An APL Compiling to Common Lisp **Andrew Sengul**
16:00	The Array Cast **Conor Hoekstra et al**
17:00	Zoom Meetup Room ID: `825 178 58157` Passcode: `175914`

Links
apl.wiki/forums
apl.wiki/community