Cooking with Dyalog

Kai Jaeger & Stephen Taylor, Dyalog 16
“Stefano, what has computer science yet to learn from APL?”
“Nothing.”

Kai Jaeger & Stephen Taylor, Dyalog 16
“But it’s the best language for a programmer who wants to make a million dollars”
Dyalog applications

- make financial plans
- evaluate mortgage portfolios
- mature pension policies
- estimate pension-fund liabilities
- match DNA

... and enrich their authors
How do we get from here...
... to here?
1. Learn an analytical skill
2. Write programs that help you

```
2PG19APLPF[]V
[0] PG19APLPF;A S T B
[1] B='550555 -555, 555, 551.00'
[2] S=(T+LEDGERL;1); L' )/LEDGER
[3] A=(~T)/LEDGER
[5] '
[7] (T/A[;1]), (T/A[;2] 3]
[8] 2 1p'
[9] 'Report Q: Accounts Overdrawn More than $100.00'
[10] '
[12] (T/A[;1]), (T/A[;2] 3]
[13] 2 1p'
[14] 'Report R: Accounts with Balances over $1000.00'
[15] '
[16] (A[;3]>1000)A[;1,2]
[17] 2 1p'
[18] 'Report S: Unassigned Account Numbers'
[19] '
[20] ' ', S[;2]
```

Figure 4. Ledger Calculations Using Picture Formatting
2. Write programs that help you

```apl
PG19APLPF;A S T B
B←'550555 555,555,551.00'
S←(T←LEDGER;1)6 ' '(LEDGER
A←(~T)\LEDGER
'Report P: Account Balances Less Than $2.00' ' '
'T←A[;3]<2
'I/A[;1],B+T\A[;2 3]
'1p' ' '
'Report Q: Accounts Overdrawn More than $100.00' ' '
'T←A[;3]<100
'I/A[;1],B+T\A[;2 3]
'1p' ' '
'Report R: Accounts with Balances over $1000.00' ' '
'A[;3]>1000)\A[;1,2]
'1p' ' '
'Report S: Unassigned Account Numbers' ' '
'SC[;2]
```
3. Package them for others to use
4. License them to your competitors
The Four-Step Programme

- Acquire an analytical skill
- Write programs you can use
- **Package them for others to use**
- License them to your competitors
My lovely code

Slick techie stuff

```
VPG19APLPF7
0] PG19APLPF;A S T R
1] D='500000-555,555,551.00'
2] S=(T$LEDGER();1)+"')$LEDGER"
3] S=('~T+)/$LEDGER
7] (T/A[;1]),B*2/A[;2 3]
8] 2 'c' '
9] 'Report Q: Accounts Overdrawn More than $100.00' 10] '
12] (T/A[;1]),B*2/A[;2 3]
13] 2 'l' '
14] 'Report R: Accounts with Balances over $1000.00' 15] '
16] (A[;3]>1000)/A[;1,2]
17] 2 'l' '
20] '
```

Figure 4. Ledger Calculations Using Picture Formatting
THE
DYALOG
COOKBOOK
for Microsoft Windows™

by Kai Jaeger & Stephen Taylor

Kai Jaeger & Stephen Taylor, Dyalog 16
What goes in the pot?
What goes in the pot?

- Workspace structure
- Packaging an EXE
- Logging events
- Handling errors
- Configuration settings
- Test suites
- Build scripts
- Storing data
- Documentation
- User interfaces
- Help subsystem
- Installer
- Interprocess communications
- Managing source code
- …
What goes in the pot?
What goes in the pot?

The APLTree project: APL tools and utility library

All downloads are available at http://download.aplwiki.com

Platform dependency

Until August 2016 the APLTree project was pretty much a Windows-only project. That has changed now.

Overview

- APLTree is supposed to offer scripts (class scripts and namespace scripts) as well as full-blown applications for every-day-problems programmers typically come across.
- APLTree is an Open Source project - see AplTreeLicensing for details.
- Sub-projects of APLTree should follow certain suggestions - see AplTreeSuggestions for details.
What goes in the pot?

<table>
<thead>
<tr>
<th>Name</th>
<th>Windows</th>
<th>Linux</th>
<th>Mac</th>
<th>History</th>
<th>GitHub</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APLTreeUtils</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>General utilities; Included in many of the following scripts</td>
</tr>
<tr>
<td>DataAndTime</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Cast date and time, do math with date, calculate Easter, week no. and more</td>
</tr>
<tr>
<td>Execute</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Start a new process or application</td>
</tr>
<tr>
<td>FilesAndDirs</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Start a new process or application</td>
</tr>
<tr>
<td>HandleError</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Error trapping on a general level</td>
</tr>
<tr>
<td>IniFiles</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Dealing with INI files</td>
</tr>
<tr>
<td>Logger</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Writing Log Files</td>
</tr>
<tr>
<td>Markdown2Help</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>CHM-like functionality implemented in Dyalog</td>
</tr>
<tr>
<td>MarkAPL</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Converts Markdown to HTML5</td>
</tr>
<tr>
<td>OS</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Anything related to your operating system</td>
</tr>
<tr>
<td>ServiceState</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Interface between Windows Services and APL applications</td>
</tr>
<tr>
<td>SevenZip</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Interface to the Open Source zipper 7zip</td>
</tr>
<tr>
<td>ShowChmHelp</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Display help files and topics within help files</td>
</tr>
<tr>
<td>WindowsEventLog</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Dealing with the Windows Event Log</td>
</tr>
<tr>
<td>WinReg</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Dealing with the Windows Registry</td>
</tr>
<tr>
<td>WinSys</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔ Link</td>
<td>✔ Link</td>
<td>Get information closely related to Windows</td>
</tr>
</tbody>
</table>
What goes in the pot?

<table>
<thead>
<tr>
<th>Name</th>
<th>Windows</th>
<th>Linux</th>
<th>Mac</th>
<th>History</th>
<th>GitHub</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADOC</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>Link</td>
<td>Link</td>
<td>Automated Documentation Generation</td>
</tr>
<tr>
<td>CompareSimple</td>
<td>✔️</td>
<td>⭐️</td>
<td>⭐️</td>
<td>Link</td>
<td>Link</td>
<td>Compare fns, operators, scripts, scripted namespaces and files</td>
</tr>
<tr>
<td>Compare</td>
<td>✔️</td>
<td>⭐️</td>
<td>⭐️</td>
<td>Link</td>
<td>Link</td>
<td>Inherits from CompareSimple and more like Merge</td>
</tr>
<tr>
<td>Fire</td>
<td>✔️</td>
<td>⭐️</td>
<td>⭐️</td>
<td>Link</td>
<td>Link</td>
<td>Search and Replace tool on workspace level</td>
</tr>
<tr>
<td>Launchy</td>
<td>✔️</td>
<td>⚹️</td>
<td>⚹️</td>
<td>Link</td>
<td>Link</td>
<td>Launch an instance of Dyalog APL/W</td>
</tr>
<tr>
<td>LogDog</td>
<td>✔️</td>
<td>⭐️</td>
<td>⭐️</td>
<td>Link</td>
<td>Link</td>
<td>Watch a (constantly changing) log file.</td>
</tr>
<tr>
<td>Tester</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>Link</td>
<td>Link</td>
<td>Testing framework</td>
</tr>
</tbody>
</table>

Legend:
- 🚨 contributor needed
- ✔️ fully implemented
- ⭐️ under construction
- ⚹️ not applicable
- ⚹️ beta version
- ⚹️ currently not planned

Suspended projects
Projects may get suspended for different reasons, for example due to the lack of demand or contributors or both.
Show suspended projects

APLTree downloads

Kai Jaeger & Stephen Taylor, Dyalog 16
What goes in the pot?
Whose cooking?
Credentials

Kai Jaeger & Stephen Taylor, Dyalog 16
Credentials

“Between us we encompass all human knowledge.”

Kai Jaeger & Stephen Taylor, Dyalog 16
All human knowledge

“Mr Wells knows everything that can be known.”

Kai Jaeger & Stephen Taylor, Dyalog 16
All human knowledge

“And I know the rest.”

Kai Jaeger & Stephen Taylor, Dyalog 16
Good as that is...
You know other solutions
Doubtless better ones
We know you’ll contribute
The Dyalog Cookbook

Stephen Taylor and Kai Jaeger

Interested in this book? Show your support by saying what you’d like to pay for it:

Name

Email

Also share your email address with the authors.

I’d buy it for...

$  

Notify Me When This Is Published
A Leanpub book

leanpub.com/thedyalogcookbook

Kai Jaeger & Stephen Taylor, Dyalog 16
In this context the `Trap` structure has an advantage over `TRAP`. When it fires, and control advances to its `Else` fork, the trap is immediately cleared. So there is no need explicitly to reset the trap to avoid an open loop.

The handling of error codes and messages can easily obscure the rest of the logic. Clarity is not always easy to find, but is well worth working for. This is particularly true where there is no convenient test for an error, only a trap for when it is encountered.
... and in Dyalog scripts

Kai Jaeger & Stephen Taylor, Dyalog 16
Collaborate through GitHub

github.com/5jt/dyalog-cookbook

Kai Jaeger & Stephen Taylor, Dyalog 16
License your competitors
Live the dream
Join the British APL Association
Contribute!

#cookbook at theaplroom.slack.com
github.com/5jt/dyalog-cookbook
leanpub.com/thedyalogcookbook

kai@aplteam.com
sjt@5jt.com