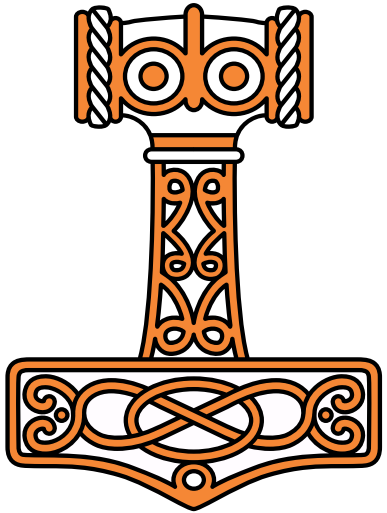
DYALOG

Olhão 2022

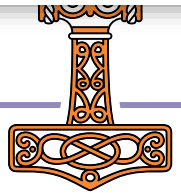
Recent Language Features



Rich Park, Rodrigo Girão Serrão

12.0	2008	August	Unicode support (⎕AVU , ⎕UCS), ⎕FCOPY , ⎕FPROPS
12.1	2009	November	I-beam (⌶), Table (⍪), ⎕XML , ⎕FCHK , User commands
13.0	2011	April	Left (⊣), Right (⊢), Variant (⍠), ⎕OPT , ⎕R , ⎕S , ⎕PROFILE , ⎕RSI , complex number and decimal float support, short arguments for Take, Drop, and Index (↑ , ↓ , ⌷)
13.1	2012	April	⎕DMX , ⎕FHIST
13.2	2013	January	Array Editor
14.0	2014	June	Trains, Tally (≢), Key (⌸), Rank operator (⍤), high-rank Index Of, multi-threading with futures and isolates
14.1	2015	June	:Disposable .NET objects and resources, gesture support, many new I-beams
15.0	2016	June	⎕MKDIR , ⎕NDELETE , ⎕NEXISTS , ⎕NGET , ⎕NINFO , ⎕NPARTS , ⎕NPUT
16.0	2017	June	At (@), Interval Index (⍸), Where (⍸), Nest (⊆), Partition (⊆), Stencil (⌺), ⎕JSON , ⎕CSV
17.0	2018	July	⎕NCOPY , ⎕NMOVE , total array ordering, high-rank Unique
17.1	2019	October	Duplicates in Interval Index (⍸) look-up array
18.0	2020	June	Atop (⍤), Over (⍥), Constant (⍨), Unique Mask (≠), duplicates from Where (⍸), empty partitions from Partitioned Enclose (⊂), date-time conversion (⎕DT), case folding/mapping (⎕C), launching with text source file, .NET Core support
18.2	2022	March	⎕ATX , shell scripting

Primitives [edit]



DYALOG

Language Features of version 18.0 in Depth

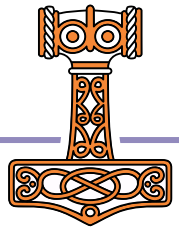
Adám Brudzewsky

New

`⎕C`	Case convert
`f⍤⍥g`	Over
`f⍥⍤g`	Atop
`≠Y`	Unique mask
`A⍨⍨`	Constant
`⎕DT`	Date-time
`1200⌶`	Format date-time

Improved

```
⎕JSON⍠'HighRank'
⎕JSON⍠'Dialect'
⎕R/⎕S⍠'Regex'
⎕NPUT⍠'NEOL'
⍳Y
X⊂Y
↑[k]Y
```



Dyalog version 18 language features

Primitive operators

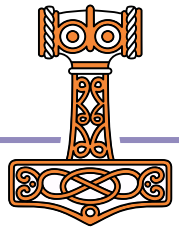
⍤ ⍥ ⍣

Primitive functions

≠ ⍳ ⊂

System functions

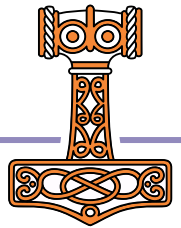
⎕C ⎕DT 1200⌶ ⎕JSON ⎕R/⎕S ⎕ATX



Primitive operators

Function composition apl.wiki/Function_composition

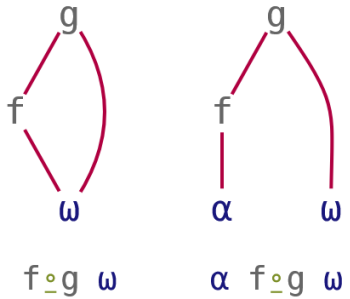
Composition	Notation	Monadic	Dyadic
Beside	`F∘G`	`F G ⍵`	`⍺ F G ⍵`
Atop	`F⍤1 G`	`F G ⍵`	`F ⍺ G ⍵`
Over	`F⍥G`	`F G ⍵`	`(G ⍺)F (G ⍵)`
Fork	`(FGH)`	`(F ⍵) G (H ⍵)`	`(⍺ F ⍵)G(⍺ F ⍵)`
Behind	`F⍛G`	`(F ⍵) G ⍵`	`(F ⍺) G ⍵`



View>Master Views>Slide Master then [PgUp] to edit title!

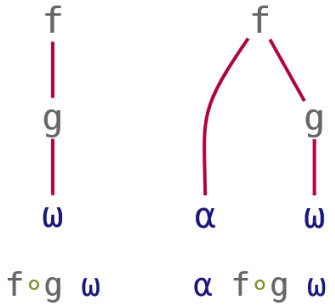
Function composition

Composition	Notation	Monadic	Dyadic
Beside	`F∘G`	`F G ⍵`	`⍺ F G ⍵`
Behind	`F⍤G`	`(F ⍵) G ⍵`	`(F ⍺) G ⍵`



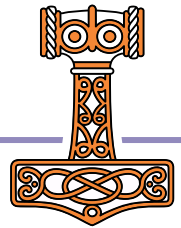
Pre-process right argument

```
2⊃∘⎕VFI¨ '3 4.2 and 5' '6 7' '12 more'
```



Pre-process left argument

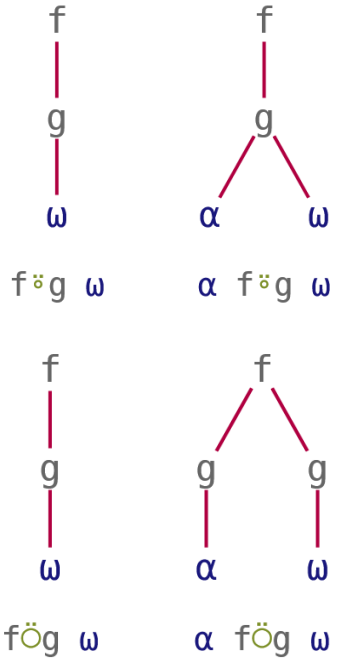
```
array ⍴⍤⍴ values
```



View>Master Views>Slide Master then [PgUp] to edit title!

Function composition

Composition	Notation	Monadic	Dyadic
Atop	F⍤G	F G ⍵	F ⍺ G ⍵
Over	F⍥G	F G ⍵	(G ⍺)F (G ⍵)



Post-process result

```
3 4 5 ⌊⍤÷ 7 2 9
```

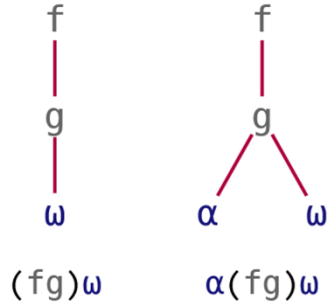
Pre-process both arguments

```
≠⍤,
+/⍤≠
```


View>Master Views>Slide Master then [PgUp] to edit title!

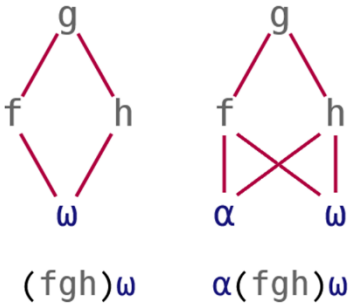
Function composition

Composition	Notation	Monadic	Dyadic
Atop	`(FG)`	`F G ω`	`F α G ω`
Fork	`(FGH)`	`(Fω)G(Hω)`	`(αFω)G(αHω)`



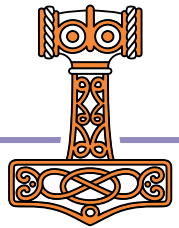
Post-process result

```
3 4 5 (⌊÷) 7 2 9
```



Pre-process both arguments

```
5 2 3.2 8 (≠ö, ≡ +/ö≠) 'ABCD'
```



View>Master Views>Slide Master then [PgUp] to edit title!

Primitive operators

Function composition

Pre-process right argument

```
2⊃∘⍋VFI¨ '3 4.2 and 5' '6 7' '12 more'
```

Pre-process left argument

```
array ⍴⍤⍴ values
```

Pre-process both arguments

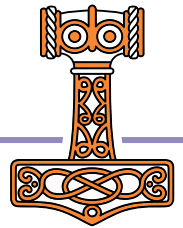
```
vec1 (≠⍤,≡+/⍤≠) vec2
```

Post-process result

```
3 4 5 ⌊⍤÷ 7 2 9
```

Pre-process separately

```
x⍉⍤(+.×)∘÷y
```

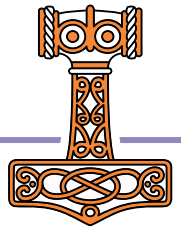


Constant `A≍`

Lightweight notation

Train `{A}×h`

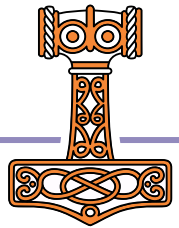
At `{A}@h`



Constant `A≍`

Lightweight notation

Train	`{A}×h`	`A×h`
At	`{A}@h`	`A@h`
Constant	`{A}`	



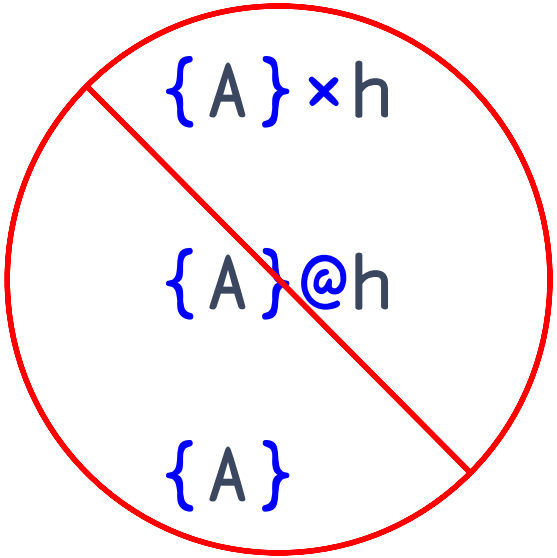
Constant $A^{\simeq}$

Lightweight notation

Train $\{A\} \times h$ $A \times h$

At $\{A\} @ h$ $A @ h$

Constant $\{A\}$ $A^{\simeq}$



Constant A⍨

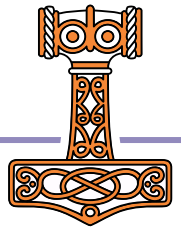
Lightweight notation

```
      3 5ρ⎕A
ABCDE
FGHIJ
KLMNO
      'jk'⍨⍤3 5ρ⎕A
```

jk	jk	jk	jk	jk
jk	jk	jk	jk	jk
jk	jk	jk	jk	jk

```
      'jk'ρ⍨ρ3 5ρ⎕A
jkjkj
kjkjk
jkjkj
      'jk'ρ∘⊂⍨ρ3 5ρ⎕A
```

jk	jk	jk	jk	jk
jk	jk	jk	jk	jk
jk	jk	jk	jk	jk

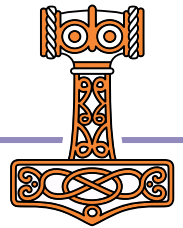


View>Master Views>Slide Master then [PgUp] to edit title!

Constant A⍨

Avoid ugly work-arounds

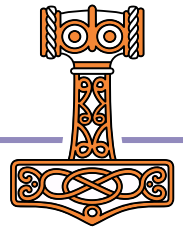
```
      mask←1 0 0 0 1 0 0 ⋄ data←'AbcdEfg'

      '□'@{mask}data
□bcd□fg

      (mask/data)←'□' ⋄ data
□bcd□fg
```



Constant A⍨

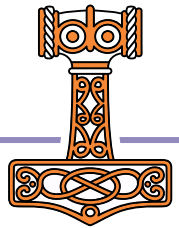
Avoid ugly work-arounds

```
      mask←1 0 0 0 1 0 0 ⋄ data←'AbcdEfg'

      mask{'⍰'@{⍺}⍵}data
VALUE ERROR
      mask{'⍰'@{⍺}⍵}data
                 ^
      mask{'⍰'@(⍺⍨)⍵}data
⍰bcd⍰fg
```



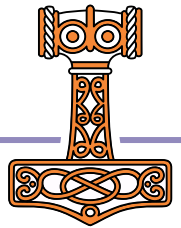
Exercises

https://is.gd/MXvf9r



Primitive Functions

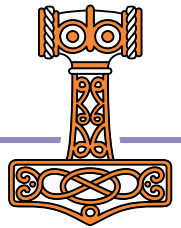
Unique mask	`≠⍵`
Where	`⍸⍵`
Partitioned enclose	`⍺⊂⍵`



Unique mask ≠⍵

a.k.a. nub-sieve

```
      ∪'Mississippi'
Misp

      {↑⍵(≠⍵)}'Mississippi'
M i s s i s s i p p i
1 1 1 0 0 0 0 0 1 0 0
```

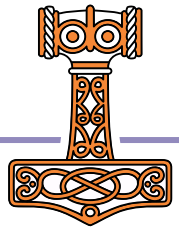


Why, though?

≠Y is to ∪Y

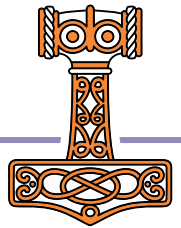
as

⍋Y is to Sort Y



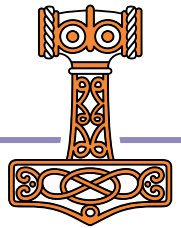
⍋Y vs Sort Y

```
      Sort 3 1 4 1 5
1 1 3 4 5
      ⍋ 3 1 4 1 5
2 4 1 3 5
      'Moses'[2 4 1 3 5]
oeMss
```



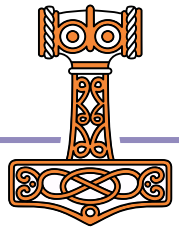
≠Y vs ∪Y

```
      ∪ 3 1 4 1 5
3 1 4 5
      ≠ 3 1 4 1 5
1 1 1 0 1
      1 1 1 0 1 / 'Moses'
Moss
```



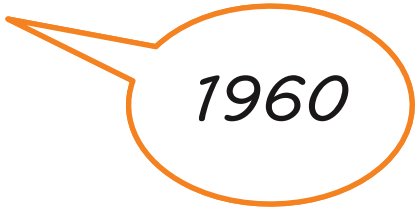
Where $\underline{\iota}\,\omega$

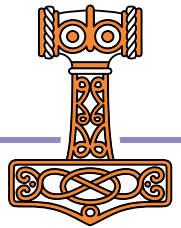
Now accepts non-negative integers (not just Bool!)



History

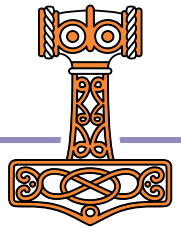
```
      PRICE←71 82 81 82 84 59
      (75≤PRICE)/PRICE
82 81 82 84
      (75≤PRICE)/⍳⍴PRICE
2 3 4 5
```





History

```
      PRICE←71 82 81 82 84 59
      (75≤PRICE)/PRICE
82 81 82 84
      (75≤PRICE)/⍳⍴PRICE
2 3 4 5
      ⍸75≤PRICE
2 3 4 5
```

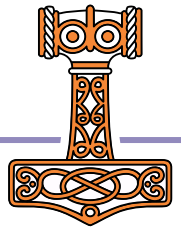


Selection

Using Where

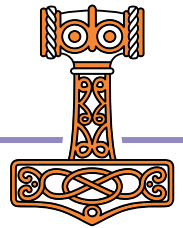
Use case: selection

```
      fruit←'Apple' 'Banana' 'Cherry' 'Date' 'Elderberry'
      select←1 1 0 1 0
      select/fruit
 Apple  Banana  Date
      select/⍳⍴select
1 2 4
      ⍸select
1 2 4
```



Use case: selection

```
      fruit←'Apple' 'Banana' 'Cherry' 'Date' 'Elderberry'
      select←1 1 0 1 0
      select/fruit
Apple  Banana  Date
      fruit[⍸select]
Apple  Banana  Date
```

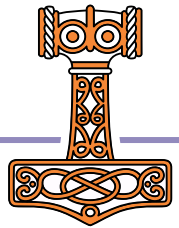


Use case: multi-selection

```
      fruit←'Apple' 'Banana' 'Cherry' 'Date' 'Elderberry'
      select←1 2 0 1 0
      select/fruit
Apple  Banana  Banana  Date
      fruit[⍸select]
      ⍸ ERROR
      fruit[⍸select]
      ∧
```

1980

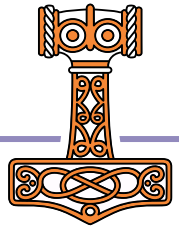
17·1



Use case: multi-selection

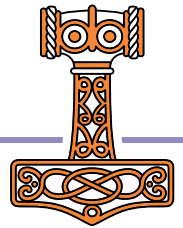
```
      fruit←'Apple' 'Banana' 'Cherry' 'Date' 'Elderberry'
      select←1 2 0 1 0
      select/fruit
Apple  Banana  Banana  Date
      fruit[⍸select]
      Banana  Banana  Date
```

18·0



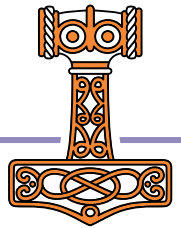
Use case: multi-dimensional selection

```
      spice←'Anise' 'Basil' 'Chili' 'Dill' 'Epazote'
      □←stuff←↑fruit spice
 Apple  Banana  Cherry  Date  Elderberry
 Anise  Basil   Chili   Dill  Epazote
      □←select←↑select (0 0 0 2 0)
1 2 0 1 0
0 0 0 2 0
```



Use case: multi-dimensional selection

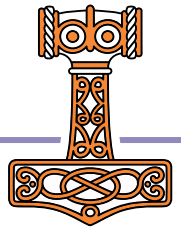
```
      stuff[⍳select]
 Apple  Banana  Banana  Date  Dill  Dill


      select/stuff
RANK ERROR
      select/stuff
           ^
```



Representing a set

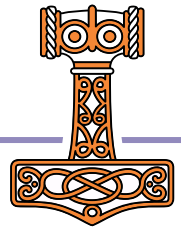
Using Where

ι



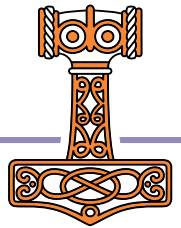
Use case: Representing a set

```
      all     ← 'a' 'b' 'c' 'd' 'e' 'f'
      mask    ←  1   0   0   1   0   1
      indices ←  1           4       6
      indices ≡ ⍳ mask
1
```



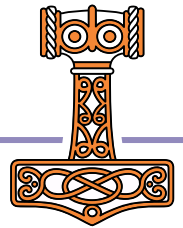
Use case: Representing a multi-set

```
      all     ← 'a' 'b' 'c' 'd' 'e' 'f'
      count   ←  1   0   0   3   0   2
      indices ←  1           4 4 4     6 6
      indices ≡ ⍳ count
1
      count   ≡ ?   indices
1
```



Use case: Representing a multi-set

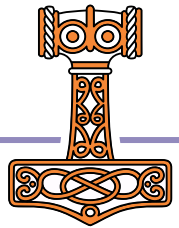
```
      all     ← 'a' 'b' 'c' 'd' 'e' 'f'
      count   ←  1   0   0   3   0   2
      indices ←  1           4 4 4     6 6
      indices ≡ ⍳ count
1

      count   ≡⍳⌸¯1⊢indices
1
```



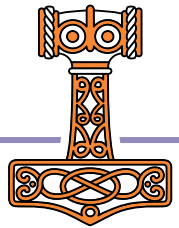
Partitioned enclose `α⊂ω`

Now accepts non-negative integers (not just Bool!)

Can take a short left argument

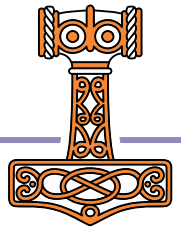


```
cutoffs ← 0  20  40  60  80  100
values  ← 3  14  15  35  65  89  92  793
```



```
      cutoffs ← 0  20 40 60  80   100
      values  ← 3 14 15 35  65 89 92 793
      cutoffs ⍸ values
1 1 1 2 4 5 5 6
      values⊂⍨1,¯2-/cutoffs⍸values
```

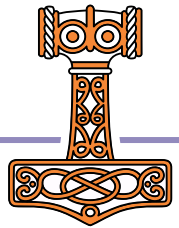
3 14 15	35		65	89 92	793



```
      1 1 ⊂ 'head' 'and' 'the' 'rest'
LENGTH ERROR
      1 1⊂'head' 'tail' 'and' 'the' 'rest'
         ^
      1 1 ⊂ 'head' 'and' 'the' 'rest'
```

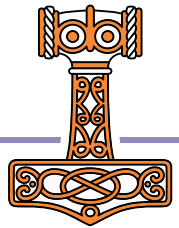
Exercises

https://is.gd/jTKznr



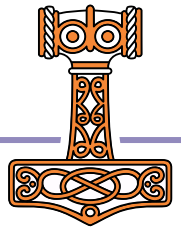
System Functions

□C □DT 1200I □JSON □ATX



Case Convert □C

□C □DT 1200□I □JSON □R/□S □ATX



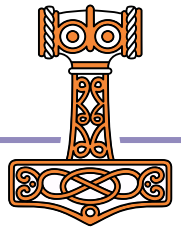
Wait, what?

Uppercase: `1(819⌶)Y` ⇨ `1 ⎕C Y`

Lowercase: `819⌶Y` ⇨ `⎕C Y`

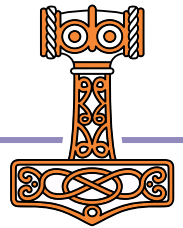
Lowercase: `0(819⌶)Y` ⇨ `⎕C Y`

```
Big←{⍺←0
      ⍺:1 ⎕C ⍵
      ⎕C ⍵}
```



Pain without gain?

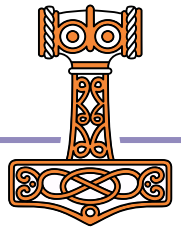
```
      819⌶'Hi'#(3J14 'PI')
DOMAIN ERROR: Invalid right argument
      819⌶'Hi'#(3J14 'PI')
          ^
      ⎕C'Hi'#(3J14 'PI')
 hi  #  3J14  pi
```



Pain without gain?

```
      + 'Hi'#(3J14 'PI')
Hi  #  3J¯14  PI
```

```
      □C'Hi'#(3J14 'PI')
hello  #  3J14  pi
```



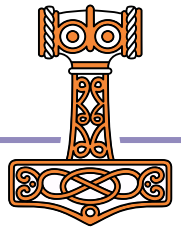
Case Convert

Monadic ⎕C: Case **Fold**

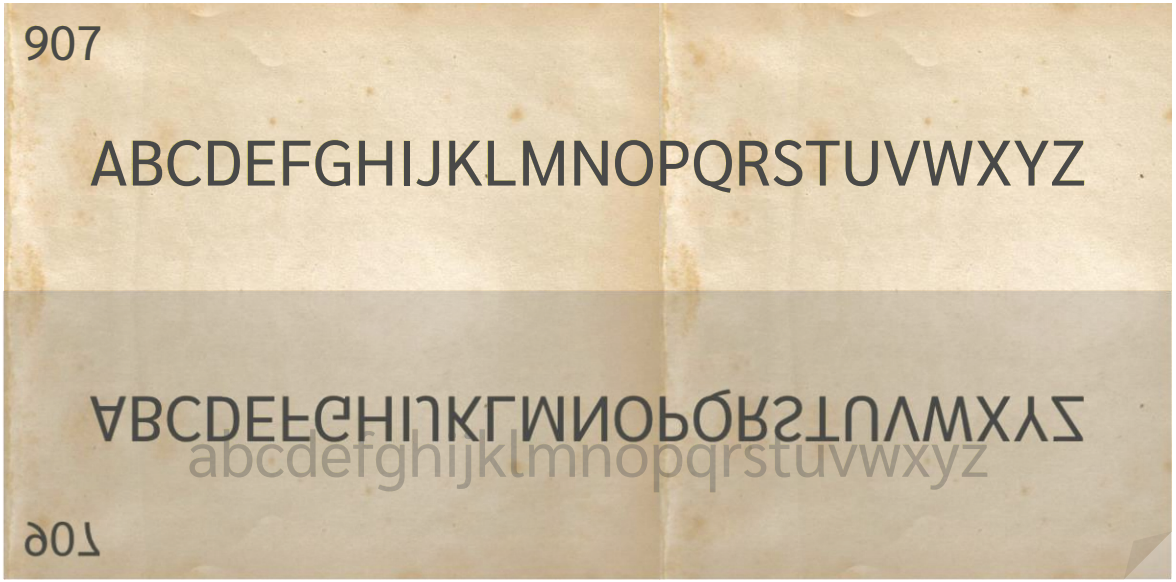
normalisation
for machine comparison

Dyadic ⎕C: Case **Map**

display form
for human readers



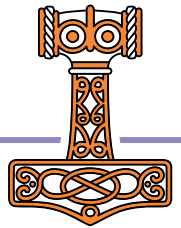
Case Folding: C Y



View>Master Views>Slide Master then [PgUp] to edit title!

Case Mapping: X □C Y

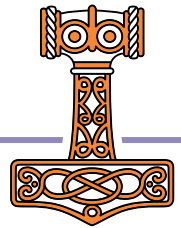
1 : Upper		Origin		¯1 : Lower
A	⇦	A|a	⇨	a
B	⇦	B|b	⇨	b
C	⇦	C|c	⇨	c
D	⇦	D|c	⇨	d
E	⇦	E|e	⇨	e
F	⇦	F|f	⇨	f
G	⇦	G|g	⇨	g



View>Master Views>Slide Master then [PgUp] to edit title!

Folding vs Mapping

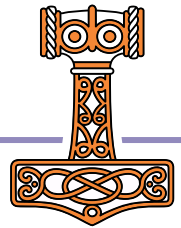
```
      'Μωυσής'
Μωυσής
      ⍺⍺C'Μωυσής'    ⍝ fold
μωυσήσ
      ¯1⍺⍺C'Μωυσής'  ⍝ map
μωυσής
```



View>Master Views>Slide Master then [PgUp] to edit title!

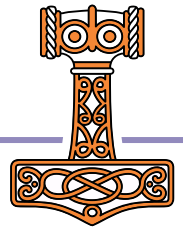
Folding vs Mapping

```
      ⎕C 'Μωυσής' 'ΜΩΥΣΉΣ'     ⍺ fold
μωυσήσ  μωυσήσ
      ¯1 ⎕C 'Μωυσής' 'ΜΩΥΣΉΣ' ⍺ map
μωυσής  μωυσήσ
```



Folding vs Mapping

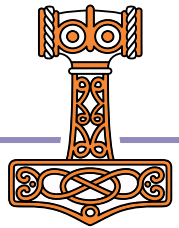
```
      ⎕C 'Μωυσής' 'ΜΩΥΣΎΗΣ'      ⍝ fold
μωυσήσ  μωυσήσ
      1 ⎕C 'Μωυσής' 'ΜΩΥΣΎΗΣ'    ⍝ map
ΜΩΥΣΎΗΣ  ΜΩΥΣΎΗΣ
```



Folding vs Mapping

"Street" in German

```
      ⎕C 'Straße' 'STRAßE'     ⍝ fold
straße  straße
      1 ⎕C 'Straße' 'STRAßE'   ⍝ map
STRAßE  STRAßE


STRASSE  STRASSE
```



Would you ever map?

- All-lowercase to generate URLs or hash-tags

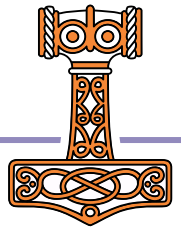
  `'Ο Μωυσής Ζει' ⇨ '#ομωυσήςζει'`

- All-caps for display purposes

  `'Sale' ⇨ 'S A L E'`

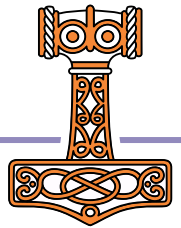
- Title-case a heading…

  `'Would you ever map?' ⇨`
  `'Would You Ever Map?'`



View>Master Views>Slide Master then [PgUp] to edit title!

```
      t←'Would you ever map?'
      ¯1*0,¯1↓' '≠t
1 ¯1 ¯1 ¯1 ¯1 ¯1 1 ¯1 ¯1 ¯1 1 ¯1 ¯1 ¯1 ¯1 1 ¯1 ¯1 ¯1
      (¯1*0,¯1↓' '≠t)⌷C t
DOMAIN ERROR: Invalid left argument
      (¯1*0,¯1↓' '≠t)⌷C t
                     ^
      (¯1*0,¯1↓' '≠t)⌷C¨t
Would You Ever Map?
```

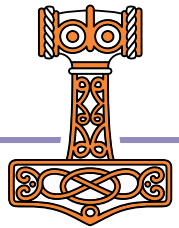


Date times

DT

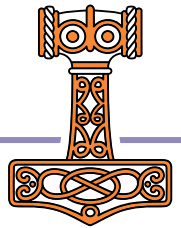
Is it Christmas Yet? – **Richard Smith**, *Dyalog '19*

dyalog.tv/Dyalog19/?v=SVcNgQewYNY



Date-time

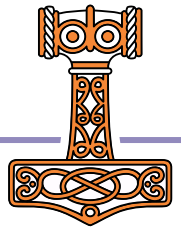
Timestamp
Time number
Military time zone



Timestamp

year month day... ms 2020 6 11 16 0 0 0 ← WEST — TS

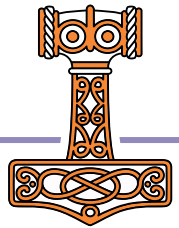
year week weekday... µs 2020 24 4 16 0 0 0



Time number

days since 1899-12-31 00:00	43992.70833
seconds since 1970-01-01 00:00	1591894800

View>Master Views>Slide Master then [PgUp] to edit title!



View>Master Views>Slide Master then [PgUp] to edit title!

Time numbers

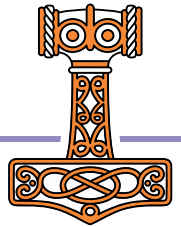
1	**Dyalog day number**
2	Dyalog component file
10	J nanoseconds
11	Shakti K milliseconds
12	JavaScript/D/Q ms
13	R chron format
20	Unix time
30	MS-DOS date/time
31	MS-Win32 FILETIME

et cetera ad abundantiam

Timestamps

¯1	**⎕TS-style: year month... ms**
¯2	Like ⎕TS but μs replacing ms
¯3	Like ⎕TS but ns replacing ms
¯10	ISO year day hour min sec μs
¯11	ISO year week weekday... μs

etc.



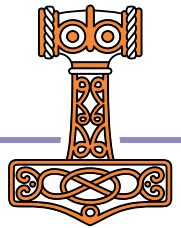
□DT syntax

Conversion

```
          outCode □DT dateTimes
   inCode outCode □DT dateTimes
```

Validation

```
          0 □DT dateTimes
   inCode 0 □DT dateTimes
```



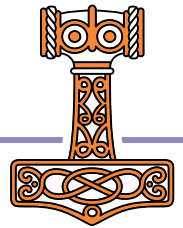
⎕DT syntax: one-element left argument

Conversion: ⎕TS-style timestamp to Unix time

```
      20 ⎕DT ⊂ 2020 06 11  16 00 00 000
1591891200
```

Validation: Leap year check

```
      0 ⎕DT ⊂ 1900 02 29
0
```



⎕DT syntax: one-element left argument

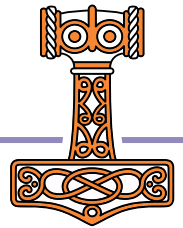
Conversion: Current ⎕TS-style UTC time

```
      ¯1 ⎕DT 'Z'
 2020 6 11 16 0 0 0
```

What time zone am I in?

```
      3600÷⍨-/20 ⎕DT 'JZ'
1
```

← WEST=UTC+1



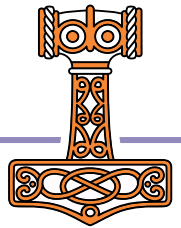
□DT syntax: two-element left argument

Conversion: Unix time to □TS-style timestamp

```
      20 ¯1 □DT 1591891200
 2020 6 11 16 0 0 0
```

Validation: Leap year check

```
      60 0 □DT 19000229
0
```



Wikidata item

Languages

한국어

Edit links

Print/export

Download as PDF

Printable version

11, 2006" in MDY, and "2011 November 6" in YMD.

The ISO 8601 format YYYY-MM-DD (2020-06-04) is intended to harmonize these formats and ensure accuracy in all situations. Many countries have adopted it as their sole official date format, though even in these areas writers may adopt abbreviated formats that are no longer recommended.

Table coding [edit source]

Basic components of a calendar date for the most common calendar systems:

Y – year
M – month
D – day

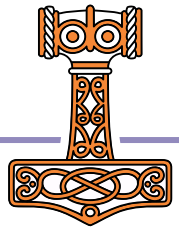
Specific formats for the basic components:

yy – two-digit year, *e.g. 06*
yyyy – four-digit year, *e.g. 2006*
m – one-digit month for months below 10, *e.g. 4*
mm – two-digit month, *e.g. 04*
mmm – three-letter abbreviation for month, *e.g. Apr*
mmmm – month spelled out in full, *e.g. April*
d – one-digit day of the month for days below 10, *e.g. 2*
dd – two-digit day of the month, *e.g. 02*

1200I syntax

Format one or more date-times

```
'YYYY-MM-DD' (1200I) dyalogDateNumbers
```

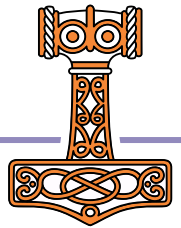


1200I syntax: patterns

Format one or more date-times

```
'YYYY-MM-DD' (1200I) dyalogDateNumbers
```

'M'	'6'
'MM'	'06'
'MMM'	'JUN'
'MMMM'	'JUNE'

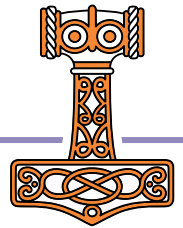


1200I syntax: numbers

Format one or more date-times

```
'YYYY-MM-DD' (1200I) dyalogDateNumbers
```

```
'M'          '6'
'MM'         '06'
'_M'         ' 6'
```

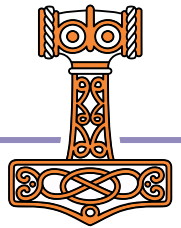


1200I syntax: names

Format one or more date-times

```
'YYYY-MM-DD' (1200I) dyalogDateNumbers
```

'MMMM'	'JUNE'
'__en__MMMM'	'JUNE'
'__ru__MMMM'	'ИЮНЬ'
'__fr__MMMM'	'JUIN'

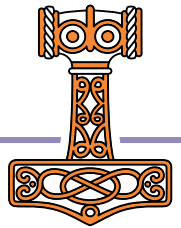


1200I syntax: names

Format one or more date-times

```
'YYYY-MM-DD' (1200I) dyalogDateNumbers
```

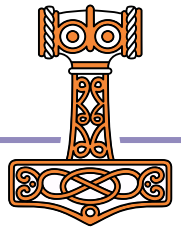
```
'mmmm'      'june'
'Mmmm'      'June'
'MMMM'      'JUNE'
```



1200I syntax: names

Format one or more date-times

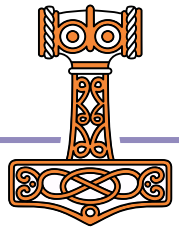
```
'YYYY-MM-DD' (1200I) dyalogDateNumbers

                __en__      __fr__

'mmmm'        'june'       'juin'
'Mmmm'        'June'       'Juin'
'MMMM'        'JUNE'       'JUIN'
```



1200I syntax: names

Format one or more date-times

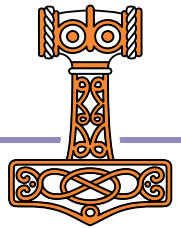
```
'YYYY-MM-DD' (1200I) dyalogDateNumbers

          __en__     __fr__
'_mmm'    'June'     'juin'
'mmmm'    'june'     'juin'
'Mmmm'    'June'     'Juin'
'MMMM'    'JUNE'     'JUIN'
```



1200I syntax: ordinals

Format one or more date-times

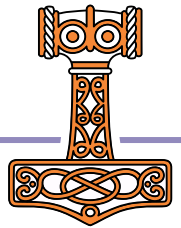
```
'YYYY-MM-DD' (1200I) dyalogDateNumbers

              __en__      __fr__
'D'           '1'         '1'
'Doo'         '1st'       '1er'
'DD'          '11'        '11'
'DDoo'        '11th'      '11'
```



1200I syntax: ordinals

Format one or more date-times

```
'YYYY-MM-DD' (1200I) dyalogDateNumbers

              __en__      __da__
'D'           '1'         '1'
'Doo'         '1st'       '1.'
'DD'          '11'        '11'
'DDoo'        '11th'      '11.'
```

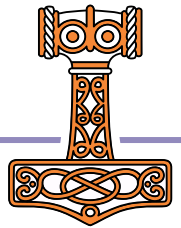


1200⌶ syntax: 12/24 hours

Format one or more date-times

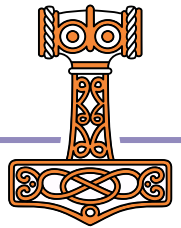
```
'hh:mm' (1200⌶) dyalogDateNumbers
```

```
'h'          '16'
't'          '4'
't pp'       '4 pm'
'tPP'        '4PM'
```



1200I examples

```
      'DDoo Mmmm YYYY "at" hh:mm:ss.fff'
11th June 2020 at 16:00:00.000
      '__da__Dddd, D. mmmm "'"YY'
Torsdag, 11. juni '20
      '%ISO%'
2020-06-11T16:00:00
```

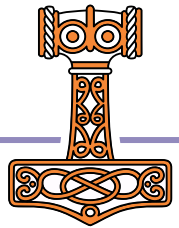


JSON Convert

⎕JSON

Webinar, part 3

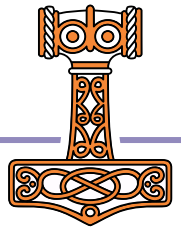
```
⎕JSON⍠'HighRank'
⎕JSON⍠'Dialect'
⎕JSON ⊂tables
```



Converting rank>1 arrays to JSON

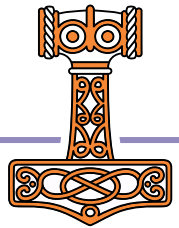
no need for ↑ and ↓ pre/post-processing

```
⎕JSON⍠'HighRank'
```



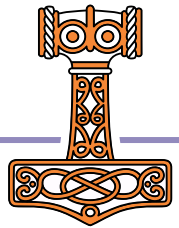
Ever tried this?

```
      data←2 3⍴⍳6
      ⎕JSON data
DOMAIN ERROR: JSON export: the right argument ca
      ⎕JSON data
      ^
```



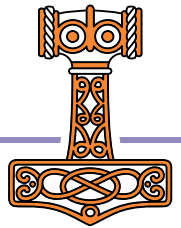
Ever tried this?

```
      data←2 3⍴⍳6
      ⎕JSON↓data
[[1,2,3],[4,5,6]]
```



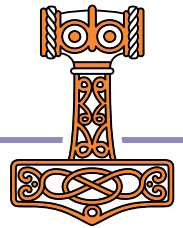
Ever tried this?

```
      data←2 3 4⍴⍳24
      ⎕JSON↓data
DOMAIN ERROR: JSON export: the right argument ca
      ⎕JSON data
      ^
```



Ever tried this?

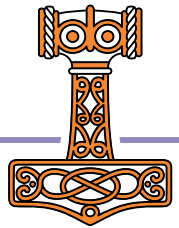
```
      data←2 3 4⍴⍳24
      ⎕JSON↓↓data
[[[1,2,3,4],[5,6,7,8],[9,10,11,12]],[[13,14,15,1
```



Ever tried this?

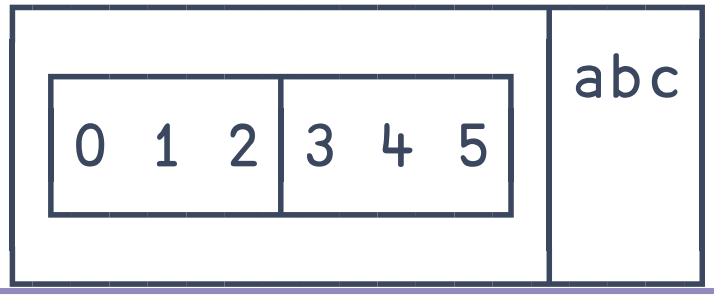
```
data←(2 3⍴⍳6) 'abc'
⎕JSON↓⍤⍉⍤(¯1+≢⍴data)⊢data

↓⍤⍉⍤(¯1+≢⍴data)⊢data
```

0 1 2	abc
3 4 5	



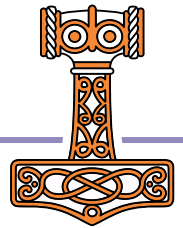
Ever tried this?

```
data←(2 3⍴⍳6) 'abc'
⎕JSON{0=≡⍵:⍵ ⋄ 1<≢⍴⍵:∇↓⍵ ⋄ ∇¨⍵}data

{0=≡⍵:⍵ ⋄ 1<≢⍴⍵:∇↓⍵ ⋄ ∇¨⍵}data
```



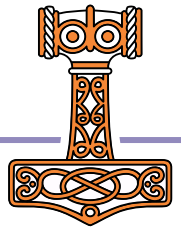
Ever tried this?

```
      data←(2 3⍴⍳6) 'abc' ⋄ 'ns'⎕NS'data'
      ⎕JSON{0=≡⍵:⍵ ⋄ 1<≢⍴⍵:∇↓⍵ ⋄ ∇¨⍵}ns
DOMAIN ERROR: JSON export: item "data[1]" of the
⎕JSON{0=≡⍵:⍵ ⋄ 1<≢⍴⍵:∇↓⍵ ⋄ ∇¨⍵}ns
      ^
```



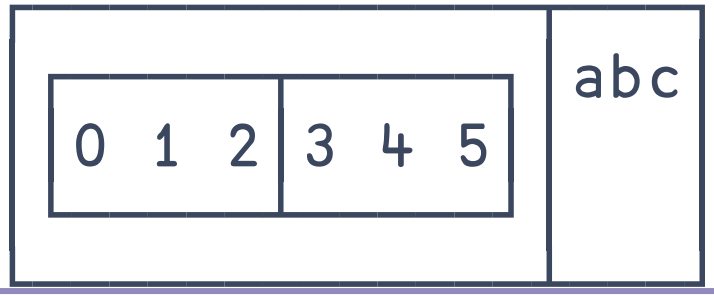
Try this!

```
      data←(2 3⍴⍳6) 'abc' ⋄ 'ns'⎕NS'data'
      ⎕JSON⍠'HighRank' 'Split'   ⊢   ns
{"data":[[[1,2,3],[4,5,6]],"abc"]}
```



Try this!

```
data←(2 3⍴⍳6) 'abc' ◇ 'ns'⎕NS'data'
⎕JSON⍠'HighRank' 'Split'    ⊢    data

⎕JSON⍠'HighRank' 'Split'⍤2 ⊢    data
```

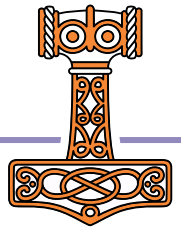


View>Master Views>Slide Master then [PgUp] to edit title!

{JSON:5,}

JSON for Humans

```
]JSON[:'Dialect'
```

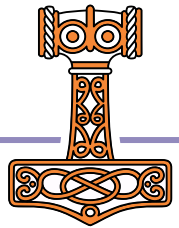


JSON

```
{
  "Settings": {
     "9&11": ["\t", "\u000B"],
     "MAXWS": "2GB",
     "ROOTDIR":
"/my-own/root/directory",
     "UserOption": "quote\"me"

  }
}
```

JSON5

```
{
  Settings: {
     "9&11": ["\t", "\v"],


  }
}
```

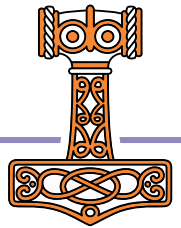


JSON

```
{
  "Settings": {
    "9&11": ["\t", "\u000B"],
    "MAXWS": "2GB",
    "ROOTDIR":
"/my-own/root/directory",
    "UserOption": "quote\"me"

  }
}
```

JSON5

```
{
  Settings: {
    "9&11": ["\t", "\v"],
    MAXWS: "2GB", // memory limit


  }
}
```

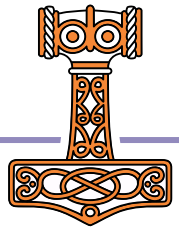


JSON

```
{
  "Settings": {
    "9&11": ["\t", "\u000B"],
    "MAXWS": "2GB",
    "ROOTDIR":
"/my-own/root/directory",
    "UserOption": "quote\"me"

  }
}
```

JSON5

```
{
  Settings: {
    "9&11": ["\t", "\v"],
    MAXWS: "2GB", /* memory limit */
    ROOTDIR: "/my-own/root/direct\
ory",
    UserOption: 'quote"me',

  }
}
```

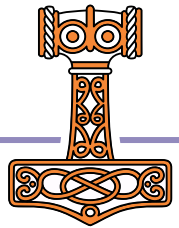


JSON

```
{
  "Settings": {
    "9&11": ["\t", "\u000B"],
    "MAXWS": "2GB",
    "ROOTDIR":
"/my-own/root/directory",
    "UserOption": "quote\"me",
    "FNAME": "[rootdir]/filename"
  }
}
```

JSON5

```
{
  Settings: {
    "9&11": ["\t", "\v"],
    MAXWS: "2GB", /* memory limit */
    ROOTDIR: "/my-own/root/direct\
ory",
    UserOption: 'quote"me',
    FNAME: '[rootdir]/filename',
  }
}
```

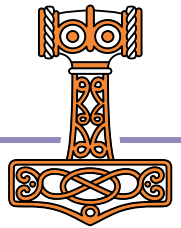


JSON Tables ⎕JSON

```
┌─────┬──────────────────────────┐
│1 2 3│1             0.5         │
│4 5 6│0.3333333333  0.25        │
│     │                          │
│     │0.2           0.1666666667│
│     │0.1428571429  0.125       │
└─────┴──────────────────────────┘
      1 ⎕JSON d
DOMAIN ERROR: JSON export: the right argument cannot be converted (⎕IO=1)
      1 ⎕JSON d
        ^
      1 (⎕JSON⍠'HighRank' 'Split') d
[[[[1,2],"AB"],["ABC","DEF"]],[[[1,2,3],[4,5,6]],...
```

Raw Text

Wrappers

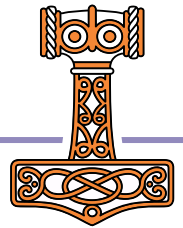


JSON Tables ⎕JSON

```
      table ←⍉⍪ 'Day' 'Ca$h Money $$$'
      ⎕←table⍪←3 2⍴'Monday' 1000 'Wednesday' 324 'Friday' 52
```

Monday	1000
Wednesday	324
Friday	52

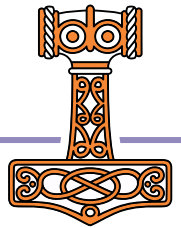
```
      ⎕JSON⍠'Compact'0⊂2,⊂table
[
  {
    "Day": "Monday",
    "Ca$h Money $$$": 1000
  }, {
    "Day": "Wednesday",
    "Ca$h Money $$$": 324
  }, {
    "Day": "Friday",
    "Ca$h Money $$$": 52
  }
]
```



Extended Attributes □ATX

Use □ATX in preference to □AT, □NC, □NR, □SIZE and □SRC (and some of the functionality of 5179I).

Press F1

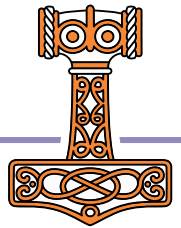


Extended Attributes □ATX

50 – 62

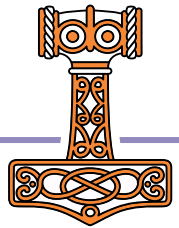
Information about source

Verbatim source: code kept as-typed with 2□FIX and Link



Exercises

https://is.gd/Y4IEaK



More Exercises

https://is.gd/IeDpNu