Getting Data and Code into the Workspace feat.]Get

Rich Park

Overview

```
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get
```

Overview

]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get

Overview

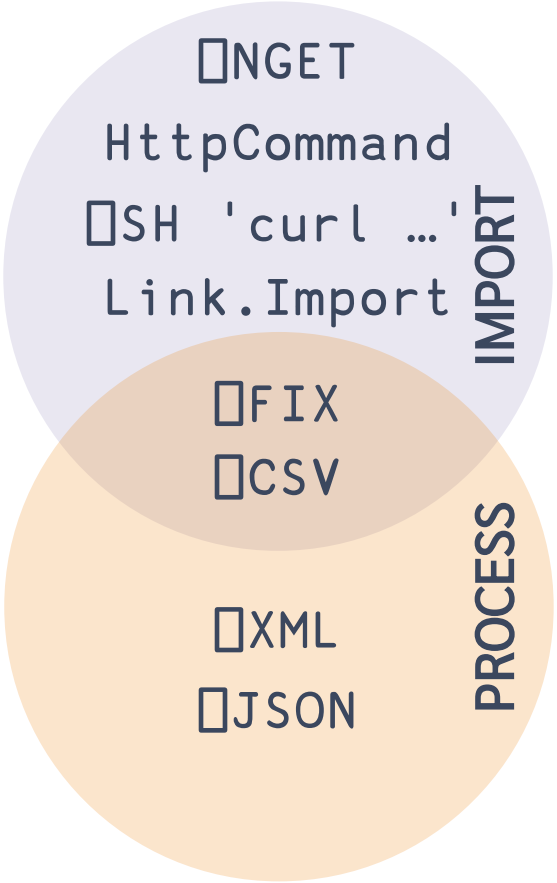
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



]EXPERIMENTAL.Get

“get pretty much anything from pretty much anywhere”

]EXPERIMENTAL.Get

“get pretty much anything from pretty much anywhere”

- One-stop utility for quickly getting bringing resources in.

]EXPERIMENTAL.Get

“get pretty much anything from pretty much anywhere”

- One-stop utility for quickly getting bringing resources in.
- Do not use at run time, as exact results may vary.

]EXPERIMENTAL.Get

Syntax

```
]get something
```

]EXPERIMENTAL.Get

Syntax

```
      ]get something
#.something
```

]EXPERIMENTAL.Get

Plain name heuristics

```
      ]get dfns                workspace from WS path
#.dfns
      ]get HttpCommand         source file from SALT path
#.HttpCommand
      ]get tools               directory relative to current
#.tools
```

]EXPERIMENTAL.Get

File extension detection

```
      ]get MyFn.aplf      Link-type source files
#.MyFn
      ]get myVar.apla     APL Array Notation files
#.myVar
      ]get text.charmat   Acre-type source files
#.text
```

]EXPERIMENTAL.Get

Data format conversion

`]get obj.json` `#.obj`	JS, JSON, JSON5, config files
`]get table.csv` `#.table`	Comma Separated Values
`]get data.xml` `#.data`	eXtensible Markup Language

]EXPERIMENTAL.Get

Retrieval

```
      ]get https://example.com/index.html
#.index
      ]get https://github.com/abrudz/Kbd
#.Kbd
```

]EXPERIMENTAL.Get

Protocols

```
      ]get https://github.com/abrudz/Kbd
#.Kbd
      ]get ftp://ftp.software.ibm.com/software
#.foo
      ]get github.com/abrudz/Kbd
#.Kbd
```

]EXPERIMENTAL.Get

Examples: unpacking

```
      ]get main.zip
#.main
      ]get github.com/user/repo/tree/main/sub
#.sub
```

]EXPERIMENTAL.Get

Examples: unpacking

```
      ]get dfns -only=dxb,dab
#.dfns
      ]map
#
·    dfns
·    ·    ∇ dab dxb
```

]EXPERIMENTAL.Get

Examples: unpacking

```
      )clear
clear ws
      ]get dfns -only=dxb,dab -unpack
#.dxb #.dab
      ]map
#
·    ∇ dab dxb
```

]EXPERIMENTAL.Get

Examples: options

```
      ]get obj.json:txt
#.obj
      10↑obj
{"myKey":1
      ]get thing -target=⎕SE.there
#.SE.there.thing
```

]EXPERIMENTAL.Get

Examples: options

```
      ]get /tmp/repo -sync
#.repo
      ]link.status
 Namespace        Directory     Files
 #.repo      ↔   c:/tmp/repo       14
```

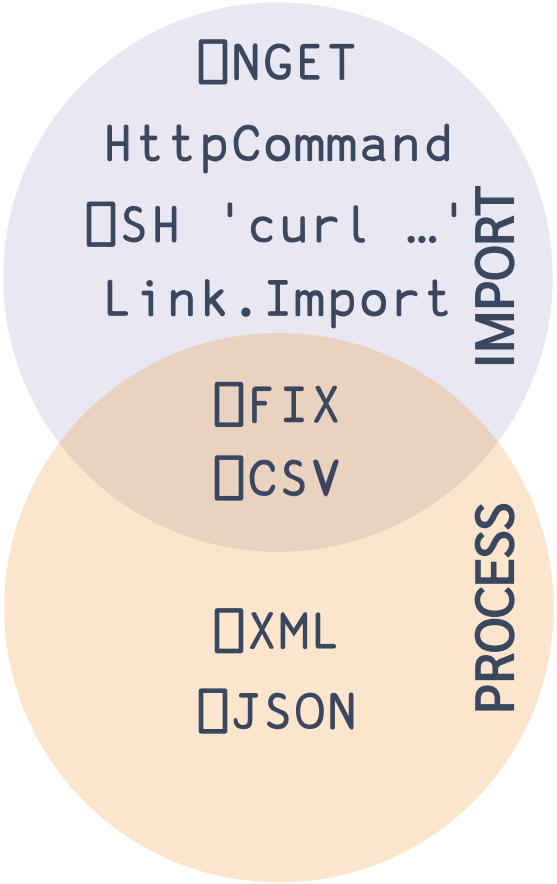
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

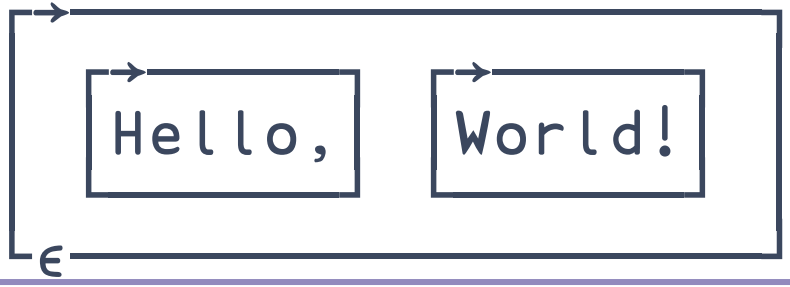
]EXPERIMENTAL.Get



⎕NGET

- Vector of characters: `⊃⎕NGET filename`

- Vector of character vectors: `⊃⎕NGET filename 1`



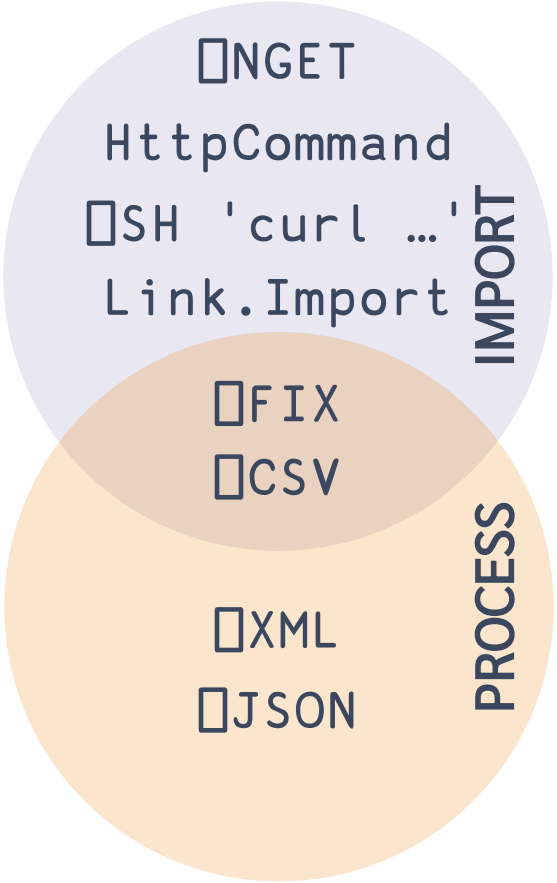
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



HttpCommand

- Fetch `HttpCommand`:

```
      ]SE.SALT.Load'HttpCommand'
```

- Vector of characters:

```
      resp←HttpCommand.Get url
      resp.HttpStatus
200
      resp.Data
```

HttpCommand

- Object oriented usage:

```
hc←⎕SE.SALT.New'HttpCommand'
```

- One-off usage:

```
((⎕SE.SALT.New'HttpCommand').Do 'GET' url).Data
```

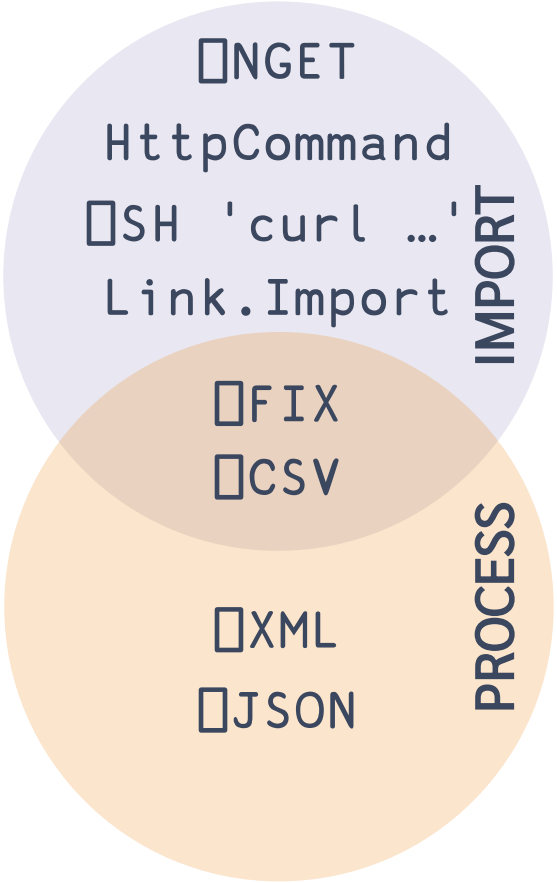
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



⎕SH 'curl …'

- Fetch raw vector of character vectors (bytes):

```
raw←⎕SH 'curl -L ',url
unicode←'UTF-8'⍥⎕UCS⍥⎕UCS¨raw
```

- Fetch raw file:

```
⎕SH 'curl -L -o ',outfile,' ',url
⋮
⎕NDELETE outfile
```

]SH 'curl …'

- Fetch zip file:

```
]SH 'curl -L -o ',outfile,' ',url
⋮
]NDELETE outfile
```

⎕SH 'curl …'

- Fetch zip file:

```
⎕SH 'curl -L -o ',outfile,' ',url
⎕MKDIR dir
:If ⊃'Windows'⍷⊃#⎕WG 'APLVersion'
    ⎕SH 'tar -xf ',outfile,' -C ',dir
:Else
    ⎕SH 'unzip ',outfile,' -d ',dir
:EndIf
⎕NDELETE outfile
```

⎕SH 'curl …'

```
tmp←739⌶0
```

- Fetch zip file:

```
⎕SH 'curl -L -o ',outfile,' ',url
⎕MKDIR tmp,'/',dir
:If ⊃'Windows'⍷⊃#.⎕WG 'APLVersion'
    ⎕SH 'tar -xf ',outfile,' -C ',dir
:Else
    ⎕SH 'unzip ',outfile,' -d ',dir
:EndIf
⎕NDELETE outfile
```

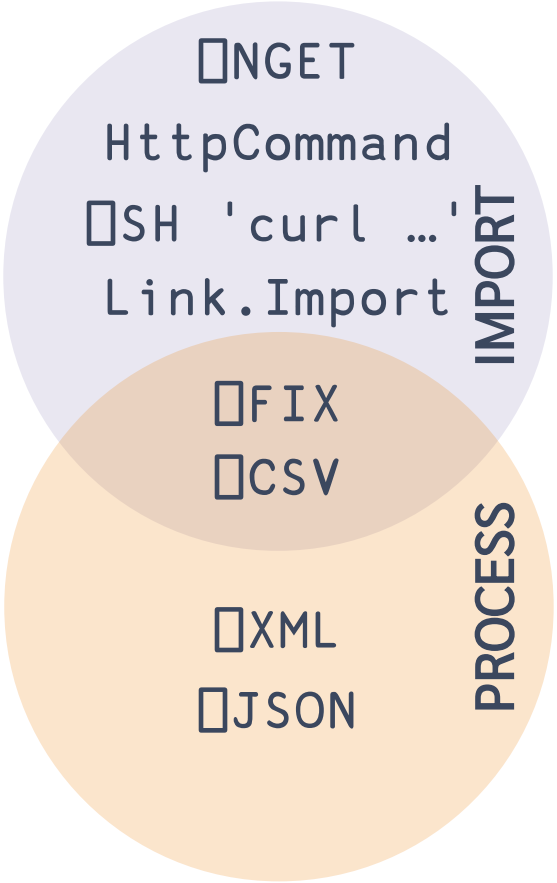
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



Link.Import

```
      ]SE.Link.Import '#.repo' '/tmp/repo'
Imported: #.repo ← c:\tmp\repo
```

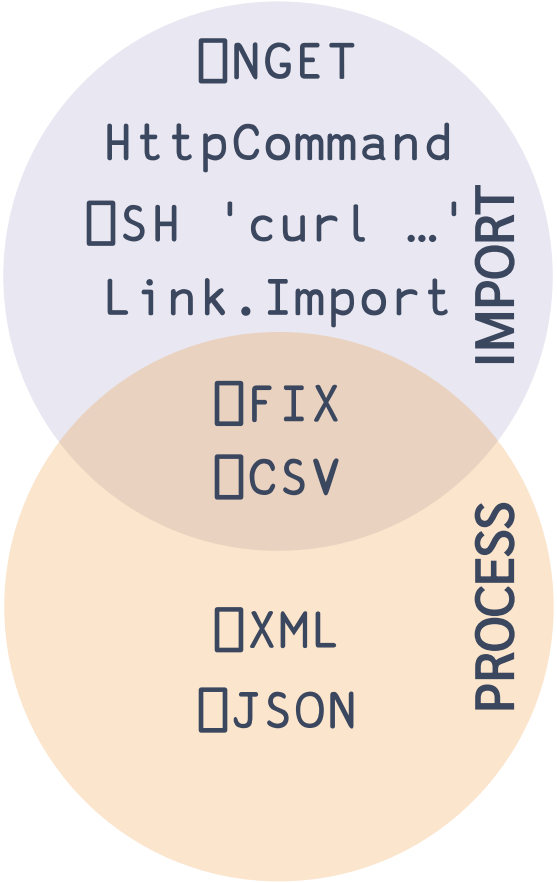
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

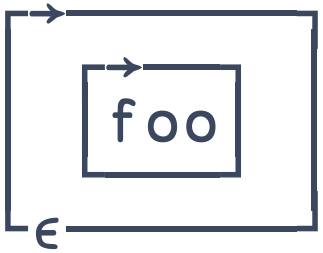
]EXPERIMENTAL.Get



⎕FIX

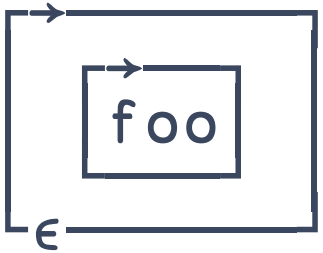
- From array:

```
⎕←2 ⎕FIX 'foo' 'bar'
```



- From file:

```
⎕←2 ⎕FIX 'file://foo.aplf'
```



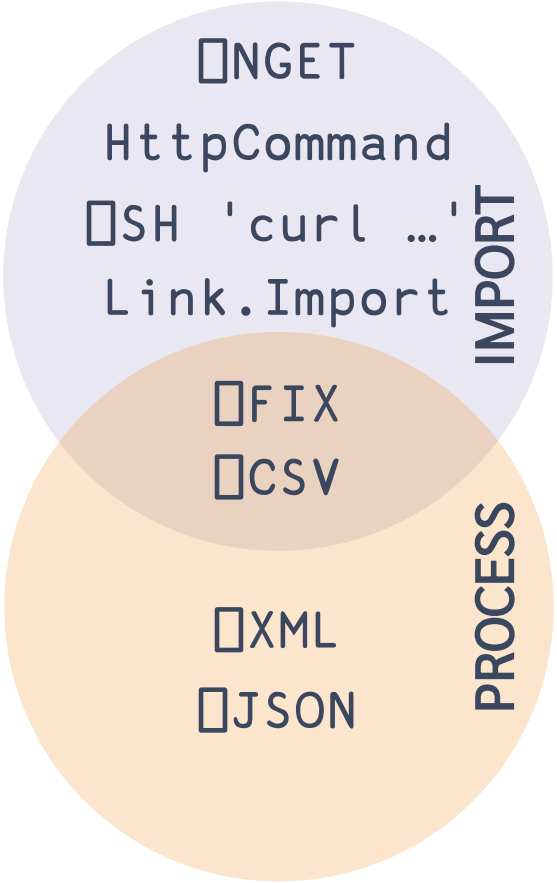
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



⎕CSV

- From array: `⎕CSV ('1,2,3' '4,5,6') ⍬ 4`

```
┌→──────┐
↓1 2 3  │
 4 5 6  │
└~──────┘
```

- From file: `⎕CSV 'foo.csv' ⍬ 4`

```
┌→──────┐
↓1 2 3  │
 4 5 6  │
└~──────┘
```

- For more techniques: is.gd/dyalog_csv

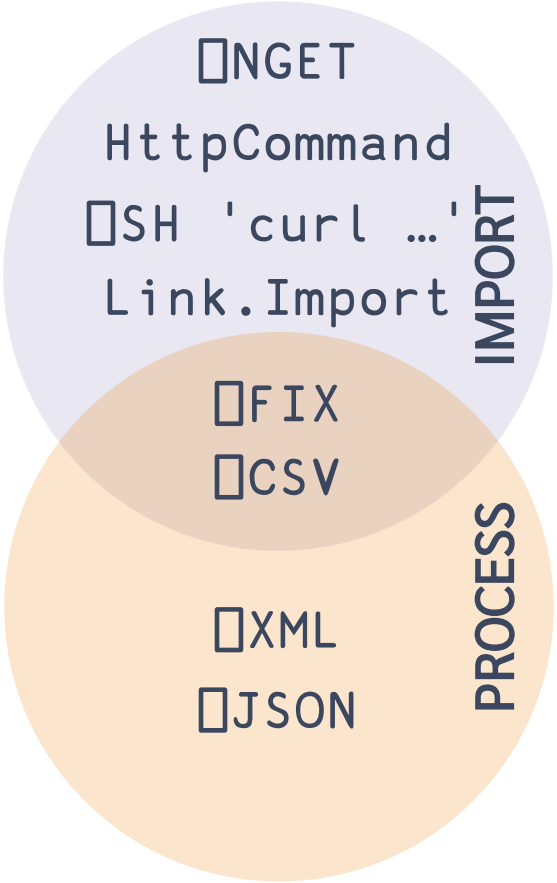
]OUTPUT.View

]TOOLS.APLCart

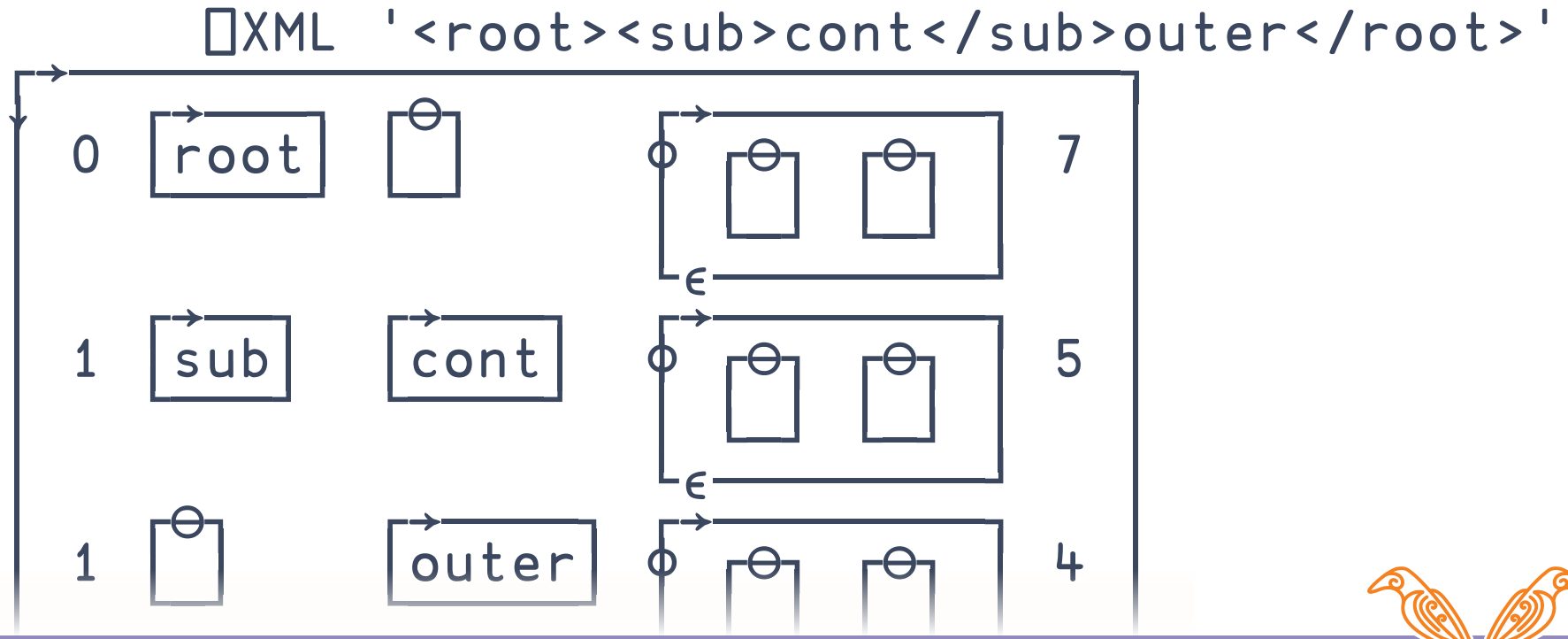
]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



]XML


]XML '<root>_{cont}outer</root>'
0
root
7
ϵ
1
sub
cont
5
ϵ
1
outer
4

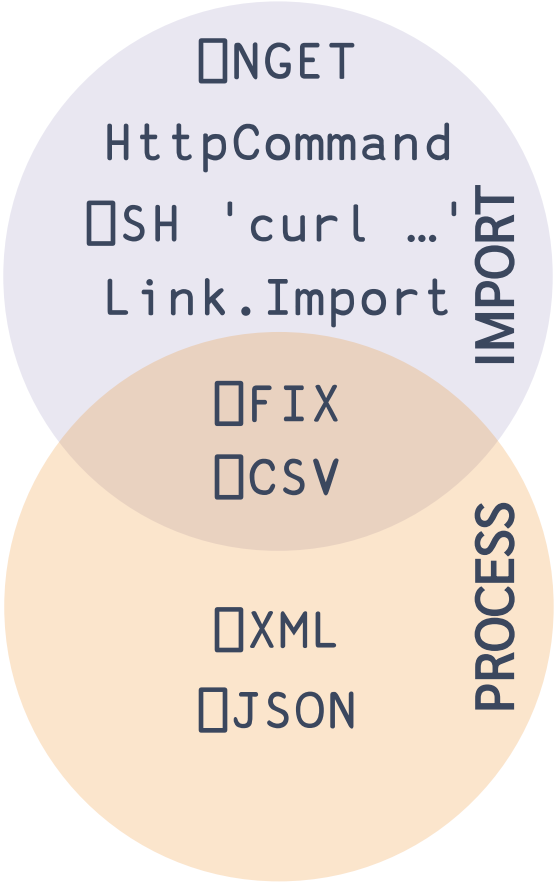
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get

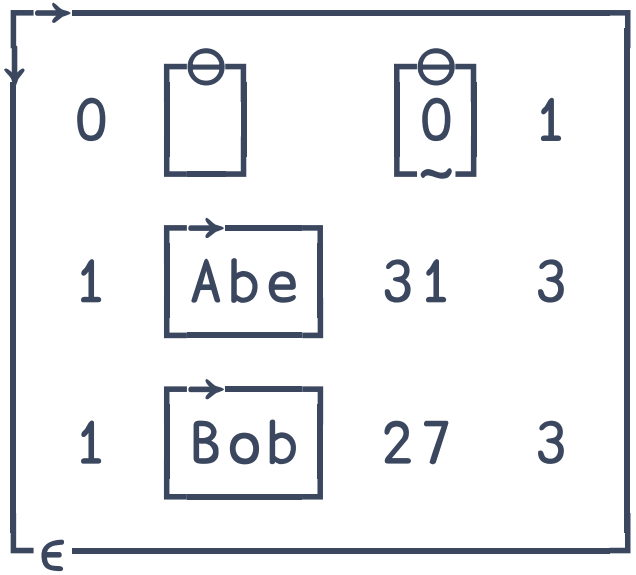


⎕JSON

```
      r←⎕JSON '{"Abe":31,"Bob":27}'
      ]names r
r.Abe r.Bob
      r.Abe
31
```

⎕JSON

```
⎕JSON⍠'M'⊢'{"Abe":31,"Bob":27}'
```



⎕JSON

```
r←⎕JSON '{"Abe":31,"Bob":27}'
```

⎕JSON

```
      r←⎕JSON '{Abe:31,/*Bob:27*/}'
DOMAIN ERROR: JSON import: invalid name at offse
      r←⎕JSON'{Abe:31,/*Bob:27*/}'
         ^
```

⎕JSON

```
r←⎕JSON '{Abe:31,/*Bob:27*/}'
```

⎕JSON

```
r←⎕JSON⍠'Dialect' 'JSON5'⊢'{Abe:31,/*Bob:2
```

Overview

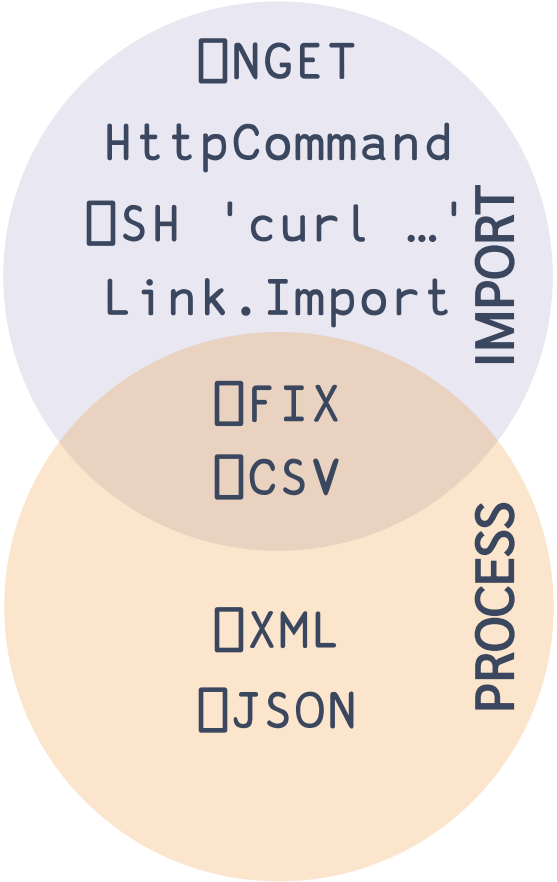
]OUTPUT.View

]TOOLS.APLCart

]WS.Names

]OUTPUT.Repr

]EXPERIMENTAL.Get



Next Webinar

June 9: TBA

Remember: BAA webinars every other week
britishaplassociation.org/webinar-schedule-2022
May 19th; June 2nd, 16th, 30th; etc.

More at: apl.wiki/activities