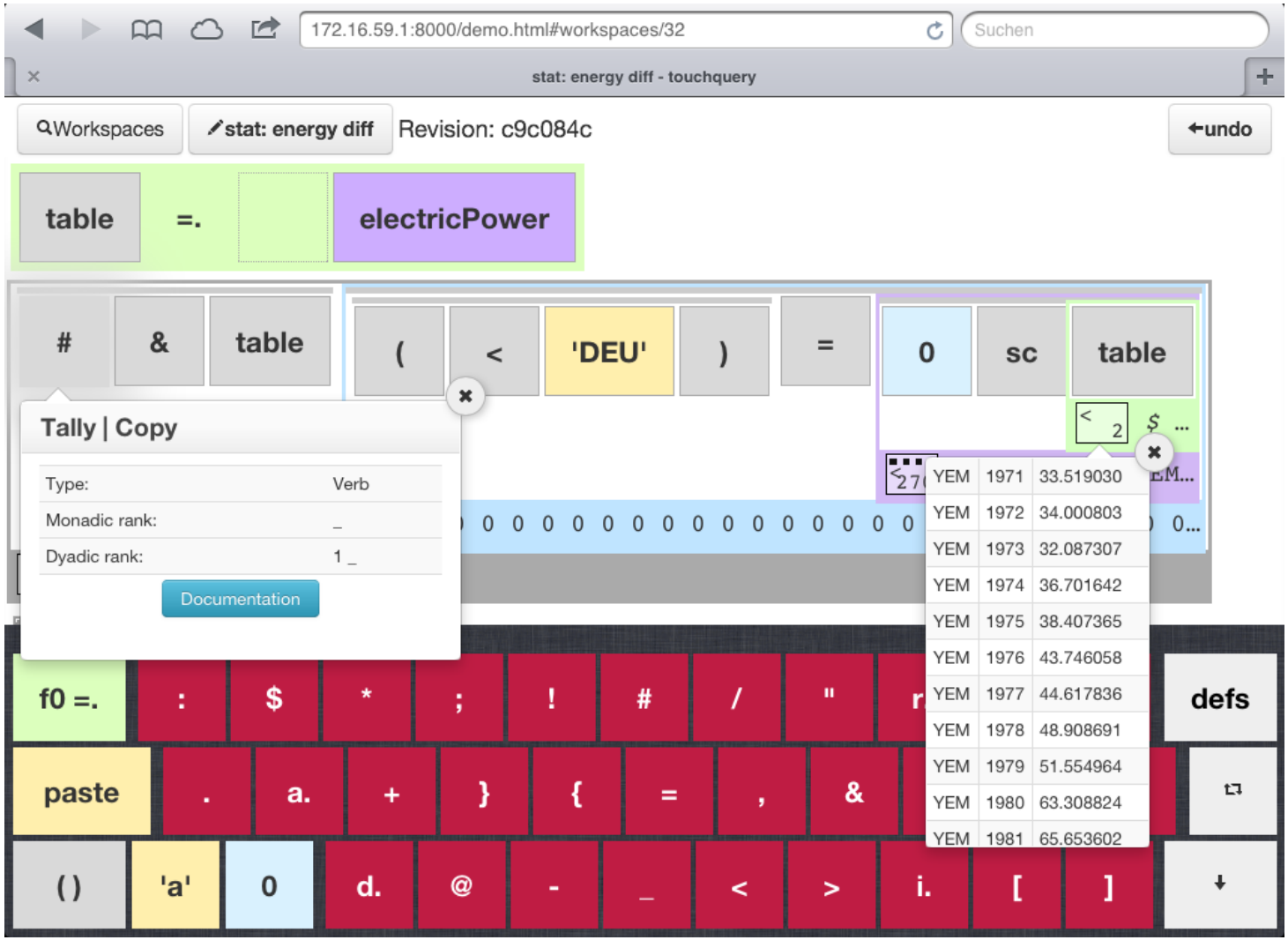
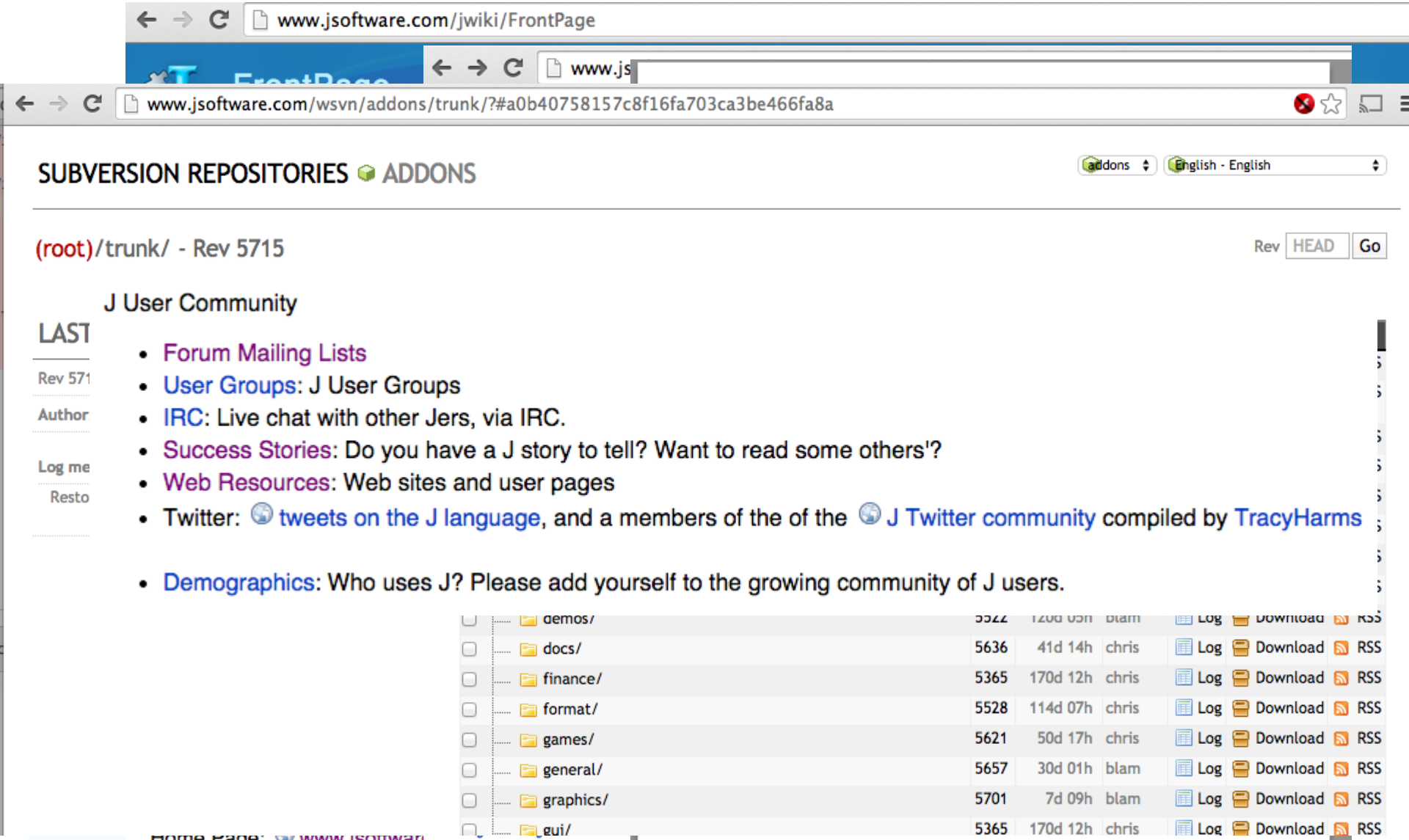
Context Oriented J

Bastian Kruck, 27.01.2015

Touchquery

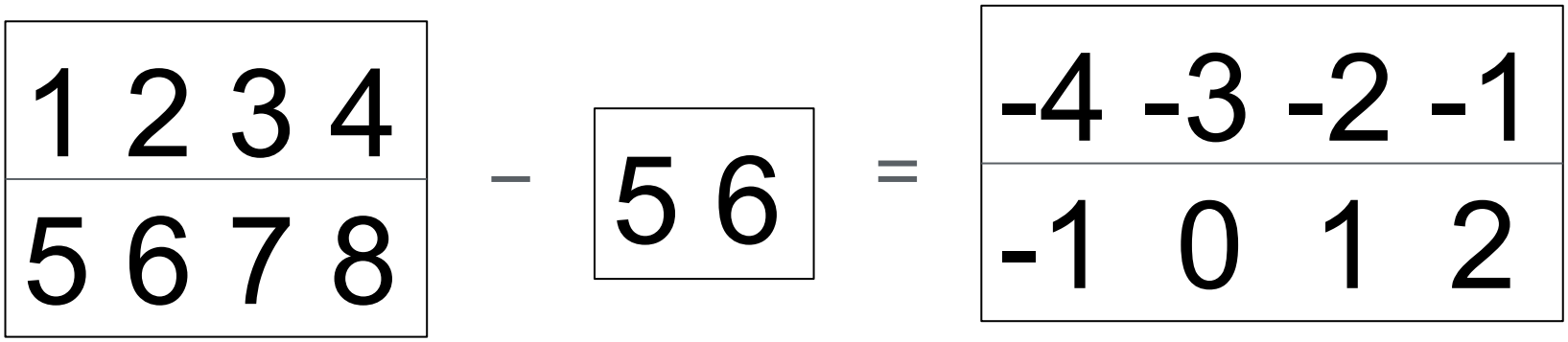

172.16.59.1:8000/demo.html#workspaces/32
Suchen
stat: energy diff - touchquery
Workspaces
stat: energy diff
Revision: c9c084c
undo
table
=.
electricPower
& table
(< 'DEU') =
0 sc table
Tally | Copy
Type: Verb
Monadic rank: _
Dyadic rank: 1 _
Documentation
YEM 1971 33.519030
YEM 1972 34.000803
YEM 1973 32.087307
YEM 1974 36.701642
YEM 1975 38.407365
YEM 1976 43.746058
YEM 1977 44.617836
YEM 1978 48.908691
YEM 1979 51.554964
YEM 1980 63.308824
YEM 1981 65.653602
f0 =.
: $ * ; ! # / "
defs
paste
. a. + } { = , &
( ) 'a' 0 d. @ - _ < > i. [ ]

What is J?


www.jsoftware.com/jwiki/FrontPage
www.jsoftware.com/wsvn/addons/trunk/?#a0b40758157c8f16fa703ca3be466fa8a
SUBVERSION REPOSITORIES ADDONS
addons
English - English
(root)/trunk/ - Rev 5715
Rev HEAD Go
J User Community
Forum Mailing Lists
User Groups: J User Groups
IRC: Live chat with other Jers, via IRC.
Success Stories: Do you have a J story to tell? Want to read some others'?
Web Resources: Web sites and user pages
Twitter: tweets on the J language, and a members of the of the J Twitter community compiled by TracyHarms
Demographics: Who uses J? Please add yourself to the growing community of J users.
docs/ 5636 41d 14h chris Log Download RSS
finance/ 5365 170d 12h chris Log Download RSS
format/ 5528 114d 07h chris Log Download RSS
games/ 5621 50d 17h chris Log Download RSS
general/ 5657 30d 01h blam Log Download RSS
graphics/ 5701 7d 09h blam Log Download RSS
gui/ 5365 170d 12h chris Log Download RSS
Same • Left
Same • Right
Catalogue • From

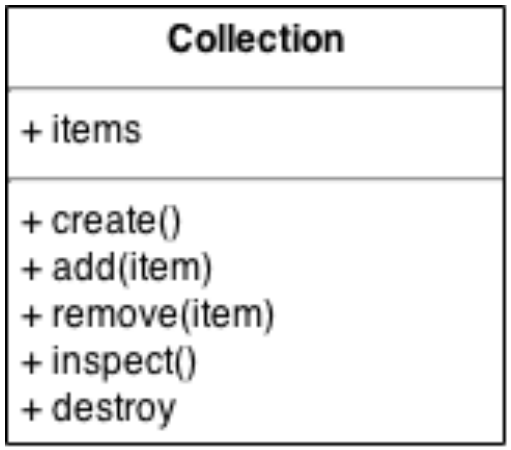
What is J? - Array Programming

- J is great in processing numeric data




```
    (1 2 3 4,: 5 6 7 8) - 5 6
_4 _3 _2 _1
_1  0  1  2
```

What is J? - OOP in J



define class “Collection”

```
coclass 'Collection'
  create  =: monad : 'items =: 0 $ 0'
  add     =: monad : '# items =: (< y) , items'
  remove  =: monad : '# items =: items -. < y'
  inspect =: monad : 'items'
  destroy =: codestroy_z_
cocurrent <'base'
```

from scope “z”

borrow destructor “codestroy”

return to “base” scope to end the class definition

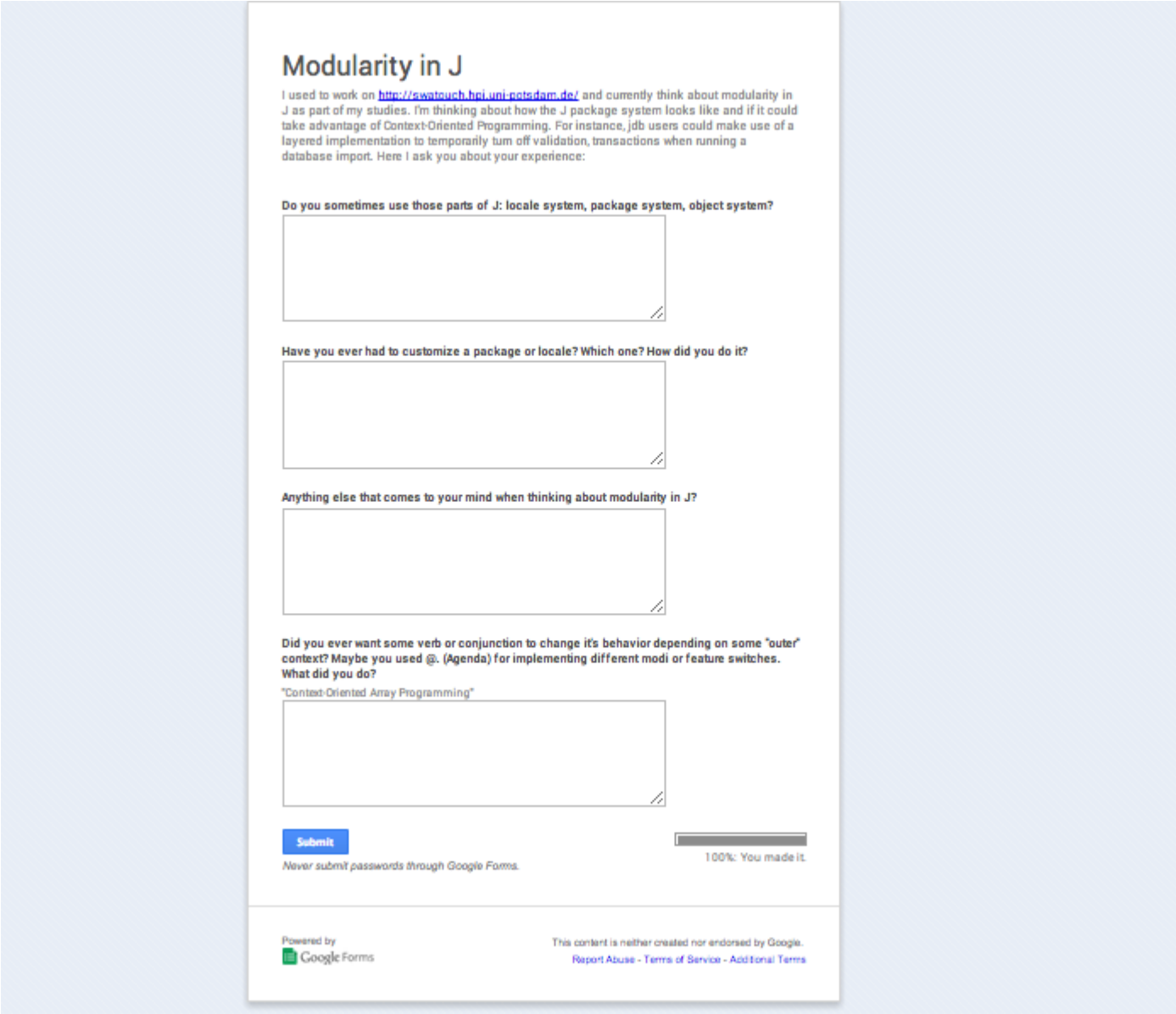
create a new object

```
myCollection =: 0 conew 'Collection'
add__myCollection 'Python'
add__myCollection 'Smalltalk'
add__myCollection 'C'
```

call method add

from object “myCollection”

Survey on Modularity In J



Modularity in J

I used to work on http://swatrouch.hpi.uni-potsdam.de/ and currently think about modularity in J as part of my studies. I'm thinking about how the J package system looks like and if it could take advantage of Context-Oriented Programming. For instance, jdb users could make use of a layered implementation to temporarily turn off validation, transactions when running a database import. Here I ask you about your experience:

Do you sometimes use those parts of J: locale system, package system, object system?

Have you ever had to customize a package or locale? Which one? How did you do it?

Anything else that comes to your mind when thinking about modularity in J?

Did you ever want some verb or conjunction to change it's behavior depending on some "outer" context? Maybe you used @. (Agenda) for implementing different modi or feature switches. What did you do?

"Context-Oriented Array Programming"

Submit

100%: You made it.

Never submit passwords through Google Forms.

Powered by Google Forms

This content is neither created nor endorsed by Google.
Report Abuse - Terms of Service - Additional Terms

Survey on Modularity In J - Answers

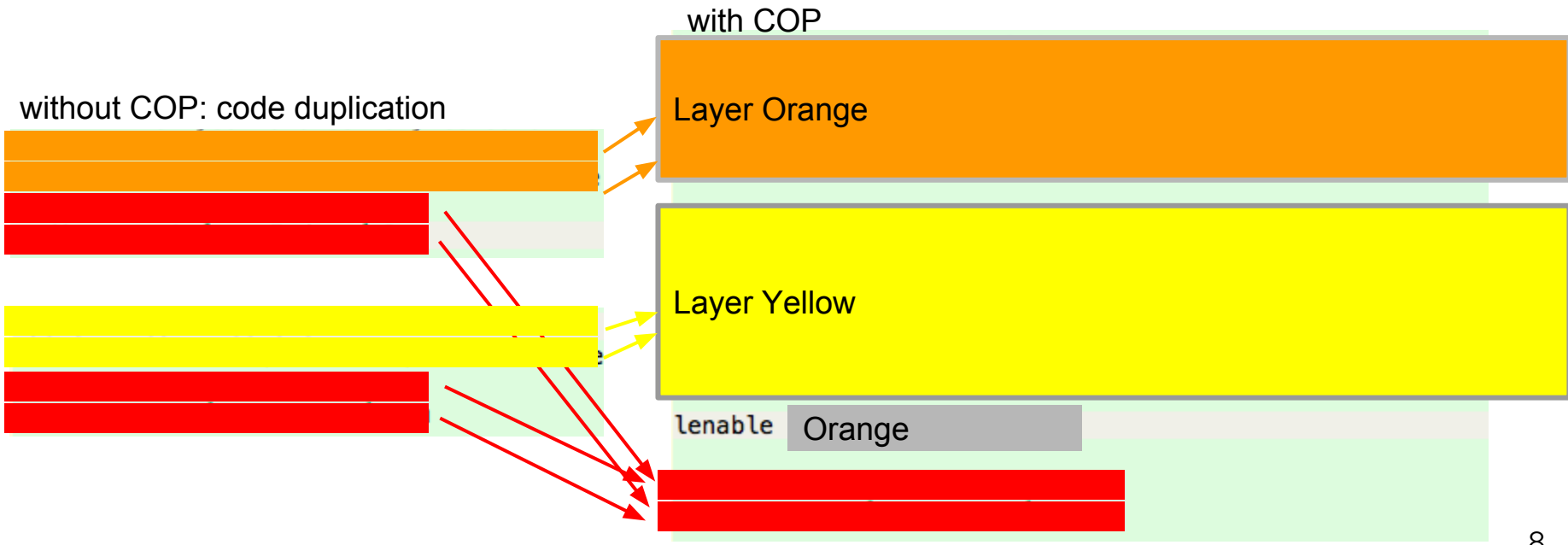
"Well written J allows "**equational modularity**" which is to say: you can change the architecture and/or control flow and/or the data structures with minimal coding changes."

"**Verbs (functions) tend to be modular by nature** if one avoids globals. In tacit J, there are no named items, so there can't be globals."

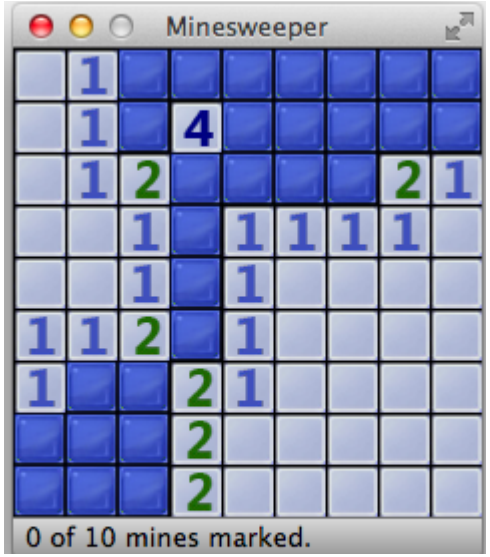
"I also commonly use plot, viewmat and some of the math solvers. **Are these packages?**"

Survey on Modularity In J - Results

- 3 ppl. answered
- packages, locales and OOP in J seem to be known, although being a minor part of the code
- I appears that many scripts are written with few architecture and maintainability in mind
- **modularity on assignment-level** will help in both, small scribbles and more advanced programs
- Idea: assignment-level layering

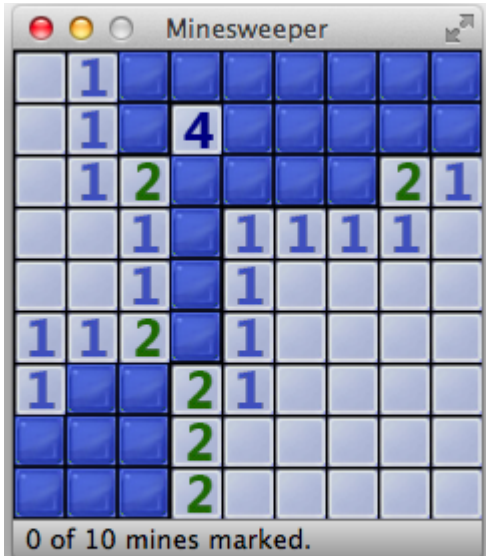


Where do we need COP in J? - Representations

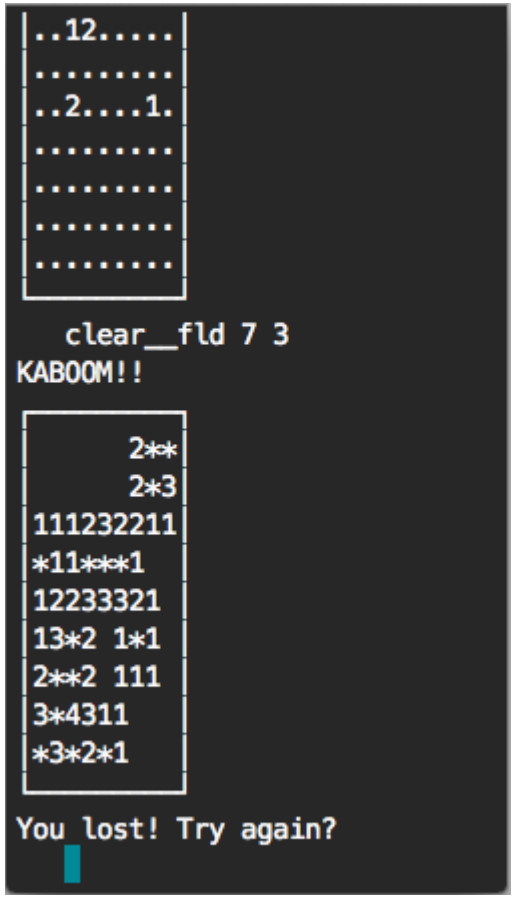


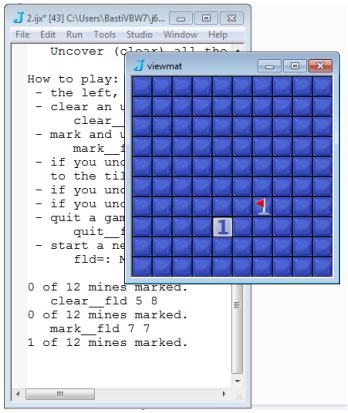
Where do we need COP in J? - Representations

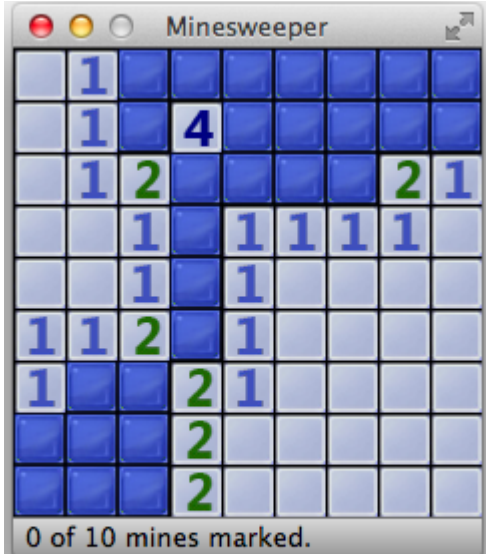

```
..12.....
.........
..2....1.
.........
.........
.........
.........

   clear__fld 7 3
KABOOM!!

      2**
      2*3
111232211
*11***1
12233321
13*2 1*1
2**2 111
3*4311
*3*2*1

You lost! Try again?
```

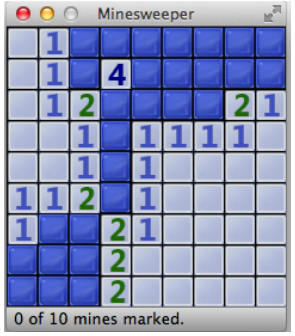


Where do we need COP in J? - Representations





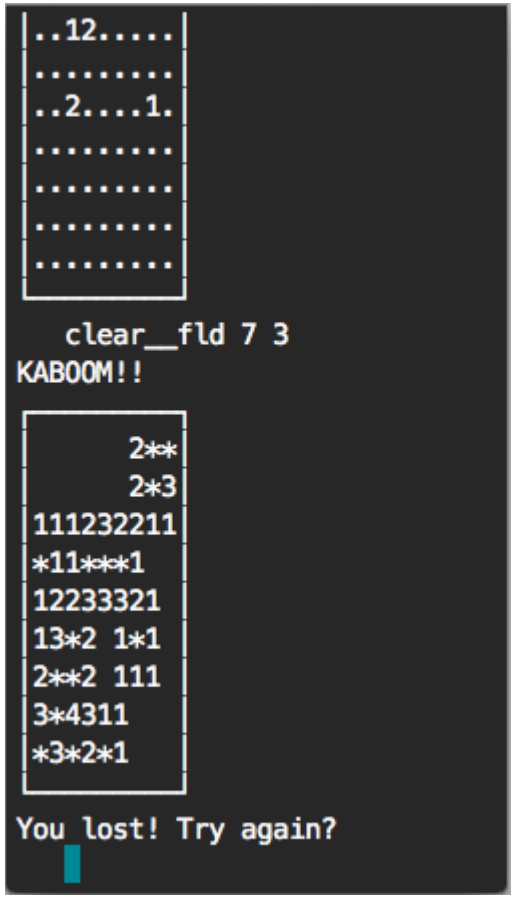


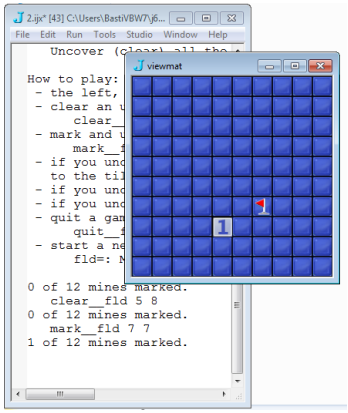


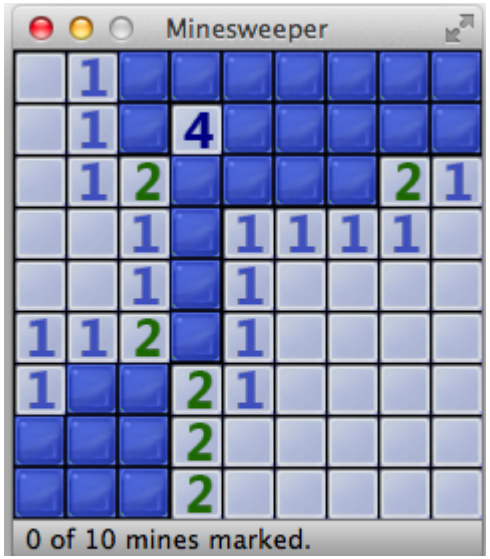
Where do we need COP in J? - Representations

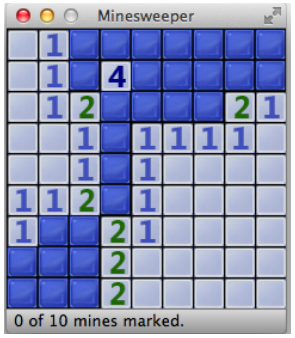
Input: mineswpcon → mineswp ← mineswpwd

Output:





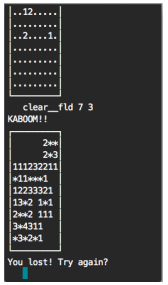
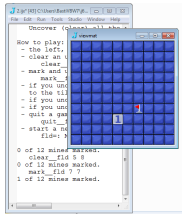




Where do we need COP in J? - Representations

Output:

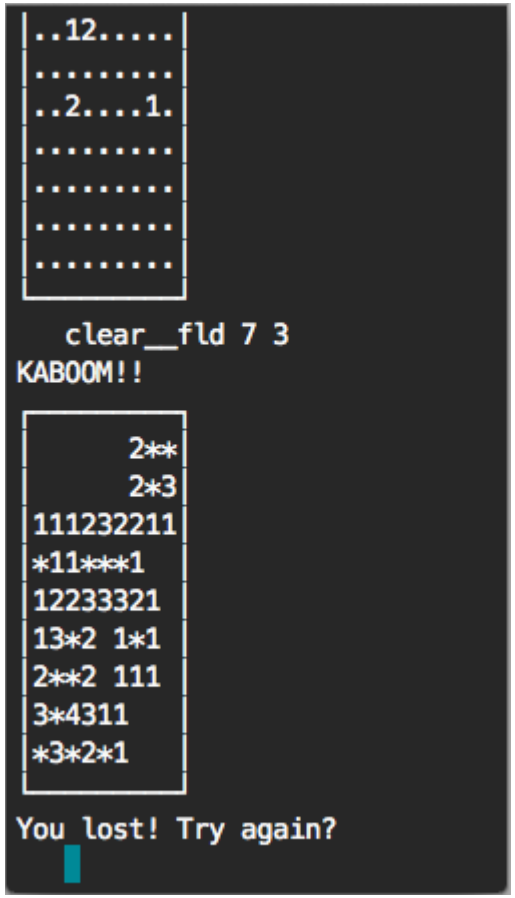
```
NB.TilesG26=: ''    NB. dummy variable
TilesG26=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles26.png' NB. can uncomment on
TilesA=: ' 12345678**.?'
TextDisplay=: 0                  NB. set to zero to display minefield using viewmat
```

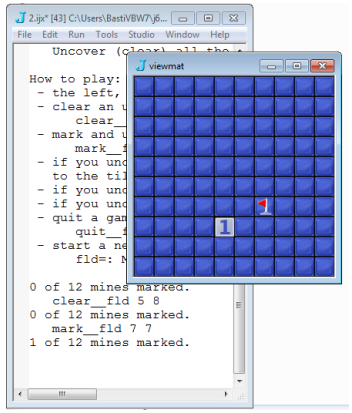
Uncomment a line, comment a line and set a flag to change the behavior from  to 

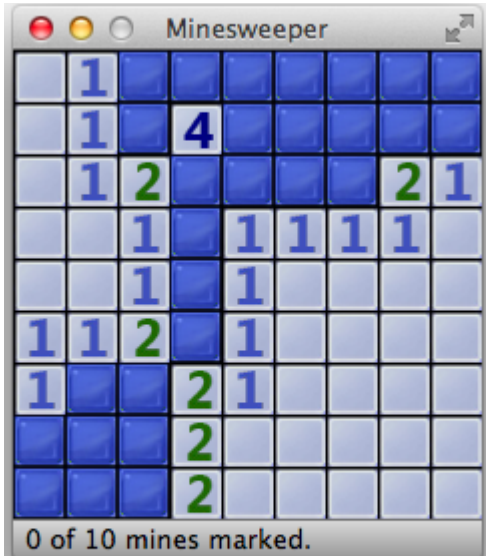
Where do we need COP in J? - Representations

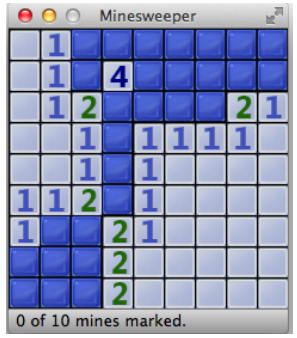
Input: mineswpcon → mineswp ← mineswpwd

Output:

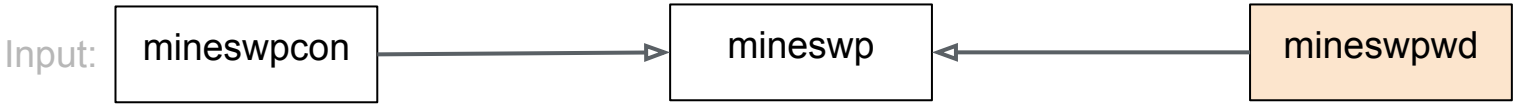








Where do we need COP in J? - Representations



Output:

```
NB. Tiles=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles18.png'
Tiles=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles26.png'
```

Uncomment a line and comment a line to change the behavior from  to 

Where do we need COP in J? - Representations

Input:

Output:

```
(function(){
  if. IFQT do.
    readimg=: readimg_jqtide_
  elseif. 'Android'-:UNAME do.
    readimg=: readimg_ja_
  end.
  empty''
})()

AddonPath=. jpath '~addons/games/minesweeper/'
NB. Tiles=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles18.png'
Tiles=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles26.png'
```

Uncomment a line and comment a line to change the behavior from  to 

Where do we need COP in J? - Representations

Input:

Output:

```
3 : 0''
  if. IFQT do.
    readimg=: readimg_jqtide_
  elseif. 'Android'-:UNAME do.
    readimg=: readimg_ja_
  end.
  empty''
)

AddonPath=. jpath '~addons/games/minesweeper/'
NB. Tiles=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles18.png'
Tiles=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles26.png'
```

NB: if-then-else only exist in function definitions that are not nestable

Uncomment a line and comment a line to change the behavior from  to 

Naming is hard

ContextJ

Naming is hard

JCop

Naming is hard

COJ

COJ: before and after

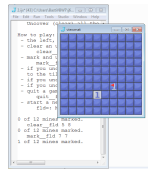
without COJ

```
AddonPath=. jpath '~addons/games/minesweeper/'
NB.TilesG26=: ''   NB. dummy variable
TilesG26=: ,((2 2 $ #) <;._3 ]) readimg AddonP
TilesA=: ' 12345678**.?'
TextDisplay=: 0                NB. set to ze

NB. . . .

mscon_update=: 3 : 0
  'isend msg'=. eval ''
  IsEnd=: isend
  smoutput msg
  tiles=. TextDisplay{:: TilesG26;TilesA
  display Tiles showField isend
  if. isend do.
    msg=. ('K'={.msg) {:: 'won';'lost'
    smoutput 'You ',msg,'! Try again?'
    destroy ''
  end.
  empty''
)

NB. . . .

display=: 3 : 0
  if. TextDisplay do.
    smoutput@< y
  else.
    closeall_jviewmat_ :: ] ''
    ([: viewrgb@; ,.&.>/"1) y
  end.
  empty''
)
```

with COJ

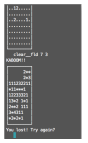
```
coinsert 'coj'

NB. describe mineswpcon when we are in J6
'mineswpcon' lwhen 'J6'
  NB. use viewmat
  AddonPath=. jpath '~addons/games/minesweeper/'
  Tiles =: ,((2 2 $ #) <;._3 ]) readimg AddonPat
  display =: monad define
    closeall_jviewmat_ :: ] ''
    ([: viewrgb@; ,.&.>/"1) y
    NB. proceed…
    display_mineswpconLayerZ_ ''
  ).
)
```

J6 Layer



```
NB. describe mineswpcon when we are not in J6
'mineswpcon' lwhen 'NotJ6'
  Tiles =: ' 12345678**.?'
  NB. asdf.
  display =: monad define
    smoutput@< y
    display_mineswpconLayerZ_ ''
  ).
)
```

NotJ6 Layer

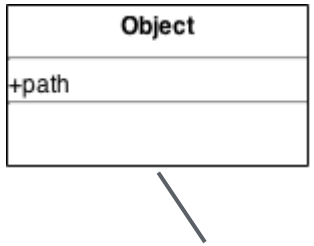


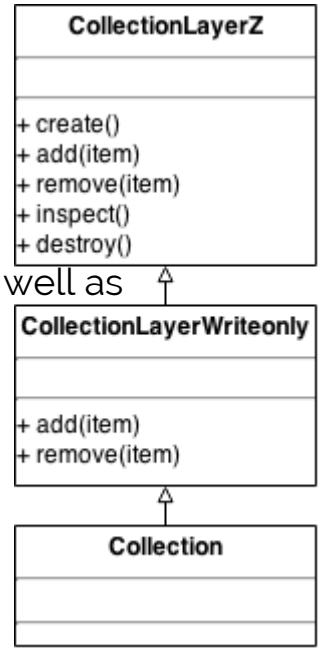
```
NB. describe mineswpcon shared layer
'mineswpcon' lwhen 'Z'
  display=: monad define
    empty''
  ).
)
```

Shared Layer

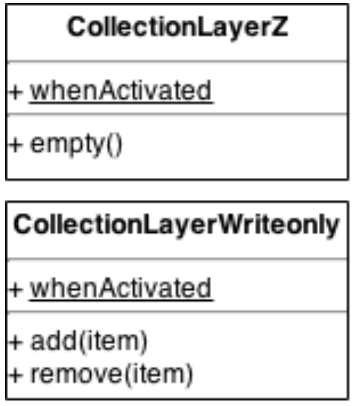
```
'mineswpcon' lenable 'NotJ6'
```

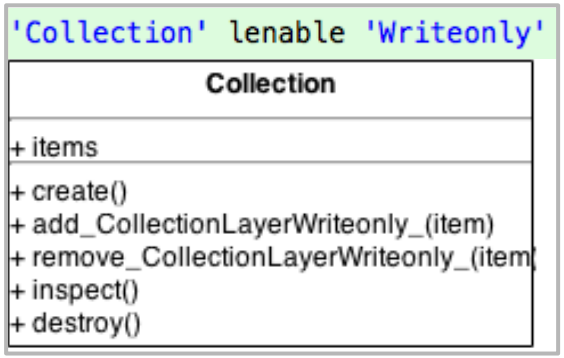
COJ Implementation: Alternatives When Implementing Layers

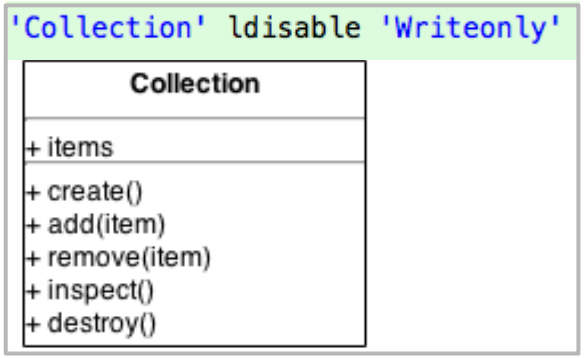




- Alter inheritance chain
  - enabling a layer requires update of lookup path of all affected objects (as well as classes)
  - expensive activation time lookup (both, CPU and memory)
  - call time lookup with stub method possible, although complex
- Override named things with the layered implementation
  - no runtime speed loss
  - acceptable activation time (no need to touch the objects)
  - proceed by calling a method from an explicitly named layer
  - order of activation is maintained
  - deactivation can be realized by activating the remaining layers in the right order





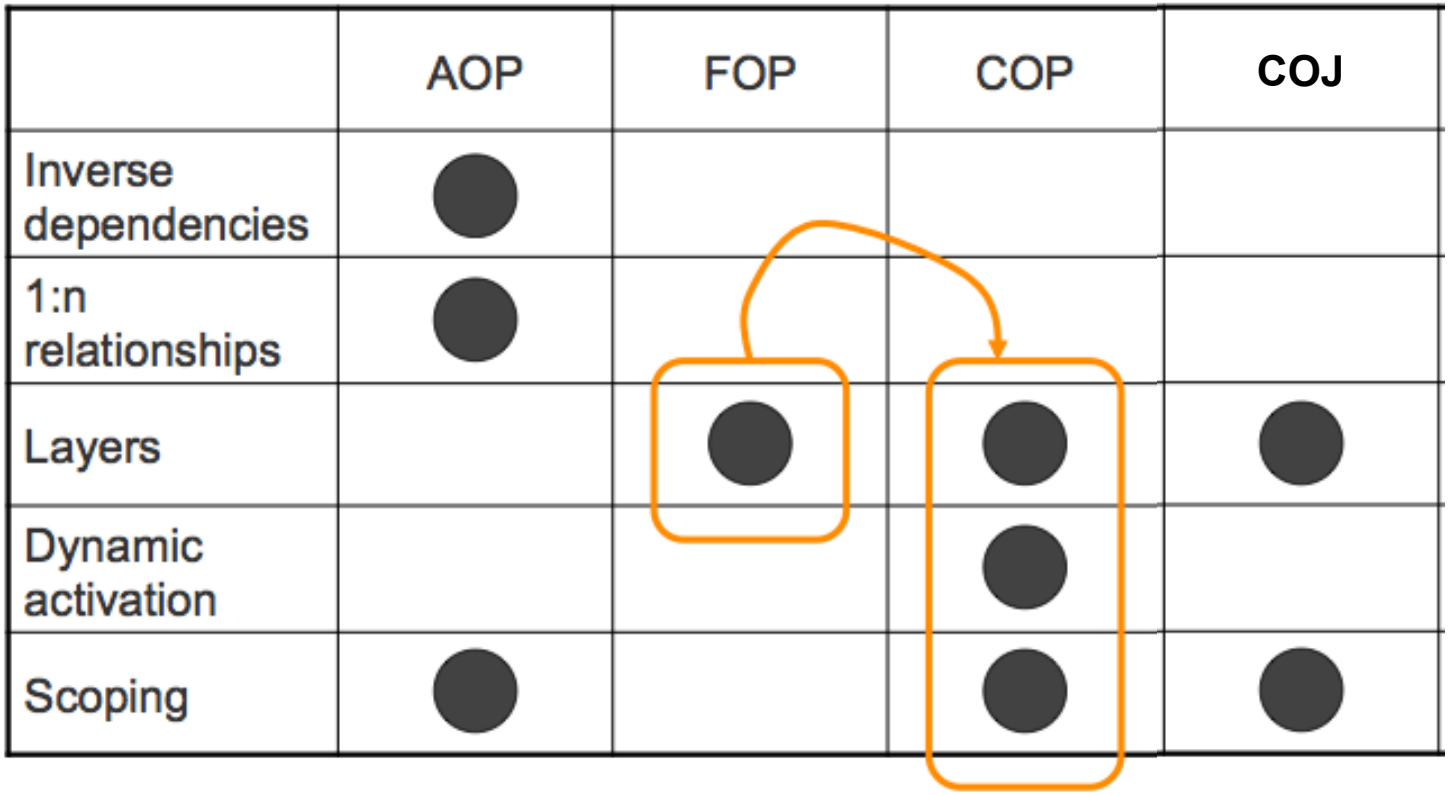


COJ Implementation: Details

- Implementation in 2 files (utils and core)
- 3 test suites with 12 tests in sum
- 316 LOC (202 LOC are tests)

```
     require 'general/unittest'
     unittest jpath '~user/projects/coj/utils_test.ijs'
Test: /users/reflektor/j64-802-user/projects/coj/utils_test.ijs
copy_methods ............................ OK
extractModifiedNames .................... OK
fromLocale .............................. OK
replaceFakeParen ........................ OK
replaceFakeParenMultiRow ................ OK
replaceFakeParenOfNoString .............. OK


     unittest jpath '~user/projects/coj/layer2_test.ijs'
Test: /users/reflektor/j64-802-user/projects/coj/layer2_test.ijs
lname ................................... OK
lnameShape .............................. OK
lwhen_collectionWriteonly ............... OK
lwhen_lwhenActivatedChanges ............. OK


     unittest jpath '~user/projects/coj/minesweeper_test.ijs'
Test: /users/reflektor/j64-802-user/projects/coj/minesweeper_test.ijs
J6_Tiles ................................ OK
NoJ6_Tiles .............................. OK
```

AOP, FOP, and COP



	AOP	FOP	COP	COJ
Inverse dependencies	●			
1:n relationships	●			
Layers		●	●	●
Dynamic activation			●	
Scoping	●		●	●

Summary: COJ is a COP Framework for J

- COJ is an assignment-based layer-in-class COP Framework for J
- activation-time lookup
- no call-time overhead
- build in userland without VM modifications
- state: prototype
- with COJ, you can replace immediate functions and smelling constructs with semantically structured code

```
3 : 0''
  if. IFQT do.
    readimg=: readimg_jqtide_
  elseif. 'Android'-:UNAME do.
    readimg=: readimg_ja_
  end.
  empty''
)
```

No more immediate functions

```
NB.TilesG26=: ''   NB. dummy variable
TilesG26=: ,((2 2 $ #) <;._3 ]) readimg AddonPath,'tiles26.png' NB. can uncomment on J6
TilesA=: ' 12345678**.?'
TextDisplay=: 0                NB. set to zero to display minefield using viewmat
```

Uncomment a line, comment a line and set a flag to change the behavior from to

No more uncommend-comment-flag constructs

```
coinsert 'coj'

NB. describe mineswpcon when we are in J6
'mineswpcon' lwhen 'J6'
  NB. use viewmat
  AddonPath=. jpath '~addons/games/minesweeper/'
  Tiles =: ,((2 2 $ #) <;._3 ]) readimg AddonPath
  display =: monad define
    closeall_jviewmat_ ''
    ([: viewrgb@; ,.&.>/"1) y
    NB. proceed…
    display_mineswpconLayerZ_ ''
  ).
)

NB. describe mineswpcon when we are not in J6
'mineswpcon' lwhen 'NotJ6'
  Tiles =: ' 12345678**.?'
  NB. asdf.
  display =: monad define
    smoutput@< y
    display_mineswpconLayerZ_ ''
  ).
)

NB. describe mineswpcon shared layer
'mineswpcon' lwhen 'Z'
  display=: monad define
    empty''
  ).
)
```

J6 Layer

NotJ6 Layer

Shared Layer

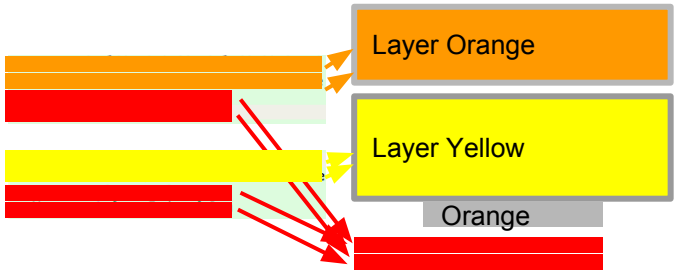
Semantically structured code

Future work

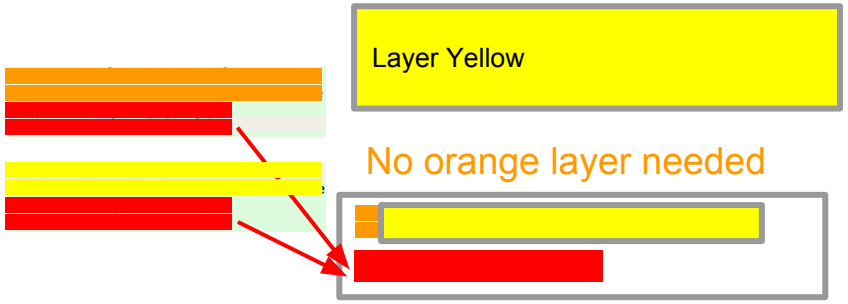
- **enable implicit proceeding** when having multiple layers enabled
- enable or disable layers across classes (is currently per-class)
- build conjunction (higher order function) to use COJ in tacits (parameter-less definitions) and time-critical cases

```
'add ''gehtNicht''' lwith C1 'Readonly'
```

- examine activation orders to enable inlining of default layer (e.g. z)



Without Smart Activation Order

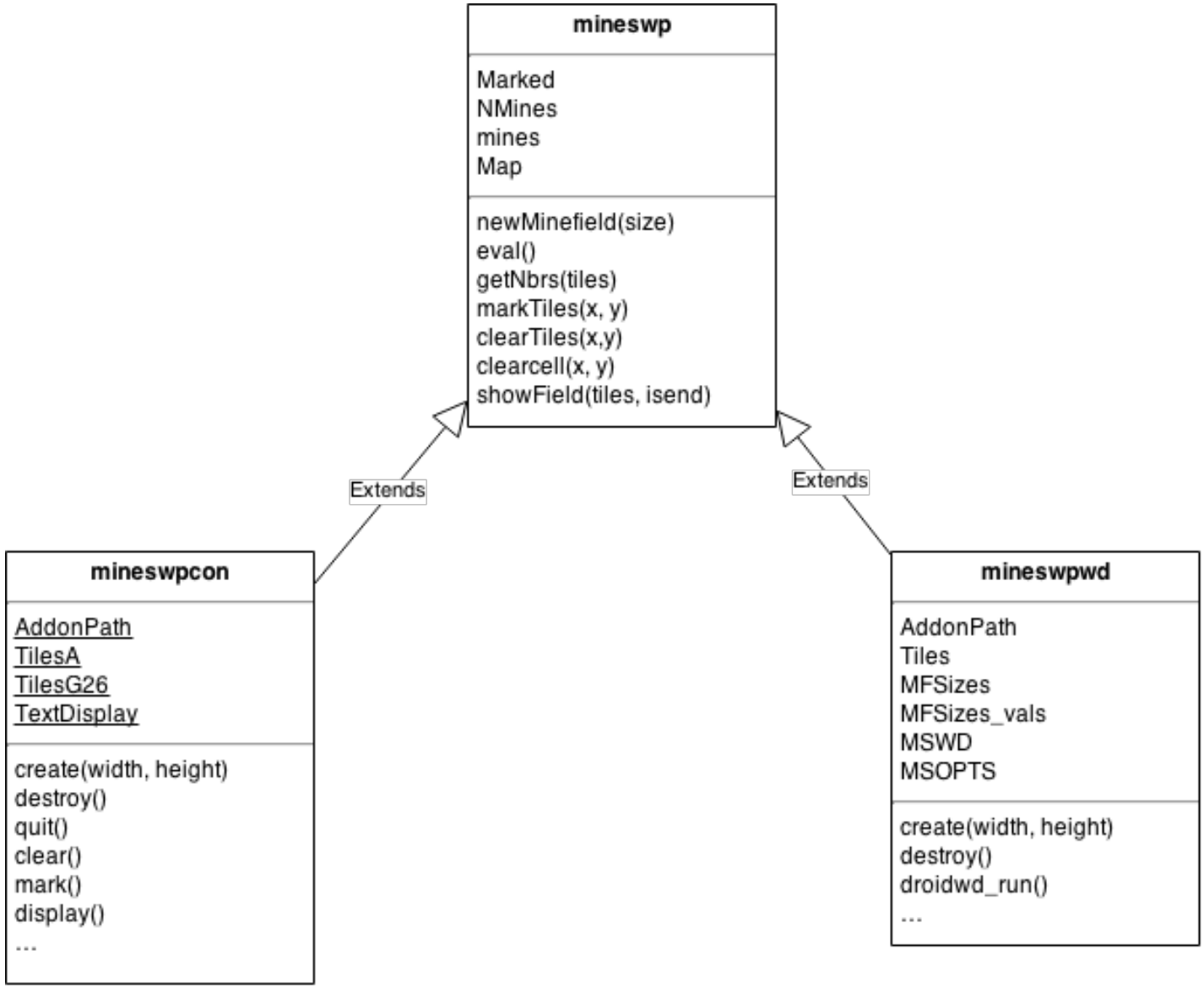


With Smart Activation Order

- examine need private layer variables
- bundle and share it

Sources

- http://swatouch.hpi.uni-potsdam.de/demo.html#workspaces/77f926b9627ebb7b1ab5527f35016205
- Minus http://www.jsoftware.com/help/dictionary/d120.htm
- /FG_SoftwareArchitekturen/ContextOrientedProgramming/Handouts/Cop14_02_AopCop.pptx.pdf
- http://www.hpi.uni-potsdam.de/hirschfeld/publications/media/AppeltauerHirschfeldHauptLinckePerscheid_2009_AComparisonOfContextOrientedProgrammingLanguages_AcmDL.pdf
- http://www.jsoftware.com/help/learning/24.htm
- http://www.jsoftware.com/help/learning/25.htm
- http://www.jsoftware.com/help/dictionary/dx018.htm
- https://hpi.de/studium/lehrveranstaltungen/it-systems-engineering/lehrveranstaltung/course/2014/kontextorientiertes_programmieren.html
- http://en.wikipedia.org/wiki/J_%28programming_language%29
- http://www.hpi.uni-potsdam.de/hirschfeld/cop/implementations/index.html
- http://www.jsoftware.com/jwiki/Addons/games/minesweeper
- http://draw.io for diagrams
- http://www.jsoftware.com/help/jforc/control_structures.htm
- http://www.jsoftware.com/help/dictionary/d001.htm
- http://www.jsoftware.com/jwiki/Vocabulary/fdot


mineswp
Marked
NMines
mines
Map
newMinefield(size)
eval()
getNbrs(tiles)
markTiles(x, y)
clearTiles(x,y)
clearcell(x, y)
showField(tiles, isend)
Extends
Extends
mineswpcon
AddonPath
TilesA
TilesG26
TextDisplay
create(width, height)
destroy()
quit()
clear()
mark()
display()
...
mineswpwd
AddonPath
Tiles
MFSizes
MFSizes_vals
MSWD
MSOPTS
create(width, height)
destroy()
droidwd_run()
...

```
  NB. define a Collection
  coclass 'Collection'
    create  =: 3 : 'items =: 0 $ 0'
    add     =: 3 : '# items =: (< y) , items'
    remove  =: 3 : '# items =: items -. < y'
    inspect =: 3 : 'items'
    destroy =: codestroy

  NB. build a collection
  cocurrent <'base'
    C1 =: 0 conew 'Collection'


NB. add_Collection_     =: 3 : '# items =: (< y) , items'
NB. remove_Collection_  =: 3 : '# items =: items -. < y'

lwith =: dyad : 0
  oldLocale =. coname''
  cocurrent y

  NB. imagine we looked up those methods from Readonly layer
  add         =. 3 : '# items'
  remove      =. 3 : '# items'

  NB. run it!
  (0!::100)x

  cocurrent oldLocale
)
```

```
   add__C1 'geht'
1
   inspect__C1 ''
┌────┐
│geht│
└────┘
   'add ''gehtNicht''' lwith C1 NB. 'Readonly'
   add 'gehtNicht'
1
   'add' lwith C1 NB. 'Readonly'
   add
3 : '# items'
   inspect__C1 ''
┌────┐
│geht│
└────┘
   add__C1 'gehtWieder'
2
   inspect__C1 ''
┌─────────┬────┐
│gehtWieder│geht│
└─────────┴────┘
```

layer2.ijs (Page 1)

```
lcreate =: 3 : 0
  oldLocale =. coname''
  cocurrent <y
  coinsert 'layer2'
  cocurrent oldLocale
)

NB. builds the name of the locale where we save the callbacks
lname =: 3 : 0
  NB. monadic case falls back to current locale
  (>coname '') lname y
:

  < ([ lcreate) x,'Layer',y
)

NB. saves an activation callback for the layer _y_
lwhen =: 3 : 0
  NB. monadic case falls back to current locale
  (>coname '') lwhen y
:
  targetLocale =. x lname y
  lwhenActivatedTxt__targetLocale =: replaceFakeParen noun define
  lwhenActivatedChanges__targetLocale =: extractModifiedNames lwhenActivatedTxt__targetLocale

  NB. initialize in the layer locale to allow proceeding by explicitly calling the layer
  x lenable__targetLocale y
)

lwhenActivated_z_ =: 3 :'0!:100 lwhenActivatedTxt'
```

layer2.ijs (Page 2)

```
NB. enables layer _y_ in layer _x_
lenable =: 3 : 0
  NB. monadic case falls back to current locale
  (>coname '') lenable y
:
  oldLocale =. coname''
  targetLocale =. x lname y
  if. (type 'lwhenActivatedTxt__targetLocale') -: <'not defined' do.
    smoutput 'error: lwhenActivatedTxt_', (>targetLocale),'_ is not defined'
    return.
  end.

  NB. apply callback to layer x
  cocurrent <x
  (0!:100) lwhenActivatedTxt__targetLocale
  cocurrent oldLocale
)

ldisable =: 3 : 0
  NB. monadic case falls back to current locale
  (>coname '') lenable y
:
  NB. TBD
  NB. lwhenActivatedChanges__targetLocale extractModifiers (x lname 'Z')
)
```

utils.ijs (only the finally used parts)

```
NB. replaces ). in the given noun by )
replaceFakeParen =: ] #~ (1&,@:-.@:(2 ').'&-: \ ]))`(]&'')@.((,0) -: $)

load 'regex'
NB. extracts the modified names out of a string of J code
extractModifiedNames_regexp =: '([[:alpha:]][[:alnum:]_]*)[[:space:]]*=:'
extractModifiedNames =: 1{"1 extractModifiedNames_regexp&(rxmatches rxfrom ])
```

test_replaceFakeParenMultiRow from utils_test.ijs

```
fp_multiRow_input =: noun define
multiple
rows
  ).
no new line ).
no dot )
and no paren .
done
)

fp_multiRow_output =: noun define
multiple
rows
)
fp_multiRow_output =: fp_multiRow_output , '  )', (10{a.),'no new line )', noun define

no dot )
and no paren .
done
)


test_replaceFakeParenMultiRow =: 3 : 0
  assert fp_multiRow_output -: replaceFakeParen fp_multiRow_input
)
```

test_lwhen_collectionWriteonly from layer2_test.ijs

```
NB. describe the writeonly layer
'Collection' lwhen 'Writeonly'
  add =: 3 : '# items'
  remove =: 3 : '# items'
  destroy =: 3 : '0'
)

'Collection' lwhen 'Writeable'
  add     =: 3 : '# items =: (< y) , items'
  remove  =: 3 : '# items =: items -. < y'
  destroy =: codestroy
)

test_lwhen_collectionWriteonly =: 3 : 0
  cocurrent <'base'
  'Collection' lenable 'Writeable'
  assert 2 = #inspect__C1 ''
  add__C1 'bar'
  assert 3 = #inspect__C1 '' NB. modification

  'Collection' lenable 'Writeonly'
  add__C1 'bar'
  assert 3 = #inspect__C1 '' NB. no modification
)
```