CallGraph demo

File format:

```
|<returnType>:<owner+methodName>:<argType>*
|<receiverTypeOrStaticMethod>
|<arguments>
|<callerWithLineNumber>
```

Opening files

- open a fresh image (set resolution to 1024x768 AND set fonts to MEDIUM)
- optionally load Freq

```
Gofer new
    url: 'http://smalltalkhub.com/mc/spasojev/FrequentlyUsedMethodsPluginForNautilus/main';
    package: 'ConfigurationOfFreQ';
    load.
(Smalltalk at: #ConfigurationOfFreQ) loadDevelopment.
```

- show where the image and changes files are
- copy the Calls.txt file to the image folder
- open the image; demo the Workspace, Transcript and Inspector
- demo “do it”, “print it”, “inspect it” with menu and shortcuts
- open an inspector on the file:

```
FileStream fileNamed: 'Calls.txt'
```

- view the contents in the inspector

System Browser — creating packages and classes

- navigate to implementers of `fileNamed:`
- demo the system browser
- add the `CallGraph` package
- define the `CallGraph` class & add a class comment
- define the `from:` initialization method and the `calls` accessor
- display the number of calls in the graph

```
| cg |
cg := CallGraph new from: (FileStream fileNamed: 'Calls.txt') contents.
cg calls size
```

- define `CallGraph class>>fromFile:` and simplify the code snippet

Test Cases

- now turn it into a test
- put a 5-line snippet of the Calls.txt into a class-side `example` method
- write a test `CallGraphTest>>testNumberOfCalls`
- show the test passes both in the system browser and the test runner
- show what happens if you break the test

Monticello

- save package with Monticello
- quickly show Smalltalkhub

Modelings Calls and Methods; Collections

- create a Call object for each log line

```
CallGraph>>from: aString
    calls := (Character cr split: aString)
            collect: [ :each | self createCall: each ]
```

- introduce the Collection classes (slides)
- introduce Booleans
- define the `createCall:` method

```
CallGraph>>createCall: callString
    | fields callee |
    fields := $| split: callString.
    self assert: fields size = 5.
    self assert: (fields at: 1) size = 0.
    callee := self getMethod: (fields at: 2).
    ^ Call new callee: callee
    "TODO -- handle the remaining fields!"
```

- introduce `Call` and `JMethod` classes
- initialize methods and define the accessor

```
CallGraph>>initialize
    methods := Dictionary new
```

- implement `getMethod`

```
CallGraph>>getMethod: signature
    | fields methodName |
    fields := $: split: signature.
    methodName := fields at: 2.
    ^ methods at: methodName
        ifAbsentPut: [ JMethod new name: methodName ]
```

- create the `JMethod>>name:` and `Call>>callee:` methods from the debugger

```
| cg |
cg := CallGraph fromFile: 'Calls.txt'.
cg methods size
```

Duck Typing

- again see that there is an assertion failure — debug to find that not all log lines are valid

```
CallGraph>>from: aString
    calls := ((Character cr split: aString)
        select: #notEmpty)
        collect: [ :each | self createCall: each ]
```

- note that symbols also understand `value:` so can (sometimes) be used in place of blocks
- show we can now compute the number of methods

```
(CallGraph fromFile: 'Calls.txt') methods size. 164
```

Modeling Classes

- create JClass objects for argument and return types
- introduce classes Dictionary

```
(CallGraph fromFile: 'Calls.txt') classes size. -> 30
```

task 3: number of methods with >0 args query in inspector: 72

task 4: number of methods with >1 args query: 18

MORE TASKS Number of Static methods Potentially Polymorphic methods Polymorphic call sites (methods called w >1 arg types) Most frequently called method Depth of call graph Root of call graph Methods called from more than 1 caller

Queries

```
(CallGraph fromFile: 'Calls.txt') calls size. "-> 2475"
(CallGraph fromFile: 'Calls.txt') methods size. "-> 168"
(CallGraph fromFile: 'Calls.txt') classes size. "-> 75"
((CallGraph fromFile: 'Calls.txt') methods select: #isStatic) size. "-> 10"
((CallGraph fromFile: 'Calls.txt') methods select: [ :m | m calls size > 1 ]) size. "-> 141"
((CallGraph fromFile: 'Calls.txt') methods select: #isPolymorphic) size. "-> 10"
```

TO DO: - owner field (static/class) - actual argument types - caller w line number

---