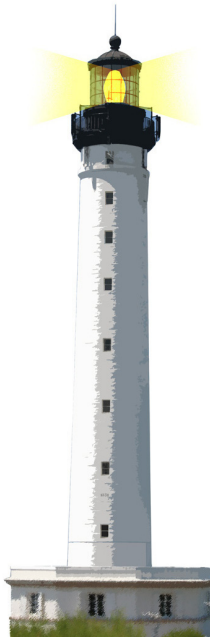
Xtreme Test Driven Development

Getting a productivity boost

S. Ducasse



Outline

- TDD on steroids
- Live programming
- Smart tools
- Absolutely gorgeous development flow

Principle

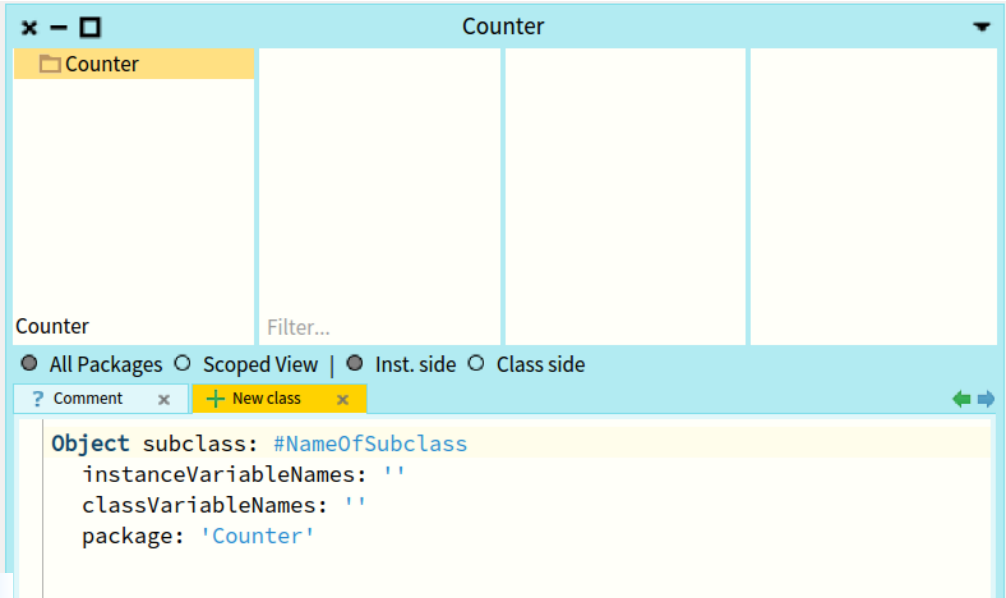
Do **not break** the flow

- Write a test
- When it breaks, define the method **on the fly in the debugger**
- Resume and continue until test is green

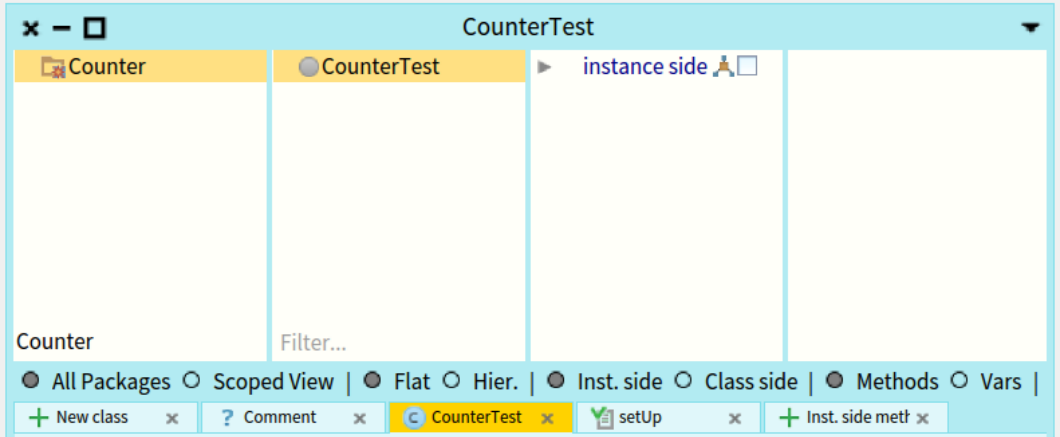
Studying an example

- A dead simple counter. Nothing simpler.
- Focus on essence of the process!
- You can do it.

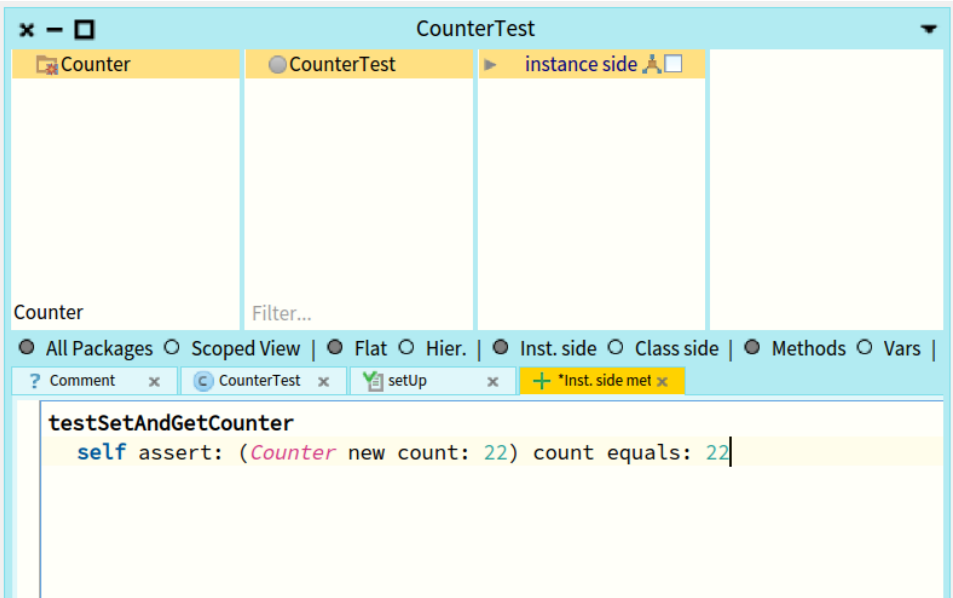
An empty package



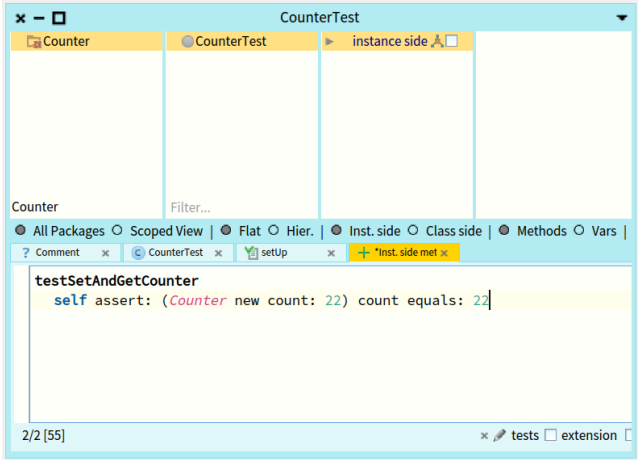
An empty test case class



A first test


CounterTest
Counter
CounterTest
instance side
Counter
Filter...
All Packages
Scoped View
Flat
Hier.
Inst. side
Class side
Methods
Vars
Comment
CounterTest
setUp
*Inst. side met
testSetAndGetCounter
self assert: (Counter new count: 22) count equals: 22

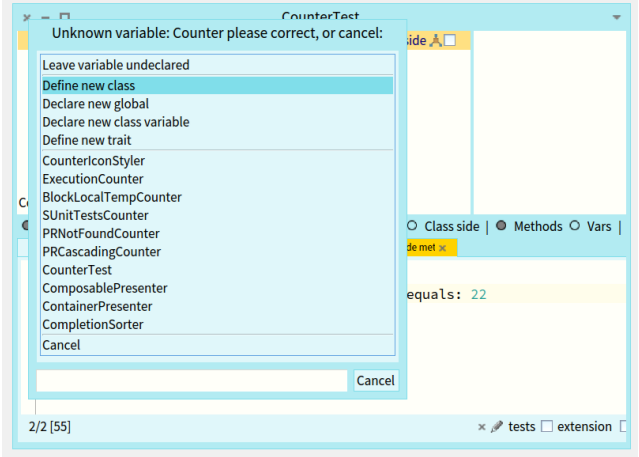
A first test



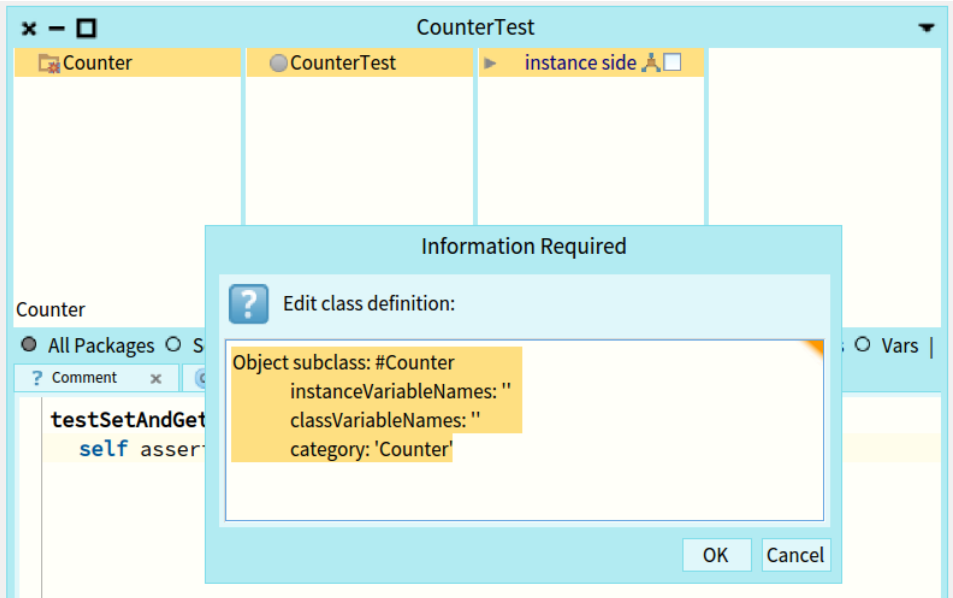
- Method is about to be compiled
- The system knows the class does not exist!

Define a class

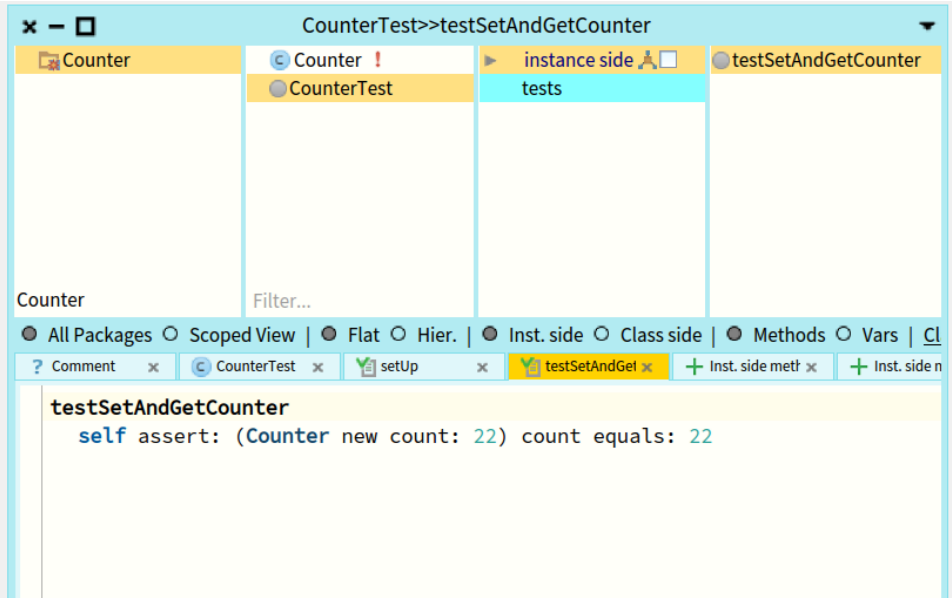
- At compile time...



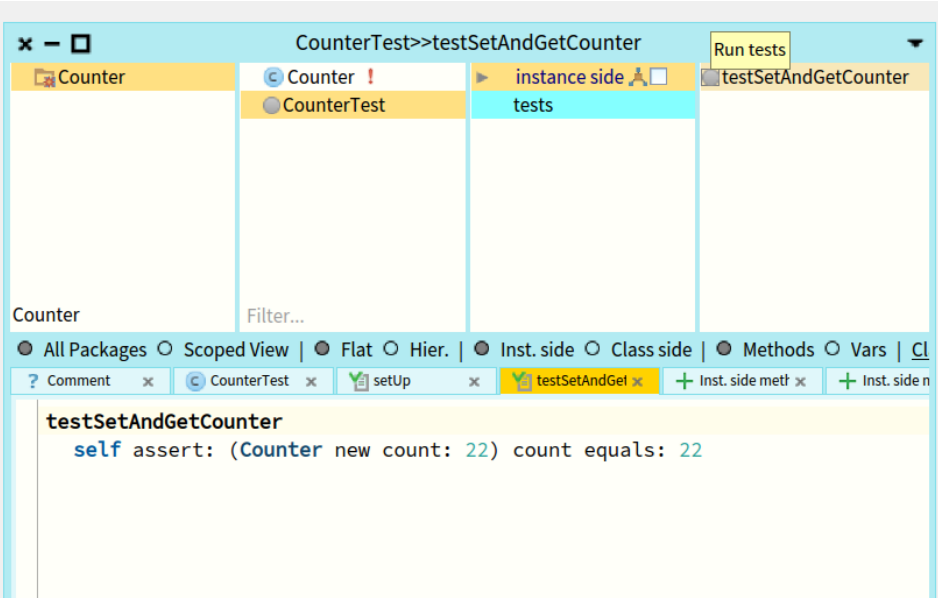
Define a class (II)



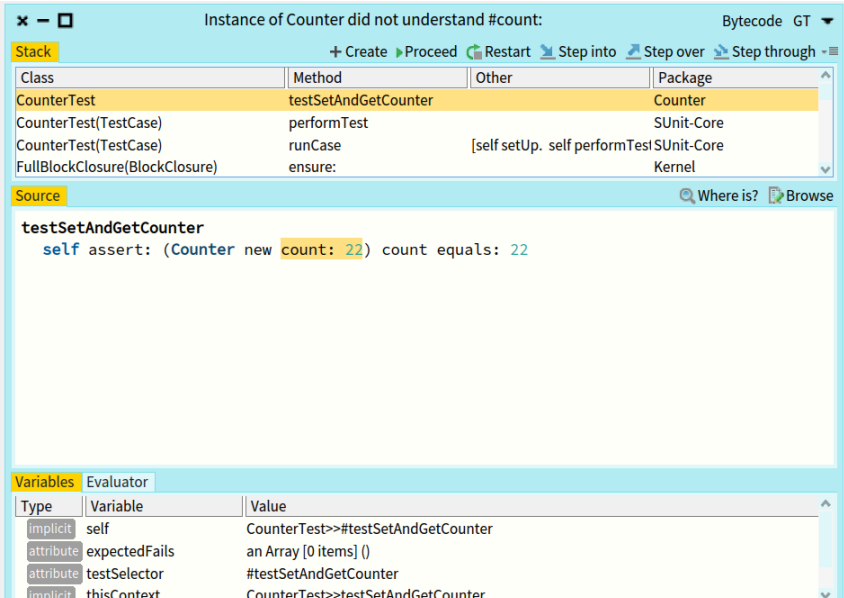
Test defined but not executed



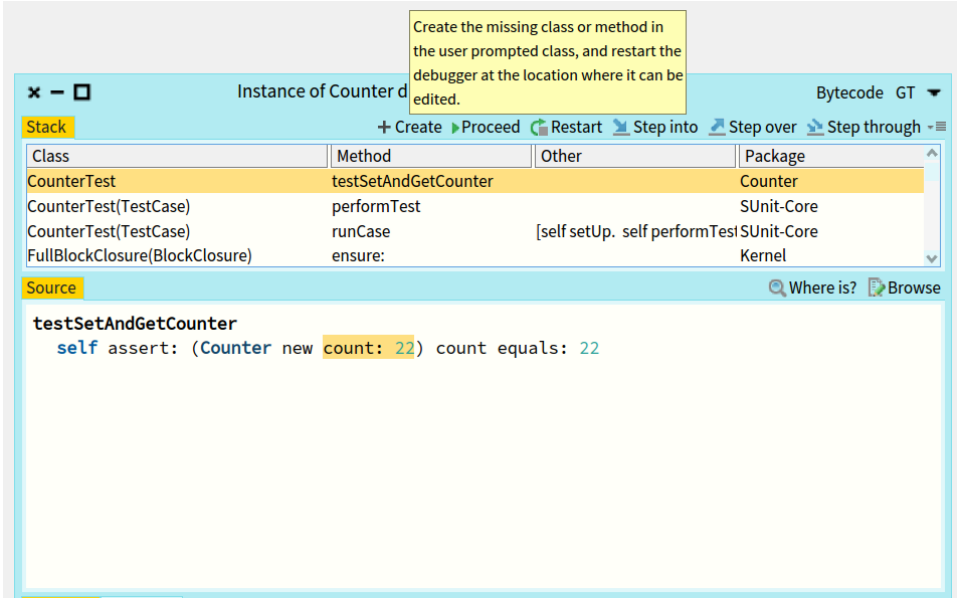
Running the test



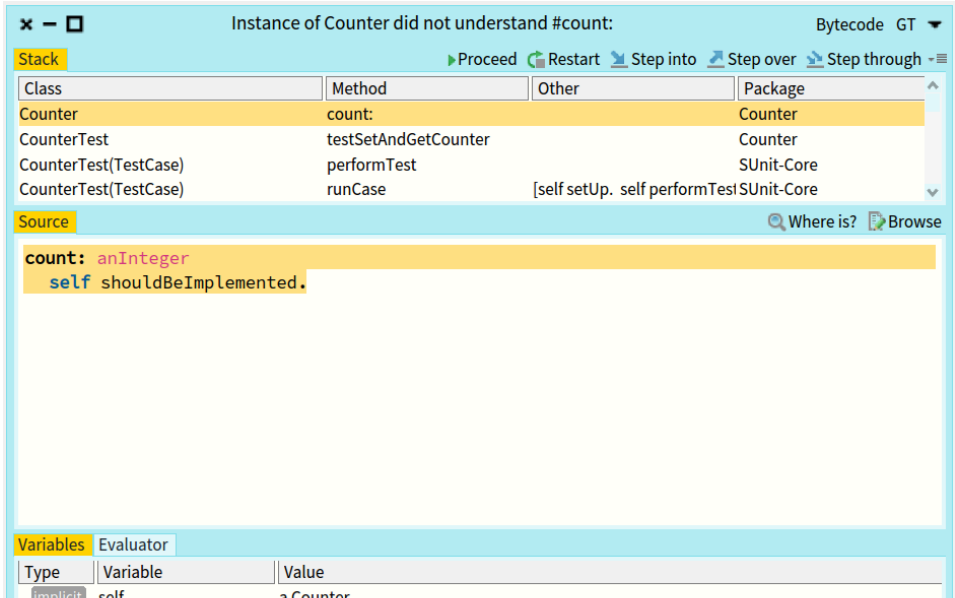
First Error


Instance of Counter did not understand #count:
Bytecode GT
Stack
Create
Proceed
Restart
Step into
Step over
Step through
Class
Method
Other
Package
CounterTest
testSetAndGetCounter
Counter
CounterTest(TestCase)
performTest
SUnit-Core
CounterTest(TestCase)
runCase
[self setUp. self performTest
SUnit-Core
FullBlockClosure(BlockClosure)
ensure:
Kernel
Source
Where is?
Browse
testSetAndGetCounter
self assert: (Counter new count: 22) count equals: 22
Variables
Evaluator
Type
Variable
Value
implicit
self
CounterTest>>#testSetAndGetCounter
attribute
expectedFails
an Array [0 items] ()
attribute
testSelector
#testSetAndGetCounter

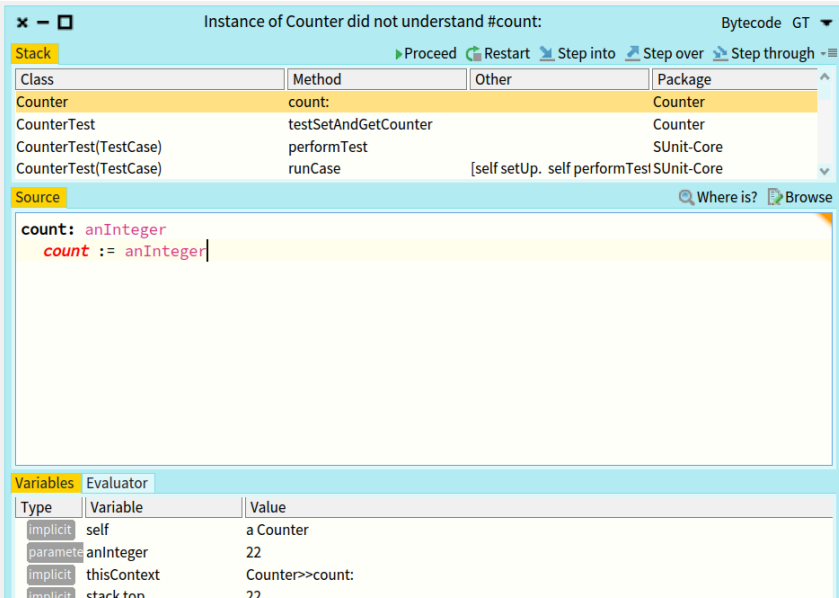
Create a method on the fly



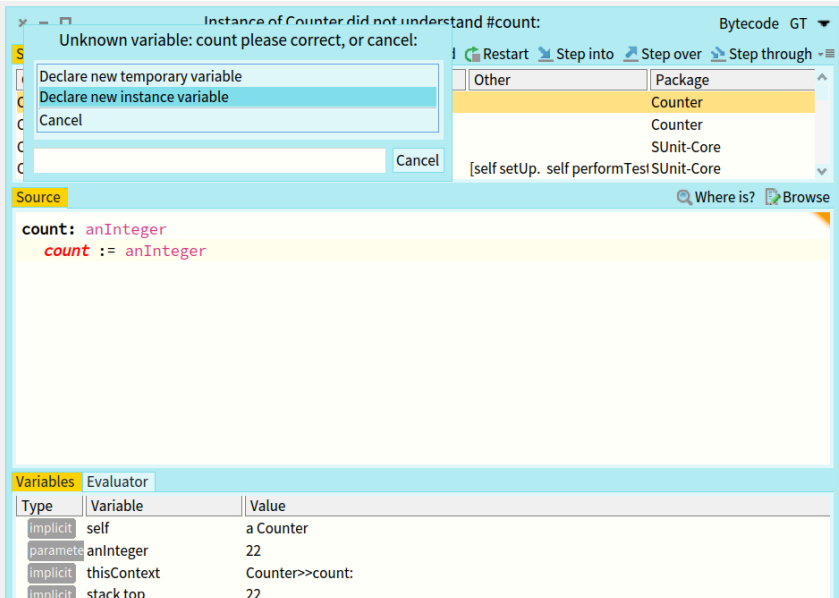
Create a method on the fly (II)



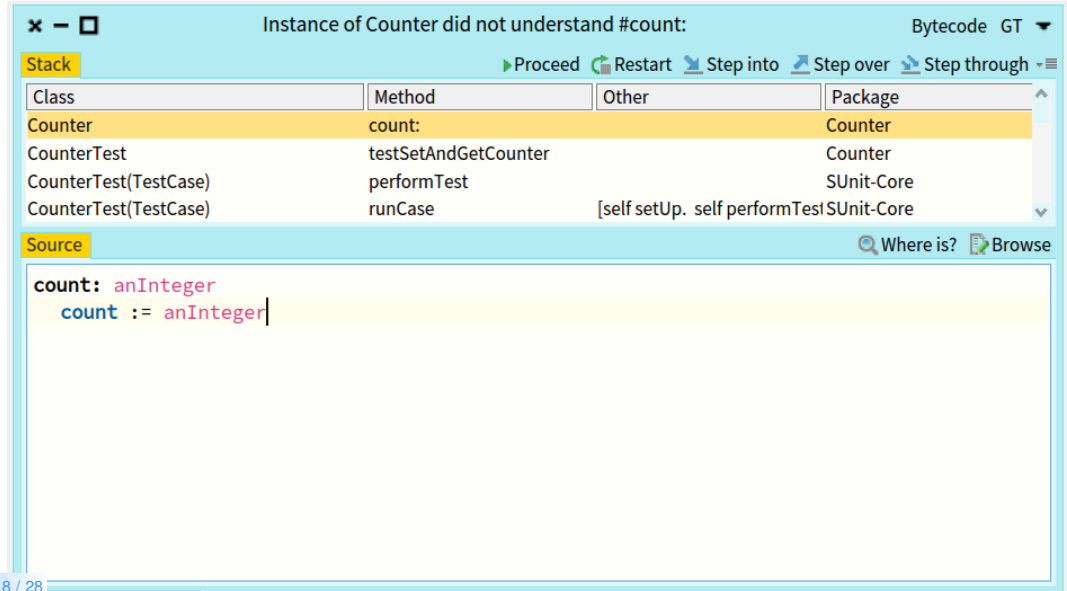
Edit the method in the debugger (III)



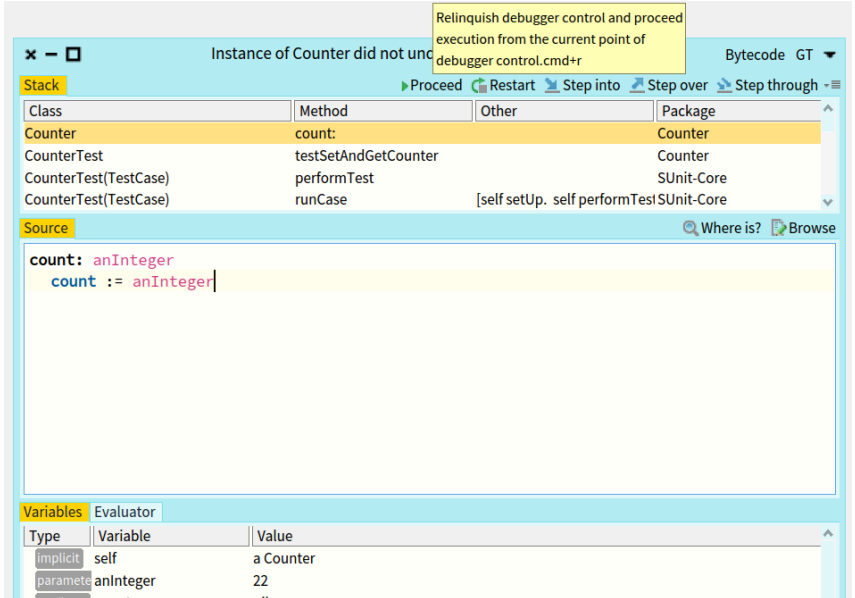
Add an instance variable on the fly



Compile....


Instance of Counter did not understand #count:
Bytecode GT
Stack
Proceed
Restart
Step into
Step over
Step through
Class
Method
Other
Package
Counter
count:
Counter
CounterTest
testSetAndGetCounter
Counter
CounterTest(TestCase)
performTest
SUnit-Core
CounterTest(TestCase)
runCase
[self setUp. self performTest
SUnit-Core
Source
Where is?
Browse
count: anInteger
count := anInteger

Continue the execution...



Supporting the programmer flow

- The system created a new method
- Removed the stack element with Error
- Replaced it with a call to the new method
- Relaunched execution
- We edited it and recompiled the method
- Continued execution

New method

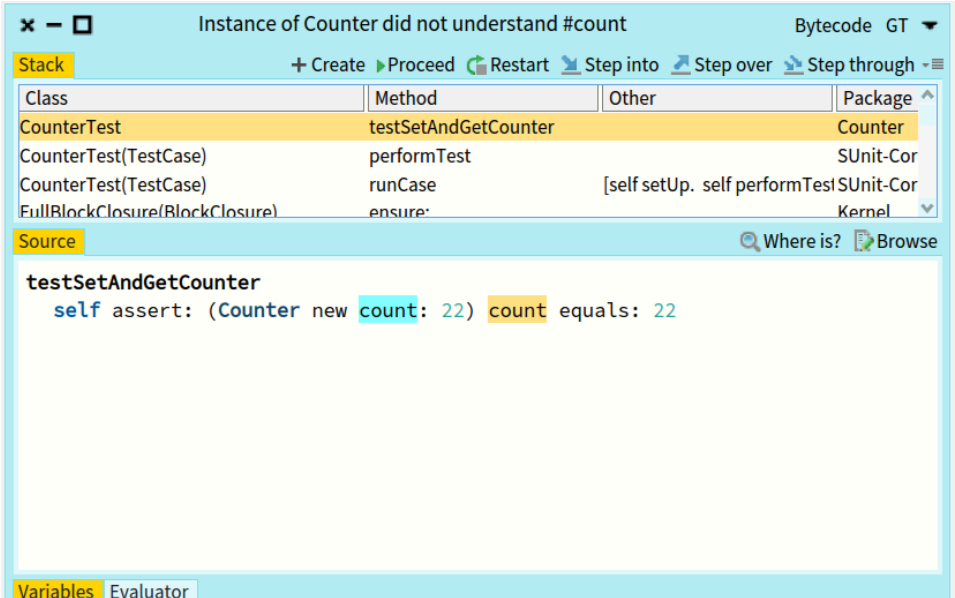
The system created a new method

- Removed the stack element with Error
- Replace it with a **call** to the new method

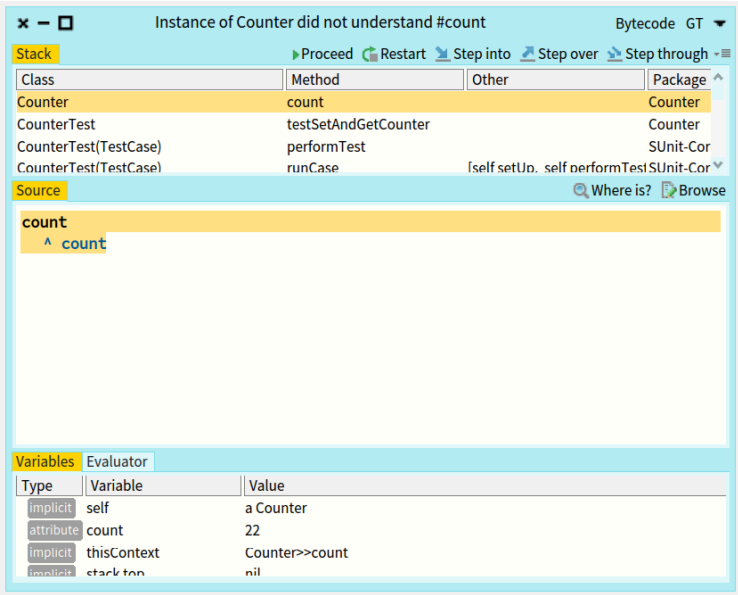
```
count: anInteger
  self shouldBeImplemented
```

- shouldBeImplemented is just an exception so that the debugger stops again

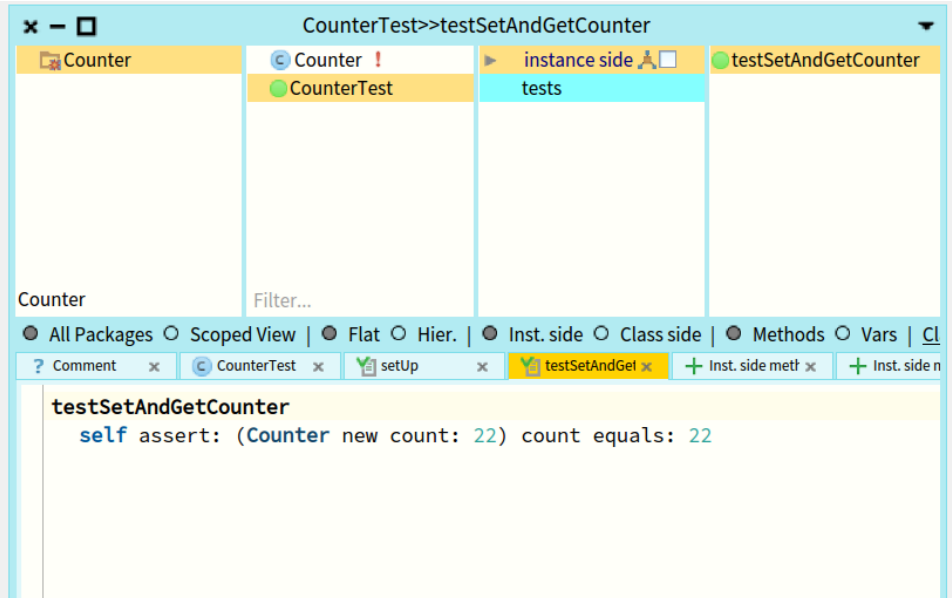
Same story....


Instance of Counter did not understand #count
Bytecode GT
Stack
Create
Proceed
Restart
Step into
Step over
Step through
Class
Method
Other
Package
CounterTest
testSetAndGetCounter
Counter
CounterTest(TestCase)
performTest
SUnit-Cor
CounterTest(TestCase)
runCase
[self setUp. self performTes
SUnit-Cor
FullBlockClosure(BlockClosure)
ensure:
Kernel
Source
Where is?
Browse
testSetAndGetCounter
self assert: (Counter new count: 22) count equals: 22
Variables
Evaluator

Debugger also precompiles methods



Test is green



One Cycle

- Run all the tests
- Ready to commit
- New test

Why XTDD is powerful

- Avoid guessing context when coding
- Much much better context
  - inspect that **specific** instance state
  - talk to that **specific** object
- Inspectable / interactable context
- Tests are not a side effect artefact but the driving force

Protip from expert Pharo developers

- Get as fast as possible one object
- Cristalize your scenario with a test
- Xtreme TDD
- Loop

A course by

S. Ducasse, L. Fabresse, G. Polito, and Pablo Tesone