# Chest

Adrien Vanègue, Pierre Laborde and Steven Costiou

March 12, 2024

Copyright 2023 by Adrien Vanègue, Pierre Laborde and Steven Costiou.

The contents of this book are protected under the Creative Commons Attribution-ShareAlike 3.0 Unported license.

You are **free**:

- to Share: to copy, distribute and transmit the work,
- to Remix: to adapt the work,

Under the following conditions:

- Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
- Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page: http://creativecommons.org/licenses/by-sa/3.0/

Any of the above conditions can be waived if you get permission from the copyright holder. Nothing in this license impairs or restricts the author's moral rights.



Your fair dealing and other rights are in no way affected by the above. This is a human-readable summary of the Legal Code (the full license): http://creativecommons.org/licenses/by-sa/3.0/legalcode

Layout and typography based on the sbabook  ${\tt LAT}_{\! E\! X}$  class by Damien Pollet.

# Contents

|     | Illustrations           | ii |
|-----|-------------------------|----|
| 1   | Introduction            | 1  |
| 1.1 | Original repository     | 2  |
| 1.2 | Install Chest           | 2  |
| 1.3 | Open Chest              | 2  |
| 1.4 | More Details            | 2  |
| 2   | API Description         | 13 |
|     | Chest class-side API    |    |
| 2.2 | Chest instance-side API | 15 |

# Illustrations

| 1-1  | Chest overview  | 1  |
|------|---|----|
| 1-2  | Chest in debug menu   | 2  |
| 1-3  | Chest debugger extension in the Pharo settings                                  | 3  |
| 1-4  | Store object context menu, inside the debugger                                  | 4  |
| 1-5  | Store object popup  | 4  |
| 1-6  | Chest tree view inside the debugger, after having stored an object              | 5  |
| 1-7  | Load object context menu, inside the debugger                                   | 5  |
| 1-8  | Load object popup   | 6  |
| 1-9  | Usage of the Chest variable inside the debugger, after having loaded the object | 6  |
| 1-10 | Chest variable, in the debugger inspector                                       | 7  |
| 1-11 | Variable from Chest playground, not visible in the debugger inspector $\ldots$  | 7  |
| 1-12 | Paste object from Chest context menu  | 8  |
| 1-13 | Injected code for Chest, after having pasted the object                         | 9  |
| 1-14 | Copy object to Chest Clipboard context menu                                     | 10 |
| 1-15 | Paste last object from Chest Clipboard context menu                             | 10 |
| 1-16 | Injected code for Chest, after having pasted the object from Chest Clipboard    | 11 |

# CHAPTER

# Introduction

Chest allows you to store objects from anywhere, to keep them around to check equality...

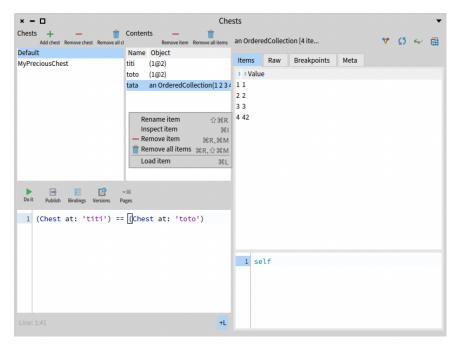


Figure 1-1 Chest overview

# 1.1 Original repository

Link to original repositoryhttps://github.com/dupriezt/Chest

## 1.2 Install Chest

```
Metacello new
   baseline: 'Chest';
   repository: 'github://pharo-spec/Chest';
   load.
```

## 1.3 Open Chest

Chest is available in the world menu of Pharo. It is available as an entry of the Debug menu

| 🔰 Pharo 🛛 Brows | se Debug                            | Sources  | System             | Library | Windows  | Help |
|-----------------|-------------------------------------|--|--------------------|---------|----------|------|
| Pł              | Disabl<br>Remov<br>Enable<br>Enable | e Breakpoints<br>e all Breakpoi<br>ve all Breakpo<br>e all Breakpoir<br>e all break/ins              | nts<br>ints<br>its |         |          |      |
|                 | Start p<br>Reset                    | er<br>profiling <u>all Pro</u><br>profiling <mark>Store</mark><br>Counter <sup>arour</sup><br>Chests | objects from       |         | eep them |      |

Figure 1-2 Chest in debug menu

You can also enable it as a debugger extension in the debugging settings of Pharo: In the debugger, Chest will provide a view as a tree of all chests with the objects inside.

Maybe we need an image of the tree view here?

## 1.4 More Details

#### Name (= ID)

Each Chest instance has an ID (String). These IDs are unique. Two chests cannot have the same ID.

## **Default Chest**

This is an instance of Chest that can be interacted with in the same way as any other Chest by sending the messages to the Chest class.

#### 1.4 More Details

| × - 🗆   | Settings Browser     |            |                 |                | •             |
|---|----------------------|------------|-----------------|----------------|---------------|
| Expand all Search for: Hit return to accept                             |                      | V 🗌 Regexp | Choose packages | Store Settings | Load Settings |
| Enable cmd-dot interrupt key  |                      |            |                 |                | ^             |
| ▶ 🛠 Halt Cache  |                      |            |                 |                |               |
| Filter the debugger stack   | V                    |            |                 |                |               |
| 🔻 📯 Debugger Extensions   |                      |            |                 |                |               |
| Sreakpoints   |                      |            |                 |                |               |
| ▶ 🛠 Bytecode  |                      |            |                 |                |               |
| ▼ 🍄 Chest   |                      |            |                 |                |               |
| Position Activate the tool tab in the right column.                     | e debugger           |            |                 |                |               |
| Show in debugger  |                      |            |                 |                |               |
| ► 🛠 Sindarin  |                      |            |                 |                |               |
| P Deprecation handling  |                      |            |                 |                |               |
| 🚦 Write message to debug log file when fall into deb                    |                      |            |                 |                |               |
| Log file name   | PharoDebug.log       |            |                 | *              |               |
| Debugger selection strategy   | OupsDebuggerSelector | V          |                 |                | ~             |
| Show in debugger<br>Activate the tool tab in the debugger right column. |                      |            | All pa          | ckages         |               |

Figure 1-3 Chest debugger extension in the Pharo settings

### Weak Chests

Chest weak

is a way to access the class WeakChest. This class can be interacted with the same way as you would with the Chest class.

 $\tt WeakChest$  has its own default chest, and can be used to manipulate this default chest or to create new weak chests.

#### **Commands in context**

#### Store object into chest

If you right-click on a code presenter with an SpCodeInteractionModel or StDebuggerContextInteractionModel (e.g: playground, debugger etc.), you can evaluate an expression and store the result in the chest of your choice, with the name of your choice:

#### Load object from a chest into a playground or a debugger

It's also possible to load objects from a chest into these code presenters:

And then variables can be seen from any other context if it has been loaded in a debugger and these variables can be seen in the debugger inspector:

Chest, as a debugger extension, provides a playground. All bindings between this playground and the debugger selected context are shared. So: all variables defined in this playground are recognized by the debugger and all variables from the debugger's selected context or loaded from Chest into the debugger are recognized by the playground. However, only the variables loaded from Chest (via the load menu entry described above) are displayed in the debugger inspector:

| × - 0                                    |  | SindarinDebuggerTest>>methodWithTwoAssignments   |         |  | •         |
|--|--|--|---------|--|-----------|
| Stack                                    |  |  | 11 P    | Chest  |           |
| Class                                    | Method   | Package  |         | + Bamove all chests  |           |
| SindarinDebuggerTest                     | methodWithTwoAssignments   | Sindarin-Tests   |         | Chests Name  |           |
| UndefinedObject                          | Doit   |  |         | v III a ChestContentDictionary('tata'->an OrderedC Default |           |
| CompiledNethod                           | value/WithReceiver:arguments:  | Kernel   |         | V (192) titi   |           |
| SpCodeDebugItCommand                     | [ aCompiledMethod valueWithReceiver: anObjection of the second se |  |         | * (1@2) toto   |           |
| FullBlockClosure (BlockClosure)          | [self value. "IMPORTANT: Do not step over next lin   | ne of code. See m Kernel   |         | V an OrderedCollection(1.2.3.42) tata                      |           |
|  |  |  |         | (III) a ChestContentDictionary() MyPreciousChest           |           |
|  |  |  |         |  |           |
| Proceed Into Over Through Run to Restart | S C + C<br>Return Where is? Cruite Advanced Step   |  |         |  |           |
| 1 methodWithTwoAssignments               |  |  |         |  |           |
| 2  |  |  |         |  |           |
| 3   a  <br>4 a := 1.                     |  |  |         |  |           |
| 4 3 := 1.<br>5 3 := 5.                   |  |  |         |  |           |
| 6 * Point x: 5 y: '3' asInte             | ter  |  |         |  |           |
|  |  |  |         |  |           |
|  |  | Doit HD  |         | ▶ 📑 🔛 📑 -=<br>Doit Publish Bindings Versions Pages         |           |
|  |  | 🗰 Debug it 🛛 🖓 St D  |         | vvv volan anargi vesoni vagei                              |           |
|  |  | See Inspect it See   |         | 1  |           |
|  |  | Print it     Store the result of the selected     expression in the chest of your choice |         |  |           |
|  |  | Store object in a chest stc_stc  | 1       |  |           |
|  |  | Code search  |         |  |           |
|  |  | Load object from chest   |         |  |           |
|  |  | Paste object from chest > >  |         |  |           |
|  |  | Run to   |         |  |           |
|  |  | 🙍 Next call in receiver  |         |  |           |
|  |  | 1 Skip   |         |  |           |
|  |  | To return<br>Next call in class  |         |  |           |
|  |  | To method entry  |         |  | +L        |
|  |  | <ul> <li>Exp.] Skip up to (stop before)</li> </ul>                                       |         |  |           |
|  |  | I IFwn 1 humo to caret   |         |  | ∀ Ø ∻ @ × |
| a SindarinDebuggerTest (Sin              |  | Next instance creation ect (nil)   |         |  | ¥ © ↔ ⊞ × |
|  | : Value  | C Find 36F joints Meta   |         |  |           |
|  | SindarinDebuggerTest   | Copy %C  | + Value |  |           |
| temp. var Σa                             |  | 3 Cut 2€X<br>1 Paste 2€V   | nil     |  |           |
|  | 18<br>18   | Paste 30V  |         |  |           |
|  | 50.<br>3Å  | ✓ Accept 365   |         |  |           |
|  | 5  | × Cancel 35L   |         |  |           |
|  | ,<br>SindarinDebuggerTest>>methodWithTwoAssignments  |  |         |  |           |
|  |  |  |         |  |           |
|  |  |  |         |  |           |
|  |  |  |         |  |           |
|  |  | 1 self   |         |  |           |
|  |  |  |         |  |           |
|  |  |  |         |  |           |
|  |  |  |         |  |           |
|  |  |  |         |  |           |

 Figure 1-4
 Store object context menu, inside the debugger

| × = Choose a variable<br>St                                |                               |  | 12                       | <b>9</b> | Chest   |           |
|--|-------------------------------|--|--------------------------|----------|---|-----------|
| St tutu  | Name Object<br>titi (1@2)     |  | *1                       |          | + +   |           |
| Default  | toto (1@2)                    | d Package  |                          |          | Add chest Remove all chests   |           |
| MyPreciousChest  | tata an OrderedCollection[12] | dWithTwoAssignments Sindarin-Tests   |                          |          | Chests Name   |           |
| ¢  |                               | hthReceiver:arguments: Renel<br>mpiledMethod valueWithReceiver: anObject arguments: (5 Spec2-Code-Com<br>lae. "IMPORTANT: Do not step over next line of code. See m Kernel | mands                    |          | **         (m) 2. ChestContentDictionary(Tata'>-an OrderedC Default           **         (m) 2. ChestContentDictionary(Tata'>-an OrderedC Default |           |
| H - H<br>Add chest Remove chest. Remove all chests<br>Pa   | Renove item Remove all items  | 6<br>hanced Dap  |                          |          |   |           |
| Confirm Cancel   |                               |  |                          |          |   |           |
| 3   a  <br>4 a := 1.<br>5 a := 5,<br>6 * Point x: 5 y: '3' | asInteger                     |  |                          |          |   |           |
|  |                               |  |                          |          | ► E E C -=<br>Do k Publish Bindings Versions Fages  |           |
|  |                               |  |                          |          | 1   |           |
|  |                               |  |                          |          |   |           |
|  |                               |  |                          |          |   | •L        |
| a SindarinDebuggerTest (Sin                                |                               | Ø 🛩 🕮  | an UndefinedObject (nil) |          |   | ∀ Ø ∾ ⊞ × |
| Type : Variable     implicit                               | Value SindarinDebuggerTest    |  | Raw Breakpoints Meta     |          |   |           |
| temp. var Σa   | 1                             |  | Variable V               | alue     |   |           |
| inst var @ testSelector                                    | ni                            |  | © self nil               |          |   |           |
| inst.var @ expectedFails                                   | nil                           |  |                          |          |   |           |
| inst. var @ testObjectPoint                                | nil                           |  |                          |          |   |           |
| implicit I stackTop<br>implicit I thisContext              | 5                             |  |                          |          |   |           |
| implicit (C) thisContext                                   | SindannbebuggerTest>>m        | ethodWithTwoAssignments  |                          |          |   |           |
|  |                               |  |                          |          |   |           |
|  |                               |  |                          |          |   |           |
|  |                               |  | 1 self                   |          |   |           |
|  |                               |  |                          |          |   |           |
|  |                               |  |                          |          |   |           |

Figure 1-5 Store object popup

| × - 0   | Sindar  | inDebuggerTest>>methodWithTwoAssignments |         |  | •   |
|---|---|--|---------|--|---|
| Stack   |   |  | ↓ª ♀    | Chest  |   |
| Class   | Method  | Package                                  |         | + The Add chest Remove all chests  |   |
| SindarinDebuggerTest  | methodWithTwoAssignments  | Sindarin-Tests                           |         | Chests   | Name  |
| UndefinedDigect<br>CompiledNethod<br>SpCodeDebugtCommand<br>FullBlockClosure (BlockClosure) | Dott<br>valueWithReceiver.arguments:<br>[ aCompiledMethod valueWithReceiver: anObject argumen<br>[self value. *IMPORTANT: Do not step over next line of code. S |  |         | ▼ (iii) a ChestContentDictionary('tata'->an Ordered<br>♥1<br>♥(1@2)<br>♥(1@2)<br>♥(1@2)<br>♥ (1@2)<br>♥ an OrderedCollection(1 2 3 42)<br>(iii) a ChestContentDictionary() | IC Default<br>tubu<br>Bili<br>toto<br>tata<br>MyPreciousChest |
| Proceed Into Over Through Ranto Roslart Return Where's                                      | + Content Advanced Step   |  |         |  |   |
| 1 methodwithTwoAssignments  |   |  |         |  |   |
| 2<br>3   a  <br>4 a := 1.<br>5 a := 5.<br>6 ^ Point x: 5 y: '3' asInteger                   |   |  |         |  |   |
|   |   |  |         | ▶ 📑 🚺 📴 + II<br>De R. Publish Sindings Versions Pages  |   |
|   |   |  |         | 1  |   |
|   |   |  |         |  |   |
|   |   |  |         |  | <del>1</del>  |
| a SindarinDebuggerTest (Sin   |   | 💋 👾 🛗 an UndefinedObject (nil)           |         |  | 🕈 Ø 🏎 🖽 ×   |
| 1 Type 1 Variable 1 Value   |   | Raw Breakpoints Meta                     |         |  |   |
| implicit @ self SindarinDebugg<br>temp.var Σ a 1  | erTest  | : Variable                               | : Value |  |   |
| inst.var @ testSelector nil   |   | © self                                   | nil     |  |   |
| inst. var @ expectedFails nil   |   |  |         |  |   |
| inst.var @ testObjectPoint nil  |   |  |         |  |   |
| implicit Σ stackTop 5<br>implicit @ thisContext SindarinDebugg                              |   |  |         |  |   |
| implicit () thisContext SindarinDebugg  | erTest>>methodWithTwoAssignments  |  |         |  |   |
|   |   |  |         |  |   |
|   |   | 1 self                                   |         |  |   |
|   |   |  |         |  |   |
|   |   |  |         |  |   |

**Figure 1-6** Chest tree view inside the debugger, after having stored an object

| × - 0  |   | SindarinDebuggerTest>>methodi | WithTwoAssignments   |   |  |      | •         |
|--|---|-------------------------------|--|---|--|------|-----------|
| Stack  |   |                               | 42   |   |  |      |           |
| Class  | Method  | Package                       |  | +<br>Add chest Remove                           | al deate   |      |           |
| SindarinDebuggerTest   | methodWithTwoAssignments  | Sindarin-Tests                |  | Chests  | 41.085   | Name |           |
| UndefmedDiptet<br>CompileOHterbod<br>SpCodeDebugitCommand<br>FullBlockClosure (BlockClosure)   | Dott<br>valueflithReceiver:anguments:<br>[ accompliedMethod valueflithReceiver: anOb<br>[ self value. "IMPORTANT: Do not step over next |                               |  | ♥ (=) a Chest<br>♥1<br>♥(1@2<br>♥(1@2<br>♥ an O |  |      |           |
| Proceed loss Over Through Run to Roslart Return Wh   | Q         +         6           here is?         Create         Advanced Step   |                               |  |   |  |      |           |
| <pre>i etboldtthussignments</pre>  |   |                               | Debug it 3ED<br>roppect it 8E1<br>init it 3E1<br>Profile it<br>Store object in Lead an object from .<br>Sould object from chest<br>Paste object from chest<br>Run to weat call in receiver |   | The Contract of the second sec |      | ×         |
| a SindarinDebuggerTest (Sin  |   | Ω ↔ ⊞ × 00                    | topy atC   |   |  |      | 🕈 Ø 😔 🖽 × |
| Type         Variable         Value           mplot         €         Schedroch           mplot         €         1           mplot         €         2           mplo | ugger Fest  | ∏<br>∡∧<br>×(                 | Sate Sate Sate Sate Sate Sate Sate Sate  | e   |  |      |           |

Figure 1-7 Load object context menu, inside the debugger

| x - 0   |  |   | >>methodWithTwoAssignments |             |   | *         |
|---|--|---|----------------------------|-------------|---|-----------|
| x - Choose a variable na<br>St<br>Default           |  |   |                            | 1. 94       | Chest   |           |
|   | Name Object Variable Nam<br>tutu 1 myVar |   |                            | +1 <i>4</i> | + +   |           |
|   | titi (1@2) titi                          | d Package   |                            |             | Add chedt Remove all cheds  |           |
|   | toto (1@2) toto                          | dWithTwoAssignments Sindarin-Test   | i.                         |             | Chests Name   |           |
| Add their Terrine that Terrine at theirs            | tata an OrderedCo tata                   | Noticetiercarguments Fored<br>mplicitization valuetifications: and/opt anguments (15pc)2.0646 (<br>Jun: "MPCRTAT: Do not stop over next line of code. See mixered<br>Security | ommands                    |             | VIII a Constructionary Chair - San Ordereck Default           Im a Chest ContextDictionary Chair - San Ordereck Default |           |
| 3   a  <br>4 a := 1.<br>5 a := 5.                   |  |   |                            |             |   |           |
| 6 * Point x: 5 y: '3' a                             | sInteger                                 |   |                            |             |   |           |
|   |  |   |                            |             | ▶ 📑 🚺 📴 +11<br>Drit Publish Bindings Versions Pages   |           |
|   |  |   |                            |             | 1   |           |
|   |  |   |                            |             |   |           |
|   |  |   |                            |             |   | +L        |
| a SindarinDebuggerTest (Sin                         |  | 0 ~   | an UndefinedObject (nil)   |             |   | ∀ Ø 4 ⊞ × |
| Type Variable                                       | + Value                                  |   | Raw Breakpoints Meta       |             |   |           |
| implicit @ self                                     | SindarinDebuggerTest                     |   | : Variable                 | : Value     |   |           |
| temp. var Σ a<br>inst. var @ testSelector           | 1  |   | © self                     | nil         |   |           |
| inst.var @ testsetector<br>inst.var @ expectedFails | nii.                                     |   |                            |             |   |           |
| inst. var (© test0bjectPoint                        | nil                                      |   |                            |             |   |           |
| implicit I stackTop                                 | 5  |   |                            |             |   |           |
| implicit ( thisContext                              | SindarinDebuggerTest>>m                  | ethodWithTwoAssignments   |                            |             |   |           |
|   |  |   |                            |             |   |           |
|   |  |   |                            |             |   |           |
|   |  |   |                            |             |   |           |
|   |  |   | 1 self                     |             |   |           |
|   |  |   |                            |             |   |           |
|   |  |   |                            |             |   |           |
|   |  |   |                            |             |   |           |

Figure 1-8 Load object popup

| × - 0   |  | SindarinDebuggerTest>>methodWithTwoAssignments |         |  | -         |
|---|--|--|---------|--|-----------|
| Stack   |  |  | 11 94   | Chest  |           |
| Class   | Wethod   | Package  |         | + Brance all checks  |           |
| SindarinDebuggerTest  | methodWithTwoAssignments                       | Sindarin-Tests                                 |         |  |           |
| UndefinedObject   | Dolt   |  |         | Chests Name  |           |
| CompiledNethod  | valueWithReceiver:arguments:                   | Kernel   |         | v (III) a ChestContentDictionary("tata"->an OrderedC Default |           |
| SpCodeDebugitCommand  | [ aCompiledMethod valueWithReceiver: anOb      |  |         | V1 tutu  |           |
| FullBlockClosure (BlockClosure)   | (self value. "IMPORTANT: Do not step over next |  |         | V (1@2) titi   |           |
|   |  |  |         | ∀(1@2) toto  |           |
|   |  |  |         | ☆ an OrderedCollection(1 2 3 42) tata                        |           |
| ▶ ¥ ≝ ≥ -£ (  | 5 5 9 + 6                                      |  |         | (III) a ChestContentDictionary() MyPreciousChest             |           |
| Proceed Into Over Through Run to Res  | tart Refare Where's? Croste Advanced Step      |  |         |  |           |
| 1 methodWithTwoAssignments<br>2<br>3   a  <br>4 a := 1.<br>5 a := 5. myVar. a := titi<br>6 ^ Point x: 5 y: '3' asIn | (1692)<br>1                                    |  |         |  |           |
|   |  |  |         | ▶ 📑 🚺 📴 × II<br>Deit Publish Bindings Versions Pages         |           |
|   |  |  |         |  |           |
|   |  |  |         | 1  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  | _         |
|   |  |  |         |  | +L        |
| a SindarinDebuggerTest (Sin   |  | 💋 🦗 🛅 an UndefinedObject (nil)                 |         |  | 💙 🧭 😔 🗎 × |
| 1 Type 1 Variable   | 1 Value  | Raw Breakpoints Meta                           |         |  |           |
| implicit () self  | SindarinDebuggerTest                           | : Variable                                     | : Value |  |           |
| temp. var Σa  | 1  | © self   | nil     |  |           |
| inst.var (6) testSelector   | nil  |  |         |  |           |
| inst.var @ expectedFails  | nil  |  |         |  |           |
| inst.var @ testObjectPoint  | nil  |  |         |  |           |
| implicit I stackTop   | 5  |  |         |  |           |
| implicit (6) thisContext  | SindarinDebuggerTest>>methodWithTwoAssignments |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  | 1 self   |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |
|   |  |  |         |  |           |

**Figure 1-9** Usage of the Chest variable inside the debugger, after having loaded the object

| × - D  |   | ${\it SindarinDebuggerTest} >> methodWithTwoAssignments$ |         |  | •         |
|--|---|--|---------|--|-----------|
| Stack  |   |  | 12 📯    | Chest  |           |
| Class  | Method  | Package  |         | +      Add chest. Remove all chests                        |           |
| □ SindarinDebuggerTest                           | methodWithTwoAssignments                              | Sindarin-Tests   |         | Chests Name  |           |
| UndefinedObject                                  | Dolt  |  |         | (III) a ChestContentDictionary['tata'->an OrderedC Default |           |
| CompiledNethod                                   | valueWithReceiver:arguments:                          | Kernel   |         |  |           |
| SpCodeDebugItCommand                             | [ aCompiledMethod valueWithReceiver: anObject         |  |         | V1 tutu  |           |
|  |   |  |         | ♥(1@2) 56  |           |
| FullBlockClosure (BlockClosure)                  | [self value. "IMPORTANT: Do not step over next line o | f code. See m Kernel                                     |         | ∀(1@2) toto  |           |
|  |   |  |         | V an OrderedCollection(1 2 3 42) tata                      |           |
|  |   |  |         | (III) a ChestContentDictionary() MyPreciousChest           |           |
| ▶ <u>x ≛ x</u> ⊕ C 5                             | 9 + 6   |  |         | a cresconcentration wyried obscrest                        |           |
| Proceed Indo Over Through Run to Risslant Return | Where's? Create Advanced Step                         |  |         |  |           |
| 1 obj methodWithTwoAssignments.                  |   |  |         |  |           |
| 3 titi   |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         | ▶ 📑 🚺 😭 + II<br>Dolt Dable Binding Version Page            |           |
|  |   |  |         | 1  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  | +L        |
| an UndefinedObject (nil)                         |   | 🕼 🛶 👩 an UndefinedObject (nil)                           |         |  | 🕈 🗘 🖙 🖽 × |
| Type Variable Value                              |   | Raw Breakpoints Meta                                     |         |  |           |
| implicit @ self nil                              |   | : Variable   | : Value |  |           |
| implicit ¶ stackTop emptySta                     | ck  | (a) self   | nil     |  |           |
| implicit () thisContext Undefine                 | d0bject>>Dolt   | Sec. 264   |         |  |           |
| Chest E myVar 1                                  | ,   |  |         |  |           |
| Chest @ titi (1@2)                               |   |  |         |  |           |
| Chest @ 00 (1(\$2)                               |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   | 1 self   |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |
|  |   |  |         |  |           |

Figure 1-10 Chest variable, in the debugger inspector

| × - D   | Sinc   | larinDebuggerTest>>methodWithTwoAssignmen | ts      |  | •         |
|---|--|---|---------|--|-----------|
| Stack   |  |   | 12 🛠    | Chest  |           |
| Class   | Method   | Package                                   |         | + The Add check Roman all checks                             |           |
| □ SindarinDebuggerTest  | methodWithTwoAssignments   | Sindarin-Tests                            |         | Chests Name  |           |
| II UndefinedObject  | Dolt   |   |         | v (III) a ChestContentDictionary/'tata'->an OrderedC Default |           |
| CompiledNethod  | valueWithReceiver:arguments:   | Kernel                                    |         |  |           |
| SpCodeDebugitCommand  | [ aCompiledMethod valueWithReceiver: anObject argum  |   |         | V1 tutu  |           |
| FullBlockClosure (BlockClosure)   | [ accomplete week of a set of the |   |         | ₩(1@2) titi  |           |
| FullBlockClosure (BlockClosure)   | [self value. IMPORTANT: Do not step over next line of code   | . See mikernel                            |         | ∀(1@2) toto  |           |
|   |  |   |         | ☆ an OrderedCollection(1 2 3 42) tata                        |           |
|   |  |   |         | (III) a ChestContentDictionary() MvPreciousChest             |           |
| Image: Note of the state of the st | A, + C<br>Manatha  |   |         | P III III III<br>PH Anna Mere Yanan Agu<br>1 Ioto (≤ 42      |           |
|   |  |   |         | Line 1:1   | •         |
| an UndefinedObject (nil)  |  | 🚺 🏎 🛗 an UndefinedObject (nil)            |         |  | 🕈 🖸 🏎 🖽 × |
| Type Variable Value   |  | Raw Breakpoints M                         | eta     |  |           |
| implicit @ self nil   |  | : Variable                                | : Value |  |           |
| implicit ¶ stackTop emptyStack  |  | fise (0)                                  | nil     |  |           |
| implicit () thisContext UndefinedOb   | ject>>Dolt   |   |         |  |           |
| Chest E myVar 1   |  |   |         |  |           |
| Chest @ titi (1@2)  |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  | 1 self                                    |         |  |           |
|   |  | 1 Sect                                    |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |
|   |  |   |         |  |           |

Figure 1-11 Variable from Chest playground, not visible in the debugger inspector

# Inject code to access an object inside a chest, in a playground or in the debugger

To make it easier to access the content of a chest in a playground or in the debugger, it is possible to use the **Paste object from chest >** sub-menu. This sub-menu allows to choose a chest and the key of an object inside the chest.

| × - 🗆                 |  |              |                |           | С                   | hest            | ts             |              |        |                  |             |      |   |     | • |
|-----------------------|--|--------------|----------------|-----------|---------------------|-----------------|----------------|--------------|--------|------------------|-------------|------|---|-----|---|
| Chests + -            | thest Remove all c                           |              | Rem            |           | Remove all iter     | <sub>ns</sub> a | SmallIn        | teger (      | (1)    |                  |             | 7    | ø | 64- | Ē |
| Default               |  |              | Object         |           |                     |                 | Integer        | Rav          | w      | Breakpoints      | Meta        |      |   |     |   |
| MyPreciousChest       |  | tutu<br>titi | 1              |           |                     |                 | key            | - Car        |        | breampoints      | value       |      |   |     |   |
|                       |  | toto         | (1@2)<br>(1@2) |           |                     |                 | key<br>decimal |              |        |                  | value<br>1  |      |   |     |   |
|                       |  | tata         |                | orodCo    | llection(12         |                 |                |              |        |                  | 1           |      |   |     |   |
|                       |  | tata         | anoru          | ereuco    | liection(12         |                 | octal          |              |        |                  | 1           |      |   |     |   |
|                       |  |              |                |           |                     |                 | binary         |              |        |                  | 1           |      |   |     |   |
|                       |  |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       |  |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       |  |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       |  |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       |  |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       |  |              |                |           |                     | d.              |                |              |        |                  |             |      |   |     |   |
| Do it Publish Binding |  | *≣           |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
| Don Publish Binding   | p versions Pi                                | ages         |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
| 1 myVar :=            | Do it and                                    |              |                | 0.0       | -                   |                 |                |              |        |                  |             |      |   |     |   |
|                       | <ul> <li>Do it and</li> <li>Do it</li> </ul> | go           |                | *         |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       | 📡 Do It                                      |              |                | អា<br>៤អា |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       | ✓ Inspect in                                 |              |                | ுகா<br>அ  |                     | 18              |                |              |        |                  |             |      |   |     |   |
|                       | Print it                                     |              |                | æ         |                     | . 1             | 1 se           | lf           |        |                  |             |      |   |     |   |
|                       | 🛞 Profile it                                 |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       | (=) Store ob                                 | ject in a    | chest          | жс,ж      | s                   |                 |                |              |        |                  |             |      |   |     |   |
|                       | Code sea                                     |              |                |           | •                   |                 |                |              |        |                  |             |      |   |     |   |
|                       | Playgrou                                     |              |                |           | •                   |                 |                | -            |        |                  |             |      |   |     |   |
|                       | Load obj                                     |              |                |           |                     |                 | _              | _            | iste o | bject from chest | > Default > | tutu |   |     |   |
|                       | Paste ob                                     | ject fron    | n chest :      |           | Default<br>MyPrecid | ur Cl           |                | tutu<br>titi |        |                  |             |      |   |     |   |
|                       | C Find                                       |              |                | Ħ         |                     | user            |                | toto         |        |                  |             |      |   |     |   |
|                       | Сору   |              |                | H         |                     |                 |                | tata         |        |                  |             |      |   |     |   |
|                       | 🔏 Cut  |              |                | H         |                     |                 |                |              |        |                  |             |      |   |     |   |
|                       | Paste  |              |                | æ         | V                   |                 |                |              |        |                  |             |      |   |     |   |
|                       | Upraste                                      |              |                |           |                     |                 |                |              |        |                  |             |      |   |     |   |

Figure 1-12 Paste object from Chest context menu

When clicking on a variable name, the necessary code to access the corresponding object is pasted where you had put your cursor in the playground/debugger:

This command allows you to access objects from a chest, without relying on your memory of the Chest API.

#### Simpler code injection in any spec code presenter

The method described above to access objects in chests implies that your objects need to be stored in a chest with a name that you must remember...

Sometimes, you would like to access objects quickly, without needing to give a name to an object that is stored...

That's why it is possible to copy / paste (=inject code to access) to / from a "Clipboard Chest", just as you would do to copy/paste text to/from the clipboard.

| × - 0                                | Ches   | sts              |               |                           |   | •    |
|--------------------------------------|--|------------------|---------------|---------------------------|---|------|
| Add chest Remove chest Remove all cl |  | a SmallInteger ( | (1)           | ۲                         | Ø | s~ 🗍 |
| Default                              | Name Object  | Integer Ra       | w Breakpoints | Meta                      |   |      |
| MyPreciousChest                      | tutu     1       titi     (1@2)       toto     (1@2)       tata     an OrderedCollection(1 2 3 4 | key<br>decimal   |               | value<br>1<br>1<br>1<br>1 |   |      |
| Do it Publish Bindings Versions Pa   | r≣<br>ages<br>: 'Default') at: 'tutu'  | 1 self           |               |                           |   |      |
| Line: 1:45                           | ۴L   | Jett             |               |                           |   |      |

Figure 1-13 Injected code for Chest, after having pasted the object

To do that, you just need to select the expression that you want to evaluate and whose results should be stored in the "Clipboard Chest", and then select **Copy object in a clipboard chest** in the context menu:

This will store the result of the expression into the default clipboard chest. Please note, that if you "copy" another object to the clipboard chest, it will replace the previous copied object. Note also that the clipboard chest is a weak chest, so your object inside the clipboard chest can become nil if it gets garbage-collected.

In order to inject the code to access the object, you should then select in the context menu: **Paste last object copied to clipboard**:

And the code to access your object is now pasted:

| - 0          | P                                      | layground               |
|--------------|--|-------------------------|
| •            |  |                         |
| Do it Publis | h Bindings Versions Pages              |                         |
| 1 42         |  |                         |
| 2<br>2 (Chas |  | . In Trade and Contract |
| 3 (Ches<br>4 | t named: 'ClipboardDefault') at        | : CCTpboardEntry        |
| 5 70         | N. De Namel en                         |                         |
| 6            | Do it and go #0                        | -                       |
| 7            | ▶ Do it #[                             |                         |
| 8            | ∰ Debug it 合策[                         |                         |
| 9            | ≤s~Inspect it ∰                        |                         |
|              | ▶ Print it #                           | P                       |
|              | No Profile it                          |                         |
|              | (  ) Store object in a chest #C, #3    | 5                       |
|              | 🕥 Copy object in a clipboard chest 👌 🛞 |                         |
|              | Code search                            | •                       |
|              | Playground                             | •                       |
|              | Load object from chest                 |                         |
|              | Paste last object copied to clipboard  |                         |
|              | Paste object from chest >              | •                       |
|              | C Find #                               | F                       |
|              | 🗂 Сору жа                              |                         |

Figure 1-14 Copy object to Chest Clipboard context menu

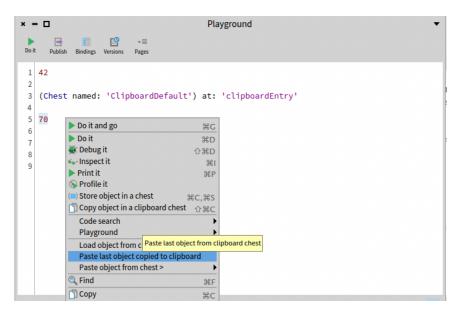
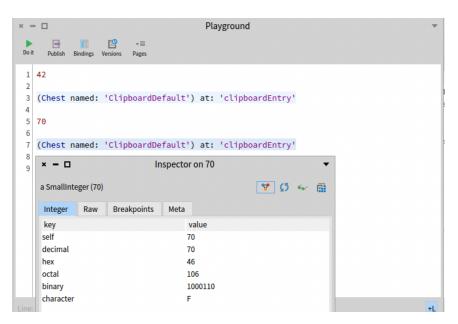


Figure 1-15 Paste last object from Chest Clipboard context menu



**Figure 1-16** Injected code for Chest, after having pasted the object from Chest Clipboard



# **API** Description

In this chapter, we detail the API of Chest and give examples of its usage.

# 2.1 Chest class-side API

In this section, we describe the API that can be used on the class Chest.

#### How to create instances

• Chest class>>#new

creates a chest with a default name that is in the form of Chest\_autoIncrementedNumber.

In the examples below, if no other chest has been created before, the names of the created chests are respectively "Chest1" and "Chest2":

```
Chest new. "Chest_1"
Chest new. "Chest_2"
```

• Chest class>>#newNamed:

creates a chest with the name given in parameter if no other chest is already named so, else an exception ChestKeyAlreadyInUseError is raised.

For example, if no other chest is already called "toto", the piece of code below creates a chest that is named as "toto":

```
Chest newNamed: 'toto'. "its name is 'toto'"
```

On the contrary, if another chest is already called "toto", the same piece of code would raise a ChestKeyAlreadyInUseError because two chests cannot have the same name:

```
Chest newNamed: 'toto'. "ChestKeyAlreadyInUseError as a chest named
'toto' already exists"
```

#### Accessors

```
    Chest class>>#allChests
```

returns an ordered collection that contains all chest instances.

If we suppose that there are no other chests than the ones created above, the piece of code returns a collection with two chests: the default chest and the chest named "toto":

```
Chest allChests "{DefaultChest. totoChest}"
```

```
    Chest class>>#chestDictionary
```

returns a dictionary containing all chests with their name as key.

If we suppose that there are no other chests than the ones created above, the piece of code returns a dictionary with two chests: the default chest, with 'Default' as a key, and the chest named "toto", with 'toto' as a key:

```
Chest chestDictionary "{'Default' -> DefaultChest. 'toto' ->
    totoChest}"
```

• Chest class>>#named:

returns the chest that is named as the string in argument if it exists, else raises an exception KeyNotFound.

The expression below can be used to get the chest named 'toto', if it exists:

```
Chest named: 'toto' "chest named 'toto'"
```

However, no chest is named 'titi', the expression below would raise a KeyNotFound:

```
Chest named: 'titi' "KeyNotFound"
```

```
• Chest class>>#defaultInstance(or Chest class>>#default).
```

You can use most of Chest's instance-side API on the class Chest itself. These methods returns the default chest that is used when you use Chest's instance-side API directly on Chest class. If the default chest doesn't already exist, calling these methods lazily create it.

For example, the expression below returns true:

Chest defaultInstance == Chest default "true"

• Chest class>>#announcer

helper method that returns the unique instance of ChestAnnouncer, which propagates changes related to chests to any subscriber.

The example belows subscribes self to the ChestAnnouncer singleton to 3 different events; when a new chest chest has been created, when the content of a chest has changed and when a chest has been removed:

```
Chest announcer weak when: ChestCreated send: #eventNewChest: to:
self;
when: ChestUpdated send:
#eventContentOfChestUpdated: to: self.
when: ChestRemoved send: #eventChestRemoved:
```

to: self.

When the event happens, the methods that are called (in the example: #eventNewChest:, #eventContentOfChestUpdated: and #eventChestRemoved:) take the related event as an argument.

```
• Chest class>>#weak
```

returns the class WeakChest, subclass of Chest. You can use the same API on this class as you would on Chest class, in order to create or access chests that hold weak references to objects. This means, that storing your objects in a weak chest doesn't prevent them from being garbage-collected.

For example, the expression below creates a new weak chest named as "wtiti", if it doesn't already exist:

```
Chest weak newNamed: 'wtiti' "weak chest named as 'wtiti'"
```

#### How to perform actions

• Chest class>>#inChest:at:put:

puts the object in third argument with the name given in second argument in the chest named as first argument. This chest is lazily created if it doesn't exist yet.

In the example below, the object 42 is stored as "toto" in the chest named "toto":

```
(Chest named: 'toto') at: 'toto' put: 42. "stores 42 as 'toto' in chest named as 'toto'.
```

If the chest "toto" didn't exist, it would be created automatically. \*Also, please note that a chest cannot contain the same object twice. So, if the chest named as "toto" already contained the object 42, it would simply be renamed as "toto" in this chest.\*

• Chest class>>#unsubscribe:

helper method that unsubscribes its argument from the unique instance of ChestAnnouncer.

For example, the expression below unsubscribes self from the ChestAnnouncer singleton:

Chest unsubscribe: self

## 2.2 Chest instance-side API

In this section, we describe the API that can be used on an instance of Chest. Methods marked by the symbol (\*) can also be used on the class Chest itself. In this case, it is equivalent to using the API on the default instance of Chest.

#### Accessors

• Chest>>#name:

returns the receiver chest's name.

For example, the expression below evaluates to true:

```
(Chest named: 'toto') name = 'toto' "true"
```

```
    Chest>>#contents: (*)
```

returns a copy of the receiver chest's contents, as a dictionary that contains all objects in the chest with their name as key.

For example, evaluating the piece of code below returns a dictionary that associates 42 to "toto" and 144 to "titi":

```
| c |
c := Chest new.
Chest inChest: c name at: 'toto' put: 42.
Chest inChest: c name at: 'titi' put: 144.
c contents "{'toto' -> 42. 'titi' -> 144}"
```

```
• Chest>>#at: :(*)
```

returns the object, contained in the receiver chest, whose name is the string in argument if it exists, else an exception KeyNotFound is raised.

For example, if the chest named as "toto" stores 42 as "titi", then the expression below returns 42:

```
[(Chest named: 'toto') at: 'titi'. "42"
```

On the contrary, if no object is named as "titi" in the chest named as "toto", then the same expression raises a KeyNotFound:

(Chest named: 'toto') at: 'titi' "KeyNotFound"

#### How to perform actions

```
• Chest>>#add: (*)
```

adds the object in argument to the receiver chest, with a default name that is in the form of <code>chestName\_autoIncrementedNumber</code>.

In the piece of code below, we add 42 then 144 to a new chest named "MyChest". Thus, 42 is stored with the key "MyChest1" and 144 is stored with the key "MyChest2":

```
| c |
c := Chest newNamed: 'MyChest'.
c add: 42. "MyChest_1 -> 42"
c add: 144 "MyChest_2 -> 144"
```

```
    Chest>>#at:put: (*)
```

adds the object in second argument to the receiver chest with the name in first argument if no other object is already named so, else an exception ChestKeyAlreadyInUseError is raised.

In the example below, in a new chest, 42 is stored as "toto". Then, trying to store 144 as "toto" raises a ChestKeyAlreadyInUseError because two objects cannot have the same name:

```
| c |
c := Chest new.
c at: 'toto' put: 42. "toto -> 42"
c at: 'toto' put: 144 "ChestKeyAlreadyInUseError"
```

\*Also, please note that a chest cannot contain the same object twice. So, if the chest named as "toto" already contained the object 42, it would simply be renamed as "toto" in this chest.\*

```
    Chest>>#at:put:ifPresent: (*)
```

adds the object in second argument to the receiver chest with the name in first argument if no other object is already named so, else the block in third argument is evaluated with zero argument.

In the exemple below, in a new chest, 42 is stored as "toto". Then, we try to store 144 as "toto" and if the key is already use, we try to store it as "titi" instead. As the key "toto" is already used by 42, 144 is stored as "titi":

• Chest>>#remove: (\*)

removes the object in argument from the receiver chest if it is there, else an exception  ${\tt KeyNot-Found}$  is raised.

For example, if the chest named "toto" contains the object 42, then the following code snippet successfully removes 42 from this chest. Then it tries to remove 42 from this chest a second time, in which case a KeyNotFound is raised because the object is not there anymore:

```
| c |
c := Chest named: 'toto'.
c contents includes: 42 "true"
c remove: 42. "42 is removed from c"
c contents includes: 42. "false"
c remove: 42 "KeyNotFound"
```

Chest>>#removeObjectNamed: (\*)

removes from the receiver chest the object named as the argument if the key exists, else an exception ObjectNotInChestError is raised.

For example, if the chest named "toto" stores an object named as "tata", then the following code snippet removes the object named as "tata" from this chest. Then it tries to remove a second time the object named as "tata" from this chest, in which case an ObjectNotInChestError is raised because the key isn't used anymore.

**API** Description

```
c := Chest new.
c at: 'toto' put: 42.
c at: 'titi' put 144.
c contents isEmpty. "false"
c empty.
```

```
c contentns isEmpty "true"
```

```
    Chest>>#remove
```

completely deletes the receiver chest from the list of existing chests. It cannot be accessed afterwards:

```
| c cname |
c := Chest new.
cname := c name.
Chest chestDictionary includesKey: cname. "true"
c remove.
Chest chestDictionary includesKey: cname. "false"
```

• Chest>>#name:

renames the receiver chest as the string in argument if no other chest is already named so, else an exception ChestKeyAlreadyInUseError is raised.

For example, if there exist only two chests named "toto" and "titi", then it is possible to rename "toto" into "tata" but not into "titi", in which case a ChestKeyAlreadyInUseError is raised:

```
| toto titi |
toto := Chest named: 'toto'.
titi := Chest named: 'titi'.
toto name: 'tata'.
toto name "tata".
```

```
2.2 Chest instance-side API
```

```
toto name: 'titi'. "ChestKeyAlreadyInUseError"
toto name "tata"
```

• Chest>>#renameObject:into::

inside the receiver chest, renames the object in first argument into the string in second argument if the object is in the chest, else an exception ObjectNotInChestError is raised, and if no other object is already named so else an exception ChestKeyAlreadyInUseError is raised.

If the chest named as "toto" contains only 42 as "titi" and 144 as "tata", then the piece of code below renames 42 to "tutu":

```
(Chest named: 'toto') at: 'titi'. "42"
(Chest named: 'toto') renameObject: 42 into: 'tutu'.
(Chest named: 'toto') at: 'tutu'. "42"
(Chest named: 'toto') at: 'titi' "KeyNotFound"
```

Under the same circumstances, the following code snippet raises an ObjectNotInChestError because it tries to rename 666 that is not in the chest named as "toto":

```
(Chest named: 'toto) contents includes: 666 "false"
(Chest named: 'toto') renameObject: 666 into 'anyValidName'
    "ObjectNotInChestError"
```

Finally, trying to rename 144 into "tutu" will raise a ChestKeyAlreadyInUseError because 42 is already named so:

```
(Chest named: 'toto') renameObject: 144 into: 'tutu'
    "ChestKeyAlreadyInUseError"
```

```
• Chest>>#inspectAt::
```

inspect the object named as the argument in the receiver chest:

For example, the following expression opens an inspector on the object stored as "titi" in the chest named "toto":

(Chest named: 'toto') inspectAt: 'titi'