Object-Centric Instrumentation with Pharo

Steven Costiou

February 5, 2019

Layout and typography based on the `sbabook` LaTeX class by Damien Pollet.

Contents

Illustrations ii

1 Introduction 1

2 What we are talking about 3
2.1 Illustration example 3
2.2 Evaluation criteria 4
2.3 Performance overhead evaluation 4

3 Anonymous subclasses 5
3.1 What are Talents 5
3.2 Example 5
3.3 Evaluation 6

4 Talents 7
4.1 What are Talents 7
4.2 Example 7
4.3 Evaluation 7

5 Proxies 9
5.1 What are Talents 9
5.2 Example 9
5.3 Evaluation 10

6 Reflectivity 11
6.1 What are Talents 11
6.2 Example 11
6.3 Evaluation 11

7 Low-level techniques 13
7.1 Example 13
7.2 Evaluation 14

8 Conclusion 15
8.1 Example 15
8.2 Evaluation 16

Bibliography 17

Illustrations

2-1 Trivial example for object-centric instrumentation 3
3-1 Installation from Github 5
3-2 Installation from Github 5
4-1 Installation from Github 7
4-2 Installation from Github 8
5-1 Installation from Github 9
5-2 Installation from Github 9
6-1 Installation from Github 11
6-2 Installation from Github 12
7-1 Installation from Github 13
7-2 Installation from Github 13
8-1 Installation from Github 15
8-2 Installation from Github 15

CHAPTER 1

Introduction

This booklet is about object-centric instrumentation in Pharo. An instrumentation is object-centric if it applies to one specific object (or a set of objects), without consideration of its class. It means the instrumentation can be applied on one object, leaving untouched all other instances of its class, or to an heterogeneous set of instances of different classes. This booklet gives an overview of available object-centric instrumentation techniques in Pharo, either present in the standard distribution or available on download. We will not go into deep technical usage description, nor into implementation details. Each chapter illustrates one solution with examples, and gives the necessary references if one wants to go deeper in the study of the solution. Each solution is evaluated against a set of criteria presented in Chapter 2, along with a short performance overhead evaluation.

CHAPTER 2

What we are talking about

In the next chapters, we evaluate each technique following a three-fold evaluation. First, the studied technique is applied on a simple example of object-centric instrumentation. Second, the technique is evaluated against a set of desirable properties. Finally, performance overhead are evaluated. Only the raw solution is evaluated, without considering the possibility of enhancing the technique by building something on top.

2.1 Illustration example

Each solution is experimented on a trivial example of object-centric behavior instrumentation. This example is illustrated in script 2-1. Two instances of `OrderedCollection` are created, and to each of these instances is sent the `add:` message with a string as a parameter. The instrumentation must happen in-between. We want to instrument the `col1` object, so that when the `add:` message is received, the size of the collection and the added object (passed as parameter) are printed in the `Transcript`.

Listing 2-1 Trivial example for object-centric instrumentation

```
|col1 col2|
col1 := OrderedCollection new.
col2 := OrderedCollection new.

"...instrumentation must happen here..."

col1 add: 'Hello World'.
col2 add: 'Hello World'.
```

2.2 Evaluation criteria

Each solution is evaluated agains the following desirable properties.

Property	**Definition**
Manipulated entity	The unit of instrumentation (*e.g.* a class, a Trait, an object...)
Reusability	The entity can be reused to instrument different objects
Flexibility	Instrumentation does not put constraint on the source code or in the coding style
Granularity	The level of at which behavior can be instrumented (*e.g.* method, AST...)
Integration	Instrumentation does not break system features

2.3 Performance overhead evaluation

- source code with instrumentation example
- describe the used method
- describe the limitations of the method

CHAPTER 3

Anonymous subclasses

3.1 What are Talents

3.2 Example

Installing Talents

aa

Example

bb

Listing 3-1 Installation from Github

```
Metacello new
  baseline: 'Talents';
  repository: 'github://tesonep/pharo-talents/src';
  load.
```

Listing 3-2 Installation from Github

```
talent := Trait named: 'MyTalent'.
talent compile: 'add: anObject
anObject logCr.
super add: anObject'.
col := OrderedCollection new.
col addTalent: talent.
col add: 'This is an added object.'
```

3.3 Evaluation

cc

Note this is a note annotation.

To do this is a todo annotation

Country	**Capital**
France	Paris
Belgium	Brussels
Country	**Capital**
France	Paris
Belgium	Brussels

CHAPTER 4

Talents

Talents is this.

4.1 What are Talents

4.2 Example

Installing Talents

aa

Example

bb

4.3 Evaluation

cc

Listing 4-1 Installation from Github

```
Metacello new
  baseline: 'Talents';
  repository: 'github://tesonep/pharo-talents/src';
  load.
```

Listing 4-2 Installation from Github

```
talent := Trait named: 'MyTalent'.
talent compile: 'add: anObject
anObject logCr.
super add: anObject'.
col := OrderedCollection new.
col addTalent: talent.
col add: 'This is an added object.'
```

Note this is a note annotation.

To do this is a todo annotation

Country	**Capital**
France	Paris
Belgium	Brussels

Country	**Capital**
France	Paris
Belgium	Brussels

CHAPTER 5

Proxies

5.1 What are Talents

5.2 Example

Installing Talents

aa

Example

bb

Listing 5-1 Installation from Github

```
Metacello new
  baseline: 'Talents';
  repository: 'github://tesonep/pharo-talents/src';
  load.
```

Listing 5-2 Installation from Github

```
talent := Trait named: 'MyTalent'.
talent compile: 'add: anObject
anObject logCr.
super add: anObject'.
col := OrderedCollection new.
col addTalent: talent.
col add: 'This is an added object.'
```

5.3 Evaluation

cc

Note this is a note annotation.

To do this is a todo annotation

Country	**Capital**
France	Paris
Belgium	Brussels
Country	**Capital**
France	Paris
Belgium	Brussels

CHAPTER 6

Reflectivity

Talents is this.

6.1 What are Talents

6.2 Example

Installing Talents

aa

Example

bb

6.3 Evaluation

cc

Listing 6-1 Installation from Github

```
Metacello new
  baseline: 'Talents';
  repository: 'github://tesonep/pharo-talents/src';
  load.
```

Listing 6-2 Installation from Github

```
talent := Trait named: 'MyTalent'.
talent compile: 'add: anObject
anObject logCr.
super add: anObject'.
col := OrderedCollection new.
col addTalent: talent.
col add: 'This is an added object.'
```

Note this is a note annotation.

To do this is a todo annotation

Country	Capital
France	Paris
Belgium	Brussels
Country	**Capital**
France	Paris
Belgium	Brussels

CHAPTER 7

Low-level techniques

7.1 Example

Installing Talents

aa

Example

bb

Listing 7-1 Installation from Github

```
Metacello new
  baseline: 'Talents';
  repository: 'github://tesonep/pharo-talents/src';
  load.
```

Listing 7-2 Installation from Github

```
talent := Trait named: 'MyTalent'.
talent compile: 'add: anObject
anObject logCr.
super add: anObject'.
col := OrderedCollection new.
col addTalent: talent.
col add: 'This is an added object.'
```

7.2 Evaluation

cc

Note this is a note annotation.

To do this is a todo annotation

Country	**Capital**
France	Paris
Belgium	Brussels
Country	**Capital**
France	Paris
Belgium	Brussels

CHAPTER 8

Conclusion

8.1 Example

Installing Talents

aa

Example

bb

Listing 8-1 Installation from Github

```
Metacello new
  baseline: 'Talents';
  repository: 'github://tesonep/pharo-talents/src';
  load.
```

Listing 8-2 Installation from Github

```
talent := Trait named: 'MyTalent'.
talent compile: 'add: anObject
anObject logCr.
super add: anObject'.
col := OrderedCollection new.
col addTalent: talent.
col add: 'This is an added object.'
```

8.2 Evaluation

cc

Note this is a note annotation.

To do this is a todo annotation

Country	**Capital**
France	Paris
Belgium	Brussels
Country	**Capital**
France	Paris
Belgium	Brussels

Bibliography