

# Learning Object-Oriented Programming, Design and TDD with Pharos

Stéphane Ducasse

March 11, 2019

Copyright 2017 by Stéphane Ducasse.

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>

# Contents

<b>Illustrations</b>	<b>ii</b>
<b>1 Some collection katas solutions</b>	<b>1</b>
1.1 Isogram . . . . .	1
1.2 Pangrams . . . . .	1
1.3 Identifying missing letters . . . . .	2
1.4 Palindrome . . . . .	3
<b>Bibliography</b>	<b>5</b>

# Illustrations



# Some collection katas solutions

This chapter contains the solution of the exercises presented in Chapter ??

## 1.1 Isogram

```
String >> isIsogramSet
  "Returns true if the receiver is an isogram, i.e., a word using no
    repetitive character."
  "'pharo' isIsogramSet
  >>> true"
  "'phaoro' isIsogramSet
  >>> false"

  ^ self size = self asSet size
```

## 1.2 Pangrams

```
String >> isPangramIn: alphabet
  "Returns true is the receiver is a pangram i.e., that it uses all
    the characters of a given alphabet."
  "'The quick brown fox jumps over the lazy dog' isPangramIn:
    'abcdefghijklmnopqrstuvwxyz'
  >>> true"
  "'tata' isPangramIn: 'at'
  >>> true"

  alphabet do: [ :aChar |
    (self includes: aChar)
      ifFalse: [ ^ false ]
  ]
```

```

    ].
    ^ true

String >> isEnglishPangram
    "Returns true is the receiver is a pangram i.e., that it uses all
      the characters of a given alphabet."
    "'The quick brown fox jumps over the lazy dog' isEnglishPangram
    >>> true"
    "'The quick brown fox jumps over the dog' isEnglishPangram
    >>> false"

    ^ self isPangramIn: 'abcdefghijklmnopqrstuvwxyz'

```

### 1.3 Identifying missing letters

```

String >> detectFirstMissingLetterFor: alphabet
    "Return the first missing letter to make a pangram of the receiver
      in the context of a given alphabet.
    Return '' otherwise"

    alphabet do: [ :aChar |
        (self includes: aChar)
        ifFalse: [ ^ aChar ]
    ].
    ^ ''

```

#### Detecting all the missing letters

We create a Set instead of an Array because Arrays in Pharo have a fixed size that should be known at creation time. We could have created an array of size 26. In addition we do not care about the order of the missing letters. A Set is a collection whose size can change, and in which we can add the element one by one, therefore it fits our requirements.

```

String >> detectAllMissingLettersFor: alphabet

    | missing |
    missing := Set new.
    alphabet do: [ :aChar |
        (self includes: aChar)
        ifFalse: [ missing add: aChar ] ].
    ^ missing

```

## 1.4 Palindrome

```
String >> isPalindrome2
"Returns true whether the receiver is an palindrome.
'anna' isPalindrome2
>>> true
'andna' isPalindrome2
>>> true
'avdna' isPalindrome2
>>> false
"
1
  to: self size//2
  do: [ :i | (self at: i) = (self at: self size + 1 - i)
          ifFalse: [ ^false ]
        ].
^true
```





# Bibliography