Learning Object-Oriented Programming, Design and TDD with Pharo

Stéphane Ducasse

March 11, 2019

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

Contents

Illustrations

CHAPTER 1

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