

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

Illustrations	ii
1 Expressions solutions	1
1.1 Evaluate message	1
1.2 Negated message	1
1.3 Better class instance creation interface	2
1.4 Printing addition and multiplication	2
1.5 Negated negation	2
1.6 evaluateWith:	2
Bibliography	3

Illustrations



Expressions solutions

Here are the possible solutions of the implementation we asked for the Expression Chapter ??.

1.1 Evaluate message

```
[ EConstant >> evaluate
  ^ value

[ ENegation >> evaluate
  ^ expression evaluate negated

[ EAddition >> evaluate
  ^ left evaluate + right evaluate

[ EMultiplication >> evaluate
  ^ left evaluate * right evaluate
```

1.2 Negated message

```
[ EAddition >>> negated
  ^ ENegation new expression: self

[ EMultiplication >>> negated
  ^ ENegation new expression: self
```

1.3 Better class instance creation interface

```
[ ENegation class >> expression: anExpression
  ^ self new expression: anExpression
[ EMultiplication class >> left: anExp right: anExp2
  ^ self new left: anExp ; right: anExp2
```

1.4 Printing addition and multiplication

```
[ EAddition >> printOn: aStream
  aStream nextPutAll: '( '.
  left printOn: aStream.
  aStream nextPutAll: ' + '.
  right printOn: aStream.
  aStream nextPutAll: ' )'
[ EMultiplication >> printOn: aStream
  aStream nextPutAll: '( '.
  left printOn: aStream.
  aStream nextPutAll: ' * '.
  right printOn: aStream.
  aStream nextPutAll: ' )'
```

1.5 Negated negation

```
[ ENegation >> negated
  ^ expression
```

1.6 evaluateWith:

```
[ EMultiplication >> evaluateWith: anObject
  ^ (right evaluateWith: anObject) + (left evaluateWith: anObject)
```

Bibliography