Nolix

Nolix Validator

2020-10-23

Table of contents

1 Introduction

1.1 What the Nolix Validator is

The Nolix Validator can validate **arguments**.

1.2 Why to use the Nolix Validator

- The Nolix Validator can validate arguments on **many different** properties.
- The Nolix Validator produces **consistent** error messages.
- The calls of the Nolix Validator can be written in **very legible** code.

1.3 Where the Nolix Validator is

The Nolix Validator is defined in the Nolix library. To use the Nolix Validator, import the Nolix library into your project.

2 Basics

2.1 Import the Nolix Validator

```
import ch.nolix.common.validator.Validator;
```

The Nolix validator is in the package ‘ch.nolix.common.validator’.

2.2 Validate that an object is not null

```
public void setContent(Object content) {

  Validator.assertThat(content).isNotNull();

  …
}
```

If the given content is null, the Validator will throw an ArgumentIsNullException. The error message of the ArgumentIsNullException will be:

“The given argument is null.”

2.3 Validate that an object is of a given type

```
public void setContent(Object content) {

  Validator.assertThat(content).isOfType(String.class);

  …
}
```

If the given content is null, the Validator will throw an ArgumentIsNullException.

If the given content is not a String, the Validator will throw an InvalidArgumentException. The message of the InvalidArgumentException will be:

”The given argument is not a String.”

3 Argument name

3.1 Include the argument's name into the error message

```
public void setContent(Object content) {

  Validator.assertThat(content).thatIsNamed("content").isNotNull();

  …
}
```

If the given content is null, the Validator will throw an ArgumentIsNullException. The error message of the ArgumentIsNullException will be:

"The given content is null."

3.2 Use constants for common argument names

```
import ch.nolix.common.constants.VariableNameCatalogue;

…

public void setContent(Object content) {

  Validator
  .assertThat(content)
  .thatIsNamed(VariableNameCatalogue.CONTENT)
  .isNotNull();

  …
}

…
```

If the given content is null, the Validator will throw an ArgumentIsNullException. The error message of the ArgumentIsNullException will be:

"The given content is null."

The VariableNameCatalogue provides constants of common argument names and can be found in the package 'ch.nolix.common.constants'. These constants are just strings.

4 Numbers

4.1 Validate that a number is not negative

```
public void setAmount(int amount) {

  Validator.assertThat(amount).isNotNegative();

  …
}
```

If the given amount negative, the Validator will throw a NegativeArgumentException. If the given amount is e.g. -25, the error message of the NegativeArgumentException will be:

“The given argument ‘-25’ is negative.”

4.2 Validate that a number is in a given range

```
public void buyPencils(int amount) {

  Validator.assertThat(amount).isBetween(100, 10000);

  …
}
```

If the given amount is not in the given range, the Validator will throw an OutOfRangeArgumentException. If the given amount is e.g. 50, the error message of the OutOfRangeArgumentException will be:

“The given argument ‘50’ is not in [100, 10000].”

5 Strings

5.1 Validate that a String is not empty

```
public void setName(String name) {

  Validator.assertThat(name).isNotEmpty();

  …
}
```

If the given name is null, the Validator will throw an ArgumentIsNullException.

If the given name is empty, the Validator will throw an EmptyArgumentException. The error message of the EmptyArgumentException will be:

"The given String is empty."

6 Containers

6.1 Validate that a container is not empty

```
public void saveMeasuredValues(double[] measuredValues) {

  Validator.assertThat(measuredValues).isNotEmpty();

  …
}
```

If the given measured values array is null, the Validator will throw an ArgumentIsNullException.

If the given measured values array is empty, the Validator will throw an EmptyArgumentException. The error message of the EmptyArgumentException will be:

"The given array is empty."

7 Elements in a container

7.1 Validate that the floating point numbers in a container are not negative

```
public void saveMeasuredValues(double[] measuredValues) {

  Validator.assertThatTheDoubles(measuredValues).areNotNegative();

  …
}
```

If the given measured values array is null, the Validator will throw an ArgumentIsNullException.

If one or severals of the given measured values are negative, the Validator will throw a NegativeArgumentException. If e.g. the 5th measured value is -10, the error message of the NegativeArgumentException will be:

“The given 5th argument ‘-10’ is negative.”

7.2 Validate that the Strings in a container are not empty

```
public void addCities(String[] cityNames) {

  Validator.assertThatTheStrings(cityNames).areNotEmpty();

  …
}
```

If the given city names array is null, the Validator will throw an ArgumentIsNullException.

If one or several of the given city names are empty, the Validator will throw an EmptyArgumentException. If e.g. the 5th city name is empty, the error message of the EmptyArgumentException will be:

"The given 5th argument is empty."

8 Reflection

8.1 Validate that a method has an annotation

```
setDataMethod(Method dataMethod) {

  Validator.assertThat(dataMethod).hasAnnotation(Data.class);

  …
}
```

If the given dataMethod is null, the Validator will throw an ArgumentIsNullException.

If the given dataMethod does not have the Data annotation, the Validator will throw an InvalidArgumentException. The error message of the InvalidArgumentException will be:

“The given method does not have the annotation ‘Data’.”

8.2 Validate that a method does not return anything

```
setRunMethod(Method runMethod) {

  Validator.assertThat(runMethod).doesNotReturnAnything();

  …
}
```

If the given runMethod is null, the Validator will throw an ArgumentIsNullException.

If the given runMethod returns something, resp. is not void, the Validator will throw an InvalidArgumentException. The error message of the InvalidArgumentException will be:

"The given method returns something."