Nolix

Nolix Exceptions

2020-11-15

Table of contents

1 Introduction

1.1 What Nolix Exceptions are

Nolix Exceptions are Exceptions for invalid **arguments**.

1.2 Why to use Nolix Exceptions

- There exist **suitable** types of Nolix Exceptions for the most situations.
- Nolix Exceptions provide **consistent** error messages.
- Nolix Exceptions have different constructors for taking all available information to create error messages that are as **informative** as possible.

1.3 Where Nolix Exceptions are

The Nolix Exceptions are in the Nolix library. To use Nolix Exceptions, import the Nolix library into your project.

2 Basics

2.1 Import InvalidArgumentException

```
import ch.nolix.common.invalidArgumentException.InvalidArgumentException;
```

The InvalidArgumentException is in the ch.nolix.common.invalidArgumentException package.

2.2 Throw a meaningful Nolix Exception

```
public void setAmount(int amount) {

  if (amount < 0) {
    throw new NegativeArgumentException(amount);
  }
  …
}
```

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

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

3 Argument names

3.1 Include the argument's name in the error message

```
public void setPrice(int amount) {

  if (amount < 0) {
    throw new NegativeArgumentException("amount", amount);
  }
  …
}
```

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

"The given amount '-25' is negative."

3.2 Use constants for common argument names

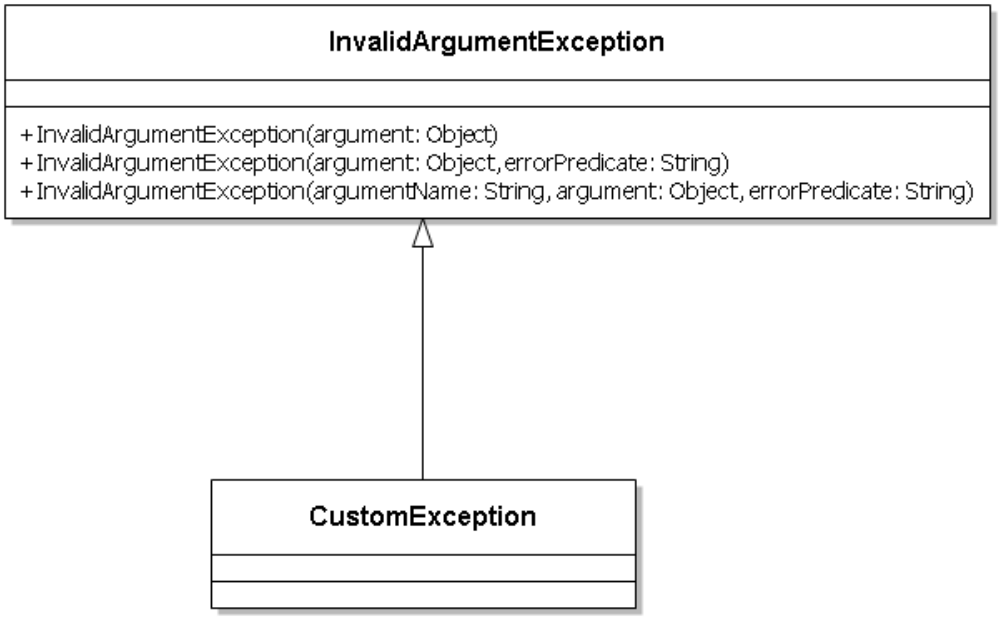
```
import ch.nolix.common.constants.VariableNameCatalogue;
…
public void setAmount(int amount) {

  if (amount < 0) {
    throw
    new NegativeArgumentException(VariableNameCatalogue.AMOUNT, amount);
  }
  …
}
```

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

"The given amount '-25' is negative."

The VariableNameCatalogue provides constants for common argument names. These constants are Strings. The VariableNameCatalogue is in the ch.nolix.common.constants package.

4 Define a custom InvalidArgumentException




```
public class CustomException extends InvalidArgumentException {
  …
}
```

A custom InvalidArgumentException can be defined by inheriting a class from it. The constructors of the InvalidArgumentException must call one of the constructors of the base class.

5 Types of InvalidArgumentExceptions

5.1 For general arguments

Exception	Suppose
ArgumentBelongsToUnexchangeableParentException	Supposed to be thrown when a given argument belongs undesirably to an **unexchangeable parent**.
ArgumentCannotRemoveAttributeException	Supposed to be thrown when on a given argument is undesirably **tried to remove an attribute**.
ArgumentDoesNotBelongToParentException	Supposed to be thrown when a given argument does undesirable **not belong to a parent**.
ArgumentDoesNotHaveAttributeException	Supposed to be thrown when a given argument does **not have a wanted attribute**.
ArgumentDoesNotSupportMethodException	Supposed to be thrown when on a given argument is **tried to call an unsupported method**.
ArgumentIsNullException	Supposed to be thrown when a given argument is undesirably **null**.
EmptyArgumentException	Supposed to be thrown when a given argument is undesirably **empty**.
EqualArgumentException	Supposed to be thrown when a given argument undesirably **equals** a given value.
FrozenArgumentException	Supposed to be thrown when a given argument is undesirably **frozen**.

Exception	Suppose
InvalidArgumentException	Supposed to be thrown when a given argument is **not valid** of any reason.
UnequalArgumentException	Supposed to be thrown when a given argument does undesirably **not equal** a given value.
UninstantiableClassException	Supposed to be thrown when a given class is undesirably **tried to be instantiated**.
UnrepresentingArgumentException	Supposed to be thrown when a given argument does **not represent** an object of a wanted type.

5.2 For boolean arguments

Exception	Suppose
FalseArgumentException	Supposed to be thrown when a given argument is undesirably **false**.
TrueArgumentException	Supposed to be thrown when a given argument is undesirably **true**.

5.3 For numeric arguments

Exception	Suppose
ArgumentIsInRangeException	Supposed to be thrown when a given argument is **in an unwanted range**.
ArgumentIsOutOfRangeException	Supposed to be thrown when a given argument is **not in a wanted range**.
ArgumentIsZeroException	Supposed to be thrown when a given number is undesirably **0**.
BiggerArgumentException	Supposed to be thrown when a given argument is undesirably **bigger than** a given maximum.
NegativeArgumentException	Supposed to be thrown when a given argument is undesirably **negative**.
NonBiggerArgumentException	Supposed to be thrown when a given argument is undesirably not **bigger** than a given limit.
NonEmptyArgumentException	Supposed to be thrown when a given argument is undesirably **not empty**.
NonNegativeArgumentException	Supposed to be thrown when a given argument is undesirably **not negative**.
NonPositiveArgumentException	Supposed to be thrown when a given argument is undesirably **not positive**.
NonSmallerArgumentException	Supposed to be thrown when a given argument is **not smaller** than a given limit.
PositiveArgumentException	Supposed to be thrown when a given argument is undesirably **positive**.
SmallerArgumentException	Supposed to be thrown when a given argument is undesirably **smaller** than a given minimum.