Google Guava class Preconditions

This class can be used to check the state or arguments passed. As the name says Preconditions, this class is used to check for preconditions and see whether method can process the data or not.

We will walk through the methods of Preconditions class.

Each method has total 3 overloaded versions

1. No extra arguments: Exceptions are thrown without any messages
2. Object Argument: Exception are thrown with String.valueOf(message) which eventually ends up calling obj.toString()
3. Additional Object arguments: they are error message arguments that are substituted into message template.

**Method checkArgument**

1. checkArgument(boolean expression)

**public** **static** **void** doubleEven(**final** **int** a) {

**boolean** isEven = (a % 2 == 0);

Preconditions.*checkArgument*(isEven);

**int** multiply = a \* 2;

System.***out***.println(multiply);

}

Output:

Exception in thread "main" **java.lang.IllegalArgumentException**

at com.google.common.base.Preconditions.checkArgument(Preconditions.java:76)

1. checkArgument(boolean expression, @Nullable Object errorMessage)

**public** **static** **void** doubleEven(**final** **int** a) {

**boolean** isEven = (a % 2 == 0);

Preconditions.*checkArgument*(isEven, "Input should be even");

**int** multiply = a \* 2;

System.***out***.println(multiply);

}

Output:

Exception in thread "main" **java.lang.IllegalArgumentException**: **Input should be even**

at com.google.common.base.Preconditions.checkArgument(Preconditions.java:92)

1. checkArgument(boolean expression, @Nullable String errorMessageTemplate, @Nullable Object... errorMessageArgs)

**public** **static** **void** doubleEven(**final** **int** a) {

**boolean** isEven = (a % 2 == 0);

Preconditions.*checkArgument*(isEven, "Input %s should be even", a);

**int** multiply = a \* 2;

System.***out***.println(multiply);

}

Output:

Exception in thread "main" **java.lang.IllegalArgumentException: Input 11 should be even**

at com.google.common.base.Preconditions.checkArgument(Preconditions.java:119)

The two methods with extra arguments are used to format the exception message. Now we will see which other methods are. I will not repeat the two overloaded versions of exception message formatting as they are same to use as above 2nd and 3rd method of checkArgument(..).

**Method checkNotNull**

**public** **static** **int** length(**final** String str) {

Preconditions.*checkNotNull*(str);

**return** str.length();

}

Output: Assuming str is null.

Exception in thread "main" java.lang.NullPointerException

at com.google.common.base.Preconditions.checkNotNull(Preconditions.java:191)

**Method checkPositionIndex**

**public** **static** **void** checkIndex(**final** **int** index, **final** **int** size){

Preconditions.*checkPositionIndex*(index, size);

}

Output: Assuming index is 11 and size is 10

Exception in thread "main" java.lang.IndexOutOfBoundsException: index (11) must not be greater than size (10)

at com.google.common.base.Preconditions.checkPositionIndex(Preconditions.java:355)

at com.google.common.base.Preconditions.checkPositionIndex(Preconditions.java:334)

**Method checkPositionIndexes**

**public** **static** **void** checkPositionIndexes(**final** **int** start, **final** **int** end, **final** **int** size){

Preconditions.*checkPositionIndexes*(start, end, size);

}

Output: Assuming start = 11, end = 13, size = 10

Exception in thread "main" java.lang.IndexOutOfBoundsException: start index (11) must not be greater than size (10)

at com.google.common.base.Preconditions.checkPositionIndexes(Preconditions.java:388)

**Method checkElementIndex**

**public** **static** **void** checkElementIndex(**final** **int** index, **final** **int** size){

Preconditions.*checkElementIndex*(index, size);

}

Output: Assuming index = 12, size = 11

Exception in thread "main" java.lang.IndexOutOfBoundsException: index (12) must be less than size (11)

at com.google.common.base.Preconditions.checkElementIndex(Preconditions.java:305)

**Method checkState**

**public** **static** **void** checkState(){

Preconditions.*checkState*(field.isEmpty());

}

checkState method is used to check the state of the object or iterator or anything that can pass boolean as parameter.

That’s all on 6 methods of Preconditions class.

In next post we will learn about Predicates of Google Guava.