GUAVA - PRECONDITIONS CLASS

Preconditions provide static methods to check that a method or a constructor is invoked with proper parameter or not. It checks the pre-conditions. Its methods throw IllegalArgumentException on failure.

Class Declaration

Following is the declaration for **com.google.common.base.Preconditions** class:

```
@GwtCompatible
public final class Preconditions
   extends Object
```

Class Methods

Sr.No	Method & Description
1	**static void checkArgument***booleanexpression* Ensures the truth of an expression involving one or more parameters to the calling method.
2	**static void checkArgument***booleanexpression, ObjecterrorMessage* Ensures the truth of an expression involving one or more parameters to the calling method.
3	**static void checkArgument** ***booleanexpression, StringerrorMessageTemplate, Object. errorMessageArgs*** Ensures the truth of an expression involving one or more parameters to the calling method.
4	**static int checkElementIndex***intindex, intsize* Ensures that index specifies a valid element in an array, list or a string of size.
5	**static int checkElementIndex***intindex, intsize, Stringdesc* Ensures that index specifies a valid element in an array, list, or a string of size.
6	**static <T> T checkNotNull***Treference* Ensures that an object reference passed as a parameter to the calling method is not null.
7	**static <T> T checkNotNull***Treference, ObjecterrorMessage* Ensures that an object reference passed as a parameter to the calling method is not null.
8	**static <T> T checkNotNull***Treference, StringerrorMessageTemplate, Object. . . errorMessageArgs* Ensures that an object reference passed as a parameter to the calling method is not null.

9	**static int checkPositionIndex***intindex, intsize* Ensures that index specifies a valid position in an array, list or a string of size.
10	**static int checkPositionIndex***intindex, intsize, Stringdesc* Ensures that index specifies a valid position in an array, list or a string of size.
11	**static void checkPositionIndexes***intstart, intend, intsize* Ensures that start and end specify a valid positions in an array, list or a string of size, and are in order.
12	**static void checkState***booleanexpression* Ensures the truth of an expression involving the state of the calling instance, but not involving any parameters to the calling method.
13	**static void checkState***booleanexpression, ObjecterrorMessage* Ensures the truth of an expression involving the state of the calling instance, but not involving any parameters to the calling method.
14	**static void checkState***booleanexpression, StringerrorMessageTemplate, Object. . . errorMessageArgs* Ensures the truth of an expression involving the state of the calling instance, but not involving any parameters to the calling method.

Methods Inherited

This class inherits methods from the following class:

- java.lang.Object

Example of Preconditions

Create the following java program using any editor of your choice in say **C:/> Guava.**

GuavaTester.java

```
import com.google.common.base.Preconditions;

public class GuavaTester {

   public static void main(String args[]) {
      GuavaTester guavaTester = new GuavaTester();

      try {
         System.out.println(guavaTester.sqrt(-3.0));
      }catch(IllegalArgumentException e) {
         System.out.println(e.getMessage());
      }

      try {
         System.out.println(guavaTester.sum(null,3));
      }catch(NullPointerException e) {
         System.out.println(e.getMessage());
      }

      try {
```

```
         System.out.println(guavaTester.getValue(6));
      }catch(IndexOutOfBoundsException e) {
         System.out.println(e.getMessage());
      }
   }

   public double sqrt(double input) throws IllegalArgumentException {
      Preconditions.checkArgument(input > 0.0,
         "Illegal Argument passed: Negative value %s.", input);
      return Math.sqrt(input);
   }

   public int sum(Integer a, Integer b){

      a = Preconditions.checkNotNull(a, "Illegal Argument passed: First parameter is
Null.");
      b = Preconditions.checkNotNull(b, "Illegal Argument passed: Second parameter is
Null.");

      return a+b;
   }

   public int getValue(int input){
      int[] data = {1,2,3,4,5};
      Preconditions.checkElementIndex(input,data.length, "Illegal Argument passed:
Invalid index.");

      return 0;
   }
}
```

Verify the Result

Compile the class using **javac** compiler as follows

```
C:\Guava>javac GuavaTester.java
```

Now run the GuavaTester to see the result.

```
C:\Guava>java GuavaTester
```

See the result.

```
Illegal Argument passed: Negative value -3.0.
Illegal Argument passed: First parameter is Null.
Illegal Argument passed: Invalid index. (6) must be less than size (5)
```