**Введение в AssertJ**

8 сентября 2019 г. · 7 мин. чтения

**1. Обзор**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#1--%D0%BE%D0%B1%D0%B7%D0%BE%D1%80)

В этой статье мы рассмотрим [AssertJ](https://joel-costigliola.github.io/assertj/" \t "_blank) — управляемую сообществом библиотеку с открытым исходным кодом, используемую для написания плавных и насыщенных утверждений в тестах Java.

Эта статья посвящена инструментам, доступным в базовом модуле AssertJ под названием **AssertJ-core** .

**2. Зависимости Maven[​](https://for-each.dev/lessons/b/-introduction-to-assertj" \l "2--%D0%B7%D0%B0%D0%B2%D0%B8%D1%81%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8-maven" \o "Прямая ссылка на 2--зависимости-maven)**

Чтобы использовать AssertJ, вам необходимо включить следующий раздел в файл pom.xml :

<dependency>  
 <groupId>org.assertj</groupId>  
 <artifactId>assertj-core</artifactId>  
 <version>3.4.1</version>  
 <scope>test</scope>  
</dependency>

Эта зависимость охватывает только основные утверждения Java. Если вы хотите использовать расширенные утверждения, вам нужно будет добавить дополнительные модули отдельно.

Обратите внимание, что для Java 7 и более ранних версий вы должны использовать версию ядра AssertJ 2.xx.

Последние версии можно найти [здесь](https://search.maven.org/classic/#search%7Cga%7C1%7Ca%3A%22assertj-core%22) .

**3. Введение**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#3-%D0%B2%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5)

AssertJ предоставляет набор классов и служебных методов, которые позволяют нам легко писать плавные и красивые утверждения для:

* Стандартная Java
* Ява 8
* Гуава
* Джода Тайм
* Neo4J и
* Компоненты качания

Подробный список всех модулей доступен на [сайте](https://joel-costigliola.github.io/assertj) проекта .

Начнем с нескольких примеров прямо из документации AssertJ:

assertThat(frodo)  
 .isNotEqualTo(sauron)  
 .isIn(fellowshipOfTheRing);  
  
assertThat(frodo.getName())  
 .startsWith("Fro")  
 .endsWith("do")  
 .isEqualToIgnoringCase("frodo");  
  
assertThat(fellowshipOfTheRing)  
 .hasSize(9)  
 .contains(frodo, sam)  
 .doesNotContain(sauron);

Приведенные выше примеры — это только вершина айсберга, но они дают нам общее представление о том, как может выглядеть запись утверждений с помощью этой библиотеки.

**4. AssertJ в действии**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#4--assertj-%D0%B2-%D0%B4%D0%B5%D0%B9%D1%81%D1%82%D0%B2%D0%B8%D0%B8)

В этом разделе мы сосредоточимся на настройке AssertJ и изучении его возможностей.

**4.1. Начиная**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#41--%D0%BD%D0%B0%D1%87%D0%B8%D0%BD%D0%B0%D1%8F)

С помощью jar библиотеки в пути к классам включить утверждения так же просто, как добавить один статический импорт в ваш тестовый класс:

import static org.assertj.core.api.Assertions.\*;

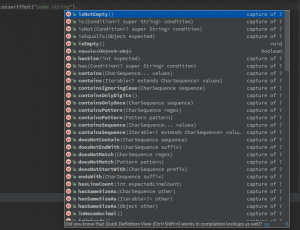
**4.2. Написание утверждений**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#42--%D0%BD%D0%B0%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D0%B5-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D0%B9)

Чтобы написать утверждение, вам всегда нужно начинать с передачи вашего объекта методу Assertions.assertThat() , а затем следовать фактическим утверждениям.

Важно помнить, что в отличие от некоторых других библиотек, приведенный ниже код на самом деле еще ничего не утверждает и **никогда не** провалит тест:

assertThat(anyRefenceOrValue);

Если вы используете функции завершения кода вашей IDE, написание утверждений AssertJ становится невероятно простым благодаря его описательным методам. Вот как это выглядит в IntelliJ IDEA 16:



Функции завершения кода IDE

Как видите, у вас есть десятки контекстных методов на выбор, и они доступны только для типа String . Давайте подробно рассмотрим некоторые из этих API и рассмотрим некоторые конкретные утверждения.

**4.3. Утверждения объекта**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#43-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F--%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%B0)

Объекты можно сравнивать различными способами либо для определения равенства двух объектов, либо для изучения полей объекта.

Давайте рассмотрим два способа, которыми мы можем сравнить равенство двух объектов. Учитывая следующие два объекта Dog fido и fidosClone :

public class Dog {   
 private String name;   
 private Float weight;  
   
 *// standard getters and setters*  
}  
  
Dog fido = new Dog("Fido", 5.25);  
  
Dog fidosClone = new Dog("Fido", 5.25);

Мы можем сравнить равенство со следующим утверждением:

assertThat(fido).isEqualTo(fidosClone);

Это не удастся, так как isEqualTo() сравнивает ссылки на объекты. Если вместо этого мы хотим сравнить их содержимое, мы можем использовать isEqualToComparingFieldByFieldRecursively() следующим образом:

assertThat(fido).isEqualToComparingFieldByFieldRecursively(fidosClone);

Fido и fidosClone равны при выполнении рекурсивного поля путем сравнения полей, потому что каждое поле одного объекта сравнивается с полем другого объекта.

Существует множество других методов утверждений, которые предоставляют различные способы сравнения и заключения контрактов между объектами, а также проверки и утверждения их полей. Чтобы узнать их все, обратитесь к официальной [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractObjectAssert.html)AbstractObjectAssert .

**4.4. Логические утверждения**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#44-%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5--%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F)

Существует несколько простых методов проверки достоверности:

* правда()
* Ложь()

Давайте посмотрим на них в действии:

assertThat("".isEmpty()).isTrue();

**4.5. Итерируемые/массивные утверждения**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#45-%D0%B8%D1%82%D0%B5%D1%80%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B5%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BD%D1%8B%D0%B5--%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F)

Для Iterable или Array существует несколько способов подтвердить, что их содержимое существует. Одним из наиболее распространенных утверждений будет проверка того, содержит ли Iterable или Array заданный элемент:

List<String> list = Arrays.asList("1", "2", "3");  
  
assertThat(list).contains("1");

или если список не пуст:

assertThat(list).isNotEmpty();

или если список начинается с заданного символа. Например «1»:

assertThat(list).startsWith("1");

Имейте в виду, что если вы хотите создать более одного утверждения для одного и того же объекта, вы можете легко соединить их вместе.

Вот пример утверждения, которое проверяет, не является ли предоставленный список пустым, содержит ли элемент «1», не содержит ли нулей и содержит ли последовательность элементов «2», «3»:

assertThat(list)  
 .isNotEmpty()  
 .contains("1")  
 .doesNotContainNull()  
 .containsSequence("2", "3");

Конечно, для этих типов существует гораздо больше возможных утверждений. Чтобы узнать их все, обратитесь к официальной [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractIterableAssert.html)AbstractIterableAssert .

**4.6. Утверждения персонажа**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#46-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F--%D0%BF%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%B6%D0%B0)

Утверждения для типов символов в основном включают сравнения и даже проверку того, взят ли данный символ из таблицы Unicode .

Вот пример утверждения, которое проверяет, является ли предоставленный символ не «a», находится ли он в таблице Unicode, больше ли «b» и является ли он строчным:

assertThat(someCharacter)  
 .isNotEqualTo('a')  
 .inUnicode()  
 .isGreaterThanOrEqualTo('b')  
 .isLowerCase();

Подробный список утверждений всех типов символов см . в [документации по](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractCharacterAssert.html)AbstractCharacterAssert .

**4.7. Утверждения класса**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#47-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F--%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0)

Утверждения для типа класса в основном касаются проверки его полей, типов класса , наличия аннотаций и окончательности класса.

Если вы хотите утверждать, что класс Runnable является интерфейсом, вам нужно просто написать:

assertThat(Runnable.class).isInterface();

или если вы хотите проверить, можно ли назначить один класс из другого:

assertThat(Exception.class).isAssignableFrom(NoSuchElementException.class);

Все возможные утверждения Class можно посмотреть в [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractClassAssert.html)AbstractClassAssert .

**4.8. Утверждения файла**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#48-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F--%D1%84%D0%B0%D0%B9%D0%BB%D0%B0)

Утверждения файла касаются проверки того, существует ли данный экземпляр файла , является ли он каталогом или файлом, имеет ли определенное содержимое, доступен для чтения или имеет заданное расширение.

Здесь вы можете увидеть пример утверждения, которое проверяет, существует ли данный файл, является ли он файлом, а не каталогом, доступен ли он для чтения и записи:

assertThat(someFile)  
 .exists()  
 .isFile()  
 .canRead()  
 .canWrite();

Все возможные утверждения Class можно посмотреть в [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractFileAssert.html)AbstractFileAssert .

**4.9. Двойные/плавающие/целочисленные утверждения**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#49-%D0%B4%D0%B2%D0%BE%D0%B9%D0%BD%D1%8B%D0%B5%D0%BF%D0%BB%D0%B0%D0%B2%D0%B0%D1%8E%D1%89%D0%B8%D0%B5%D1%86%D0%B5%D0%BB%D0%BE%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5--%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F)

**Double/Float/Integer и другие типы чисел**

Числовые утверждения — это сравнение числовых значений в пределах заданного смещения или без него. Например, если вы хотите проверить, равны ли два значения в соответствии с заданной точностью, мы можем сделать следующее:

assertThat(5.1).isEqualTo(5, withPrecision(1d));

Обратите внимание, что мы используем уже импортированный вспомогательный метод withPrecision(Double offset) для создания объектов Offset .

Дополнительные утверждения см . в [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractDoubleAssert.html) по AbstractDoubleAssert .

**4.10. Утверждения InputStream[​](https://for-each.dev/lessons/b/-introduction-to-assertj" \l "410-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F--inputstream" \o "Прямая ссылка на 410-утверждения--inputstream)**

Доступно только одно специфическое для InputStream утверждение:

* hasSameContentAs (ожидаемый поток ввода)

и в действии:

assertThat(given).hasSameContentAs(expected);

**4.11. Утверждения карты**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#411-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F--%D0%BA%D0%B0%D1%80%D1%82%D1%8B)

Утверждения карты позволяют вам проверить, содержит ли карта определенную запись, набор записей или ключи/значения по отдельности.

А здесь вы можете увидеть пример утверждения, которое проверяет, не является ли данная карта пустой, содержит ли числовой ключ «2», не содержит ли числовой ключ «10» и содержит ли запись: ключ: 2, значение: «а» :

assertThat(map)  
 .isNotEmpty()  
 .containsKey(2)  
 .doesNotContainKeys(10)  
 .contains(entry(2, "a"));

Дополнительные утверждения см . в [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractMapAssert.html)по AbstractMapAssert .

**4.12. Выбрасываемые утверждения**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#412-%D0%B2%D1%8B%D0%B1%D1%80%D0%B0%D1%81%D1%8B%D0%B2%D0%B0%D0%B5%D0%BC%D1%8B%D0%B5--%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F)

Выбрасываемые утверждения позволяют, например: проверять сообщения об исключениях, трассировки стека, проверять причины или проверять, было ли уже выброшено исключение.

Давайте посмотрим на пример утверждения, которое проверяет, было ли выброшено данное исключение, и имеет сообщение, оканчивающееся на «с»:

assertThat(ex).hasNoCause().hasMessageEndingWith("c");

Дополнительные утверждения см . в [документации](https://joel-costigliola.github.io/assertj/core-8/api/org/assertj/core/api/AbstractThrowableAssert.html) по AbstractThrowableAssert .

**5. Описание утверждений**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#5-%D0%BE%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D0%B5-%D1%83%D1%82%D0%B2%D0%B5%D1%80%D0%B6%D0%B4%D0%B5%D0%BD%D0%B8%D0%B9)

Чтобы достичь еще более высокого уровня детализации, вы можете создавать динамически генерируемые пользовательские описания для своих утверждений. Ключом к этому является метод as(String description, Object… args) .

Если вы определяете свое утверждение следующим образом:

assertThat(person.getAge())  
 .as("%s's age should be equal to 100", person.getName())  
 .isEqualTo(100);

вот что вы получите при запуске тестов:

[Alex's age should be equal to 100] expected:<100> but was:<34>

**6. Ява 8**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#6-%D1%8F%D0%B2%D0%B0-8)

AssertJ в полной мере использует возможности функционального программирования Java 8. Давайте погрузимся в пример и посмотрим его в действии. Сначала давайте посмотрим, как мы это делаем в Java 7:

assertThat(fellowshipOfTheRing)  
 .filteredOn("race", HOBBIT)  
 .containsOnly(sam, frodo, pippin, merry);

Здесь мы фильтруем коллекцию по расе Hobbit и в Java 8 можем сделать примерно так:

assertThat(fellowshipOfTheRing)  
 .filteredOn(character -> character.getRace().equals(HOBBIT))  
 .containsOnly(sam, frodo, pippin, merry);

Мы будем изучать возможности AssertJ Java8 в следующей статье из этой серии. Приведенные выше примеры были взяты с [сайта](https://joel-costigliola.github.io/assertj/) AssertJ .

**7. Заключение**[**​**](https://for-each.dev/lessons/b/-introduction-to-assertj#7--%D0%B7%D0%B0%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)

В этой статье мы кратко рассмотрели возможности, которые дает нам AssertJ, наряду с наиболее популярными утверждениями для основных типов Java.

Реализацию всех примеров и фрагментов кода можно найти в [проекте на GitHub](https://github.com/for-each/tutorials/tree/master/testing-modules/assertion-libraries) .