OZON tech

Побеждаем Null с NullAway и JSpecify (опять)

Пётр Портнов
ведущий разработчик
информационных систем, Ozon

Пётр Портнов

Разработчик поисковой платформы Ozon,
open-source контрибутор, читаю пары
по Java и по алгоритмам

pportnov@ozon.ru

Дисклеймеры & Вводные

01 **Доклад предназначен для всех**

02 **Будут хардкорные юзкейсы**

03 **Примеры кода могут быть упрощены**

04 **Если вы видите паззлер, то это не паззлер**

05 ***[н а л]* или *[н у л]***

AGENDA

01 **Всё ещё NPE** 

02 **Non-null by default**

03 **JSpecify & NullAway**

04 **Интероп**

05 **Проблемы**

06 **Результаты**

Наша история

ДО

- Непонятная nullability
- Precondition’ы по поводу и без
- Даже NPE проде

ПОСЛЕ

- Понятная nullabilty
- Проверки только там, где они нужны
- 0 NPE

01 {

Всё ещё NPE

ВНИМАНИЕ

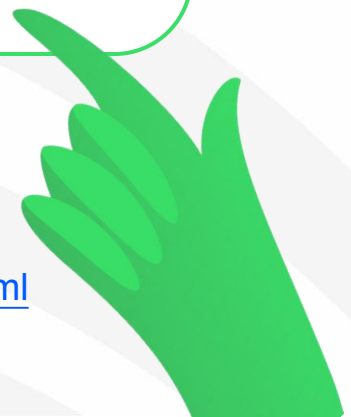
Кто знает, что такое NPE?

NullPointerException

Thrown when an application attempts to use `null` in a case where an object is required. These include:

- Calling the instance method of a `null` object.
- Accessing or modifying the field of a `null` object.
- Taking the length of `null` as if it were an array.
- Accessing or modifying the slots of `null` as if it were an array.
- Throwing `null` as if it were a `Throwable` value.

Applications should throw instances of this class to indicate other illegal uses of the `null` object. `NullPointerException` objects may be constructed by the virtual machine as if suppression were disabled and/or the stack trace was not writable.



https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/lang/NullPointerException.html

На древность намекает даже слово `Pointer` в названии

Посмотрим на практике

Видите ли вы проблему?

```
private final Map<FeatureType, FeatureMapper> mappersByType;

...

public MappedFeature map(Feature feature) {
    return mappersByType
        .get(feature.type())
        .map(feature);
}
```

Мы тоже не увидели

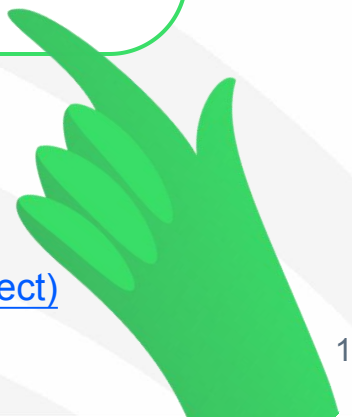
Incident: При релизе сервиса o2-midway
возросла нагрузка на сервис feature-meta-store

Cause: `java.lang.NullPointerException:`

```
Cannot invoke "TypedFeatureMapper.map(Feature)"
because the return value of
"java.util.Map.get(Object)" is null
```

`Map.get(Object)`

Returns:
the value to which the specified key is mapped, or `null` if this map contains no mapping for the key



https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/util/Map.html#get(java.lang.Object)

Ну ладно, будем бдить за `Map#get(Object)`

Ведь всё остальное просто

Ну ладно, будем бдить за `Map#get(Object)`

Просто же?

Ах Streamы мои Streamы

```
public Set<String> aneks() {
    var set = new HashSet<String>();
    set.add("дилижанс");
    set.add("каравелло");
    set.add("нюанс");
    // Не всем анекдоты ещё написаны...
    set.add(null);

    return set;
}
```

Ах Streamы мои Streamы

```
var aneks = aneks().stream();
System.out.println(
    aneks.collect(Collectors.toList())
);
System.out.println(
    aneks.collect(Collectors.toUnmodifiableList())
);
```

Ах Streamы мои Streamы

```
var aneks = aneks();
System.out.println(
    aneks.stream().collect(Collectors.toList())
);
System.out.println(
    aneks.stream().collect(Collectors.toUnmodifiableList())
);
```

Ах Streamы мои Streamы

```
[null, дилижанс, каравелло, нюанс]
Exception in thread "main" java.lang.NullPointerException
     at Objects.requireNonNull(Objects.java:220)
     at ImmutableCollections.listFromTrustedArray
     at ImmutableCollections$Access$1.listFromTrustedArray
     at Collectors.lambda$toUnmodifiableList$1
     at ReferencePipeline.collect
     at main

Process finished with exit code 1
```

А если попробовать эквивалентную замену от IDEA?

```
var aneks = aneks();
System.out.println(
    aneks.stream().collect(Collectors.toList())
);
System.out.println(
    aneks.stream().toList()
);
```

Всё-таки не эквивалентную

```
[null, дилижанс, каравелло, нюанс]
[null, дилижанс, каравелло, нюанс]

Process finished with exit code 0
```

Ну и, наконец, параноидальный пример

Где может быть NPE?

```
interface Something {
    Optional<Metadata> metadata();
}

static Something something(long id) { ... }

var meta = something(123)
    .metadata()
    .orElse(Metadata.DEFAULT);
```

Ну и, наконец, параноидальный пример

Туть

```
interface Something {
    Optional<Metadata> metadata();
}

static Something something(long id) { ... }

var meta = something(123)
    .metadata()
    .orElse(Metadata.DEFAULT);
```

Ну и, наконец, параноидальный пример

И туть

```
interface Something {
    Optional<Metadata> metadata();
}

static Something something(long id) { ... }

var meta = something()
    .metadata()
    .orElse(Metadata.DEFAULT);
```

Паранойя?

Mock returning java.util.Optional should return Optional.empty() by default (Java 8) #191

New issue

Closed

netmikey opened on Apr 2, 2015

(From the discussion at https://code.google.com/p/mockito/issues/detail?id=497)

> A stubbed method that returns `Optional<?>` should return `Optional.empty()` if no explicit behavior is defined. This is equivalent to returning null for regular methods.
>
> Introducing this feature would add no extra dependencies. However, it would require a test whether the code is running in Java 8.

Since the thread ends with the request for a GitHub ticket and I couldn't find one... here it is :)

Assignees

No one assigned

Labels

enhancement java-8

Type

No type

Projects

No projects

https://github.com/mockito/mockito/issues/191

Я уже жаловался на это

https://jokerconf.com/archive/2024/talks/7ac7609680e5451ab353544934bef137/

OZON tech

Побеждаем Null с NullAway и JSpecify (опять)

Тоже я,
но курсивом

https://ozon.ru/t/flYxOZP


ЧЕТЫРЕЖДЫ
Special
for JPoint
ЯРОСТЬ!!!

02

Non-null by default

Давайте запретим `null`ы

У нас была

Своя аннотация

```
/// Пакет использует семантику non-null по умолчанию.
@Nonnull
@Documented
@Target({TYPE, PACKAGE})
@Retention(RUNTIME)
@TypeQualifierDefault({METHOD, PARAMETER, FIELD, CONSTRUCTOR})
public @interface NonNullByDefault {}
```

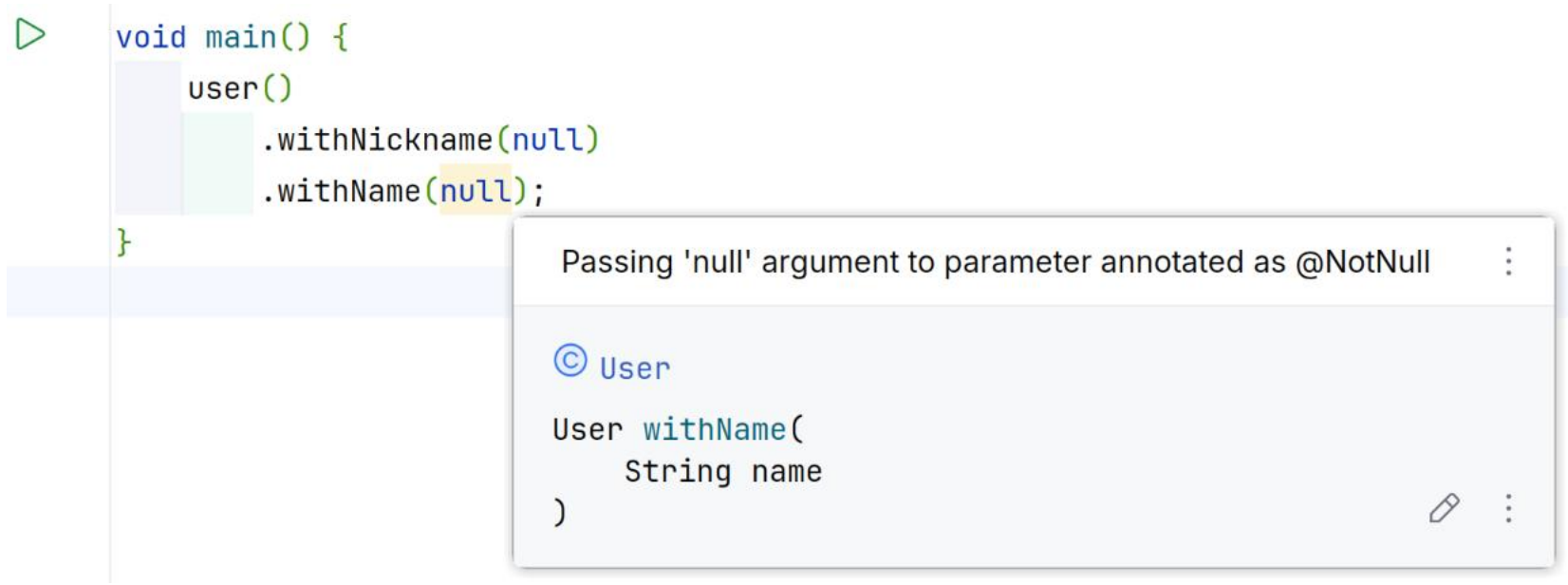
Ставим на пакет и понимаем свою nullability

```
@NonNullByDefault
package ru.ozon.jpoint2026;
```

Обещание nullability

```
package ru.ozon.jpoint2026;

interface User {
    User withName(String name);

    User withNickname(@Nullable String nickname);
}
```

И IDEA нас понимает



Осталось разметить

Но чем?

Java-X

javax.annotation.@Nullable

```
@Documented
@TypeQualifierNickname
@Nonnull(when = UNKNOWN)
@Retention(RUNTIME)
public @interface Nullable {}
```

Что такое TYPE_USE

Аннотация относится к ближайшему правому типу

```
@Nullable List<String> nullableList;
List<@Nullable String> listOfNullables;

@Nullable Map<String, User> nullableMap;
Map<@Nullable String, User> mapOfNullableKeys;
Map<String, @Nullable User> mapOfNullableValues;

String @Nullable [] nullableArray;
@Nullable String[] arrayOfNullable;
```

Что такое TYPE_USE

Аннотация относится к ближайшему правому типу

```
@Nullable List<String> nullableList;
List<@Nullable String> listOfNullables;

@Nullable Map<String, User> nullableMap;
Map<@Nullable String, User> mapOfNullableKeys;
Map<String, @Nullable User> mapOfNullableValues;

String @Nullable [] nullableArray;
@Nullable String[] arrayOfNullable;
```

Что такое TYPE_USE

Аннотация относится к ближайшему правому типу

```
@Nullable List<String> nullableList;
List<@Nullable String> listOfNullables;

@Nullable Map<String, User> nullableMap;
Map<@Nullable String, User> mapOfNullableKeys;
Map<String, @Nullable User> mapOfNullableValues;

String @Nullable [] nullableArray;
@Nullable String[] arrayOfNullable;
```

Что такое TYPE_USE

Аннотация относится к ближайшему правому типу

```
@Nullable List<String> nullableList;
List<@Nullable String> listOfNullables;

@Nullable Map<String, User> nullableMap;
Map<@Nullable String, User> mapOfNullableKeys;
Map<String, @Nullable User> mapOfNullableValues;

String @Nullable [] nullableArray;
@Nullable String[] arrayOfNullable;
```

Что такое TYPE_USE

Аннотация относится к ближайшему правому типу

```
@Nullable List<String> nullableList;
List<@Nullable String> listOfNullables;

@Nullable Map<String, User> nullableMap;
Map<@Nullable String, User> mapOfNullableKeys;
Map<String, @Nullable User> mapOfNullableValues;

String @Nullable [] nullableArray;
@Nullable String[] arrayOfNullable;
```

JetBrains

org.jetbrains.annotations.@Nullable

```
@Documented
@Retention(CLASS)
@Target({
    METHOD, FIELD, PARAMETER, LOCAL_VARIABLE, TYPE_USE
})
public @interface Nullable {
  @NonNls String value() default "";
}
```

Javadoc’и выглядят странно

```
/// Does something!
///
/// @param str input string
public void doSomething(@Nullable String str) { ... }
```

doSomething

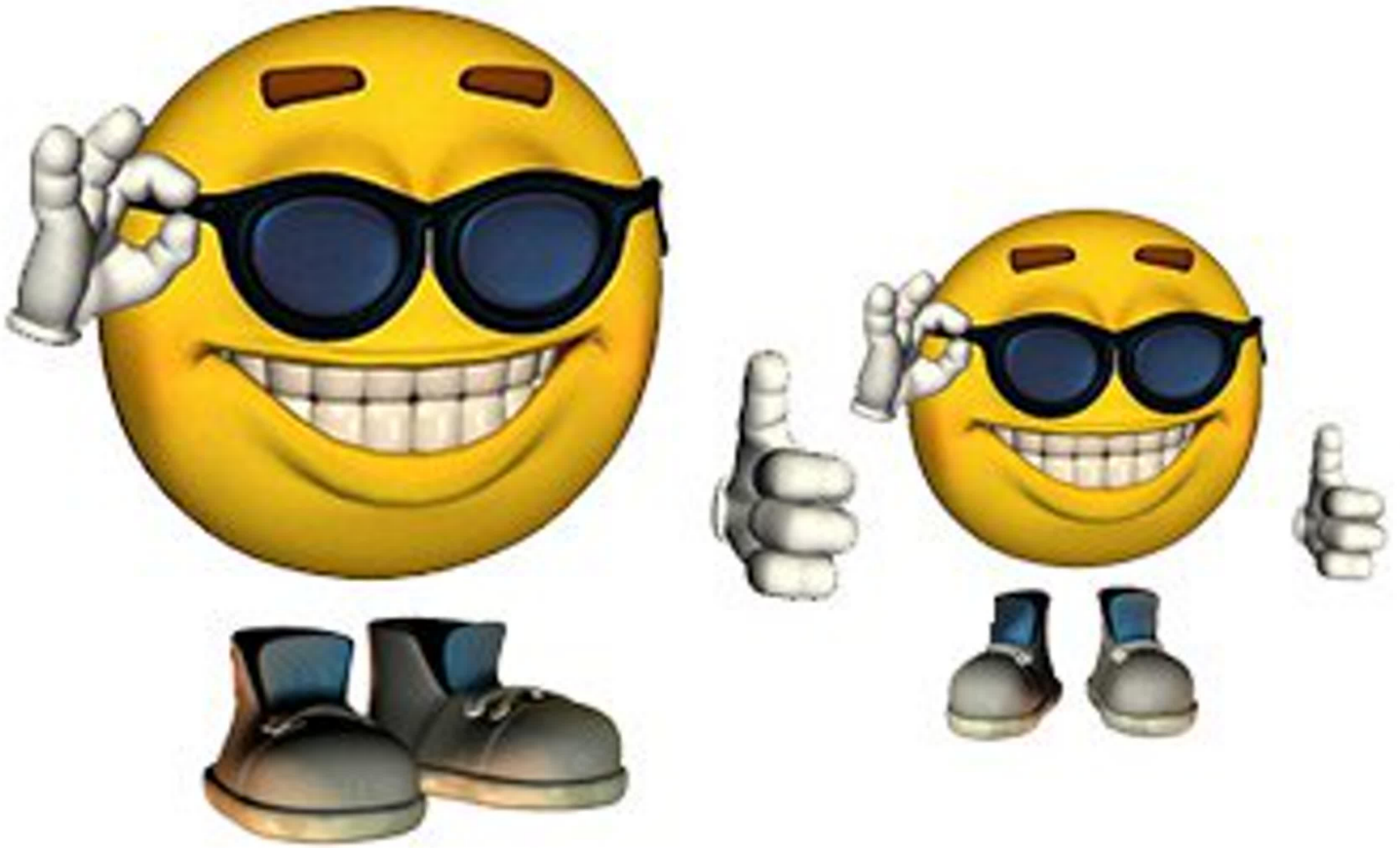
```
public void doSomething(@Nullable
                        @Nullable String str)
```

Does something!

Parameters:

`str` - input string

`@Nullable` мне и моему параметру тоже



Eclipse JDT

`org.eclipse.jdt.annotation.@Nullable`

```
@Documented
@Retention(RetentionPolicy.CLASS)
@Target({ TYPE_USE })
public @interface Nullable {}
```

Я карта

jakarta.annotation.@Nullable

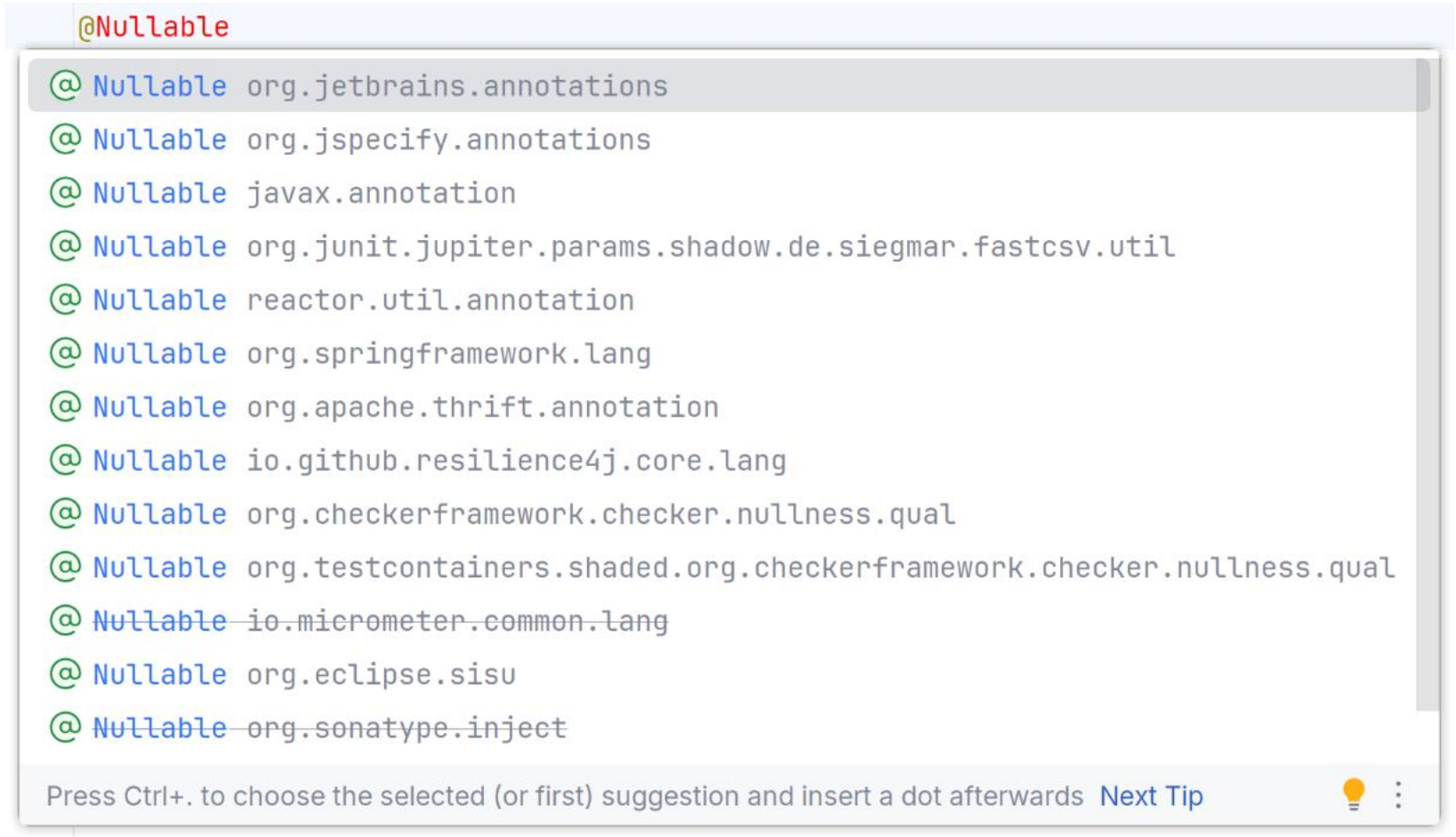
```
@Documented
@Retention(RUNTIME)
public @interface Nullable {}
```

Чужие велосипеды

`org.springframework.lang.Nullable`

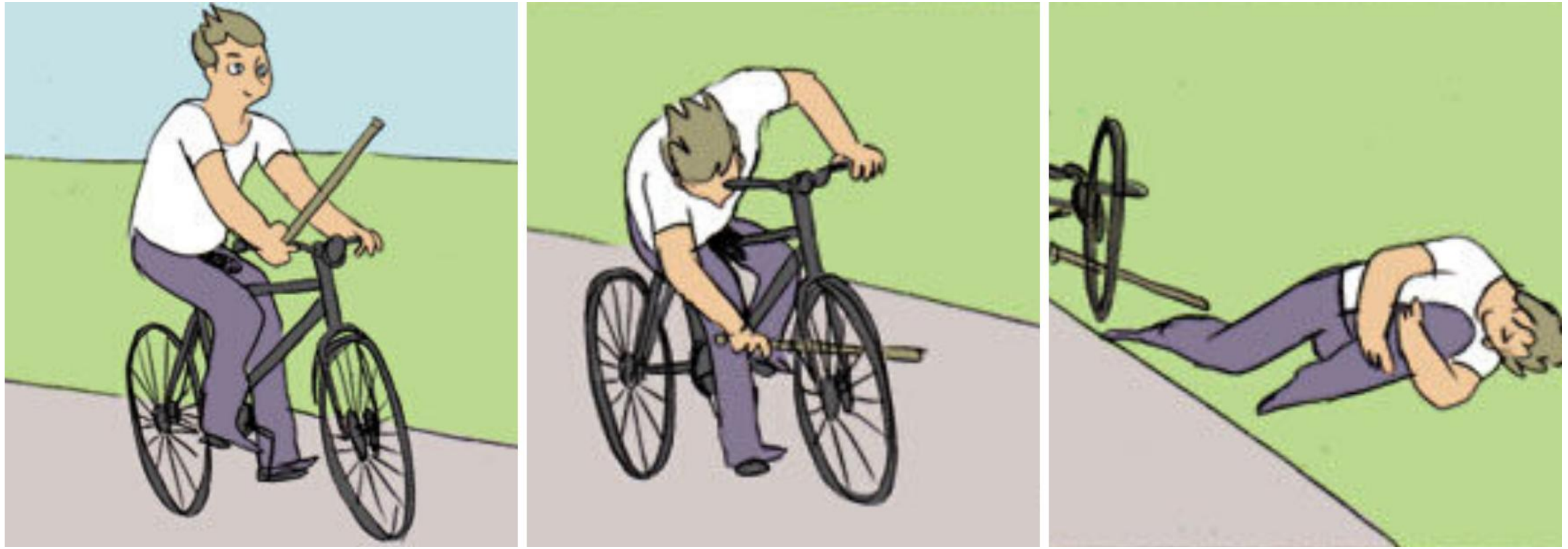
```
@Target({METHOD, PARAMETER, FIELD})
@Retention(RUNTIME)
@Documented
@CheckForNull
@TypeQualifierNickname
@Deprecated(since = "7.0")
public @interface Nullable {}
```

Чужие велосипеды

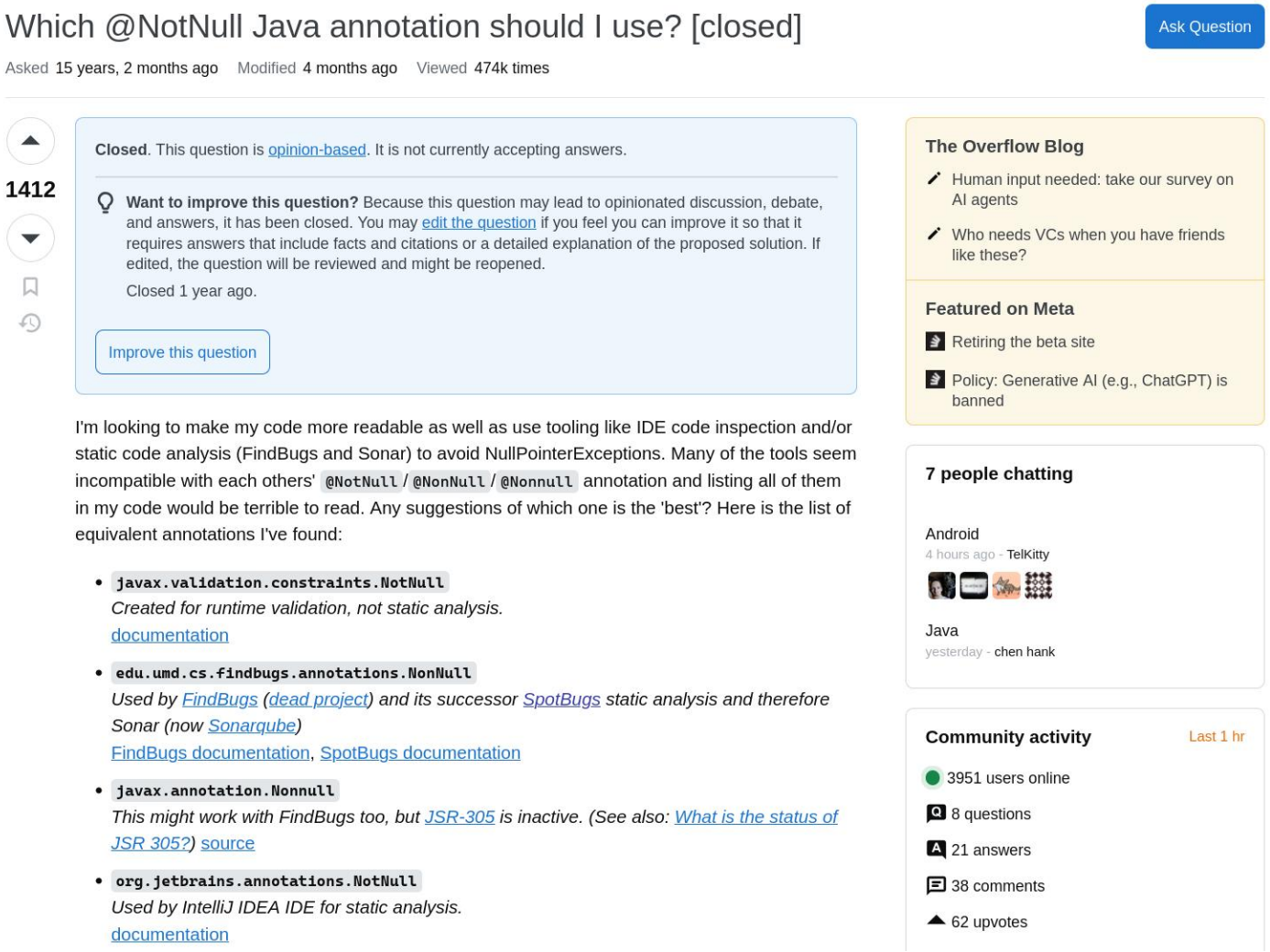


Обязательно

Свой велосипед



Проблема стара как... Java 5?



Which @NotNull Java annotation should I use? [closed]

Ask Question

Asked 15 years, 2 months ago Modified 4 months ago Viewed 474k times

1412

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Because this question may lead to opinionated discussion, debate, and answers, it has been closed. You may edit the question if you feel you can improve it so that it requires answers that include facts and citations or a detailed explanation of the proposed solution. If edited, the question will be reviewed and might be reopened.

Closed 1 year ago.

Improve this question

I'm looking to make my code more readable as well as use tooling like IDE code inspection and/or static code analysis (FindBugs and Sonar) to avoid NullPointerExceptions. Many of the tools seem incompatible with each others' `@NotNull` / `@NonNull` / `@Nonnull` annotation and listing all of them in my code would be terrible to read. Any suggestions of which one is the 'best'? Here is the list of equivalent annotations I've found:

- `javax.validation.constraints.NotNull`
  Created for runtime validation, not static analysis.
  documentation
- `edu.umd.cs.findbugs.annotations.NonNull`
  Used by FindBugs (dead project) and its successor SpotBugs static analysis and therefore Sonar (now Sonarqube)
  FindBugs documentation, SpotBugs documentation
- `javax.annotation.Nonnull`
  This might work with FindBugs too, but JSR-305 is inactive. (See also: What is the status of JSR 305?) source
- `org.jetbrains.annotations.NotNull`
  Used by IntelliJ IDEA IDE for static analysis.
  documentation

The Overflow Blog

Human input needed: take our survey on AI agents

Who needs VCs when you have friends like these?

Featured on Meta

Retiring the beta site

Policy: Generative AI (e.g., ChatGPT) is banned

7 people chatting

Android
4 hours ago - TelKitty

Java
yesterday - chen hank

Community activity Last 1 hr

3951 users online

8 questions

21 answers

38 comments

62 upvotes

https://stackoverflow.com/questions/4963300/which-notnull-java-annotation-should-i-use

Мы не говорим о JSR 305

https://jcp.org/en/jsr/detail?id=305

Чужой велосипед!

41

Now that JSpecify have released version 1.0 of their annotations, those should be the way to go. In fact: both the introduction page and their 2019 presentation actively link to this very question and specify that their goal is for it to finally have a good answer.

It has major participants like Android, Guava and Kotlin. Also, as of October 2025, Projects such as Spring Framework / Spring Boot have begun migrating from other annotations to jSpecify.

Share Improve this answer Follow

edited Oct 19, 2025 at 16:12

Jens Bannmann

5,165 ● 7 ● 54 ● 81

answered Aug 25, 2021 at 9:03

Cristan

14.7k ● 8 ● 74 ● 77

https://stackoverflow.com/a/68919918

JSPECIFY

BASED ON THAT STACKOVERFLOW THREAD

and xkcd 927

03 {

JSpecify & NullAway

Я уже жаловался на это

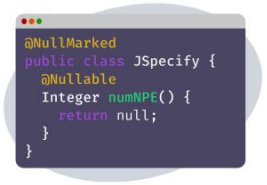
https://jokerconf.com/archive/2024/talks/7ac7609680e5451ab353544934bef137/

Что они сами о себе пишут?

JSpecify

Standard Annotations for Java Static Analysis

Learn More



Standard Annotations

JSpecify is releasing the first artifact of tool-independent annotations for powering static analysis checks in your Java code.

Next Level Static Analysis

JSpecify defines precise semantics, letting analysis tools find more bugs, and more consistently. Library owners won't have to decide which tool to support.

Community Effort

JSpecify is developed by consensus of members representing a variety of stakeholders in Java static analysis, and we welcome your participation.

https://jspecify.dev/

Кто в деле?

Организация	Проект
EISOP Team	EISOP
Google	Android, Error Prone, Guava
JetBrains	Kotlin, IntelliJ IDEA
Meta*	Infer
Microsoft	Azure SDK for Java

* — признана экстремистской организацией и запрещена на территории Российской Федерации

https://jspecify.dev/about/

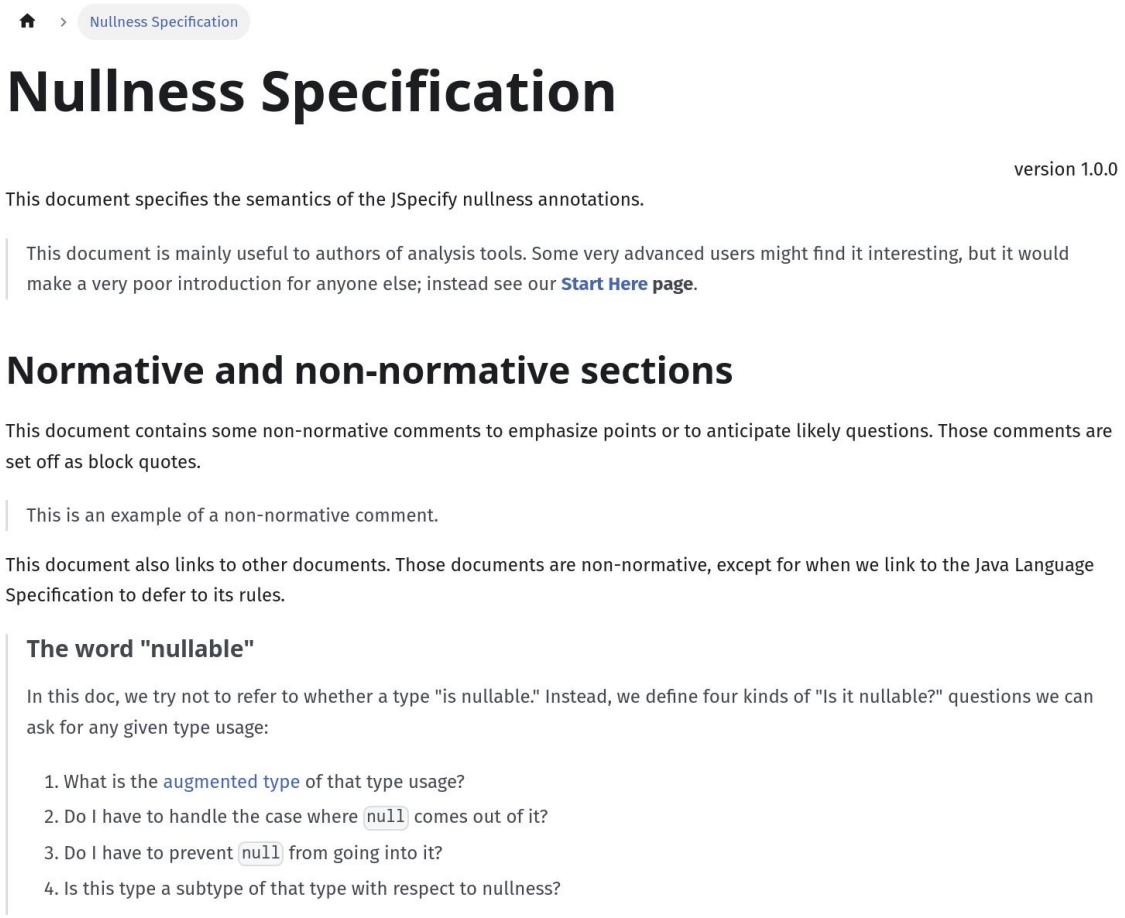
Кто в деле?

Организация	Проект
Oracle	OpenJDK
PMD Team	PMD
Sonar	SonarQube, SonarCloud, SonarLint
Square	(various)
Uber	NullAway
Broadcom	Spring

https://jspecify.dev/about/

В чём, вообще, смысл?

Потрясающая спека



Nullness Specification

Nullness Specification

version 1.0.0

This document specifies the semantics of the JSpecify nullness annotations.

> This document is mainly useful to authors of analysis tools. Some very advanced users might find it interesting, but it would make a very poor introduction for anyone else; instead see our **Start Here page**.

Normative and non-normative sections

This document contains some non-normative comments to emphasize points or to anticipate likely questions. Those comments are set off as block quotes.

> This is an example of a non-normative comment.

This document also links to other documents. Those documents are non-normative, except for when we link to the Java Language Specification to defer to its rules.

> ### The word "nullable"
>
> In this doc, we try not to refer to whether a type "is nullable." Instead, we define four kinds of "Is it nullable?" questions we can ask for any given type usage:
>
> 1. What is the augmented type of that type usage?
> 2. Do I have to handle the case where `null` comes out of it?
> 3. Do I have to prevent `null` from going into it?
> 4. Is this type a subtype of that type with respect to nullness?

https://jspecify.dev/docs/spec/

В чём, вообще, смысл?

Подробная спека...


Nullness Specification

https://jspecify.dev/docs/spec/

Как применить?

https://jspecify.dev/docs/applying/

- Найти и разметить nullable
- Добавить `@NullMarked`
- Прогнать анализ на размеченном коде
- Прогнать анализ на вызывающем коде
- Повторить для нового кода

А ещё: исполнение спек можно делегировать...

Но давайте попробуем

Local Reasoning

Вызывающий код оперирует только публичной информацией (сигнатурами)

Local reasoning

Где баг?

```
void main() {
    foo().bar();
}

Foo foo() {
    return ThreadLocalRandom.current().nextBoolean()
        ? new Foo();
        : null;
}
```

Local reasoning

Посмотрим отдельно: анализ `main`

```
void main() {
    foo().bar();
}

Foo foo();
```

Local reasoning

Посмотрим отдельно: анализ `foo`

```
void main();



Foo foo() {
    return ThreadLocalRandom.current().nextBoolean()
        ? new Foo();
        : null;
}
```

Local reasoning

Тут (был) баг

```
void main();
```

```
@Nullable Foo foo() {
    return ThreadLocalRandom.current().nextBoolean()
        ? new Foo();
        : null;
}
```

NULLAWAY

NullAway

Статический анализатор, плагин к Error Prone, реализующий nullability-анализ.

https://github.com/uber/nullaway

Я уже жаловался на это

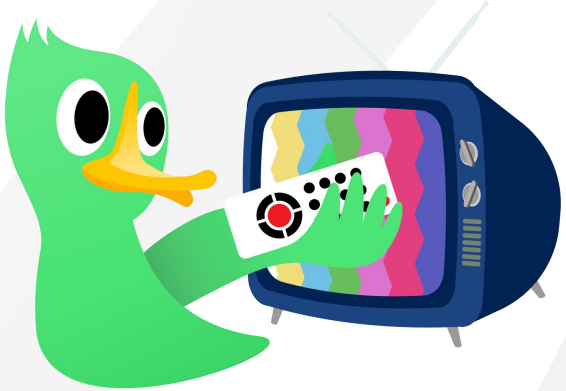
OZON tech

Ловим баги с Error Prone [18+]

Пётр Портнов
ведущий разработчик
информационных систем, Ozon

https://snowone.ru/talks/509a9ae7cc1e43738dfce080df5a19db/

Недостаточно подсветки в IDE



NullAway умеет работать как в «классическом», так и в JSpecify-режиме

Наши настройки

Настоятельно рекомендуем

```
nullaway {
    jspecifyMode = true
    onlyNullMarked = true
}

errorprone {
    error('RequireExplicitNullMarking')
    nullaway {
        error()
    }
}
```

Итого

- **JSpecify** — спецификация и аннотации
- **NullAway** — реализация спецификации в форме плагина для Error Prone

Простые generic-и

Как исправить анеки?

```
public Set<@Nullable String> aneks() {
    var set = new HashSet<String>();
    set.add("дилижанс");
    set.add("каравелло");
    set.add("нюанс");
    // Не всем анекдоты ещё написаны...
    set.add(null);

    return set;
}
```

Простые generic-и

Как исправить анеки?

```
public Set<@Nullable String> aneks() {
    ...
    return set;
//         ^
// error: [NullAway] incompatible types:
//  HashSet<String> cannot be converted to Set<@Nullable
String>
//  (HashSet<String> is a subtype of Set<String>)
}
```

Простые generic-и

Вот теперь норм

```
public Set<@Nullable String> aneks() {
    var set = new HashSet<@Nullable String>();
    set.add("дилижанс");
    set.add("каравелло");
    set.add("нюанс");
    // Не всем анекдоты ещё написаны...
    set.add(null);

    return set;
}
```

@Nullable — часть типа

При этом для самих локальных переменных он выводится:

```
void conditional(boolean flag) {
    var str = flag ? "Yay" : null;
    System.out.println(str.length());
//                             ^
// error: [NullAway] dereferenced expression str is
@Nullable
}
```

Свой тип

При этом для самих локальных переменных он выводится:

```
record Container<T>(T value) {}

void main() {
    var container = new Container<String>(null);
}
```

Свой тип

Свои проблемы

```
record Container<T>(T value) {}

void main() {
    var container = new Container<String>(null);
//                                        ^
// error: [NullAway] passing @Nullable parameter 'null'
//  where @NonNull is required
}
```

Свой тип

Это мы теперь умеем!

```
record Container<T>(T value) {}

void main() {
    var container = new Container<@Nullable String>(null);
}
```

Свой тип

Или нет

```
record Container<T>(T value) {}

void main() {
    var container = new Container<@Nullable String>(null);
//                                ^
// error: [NullAway] Generic type parameter
//  cannot be @Nullable,
//  as type variable T of type Container
//  does not have a @Nullable upper bound
}
```

Ковариантность, контрвариантность, инвариантность…

Свой тип

Явно разрешили `null`ы и радуемся

```
record Container<T extends @Nullable Object>(T value) {}

void main() {
    var container = new Container<@Nullable String>(null);
}
```

Poly-null

```
static <T, R> @Nullable R tryMap(
    @Nullable T value,
    Function<? super T, ? extends R> mapper
) {
    return value == null ? null : mapper.apply(value);
}
```

Poly-null

```
void main() {
    var count = tryMap(
        maybeValue(),
        String::getBytes
    ).length;
}

@Nullable String maybeValue() { ... }
```

Poly-null

```
void main() {
    var count = tryMap(
        maybeValue(),
        String::getBytes
    ).length;
//   ^
// error: [NullAway] dereferenced expression
//  tryMap(maybeValue(), String::getBytes)
//  is @Nullable
}
```

Poly-null

```
void main() {
    var count = tryMap(
        maybeValue(),
        String::getBytes
    ).length;
    if (bytes == null) {
        throw new IllegalArgumentException("...");
    }
    var count = bytes.length;
}
```

Poly-null

А если там всегда non-null?

```
void main() {
    var count = tryMap(
        value(),
        String::getBytes
    ).length;
}

String value() { ... }
```

Poly-null

А если там всегда non-null?

```
void main() {
    var count = tryMap(
        value(),
        String::getBytes
    ).length;
//   ^
// error: [NullAway] dereferenced expression
//  tryMap(maybeValue(), String::getBytes)
//  is @Nullable
}
```

Poly-null

Poly-null

Вспоминаем про local reasoning

```
static <T, R> @Nullable R tryMap(
    @Nullable T value,
    Function<? super T, ? extends R> mapper
);

String value();

var count = tryMap(
    value(),
    String::getBytes
).length;
```

Poly-null

Надо как-то разметить `tryMap`

```
static <T, R> @Nullable R tryMap(
    @Nullable T value,
    Function<? super T, ? extends R> mapper
);
```

Poly-null

Есть идея

`@PolyNull`

A `@PolyNull` annotation could address this case:

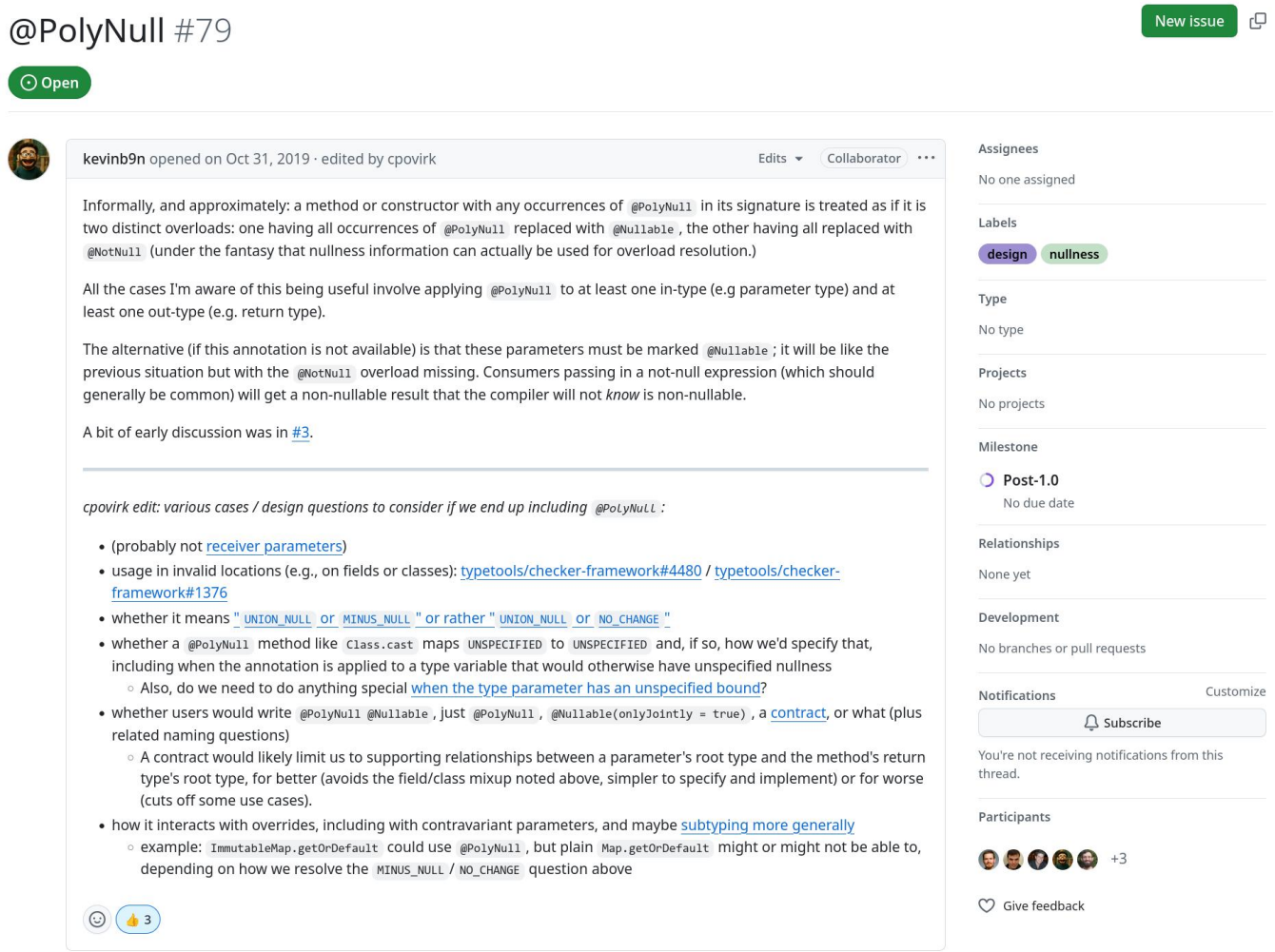
```
@NullMarked
class Class</*non-null*/ T> {
  @PolyNull T cast(@PolyNull Object obj) { ... }
}
```

It declares that nullness is conserved between inputs and outputs, or in layperson's terms, "null thing goes in, null thing comes out".

https://github.com/jspecify/jspecify/wiki/polynull

Poly-null

Есть нюанс



@PolyNull #79

New issue

Open

kevinb9n opened on Oct 31, 2019 · edited by cpovirk

Edits · Collaborator

Informally, and approximately: a method or constructor with any occurrences of `@PolyNull` in its signature is treated as if it is two distinct overloads: one having all occurrences of `@PolyNull` replaced with `@Nullable`, the other having all replaced with `@NotNull` (under the fantasy that nullness information can actually be used for overload resolution.)

All the cases I'm aware of this being useful involve applying `@PolyNull` to at least one in-type (e.g parameter type) and at least one out-type (e.g. return type).

The alternative (if this annotation is not available) is that these parameters must be marked `@Nullable`; it will be like the previous situation but with the `@NotNull` overload missing. Consumers passing in a not-null expression (which should generally be common) will get a non-nullable result that the compiler will not *know* is non-nullable.

A bit of early discussion was in #3.

---

cpovirk edit: various cases / design questions to consider if we end up including `@PolyNull`:

- (probably not receiver parameters)
- usage in invalid locations (e.g., on fields or classes): typetools/checker-framework#4480 / typetools/checker-framework#1376
- whether it means "`UNION_NULL` or `MINUS_NULL`" or rather "`UNION_NULL` or `NO_CHANGE`"
- whether a `@PolyNull` method like `Class.cast` maps `UNSPECIFIED` to `UNSPECIFIED` and, if so, how we'd specify that, including when the annotation is applied to a type variable that would otherwise have unspecified nullness
  - Also, do we need to do anything special when the type parameter has an unspecified bound?
- whether users would write `@PolyNull @Nullable`, just `@PolyNull`, `@Nullable(onlyJointly = true)`, a contract, or what (plus related naming questions)
  - A contract would likely limit us to supporting relationships between a parameter's root type and the method's return type's root type, for better (avoids the field/class mixup noted above, simpler to specify and implement) or for worse (cuts off some use cases).
- how it interacts with overrides, including with contravariant parameters, and maybe subtyping more generally
  - example: `ImmutableMap.getOrDefault` could use `@PolyNull`, but plain `Map.getOrDefault` might or might not be able to, depending on how we resolve the `MINUS_NULL` / `NO_CHANGE` question above

👍 3

Assignees
No one assigned

Labels
design nullness

Type
No type

Projects
No projects

Milestone
Post-1.0
No due date

Relationships
None yet

Development
No branches or pull requests

Notifications Customize
Subscribe
You're not receiving notifications from this thread.

Participants
+3

Give feedback

https://github.com/jspecify/jspecify/issues/79

Poly-null дома

IDEA подсказывает

```
@Contract("null, _ -> null")
static <T, R> @Nullable R tryMap(
    @Nullable T value,
    Function<? super T, ? extends R> mapper
) {
    return value == null ? null : mapper.apply(value);
}
```

Poly-null дома

`org.jetbrains.annotations.@Contract`

```
@Contract("!null, _ -> !null; null, _ -> null")
static <T, R> @Nullable R tryMap(
    @Nullable T value,
    Function<? super T, ? extends R> mapper
) {
    return value == null ? null : mapper.apply(value);
}
```

Контракты

Ментальная модель

```
@Contract("!null, _ -> !null")
static <T, R> @Nullable R tryMap(@Nullable T value, ...);

// ≈

static <T, R>           R tryMap(          T value, ...);

static <T, R> @Nullable R tryMap(@Nullable T value, ...);
```

Контракты

Ментальная модель

```
@Contract("null, _ -> null")
static <T, R> @Nullable R tryMap(@Nullable T value, ...);

// ≈

static <T, R>       null R tryMap(       null value, ...);

static <T, R> @Nullable R tryMap(@Nullable T value, ...);
```

Контракты

Можно использовать и для других правил

```
@Contract("null -> fail")
static void prohibitNull(@Nullable Object object) {
    if (object == null) {
        throw new BusinessException("...");
    }
}

var str = maybeValue();
prohibitNull(str);
System.out.println(str.length());
```

Контракты

Можно использовать и для других правил

```
@Contract("null -> fail")
static void prohibitNull(@Nullable Object object) {
    if (object == null) {
        throw new BusinessException("...");
    }
}

var str = maybeValue();
prohibitNull(str);
System.out.println(str.length());
```

Контракты

Можно использовать и для других правил

```
@Contract("null -> true")
static boolean isNullOrBlank(@Nullable String string) {
    return string == null || string.isBlank();
}

var str = maybeValue();
if (!isNullOrBlank(str)) {
    System.out.println(str.length());
}
```

Контракты

Можно использовать и для других правил

```
@Contract("null -> true")
static boolean isNullOrBlank(@Nullable String string) {
    return string == null || string.isBlank();
}

var str = maybeValue();
if (!isNullOrBlank(str)) {
    System.out.println(str.length());
}
```

Контракты

NullAway экспериментально умеет следить и за их соблюдением:

```
nullaway {
    checkContracts = true
}
```

Но у меня не завелось ¯_(ツ)_/¯

Контракты

Нюанс

Introduce method contracts #757

New issue

Open

sdeleuze opened on Aug 13, 2025 · edited by sdeleuze — Edits — Collaborator

With projects like Spring, Micrometer and many others starting to leverage JSpecify combined with a nullness checker, there is a very popular ask (both from Spring commiters and from community members) that we need parameter contracts to be able to achieve 0 checker errors (green build) without too much annotations like `@SupressWarnings("NullAway")`. The scope of JSpecify 1.0.0 does not allow to reach that IMO reasonable goal, leading to the proliferation of multiple variants of `@Contract` (JetBrains one, Spring one) annotation + requiring special support in the various checkers. This is not ideal.

Method level `@Contract` leverages its own expression language, and my understanding is that a majority of members of the JSpecify working group (including me) want to avoid such expression language. I tend to think we could replace the method level `@Contract` annotation with an expression language by a parameter level annotation with enums. It would cover the majority of the use cases, even if not 100%.

I am wondering if it could make sense for JSpecify to introduce a new annotation like `@ParameterContract` (name to be refined, hopefully with a shorter one), with no retention at runtime, defined as follow:

```
@Documented
@Target(ElementType.PARAMETER)
public @interface ParameterContract {
    Condition when();
    Effect then();
}

public enum Condition {
    NONE,
    IS_TRUE,
    IS_FALSE,
    IS_NULL,
    IS_NON_NULL
}

public enum Effect {
    RETURN_TRUE,
    RETURN_FALSE,
    RETURN_NULL,
    RETURN_NON_NULL,
    RETURN_SAME_NULLNESS,
    FAIL
}
```

Assignees
No one assigned

Labels
design · new-feature · nullness

Type
No type

Projects
No projects

Milestone
No milestone

Relationships
None yet

Development
No branches or pull requests

Notifications Customize
Subscribe
You're not receiving notifications from this thread.

Participants
+5

Give feedback

https://github.com/jspecify/jspecify/issues/757

А если мы лучше знаем…

Тотальная табличка

```
public final class TotalEnumMap<E extends Enum<E>, V> {
    private final EnumMap<E, V> map;

    ...
}
```

А если мы лучше знаем…

Тотальная табличка

```
public final class TotalEnumMap<E extends Enum<E>, V> {
    private final EnumMap<E, V> map;

    ...
}
```

А если мы лучше знаем…

Тотальная табличка

```
public TotalEnumMap(
    Class<E> enumType,
    Function<? super E, ? extends V> mapping
) {
    var map = new EnumMap<E, V>(enumType);
    for (var key : enumType.getEnumConstants()) {
        map.put(key, mapping.apply(key));
    }
    this.map = map;
}
```

А если мы лучше знаем…

Тотальная табличка

```
public TotalEnumMap(
    Class<E> enumType,
    Function<? super E, ? extends V> mapping
) {
    var map = new EnumMap<E, V>(enumType);
    for (var key : enumType.getEnumConstants()) {
        map.put(key, mapping.apply(key));
    }
    this.map = map;
}
```

А если мы лучше знаем…

Тотальная табличка

```
public V get(E key) {
    return map.get(key);
}
```

А если мы лучше знаем…

Damn you, local reasoning

```
public V get(E key) {
    return map.get(key);
//         ^
// error: [NullAway] returning @Nullable expression
//  from method with @NonNull return type
}
```

А если мы лучше знаем…

assert

```
public V get(E key) {
    var value = map.get(key);
    assert value != null : "Guaranteed by constructor";
    return value;
}
```

А если мы лучше знаем…

assert

```
public V get(E key) {
    var value = map.get(key);
    assert value != null : "Guaranteed by constructor";
    return value;
}

// build.gradle
nullaway {
    assertsEnabled = true
}
```

А если мы лучше знаем…

SuppressWarnings

```
public final class TotalEnumMap
      <E extends Enum<E>, V extends @Nullable Object> {
    ...

    @SuppressWarnings("NullAway")
    // Guaranteed by constructor
    public V get(E key) {
        return map.get(key);
    }
}
```

04 Интероп

Spring

- Spring Boot 4.0
- Spring Framework 7.0
- Spring Data 4.0, Spring Security 7.0, …
- Micrometer 1.16, Micrometer Tracing 1.6
- Context Propagation 1.2
- Reactor 2025.0

https://spring.io/blog/2025/11/12/null-safe-applications-with-spring-boot-4

JUnit

6.0.0

```
@Contract("null, _ -> fail")
public static void assertNotNull(
    @Nullable Object actual,
    Supplier<@Nullable String> messageSupplier
) {
    AssertNotNull.assertNotNull(actual, messageSupplier);
}
```

https://docs.junit.org/6.0.0/release-notes.html

JUnit

Было бы правильнее

```
@Contract("null, _ -> fail")
public static void assertNotNull(
    @Nullable Object actual,
    Supplier<? extends @Nullable String> messageSupplier
) {
    AssertNotNull.assertNotNull(actual, messageSupplier);
}
```

https://docs.junit.org/6.0.0/release-notes.html

PECS

Producer-Extends Consumer-Super

```
void use(Supplier<@Nullable String> sup) { ... }
```

PECS

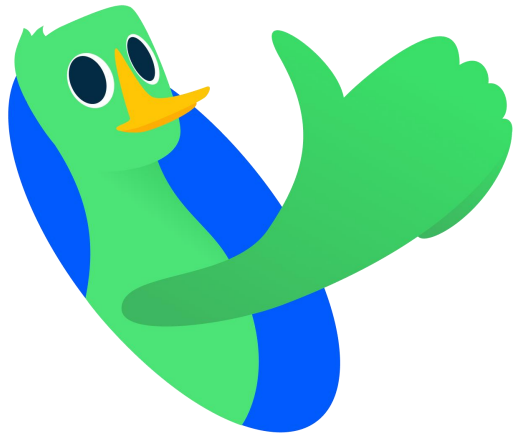
Producer-Extends Consumer-Super

```
void use(Supplier<@Nullable String> sup) { ... }

Supplier<@Nullable String> supOfNullable() { ... }
```

PECS

Producer-Extends Consumer-Super

```
void use(Supplier<@Nullable String> sup) { ... }

Supplier<@Nullable String> supOfNullable() { ... }

void main() {
    use(supOfNullable());
}
```



PECS

Producer-Extends Consumer-Super

```
void use(Supplier<@Nullable String> sup) { ... }

Supplier<String> supOfNonNull() { ... }

void main() {
    use(supOfNonNull());
}
```

PECS

Producer-Extends Consumer-Super

```
void use(Supplier<@Nullable String> sup) { ... }

Supplier<String> supOfNonNull() { ... }

void main() {
    use(supOfNonNull());
//                  ^
// error: [NullAway] incompatible types:
//  Supplier<String> cannot be converted
//  to Supplier<@Nullable String>
}
```

PECS

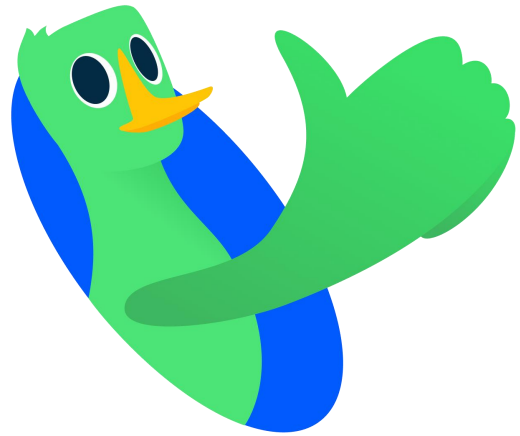
Producer-Extends Consumer-Super

```
void use(Supplier<? extends @Nullable String> sup) { ... }

Supplier<String> supOfNonNull() { ... }

void main() {
    use(supOfNonNull());
}
```

PECS

Producer-Extends Consumer-Super

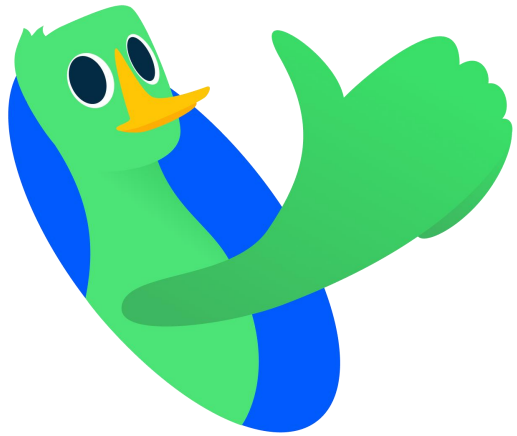
```
void use(Supplier<? extends @Nullable String> sup) { ... }

Supplier<String> supOfNonNull() { ... }

void main() {
    use(supOfNonNull());
}
```



PECS

Producer-Extends Consumer-Super

```
void use(Supplier<? extends @Nullable String> sup) { ... }

Supplier<@Nullable String> supOfNullable() { ... }

void main() {
    use(supOfNullable());
}
```



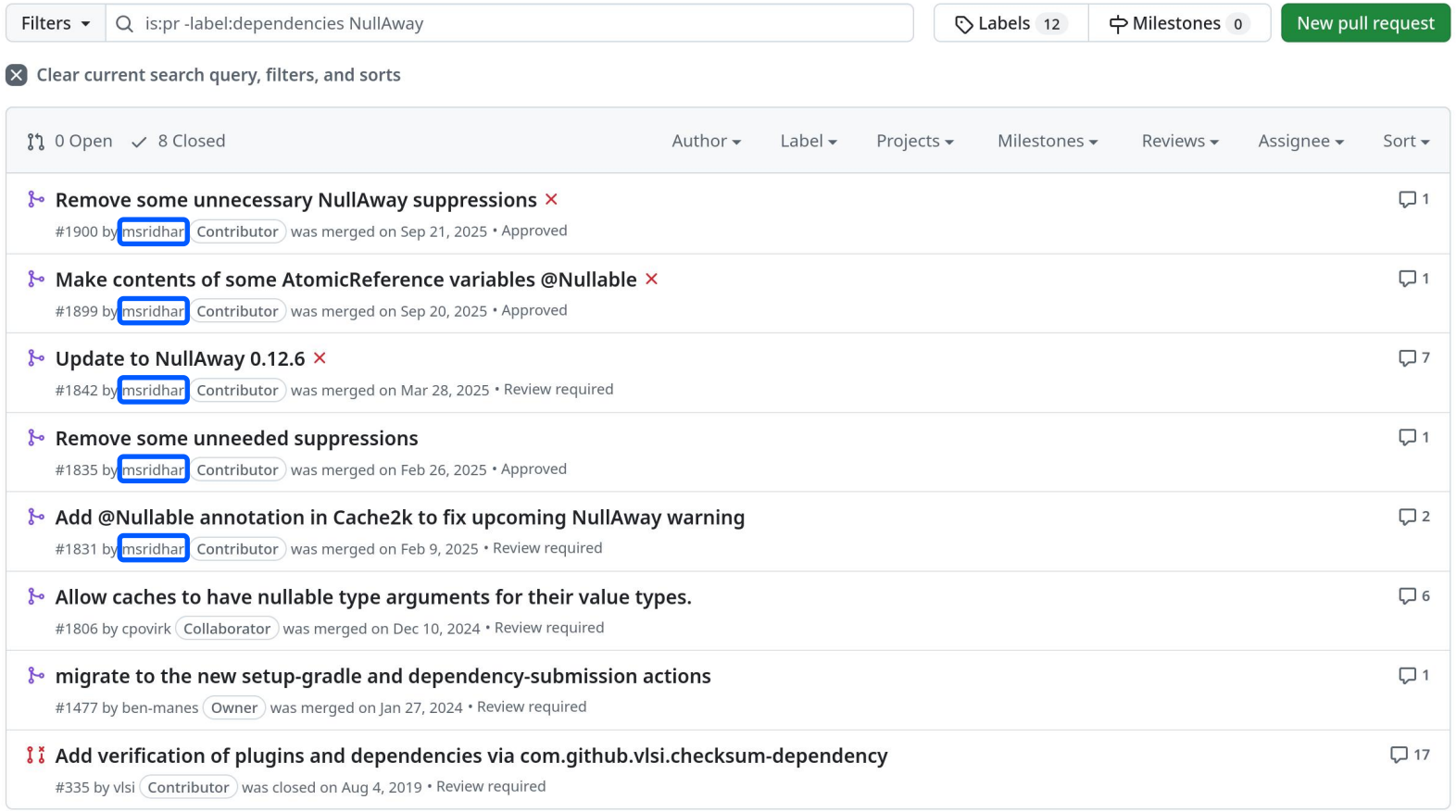
Guava

33.4.1

```
$ cd guava
$ rg '@NullMarked' | wc --lines
    510
```

https://github.com/google/guava/releases/tag/v33.4.1

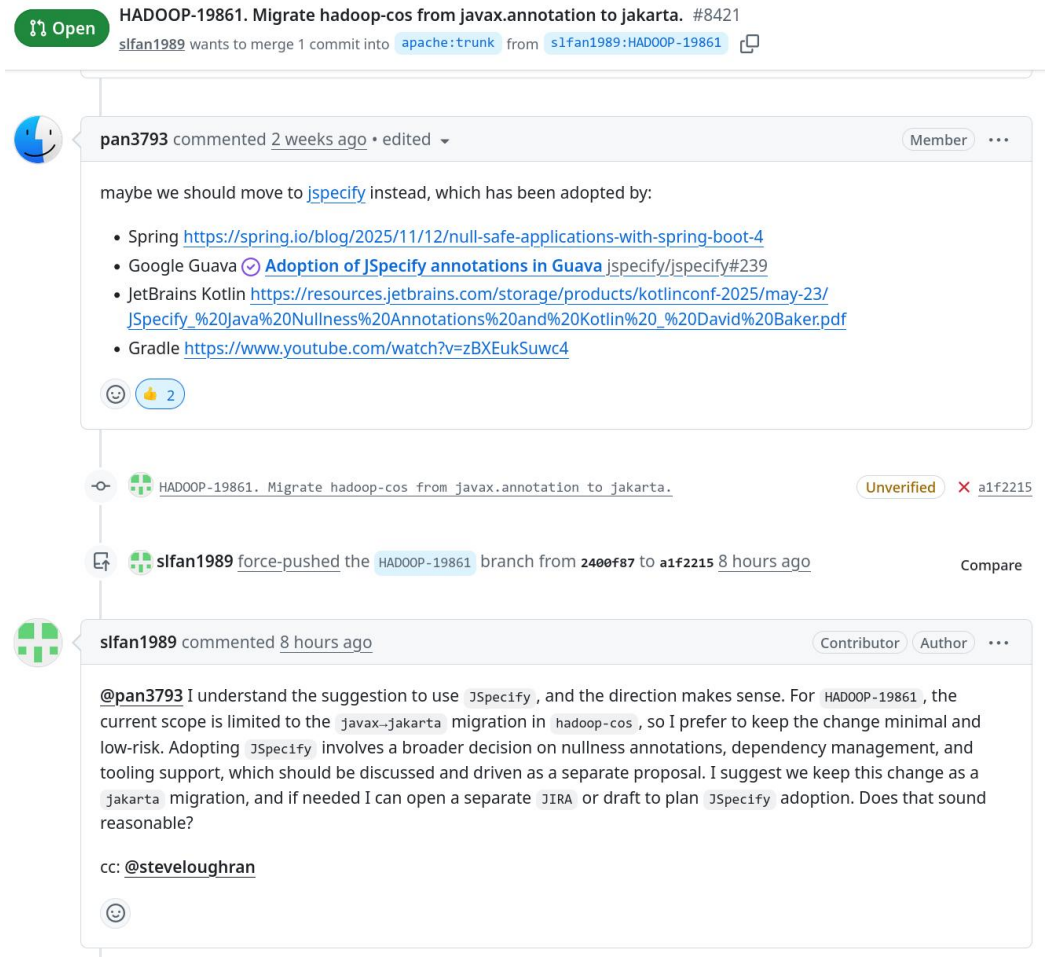
Caffeine



https://github.com/ben-manes/caffeine/pulls?q=is%3Apr+-label%3Adependencies+NullAway

Кто-то пока в процессе

Hadoop



Open **HADOOP-19861. Migrate hadoop-cos from javax.annotation to jakarta.** #8421

slfan1989 wants to merge 1 commit into `apache:trunk` from `slfan1989:HADOOP-19861`

pan3793 commented 2 weeks ago • edited — Member

maybe we should move to jspecify instead, which has been adopted by:

- Spring https://spring.io/blog/2025/11/12/null-safe-applications-with-spring-boot-4
- Google Guava **Adoption of JSpecify annotations in Guava** jspecify/jspecify#239
- JetBrains Kotlin https://resources.jetbrains.com/storage/products/kotlinconf-2025/may-23/JSpecify_%20Java%20Nullness%20Annotations%20and%20Kotlin%20_%20David%20Baker.pdf
- Gradle https://www.youtube.com/watch?v=zBXEukSuwc4

2

`HADOOP-19861. Migrate hadoop-cos from javax.annotation to jakarta.` Unverified a1f2215

slfan1989 force-pushed the `HADOOP-19861` branch from `2400f87` to `a1f2215` 8 hours ago — Compare

slfan1989 commented 8 hours ago — Contributor, Author

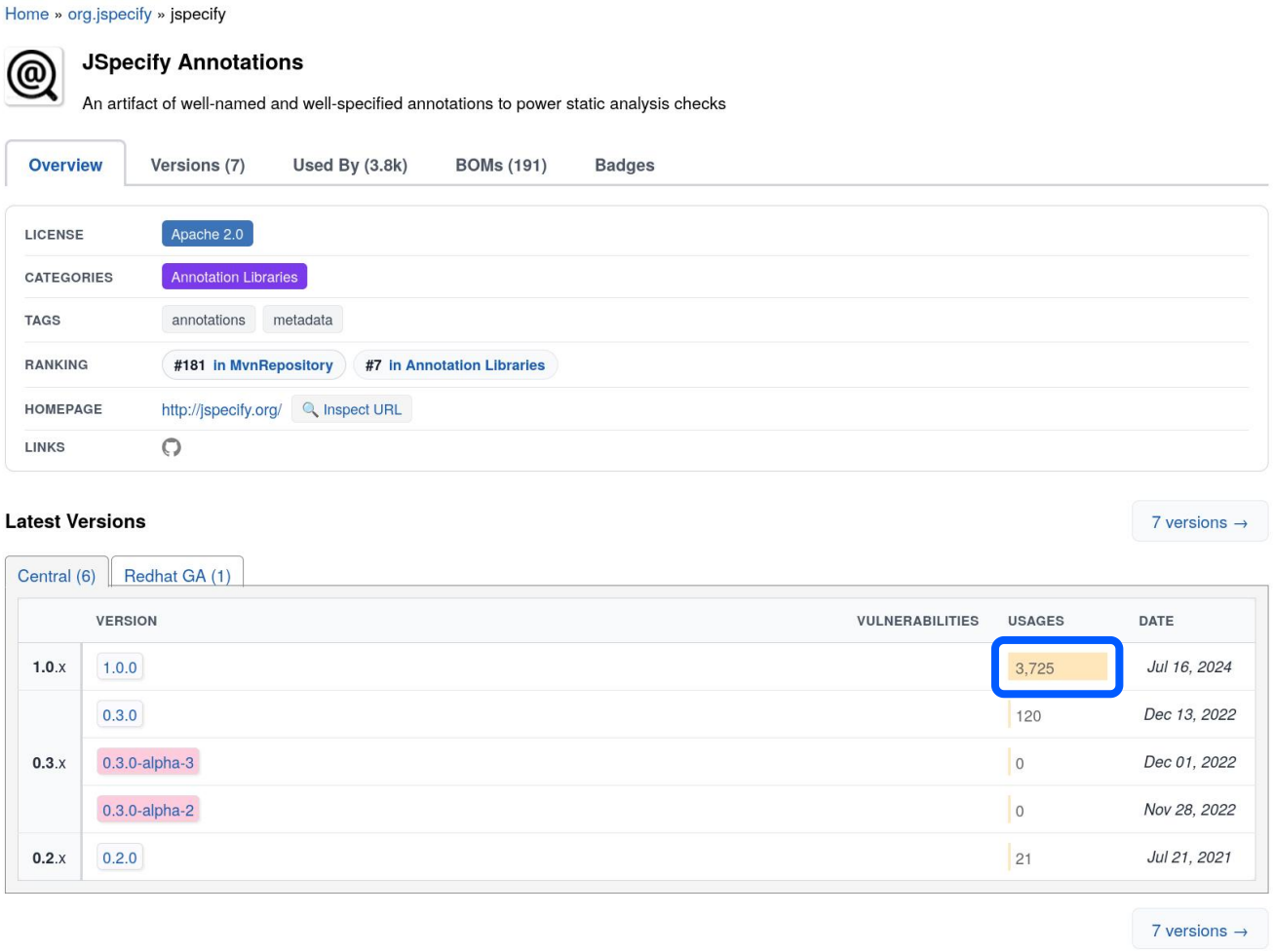
@pan3793 I understand the suggestion to use `JSpecify`, and the direction makes sense. For `HADOOP-19861`, the current scope is limited to the `javax→jakarta` migration in `hadoop-cos`, so I prefer to keep the change minimal and low-risk. Adopting `JSpecify` involves a broader decision on nullness annotations, dependency management, and tooling support, which should be discussed and driven as a separate proposal. I suggest we keep this change as a `jakarta` migration, and if needed I can open a separate `JIRA` or draft to plan `JSpecify` adoption. Does that sound reasonable?

cc: @steveloughran

https://github.com/apache/hadoop/pull/8421

mvnrepository

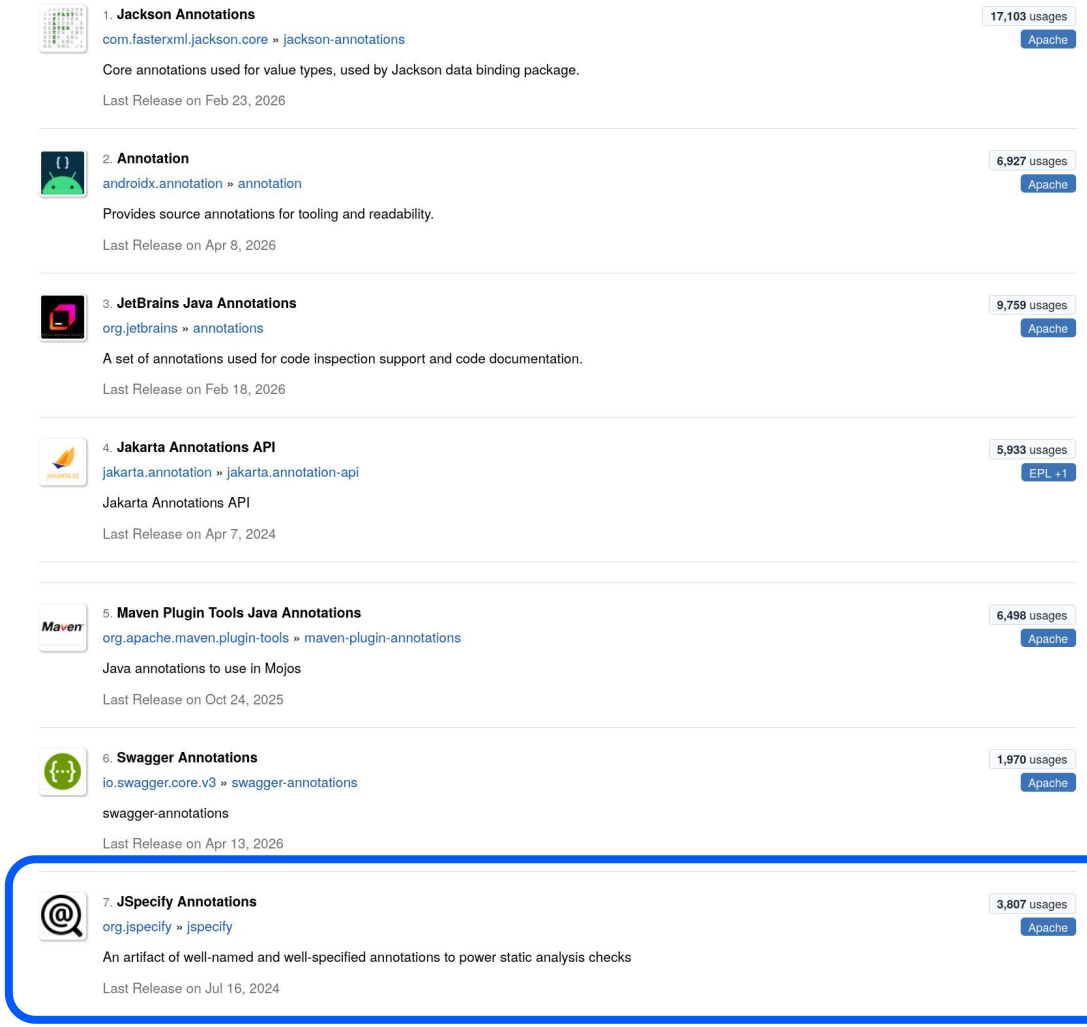
Почти 4000 зависимых



https://mvnrepository.com/artifact/org.jspecify/jspecify

mvnrepository

Топ 7 среди аннотаций



1. **Jackson Annotations** — 17,103 usages
com.fasterxml.jackson.core » jackson-annotations — Apache
Core annotations used for value types, used by Jackson data binding package.
Last Release on Feb 23, 2026

2. **Annotation** — 6,927 usages
androidx.annotation » annotation — Apache
Provides source annotations for tooling and readability.
Last Release on Apr 8, 2026

3. **JetBrains Java Annotations** — 9,759 usages
org.jetbrains » annotations — Apache
A set of annotations used for code inspection support and code documentation.
Last Release on Feb 18, 2026

4. **Jakarta Annotations API** — 5,933 usages
jakarta.annotation » jakarta.annotation-api — EPL +1
Jakarta Annotations API
Last Release on Apr 7, 2024

5. **Maven Plugin Tools Java Annotations** — 6,498 usages
org.apache.maven.plugin-tools » maven-plugin-annotations — Apache
Java annotations to use in Mojos
Last Release on Oct 24, 2025

6. **Swagger Annotations** — 1,970 usages
io.swagger.core.v3 » swagger-annotations — Apache
swagger-annotations
Last Release on Apr 13, 2026

7. **JSpecify Annotations** — 3,807 usages
org.jspecify » jspecify — Apache
An artifact of well-named and well-specified annotations to power static analysis checks
Last Release on Jul 16, 2024

https://mvnrepository.com/open-source/annotation-libraries

IntelliJ IDEA

Уже давно понимает семантику `@NullMarked`

IntelliJ IDEA

И более сложные сценарии

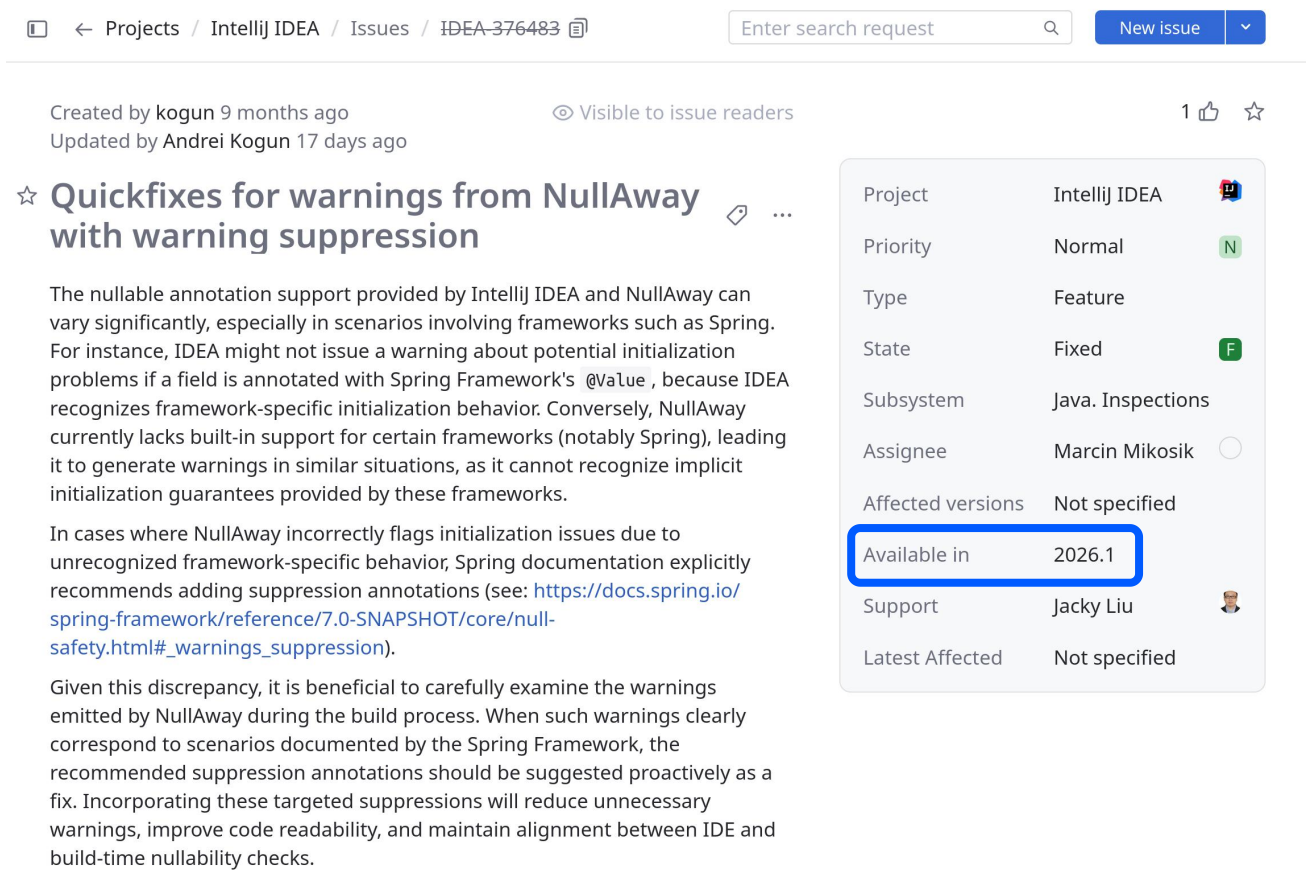
```
var surelyBytes : byte[] = tryMap(value(), String::getBytes);
if (surelyBytes == null) throw new IllegalArgumentException("value was null");
System.out.println(surelyBytes.length);
var maybeBytes : byte[] = tryMap(maybeValue(), String::getBytes);
System.out.println(maybeBytes.length);
```

Condition 'surelyBytes == null' is always 'false'

Remove 'if' statement Alt+Shift+Enter More actions... Alt+Enter

IntelliJ IDEA

Умеет предлагать Nullaway-специфичные suppresison'ы и отдаёт приоритет JSpecify



https://youtrack.jetbrains.com/issue/IDEA-376483

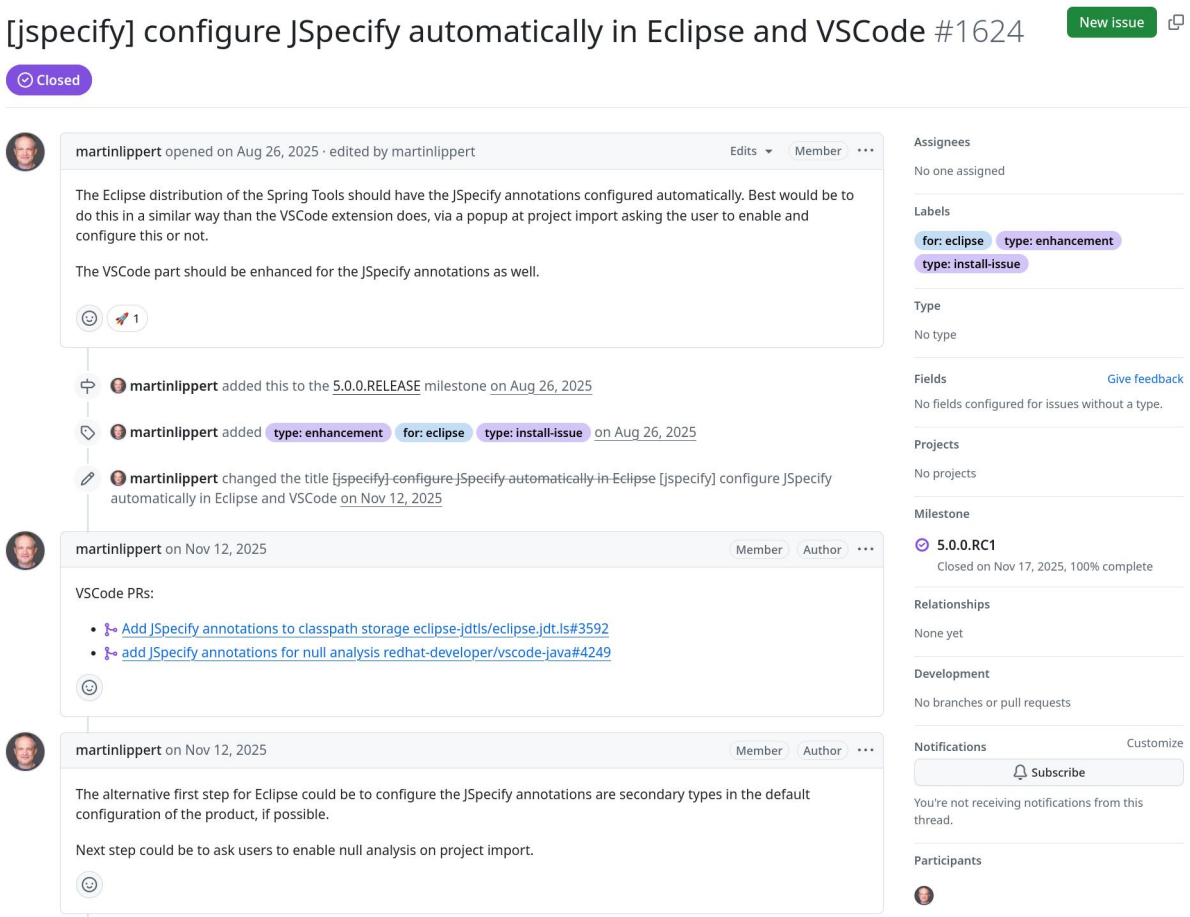
IntelliJ IDEA

Умеет предлагать Nullaway-специфичные suppresison’ы и отдаёт приоритет JSpecify

https://youtrack.jetbrains.com/issue/IDEA-376483

Eclipse / VSC

Поддерживаются благодаря плагину spring-tools



[jspecify] configure JSpecify automatically in Eclipse and VSCode #1624

New issue

Closed

martinlippert opened on Aug 26, 2025 · edited by martinlippert — Edits — Member

The Eclipse distribution of the Spring Tools should have the JSpecify annotations configured automatically. Best would be to do this in a similar way than the VSCode extension does, via a popup at project import asking the user to enable and configure this or not.

The VSCode part should be enhanced for the JSpecify annotations as well.

🚀 1

martinlippert added this to the 5.0.0.RELEASE milestone on Aug 26, 2025

martinlippert added type: enhancement for: eclipse type: install-issue on Aug 26, 2025

martinlippert changed the title ~~[jspecify] configure JSpecify automatically in Eclipse~~ [jspecify] configure JSpecify automatically in Eclipse and VSCode on Nov 12, 2025

martinlippert on Nov 12, 2025 — Member — Author

VSCode PRs:

- Add JSpecify annotations to classpath storage eclipse-jdtls/eclipse.jdt.ls#3592
- add JSpecify annotations for null analysis redhat-developer/vscode-java#4249

martinlippert on Nov 12, 2025 — Member — Author

The alternative first step for Eclipse could be to configure the JSpecify annotations are secondary types in the default configuration of the product, if possible.

Next step could be to ask users to enable null analysis on project import.

Assignees: No one assigned

Labels: for: eclipse, type: enhancement, type: install-issue

Type: No type

Fields: Give feedback — No fields configured for issues without a type.

Projects: No projects

Milestone: 5.0.0.RC1 — Closed on Nov 17, 2025, 100% complete

Relationships: None yet

Development: No branches or pull requests

Notifications: Customize — Subscribe — You're not receiving notifications from this thread.

Participants

https://github.com/spring-projects/spring-tools/issues/1624

Рантайм-проверки

С помощью cabe

```
// build.gradle
plugins {
    id 'com.dua3.cabe' version '3.3.0'
}
```

https://github.com/xzel23/cabe

Рантайм-проверки

С помощью cabe

```
public void allowsNull(@Nullable String input) {}
public void prohibitsNull(String input) {}

void main() {
    String value = null;
    allowsNull(value);
    prohibitsNull(value);
}
```

https://github.com/xzel23/cabe

Рантайм-проверки

С помощью cabe

```
public void allowsNull(@Nullable String input) {}
public void prohibitsNull(String input) {}

void main() {
    String value = null;
    allowsNull(value);
    prohibitsNull(value);
}
```

https://github.com/xzel23/cabe

Рантайм-проверки

С помощью cabe

```
Exception in thread "main"
java.lang.NullPointerException: input is null
        prohibitsNull(CabeExample.java)
        main(CabeExample.java:7)
```

https://github.com/xzel23/cabe

Рантайм-проверки

Работают через модификацию байткода

```
0: aload_1
1: aconst_null
2: if_acmpne 15
5: new #30 // class NullPointerException
8: dup
9: ldc #32 // String input is null
11: invokespecial #34 // NullPointerException."<init>"
14: athrow
```

https://github.com/xzel23/cabe

Рантайм-проверки

Работают через модификацию байткода

```
public void prohibitsNull(String input) {
    if (input == null) {
        throw new NullPointerException("input is null");
    }
}
```

https://github.com/xzel23/cabe

Kotlin

Может полагаться на разметку Nullability

```
public interface Parameterized<T extends @Nullable Object> {
    // Parameterized<@NonNull String>: returns String
    // Parameterized<@Nullable String>: returns String?
    T same();

    // returns T?
    @Nullable T alwaysNullable();

    // returns T & Any
    @NonNull T alwaysNonNull();
}
```

https://youtu.be/z_IO2_XeW2Q

Kotlin

Может полагаться на разметку Nullability

```
public interface Parameterized<T extends @Nullable Object> {
    // Parameterized<@NonNull String>: returns String
    // Parameterized<@Nullable String>: returns String?
    T same();

    // returns T?
    @Nullable T alwaysNullable();

    // returns T & Any
    @NonNull T alwaysNonNull();
}
```

https://youtu.be/z_IO2_XeW2Q

Kotlin

Может полагаться на разметку Nullability

```
public interface Parameterized<T extends @Nullable Object> {
    // Parameterized<@NonNull String>: returns String
    // Parameterized<@Nullable String>: returns String?
    T same();

    // returns T?
    @Nullable T alwaysNullable();

    // returns T & Any
    @NonNull T alwaysNonNull();
}
```

https://youtu.be/z_IO2_XeW2Q

Kotlin

Может полагаться на разметку Nullability

```
public interface Parameterized<T extends @Nullable Object> {
    // Parameterized<@NonNull String>: returns String
    // Parameterized<@Nullable String>: returns String?
    T same();

    // returns T?
    @Nullable T alwaysNullable();

    // returns T & Any
    @NonNull T alwaysNonNull();
}
```

https://youtu.be/z_IO2_XeW2Q

Kotlin

Может полагаться на разметку Nullability

```
val p1: Parameterized<String> = createParametrized()
val p2: Parameterized<String?> = createParametrized()

p1.
```

```
alwaysNonNull()          String
alwaysNullable()         String?
same()                   String
```

https://youtu.be/z_IO2_XeW2Q

Kotlin

Может полагаться на разметку Nullability

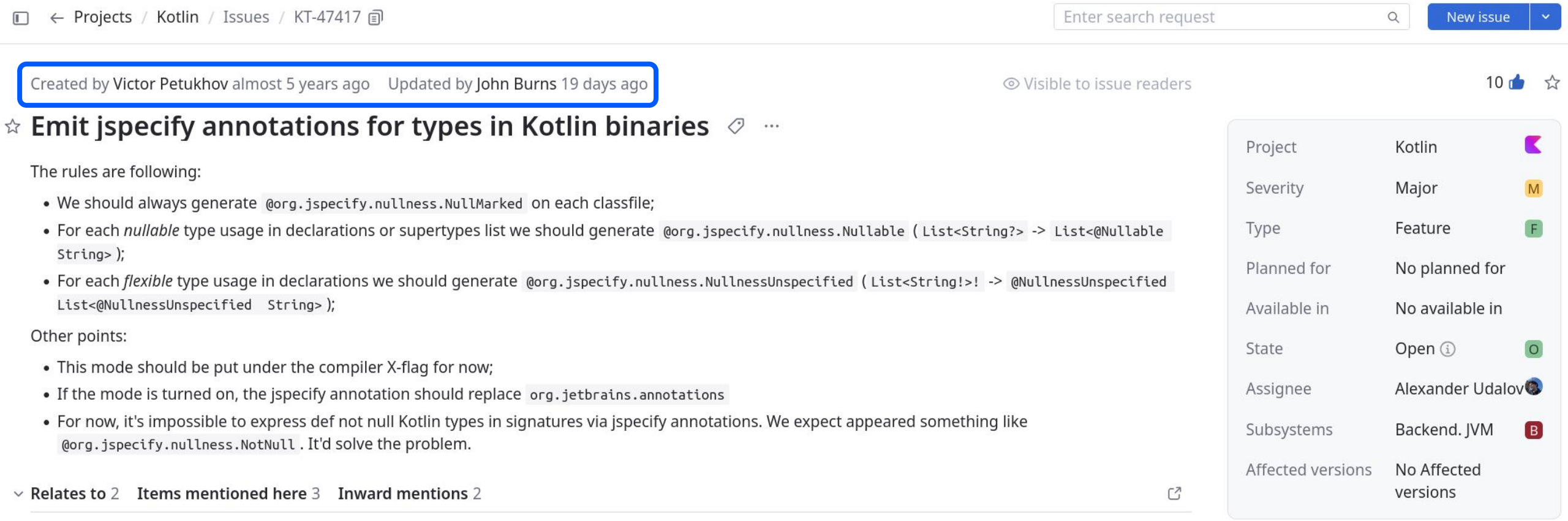
```
val p1: Parameterized<String> = createParametrized()
val p2: Parameterized<String?> = createParametrized()

p2.
```

```
alwaysNonNull()          String
alwaysNullable()         String?
same()                   String?
```

https://youtu.be/z_IO2_XeW2Q

Kotlin

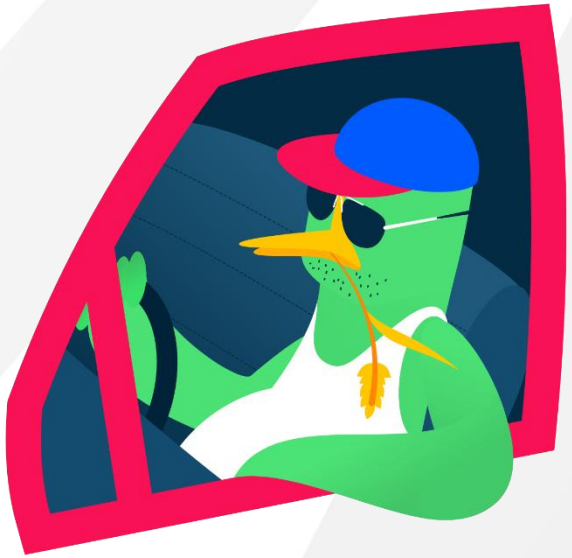
Может начать использовать JSpecify для разметки nullability генерируемого байткода



https://youtrack.jetbrains.com/issue/KT-47417

Не бойтесь использовать

JSpecify и Nullaway.
Экосистема созрела

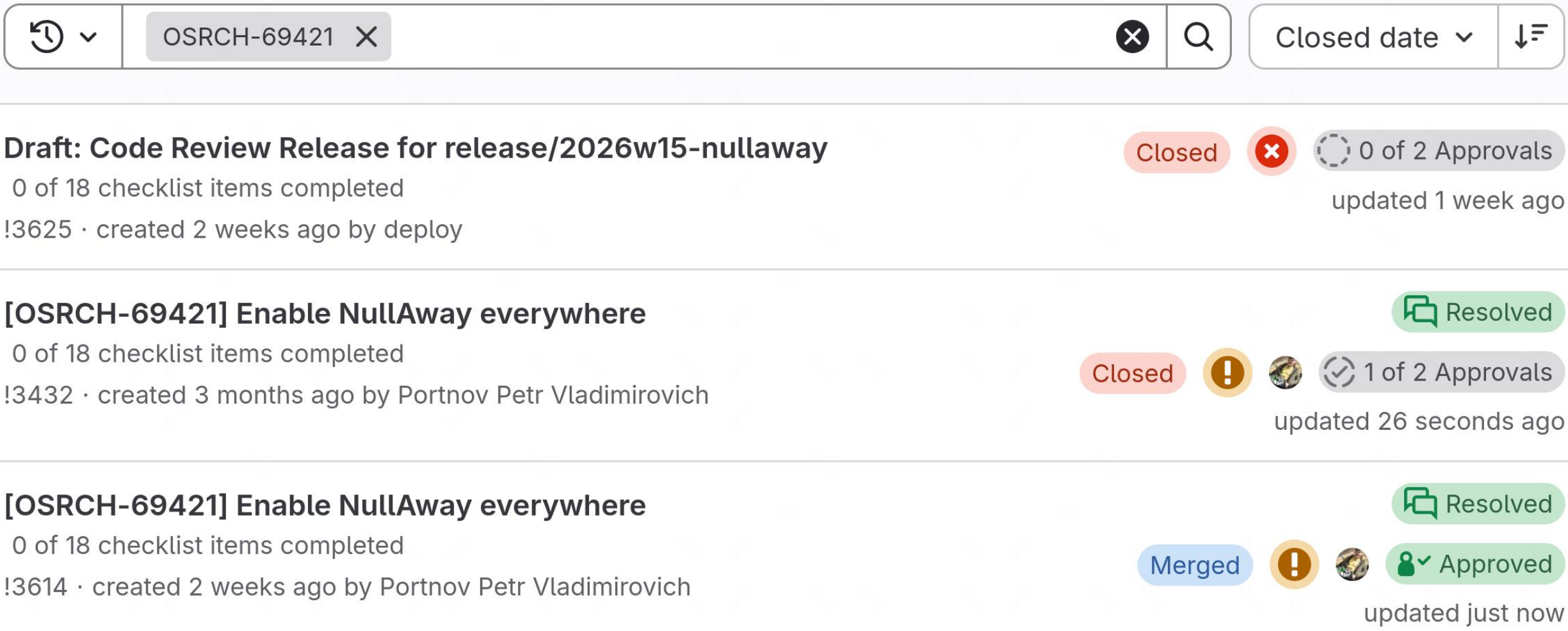


Все новые проекты начинаем с JSpecify

Наш опыт

Старый код лениво размечаем

Наш опыт


OSRCH-69421
Closed date
Draft: Code Review Release for release/2026w15-nullaway
0 of 18 checklist items completed
!3625 · created 2 weeks ago by deploy
Closed
0 of 2 Approvals
updated 1 week ago
[OSRCH-69421] Enable NullAway everywhere
0 of 18 checklist items completed
!3432 · created 3 months ago by Portnov Petr Vladimirovich
Resolved
Closed
1 of 2 Approvals
updated 26 seconds ago
[OSRCH-69421] Enable NullAway everywhere
0 of 18 checklist items completed
!3614 · created 2 weeks ago by Portnov Petr Vladimirovich
Resolved
Merged
Approved
updated just now

Наш опыт

Наш опыт

Даже на больших кодовых базах всё стабильно

```
$ rg '"NullAway"'
    // Сложный инвариант
    @SuppressWarnings("NullAway")

    // Разметка в зависимости
    @SuppressWarnings("NullAway")
    @SuppressWarnings("NullAway")

    // SnapshotExtension
    @SuppressWarnings({"unused", "NullAway"})
    @SuppressWarnings({"unused", "NullAway"})
```

Наш опыт

Даже на больших кодовых базах всё стабильно

```
$ rg '"NullAway"'
    // Сложный инвариант
    @SuppressWarnings("NullAway")

    // Разметка в зависимости
    @SuppressWarnings("NullAway")
    @SuppressWarnings("NullAway")

    // SnapshotExtension
    @SuppressWarnings({"unused", "NullAway"})
    @SuppressWarnings({"unused", "NullAway"})
```

Наш опыт

Даже на больших кодовых базах всё стабильно

```
$ rg '"NullAway"'
    // Сложный инвариант
    @SuppressWarnings("NullAway")

    // Разметка в зависимости
    @SuppressWarnings("NullAway")
    @SuppressWarnings("NullAway")

    // SnapshotExtension
    @SuppressWarnings({"unused", "NullAway"})
    @SuppressWarnings({"unused", "NullAway"})
```

Наш опыт

Даже на больших кодовых базах всё стабильно

```
$ rg '"NullAway"'
    // Сложный инвариант
    @SuppressWarnings("NullAway")

    // Разметка в зависимости
    @SuppressWarnings("NullAway")
    @SuppressWarnings("NullAway")

    // SnapshotExtension
    @SuppressWarnings({"unused", "NullAway"})
    @SuppressWarnings({"unused", "NullAway"})
```

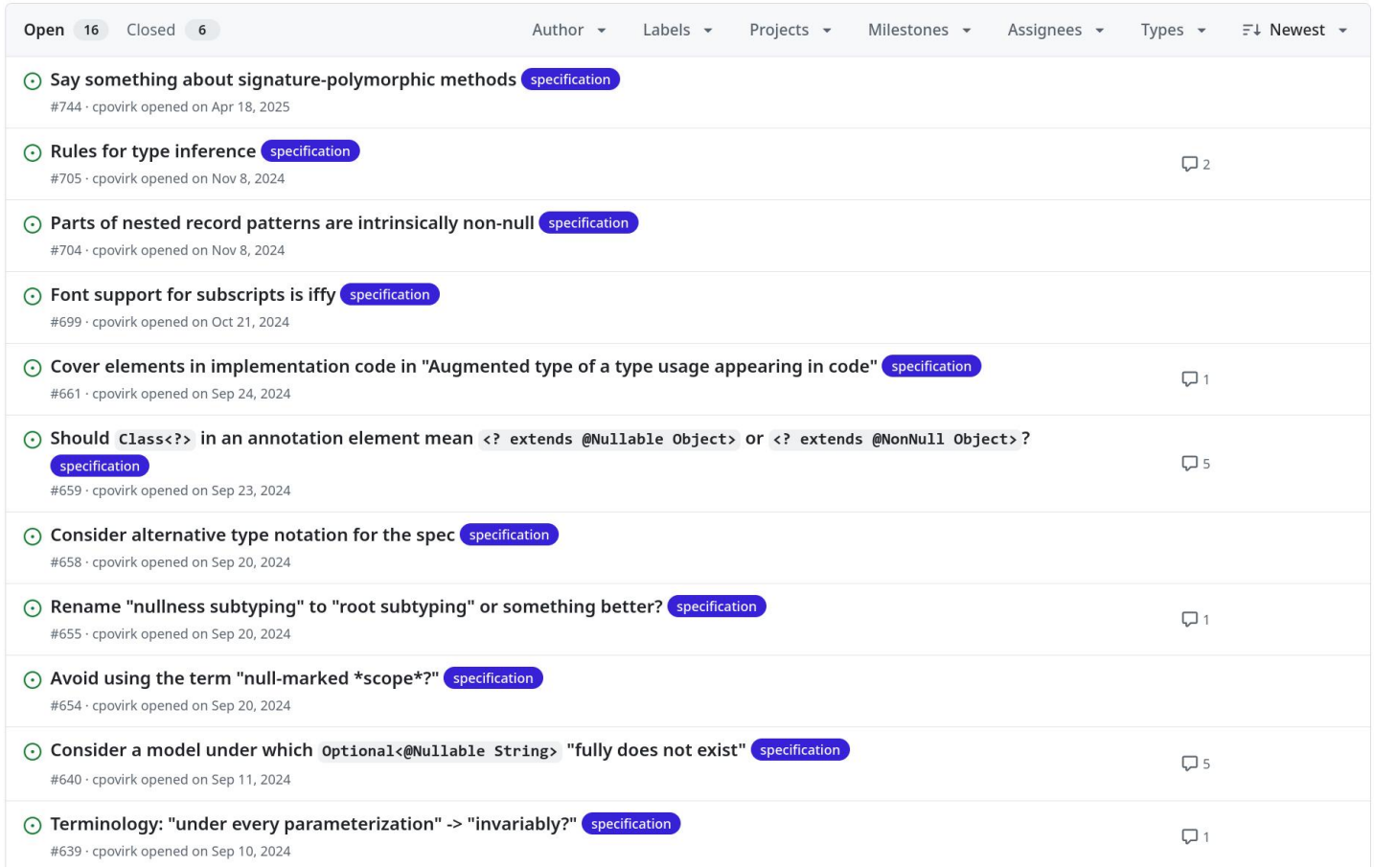
Наш опыт

Не встречали NPE в размеченных проектах

05 Проблемы

Спека

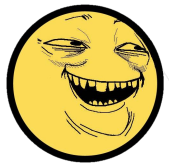
(пока не лишена пробелов)



https://github.com/jspecify/jspecify/issues?q=is%3Aissue%20state%3Aopen%20label%3Aspecification

Спека

Кроме этого



Font support for subscripts is iffy #699

Open

cpovirk opened on Oct 21, 2024 — Collaborator

The subscripts in https://jspecify.dev/docs/spec/#substitution render fine on my desktop, but they're squares (though I think maybe *subscript* squares? I'd have to check) on my phone. Maybe we can force another font. Or we can just not worry about it. I'm filing this largely to remind myself that I've seen this only on a phone.

cpovirk added specification on Oct 21, 2024

https://github.com/jspecify/jspecify/issues?q=is%3Aissue%20state%3Aopen%20label%3Aspecification

Но текущее состояние покрывает бóльшую часть потребностей

Спека

Для ad-hoc потребностей есть тул-специфичные утилиты

NullAway / annotations / src / main / java / com / uber / nullaway / **annotations** /

Add file ···

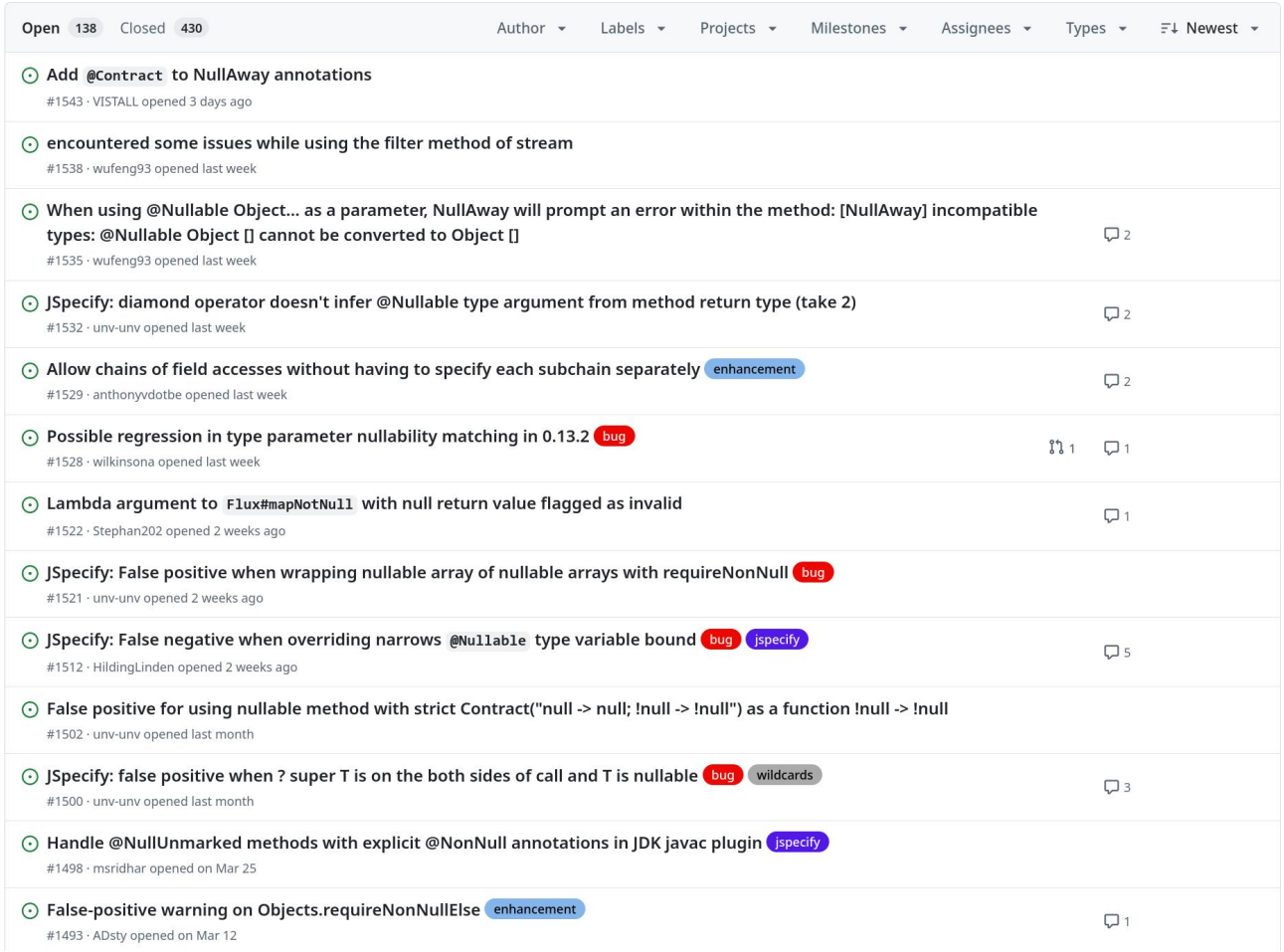
FxMorin and msridhar Add `PureExceptLambda` annotation (#1325) ··· ✓ 39d65f8 · 5 months ago History

Name	Last commit message	Last commit date
..		
EnsuresNonNull.java	Expose @EnsuresNonNull and @RequiresNonNull in annotations package (#999	2 years ago
EnsuresNonNullIf.java	Support @EnsuresNonNullIf (#1044)	2 years ago
Initializer.java	Add an initial annotations artifact (with sample **@Initializer** impleme...	4 years ago
MonotonicNonNull.java	Better `@MonotonicNonNull` support (#1149)	last year
PureExceptLambda.java	Add `PureExceptLambda` annotation (#1325)	5 months ago
RequiresNonNull.java	Expose @EnsuresNonNull and @RequiresNonNull in annotations package (#999	2 years ago

https://github.com/uber/NullAway/tree/master/annotations

Баги

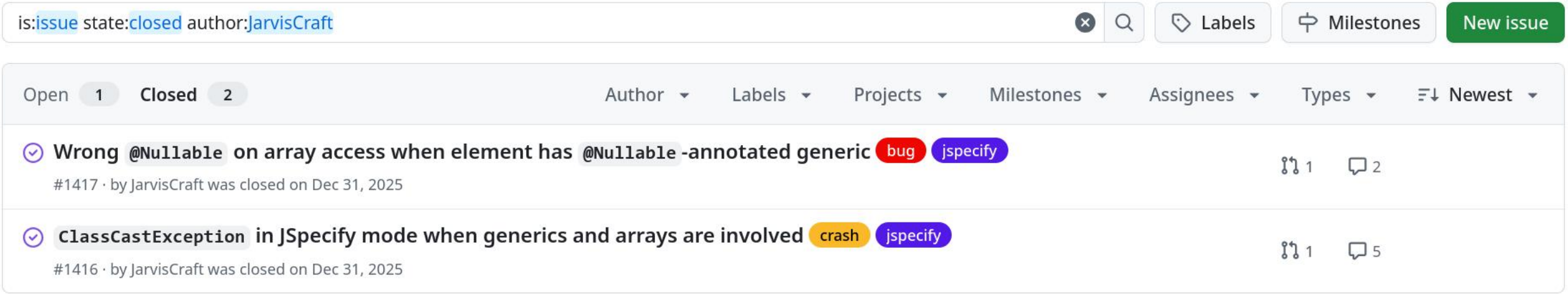
(существуют)



https://github.com/uber/NullAway/issues

Баги

Реально активно фиксятся контрибуторами



https://github.com/uber/NullAway/issues

В IDE тоже

```
<T> void exact(GenericFactory<T> factory) {
    var created : T  = factory.create();
```

Expression 'factory.create()' might evaluate to null but is assigned to a non-null variable

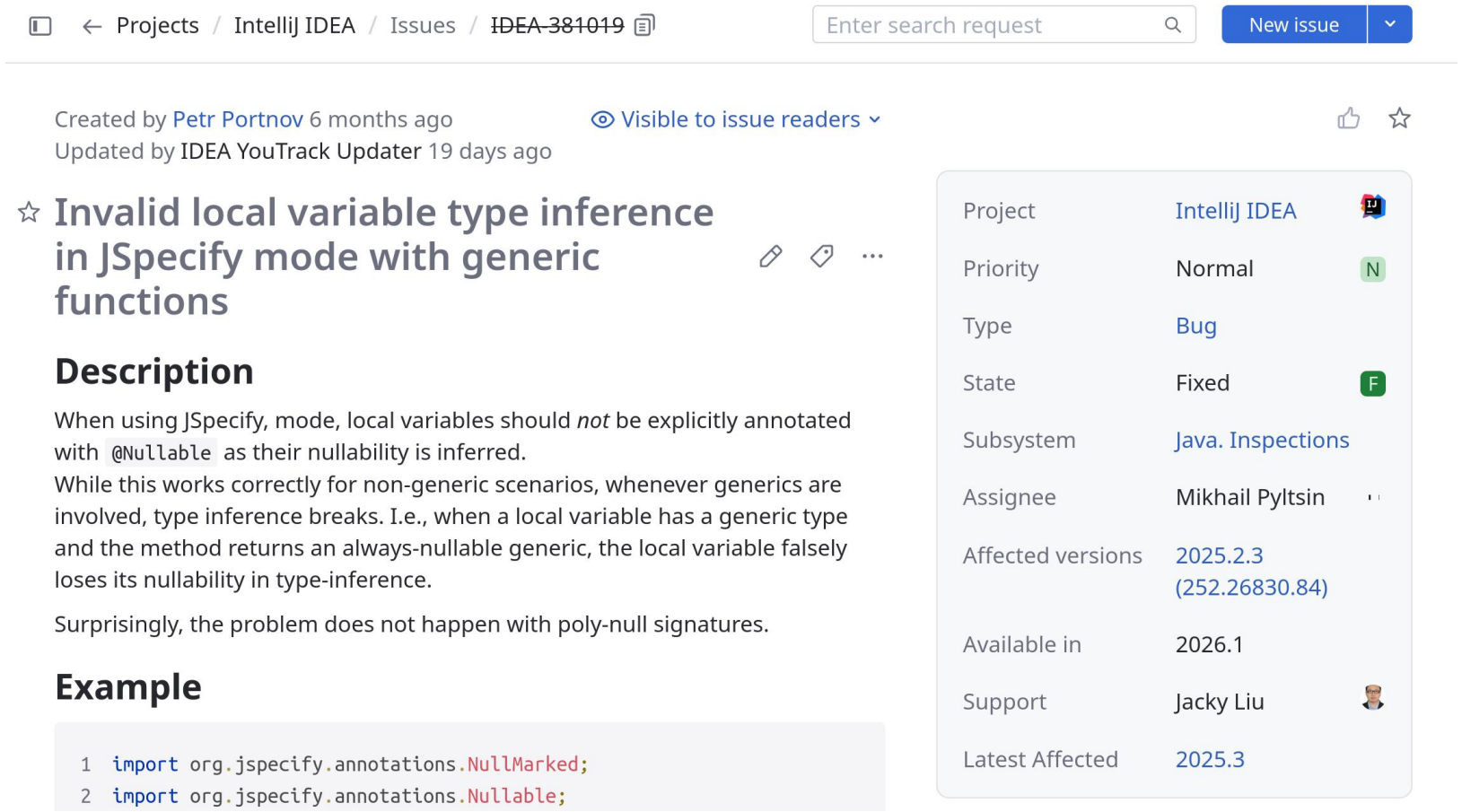
Replace with 'Objects.requireNonNull(factory.create())' Alt+Shift+Enter More actions... Alt+Enter

```
@FunctionalInterface
interface GenericFactory<T> {
    @Nullable T create();
}
```

https://youtrack.jetbrains.com/issue/IDEA-381019

В IDE тоже

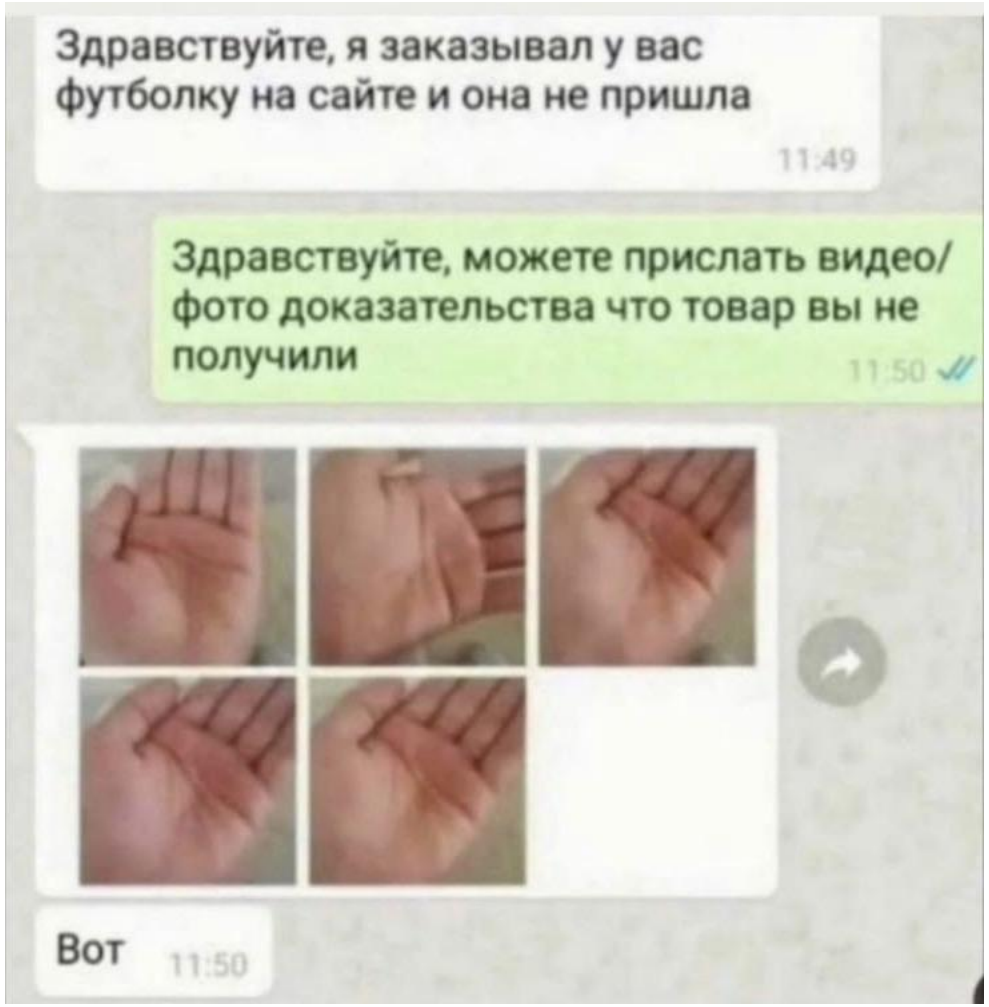
И там их тоже активно чинят



https://youtrack.jetbrains.com/issue/IDEA-381019

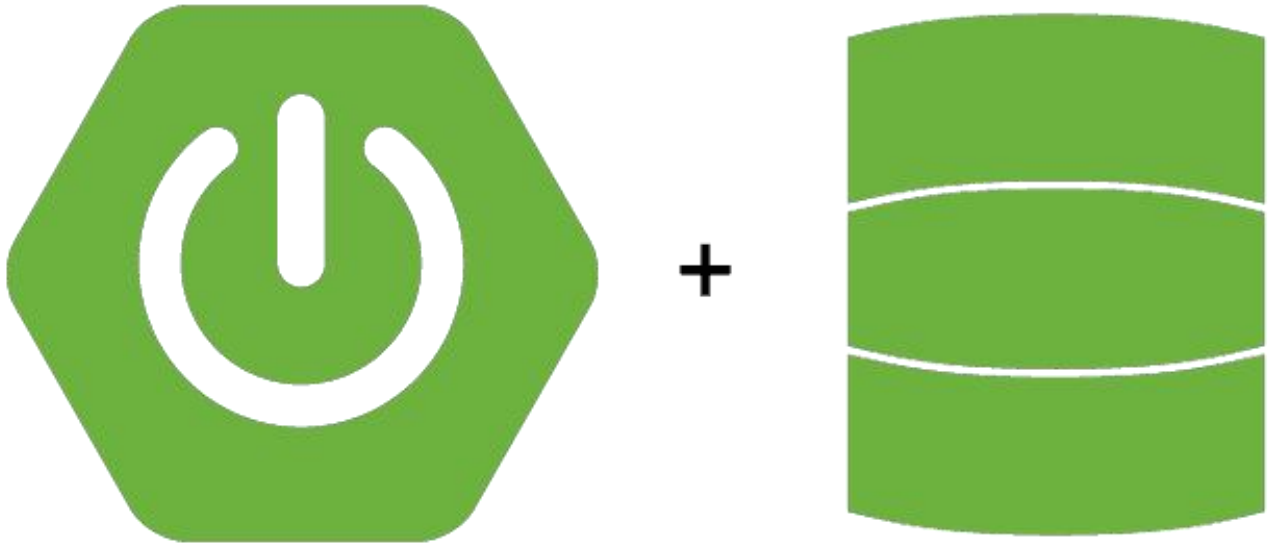
Библиотеки и фреймворки

Пока не все знают / хотят знать о JSpecify:
Hibernate ничего не знает о `@NullMarked` (и, вообще, JSpecify)



Библиотеки и фреймворки

Пока не все знают / хотят знать о JSpecify:
Spring Data знает



https://docs.spring.io/spring-data/commons/reference/repositories/null-handling.html#_jspecify

Библиотеки и фреймворки

Могут быть не размечены в апстриме
Но можно указать отдельную разметку nullability

```
dependencies {
    ...

    annotationProcessor(
        'ru.ozon.library-models:lucene-model:10.4.0'
    )
}
```

https://github.com/uber/NullAway/wiki/Configuration#library-models

NullAway

Может не знать о ваших специфических библиотеках

```
// build.gradle
nullaway {
    excludedFieldAnnotations += [
        // Mockito
        'org.mockito.Captor',
        'org.mockito.InjectMocks',
        'org.mockito.Mock',
        'org.mockito.Spy',
    ]
}
```

https://github.com/uber/NullAway/wiki/Configuration#excluded-field-annotations

Генерируемый код

Как правило, неразмечен
Но можно честно признаться в этом Nullaway

```
// build.gradle
nullaway {
    treatGeneratedAsUnannotated = true
}
```

https://github.com/uber/NullAway/wiki/Configuration#treat-generated-as-unannotated

Генерируемый код

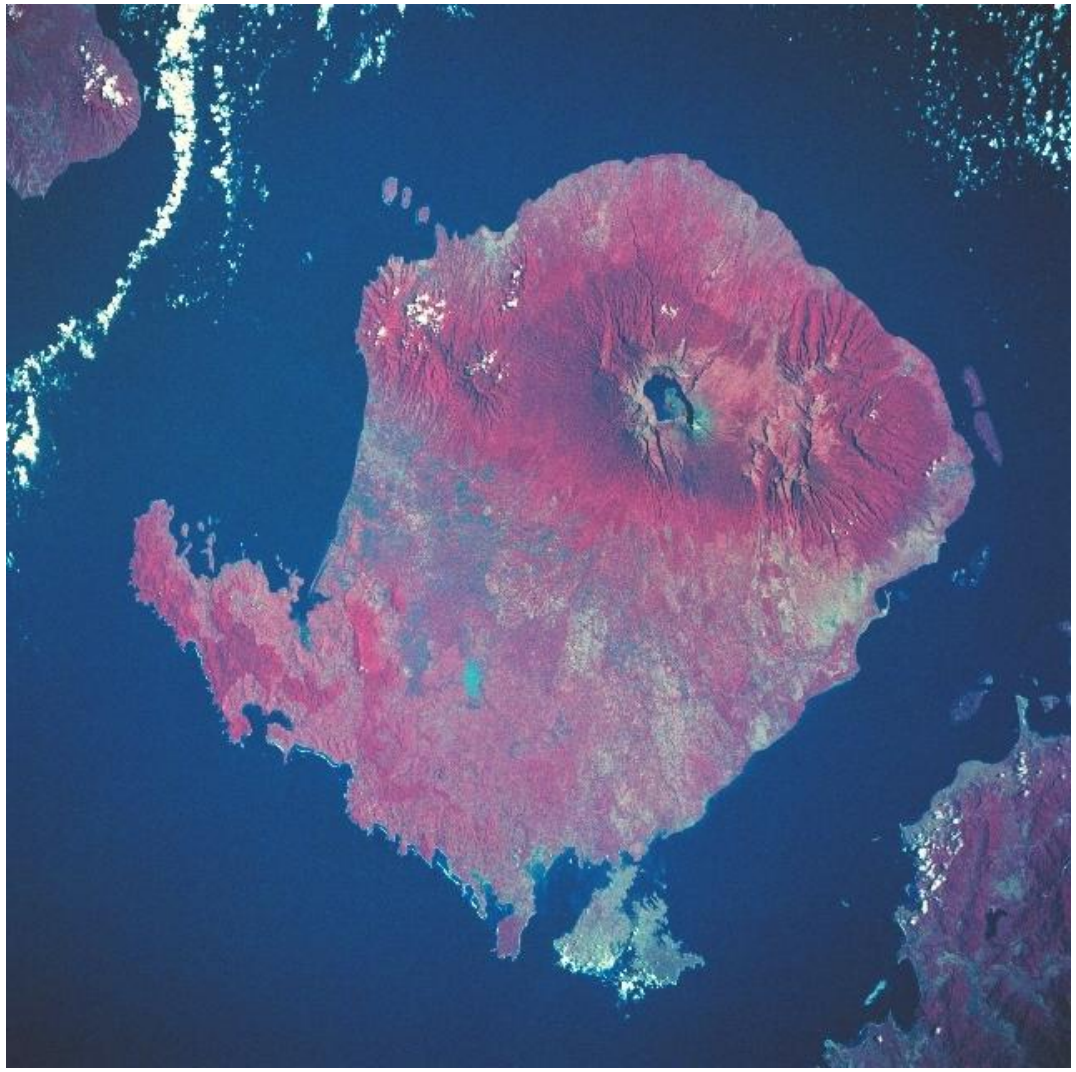
Как правило, неразмечен
Но можно честно признаться в этом Nullaway

```
@NullUnmarked // protoc-generated code.
package ru.ozon.search.o2.base.api;

@NullUnmarked // JCstress-generated code.
package ru.ozon.search.o2.base.blending.jcstress;
```

https://github.com/uber/NullAway/wiki/Configuration#excluded-field-annotations

Lombok

(кродеться)



Lombok

(кродеться)

```
public class IntegrationTestExtension {
    ...

    @Builder
    private IntegrationTestExtension(
        @Singular Map<Class<? extends Module>, Module>
            moduleOverrides,
        ...
    ) { ... }
}
```

Lombok

(кродеться)

```
    @Builder
//  ^
// error: [NullAway] initializer method
//  does not guarantee @NonNull fields
//  moduleOverrides$key (line 87),
//  moduleOverrides$value (line 87)
//  (remember to check for exceptions or early returns).
```

Lombok

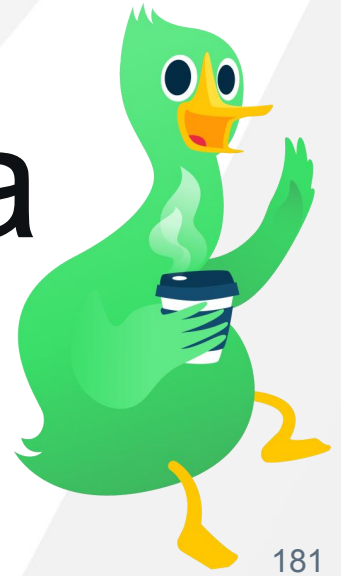
Можно исключить

Lombok

Можно локально исключить

```
public class IntegrationTestExtension {
    ...

    @NullUnmarked
    public static final class
        IntegrationTestExtensionBuilder {}
}
```

Lombok стараются правильно размечать nullability или помечают сгенерированность кода



Требует свежего тулчейна

JDK / JDK-8225377

type annotations are not visible to javac plugins across compilation boundaries

Resolved

Export

Details

Type:	Bug	Resolution:	Fixed
Priority:	P3	Fix Version/s:	22
Affects Version/s:	13, 21		
Component/s:	tools		
Labels:	javac-plugin jdk1 jdk21u-fix-SQE-ok		
Subcomponent:	javac		
Resolved In Build:	b23		

People

Assignee:	Liam Miller-Cushon
Reporter:	Liam Miller-Cushon
Votes:	1 Vote for this issue
	12 Start watching this issue

	2019-06-05 14:53
Updated:	2025-11-01 14:05
Resolved:	2023-11-03 10:32

⚠ На старых версиях требует:
`-XDaddTypeAnnotationsToSymbol=true`

Backports

Issue	Fix Version	Assignee	Priority	Status	Resolution	Resolved In Build
JDK-8323370	21.0.3-oracle	Liam Miller-Cushon	P3	Resolved	Fixed	b01
JDK-8319885	21.0.2	Liam Miller-Cushon	P3	Closed	Fixed	b07
JDK-8321934	17.0.11	Liam Miller-Cushon	P3	Resolved	Fixed	b01
JDK-8322203	11.0.23	Liam Miller-Cushon	P3	Resolved	Fixed	b01

https://bugs.openjdk.org/browse/JDK-8225377

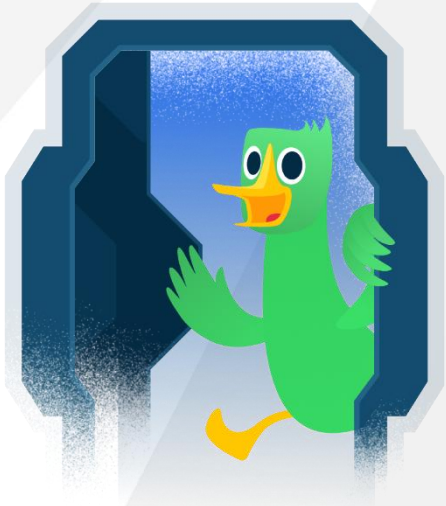
Требует свежего тулчейна

Но лучше собирайтесь последним javac!

```
// Свежий тулчейн
java.toolchain.languageVersion = JavaLanguageVersion.of(25)

compileJava {
    // Старый таргет
    options.release = 8
}
```

Java 8 вышла 12 лет назад

Письмо в будущее:
какие проблемы ушли?



06 Результаты

Что мы для себя решили?

Что мы для себя решили?

Форсируем JSpecify и Nullaway во всех проектах

Что мы для себя решили?

- Форсируем JSpecify и Nullaway во всех проектах
- Распространяем простую конфигурацию по умолчанию

Что мы для себя решили?

- Форсируем JSpecify и Nullaway во всех проектах
- Распространяем простую конфигурацию по умолчанию
- **Объясняем, как просто пользоваться инструментами**

Что мы для себя решили?

- Форсируем JSpecify и Nullaway во всех проектах
- Объясняем, как просто пользоваться инструментами
- Распространяем простую конфигурацию по умолчанию
- Приучаем всех заводить баги на тулинг

Выводы

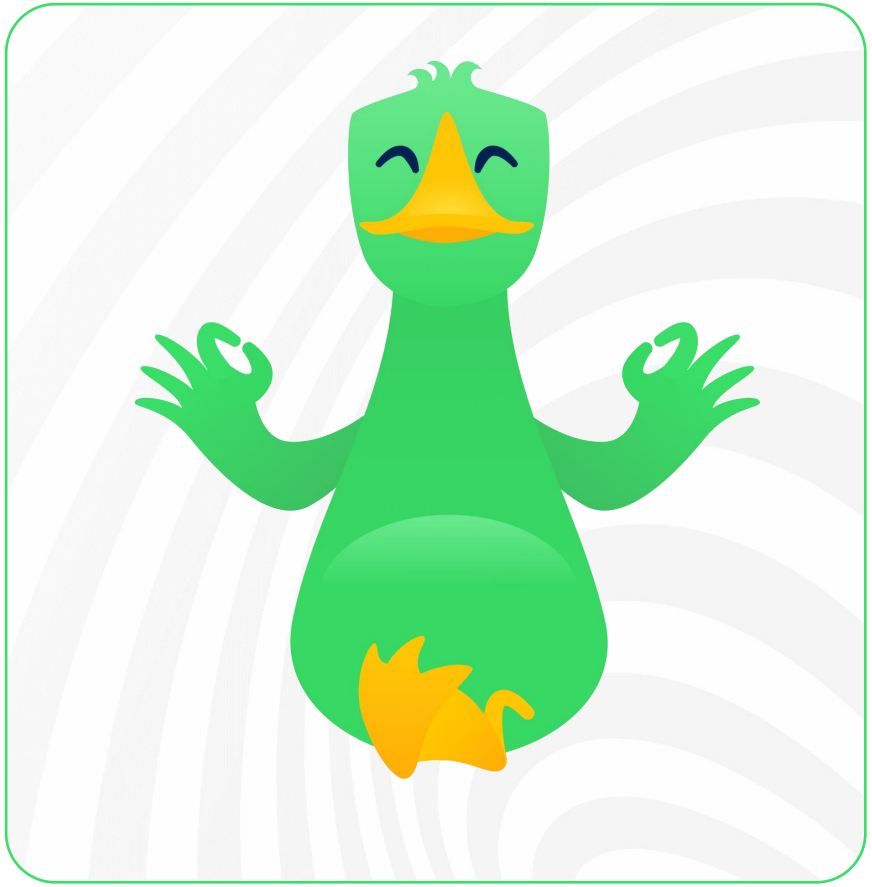
01 Мы всё ещё не защищены от NPE

02 Можно помочь и себе и окружающим

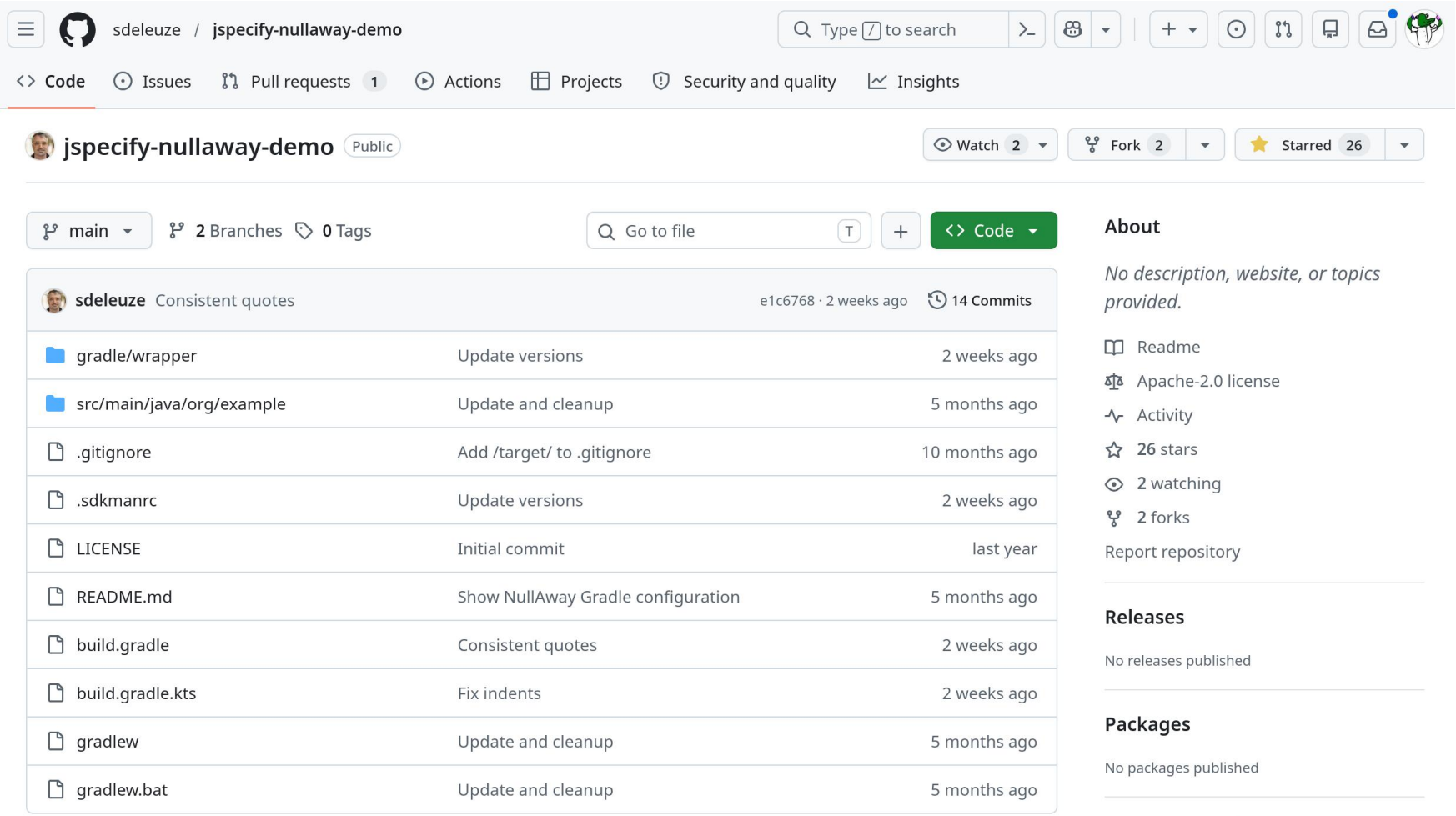
03 Для разметки nullability в 2026 — JSpecify

04 Пусть за руками следит тулинг

05 NullAway — лапочки



Куда подглядеть?



https://github.com/sdeleuze/jspecify-nullaway-demo

Темы для кулуаров

01 Null-safety в языке

02 Почему просто не null-safe язык?

03 ИИ для разметки кодовых баз

04 Q/A про наш опыт интеграции

05 Ещё необычные сценарии разметки

OZON tech

Спасибо за внимание!

Пётр Портнов
ведущий разработчик
информационных систем, Ozon

pportnov@ozon.ru github.com/JarvisCraft