Deprecation Survey

Welcome

Programming languages

ID: 88

1) Is Java the main programming language in one of your projects?*

* this question is required

( ) Yes

( ) No

ID: 40

2) What languages, other than Java, do you develop in?

Please separate their names with a semicolon (;)

____

About you

ID: 8

3) How many years of programming experience do you have?

____

ID: 38

4) What setting do you work in, when developing in Java?*

Check all that apply
* this question is required

[ ] Open source

[ ] Industrial, proprietary

ID: 5

5) Which companies do you work for, when developing in Java?

If you work at multiple companies, please separate their names with a semicolon (;)

____

ID: 39

6) What open source product(s) do you actively develop, when developing in Java?

If you actively work on multiple products, please separate their names with a semicolon (;)

____

ID: 7

7) Which of the following best describes your primary work area?

( ) Development

( ) Test

( ) Design

( ) Documentation

( ) Product Support

( ) Mangement and Administration

( ) Research

( ) Other:: ____

ID: 9

8) How many years have you been working in your current role?

____

ID: 10

9) Which of the following would best describe you, when developing in Java?*

Check all that apply
* this question is required

[ ] I often create/maintain libraries, APIs, or third-party components (that is, I am a producer of Java APIs)

[ ] I often use a libraries, APIs, or third-party component when I develop (that is, I am a consumer of Java APIs)

[ ] Other:: ____

Perspective of a Java API developer

ID: 11

10) You mentioned earlier that you develop Java based libraries, APIs, or third-party components.
Which of the following is the target audience of these products?

Check all that apply

[ ] External developers (e.g., company clients, open source developers)

[ ] Internal developers

[ ] Other:: ____

ID: 12

11) How often do you deprecate methods/features?

( ) Never

( ) Rarely

( ) Occasionally

( ) A moderate amount

( ) A great deal

ID: 132

12) You mention that you never deprecate features, could you elaborate on the reason?

____

____

____

____

ID: 72

13) In your experience, when you deprecated a feature, how frequently have you used one of the following deprecation mechanisms that Java provides?

	Never	**Almost never**	**Occasionally/Sometimes**	**Almost every time**	**Every time**
@deprecated Javadoc annotation only	( )	( )	( )	( )	( )
@Deprecated Java code annotation only	( )	( )	( )	( )	( )
Both @deprecated (javadoc) and @Deprecated (code), together	( )	( )	( )	( )	( )

ID: 13

14) Below we list some reasons why one could want to deprecate a feature.
In your experience, how frequently have these reasons motivated you?

	Never	**Almost never**	**Occasionally/Sometimes**	**Almost every time**	**Every time**
A new, better feature/method has been developed	( )	( )	( )	( )	( )
A functional issue (e.g., wrong	( )	( )	( )	( )	( )

calculations) has emerged					
A non-functional issue (e.g., performance issue) has emerged	( )	( )	( )	( )	( )
The old interface encouraged bad coding practices	( )	( )	( )	( )	( )
To mark it is a beta/experimental feature	( )	( )	( )	( )	( )
To communicate with the clients that it is going to be removed	( )	( )	( )	( )	( )
Based on a business/manageme nt decision	( )	( )	( )	( )	( )
Usage of the feature is no longer required	( )	( )	( )	( )	( )
The feature is not yet implemented (it exists as a stub)	( )	( )	( )	( )	( )

ID: 43

15) Do you see any difference in the deprecation practices of industry vs. open source? If so, please explain.

____

____

____

ID: 32

16) In your experience, after deprecating a feature/method, when did you remove it?

	Never	**Almost never**	**Occasionaly/Sometimes**	**Almost every time**	**Every time**
Before the upcoming release	( )	( )	( )	( )	( )
In the upcoming release	( )	( )	( )	( )	( )
In the release after the upcoming one	( )	( )	( )	( )	( )
After more than 2 releases	( )	( )	( )	( )	( )
Never	( )	( )	( )	( )	( )

Perspective of an API consumer

ID: 133

17) How often do you upgrade the version of the libraries/APIs that you use?

( ) Never

( ) Sometimes

( ) Always

( ) It depends: ____

ID: 37

18) Below are situations that may occur when a new version of a library/API is released. In your experience, how frequently have these situations prevented you from upgrading to a new library/API version?

	Never	**Rarely**	**Ocassionally**	**A moderate amount**	**A great deal**
A feature/method you are already using gets deprecated	( )	( )	( )	( )	( )
A feature/method you are using has changed in a breaking way (that is: it generates a compilation error with your code)	( )	( )	( )	( )	( )
A feature/method you are using has been removed	( )	( )	( )	( )	( )

A feature/method you are using has changed its behavior in a non breaking way (that is: your code compiles without errors, but has different behavior)	( )	( )	( )	( )	( )

ID: 134

19) Why do you not upgrade the version of the APIs you use?

____

____

____

____

ID: 99

20) In your experience, how frequently have you done one of the following when you upgraded to a new version of a library/API?

	Never	**Almost never**	**Occasionally/Sometimes**	**Almost every time**	**Every time**
I added a reference to a newly deprecated feature	( )	( )	( )	( )	( )

(the feature was not referenced in my codebase before)					
I kept a reference to a feature that was already in my codebase and that is deprecated in the new API release	( )	( )	( )	( )	( )
I kept a reference to a feature that was already in my codebase and that was deprecated in past API releases	( )	( )	( )	( )	( )

ID: 20

21) Below there are ways in which one can react to the deprecation of a feature/method. In your experience, how frequently have you reacted in the following ways?

	Never	**Almost never**	**Occasionally/Sometimes**	**Almost every time**	**Every time**
Doing nothing	( )	( )	( )	( )	( )
Postponing any reaction until the feature is removed	( )	( )	( )	( )	( )
Discussing the reaction with the team	( )	( )	( )	( )	( )
Reading the documentation and basing the decision on that	( )	( )	( )	( )	( )
Replacing the deprecated feature with the recommended replacement from the API	( )	( )	( )	( )	( )
Replacing the deprecated feature with some in-house solution	( )	( )	( )	( )	( )
Removing the reference to the deprecated feature	( )	( )	( )	( )	( )

ID: 28

22) Why have you removed a deprecated call without replacing it as recommended by the API developers?

We are asking this question, because previously you specified that you are (extremely) unlikely to replace it with the recommended replacement from the API

ID: 29

23) Why have you not reacted to a deprecated feature?

We are asking this question, because previously you specified that you are (extremely) likely to do nothing when a feature/method becomes deprecated

ID: 30

24) What in the documentation has prompted you to react or not react to a deprecation?

We are asking this question, because previously you specified that you are (extremely) likely to base your decision on the documentation

ID: 44

25) Below we list some reasons why one could want to react to a deprecated a feature. In your experience, how frequently have these reasons motivated you?

	Neve r	**Almos t never**	**Ocassionally/sometim es**	**Almos t every time**	**Ever y time**
A new, better feature/method has been developed	( )	( )	( )	( )	( )
A functional issue (such as wrong calculations) has emerged	( )	( )	( )	( )	( )
A non-functional issue (such as performance issue) has emerged	( )	( )	( )	( )	( )
The old interface encouraged bad coding practices	( )	( )	( )	( )	( )
It is a beta/experimental feature	( )	( )	( )	( )	( )
The documentation says it is going to be removed	( )	( )	( )	( )	( )
The API developers have removed deprecated features in the past	( )	( )	( )	( )	( )

A business/manageme nt decision	( )	( )	( )	( )	( )
Usage of the feature is no longer required	( )	( )	( )	( )	( )
The feature is not yet implemented (it exists as a stub)	( )	( )	( )	( )	( )

Problems with deprecation

ID: 35

26) From a Java API producer's perspective:

Do you see any weaknesses with the @deprecated/@Deprecated annotation? If so, please list these weaknesses and provide us with ideas on how to address them.

____
____
____
____

ID: 36

27) From a Java API consumer's perspective:

Do you see any weaknesses when it comes to encountering a deprecated entity?

If so, please list these weaknesses and provide us with ideas on how to address them.

____

____

____

____

New deprecation implementation

ID: 108

28) How desirable are the following changes to the deprecated mechanism in Java?

	Very undesirable	**Undesirable**	**Neutral**	**Desirable**	**Very desirable**
Issue runtime warnings on the usage of a deprecated feature	( )	( )	( )	( )	( )
Automated refactoring to replace deprecated calls	( )	( )	( )	( )	( )
Create generic warning	( )	( )	( )	( )	( )

that replaces the deprecation warning and allows for a warning to be thrown for reasons other than just deprecation					
Have different warning strengths that give developers more control (for example, severe warning could make code not compile, while mild warning could issue a compiler warning)	( )	( )	( )	( )	( )
Make deprecation a breaking change such that client code does not compile when a deprecated entity is used	( )	( )	( )	( )	( )

New enum that specifies reason behind deprecation (e.g. obsolete, condemned)	( )	( )	( )	( )	( )
Specify version in which it was deprecated and whether it will be removed	( )	( )	( )	( )	( )
Specify replacement feature as a String	( )	( )	( )	( )	( )

ID: 136

29) Feel free to describe more changes that you would consider desirable

____

____

____

____

Debriefing

ID: 98

30) Feel free to add any further comment on this survey, topic, etc.

____

____

____

____

Thank You!