標準チェック

標準のCheckstyleチェックは、一般的なJavaコーディングスタイルに適用でき、外部ライブラリは必要ありません。標準チェックは基本配布に含まれています。

サイトナビゲーションメニューを使用すると、機能ごとに個々のチェックを参照できます。

Checkstyleは、ソースコードに適用できる多くのチェックを提供します。以下はアルファベット順のリファレンスです。サイトのナビゲーションメニューには、機能別に整理されたリファレンスがあります。

AbbreviationAsWordInName	識別子名の略語（連続する大文字）の長さを検証し、キャメルケースの命名を強制することもできます。
AbstractClassName	あるパターンに準拠する抽象クラスの名前を確認し、`abstract`修飾子が存在することを確認します。
AnnotationLocation	言語要素の注釈の場所を確認します。
AnnotationOnSameLine	注釈がターゲットと同じ行にあることを確認します。
AnnotationUseStyle	注釈の要素のスタイルをチェックします。
AnonInnerLength	長い匿名内部クラスをチェックします。
ArrayTrailingComma	配列の初期化に末尾のコンマが含まれていることを確認します。
ArrayTypeStyle	配列型定義のスタイルをチェックします。
AtclauseOrder	javadocブロックタグまたはjavadocタグの順序を確認します。
DoubleBraceInitializationを回避する	二重中括弧の初期化を検出します。
PreventEscapedUnicodeCharacters	Unicodeエスケープ（\ u221eなど）の使用を制限します。
PreventInlineConditionals	インライン条件を検出します。
PreventNestedBlocks	ネストされたブロック（コードで自由に使用されるブロック）を検索します。
PreventNoArgumentSuperConstructorCall	引数なしのスーパークラスコンストラクターの呼び出しが存在するかどうかを確認します。
PreventStarImport	*表記を使用するインポートステートメントがないことを確認します。
PreventStaticImport	静的インポートステートメントがないことを確認します。
BooleanExpressionComplexity	式内のブール演算子（&&、||、&、|、およびへ）の数を制限します。
CatchParameterName	パラメータ`catch`名が指定されたパターンに準拠していることを確認します。
ClassDataAbstractionCoupling	指定されたクラスまたはレコード内の他のクラスのインスタンス化の数を測定します。
ClassFanOutComplexity	特定のクラス/レコード/インターフェース/列挙

	型/アノテーションが依存する他のタイプの数をチェックします。
ClassMemberImpliedModifier	クラスとレコードのネストされた型の暗黙の修飾子をチェックします。
ClassTypeParameterName	クラスタイプのパラメータ名が指定されたパターンに準拠していることを確認します。
コメントインデント	コメントと周囲のコードの間のインデントを制御します。
ConstantName	定数名が指定されたパターンに準拠していることを確認します。
CovariantEquals	共変メソッドを定義するクラスとレコード `equals()` もメソッドをオーバーライドすることを確認します`equals(Object)`。
CustomImportOrder	インポート宣言のグループがユーザーが指定した順序で表示されることを確認します。
循環的複雑度	指定された制限に対して循環的複雑度をチェックします。
DeclarationOrder	クラス、レコード、またはインターフェイス宣言の一部が、Javaプログラミング言語のコード規約で提案されている順序で表示されることを確認します。
DefaultComesLast	ステートメント 内の`default`すべての `case`sの後に あることを確認してください。`switch`
DescendantToken	他のトークンの下にある制限されたトークンをチェックします。
DesignForExtension	クラスが拡張（サブクラスの作成）用に設計されていることを確認します。
EmptyBlock	空のブロックをチェックします。
EmptyCatchBlock	空のキャッチブロックをチェックします。
EmptyForInitializerPad	イニシャライザの空のパディングをチェックします。つまり、初期化子の空で空白が必要なのか、そのような空白が禁止されているのかです。
EmptyForIteratorPad	イテレータの空のパディングをチェックします。つまり、イテレータの空で空白が必要なのか、そのような空白が禁止されているのかということです。
EmptyLineSeparator	パッケージの前に空の行区切り文字、すべてのインポート宣言、フィールド、コンストラクター、メソッド、ネストされたクラス、静的初期化子、およびインスタンス初期化子をチェックします。
EmptyStatement	空のステートメント（スタンドアロンの`";"`セミコロン）を検出します。
EqualsAvoidNull	文字列リテラルの任意の組み合わせがequals（）比較の左側にあることを確認します。
EqualsHashCode	`equals()` 他のクラスをオーバーライドするか 、オーバーライドするクラスをチェックします

	hashCode()。
ExecutableStatementCount	実行可能ステートメントの数を指定された制限に制限します。
ExplicitInitialization	クラスまたはオブジェクトメンバーのいずれかが、その型値のデフォルトに明示的に初期化されているかどうかを確認します（nullオブジェクト参照の場合、数値型の場合はゼロ、charおよびfalseの場合はゼロboolean。
フォールスルー	switchステートメントのフォールスルーをチェックします。
FileLength	長いソースファイルをチェックします。
FileTabCharacter	ソースコードにタブ文字（ '\ t'）がないことを確認します。
FinalClass	プライベートコンストラクターのみを持ち、子孫クラスを持たないクラスがfinalとして宣言されていることを確認します。
FinalLocalVariable	値が変更されないローカル変数がfinalとして宣言されていることを確認します。
FinalParameters	メソッド、コンストラクター、catch、およびfor-eachブロックのパラメーターがfinalであることを確認します。
GenericWhitespace	汎用トークン（山かっこ）「<」および「>」の周りの空白が通常の規則に正しいことを確認します。
ヘッダ	ソースファイルが指定されたヘッダーで始まることを確認します。
HiddenField	ローカル変数またはパラメーターが、同じクラスで定義されているフィールドをシャドウイングしないことを確認します。
HideUtilityClassConstructor	ユーティリティクラス（APIに静的メソッドまたはフィールドのみを含むクラス）にパブリックコンストラクターがないことを確認します。
IllegalCatch	特定の例外タイプがcatchステートメントに表示されないことを確認します。
IllegalIdentifierName	制限付きキーワードやコンテキストキーワードなど、不正な名前のセットのパターンを使用して識別子をチェックします。
IllegalImport	一連の不正なパッケージからのインポートをチェックします。
違法なインスタンス化	ファクトリメソッドが優先される場合に、不正なインスタンス化をチェックします。
IllegalThrows	指定されたタイプがスローされるように宣言されていないことを確認します。
IllegalToken	不正なトークンをチェックします。
IllegalTokenText	指定されたトークンのテキストが不正なパターンに一致していないかどうかを確認します。
IllegalType	特定のクラスまたはインターフェースが使用さ

	れていないことを確認します。
ImportControl	各パッケージおよびファイルにインポートできるものを制御します。
ImportOrder	インポートの順序/グループ化を確認します。
インデント	Javaコードの正しいインデントをチェックします。
InnerAssignment	などの部分式の割り当てをチェックし `String s = Integer.toString(i = 2);`ます。
InnerTypeLast	ネストされた（内部）クラス/インターフェイスが、すべてのinitおよびstatic initブロック、メソッド、コンストラクター、およびフィールド宣言の後に、プライマリ（トップレベル）クラスの下部で宣言されていることを確認します。
InterfaceIsType	Joshua Bloch、Effective Java、Item17を実装します-型を定義するためにのみインターフェースを使用します。
InterfaceMemberImpliedModifier	インターフェイスメンバーとネストされた型の暗黙的な修飾子をチェックします。
InterfaceTypeParameterName	インターフェイスタイプのパラメータ名が指定されたパターンに準拠していることを確認します。
InvalidJavadocPosition	Javadocが正しい位置にあることを確認します。
JavadocBlockTagLocation	javadocブロックタグが行の先頭にのみ表示され、先頭のアスタリスクと空白を無視することを確認します。
JavadocContentLocation	プロジェクト内のすべてのJavadocコメントについて、Javadocコンテンツが同じ位置から始まることを確認します。
JavadocMethod	メソッドまたはコンストラクターのJavadocをチェックします。
JavadocMissingLeadingAsterisk	javadocの 各行 に 先頭のアスタリスク があるかどうかを確認します。
JavadocMissingWhitespaceAfterAsterisk	先頭のアスタリスクの後に少なくとも1つの空白があることを確認します。
JavadocPackage	各Javaパッケージにコメントに使用されるJavadocファイルがあることを確認します。
JavadocParagraph	Javadoc段落をチェックします。
JavadocStyle	Javadocコメントを検証して、それらが整形式であることを確認します。
JavadocTagContinuationIndentation	ブロックタグの継続行のインデントをチェックします。
JavadocType	タイプ定義についてJavadocコメントをチェックします。
JavadocVariable	変数にJavadocコメントがあることを確認します。
JavaNCSS	Non Commenting Source Statements

	（NCSS）をカウントすることにより、メソッド、クラス、およびファイルの複雑さを判別します。
LambdaBodyLength	ラムダ本体の長さをチェックします。
LambdaParameterName	ラムダパラメーター名をチェックします。
LeftCurly	'{'コードブロック の左中括弧（）の配置をチェックします。
LineLength	長い行をチェックします。
LocalFinalVariableName	ローカルの最終変数名が指定されたパターンに準拠していることを確認します。
LocalVariableName	ローカルの非最終変数名が指定されたパターンに準拠していることを確認します。
MagicNumber	マジックナンバーが定数として定義されていない数値リテラルである「マジックナンバー」がないことを確認します。
MatchXpath	Xpathクエリを評価し、一致するすべてのASTノードで違反を報告します。
メンバー名	インスタンス変数名が指定されたパターンに準拠していることを確認します。
MethodCount	アクセス修飾子または総数によって、各型宣言で宣言されたメソッドの数をチェックします。
MethodLength	長いメソッドとコンストラクターをチェックします。
MethodName	メソッド名が指定されたパターンに準拠していることを確認します。
MethodParamPad	メソッド定義、コンストラクター定義、メソッド呼び出し、またはコンストラクター呼び出しの識別子の間のパディングをチェックします。パラメータリストの左括弧。
MethodTypeParameterName	メソッドタイプのパラメータ名が指定されたパターンに準拠していることを確認します。
MissingCtor	クラス（抽象クラスを除く）がコンストラクターを定義し、デフォルトのコンストラクターに依存しないことを確認します。
MissingDeprecated	注釈@DeprecatedとJavadocタグの @deprecated いずれかが存在する場合、それらの両方が存在することを確認します。
MissingJavadocMethod	メソッドまたはコンストラクターのJavadocコメントが欠落していないかどうかをチェックします。
MissingJavadocPackage	package-info.javaファイルに欠落しているJavadocコメントをチェックします。
MissingJavadocType	クラス、列挙型、インターフェイス、およびアノテーションインターフェイス定義の欠落しているJavadocコメントをチェックします。
MissingOverride	javadocタグが @Override存在するときに、注釈が存在する ことを確認します。@inheritDoc

MissingSwitchDefault	Checks that switch statement has a `default` clause.
ModifiedControlVariable	Checks that for loop control variables are not modified inside the for block.
ModifierOrder	Checks that the order of modifiers conforms to the suggestions in the Java Language specification, § 8.1.1, 8.3.1, 8.4.3 and 9.4.
MultipleStringLiterals	Checks for multiple occurrences of the same string literal within a single file.
MultipleVariableDeclarations	Checks that each variable declaration is in its own statement and on its own line.
MutableException	Ensures that exception classes (classes with names conforming to some pattern and explicitly extending classes with names conforming to other pattern) are immutable, that is, that they have only final fields.
NeedBraces	Checks for braces around code blocks.
NestedForDepth	Restricts nested `for` blocks to a specified depth.
NestedIfDepth	Restricts nested if-else blocks to a specified depth.
NestedTryDepth	Restricts nested try-catch-finally blocks to a specified depth.
NewlineAtEndOfFile	Checks whether files end with a line separator.
NoArrayTrailingComma	Checks that array initialization do not contain a trailing comma.
NoClone	Checks that the clone method is not overridden from the Object class.
NoCodeInFile	Checks whether file contains code.
NoFinalizer	Checks that there is no method `finalize` with zero parameters.
NoLineWrap	Checks that chosen statements are not line-wrapped.
NonEmptyAtclauseDescription	Checks that the block tag is followed by description.
NoEnumTrailingComma	Checks that enum definition does not contain a trailing comma.
NoWhitespaceAfter	Checks that there is no whitespace after a token.
NoWhitespaceBefore	Checks that there is no whitespace before a token.
NoWhitespaceBeforeCaseDefaultColon	Checks that there is no whitespace before the colon in a switch block.
NPathComplexity	Checks the NPATH complexity against a

	specified limit.
OneStatementPerLine	Checks that there is only one statement per line.
OneTopLevelClass	Checks that each top-level class, interface, enum or annotation resides in a source file of its own.
OperatorWrap	Checks the policy on how to wrap lines on operators.
OrderedProperties	Detects if keys in properties files are in correct order.
OuterTypeFilename	Checks that the outer type name and the file name match.
OuterTypeNumber	Checks for the number of types declared at the *outer* (or *root*) level in a file.
OverloadMethodsDeclarationOrder	Checks that overloaded methods are grouped together.
PackageAnnotation	Checks that all package annotations are in the package-info.java file.
PackageDeclaration	Ensures that a class has a package declaration, and (optionally) whether the package name matches the directory name for the source file.
PackageName	Checks that package names conform to a specified pattern.
ParameterAssignment	Disallows assignment of parameters.
ParameterName	Checks that method parameter names conform to a specified pattern.
ParameterNumber	Checks the number of parameters of a method or constructor.
ParenPad	Checks the policy on the padding of parentheses; that is whether a space is required after a left parenthesis and before a right parenthesis, or such spaces are forbidden.
PatternVariableName	Checks that pattern variable names conform to a specified pattern.
RecordComponentNumber	Checks the number of record components in the header of a record definition.
RecordComponentName	Checks that record component names conform to a specified pattern.
RecordTypeParameterName	Checks that record type parameter names conform to a specified pattern.
RedundantImport	Checks for redundant import statements.
RedundantModifier	Checks for redundant modifiers.
Regexp	Checks that a specified pattern exists, exists less than a set number of times, or does not exist in the file.

RegexpHeader	Checks the header of a source file against a header that contains a pattern for each line of the source header.
RegexpMultiline	Checks that a specified pattern matches across multiple lines in any file type.
RegexpOnFilename	Checks that a specified pattern matches based on file and/or folder path.
RegexpSingleline	Checks that a specified pattern matches a single line in any file type.
RegexpSinglelineJava	Checks that a specified pattern matches a single line in Java files.
RequireThis	Checks that references to instance variables and methods of the present object are explicitly of the form "this.varName" or "this.methodName(args)" and that those references don't rely on the default behavior when "this." is absent.
ReturnCount	Restricts the number of return statements in methods, constructors and lambda expressions.
RightCurly	Checks the placement of right curly braces (`'}'`) for code blocks.
SeparatorWrap	Checks line wrapping with separators.
SimplifyBooleanExpression	Checks for over-complicated boolean expressions.
SimplifyBooleanReturn	Checks for over-complicated boolean return statements.
SingleLineJavadoc	Checks that a Javadoc block can fit in a single line and doesn't contain block tags.
RequireEmptyLineBeforeBlockTagGroup	Checks that one blank line before the block tag if it is present in Javadoc.
SingleSpaceSeparator	Checks that non-whitespace characters are separated by no more than one whitespace.
StaticVariableName	Checks that `static`, non-`final` variable names conform to a specified pattern.
StringLiteralEquality	Checks that string literals are not used with `==` or `!=`.
SummaryJavadoc	Checks that Javadoc summary sentence does not contain phrases that are not recommended to use.
SuperClone	Checks that an overriding clone() method invokes super.clone().
SuperFinalize	Checks that an overriding finalize() method invokes super.finalize().
SuppressWarnings	Allows to specify what warnings that

	`@SuppressWarnings` is not allowed to suppress.
ThrowsCount	Restricts throws statements to a specified count.
TodoComment	Checks for TODO: comments.
TrailingComment	The check to ensure that lines with code do not end with comment.
Translation	Ensures the correct translation of code by checking property files for consistency regarding their keys.
TypecastParenPad	Checks the policy on the padding of parentheses for typecasts.
TypeName	Checks that type names conform to a specified pattern.
UncommentedMain	Detects uncommented `main` methods.
UniqueProperties	Detects duplicated keys in properties files.
UnnecessaryParentheses	Checks if unnecessary parentheses are used in a statement or expression.
UnnecessarySemicolonInEnumeration	Checks if unnecessary semicolon is in enum definitions.
UnnecessarySemicolonInTryWithResources	Checks if unnecessary semicolon is used in last resource declaration.
UnnecessarySemicolonAfterOuterTypeDeclaration	Checks if unnecessary semicolon is used after type declaration.
UnnecessarySemicolonAfterTypeMemberDeclaration	Checks if unnecessary semicolon is used after type member declaration.
UnusedImports	Checks for unused import statements.
UpperEll	Checks that long constants are defined with an upper ell.
UnusedLocalVariable	Checks that a local variable is declared and/or assigned, but not used.
VariableDeclarationUsageDistance	Checks the distance between declaration of variable and its first usage.
VisibilityModifier	Checks visibility of class members.
WhitespaceAfter	Checks that a token is followed by whitespace, with the exception that it does not check for whitespace after the semicolon of an empty for iterator.
WhitespaceAround	トークンが空白で囲まれていることを確認します。
WriteTag	ユーザー定義のJavadocタグが、定義された形式でJavadocコメントに存在する必要があります。

ホルダーチェック

これらのチェックは通常のチェックではなく、通常、フィルターがそれ自体では取得できない情報を収集するための特殊なフィルターに関連付けられています。

SuppressWarningsHolder	`@SuppressWarnings`注釈 からの一連のチェック抑制を維持し ます。