


# Checks


The [Standard Checkstyle Checks](#) are applicable to general Java coding style and require no external libraries. The standard checks are included in the base distribution.

The site navigation menu lets you browse the individual checks by functionality.

Checkstyle [provides many checks that you can apply to your source code](#). Below is an alphabetical reference, the site navigation menu provides a reference organized by functionality.

<a href="#">AbbreviationAsWordInName</a>	The Check validate abbreviations(consecutive capital letters) length in identifier name, it also allow in enforce camel case naming.
<a href="#">AbstractClassName</a>	Ensures that the <a href="#">names of abstract classes</a> conforming to some regular expression.
<a href="#">AnnotationLocation</a>	Check location of annotation on language elements.
<a href="#">AnnotationUseStyle</a>	This check controls the style with the usage of annotations.
<a href="#">AnonInnerLength</a>	Checks for long anonymous inner classes.
<a href="#">ArrayTrailingComma</a>	Checks if array initialization contains optional trailing comma.
<a href="#">ArrayTypeStyle</a>	Checks the style of array type definitions.
<a href="#">AtclauseOrder</a>	Checks the order of at-clauses.
<a href="#">AvoidEscapedUnicodeCharacters</a>	Restrict using Unicode escapes.
<a href="#">AvoidInlineConditionals</a>	Detects inline conditionals.
<a href="#">AvoidNestedBlocks</a>	Finds nested blocks.
<a href="#">AvoidStarImport</a>	Check that finds import statements that use the * notation.
<a href="#">AvoidStaticImport</a>	Check that finds static imports.
<a href="#">BooleanExpressionComplexity</a>	Restricts nested boolean operators (&&,   , &,   and ^) to a specified depth (default = 3).
<a href="#">ClassDataAbstractionCoupling</a>	This metric measures the number of instantiations of other classes within the given class.
<a href="#">ClassFanOutComplexity</a>	The number of other classes a given class relies on.
<a href="#">ClassTypeParameterName</a>	Checks that class type parameter names conform to a format specified by the format property.
<a href="#">ConstantName</a>	Checks that constant names conform to a format specified by the format property.
<a href="#">CovariantEquals</a>	Checks that if a class defines a covariant method equals, then it defines method equals(java.lang.Object).
<a href="#">CustomImportOrder</a>	Checks that the groups of import declarations appear in the order specified by the user.
<a href="#">CyclomaticComplexity</a>	Checks cyclomatic complexity against a specified limit.
<a href="#">DeclarationOrder</a>	Checks that the parts of a class or interface declaration appear in the order suggested by the <a href="#">Code Conventions for the Java Programming Language</a>  .
<a href="#">DefaultComesLast</a>	Check that the default is after all the cases in a switch statement.
<a href="#">DescendantToken</a>	Checks for restricted tokens beneath other tokens.
<a href="#">DesignForExtension</a>	Checks that classes are designed for inheritance.
<a href="#">EmptyBlock</a>	Checks for empty blocks.
<a href="#">EmptyCatchBlock</a>	Checks for empty catch blocks with few options to skip violation.

EmptyForInitializerPad	Checks the padding of an empty for initializer; that is whether a space is required at an empty for initializer, or such spaces are forbidden.
EmptyForIteratorPad	Checks the padding of an empty for iterator; that is whether a space is required at an empty for iterator, or such spaces are forbidden.
EmptyLineSeparator	Checks for blank line separators.
EmptyStatement	Detects empty statements (standalone ';').
EqualsAvoidNull	Checks that any combination of String literals with optional assignment is on the left side of an equals() comparison.
EqualsHashCode	Checks that classes that override equals() also override hashCode().
ExecutableStatementCount	Restricts the number of executable statements to a specified limit (default = 30).
ExplicitInitialization	Checks if any class or object member explicitly initialized to default for its type value (null for object references, zero for numeric types and char and false for boolean).
FallThrough	Checks for fall through in switch statements Finds locations where a case contains Java code - but lacks a break, return, throw or continue statement.
FileLength	Checks for long source files.
FileTabCharacter	Checks to see if a file contains a tab character.
FinalClass	Checks that class which has only private ctors is declared as final.
FinalLocalVariable	Ensures that local variables that never get their values changed, must be declared final.
FinalParameters	Check that method/constructor/catch/foreach parameters are final.
GenericWhitespace	Checks that the whitespace around the Generic tokens < and > are correct to the <i>typical</i> convention.
Header	Checks the header of the source against a fixed header file.
HiddenField	Checks that a local variable or a parameter does not shadow a field that is defined in the same class.
HideUtilityClassConstructor	Make sure that utility classes (classes that contain only static methods) do not have a public constructor.
IllegalCatch	Catching java.lang.Exception, java.lang.Error or java.lang.RuntimeException is almost never acceptable.
IllegalImport	Checks for imports from a set of illegal packages.
IllegalInstantiation	Checks for illegal instantiations where a factory method is preferred.
IllegalThrows	Throwing java.lang.Error or java.lang.RuntimeException is almost never acceptable.
IllegalToken	Checks for illegal tokens.
IllegalTokenText	Checks for illegal token text.
IllegalType	Checks that particular class are never used as types in variable declarations, return values or parameters.
ImportControl	Check that controls what packages can be imported in each package.
ImportOrder	Ensures that groups of imports come in a specific order.
Indentation	Checks correct indentation of Java Code.
InnerAssignment	Checks for assignments in subexpressions, such as in <code>String s = Integer.toString(i = 2);</code> .

InnerTypeLast	Check nested (internal) classes/interfaces are declared at the bottom of the class after all method and field declarations.
InterfaceIsType	Implements Bloch, Effective Java, Item 17 - Use Interfaces only to define types.
InterfaceTypeParameterName	Checks that interface type parameter names conform to a format specified by the format property.
JavaNCSS	This check calculates the Non Commenting Source Statements (NCSS) metric for Java source files and methods.
JavadocMethod	Checks the Javadoc of a method or constructor.
JavadocPackage	Checks that all packages have a package documentation.
JavadocTagContinuationIndentation	Checks the indentation of the continuation lines in at-clauses.
JavadocParagraph	Checks Javadoc paragraphs.
JavadocStyle	Custom Checkstyle Check to validate Javadoc.
JavadocType	Checks the Javadoc of a type.
JavadocVariable	Checks that a variable has Javadoc comment.
LeftCurly	Checks the placement of left curly braces on types, methods and other blocks:
LineLength	Checks for long lines.
LocalFinalVariableName	Checks that local final variable names conform to a format specified by the format property.
LocalVariableName	Checks that local, non-final variable names conform to a format specified by the format property.
MagicNumber	Checks for magic numbers.
MemberName	Checks that instance variable names conform to a format specified by the format property.
MethodCount	Checks the number of methods declared in each type.
MethodLength	Checks for long methods.
MethodName	Checks that method names conform to a format specified by the format property.
MethodParamPad	Checks the padding between the identifier of a method definition, constructor definition, method call, or constructor invocation; and the left parenthesis of the parameter list.
MethodTypeParameterName	Checks that class type parameter names conform to a format specified by the format property.
MissingCtor	Checks that classes (except abstract one) define a ctor and don't rely on the default one.
MissingDeprecated	This class is used to verify that both the java.lang.Deprecated annotation is present and the @deprecated Javadoc tag is present when either is present.
MissingOverride	This class is used to verify that the java.lang.Override annotation is present when the { @inheritDoc } javadoc tag is present.
MissingSwitchDefault	Checks that switch statement has "default" clause.
ModifiedControlVariable	Check for ensuring 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 <a href="#">Java Language specification, sections 8.1.1, 8.3.1 and 8.4.3</a>  .
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 exceptions (defined as any class name conforming to some regular expression) are immutable.
NPathComplexity	Checks the npath complexity against a specified limit (default = 200).
NeedBraces	Checks for braces around code blocks.
NestedForDepth	Restricts nested <code>for</code> blocks to a specified depth (default = 1).
NestedIfDepth	Restricts nested if-else blocks to a specified depth (default = 1).
NestedTryDepth	Restricts nested try-catch-finally blocks to a specified depth (default = 1).
NewlineAtEndOfFile	Checks that there is a newline at the end of each file.
NoClone	Checks that the clone method is not overridden from the Object class.
NoFinalizer	Checks that no method having zero parameters is defined using the name <i>finalize</i> .
NonEmptyAtclauseDescription	Checks that the at-clause tag is followed by description .
NoLineWrap	Checks that chosen statements are not line-wrapped.
NoWhitespaceAfter	Checks that there is no whitespace after a token.
NoWhitespaceBefore	Checks that there is no whitespace before a token.
OneStatementPerLine	Checks there is only one statement per line.
OneTopLevelClass	Checks that each top-level class, interfaces or enum resides in a source file of its own.
OperatorWrap	Checks line wrapping for operators.
OuterTypeFilename	Checks that the outer type name and the file name match.
OuterTypeName	Checks for the number of defined types at the "outer" level.
OverloadMethodsDeclarationOrder	Checks that overload methods are grouped together.
PackageAnnotation	This check makes sure that all package annotations are in the package-info.java file.
PackageDeclaration	Ensures there is a package declaration and (optionally) in the correct directory.
PackageName	Checks that package names conform to a format specified by the format property.
ParameterAssignment	Disallow assignment of parameters.
ParameterName	Checks that parameter names conform to a format specified by the format property.
ParameterNumber	Checks the number of parameters that a method or constructor has.
ParenPad	Checks 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, with the exception that it does not check for padding of the right parenthesis at an empty for iterator.
RedundantImport	Checks for imports that are redundant.
RedundantModifier	Checks for redundant modifiers in interface and annotation definitions.
Regexp	A check that makes sure that a specified pattern exists (or not) in the file.
RegexpHeader	Checks the header of the source against a header file that contains a
RegexpMultiline	Implementation of a check that looks that matches across multiple lines in any file type.

<a href="#">RegexpSingleline</a>	Implementation of a check that looks for a single line in any file type.
<a href="#">RegexpSinglelineJava</a>	Implementation of a check that looks for a single line in Java files.
<a href="#">RequireThis</a>	Checks that code doesn't rely on the "this" default.
<a href="#">ReturnCount</a>	Restricts return statements to a specified count (default = 2).
<a href="#">RightCurly</a>	Checks the placement of right curly braces.
<a href="#">SeparatorWrap</a>	Checks line wrapping with separators.
<a href="#">SingleLineJavadoc</a>	Checks that a Javadoc block which can fit on a single line and doesn't contain at-clauses
<a href="#">SimplifyBooleanExpression</a>	Checks for overly complicated boolean expressions.
<a href="#">SimplifyBooleanReturn</a>	Checks for overly complicated boolean return statements.
<a href="#">StaticVariableName</a>	Checks that static, non-final variable names conform to a format specified by the format property.
<a href="#">StringLiteralEquality</a>	Checks that string literals are not used with == or !=.
<a href="#">SummaryJavadoc</a>	Checks that Javadoc summary sentence does not contain phrases that are not recommended to use.
<a href="#">SuperClone</a>	Checks that an overriding clone() method invokes super.clone().
<a href="#">SuperFinalize</a>	Checks that an overriding finalize() method invokes super.finalize().
<a href="#">SuppressWarnings</a>	This check allows you to specify what warnings that
<a href="#">SuppressWarningsHolder</a>	This check allows for finding code that should not be reported by Checkstyle
<a href="#">ThrowsCount</a>	Restricts throws statements to a specified count (default = 4).
<a href="#">TodoComment</a>	A check for TODO comments.
<a href="#">TrailingComment</a>	The check to ensure that requires that comments be the only thing on a line.
<a href="#">Translation</a>	The TranslationCheck class helps to ensure the correct translation of code by checking property files for consistency regarding their keys.
<a href="#">TypeName</a>	Checks that type names conform to a format specified by the format property.
<a href="#">TypecastParenPad</a>	Checks the padding of parentheses for typecasts.
<a href="#">UncommentedMain</a>	Detects uncommented main methods.
<a href="#">UniqueProperties</a>	Detects duplicated keys in properties files.
<a href="#">UnnecessaryParentheses</a>	Checks if unnecessary parentheses are used in a statement or expression.
<a href="#">UnusedImports</a>	Checks for unused import statements.
<a href="#">UpperEll</a>	Checks that long constants are defined with an upper ell.
<a href="#">VariableDeclarationUsageDistance</a>	Checks the distance between declaration of variable and its first usage.
<a href="#">VisibilityModifier</a>	Checks visibility of class members.
<a href="#">WhitespaceAfter</a>	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.
<a href="#">WhitespaceAround</a>	Checks that a token is surrounded by whitespace.
<a href="#">WriteTag</a>	Outputs a Javadoc tag as information.