关于rules的描述及判断集中在src/java目录下的edu.umd.cs.findbugs.detect包中

Bug patterns见：<http://findbugs.sourceforge.net/bugDescriptions.html>

调用BugInstance的方法及BugInstance构造方法参数 实例：

bugAccumulator.accumulateBug(new BugInstance(this, "RV\_DONT\_JUST\_NULL\_CHECK\_READLINE", NORMAL\_PRIORITY).addClassAndMethod(this), this);

bugReporter.reportBug(new BugInstance(this, "CN\_IMPLEMENTS\_CLONE\_BUT\_NOT\_CLONEABLE", priority).addClass(this)

.addMethod(cloneMethodAnnotation));

1.

名称：IO: Doomed attempt to append to an object output stream (IO\_APPENDING\_TO\_OBJECT\_OUTPUT\_STREAM)

描述：This code opens a file in append mode and then wraps the result in an object output stream. This won't allow you to append to an existing object output stream stored in a file. If you want to be able to append to an object output stream, you need to keep the object output stream open.

The only situation in which opening a file in append mode and the writing an object output stream could work is if on reading the file you plan to open it in random access mode and seek to the byte offset where the append started.

TODO: example.

相关代码：AppendingingToObjectOutputStream.java

举例：

备注：

2.

名称：AT: Sequence of calls to concurrent abstraction may not be atomic (AT\_OPERATION\_SEQUENCE\_ON\_CONCURRENT\_ABSTRACTION)

描述：This code contains a sequence of calls to a concurrent abstraction (such as a concurrent hash map). These calls will not be executed atomically.

相关代码：AtomicityProblem.java

举例：

备注：\* if we get from a ConcurrentHashMap and assign to a variable... and don't do

\* anything else and perform a null check on it... and then do a set on it...

\* (or anything else inside the if that modifies it?) then we have a bug.

3.

名称：BAC\_BAD\_APPLET\_CONSTRUCTOR（List中没有）

描述：

相关代码：BadAppletConstructor.java

举例：

备注：

4.

名称：BOA: Class overrides a method implemented in super class Adapter wrongly (BOA\_BADLY\_OVERRIDDEN\_ADAPTER)

描述：This method overrides a method found in a parent class, where that class is an Adapter that implements a listener defined in the java.awt.event or javax.swing.event package. As a result, this method will not get called when the event occurs.

相关代码：BadlyOverriddenAdapter.java

举例：

备注：AST

5. 名称：SQL: Method attempts to access a prepared statement parameter with index 0 (SQL\_BAD\_PREPARED\_STATEMENT\_ACCESS)

描述：A call to a setXXX method of a prepared statement was made where the parameter index is 0. As parameter indexes start at index 1, this is always a mistake.

相关代码：BadResultSetAccess.java

举例：

备注：

6. 名称：SQL: Method attempts to access a result set field with index 0 (SQL\_BAD\_RESULTSET\_ACCESS)

描述：A call to getXXX or updateXXX methods of a result set was made where the field index is 0. As ResultSet fields start at index 1, this is always a mistake.

相关代码：BadResultSetAccess.java

举例：

备注：

7.

名称：RE: "." or "|" used for regular expression (RE\_POSSIBLE\_UNINTENDED\_PATTERN)

描述：A String function is being invoked and "." or "|" is being passed to a parameter that takes a regular expression as an argument. Is this what you intended?

相关代码：BadSyntaxForRegularExpression.java

举例：

s.replaceAll(".", "/") will return a String in which every character has been replaced by a '/' character

s.split(".") always returns a zero length array of String

"ab|cd".replaceAll("|", "/") will return "/a/b/|/c/d/"

"ab|cd".split("|") will return array with six (!) elements: [, a, b, |, c, d]

备注：

8.

名称：

RE: File.separator used for regular expression (RE\_CANT\_USE\_FILE\_SEPARATOR\_AS\_REGULAR\_EXPRESSION)

描述：The code here uses File.separator where a regular expression is required. This will fail on Windows platforms, where the File.separator is a backslash, which is interpreted in a regular expression as an escape character. Amoung other options, you can just use File.separatorChar=='\\' ? "\\\\" : File.separator instead of File.separator

相关代码：BadSyntaxForRegularExpression.java

举例：

备注：

9.

名称：RE: Invalid syntax for regular expression (RE\_BAD\_SYNTAX\_FOR\_REGULAR\_EXPRESSION)

描述：The code here uses a regular expression that is invalid according to the syntax for regular expressions. This statement will throw a PatternSyntaxException when executed.

相关代码：BadSyntaxForRegularExpression.java

举例：

备注：

10.

名称：RV: Method checks to see if result of String.indexOf is positive (RV\_CHECK\_FOR\_POSITIVE\_INDEXOF)

描述：The method invokes String.indexOf and checks to see if the result is positive or non-positive. It is much more typical to check to see if the result is negative or non-negative. It is positive only if the substring checked for occurs at some place other than at the beginning of the String.

相关代码：BadUseOfReturnValue.java

举例：

备注：

11.

名称：RV: Method discards result of readLine after checking if it is non-null (RV\_DONT\_JUST\_NULL\_CHECK\_READLINE)

描述：The value returned by readLine is discarded after checking to see if the return value is non-null. In almost all situations, if the result is non-null, you will want to use that non-null value. Calling readLine again will give you a different line.

相关代码：BadUseOfReturnValue.java

举例：

备注：

12.

名称：NP: Method with Boolean return type returns explicit null (NP\_BOOLEAN\_RETURN\_NULL)

描述：A method that returns either Boolean.TRUE, Boolean.FALSE or null is an accident waiting to happen. This method can be invoked as though it returned a value of type boolean, and the compiler will insert automatic unboxing of the Boolean value. If a null value is returned, this will result in a NullPointerException.

相关代码：BooleanReturnNull.java

举例：

备注：

13.

名称：NP: equals() method does not check for null argument (NP\_EQUALS\_SHOULD\_HANDLE\_NULL\_ARGUMENT)

描述： This implementation of equals(Object) violates the contract defined by java.lang.Object.equals() because it does not check for null being passed as the argument. All equals() methods should return false if passed a null value.

相关代码：BuildUnconditionalParamDerefDatabase.java

举例：

备注：

14.

名称：NP: Parameter must be non-null but is marked as nullable (NP\_PARAMETER\_MUST\_BE\_NONNULL\_BUT\_MARKED\_AS\_NULLABLE)

描述：This parameter is always used in a way that requires it to be non-null, but the parameter is explicitly annotated as being Nullable. Either the use of the parameter or the annotation is wrong.

相关代码：BuildUnconditionalParamDerefDatabase.java

举例：

备注：

15.

名称：TESTING

描述：

相关代码：CallToUnconditionalThrower.java

举例：

备注：

if (newResult) {

bugReporter.reportBug(new BugInstance(this, "TESTING", Priorities.NORMAL\_PRIORITY)

.addClassAndMethod(classContext.getJavaClass(), method)

.addString("Call to method that always throws Exception").addMethod(primaryXMethod)

.describe(MethodAnnotation.METHOD\_CALLED).addSourceLine(classContext, method, loc));

}

16.

名称：DMI\_UNSUPPORTED\_METHOD

描述：

相关代码：CallToUnsupportedMethod.java

举例：

备注：

17.

名称：TESTING

描述：

相关代码：CbeckMustOverrideSuperAnnotation.java

举例：

备注：

18.

名称：TESTING

描述：

相关代码：CheckAnalysisContextContainedAnnotation.java

举例：

备注：

19.

名称：

描述：

相关代码：CheckCalls.java

举例：

备注：AST;CFG

20.

名称：FB\_MISSING\_EXPECTED\_WARNING

描述：

相关代码：CheckExpextedWarnings.java

举例：

备注：

21.

名称：JCIP: Fields of immutable classes should be final (JCIP\_FIELD\_ISNT\_FINAL\_IN\_IMMUTABLE\_CLASS)

描述：The class is annotated with net.jcip.annotations.Immutable or javax.annotation.concurrent.Immutable, and the rules for those annotations require that all fields are final. .

相关代码：CheckImmutableAnnotation.java

举例：

备注：

22. 名称：NP: Method relaxes nullness annotation on return value (NP\_METHOD\_RETURN\_RELAXING\_ANNOTATION)

描述：A method should always implement the contract of a method it overrides. Thus, if a method takes is annotated as returning a @Nonnull value, you shouldn't override that method in a subclass with a method annotated as returning a @Nullable or @CheckForNull value. Doing so violates the contract that the method shouldn't return null.

相关代码：CheckRelaxingNullnessAnnotation.java

举例：

备注：

23.

名称：NP: Method tightens nullness annotation on parameter (NP\_METHOD\_PARAMETER\_TIGHTENS\_ANNOTATION)

描述：A method should always implement the contract of a method it overrides. Thus, if a method takes a parameter that is marked as @Nullable, you shouldn't override that method in a subclass with a method where that parameter is @Nonnull. Doing so violates the contract that the method should handle a null parameter.

相关代码：CheckRelaxingNullnessAnnotation.java

举例：

备注：

24. 名称：TQ: Comparing values with incompatible type qualifiers (TQ\_COMPARING\_VALUES\_WITH\_INCOMPATIBLE\_TYPE\_QUALIFIERS)

描述：A value specified as carrying a type qualifier annotation is compared with a value that doesn't ever carry that qualifier.

More precisely, a value annotated with a type qualifier specifying when=ALWAYS is compared with a value that where the same type qualifier specifies when=NEVER.

For example, say that @NonNegative is a nickname for the type qualifier annotation @Negative(when=When.NEVER). The following code will generate this warning because the return statement requires a @NonNegative value, but receives one that is marked as @Negative.

相关代码：CheckTypeQualifiers.java

举例：

public boolean example(@Negative Integer value1, @NonNegative Integer value2) {

return value1.equals(value2);

}

备注：

25.

名称：TQ: Value that might not carry a type qualifier is always used in a way requires that type qualifier (TQ\_MAYBE\_SOURCE\_VALUE\_REACHES\_ALWAYS\_SINK)

描述：A value that is annotated as possibility not being an instance of the values denoted by the type qualifier, and the value is guaranteed to be used in a way that requires values denoted by that type qualifier.

相关代码：CheckTypeQualifiers.java

举例：

备注：emitSourceWarning()

名称：TQ: Value that might carry a type qualifier is always used in a way prohibits it from having that type qualifier (TQ\_MAYBE\_SOURCE\_VALUE\_REACHES\_NEVER\_SINK)

描述：A value that is annotated as possibility being an instance of the values denoted by the type qualifier, and the value is guaranteed to be used in a way that prohibits values denoted by that type qualifier.

相关代码：CheckTypeQualifiers.java

举例：

备注：emitSourceWarning()

名称：TQ: Value required to have type qualifier, but marked as unknown (TQ\_EXPLICIT\_UNKNOWN\_SOURCE\_VALUE\_REACHES\_ALWAYS\_SINK)

描述：A value is used in a way that requires it to be always be a value denoted by a type qualifier, but there is an explicit annotation stating that it is not known where the value is required to have that type qualifier. Either the usage or the annotation is incorrect.

相关代码：CheckTypeQualifiers.java

举例：

备注：emitSourceWarning()

名称：TQ: Value required to not have type qualifier, but marked as unknown (TQ\_EXPLICIT\_UNKNOWN\_SOURCE\_VALUE\_REACHES\_NEVER\_SINK)

描述：A value is used in a way that requires it to be never be a value denoted by a type qualifier, but there is an explicit annotation stating that it is not known where the value is prohibited from having that type qualifier. Either the usage or the annotation is incorrect.

相关代码：CheckTypeQualifiers.java

举例：

备注：emitSourceWarning()

名称：TQ: Value without a type qualifier used where a value is required to have that qualifier (TQ\_UNKNOWN\_VALUE\_USED\_WHERE\_ALWAYS\_STRICTLY\_REQUIRED)

描述：A value is being used in a way that requires the value be annotation with a type qualifier. The type qualifier is strict, so the tool rejects any values that do not have the appropriate annotation.

To coerce a value to have a strict annotation, define an identity function where the return value is annotated with the strict annotation. This is the only way to turn a non-annotated value into a value with a strict type qualifier annotation.

相关代码：CheckTypeQualifiers.java

举例：

备注：warning

名称：TQ: Value annotated as carrying a type qualifier used where a value that must not carry that qualifier is required (TQ\_ALWAYS\_VALUE\_USED\_WHERE\_NEVER\_REQUIRED)

描述：A value specified as carrying a type qualifier annotation is consumed in a location or locations requiring that the value not carry that annotation.

More precisely, a value annotated with a type qualifier specifying when=ALWAYS is guaranteed to reach a use or uses where the same type qualifier specifies when=NEVER.

相关代码：CheckTypeQualifiers.java

举例：For example, say that @NonNegative is a nickname for the type qualifier annotation @Negative(when=When.NEVER). The following code will generate this warning because the return statement requires a @NonNegative value, but receives one that is marked as @Negative.

public @NonNegative Integer example(@Negative Integer value) {

return value;

}

备注：warning

名称：TQ: Value annotated as never carrying a type qualifier used where value carrying that qualifier is required (TQ\_NEVER\_VALUE\_USED\_WHERE\_ALWAYS\_REQUIRED)

描述：A value specified as not carrying a type qualifier annotation is guaranteed to be consumed in a location or locations requiring that the value does carry that annotation.

More precisely, a value annotated with a type qualifier specifying when=NEVER is guaranteed to reach a use or uses where the same type qualifier specifies when=ALWAYS.

TODO: example

相关代码：CheckTypeQualifiers.java

举例：

备注：warning

名称：CN: Class implements Cloneable but does not define or use clone method (CN\_IDIOM)

描述：Class implements Cloneable but does not define or use the clone method.

相关代码：CloneIdiom.java

举例：

备注：AST

名称：CN: clone method does not call super.clone() (CN\_IDIOM\_NO\_SUPER\_CALL)

描述：This non-final class defines a clone() method that does not call super.clone(). If this class ("A") is extended by a subclass ("B"), and the subclass B calls super.clone(), then it is likely that B's clone() method will return an object of type A, which violates the standard contract for clone().

If all clone() methods call super.clone(), then they are guaranteed to use Object.clone(), which always returns an object of the correct type.

相关代码：CloneIdiom.java

举例：

备注：AST

名称：CN: Class defines clone() but doesn't implement Cloneable (CN\_IMPLEMENTS\_CLONE\_BUT\_NOT\_CLONEABLE)

描述：This class defines a clone() method but the class doesn't implement Cloneable. There are some situations in which this is OK (e.g., you want to control how subclasses can clone themselves), but just make sure that this is what you intended.

相关代码：CloneIdiom.java

举例：

备注：

名称：Se: Comparator doesn't implement Serializable (SE\_COMPARATOR\_SHOULD\_BE\_SERIALIZABLE)

描述：This class implements the Comparator interface. You should consider whether or not it should also implement the Serializable interface. If a comparator is used to construct an ordered collection such as a TreeMap, then the TreeMap will be serializable only if the comparator is also serializable. As most comparators have little or no state, making them serializable is generally easy and good defensive programming.

相关代码：ComparatorIdiom.java

举例：

备注：

名称：CI: Class is final but declares protected field (CI\_CONFUSED\_INHERITANCE)

描述：This class is declared to be final, but declares fields to be protected. Since the class is final, it can not be derived from, and the use of protected is confusing. The access modifier for the field should be changed to private or public to represent the true use for the field.

相关代码：ConfusedInheritance.java

举例：

备注：

名称：IA: Potentially ambiguous invocation of either an inherited or outer method (IA\_AMBIGUOUS\_INVOCATION\_OF\_INHERITED\_OR\_OUTER\_METHOD)

描述：An inner class is invoking a method that could be resolved to either a inherited method or a method defined in an outer class. For example, you invoke foo(17), which is defined in both a superclass and in an outer method. By the Java semantics, it will be resolved to invoke the inherited method, but this may not be want you intend.

If you really intend to invoke the inherited method, invoke it by invoking the method on super (e.g., invoke super.foo(17)), and thus it will be clear to other readers of your code and to FindBugs that you want to invoke the inherited method, not the method in the outer class.

If you call this.foo(17), then the inherited method will be invoked. However, since FindBugs only looks at classfiles, it can't tell the difference between an invocation of this.foo(17) and foo(17), it will still complain about a potential ambiguous invocation.

相关代码：ConfusionBetweenInheritedAndOuterMethod.java

举例：

备注：

名称：CAA\_COVARIANT\_ARRAY\_FIELD

CAA\_COVARIANT\_ARRAY\_RETURN

CAA\_COVARIANT\_ARRAY\_LOCAL

CAA\_COVARIANT\_ARRAY\_ELEMENT\_STORE

描述：

相关代码：CovariantArrayAssignment.java

举例：

备注：

名称：PT: Absolute path traversal in servlet (PT\_ABSOLUTE\_PATH\_TRAVERSAL)

描述：The software uses an HTTP request parameter to construct a pathname that should be within a restricted directory, but it does not properly neutralize absolute path sequences such as "/abs/path" that can resolve to a location that is outside of that directory. See http://cwe.mitre.org/data/definitions/36.html for more information.

FindBugs looks only for the most blatant, obvious cases of absolute path traversal. If FindBugs found any, you almost certainly have more vulnerabilities that FindBugs doesn't report. If you are concerned about absolute path traversal, you should seriously consider using a commercial static analysis or pen-testing tool.

相关代码：CrossSiteScripting.java

举例：

备注：

名称：PT: Relative path traversal in servlet (PT\_RELATIVE\_PATH\_TRAVERSAL)

描述：The software uses an HTTP request parameter to construct a pathname that should be within a restricted directory, but it does not properly neutralize sequences such as ".." that can resolve to a location that is outside of that directory. See http://cwe.mitre.org/data/definitions/23.html for more information.

FindBugs looks only for the most blatant, obvious cases of relative path traversal. If FindBugs found any, you almost certainly have more vulnerabilities that FindBugs doesn't report. If you are concerned about relative path traversal, you should seriously consider using a commercial static analysis or pen-testing tool.

相关代码：CrossSiteScripting.java

举例：

备注：

名称：HRS: HTTP cookie formed from untrusted input (HRS\_REQUEST\_PARAMETER\_TO\_COOKIE)

描述：This code constructs an HTTP Cookie using an untrusted HTTP parameter. If this cookie is added to an HTTP response, it will allow a HTTP response splitting vulnerability. See http://en.wikipedia.org/wiki/HTTP\_response\_splitting for more information.

FindBugs looks only for the most blatant, obvious cases of HTTP response splitting. If FindBugs found any, you almost certainly have more vulnerabilities that FindBugs doesn't report. If you are concerned about HTTP response splitting, you should seriously consider using a commercial static analysis or pen-testing tool.

相关代码：CrossSiteScripting.java

举例：

备注：

名称：XSS\_REQUEST\_PARAMETER\_TO\_SEND\_ERROR

HRS\_REQUEST\_PARAMETER\_TO\_HTTP\_HEADER

XSS\_REQUEST\_PARAMETER\_TO\_JSP\_WRITER

XSS\_REQUEST\_PARAMETER\_TO\_SERVLET\_WRITER

描述：见http://findbugs.sourceforge.net/bugDescriptions.html

相关代码：CrossSiteScripting.java

举例：

备注：

名称：Dm: Reliance on default encoding (DM\_DEFAULT\_ENCODING)

描述：Found a call to a method which will perform a byte to String (or String to byte) conversion, and will assume that the default platform encoding is suitable. This will cause the application behaviour to vary between platforms. Use an alternative API and specify a charset name or Charset object explicitly.

相关代码：DefaultEncodingDetector.java

举例：

备注：

名称：DP: Method invoked that should be only be invoked inside a doPrivileged block (DP\_DO\_INSIDE\_DO\_PRIVILEGED)

描述：This code invokes a method that requires a security permission check. If this code will be granted security permissions, but might be invoked by code that does not have security permissions, then the invocation needs to occur inside a doPrivileged block.

相关代码：DoInsideDoPrivileged.java

举例：

备注：

名称：DP: Classloaders should only be created inside doPrivileged block (DP\_CREATE\_CLASSLOADER\_INSIDE\_DO\_PRIVILEGED)

描述：This code creates a classloader, which needs permission if a security manage is installed. If this code might be invoked by code that does not have security permissions, then the classloader creation needs to occur inside a doPrivileged block.

相关代码：DoInsideDoPrivileged.java

举例：

备注：

名称：IMSE\_DONT\_CATCH\_IMSE

描述：

相关代码：DontCatchIllegalMonitorStateException.java

举例：

备注：

名称：RV\_RETURN\_VALUE\_OF\_PUTIFABSENT\_IGNORED

描述：

相关代码：DontIgnoreResultOfPutIfAbsent.java

举例：

备注：

名称：NM\_FUTURE\_KEYWORD\_USED\_AS\_MEMBER\_IDENTIFIER

描述：

相关代码：DontUseEnum.java

举例：

备注：AST

名称：NM\_FUTURE\_KEYWORD\_USED\_AS\_IDENTIFIER

描述：

相关代码：DontUseEnum.java

举例：

备注：AST

名称：DE\_MIGHT\_DROP

描述：

相关代码：DroppedException.java

举例：

备注：

名称：DE\_MIGHT\_IGNORE

描述：

相关代码：DroppedException.java

举例：

备注：

名称：DMI\_EMPTY\_DB\_PASSWORD

DMI\_CONSTANT\_DB\_PASSWORD

DMI\_USELESS\_SUBSTRING

DMI\_HARDCODED\_ABSOLUTE\_FILENAME

描述：

相关代码：DumbMethodInvocations.java

举例：

备注：

名称：DM\_INVALID\_MIN\_MAX

DMI\_DOH

DMI\_ARGUMENTS\_WRONG\_ORDER

DMI\_FUTILE\_ATTEMPT\_TO\_CHANGE\_MAXPOOL\_SIZE\_OF\_SCHEDULED\_THREAD\_POOL\_EXECUTOR

DMI\_COLLECTION\_OF\_URLS

INT\_VACUOUS\_COMPARISON

BC\_EQUALS\_METHOD\_SHOULD\_WORK\_FOR\_ALL\_OBJECTS

DMI\_RANDOM\_USED\_ONLY\_ONCE

TESTING

DM\_BOXED\_PRIMITIVE\_TOSTRING

DM\_BOXED\_PRIMITIVE\_FOR\_PARSING

DM\_BOXED\_PRIMITIVE\_FOR\_COMPARE

DMI\_VACUOUS\_CALL\_TO\_EASYMOCK\_METHOD

DMI\_THREAD\_PASSED\_WHERE\_RUNNABLE\_EXPECTED

DMI\_LONG\_BITS\_TO\_DOUBLE\_INVOKED\_ON\_INT

RV\_ABSOLUTE\_VALUE\_OF\_RANDOM\_INT

RV\_ABSOLUTE\_VALUE\_OF\_HASHCODE

RV\_REM\_OF\_HASHCODE

RV\_REM\_OF\_RANDOM\_INT

INT\_BAD\_REM\_BY\_1

INT\_BAD\_COMPARISON\_WITH\_SIGNED\_BYTE

INT\_BAD\_COMPARISON\_WITH\_NONNEGATIVE\_VALUE

BIT\_IOR\_OF\_SIGNED\_BYTE

BIT\_ADD\_OF\_SIGNED\_BYTE

NP\_IMMEDIATE\_DEREFERENCE\_OF\_READLINE

RV\_01\_TO\_INT

DM\_NEXTINT\_VIA\_NEXTDOUBLE

SW\_SWING\_METHODS\_INVOKED\_IN\_SWING\_THREAD

DMI\_ANNOTATION\_IS\_NOT\_VISIBLE\_TO\_REFLECTION

DMI\_CALLING\_NEXT\_FROM\_HASNEXT

DM\_STRING\_CTOR

DM\_RUN\_FINALIZERS\_ON\_EXIT

DM\_STRING\_VOID\_CTOR

DM\_EXIT

DM\_GC

DM\_BOOLEAN\_CTOR

DM\_STRING\_TOSTRING

DM\_CONVERT\_CASE

DM\_BOXED\_PRIMITIVE\_TOSTRING

DM\_NEW\_FOR\_GETCLASS

DM\_USELESS\_THREAD

DMI\_BIGDECIMAL\_CONSTRUCTED\_FROM\_DOUBLE

INT\_BAD\_COMPARISON\_WITH\_INT\_VALUE

INT\_VACUOUS\_BIT\_OPERATION

描述：

相关代码：DumbMethods.java

举例：

备注：

名称：DB\_DUPLICATE\_BRANCHES

DB\_DUPLICATE\_SWITCH\_CLAUSES

描述：

相关代码：DuplicateBranches.java

举例：

备注：

名称：AM\_CREATES\_EMPTY\_ZIP\_FILE\_ENTRY

AM\_CREATES\_EMPTY\_JAR\_FILE\_ENTRY

描述：

相关代码：EmptyZipFileEntry.java

举例：

备注：

名称：EQ\_CHECK\_FOR\_OPERAND\_NOT\_COMPATIBLE\_WITH\_THIS

描述：

相关代码：EqualsOperandShouldHaveClassCompatibleWithThis.java

举例：

备注：

名称：FI\_FINALIZER\_ONLY\_NULLS\_FIELDS

FI\_FINALIZER\_NULLS\_FIELDS

描述：

相关代码：FinalizerNullsFields.java

举例：

备注：

名称：NP\_NULL\_INSTANCEOF

BC\_IMPOSSIBLE\_CAST

BC\_IMPOSSIBLE\_INSTANCEOF

BC\_VACUOUS\_INSTANCEOF

BC\_IMPOSSIBLE\_DOWNCAST\_OF\_TOARRAY

BC\_IMPOSSIBLE\_DOWNCAST

BC\_IMPOSSIBLE\_CAST

BC\_IMPOSSIBLE\_INSTANCEOF

BC\_UNCONFIRMED\_CAST

BC\_UNCONFIRMED\_CAST\_OF\_RETURN\_VALUE

BC\_BAD\_CAST\_TO\_CONCRETE\_COLLECTION

BC\_BAD\_CAST\_TO\_ABSTRACT\_COLLECTION

描述：

相关代码：FindBadCast2.java

举例：

备注：

名称：QF\_QUESTIONABLE\_FOR\_LOOP

描述：

相关代码：FindBadForLoop.java

举例：

备注：

名称：CD\_CIRCULAR\_DEPENDENCY

描述：

相关代码：FindCircularDependencies.java

举例：

备注：

名称：CO\_COMPARETO\_INCORRECT\_FLOATING

CO\_COMPARETO\_RESULTS\_MIN\_VALUE

描述：

相关代码：FindComparatorProblems.java

举例：

备注：

名称：DC\_DOUBLECHECK

DC\_PARTIALLY\_CONSTRUCTED

描述：

相关代码：FindDoubleCheck.java

举例：

备注：

名称：ESync\_EMPTY\_SYNC

描述：

相关代码：FindEmptySynchronizedBlock.java

举例：

备注：

名称：SA\_FIELD\_SELF\_ASSIGNMENT

SA\_LOCAL\_DOUBLE\_ASSIGNMENT

描述：

相关代码：FindFieldAssignment.java

举例：

备注：

名称：FI\_PUBLIC\_SHOULD\_BE\_PROTECTED

FI\_EMPTY

FI\_NULLIFY\_SUPER

FI\_USELESS

FI\_MISSING\_SUPER\_CALL

FI\_EXPLICIT\_INVOCATION

描述：

相关代码：FindFinalizeInvocations.java

举例：

备注：AST

名称：FE\_FLOATING\_POINT\_EQUALITY

FE\_TEST\_IF\_EQUAL\_TO\_NOT\_A\_NUMBER

描述：

相关代码：FindFloatEquality.java

举例：

备注：

名称：FL\_MATH\_USING\_FLOAT\_PRECISION

描述：

相关代码：FindFloatMath.java

举例：

备注：

名称：EQ\_OTHER\_USE\_OBJECT

EQ\_OTHER\_NO\_OBJECT

EQ\_SELF\_USE\_OBJECT

EQ\_SELF\_NO\_OBJECT

EQ\_DONT\_DEFINE\_EQUALS\_FOR\_ENUM

EQ\_COMPARETO\_USE\_OBJECT\_EQUALS

CO\_SELF\_NO\_OBJECT

HE\_HASHCODE\_USE\_OBJECT\_EQUALS

HE\_HASHCODE\_NO\_EQUALS

HE\_EQUALS\_USE\_HASHCODE

HE\_EQUALS\_NO\_HASHCODE

HE\_INHERITS\_EQUALS\_USE\_HASHCODE

EQ\_DOESNT\_OVERRIDE\_EQUALS

EQ\_ABSTRACT\_SELF

CO\_ABSTRACT\_SELF

HE\_USE\_OF\_UNHASHABLE\_CLASS

HE\_SIGNATURE\_DECLARES\_HASHING\_OF\_UNHASHABLE\_CLASS

描述：

相关代码：FindHEmismatch.java

举例：

备注：

名称：MSF\_MUTABLE\_SERVLET\_FIELD

IS\_FIELD\_NOT\_GUARDED

IS2\_INCONSISTENT\_SYNC

描述：

相关代码：FindInconsistentSync2.java

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：

名称：

描述：

相关代码：

举例：

备注：