Chart-4

Buggy:

```
4506
4507                    Collection c = r.getAnnotations();
4508                    Iterator i = c.iterator();
4509                    while (i.hasNext()) {
4510                        XYAnnotation a = (XYAnnotation) i.next();
4511                        if (a instanceof XYAnnotationBoundsInfo) {
4512                            includedAnnotations.add(a);
4513                        }
4514                    }
4515            }
```

Fixed:

```
4499
4500                if (r != null) {
4501                    Collection c = r.getAnnotations();
4502                    Iterator i = c.iterator();
4503                    while (i.hasNext()) {
4504                        XYAnnotation a = (XYAnnotation) i.next();
4505                        if (a instanceof XYAnnotationBoundsInfo) {
4506                            includedAnnotations.add(a);
4507                        }
4508                    }
4509                }
4510            }
4511        }
```

Closure-82

Buggy:

```
176   public final boolean isEmptyType() {
177     return isNoType() || isNoObjectType() || isNoResolvedType();
178   }
```

Fixed:

```
169   public final boolean isEmptyType() {
170     return isNoType() || isNoObjectType() || isNoResolvedType() ||
171         (registry.getNativeFunctionType(
172             JSTypeNative.LEAST_FUNCTION_TYPE) == this);
173   }
```

Lang-61

Buggy:

```
1788            }
1789            char[] thisBuf = buffer;
1790            int len = thisBuf.length - strLen;
1791            outer:
1792            for (int i = startIndex; i < len; i++) {
1793                for (int j = 0; j < strLen; j++) {
1794                    if (str.charAt(j) != thisBuf[i + j]) {
1795                        continue outer;
1796                    }
1797                }
1798                return i;
1799            }
1800            return -1;
```

Fixed:

```
1780            return -1;
1781        }
1782        char[] thisBuf = buffer;
1783        int len = size - strLen + 1;
1784        outer:
1785        for (int i = startIndex; i < len; i++) {
1786            for (int j = 0; j < strLen; j++) {
1787                if (str.charAt(j) != thisBuf[i + j]) {
1788                    continue outer;
1789                }
1790            }
1791            return i;
1792        }
1793        return -1;
1794    }
1795
```

Mockito-11
Buggy:

```
67      @Override
68      public boolean equals(Object o) {
69              return method.equals(o);
70      }
71
72      @Override
73      public int hashCode() {
74          return 1;
75      }
76  }
```

Fixed:

```
60      @Override
61      public boolean equals(Object o) {
62          if (this == o) {
63              return true;
64          }
65          if (o instanceof DelegatingMethod) {
66              DelegatingMethod that = (DelegatingMethod) o;
67              return method.equals(that.method);
68          } else {
69              return method.equals(o);
70          }
71      }
72
73      @Override
74      public int hashCode() {
75          return method.hashCode();
76      }
77  }
```

Closure-45
Buggy:

```
732           for (Assign assign : assignsByVar.get(var)) {
733             if (assign.isPropertyAssign) {
734               hasPropertyAssign = true;
735             } else if (!NodeUtil.isLiteralValue(
736                 assign.assignNode.getLastChild(), true)) {
737               assignedToUnknownValue = true;
738             }
739           }
740
741           if (assignedToUnknownValue && hasPropertyAssign) {
```

```
904       this.maybeAliased =
!assignNode.getParent().isExprResult();
905       this.mayHaveSecondarySideEffects =
906           maybeAliased ||
```

Fixed:

```
732           boolean maybeEscaped = false;
733           for (Assign assign : assignsByVar.get(var)) {
734             if (assign.isPropertyAssign) {
735               hasPropertyAssign = true;
736             } else if (!NodeUtil.isLiteralValue(
737                 assign.assignNode.getLastChild(), true)) {
738               assignedToUnknownValue = true;
739             }
740             if (assign.maybeAliased) {
741               maybeEscaped = true;
742             }
743           }
744
745           if ((assignedToUnknownValue || maybeEscaped)
 && hasPropertyAssign) {
```

```
908       this.maybeAliased =
NodeUtil.isExpressionResultUsed(assignNode);
909       this.mayHaveSecondarySideEffects =
```

Lang-36

Buggy:

```
491         if (!Character.isDigit(lastChar)) {
492             if (expPos > -1 && expPos < str.length() - 1) {
```

Fixed:

```
491         if (!Character.isDigit(lastChar) && lastChar != '.') {
492             if (expPos > -1 && expPos < str.length() - 1) {
493                 exp = str.substring(expPos + 1, str.length() -
```

Math-78

Buggy:

```
190
191                         // this is a corner case:
192                         // - there was an event near ta,
193                         // - there is another event between ta and tb
194                         // - when ta was computed, convergence was reached on the "wrong side" of th
    e interval
195                         // this implies that the real sign of ga is the same as gb, so we need to sl
    ightly
196                         // shift ta to make sure ga and gb get opposite signs and the solver won't c
    omplain
197                         // about bracketing
198                             // this should never happen
```

Fixed:

```
190
191                     if (ga * gb > 0) {
192                         // this is a corner case:
193                         // - there was an event near ta,
194                         // - there is another event between ta and tb
195                         // - when ta was computed, convergence was reached on the "wrong side" of th
    e interval
196                         // this implies that the real sign of ga is the same as gb, so we need to sl
    ightly
197                         // shift ta to make sure ga and gb get opposite signs and the solver won't c
    omplain
198                         // about bracketing
199                         final double epsilon = (forward ? 0.25 : -0.25) * convergence;
200                         for (int k = 0; (k < 4) && (ga * gb > 0); ++k) {
201                             ta += epsilon;
202                             interpolator.setInterpolatedTime(ta);
203                             ga = handler.g(ta, interpolator.getInterpolatedState());
204                         }
205                         if (ga * gb > 0) {
206                             // this should never happen
207                             throw MathRuntimeException.createInternalError(null);
208                         }
209                     }
```

Math-66

Buggy:

```
43      public BrentOptimizer() {
44          setMaxEvaluations(Integer.MAX_VALUE);
45          setMaximalIterationCount(100);
46          setAbsoluteAccuracy(1E-10);
47          setRelativeAccuracy(1.0e-14);
48      }
49
50      /**
51       * Perform the optimization.
52       *
53       * @return the optimum.
54       */
55      protected double doOptimize()
56          throws MaxIterationsExceededException, FunctionEvaluationException {
57          throw new UnsupportedOperationException();
58      }
59      public double optimize(final UnivariateRealFunction f, final GoalType goalType, final double mi
    n, final double max, final double startValue) throws MaxIterationsExceededException, FunctionEvaluat
    ionException {
60          clearResult();
61          return localMin(getGoalType() == GoalType.MINIMIZE,
62                          f, goalType, min, startValue, max,
63                          getRelativeAccuracy(), getAbsoluteAccuracy());
64      }
65      public double optimize(final UnivariateRealFunction f, final GoalType goalType, final double mi
    n, final double max) throws MaxIterationsExceededException, FunctionEvaluationException {
66          return optimize(f, goalType, min, max, min + GOLDEN_SECTION * (max - min));
67      }
```

```
93      private double localMin(boolean isMinim,
94                              UnivariateRealFunction f,
95                              GoalType goalType,
96                              double lo, double mid, double hi,
97                              double eps, double t)
98          throws MaxIterationsExceededException, FunctionEvaluationException {
99          if (eps <= 0) {
100             throw new NotStrictlyPositiveException(eps);
101         }
102         if (t <= 0) {
103             throw new NotStrictlyPositiveException(t);
104         }
105         double a, b;
106         if (lo < hi) {
107             a = lo;
108             b = hi;
109         } else {
110             a = hi;
111             b = lo;
112         }
113
114         double x = mid;
115         double v = x;
116         double w = x;
117         double d = 0;
118         double e = 0;
119         double fx = computeObjectiveValue(f, x);
120         if (goalType == GoalType.MAXIMIZE) {
121             fx = -fx;
122         }
123         double fv = fx;
124         double fw = fx;
125
126         int count = 0;
127         while (count < maximalIterationCount) {
128             double m = 0.5 * (a + b);
```

```
                }

                double fu = computeObjectiveValue(f, u);
                if (goalType == GoalType.MAXIMIZE) {
                    fu = -fu;
                }
```

```
            } else { // termination
                setResult(x, (goalType == GoalType.MAXIMIZE) ? -fx : fx, count);
                return x;
            }
            ++count;
        }
        throw new MaxIterationsExceededException(maximalIterationCount);
    }
}
```

Fixed:

```
    public BrentOptimizer() {
        setMaxEvaluations(1000);
        setMaximalIterationCount(100);
        setAbsoluteAccuracy(1e-11);
        setRelativeAccuracy(1e-9);
    }

    /**
     * Perform the optimization.
     *
     * @return the optimum.
     */
    protected double doOptimize()
        throws MaxIterationsExceededException, FunctionEvaluationException {
```

```
        return localMin(getGoalType() == GoalType.MINIMIZE,
                        getMin(), getStartValue(), getMax(),
                        getRelativeAccuracy(), getAbsoluteAccuracy());
```

```
    }
```

```
                }

                double fu = computeObjectiveValue(u);
                if (!isMinim) {
                    fu = -fu;
```

```
86      private double localMin(boolean isMinim,
87                              double lo, double mid, double hi,
88                              double eps, double t)
89          throws MaxIterationsExceededException, FunctionEvaluationException {
90          if (eps <= 0) {
91              throw new NotStrictlyPositiveException(eps);
92          }
93          if (t <= 0) {
94              throw new NotStrictlyPositiveException(t);
95          }
96          double a, b;
97          if (lo < hi) {
98              a = lo;
99              b = hi;
100         } else {
101             a = hi;
102             b = lo;
103         }
104
105         double x = mid;
106         double v = x;
107         double w = x;
108         double d = 0;
109         double e = 0;
110         double fx = computeObjectiveValue(x);
111         if (!isMinim) {
112             fx = -fx;
113         }
114         double fv = fx;
115         double fw = fx;
116
117         while (true) {
```

```
227                 } else { // termination
228                     setFunctionValue(isMinim ? fx : -fx);
229                     return x;
230                 }
231                 incrementIterationsCounter();
232             }
233         }
234     }
```

Chart 24

```
130     public Paint getPaint(double value) {
131         double v = Math.max(value, this.lowerBound);
132         v = Math.min(v, this.upperBound);
133         int g = (int) ((value - this.lowerBound) / (this.upperBound
134                 - this.lowerBound) * 255.0);
135         return new Color(g, g, g);
136     }
```

```
130     public Paint getPaint(double value) {
131         double v = Math.max(value, this.lowerBound);
132         v = Math.min(v, this.upperBound);
133         int g = (int) ((v - this.lowerBound) / (this.upperBound
134                 - this.lowerBound) * 255.0);
135         return new Color(g, g, g);
136     }
```

Math-31

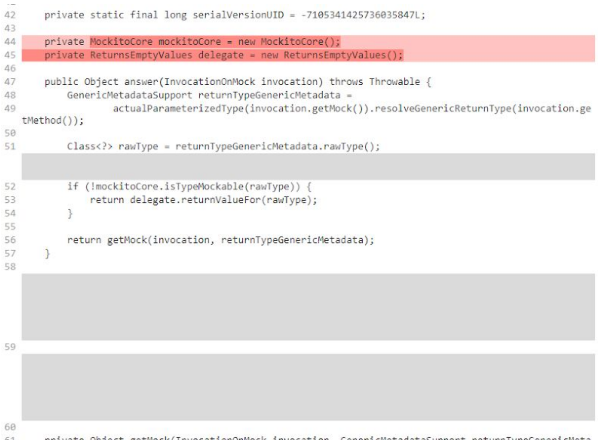
```
{
    "Project: ": "Math-31",
    "  Class: ": "org.apache.commons.math3.util.ContinuedFraction",
    "  Lines: ": [
        134,
        135,
        140,
        141,
        142,
        143,
        144,
        145,
        146,
        147,
        148,
        149,
        150,
        151,
        152,
        153,
        154,
        155,
        156,
        157,
        158,
        159,
        160,
        161,
        162,
        163,
        164,
        165,
        166,
        167,
        168,
        169,
        170,
        185,
        186,
        187,
        188,
        189
    ]
```

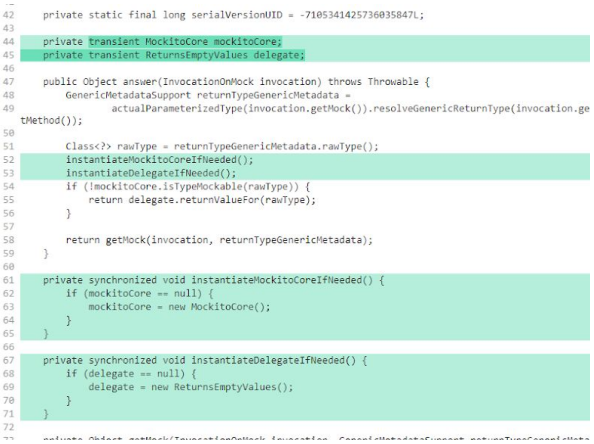

Time -22


```
{
    "Project: ": "Time-22",
    "  Class: ": "org.joda.time.base.BasePeriod",
    "  Lines: ": [
        224,
        225,
        226,
        227,
        228,
        229,
        222,
        223
    ]
},
```

```
"Project: ": "Mockito-23",
"  Class: ": "org.mockito.internal.stubbing.defaultanswers.ReturnsDeepStubs",
"  Lines: ": [
     128,
     135,
     136,
     137,
     138,
     139,
     44,
     45,
     51,
     52,
     53,
     54,
     60,
     61,
     62,
     63,
     64,
     65,
     66,
     67,
     68,
     69,
     70,
     71,
     72,
     111,
     112,
     113,
     114,
     126,
     127
]
```


mockito-23





Project: Lang

Bug/Warning no: 23

Warning Reported by Spot Bugs:

```
    {
        "    Proj": "Lang-23",
        "   Class": "org.apache.commons.lang3.text.ExtendedMessageFormat",
        "     Cat": "STYLE",
        "  Abbrev": "Eq",
        "    Type": "EQ_DOESNT_OVERRIDE_EQUALS",
        "Priority": "2",
        "    Rank": "17",
        "     Msg": "org.apache.commons.lang3.text.ExtendedMessageFormat doesn't override
java.text.MessageFormat.equals(Object)",
        "  Method": "equals",
        "   Field": "",
        "   Lines": [
            [
                1,
                1,
                null
            ]
        ]
    },
```


Methodology: Removed/Fixed Warning

Actual Code Changes in the Buggy and the Fixed version of the Code:

```
262     */
263
264    /**
265     * Return the hashcode.
266     *
267     * @return the hashcode
268     */
269
```

```
263     */
264    @Override
265    public boolean equals(Object obj) {
266        if (obj == this) {
267            return true;
268        }
269        if (obj == null) {
270            return false;
271        }
272        if (!super.equals(obj)) {
273            return false;
274        }
275        if (ObjectUtils.notEqual(getClass(), obj.getClass())) {
276          return false;
277        }
278        ExtendedMessageFormat rhs = (ExtendedMessageFormat)obj;
279        if (ObjectUtils.notEqual(toPattern, rhs.toPattern)) {
280            return false;
281        }
282        if (ObjectUtils.notEqual(registry, rhs.registry)) {
283            return false;
284        }
285        return true;
286    }
287
288    /**
289     * Return the hashcode.
290     *
291     * @return the hashcode
292     */
293    @Override
294    public int hashCode() {
295        int result = super.hashCode();
296        result = HASH_SEED * result + ObjectUtils.hashCode(registry);
297        result = HASH_SEED * result + ObjectUtils.hashCode(toPattern);
298        return result;
299    }
300
```

Category: Full Match

Comment: org.apache.commons.lang3.text.ExtendedMessageFormat **now overrides** java.text.MessageFormat.equals(Object) in the fixed version of the code.

Project: Lang

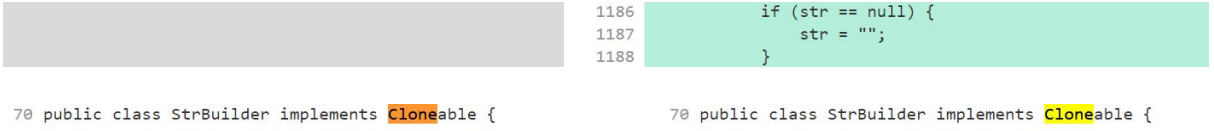
Bug/Warning no: 47

Warning Reported by Spot Bugs:

```
    {
        "    Proj": "Lang-47",
        "   Class": "org.apache.commons.lang.text.StrBuilder",
        "     Cat": "BAD_PRACTICE",
        "  Abbrev": "CN",
        "    Type": "CN_IDIOM",
        "Priority": "2",
        "    Rank": "16",
        "     Msg": "Class org.apache.commons.lang.text.StrBuilder implements Cloneable but does not define or use
clone method",
        "  Method": "",
        "   Field": "",
        "   Lines": [
            [
                98,
                2551,
                null
            ]
        ]
    },
```


Methodology: Differential Based Warning

Actual Code Changes in the Buggy and the Fixed version of the Code:



Category: Mismatch

Comment: Warning was “StrBuilder implements Cloneable but does not define or use clone method”. The issue is not resolved, but some unexpected code changes are being done.

Project: Lang

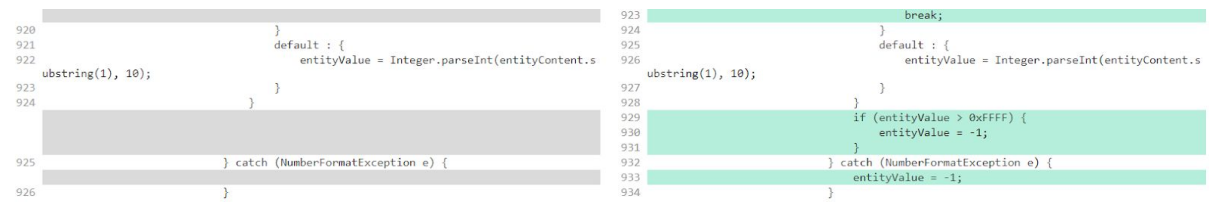
Bug/Warning no: 62

Warning Reported by Error Prone:


```
    {
        " Proj": "Lang-62",
        "Class": "org.apache.commons.lang.Entities",
        " Type": "error",
        "  Cat": "FallThrough",
        "  Msg": "Execution may fall through from the previous case; add a `// fall through` comment before this
line if it was deliberate",
        " Code": "                                    default : {",
        " Mark": "                                    ^",
        " Line": 921
    },
```


Methodology: Removed/Fixed Warning

Actual Code Changes in the Buggy and the Fixed version of the Code:



Category: Bad Practice

Comment: The warning was "Execution may fall through from the previous case; add a `// fall through` comment before this line if it was deliberate", but it is not added in the fixed version of the same.

Project: Math

Bug/Warning no: 48

Warning Reported by Spot Bugs:


```
{
    "    Proj": "Math-48",
    "   Class": "",
    "     Cat": "",
    "  Abbrev": "",
    "    Type": "NO_WARNING",
    "Priority": "",
    "    Rank": "",
    "     Msg": "",
    "  Method": "",
    "   Field": "",
    "   Lines": []
},
```


Methodology: Removed/Fixed Warning

Category: No Warning

Comment: No Warning.

Project: Math

Bug/Warning no: 57

Warning Reported by Error Prone:


```
{
    " Proj": "Math-57",
    "Class": "org.apache.commons.math.stat.clustering.KMeansPlusPlusClusterer",
    " Type": "warning",
    "  Cat": "NarrowingCompoundAssignment",
    "  Msg": "Compound assignments from double to int hide lossy casts",
    " Code": "                sum += d * d;",
    " Mark": "                    ^",
    " Line": 180
},
```


Methodology: Removed/Fixed Warning

Actual Code Changes in the Buggy and the Fixed version of the Code:

Category: Full Match

Comment: The warning was "Compound assignments from double to int hide lossy casts". The data type was converted from int to double in the fixed version.

Project: Math

Bug/Warning no: 102

Warning Reported by Error Prone:


```
    {
        " Proj": "Math-102",
        "Class": "org.apache.commons.math.stat.inference.ChiSquareTestImpl",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "chiSquareTest implements method in ChiSquareTest; expected @Override",
        " Code": "    public double chiSquareTest(double[] expected, long[] observed)",
        " Mark": "                  ^",
        " Line": 95
    },
```


Methodology: Differential Based Warning

Actual Code Changes in the Buggy and the Fixed version of the Code:

```
95     public double chiSquareTest(double[] expected, long[] o
   bserved)
96         throws IllegalArgumentException, MathException {
97         distribution.setDegreesOfFreedom(expected.length -
    1.0);
98         return 1.0 - distribution.cumulativeProbability(
99             chiSquare(expected, observed));
100    }
```

Before Change

```
112    public double chiSquareTest(double[] expected, long[] o
   bserved)
113        throws IllegalArgumentException, MathException {
114        distribution.setDegreesOfFreedom(expected.length -
    1.0);
115        return 1.0 - distribution.cumulativeProbability(
116            chiSquare(expected, observed));
117    }
```

After Change

```
73         }
74         double sumSq = 0.0d;
75         double dev = 0.0d;
76         for (int i = 0; i < observed.length; i++) {
77             dev = ((double) observed[i] - expected[i]);
78             sumSq += dev * dev / expected[i];
```

Before Change

```
73         }
74         double sumExpected = 0d;
75         double sumObserved = 0d;
76         for (int i = 0; i < observed.length; i++) {
77             sumExpected += expected[i];
78             sumObserved += observed[i];
79         }
80         double ratio = 1.0d;
81         boolean rescale = false;
82         if (Math.abs(sumExpected - sumObserved) > 10E-6) {
83             ratio = sumObserved / sumExpected;
84             rescale = true;
85         }
86         double sumSq = 0.0d;
87         double dev = 0.0d;
88         for (int i = 0; i < observed.length; i++) {
89             if (rescale) {
90                 dev = ((double) observed[i] - ratio * expec
   ted[i]);
91                 sumSq += dev * dev / (ratio * expected[i]);
92             } else {
93                 dev = ((double) observed[i] - expected[i]);
94                 sumSq += dev * dev / expected[i];
95             }
```

After Change

Additional unneeded code changes

Category: Mismatch

Comment: The warning was "chiSquareTest implements method in ChiSquareTest; expected @Override". But, @override is not added. Some additional implementation is done.

Project: Mockito

Bug/Warning no: 25

Warning Reported by Error Prone:


```
{
    " Proj": "Mockito-25",
    "Class": "org.mockito.internal.stubbing.defaultanswers.ReturnsDeepStubs",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "answer implements method in Answer; expected @Override",
    " Code": "            public Object answer(InvocationOnMock invocation) throws Throwable {",
    " Mark": "                                 ^",
    " Line": 86
},
```


Methodology: Differential Based Warning

Actual Code Changes in the Buggy and the Fixed version of the Code:

```
85          container.addAnswer(new Answer<Object>() {
86              public Object answer(InvocationOnMock invocation)
    throws Throwable {
87                  return mock;
88              }
89          }, false);
90
91          return mock;
92      }
```

```
108         container.addAnswer(new Answer<Object>() {
109             public Object answer(InvocationOnMock invocation
    throws Throwable {
110                 return mock;
111             }
112         }, false);
113
114         return mock;
115     }
```

Category: Mismatch

Comment: The warning was "answer implements method in Answer; expected @Override". But, @override is not added. Some additional implementation is done.

Project: Chart

Bug/Warning no: 08

Warning Reported by Error Prone:


```
{
    " Proj": "Chart-8",
    "Class": "org.jfree.data.time.Week",
    " Type": "error",
    "  Cat": "ChainingConstructorIgnoresParameter",
    "  Msg": "The called constructor accepts a parameter with the same name and type as one of its caller's
parameters, but its caller doesn't pass that parameter to it.  It's likely that it was intended to.",
    " Code": "        this(time, RegularTimePeriod.DEFAULT_TIME_ZONE, Locale.getDefault());",
    " Mark": "             ^",
    " Line": 175
},
```


Methodology: Differential Based Warning, Removed/Fixed Warning.

Actual Code Changes in the Buggy and the Fixed version of the Code:

```
175         this(time, RegularTimePeriod.DEFAULT_TIME_ZONE,
    Locale.getDefault());
```

```
175         this(time, zone, Locale.getDefault());
```

Category: Full Match

Comment: The warning was "The called constructor accepts a parameter with the same name and type as one of its caller's parameters, but it's caller doesn't pass that parameter to it. It's likely that it was intended to". And, the parameter has been added in the fixed version.

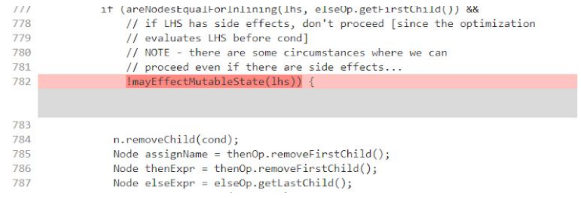
Project: Closure

Bug/Warning no: 132

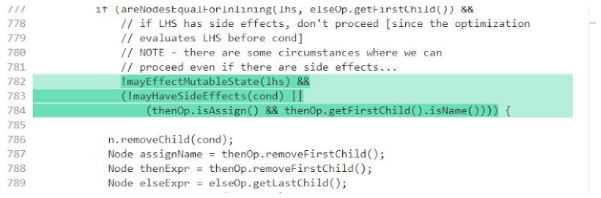
Warning Reported by diffs_parsed.json:


```
{
    "Project: ": "Closure-132",
    "  Class: ": "com.google.javascript.jscomp.PeepholeSubstituteAlternateSyntax",
    "  Lines: ": [
        784,
        785,
        782,
        783
    ]
},
```


Actual Code Changes in the Buggy and the fixed version of the code:



```
///         if (areNodesEqualForInlining(lhs, elseOp.getFirstChild()) &&
778             // if LHS has side effects, don't proceed [since the optimization
779             // evaluates LHS before cond]
780             // NOTE - there are some circumstances where we can
781             // proceed even if there are side effects...
782             !mayEffectMutableState(lhs)) {
783
784           n.removeChild(cond);
785           Node assignName = thenOp.removeFirstChild();
786           Node thenExpr = thenOp.removeFirstChild();
787           Node elseExpr = elseOp.getLastChild();
```



```
///         if (areNodesEqualForInlining(lhs, elseOp.getFirstChild()) &&
778             // if LHS has side effects, don't proceed [since the optimization
779             // evaluates LHS before cond]
780             // NOTE - there are some circumstances where we can
781             // proceed even if there are side effects...
782             !mayEffectMutableState(lhs) &&
783             (!mayHaveSideEffects(cond) ||
784                 (thenOp.isAssign() && thenOp.getFirstChild().isName()))) {
785
786           n.removeChild(cond);
787           Node assignName = thenOp.removeFirstChild();
788           Node thenExpr = thenOp.removeFirstChild();
789           Node elseExpr = elseOp.getLastChild();
```

Category: Domain Specific

Comments: None of the tools that we used identified /reported this error. This could be due to some domain specific configuration of the tool, or the bug patterns that was designed for those three tools doesn't cover this pattern. Moreover, the code changes are done to make the condition check even more specific to filter out data items.

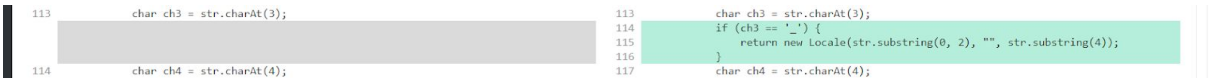
Project: Lang

Bug/Warning no: 54

Warning Reported by diffs_parsed.json:


```
{
    "Project: ": "Lang-54",
    "  Class: ": "org.apache.commons.lang.LocaleUtils",
    "  Lines: ": [
        113,
        114,
        115,
        116,
        117
    ]
},
```

Actual Code Changes in the Buggy and the fixed version of the code:



Category: Near Miss

Comments: An additional condition check is added, and the if loop performs a substring operation and returns the resultant substring. This may affect future values as they are further doing many charAt operations which throws an exception as a result. But this is not identified by any of the tools.

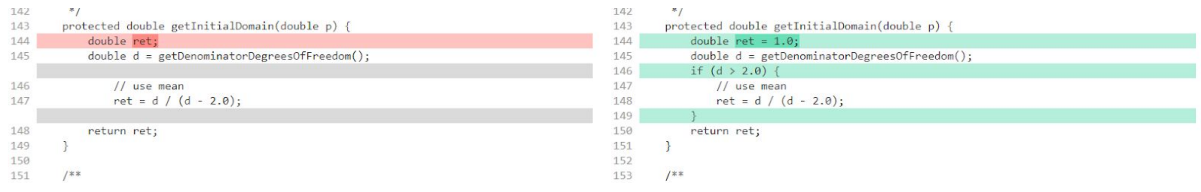
Project: Math

Bug/Warning no: 95

Warning Reported by diffs_parsed.json:

```
{
    "Project: ": "Math-95",
    "  Class: ": "org.apache.commons.math.distribution.FDistributionImpl",
    "  Lines: ": [
        144,
        145,
        146,
        147,
        148,
        149,
        150
    ]
},
```

Actual Code Changes in the Buggy and the fixed version of the code:



Category: Domain specific

Comments: None of the tools that we used identified /reported this error. This could be due to some domain specific configuration of the tool, or the bug patterns that was designed for those three tools doesn't cover this pattern. Moreover, the code changes are done to provide default value to the variable "ret" and an additional condition check has been provided for the computation.

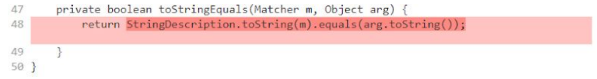
Project: Mockito

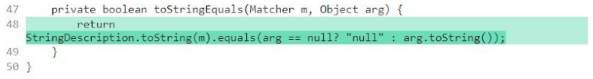
Bug/Warning no: 38

Warning Reported by diffs_parsed.json:


```
{
    "Project: ": "Mockito-38",
    "  Class: ": "org.mockito.internal.verification.argumentmatching.ArgumentMatchingTool",
    "  Lines: ": [
        48
    ]
},
```


Actual Code Changes in the Buggy and the fixed version of the code:





Category: Domain Specific

Comments: None of the tools that we used identified /reported this error. This could be due to some domain specific configuration of the tool, or the bug patterns that was designed for those three tools doesn't cover this pattern. This is not actually an error. In the return statement, an additional condition check is done to check whether the variable is null or not. If it is null, then null gets returned else the resulting string gets returned as the output.

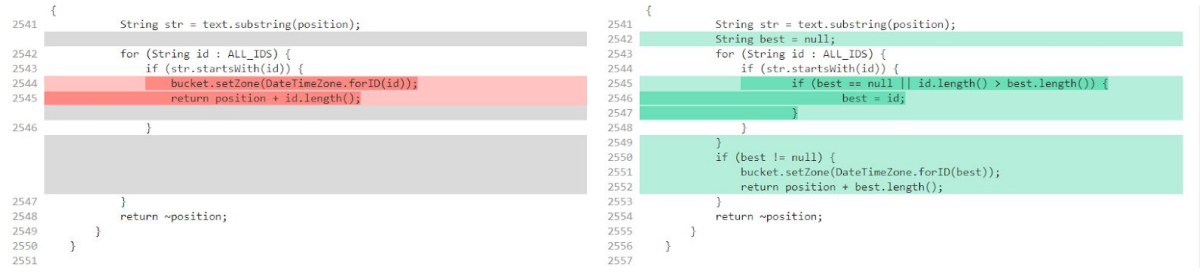
Project: Time

Bug/Warning no: 20

Warning Reported by diffs_parsed.json:


```
{
    "Project: ": "Time-20",
    "  Class: ": "org.joda.time.format.DateTimeFormatterBuilder",
    "  Lines: ": [
        2541,
        2542,
        2543,
        2544,
        2545,
        2546,
        2547,
        2548,
        2549,
        2550,
        2551,
        2552,
        2553,
        2554
    ]
},
```


Actual Code Changes in the Buggy and the fixed version of the code:



```
{
2541            String str = text.substring(position);

2542            for (String id : ALL_IDS) {
2543                if (str.startsWith(id)) {
2544                    bucket.setZone(DateTimeZone.forID(id));
2545                    return position + id.length();

2546                }
2547            }
2548            return ~position;
2549        }
2550    }
2551
```

```
{
2541            String str = text.substring(position);
2542            String best = null;
2543            for (String id : ALL_IDS) {
2544                if (str.startsWith(id)) {
2545                        if (best == null || id.length() > best.length()) {
2546                            best = id;
2547                        }
2548                }
2549            }
2550            if (best != null) {
2551                bucket.setZone(DateTimeZone.forID(best));
2552                return position + best.length();
2553            }
2554            return ~position;
2555        }
2556    }
2557
```

Category: False Positive

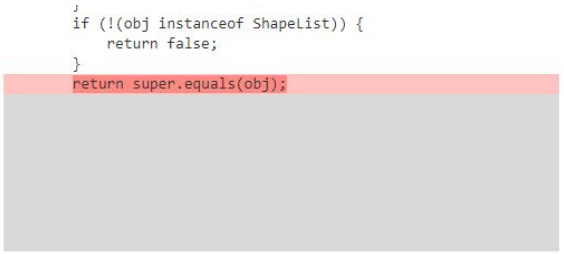
Comments: An additional string variable named “best” is defined and initialized to null. Then, the length of the string is compared to the length of the id and depending upon the comparison results, the setZone operation is performed and the value is returned. So, additional operations are being performed without any warning.

CHART-6

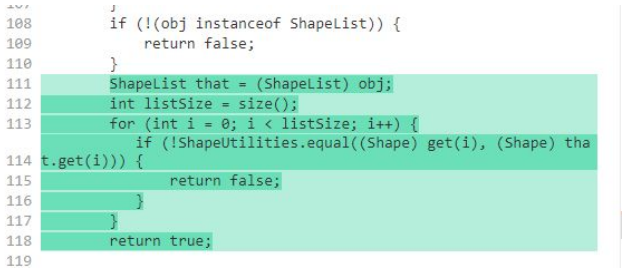
Lines for which warnings were reported by Static analysis Tools:

```
"Project: ": "Chart-6",
"  Class: ": "org.jfree.chart.util.ShapeList",
"  Lines: ": [
     111,
     112,
     113,
     114,
     115,
     116,
     117,
     118,
     119
]
```

Difference Between Buggy and Fixed files for above lines, so error was fixed in fixed version.



```
        if (!(obj instanceof ShapeList)) {
            return false;
        }
        return super.equals(obj);
```



```
108         if (!(obj instanceof ShapeList)) {
109             return false;
110         }
111         ShapeList that = (ShapeList) obj;
112         int listSize = size();
113         for (int i = 0; i < listSize; i++) {
114             if (!ShapeUtilities.equal((Shape) get(i), (Shape) tha
    t.get(i))) {
115                 return false;
116             }
117         }
118         return true;
119
```

Buggy Fixed

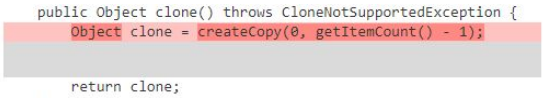
Bug was fixed but none of the tool was able to identify the real bug.

Chart-17

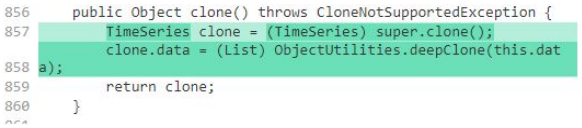
Warning - Spot Bugs


```
"    Proj": "Chart-17",
"   Class": "org.jfree.data.time.TimeSeries",
"     Cat": "BAD_PRACTICE",
"  Abbrev": "CN",
"    Type": "CN_IDIOM_NO_SUPER_CALL",
"Priority": "2",
"    Rank": "16",
"     Msg": "org.jfree.data.time.TimeSeries.clone() does not call super.clone()",
"  Method": "clone",
"   Field": "",
"   Lines": [
    [
        857,
        858,
        null
    ]
```


Code Snippets:



```
public Object clone() throws CloneNotSupportedException {
    Object clone = createCopy(0, getItemCount() - 1);

    return clone;
}
```



```
856     public Object clone() throws CloneNotSupportedException {
857         TimeSeries clone = (TimeSeries) super.clone();
858         clone.data = (List) ObjectUtilities.deepClone(this.dat
    a);
859         return clone;
860     }
```

Buggy Fixed

Math-57

Warning: Error Prone


```
{
    " Proj": "Math-57",
    "Class": "org.apache.commons.math.stat.clustering.KMeansPlusPlusClusterer",
    " Type": "warning",
    "  Cat": "NarrowingCompoundAssignment",
    "  Msg": "Compound assignments from double to int hide lossy casts",
    " Code": "                    sum += d * d;",
    " Mark": "                        ^",
    " Line": 180
},
```

```
        // the nearest center that has already been chosen.
        int sum = 0;
        for (int i = 0; i < pointSet.size(); i++) {
```

Buggy

```
174         // the nearest center that has already been chosen.
175         double sum = 0;
176         for (int i = 0; i < pointSet.size(); i++) {
```

Fixed

Closure-91
Warning- Spot Bugs

```
    "    Proj": "Closure-91",
    "   Class": "com.google.javascript.jscomp.CheckGlobalThis",
    "     Cat": "STYLE",
    "  Abbrev": "DLS",
    "    Type": "DLS_DEAD_LOCAL_STORE",
    "Priority": "2",
    "    Rank": "17",
    "     Msg": "Dead store to $L5 in com.google.javascript.jscomp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node,
Node)",
    "  Method": "shouldTraverse",
    "   Field": "",
    "   Lines": [
        [
            119,
            119,
            null
        ]
```

```
    // Don't traverse functions that are getting lent to a pro
totype.
    }
```

Buggy

```
114     // Don't traverse functions that are getting lent to a pro
totype.
115     Node gramps = parent.getParent();
116     if (NodeUtil.isObjectLitKey(parent, gramps)) {
117       JSDocInfo maybeLends = gramps.getJSDocInfo();
118       if (maybeLends != null &&
119           maybeLends.getLendsName() != null &&
120           maybeLends.getLendsName().endsWith(".prototype")) {
121         return false;
122       }
123     }
124   }
```

Fixed

Mockito-17

Warning- Error Prone

```
" Proj": "Mockito-17",
"Class": "org.mockito.internal.creation.MockSettingsImpl",
" Type": "warning",
"  Cat": "MissingOverride",
"  Msg": "serializable implements method in MockSettings; expected @Override",
" Code": "    public MockSettings serializable() {",
" Mark": "                       ^",
" Line": 21
```

```
" Proj": "Mockito-17",
"Class": "org.mockito.internal.creation.MockSettingsImpl",
" Type": "warning",
"  Cat": "MissingOverride",
"  Msg": "extraInterfaces implements method in MockSettings; expected @Override",
" Code": "    public MockSettings extraInterfaces(Class<?>... extraInterfaces) {",
" Mark": "                       ^",
" Line": 25
```

Code Snippets:

Bug a

```
20
21    public MockSettings serializable() {
22        return this.extraInterfaces(java.io.Serializable.class);
23    }
```

Buggy

```
20    private boolean serializable;
21
22    public MockSettings serializable() {
23        this.serializable = true;
24        return this;
25    }
```

Fixed

Bug b

```
73    public boolean isSerializable() {
74        return
extraInterfaces != null && java.util.Arrays.asList(extraInterface
s).contains(java.io.Serializable.class);
75    }
```

Buggy

```
75    public boolean isSerializable() {
76        return serializable;


77    }
```

Fixed

Mockito-33

Warning- Error Prone


```
{
    " Proj": "Mockito-33",
    "Class": "org.mockito.internal.invocation.InvocationMatcher",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "getLocation implements method in PrintableInvocation; expected @Override",
    " Code": "    public Location getLocation() {",
    " Mark": "                   ^",
    " Line": 102
},
{
    " Proj": "Mockito-33",
    "Class": "org.mockito.internal.invocation.InvocationMatcher",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "toString implements method in PrintingFriendlyInvocation; expected @Override",
    " Code": "    public String toString(PrintSettings printSettings) {",
    " Mark": "                  ^",
    " Line": 106
}
```


Code Snippets:

```
            /* Avoid unnecessary cloning */
        return m1.equals(m2);


    }
```

Buggy

```
98          if (m1.getName() != null && m1.getName().equals(m2.getNa
me())) {
99              /* Avoid unnecessary cloning */
100             Class[] params1 = m1.getParameterTypes();
101             Class[] params2 = m2.getParameterTypes();
102             if (params1.length == params2.length) {
103                 for (int i = 0; i < params1.length; i++) {
104                     if (params1[i] != params2[i])
105                         return false;
106                 }
107                 return true;
108             }
109         }
110         return false;
111     }
```

Fixed

Lang-51

Warning: Error Prone

```
        " Proj": "Lang-51",
        "Class": "org.apache.commons.lang.BooleanUtils",
        " Type": "error",
        "  Cat": "FallThrough",
        "  Msg": "Execution may fall through from the previous case; add a `// fall through` comment before this
was deliberate",
        " Code": "            case 4: {",
        " Mark": "            ^",
        " Line": 683
    `
```

Code Snippet

```
681             }
682         }
```

Buggy

```
681             }
682             return false;
683         }
```

Fixed

Lang-48

No warning reported

Code Snippet:

```
            // The simple case, not an array, just test the element
            isEquals = lhs.equals(rhs);
```

Buggy

```
380         if (lhs instanceof java.math.BigDecimal) {
381             isEquals = (((java.math.BigDecimal)lhs).compareTo(rhs) == 0);
382         } else {
383             // The simple case, not an array, just test the element
384             isEquals = lhs.equals(rhs);
385         }
```

Fixed

Lang-7

Warning Infer

```
"      Proj": "Lang-36",
"     Class": "org.apache.commons.lang3.math.NumberUtils",
" Bug_Class": "PROVER",
"      Kind": "ERROR",
"  Bug_Type": "NULL_DEREFERENCE",
"       Msg": "object `f` last assigned on line 518 could be null and is dereferenced at line 519.",
"  Severity": "HIGH",
"Visibility": "user",
"     Lines": [
    448,
    449,
    452,
    484,
    518,
    455,
    519,
    487,
    489,
    491,
    492,
    462,
    495,
    465,
    498,
    499,
    500,
    469,
    470,
    472
```


Code Snippet:

```
        }
        if (str.startsWith("--")) {
            return null;
        }
        if (str.startsWith("0x") || str.startsWith("-0x") || st
```

Buggy

```
451         }
452         if (str.startsWith("0x") || str.startsWith("-0x") || st
```

Fixed

Lang-60

Warning:- spot bugs


```
{
        "    Proj": "Lang-60",
        "   Class": "org.apache.commons.lang.text.StrBuilder",
        "     Cat": "BAD_PRACTICE",
        "  Abbrev": "CN",
        "    Type": "CN_IDIOM",
        "Priority": "2",
        "    Rank": "16",
        "     Msg": "Class org.apache.commons.lang.text.StrBuilder implements Cloneable but does not define or use clone
method",
        "  Method": "",
        "   Field": "",
        "   Lines": [
            [
                98,
                2203,
                null
```


Code Snippet

```
1729        char[] thisBuf = buffer;
1730        for (int i = startIndex; i < thisBuf.length; i++) {
1731            if (thisBuf[i] == ch) {
```

Buggy Code

```
1729        char[] thisBuf = buffer;
1730        for (int i = startIndex; i < size; i++) {
1731            if (thisBuf[i] == ch) {
```

Fixed Code

Time-6
Warning- Error Prone


```
{
    " Proj": "Time-6",
    "Class": "org.joda.time.chrono.GJChronology",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "getDifference overrides method in CutoverField; expected @Override",
    " Code": "        public int getDifference(long minuendInstant, long subtrahendInstant) {",
    " Mark": "                             ^",
    " Line": 1017
}
```


Code Snippet

```
        } else {
            cutoverInstant = gregorianCutover.toInstant();
        }
```

Buggy Code

```
194         } else {
195             cutoverInstant = gregorianCutover.toInstant();
196             LocalDate cutoverDate = new LocalDate(cutoverInstant.getMillis(), GregorianChronology.getInstance(zone));
197             if (cutoverDate.getYear() <= 0) {
198                 throw new IllegalArgumentException("Cutover too early. Must be on or after 0001-01-01.");
199             }
200         }
```

Fixed Code

Math-14

No Warning- Spotbugs Removed Warning

```
266     private RealMatrix squareRoot(RealMatrix m) {
267             final EigenDecomposition dec = new EigenDecomposition(m);
268             return dec.getSquareRoot();
```

Buggy Code

```
266     private RealMatrix squareRoot(RealMatrix m) {
267         if (m instanceof DiagonalMatrix) {
268             final int dim = m.getRowDimension();
269             final RealMatrix sqrtM = new DiagonalMatrix(dim);
270             for (int i = 0; i < dim; i++) {
271                 sqrtM.setEntry(i, i, FastMath.sqrt(m.getEntry(i, i)));
272             }
273             return sqrtM;
274         } else {
275             final EigenDecomposition dec = new EigenDecomposition(m);
276             return dec.getSquareRoot();
277         }
```

Fixed Code

Detailed Report of Code Review (Manual Analysis)

Project: Chart

Bug/warning no.: 21

Warning Reported by Error-Prone:

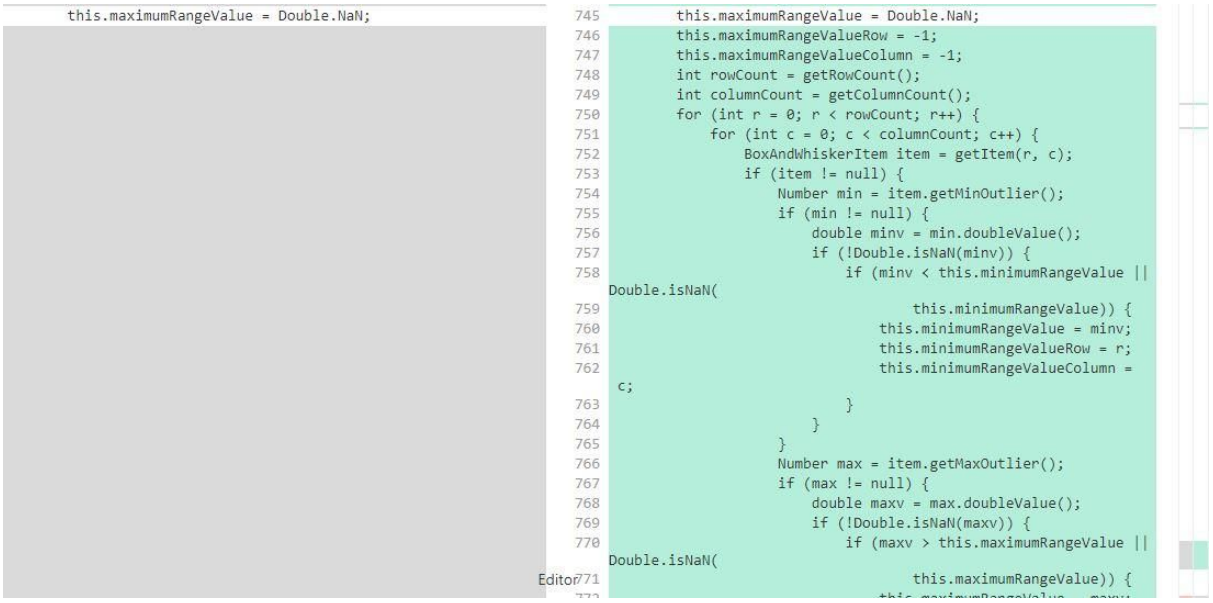

```
[
    {
        " Proj": "Chart-21",
        "Class": "org.jfree.data.statistics.DefaultBoxAndWhiskerCategoryDataset",
        " Type": "warning",
        "  Cat": "EqualsHashCode",
        "  Msg": "Classes that override equals should also override hashCode.",
        " Code": "    public boolean equals(Object obj) {",
        " Mark": "                   ^",
        " Line": 751
    },
    {
        " Proj": "Chart-21",
        "Class": "org.jfree.data.statistics.DefaultBoxAndWhiskerCategoryDataset",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "equals overrides method in Object; expected @Override",
        " Code": "    public boolean equals(Object obj) {",
        " Mark": "                   ^",
        " Line": 751
    },
    {
        " Proj": "Chart-21",
        "Class": "org.jfree.data.statistics.DefaultBoxAndWhiskerCategoryDataset",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "clone implements method in PublicCloneable; expected @Override",
        " Code": "    public Object clone() throws CloneNotSupportedException {",
        " Mark": "                   ^",
        " Line": 770
    },
```


Methodology: Diff based warning

Warning Reported by Spotbugs

```
{
    "    Proj": "Chart-21",
    "   Class": "org.jfree.data.statistics.DefaultBoxAndWhiskerCategoryDataset",
    "     Cat": "BAD_PRACTICE",
    "  Abbrev": "HE",
    "    Type": "HE_EQUALS_USE_HASHCODE",
    "Priority": "1",
    "    Rank": "14",
    "     Msg": "org.jfree.data.statistics.DefaultBoxAndWhiskerCategoryDataset defines equals and uses Object.hashCode()",
    "  Method": "equals",
    "   Field": "",
    "   Lines": [
        [
            752,
            760,
            null
        ]
    ]
},
```


Comparison of buggy and fixed versions of code :



```
this.maximumRangeValue = Double.NaN;
```

```
745         this.maximumRangeValue = Double.NaN;
746         this.maximumRangeValueRow = -1;
747         this.maximumRangeValueColumn = -1;
748         int rowCount = getRowCount();
749         int columnCount = getColumnCount();
750         for (int r = 0; r < rowCount; r++) {
751             for (int c = 0; c < columnCount; c++) {
752                 BoxAndWhiskerItem item = getItem(r, c);
753                 if (item != null) {
754                     Number min = item.getMinOutlier();
755                     if (min != null) {
756                         double minv = min.doubleValue();
757                         if (!Double.isNaN(minv)) {
758                             if (minv < this.minimumRangeValue ||
Double.isNaN(
759                                     this.minimumRangeValue)) {
760                                 this.minimumRangeValue = minv;
761                                 this.minimumRangeValueRow = r;
762                                 this.minimumRangeValueColumn =
 c;
763                             }
764                         }
765                     }
766                     Number max = item.getMaxOutlier();
767                     if (max != null) {
768                         double maxv = max.doubleValue();
769                         if (!Double.isNaN(maxv)) {
770                             if (maxv > this.maximumRangeValue ||
Double.isNaN(
Editor771                                     this.maximumRangeValue)) {
```

Comment: Error Prone two types of warnings Missing override in two places of code and EqualsHashcode.the changes made are totally irrelevant to the bug detected.

Category: False Positive

Project: Closure

Bug/warning no.: 33

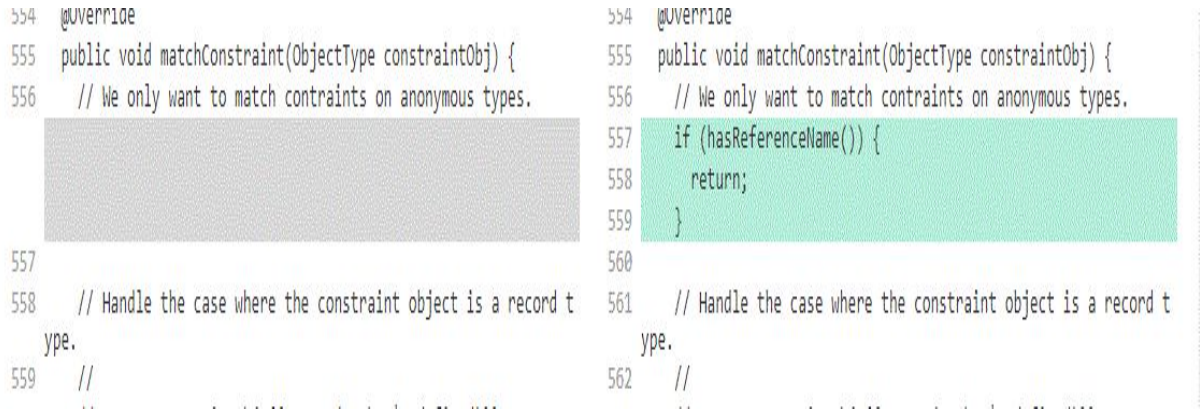
Result of Diff Parsed Json file

```
{
    "Project: ": "Closure-33",
    "  Class: ": "com.google.javascript.rhino.jstype.PrototypeObjectType",
    "  Lines: ": [
        556,
        557,
        558,
        559,
        560
    ]
},
```


Warning shown by Error Prone Parsed Json File:


```
},
{
    " Proj": "Closure-33",
    "Class": "com.googlescript.rhino.jstype.PrototypeObjectType",
    " Type": "warning",
    "  Cat": "ReferenceEquality",
    "  Msg": "Comparison using reference equality instead of value equality",
    " Code": "    return propertyType != nativePropertyType;",
    " Mark": "                          ^",
    " Line": 326
},
{
    " Proj": "Closure-33",
    "Class": "com.googlescript.rhino.jstype.PrototypeObjectType",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "setOwnerFunction overrides method in ObjectType; expected @Override",
    " Code": "  void setOwnerFunction(FunctionType type) {",
    " Mark": "       ^",
    " Line": 515
},
```


Comparison of buggy and fixed versions of code :



```
554  @Override
555  public void matchConstraint(ObjectType constraintObj) {
556    // We only want to match contraints on anonymous types.
557
558    // Handle the case where the constraint object is a record t
     ype.
559    //
```

```
554  @Override
555  public void matchConstraint(ObjectType constraintObj) {
556    // We only want to match contraints on anonymous types.
557    if (hasReferenceName()) {
558      return;
559    }
560
561    // Handle the case where the constraint object is a record t
     ype.
562    //
```

Comment : An if block was added and no changes made related to the bug

Category :Mismatch

Project: Lang

Bug/warning no.: 16

Warning Reported by Error-Prone

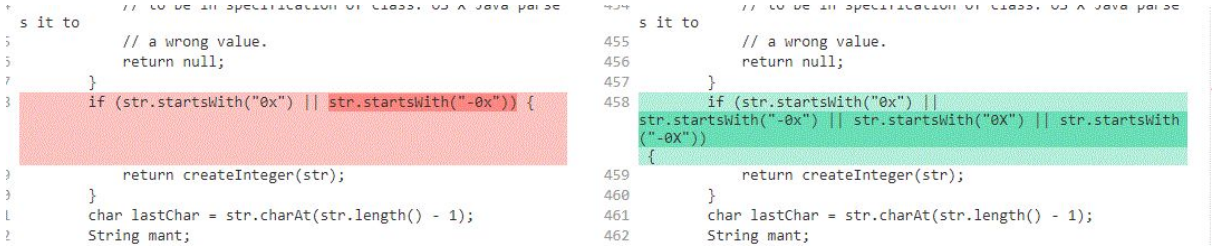
```
{
    "      Proj": "Lang-16",
    "     Class": "org.apache.commons.lang3.math.NumberUtils",
    " Bug_Class": "PROVER",
    "      Kind": "ERROR",
    "  Bug_Type": "NULL_DEREFERENCE",
    "       Msg": "object `f` last assigned on line 517 could be null and is dereferenced at line 518.",
    "  Severity": "HIGH",
    "Visibility": "user",
    "     Lines": [
        448,
        480,
        451,
        517,
        518,
        486,
        488,
        458,
        490,
        491,
        461,
        494,
        465,
        466,
        497,
        468,
        498,
        499,
        444,
        445
    ],
    " Procedure": "Number NumberUtils.createNumber(String)"
```



```
{
    "      Proj": "Lang-16",
    "     Class": "org.apache.commons.lang3.math.NumberUtils",
    " Bug_Class": "PROVER",
    "      Kind": "ERROR",
    "  Bug_Type": "NULL_DEREFERENCE",
    "       Msg": "object `d` last assigned on line 531 could be null and is dereferenced at line 532.",
    "  Severity": "HIGH",
    "Visibility": "user",
    "     Lines": [
        448,
        480,
        451,
        486,
        488,
        458,
        490,
        491,
        461,
        494,
        465,
        466,
        497,
        532,
        468,
        498,
        499,
        531,
        444,
        445
    ],
    " Procedure": "Number NumberUtils.createNumber(String)"
},
```


Methodology: Differential Based Warning

Comparison of buggy and fixed versions of code :



```
            // to be in specification of class. 03 x java parse
s it to
            // a wrong value.
            return null;
        }
        if (str.startsWith("0x") || str.startsWith("-0x")) {
            return createInteger(str);
        }
        char lastChar = str.charAt(str.length() - 1);
        String mant;
```

```
            // to be in specification of class. 03 x java parse
s it to
            // a wrong value.
            return null;
        }
        if (str.startsWith("0x") ||
str.startsWith("-0x") || str.startsWith("0X") || str.startsWith
("-0X"))
 {
            return createInteger(str);
        }
        char lastChar = str.charAt(str.length() - 1);
        String mant;
```

Comment:The bug is Null_Deference and the changes made are in a different LOC and in a' if condition'.

Category: Mismatch

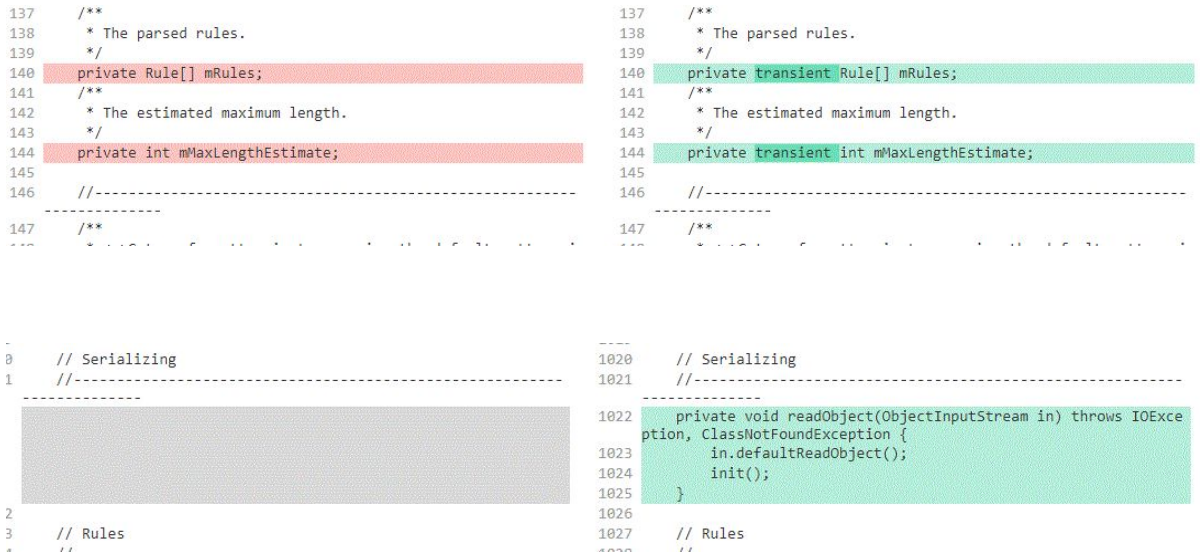
Bug/warning no.: 56

Warning Reported by Spotbugs


```
},
{
    "    Proj": "Lang-56",
    "   Class": "org.apache.commons.lang.time.FastDateFormat",
    "     Cat": "BAD_PRACTICE",
    "  Abbrev": "Se",
    "    Type": "SE_BAD_FIELD",
    "Priority": "2",
    "    Rank": "16",
    "     Msg": "Class org.apache.commons.lang.time.FastDateFormat defines non-transient non-serializable instance field mRules",
    "  Method": "",
    "   Field": "mRules",
    "   Lines": []
},
```


Methodology: Removed or Fixed Warning

Comparison of Fixed and Buggy version of code:



```
137     /**
138      * The parsed rules.
139      */
140     private Rule[] mRules;
141     /**
142      * The estimated maximum length.
143      */
144     private int mMaxLengthEstimate;
145
146     //-------------------------------------------------------------
147     /**
```

```
137     /**
138      * The parsed rules.
139      */
140     private transient Rule[] mRules;
141     /**
142      * The estimated maximum length.
143      */
144     private transient int mMaxLengthEstimate;
145
146     //-------------------------------------------------------------
147     /**
```

```
0     // Serializing
1     //-------------------------------------------------------------
2
3     // Rules
```

```
1020     // Serializing
1021     //-------------------------------------------------------------
1022     private void readObject(ObjectInputStream in) throws IOException, ClassNotFoundException {
1023         in.defaultReadObject();
1024         init();
1025     }
1026
1027     // Rules
```

Comment: Transient keyword was added and a loc was added to make it serializable, this addresses the bug completely and resolves it.

Category: Fullmatch

Bug/warning no.: 59

Warning Reported by Spot-bugs


```
},
{
    "    Proj": "Lang-59",
    "   Class": "org.apache.commons.lang.text.StrBuilder",
    "     Cat": "BAD_PRACTICE",
    "  Abbrev": "CN",
    "    Type": "CN_IDIOM",
    "Priority": "2",
    "    Rank": "16",
    "     Msg": "Class org.apache.commons.lang.text.StrBuilder implements Cloneable but does not define or use clone method",
    "  Method": "",
    "   Field": "",
    "   Lines": [
        [
            98,
            2203,
            null
        ]
    ]
},
```


Methodology: Differential based warning

Comparison of Fixed and Buggy version of code:

```
879         if (width > 0) {
880             ensureCapacity(size + width);
881             String str = (obj == null ? getNullText() : obj.toS
    tring());
882             int strLen = str.length();
883             if (strLen >= width) {
884                 str.getChars(0, strLen, buffer, size);
885             } else {
886                 int padLen = width - strLen;
887                 str.getChars(0, strLen, buffer, size);
888                 for (int i = 0; i < padLen; i++) {
889                     buffer[size + strLen + i] = padChar;
890                 }
891             }
```

```
879         if (width > 0) {
880             ensureCapacity(size + width);
881             String str = (obj == null ? getNullText() : obj.toS
    tring());
882             int strLen = str.length();
883             if (strLen >= width) {
884                 str.getChars(0, width, buffer, size);
885             } else {
886                 int padLen = width - strLen;
887                 str.getChars(0, strLen, buffer, size);
888                 for (int i = 0; i < padLen; i++) {
889                     buffer[size + strLen + i] = padChar;
890                 }
891             }
```

Comment: The warning category is BAD_Practice i.e. Strbuilder implements cloneable but was not defined anywhere and the change is made only in the parameter i.e. strlen is changed to width.

Category: Mismatch

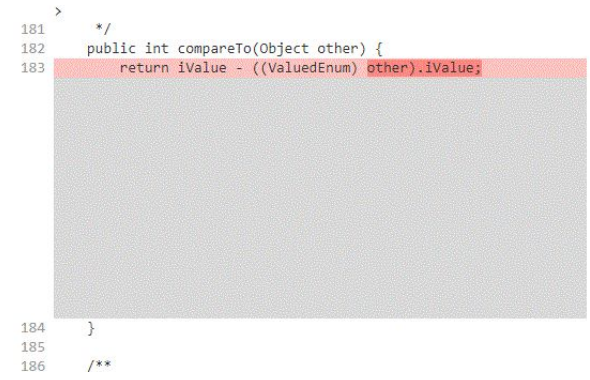
Bug/warning no.: 64

Warning Reported by Error-Prone:

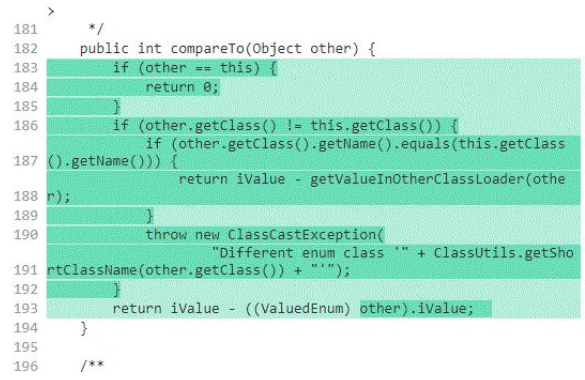

```
{
    " Proj": "Lang-64",
    "Class": "org.apache.commons.lang.enums.ValuedEnum",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "compareTo overrides method in Enum; expected @Override",
    " Code": "    public int compareTo(Object other) {",
    " Mark": "                ^",
    " Line": 182
},
```


Methodology: Diff-based Warning

Comparison of buggy and fixed versions of code:



```
     */
    public int compareTo(Object other) {
        return iValue - ((ValuedEnum) other).iValue;
    }

    /**
     * @param other  the object to determine the value for
     * @return the value
     */
            // ignore - should never happen
            // ignore - should never happen
            // ignore - should never happen
```



```
     */
    public int compareTo(Object other) {
        if (other == this) {
            return 0;
        }
        if (other.getClass() != this.getClass()) {
            if (other.getClass().getName().equals(this.getClass
().getName())) {
                return iValue - getValueInOtherClassLoader(othe
r);
            }
            throw new ClassCastException(
                    "Different enum class '" + ClassUtils.getSho
rtClassName(other.getClass()) + "'");
        }
        return iValue - ((ValuedEnum) other).iValue;
    }

    /**
     * @param other  the object to determine the value for
     * @return the value
     */
    private int getValueInOtherClassLoader(Object other) {
        try {
            Method mth = other.getClass().getMethod("getValue",
null);
            Integer value = (Integer) mth.invoke(other, null);
            return value.intValue();
        } catch (NoSuchMethodException e) {
            // ignore - should never happen
        } catch (IllegalAccessException e) {
            // ignore - should never happen
        } catch (InvocationTargetException e) {
            // ignore - should never happen
        }
        throw new IllegalStateException("This should not happe
n");
    }
```

Comment:

The warning was Missing override and there was addition of if else block and try catch throw block in the code.

Category: Mismatch

Project: Math

Bug/warning no.: 50

Warning Reported by Spot-bugs:

```
{
    "    Proj": "Math-50",
    "   Class": "org.apache.commons.math.analysis.solvers.BaseSecantSolver",
    "     Cat": "STYLE",
    "  Abbrev": "FE",
    "    Type": "FE_FLOATING_POINT_EQUALITY",
    "Priority": "1",
    "    Rank": "15",
    "     Msg": "Test for floating point equality in org.apache.commons.math.analysis.solvers.BaseSecantSolver.doSolve()",
    "  Method": "doSolve",
    "   Field": "",
    "   Lines": [
        [
            187,
            187,
            null
        ]
    ]
},
```


Methodology: Diff- based warning

Warning Reported by Spot-bugs:


```
{
    "    Proj": "Math-50",
    "   Class": "org.apache.commons.math.analysis.solvers.BaseSecantSolver",
    "     Cat": "STYLE",
    "  Abbrev": "FE",
    "    Type": "FE_FLOATING_POINT_EQUALITY",
    "Priority": "1",
    "    Rank": "15",
    "     Msg": "Test for floating point equality in org.apache.commons.math.analysis.solvers.BaseSecantSolver.doSolve()",
    "  Method": "doSolve",
    "   Field": "",
    "   Lines": [
        [
            187,
            187,
            null
        ]
    ]
},
```


Methodology: Removed/Fixed Warning

Comparison of buggy and fixed version:

```
184                 break;
185             case REGULA_FALSI:
186                 // Nothing.
187                 if (x == x1) {
188                     x0 = 0.5 * (x0 + x1 - FastMath.max(rtol
    * FastMath.abs(x1), atol));
189                     f0 = computeObjectiveValue(x0);
190                 }
191                 break;
192             default:
193                 // Should never happen.
```

```
184                 break;
185             case REGULA_FALSI:
186                 // Nothing.
187                 break;
188             default:
189                 // Should never happen.
```

Comment: The Warning is FE_Floating_point_Equality and instead of solving the warning an if block as been removed.

Category: Mismatch

Bug/warning no.: 17

Warning Reported by error-prone:


```
},
{
    " Proj": "Math-17",
    "Class": "org.apache.commons.math3.dfp.Dfp",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "multiply implements method in FieldElement; expected @Override",
    " Code": "    public Dfp multiply(final int x) {",
    " Mark": "                ^",
    " Line": 1602
},
```


Methodology: Differential Based Warning

Comparison of buggy and fixed versions:

```
     * @param x multiplicand
     * @return product of this and x
     */
    public Dfp multiply(final int x) {

            return multiplyFast(x);



    }

    /** Multiply this by a single digit 0&lt;=x&lt;radix.
     * There are speed advantages in this special case.
     * @param x multiplicand
```

```
1599     * @param x multiplicand
1600     * @return product of this and x
1601     */
1602    public Dfp multiply(final int x) {
1603        if (x >= 0 && x < RADIX) {
1604            return multiplyFast(x);
1605        } else {
1606            return multiply(newInstance(x));
1607        }
1608    }
1609
1610    /** Multiply this by a single digit 0&lt;=x&lt;radix.
1611     * There are speed advantages in this special case.
1612     * @param x multiplicand
```

Comment: The warning is missing override by error-prone, an" if else" block is added in LOC.

Category: False Positive

Bug/warning no.: 83

Warning Reported by Infer:


```
    {
        "      Proj": "Math-83",
        "     Class": "org.apache.commons.math.optimization.linear.SimplexTableau",
        " Bug_Class": "PROVER",
        "      Kind": "ERROR",
        "  Bug_Type": "NULL_DEREFERENCE",
        "       Msg": "object returned by `getBasicRow(this,(getArtificialVariableOffset()+artificialVar))` could be null and is
dereferenced at line 249.",
        "  Severity": "HIGH",
        "Visibility": "user",
        "     Lines": [
            248,
            249,
            247
        ],
        " Procedure": "void SimplexTableau.initialize()"
    }
]
```


Methodology: Removed or fixed warning

Comparison of Buggy and Fixed versions of code:

```
289     */
290    private Integer getBasicRow(final int col, boolean ignoreObj
    ectiveRows) {
291        Integer row = null;
292        int start = getNumObjectiveFunctions();
293        for (int i = start; i < getHeight(); i++) {
294            if (MathUtils.equals(getEntry(i, col), 1.0, epsilon)
    && (row == null)) {
295                row = i;
296            } else if (!MathUtils.equals(getEntry(i, col), 0.0,
```

```
289     */
290    private Integer getBasicRow(final int col, boolean ignoreObj
    ectiveRows) {
291        Integer row = null;
292        int start =
    ignoreObjectiveRows ? getNumObjectiveFunctions() : 0;
293        for (int i = start; i < getHeight(); i++) {
294            if (MathUtils.equals(getEntry(i, col), 1.0, epsilon)
    && (row == null)) {
295                row = i;
296            } else if (!MathUtils.equals(getEntry(i, col), 0.0,
```

```
340     double[] coefficients = new double[getOriginalNumDecisionV
ariables()];
341     Integer negativeVarBasicRow =
getBasicRow(getNegativeDecisionVariableOffset());
342     double mostNegative = negativeVarBasicRow == null ? 0 : ge
tEntry(negativeVarBasicRow, getRhsOffset());
343     Set<Integer> basicRows = new HashSet<Integer>();
344     for (int i = 0; i < coefficients.length; i++) {
345         Integer basicRow =
getBasicRow(getNumObjectiveFunctions() + i);
346         if (basicRows.contains(basicRow)) {
347             // if multiple variables can take a given value
348             // then we choose the first and set the rest equal
to 0
```

```
340     double[] coefficients = new double[getOriginalNumDecisionV
ariables()];
341     Integer negativeVarBasicRow =
getBasicRowForSolution(getNegativeDecisionVariableOffset());
342     double mostNegative = negativeVarBasicRow == null ? 0 : ge
tEntry(negativeVarBasicRow, getRhsOffset());
343     Set<Integer> basicRows = new HashSet<Integer>();
344     for (int i = 0; i < coefficients.length; i++) {
345         Integer basicRow =
getBasicRowForSolution(getNumObjectiveFunctions() + i);
346         if (basicRows.contains(basicRow)) {
347             // if multiple variables can take a given value
348             // then we choose the first and set the rest equal
to 0
```

Comment: The warning is null_deference, the change was made in the start variable and not in the row variable, which was returned as null.

Category: Mismatch

Bug/warning no.: 91

Warning Reported by Spot bugs:


```
{
    "    Proj": "Math-91",
    "   Class": "org.apache.commons.math.fraction.Fraction",
    "     Cat": "BAD_PRACTICE",
    "  Abbrev": "Co",
    "    Type": "CO_COMPARETO_INCORRECT_FLOATING",
    "Priority": "2",
    "    Rank": "16",
    "     Msg": "org.apache.commons.math.fraction.Fraction.compareTo(Fraction) incorrectly handles double value",
    "  Method": "compareTo",
    "   Field": "",
    "   Lines": [
        [
            261,
            261,
            null
        ]
    ]
},
```


Methodology: Removed or fixed warning

Comparison of buggy and fixed versions of code:

```
253      * Compares this object to another based on size.
254      * @param object the object to compare to
255      * @return -1 if this is less than <tt>object</tt>, +1 if th
    is is greater
256      *         than <tt>object</tt>, 0 if they are equal.
257      */
258     public int compareTo(Fraction object) {
259         double nOd = doubleValue();
260         double dOn = object.doubleValue();
261         return (nOd < dOn) ? -1 : ((nOd > dOn) ? +1 : 0);
262     }
263
```

```
253      * Compares this object to another based on size.
254      * @param object the object to compare to
255      * @return -1 if this is less than <tt>object</tt>, +1 if th
    is is greater
256      *         than <tt>object</tt>, 0 if they are equal.
257      */
258     public int compareTo(Fraction object) {
259         long nOd = ((long) numerator) * object.denominator;
260         long dOn = ((long) denominator) * object.numerator;
261         return (nOd < dOn) ? -1 : ((nOd > dOn) ? +1 : 0);
262     }
263
```

Comment: The warning is Bad_Practice and the type is Co_Compareto_Incorrect_Floating that is fixed by changing the keyword.

Category: Full Match

Project : Mockito

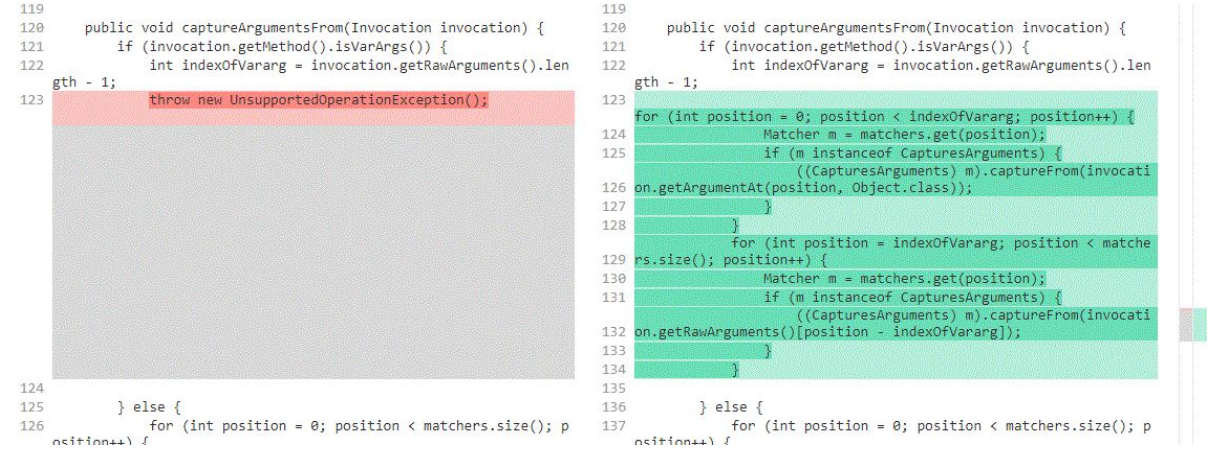
Bug/warning no.: 1

Warning Reported by Spot bugs:


```
{
    "    Proj": "Mockito-1",
    "   Class": "org.mockito.internal.invocation.InvocationMatcher",
    "     Cat": "STYLE",
    "  Abbrev": "DLS",
    "    Type": "DLS_DEAD_LOCAL_STORE",
    "Priority": "2",
    "    Rank": "17",
    "     Msg": "Dead store to $L2 in org.mockito.internal.invocation.InvocationMatcher.captureArgumentsFrom(Invocation)",
    "  Method": "captureArgumentsFrom",
    "   Field": "",
    "   Lines": [
        [
            122,
            122,
            null
        ]
    ]
},
```


Methodology: Removed / Fixed warning

Comparison of buggy and fixed versions:



```
119
120     public void captureArgumentsFrom(Invocation invocation) {
121         if (invocation.getMethod().isVarArgs()) {
122             int indexOfVararg = invocation.getRawArguments().len
gth - 1;
123             throw new UnsupportedOperationException();
124
125         } else {
126             for (int position = 0; position < matchers.size(); p
osition++) {
```

```
119
120     public void captureArgumentsFrom(Invocation invocation) {
121         if (invocation.getMethod().isVarArgs()) {
122             int indexOfVararg = invocation.getRawArguments().len
gth - 1;
123
for (int position = 0; position < indexOfVararg; position++) {
124                 Matcher m = matchers.get(position);
125                 if (m instanceof CapturesArguments) {
                        ((CapturesArguments) m).captureFrom(invocati
126 on.getArgumentAt(position, Object.class));
127                 }
128             }
            for (int position = indexOfVararg; position < matche
129 rs.size(); position++) {
130                 Matcher m = matchers.get(position);
131                 if (m instanceof CapturesArguments) {
                        ((CapturesArguments) m).captureFrom(invocati
132 on.getRawArguments()[position - indexOfVararg]);
133                 }
134             }
135
136         } else {
137             for (int position = 0; position < matchers.size(); p
osition++) {
```

Comment: The warning is DLS_Dead_Local_store, the throw block has been removed and variable assigned has been used.

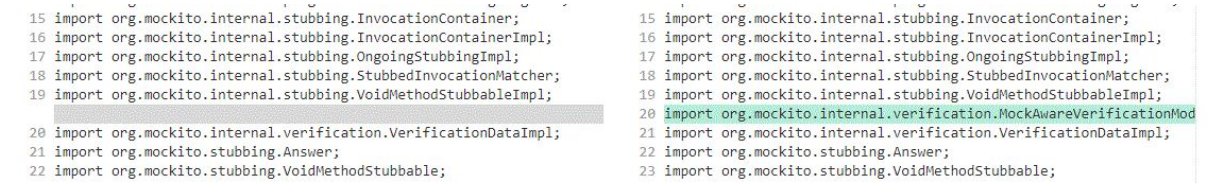
Category: Full match

Bug/warning no. 14:

Warning reported by Error prone_Parsed_json:

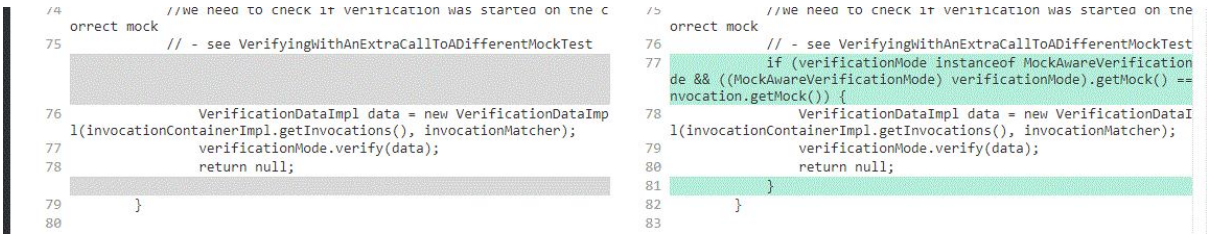

```
        " Proj": "Mockito-14",
        "Class": "org.mockito.internal.MockHandler",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "handle implements method in MockitoInvocationHandler; expected @Override",
        " Code": "    public Object handle(Invocation invocation) throws Throwable {",
        " Mark": "                  ^",
        " Line": 57
    },
    {
        " Proj": "Mockito-14",
        "Class": "org.mockito.internal.MockHandler",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "voidMethodStubbable implements method in MockHandlerInterface; expected @Override",
        " Code": "    public VoidMethodStubbable<T> voidMethodStubbable(T mock) {",
        " Mark": "                                  ^",
        " Line": 103
    },
```


Comparison of buggy and fixed version of code:



```
15 import org.mockito.internal.stubbing.InvocationContainer;
16 import org.mockito.internal.stubbing.InvocationContainerImpl;
17 import org.mockito.internal.stubbing.OngoingStubbingImpl;
18 import org.mockito.internal.stubbing.StubbedInvocationMatcher;
19 import org.mockito.internal.stubbing.VoidMethodStubbableImpl;
20 import org.mockito.internal.verification.VerificationDataImpl;
21 import org.mockito.stubbing.Answer;
22 import org.mockito.stubbing.VoidMethodStubbable;
```

```
15 import org.mockito.internal.stubbing.InvocationContainer;
16 import org.mockito.internal.stubbing.InvocationContainerImpl;
17 import org.mockito.internal.stubbing.OngoingStubbingImpl;
18 import org.mockito.internal.stubbing.StubbedInvocationMatcher;
19 import org.mockito.internal.stubbing.VoidMethodStubbableImpl;
20 import org.mockito.internal.verification.MockAwareVerificationMod
21 import org.mockito.internal.verification.VerificationDataImpl;
22 import org.mockito.stubbing.Answer;
23 import org.mockito.stubbing.VoidMethodStubbable;
```



```
            //We need to check if verification was started on the c
orrect mock
            // - see VerifyingWithAnExtraCallToADifferentMockTest
                VerificationDataImpl data = new VerificationDataImp
l(invocationContainerImpl.getInvocations(), invocationMatcher);
                verificationMode.verify(data);
                return null;
        }
```

```
            //We need to check if verification was started on the
orrect mock
            // - see VerifyingWithAnExtraCallToADifferentMockTest
            if (verificationMode instanceof MockAwareVerification
de && ((MockAwareVerificationMode) verificationMode).getMock() ==
nvocation.getMock()) {
                VerificationDataImpl data = new VerificationDataI
l(invocationContainerImpl.getInvocations(), invocationMatcher);
                verificationMode.verify(data);
                return null;
            }
        }
```

Comment: In this a library was imported and an if block was added nut the bug was not fixed

Category: Mismatch

Project: Time

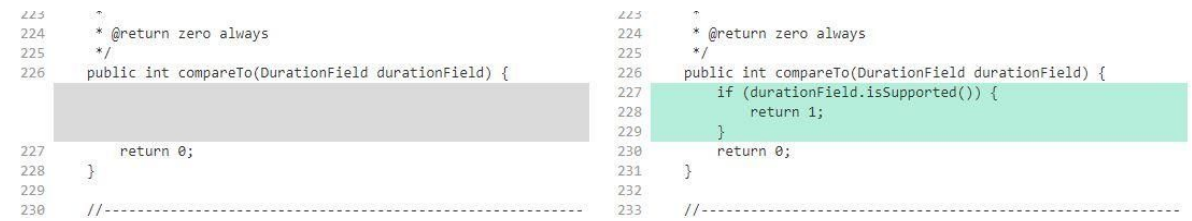
Bug/Warning number: 2

Warning shown by Error-Prone:

```
{
    " Proj": "Time-2",
    "Class": "org.joda.time.field.UnsupportedDurationField",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "compareTo implements method in Comparable; expected @Override",
    " Code": "    public int compareTo(DurationField durationField) {",
    " Mark": "                ^",
    " Line": 226
},
```

Methodology: Diff based warning

Comparison of buggy and fixed code:



```
     * @return zero always
     */
    public int compareTo(DurationField durationField) {

        return 0;
    }

    //-----------------------------------------------------------
```

```
     * @return zero always
     */
    public int compareTo(DurationField durationField) {
        if (durationField.isSupported()) {
            return 1;
        }
        return 0;
    }

    //-----------------------------------------------------------
```

Comment: The warning is missing override and an if block was added.

Category: False Positive

Project: Chart

Bug /Warning no: 23

Warning reported by error-prone:

```
{
    " Proj": "Chart-23",
    "Class": "org.jfree.chart.renderer.category.MinMaxCategoryRenderer",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "paintIcon implements method in Icon; expected @Override",
    " Code": "          public void paintIcon(Component c, Graphics g, int x, int y) {",
    " Mark": "                       ^",
    " Line": 452
},
```

Methodology: Differential Based Warnings

Actual code changes in both Buggy and Fixed version of the code:

```
425   /**
426    * Tests this instance for equality with an arbitrary object.  The icon fields
427    * are NOT included in the test, so this implementation is a little weak.
428    *
429    * @param obj  the object (<code>null</code> permitted).
430    *
431    * @return A boolean.
432    *
433    * @since 1.0.7
434    */
435
436   /**
```

```
425   /**
426    * Tests this instance for equality with an arbitrary object.  The icon fields
427    * are NOT included in the test, so this implementation is a little weak.
428    *
429    * @param obj  the object (<code>null</code> permitted).
430    *
431    * @return A boolean.
432    *
433    * @since 1.0.7
434    */
435   public boolean equals(Object obj) {
436       if (obj == this) {
437           return true;
438       }
439       if (!(obj instanceof MinMaxCategoryRenderer)) {
440           return false;
441       }
442       MinMaxCategoryRenderer that = (MinMaxCategoryRenderer) obj;
443       if (this.plotLines != that.plotLines) {
444           return false;
445       }
446       if (!PaintUtilities.equal(this.groupPaint, that.groupPaint)) {
447           return false;
448       }
449       if (!this.groupStroke.equals(that.groupStroke)) {
450           return false;
451       }
452       return super.equals(obj);
453   }
454
455   /**
```

Category: Mismatch

Comment:

The warning was missing override, but the code changes were something irrelevant.

Warning reported by Spot Bugs:

```
{
      "    Proj": "Chart-23",
      "   Class": "",
      "     Cat": "",
      "  Abbrev": "",
      "    Type": "NO_WARNING",
      "Priority": "",
      "    Rank": "",
      "     Msg": "",
      "  Method": "",
      "   Field": "",
      "   Lines": []
},
```


Methodology: Removed/Fixed warnings

Category: Near misses

Comment: Spot Bugs has similar implementation of bug (ES, EQ, HE) but was unable to catch this warning.

Project: Closure

Bug /Warning no: 49

Warning reported by Spot Bugs:


```
    {
        "    Proj": "Closure-49",
        "   Class": "com.google.javascript.jscomp.MakeDeclaredNamesUnique",
        "     Cat": "STYLE",
        "  Abbrev": "SF",
        "    Type": "SF_SWITCH_NO_DEFAULT",
        "Priority": "2",
        "    Rank": "19",
        "     Msg": "Switch statement found in com.google.javascript.jscomp.MakeDeclaredNamesUnique.shouldTraverse(NodeTraversal, Node,
Node) where default case is missing",
        "  Method": "shouldTraverse",
        "   Field": "",
        "   Lines": [
            [
                116,
                147,
                null
            ]
        ]
    },


    {
        "    Proj": "Closure-49",
        "   Class": "com.google.javascript.jscomp.MakeDeclaredNamesUnique",
        "     Cat": "STYLE",
        "  Abbrev": "SF",
        "    Type": "SF_SWITCH_NO_DEFAULT",
        "Priority": "2",
        "    Rank": "19",
        "     Msg": "Switch statement found in com.google.javascript.jscomp.MakeDeclaredNamesUnique.visit(NodeTraversal, Node, Node) where
default case is missing",
        "  Method": "visit",
        "   Field": "",
        "   Lines": [
            [
                157,
                183,
                null
            ]
        ]
    },
```

Methodology: Differential based warnings

Actual code changes in both Buggy and Fixed version of the code (First warning):

```
@Override
public boolean shouldTraverse(NodeTraversal t, Node n, Node parent) {

  switch (n.getType()) {
    case Token.FUNCTION:
      {
        // Add recursive function name, if needed.
        // NOTE: "enterScope" is called after we need to pick up this name.
        Renamer renamer = nameStack.peek().forChildScope();

        // If needed, add the function recursive name.
        String name = n.getFirstChild().getString();
        if (name != null && !name.isEmpty() && parent != null
            && !NodeUtil.isFunctionDeclaration(n)) {
          renamer.addDeclaredName(name);
        }


        // Add the function parameters


        // Add the function body declarations


        nameStack.push(renamer);
      }
      break;

    case Token.CATCH:
      {
        Renamer renamer = nameStack.peek().forChildScope();

        String name = n.getFirstChild().getString();
        renamer.addDeclaredName(name);

        nameStack.push(renamer);
      }
      break;
  }

  return true;
}
```

```
@Override
public boolean shouldTraverse(NodeTraversal t, Node n, Node parent) {

  switch (n.getType()) {
    case Token.FUNCTION:
      {
        // Add recursive function name, if needed.
        // NOTE: "enterScope" is called after we need to pick up this name.
        Renamer renamer = nameStack.peek().forChildScope();

        // If needed, add the function recursive name.
        String name = n.getFirstChild().getString();
        if (name != null && !name.isEmpty() && parent != null
            && !NodeUtil.isFunctionDeclaration(n)) {
          renamer.addDeclaredName(name);
        }

        nameStack.push(renamer);
      }
      break;

    case Token.LP: {
        Renamer renamer = nameStack.peek().forChildScope();

        // Add the function parameters
        for (Node c = n.getFirstChild(); c != null; c = c.getNext()) {
          String name = c.getString();
          renamer.addDeclaredName(name);
        }

        // Add the function body declarations
        Node functionBody = n.getNext();
        findDeclaredNames(functionBody, null, renamer);

        nameStack.push(renamer);
      }
      break;

    case Token.CATCH:
      {
        Renamer renamer = nameStack.peek().forChildScope();

        String name = n.getFirstChild().getString();
        renamer.addDeclaredName(name);

        nameStack.push(renamer);
      }
      break;
  }

  return true;
}
```

Category: Mismatch

Comment: For the first warning in method “Should Traverse” there was no code changes made to address the default case missing warning, instead they have added codes that are unrelated to this warning (inside case).

Actual code changes in both Buggy and Fixed version of the code (Second warning):

```
@Override
public void visit(NodeTraversal t, Node n, Node parent) {
  switch (n.getType()) {
    case Token.NAME:
      String newName = getReplacementName(n.getString());
      if (newName != null) {
        Renamer renamer = nameStack.peek();
        if (renamer.stripConstIfReplaced()) {
          // TODO(johnlenz): Do we need to do anything about the javadoc?
          n.removeProp(Node.IS_CONSTANT_NAME);
        }
        n.setString(newName);
        t.getCompiler().reportCodeChange();
      }
      break;

    case Token.FUNCTION:
      // Remove the function body scope

      // Remove function recursive name (if any).
      nameStack.pop();
      break;


      // Note: The parameters and function body variables live in the
      // same scope, we introduce the scope when in the "shouldTraverse"
      // visit of LP, but remove it when when we exit the function above.


    case Token.CATCH:
      // Remove catch except name from the stack of names.
      nameStack.pop();
      break;
  }
}
```

```
@Override
public void visit(NodeTraversal t, Node n, Node parent) {
  switch (n.getType()) {
    case Token.NAME:
      String newName = getReplacementName(n.getString());
      if (newName != null) {
        Renamer renamer = nameStack.peek();
        if (renamer.stripConstIfReplaced()) {
          // TODO(johnlenz): Do we need to do anything about the javadoc?
          n.removeProp(Node.IS_CONSTANT_NAME);
        }
        n.setString(newName);
        t.getCompiler().reportCodeChange();
      }
      break;

    case Token.FUNCTION:
      // Remove the function body scope
      nameStack.pop();
      // Remove function recursive name (if any).
      nameStack.pop();
      break;

    case Token.LP:
      // Note: The parameters and function body variables live in the
      // same scope, we introduce the scope when in the "shouldTraverse"
      // visit of LP, but remove it when when we exit the function above.
      break;

    case Token.CATCH:
      // Remove catch except name from the stack of names.
      nameStack.pop();
      break;
  }
}
```

Category: Mismatch

Comment: For the second warning in method “Visit” there was no code changes made to address the default case missing warning, instead they have added codes that are unrelated to this warning (inside case).

Project: Lang

Bug/ Warning no: 1

Warning reported by Infer:

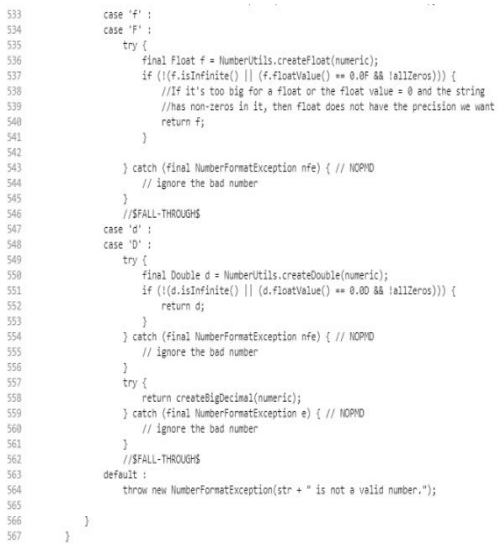
```
{
    "      Proj": "Lang-1",
    "     Class": "org.apache.commons.lang3.math.NumberUtils",
    " Bug_Class": "PROVER",
    "      Kind": "ERROR",
    "  Bug_Type": "NULL_DEREFERENCE",
    "       Msg": "object `d` last assigned on line 550 could be null and is dereferenced at line 551.",
    "  Severity": "HIGH",
    "Visibility": "user",
    "     Lines": [
        513,
        516,
        517,
        518,
        550,
        551,
        450,
        451,
        454,
        458,
        459,
        460,
        461,
        462,
        466,
        476,
        480,
        481,
        485,
        486,
        499,
        505,
        507,
        509,
        510
    ],
    " Procedure": "Number NumberUtils.createNumber(String)"
},
```



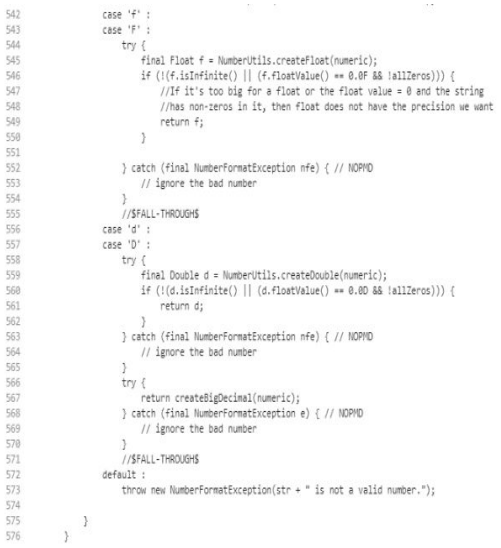
```
{
    "      Proj": "Lang-1",
    "     Class": "org.apache.commons.lang3.math.NumberUtils",
    " Bug_Class": "PROVER",
    "      Kind": "ERROR",
    "  Bug_Type": "NULL_DEREFERENCE",
    "       Msg": "object `f` last assigned on line 536 could be null and is dereferenced at line 537.",
    "  Severity": "HIGH",
    "Visibility": "user",
    "     Lines": [
        513,
        516,
        517,
        518,
        536,
        537,
        450,
        451,
        454,
        458,
        459,
        460,
        461,
        462,
        466,
        476,
        480,
        481,
        485,
        486,
        499,
        505,
        507,
        509,
        510
    ],
    " Procedure": "Number NumberUtils.createNumber(String)"
```


Methodology: Differential Based Warning

Actual Code changes on Buggy and Fixed version of the code (First warning):



```
            case 'f' :
            case 'F' :
                try {
                    final Float f = NumberUtils.createFloat(numeric);
                    if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
                        //If it's too big for a float or the float value = 0 and the string
                        //has non-zeros in it, then float does not have the precision we want
                        return f;
                    }

                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            case 'd' :
            case 'D' :
                try {
                    final Double d = NumberUtils.createDouble(numeric);
                    if (!(d.isInfinite() || (d.floatValue() == 0.0D && !allZeros))) {
                        return d;
                    }
                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                try {
                    return createBigDecimal(numeric);
                } catch (final NumberFormatException e) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            default :
                throw new NumberFormatException(str + " is not a valid number.");

        }
    }
```

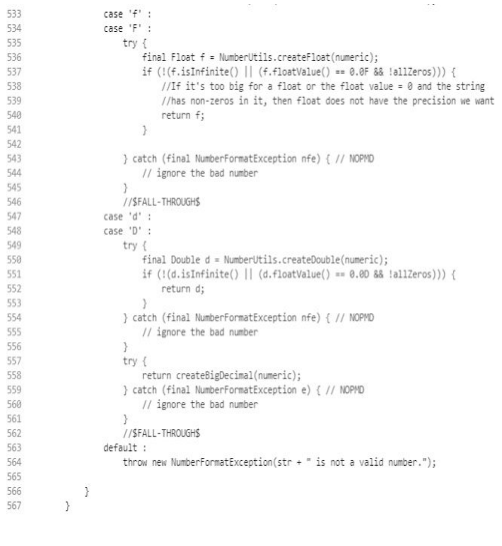


```
            case 'f' :
            case 'F' :
                try {
                    final Float f = NumberUtils.createFloat(numeric);
                    if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
                        //If it's too big for a float or the float value = 0 and the string
                        //has non-zeros in it, then float does not have the precision we want
                        return f;
                    }

                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            case 'd' :
            case 'D' :
                try {
                    final Double d = NumberUtils.createDouble(numeric);
                    if (!(d.isInfinite() || (d.floatValue() == 0.0D && !allZeros))) {
                        return d;
                    }
                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                try {
                    return createBigDecimal(numeric);
                } catch (final NumberFormatException e) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            default :
                throw new NumberFormatException(str + " is not a valid number.");

        }
    }
```

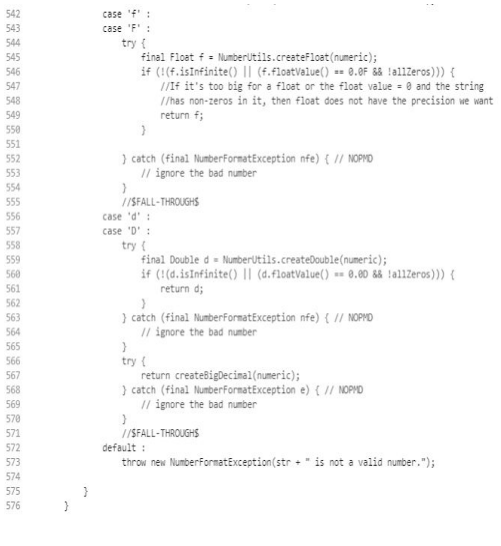
Category: Mismatch

Comment: The warning was null deference for an object f. But the developer didn't address this error, instead he made changes to some other part of the code. He is using the null object for this condition check in if statement. It is not just a warning but an error itself. It has got high severity and he needs to address it.

Actual Code changes on Buggy and Fixed version of the code (Second warning):



```
            case 'f' :
            case 'F' :
                try {
                    final Float f = NumberUtils.createFloat(numeric);
                    if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
                        //If it's too big for a float or the float value = 0 and the string
                        //has non-zeros in it, then float does not have the precision we want
                        return f;
                    }

                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            case 'd' :
            case 'D' :
                try {
                    final Double d = NumberUtils.createDouble(numeric);
                    if (!(d.isInfinite() || (d.floatValue() == 0.0D && !allZeros))) {
                        return d;
                    }
                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                try {
                    return createBigDecimal(numeric);
                } catch (final NumberFormatException e) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            default :
                throw new NumberFormatException(str + " is not a valid number.");

        }
    }
```



```
            case 'f' :
            case 'F' :
                try {
                    final Float f = NumberUtils.createFloat(numeric);
                    if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
                        //If it's too big for a float or the float value = 0 and the string
                        //has non-zeros in it, then float does not have the precision we want
                        return f;
                    }

                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            case 'd' :
            case 'D' :
                try {
                    final Double d = NumberUtils.createDouble(numeric);
                    if (!(d.isInfinite() || (d.floatValue() == 0.0D && !allZeros))) {
                        return d;
                    }
                } catch (final NumberFormatException nfe) { // NOPMD
                    // ignore the bad number
                }
                try {
                    return createBigDecimal(numeric);
                } catch (final NumberFormatException e) { // NOPMD
                    // ignore the bad number
                }
                //$FALL-THROUGH$
            default :
                throw new NumberFormatException(str + " is not a valid number.");

        }
    }
```

Category: Mismatch

Comment: The warning was null deference for an object d. But the developer didn't address this error, instead he made changes to some other part of the code. He is using the null object for this condition check in if statement. It is not just a warning but an error itself. It has got high severity and he needs to address it.

Project: Lang

Bug/ Warning no: 3

Warning reported by Spot Bugs:


```
{
     "    Proj": "Lang-3",
     "   Class": "org.apache.commons.lang3.math.NumberUtils",
     "     Cat": "STYLE",
     "  Abbrev": "DLS",
     "    Type": "DLS_DEAD_LOCAL_STORE",
     "Priority": "2",
     "    Rank": "17",
     "     Msg": "Dead store to $L9 in org.apache.commons.lang3.math.NumberUtils.createNumber(String)",
     "  Method": "createNumber",
     "   Field": "",
     "   Lines": [
          [
              497,
              497,
              null
          ]
     ]
},
```


Methodology: Removed/Fixed warnings

Actual code changes in buggy and fixed version of the code:

```
450   public static Number createNumber(final String str) throws NumberFormatException {
451       if (str == null) {
452           return null;
453       }
454       if (StringUtils.isBlank(str)) {
455           throw new NumberFormatException("A blank string is not a valid number");
456       }
457       // Need to deal with all possible hex prefixes here
458       final String[] hex_prefixes = {"0x", "0X", "-0x", "-0X", "#", "-#"};
459       int pfxLen = 0;
460       for(final String pfx : hex_prefixes) {
461           if (str.startsWith(pfx)) {
462               pfxLen += pfx.length();
463               break;
464           }
465       }
466       if (pfxLen > 0) { // we have a hex number
467           final int hexDigits = str.length() - pfxLen;
468           if (hexDigits > 16) { // too many for Long
469               return createBigInteger(str);
470           }
471           if (hexDigits > 8) { // too many for an int
472               return createLong(str);
473           }
474           return createInteger(str);
475       }
476       final char lastChar = str.charAt(str.length() - 1);
477       String mant;
478       String dec;
479       String exp;
480       final int decPos = str.indexOf('.');
481       final int expPos = str.indexOf('e') + str.indexOf('E') + 1; // assumes both not present
482       // if both e and E are present, this is caught by the checks on expPos (which prevent IOOBE)
483       // and the parsing which will detect if e or E appear in a number due to using the wrong offset
484
485       int numDecimals = 0; // Check required precision (LANG-693)
486       if (decPos > -1) { // there is a decimal point
487
488           if (expPos > -1) { // there is an exponent
489               if (expPos < decPos || expPos > str.length()) { // prevents double exponent causing IOOBE
490                   throw new NumberFormatException(str + " is not a valid number.");
491               }
492               dec = str.substring(decPos + 1, expPos);
493           } else {
494               dec = str.substring(decPos + 1);
495           }
496           mant = str.substring(0, decPos);
497           numDecimals = dec.length(); // gets number of digits past the decimal to ensure no loss of precision for floatin
    g point numbers.
498       } else {
499           if (expPos > -1) {
```

```
450   public static Number createNumber(final String str) throws NumberFormatException {
451       if (str == null) {
452           return null;
453       }
454       if (StringUtils.isBlank(str)) {
455           throw new NumberFormatException("A blank string is not a valid number");
456       }
457       // Need to deal with all possible hex prefixes here
458       final String[] hex_prefixes = {"0x", "0X", "-0x", "-0X", "#", "-#"};
459       int pfxLen = 0;
460       for(final String pfx : hex_prefixes) {
461           if (str.startsWith(pfx)) {
462               pfxLen += pfx.length();
463               break;
464           }
465       }
466       if (pfxLen > 0) { // we have a hex number
467           final int hexDigits = str.length() - pfxLen;
468           if (hexDigits > 16) { // too many for Long
469               return createBigInteger(str);
470           }
471           if (hexDigits > 8) { // too many for an int
472               return createLong(str);
473           }
474           return createInteger(str);
475       }
476       final char lastChar = str.charAt(str.length() - 1);
477       String mant;
478       String dec;
479       String exp;
480       final int decPos = str.indexOf('.');
481       final int expPos = str.indexOf('e') + str.indexOf('E') + 1; // assumes both not present
482       // if both e and E are present, this is caught by the checks on expPos (which prevent IOOBE)
483       // and the parsing which will detect if e or E appear in a number due to using the wrong offset
484
485       int numDecimals = 0; // Check required precision (LANG-693)
486       if (decPos > -1) { // there is a decimal point
487
488           if (expPos > -1) { // there is an exponent
489               if (expPos < decPos || expPos > str.length()) { // prevents double exponent causing IOOBE
490                   throw new NumberFormatException(str + " is not a valid number.");
491               }
492               dec = str.substring(decPos + 1, expPos);
493           } else {
494               dec = str.substring(decPos + 1);
495           }
496           mant = str.substring(0, decPos);
497           numDecimals = dec.length(); // gets number of digits past the decimal to ensure no loss of precision for floatin
    g point numbers.
498       } else {
499           if (expPos > -1) {
```

```
592       try {
593           final Float f = createFloat(str);
594           if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
595               return f;
596           }
597       } catch (final NumberFormatException nfe) { // NOPMD
598           // ignore the bad number
599       }
600       try {
601           final Double d = createDouble(str);
602           if (!(d.isInfinite() || (d.doubleValue() == 0.0D && !allZeros))) {
603               return d;
604           }
605       } catch (final NumberFormatException nfe) { // NOPMD
606           // ignore the bad number
607       }
608
609       return createBigDecimal(str);
610   }
```

```
592       try {
593           if(numDecimals <= 7){// If number has 7 or fewer digits past the decimal point then make it a float
594               final Float f = createFloat(str);
595               if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
596                   return f;
597               }
598           }
599       } catch (final NumberFormatException nfe) { // NOPMD
600           // ignore the bad number
601       }
602       try {
603           if(numDecimals <= 16){// If number has between 8 and 16 digits past the decimal point then make it a double
604               final Double d = createDouble(str);
605               if (!(d.isInfinite() || (d.doubleValue() == 0.0D && !allZeros))) {
606                   return d;
607               }
608           }
609       } catch (final NumberFormatException nfe) { // NOPMD
610           // ignore the bad number
611       }
612
613       return createBigDecimal(str);
614   }
```

Category: Full Match

Comment:

The warning was DLS dead local store and the lines of code were added to address this exception in try block.

Project: Lang

Bug/ Warning no: 10

Warning reported by diff-pared. json report:


```
{
     "Project: ": "Lang-10",
     "  Class: ": "org.apache.commons.lang3.time.FastDateParser",
     "  Lines: ": [
          304,
          307,
          308,
          309,
          310,
          311,
          312,
          313,
          314
     ]
},
```


Actual Code changes in buggy and fixed version of the code:

```
303   private static StringBuilder escapeRegex(StringBuilder regex, String value, boolean unquote) {
304       boolean wasWhite= false;
305       for(int i= 0; i<value.length(); ++i) {
306           char c= value.charAt(i);
307           if(Character.isWhitespace(c)) {
308               if(!wasWhite) {
309                   wasWhite= true;
310                   regex.append("\\s*+");
311               }
312               continue;
313           }
314           wasWhite= false;
315           switch(c) {
316           case '\'':
317               if(unquote) {
318                   if(++i==value.length()) {
319                       return regex;
320                   }
321                   c= value.charAt(i);
322               }
323               break;
324           case '?':
325           case '[':
326           case ']':
327           case '(':
328           case ')':
329           case '{':
330           case '}':
331           case '\\':
332           case '|':
333           case '*':
334           case '+':
335           case '^':
336           case '$':
```

```
303   private static StringBuilder escapeRegex(StringBuilder regex, String value, boolean unquote) {
304       for(int i= 0; i<value.length(); ++i) {
305           char c= value.charAt(i);
306           switch(c) {
307           case '\'':
308               if(unquote) {
309                   if(++i==value.length()) {
310                       return regex;
311                   }
312                   c= value.charAt(i);
313               }
314               break;
315           case '?':
316           case '[':
317           case ']':
318           case '(':
319           case ')':
320           case '{':
321           case '}':
322           case '\\':
323           case '|':
324           case '*':
325           case '+':
326           case '^':
327           case '$':
```

Category: Domain specific

Comment:

None of the tools that we used identified /reported this error. This could be due to some domain specific configuration of the tool, or the bug patterns that was designed for those three tools doesn't cover this pattern. Moreover, the code changes are for handling the whitespace, this won't be a real bug or error.

Project: Lang

Bug/ Warning no: 58

Warning reported by Spot Bugs:

```
    {
        "    Proj": "Lang-58",
        "   Class": "org.apache.commons.lang.math.NumberUtils",
        "     Cat": "STYLE",
        "  Abbrev": "SF",
        "    Type": "SF_SWITCH_NO_DEFAULT",
        "Priority": "2",
        "    Rank": "19",
        "     Msg": "Switch statement found in org.apache.commons.lang.math.NumberUtils.createNumber(String) where default case is
missing",
        "  Method": "createNumber",
        "   Field": "",
        "   Lines": [
            [
                449,
                491,
                null
            ]
        ]
    },
```


Methodology: Differential based warning

Actual Code changes in Buggy and Fixed version of the code:

```
449         switch (lastChar) {
450             case 'l' :
451             case 'L' :
452                 if (dec == null
453                     && exp == null
454                     && isDigits(numeric.substring(1))
455                     && (numeric.charAt(0) == '-' || Character.isDigit(numeric.charAt(0)))) {
456                     try {
457                         return createLong(numeric);
458                     } catch (NumberFormatException nfe) {
459                         //Too big for a long
460                     }
461                     return createBigInteger(numeric);
462
463                 }
464                 throw new NumberFormatException(str + " is not a valid number.");
465             case 'f' :
466             case 'F' :
467                 try {
468                     Float f = NumberUtils.createFloat(numeric);
469                     if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
470                         //If it's too big for a float or the float value = 0 and the string
471                         //has non-zeros in it, then float does not have the precision we want
472                         return f;
473                     }
474
475                 } catch (NumberFormatException nfe) {
476                     // ignore the bad number
477                 }
478                 //Fall through
479             case 'd' :
480             case 'D' :
481                 try {
482                     Double d = NumberUtils.createDouble(numeric);
483                     if (!(d.isInfinite() || (d.floatValue() == 0.0D && !allZeros))) {
484                         return d;
485                     }
486                 } catch (NumberFormatException nfe) {
487                     // ignore the bad number
488                 }
489                 try {
490                     return createBigDecimal(numeric);
491                 } catch (NumberFormatException e) {
492                     // ignore the bad number
493                 }
494                 //Fall through
495             default :
496                 throw new NumberFormatException(str + " is not a valid number.");
497
498         }
```

```
449         switch (lastChar) {
450             case 'l' :
451             case 'L' :
452                 if (dec == null
453                     && exp == null
454                     && (numeric.charAt(0) == '-' && isDigits(numeric.substring(1)) || isDigits(numeric))) {
455
456                     try {
457                         return createLong(numeric);
458                     } catch (NumberFormatException nfe) {
459                         //Too big for a long
460                     }
461                     return createBigInteger(numeric);
462
463                 }
464                 throw new NumberFormatException(str + " is not a valid number.");
465             case 'f' :
466             case 'F' :
467                 try {
468                     Float f = NumberUtils.createFloat(numeric);
469                     if (!(f.isInfinite() || (f.floatValue() == 0.0F && !allZeros))) {
470                         //If it's too big for a float or the float value = 0 and the string
471                         //has non-zeros in it, then float does not have the precision we want
472                         return f;
473                     }
474
475                 } catch (NumberFormatException nfe) {
476                     // ignore the bad number
477                 }
478                 //Fall through
479             case 'd' :
480             case 'D' :
481                 try {
482                     Double d = NumberUtils.createDouble(numeric);
483                     if (!(d.isInfinite() || (d.floatValue() == 0.0D && !allZeros))) {
484                         return d;
485                     }
486                 } catch (NumberFormatException nfe) {
487                     // ignore the bad number
488                 }
489                 try {
490                     return createBigDecimal(numeric);
491                 } catch (NumberFormatException e) {
492                     // ignore the bad number
493                 }
494                 //Fall through
495             default :
496                 throw new NumberFormatException(str + " is not a valid number.");
497
498         }
```

Category: False Positive

Comment:

The warning was missing default case in switch condition check. But both the buggy and fixed version of code already has default case in them which throws exception. The only code changes that were made was few more condition added inside one case. It is not even warning but was falsely reported as warning or bug.

Project: Math

Bug/ Warning no: 6

Warning reported by Infer:

```
{
      "      Proj": "Math-6",
      "     Class": "",
      " Bug_Class": "",
      "      Kind": "",
      "  Bug_Type": "NO_WARNING",
      "       Msg": "",
      "  Severity": "",
      "Visibility": "",
      "     Lines": "",
      " Procedure": ""
},
```


Methodology: Removed/Fixed warning

Warning reported / catched in Diff-parsed. Json report:


```
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.nonlinear.scalar.noderiv.PowellOptimizer",
    "  Lines: ": [
        192,
        193,
        226,
        225,
        227,
        191
    ]
},
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.nonlinear.scalar.gradient.NonLinearConjugateGradientOptimizer",
    "  Lines: ": [
        275,
        276,
        277,
        214,
        215,
        216,
        217,
        221,
        222,
        223
    ]
},
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.nonlinear.scalar.noderiv.SimplexOptimizer",
    "  Lines: ": [
        158,
        175
    ]
},
```

```
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.nonlinear.vector.jacobian.GaussNewtonOptimizer",
    "  Lines: ": [
        160,
        106,
        107,
        108,
        158,
        159
    ]
},
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.nonlinear.scalar.noderiv.CMAESOptimizer",
    "  Lines: ": [
        387,
        388,
        389
    ]
},
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.BaseOptimizer",
    "  Lines: ": [
        51
    ]
},
{
    "Project: ": "Math-6",
    "  Class: ": "org.apache.commons.math3.optim.nonlinear.vector.jacobian.LevenbergMarquardtOptimizer",
    "  Lines: ": [
        322,
        323,
        324,
        325,
        489
    ]
},
```


Category: False positive

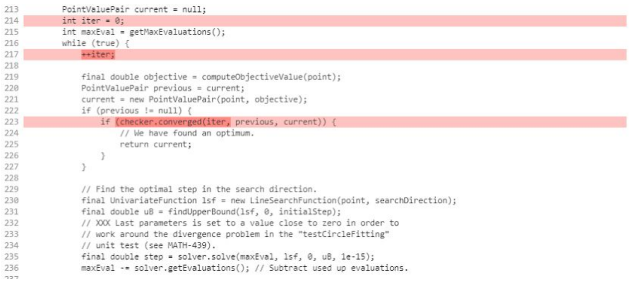
Actual code of buggy and fixed version of org. apache. commons.math3.optim.nonlinear.scalar.noderiv. PowellOptimizer Class:

Comment:

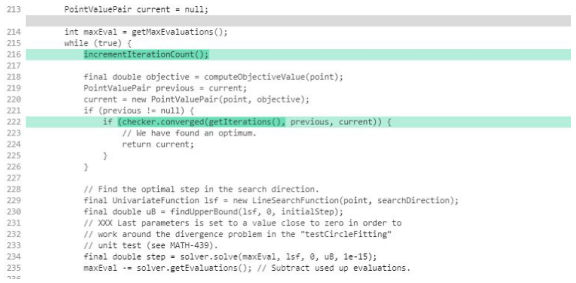
The variable used for iteration in while loop was removed, and it is done using method call. This may or may not be done intentionally. This can probably lead to error.

Category: False positive

Actual code of buggy and fixed version of org.apache.commons.math3.optim.nonlinear.scalar. gradient.NonLinearConjugateGradientOptimizer Class:



```
PointValuePair current = null;
int iter = 0;
int maxEval = getMaxEvaluations();
while (true) {
    ++iter;

    final double objective = computeObjectiveValue(point);
    PointValuePair previous = current;
    current = new PointValuePair(point, objective);
    if (previous != null) {
        if (checker.converged(iter, previous, current)) {
            // We have found an optimum.
            return current;
        }
    }

    // Find the optimal step in the search direction.
    final UnivariateFunction lsf = new LineSearchFunction(point, searchDirection);
    final double uB = findUpperBound(lsf, 0, initialStep);
    // XXX Last parameters is set to a value close to zero in order to
    // work around the divergence problem in the "testCircleFitting"
    // unit test (see MATH-439).
    final double step = solver.solve(maxEval, lsf, 0, uB, 1e-15);
    maxEval -= solver.getEvaluations(); // Subtract used up evaluations.
```



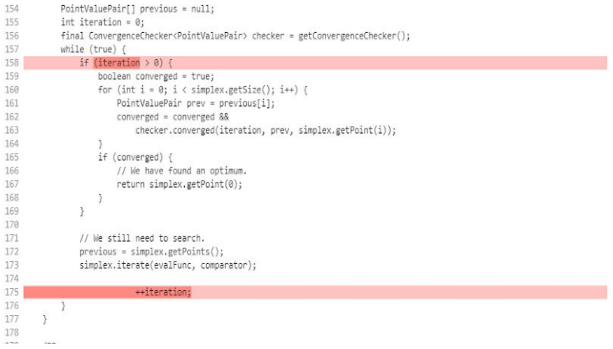
```
PointValuePair current = null;

int maxEval = getMaxEvaluations();
while (true) {
    incrementIterationCount();

    final double objective = computeObjectiveValue(point);
    PointValuePair previous = current;
    current = new PointValuePair(point, objective);
    if (previous != null) {
        if (checker.converged(getIterations(), previous, current)) {
            // We have found an optimum.
            return current;
        }
    }

    // Find the optimal step in the search direction.
    final UnivariateFunction lsf = new LineSearchFunction(point, searchDirection);
    final double uB = findUpperBound(lsf, 0, initialStep);
    // XXX Last parameters is set to a value close to zero in order to
    // work around the divergence problem in the "testCircleFitting"
    // unit test (see MATH-439).
    final double step = solver.solve(maxEval, lsf, 0, uB, 1e-15);
    maxEval -= solver.getEvaluations(); // Subtract used up evaluations.
```

Comment:

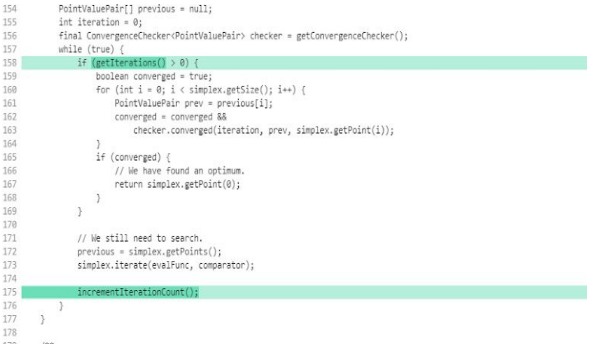
The same error happened in this class also. The variable used for iteration in while loop was removed, and it is done using method call. This may or may not be done intentionally. This can probably lead to error.

Category: False positive

Actual code of buggy and fixed version of org.apache.commons.math3.optim.nonlinear.scalar.noderiv. SimplexOptimizer Class:



```
PointValuePair[] previous = null;
int iteration = 0;
final ConvergenceChecker<PointValuePair> checker = getConvergenceChecker();
while (true) {
    if (iteration > 0) {
        boolean converged = true;
        for (int i = 0; i < simplex.getSize(); i++) {
            PointValuePair prev = previous[i];
            converged = converged &&
                checker.converged(iteration, prev, simplex.getPoint(i));
        }
        if (converged) {
            // We have found an optimum.
            return simplex.getPoint(0);
        }
    }

    // We still need to search.
    previous = simplex.getPoints();
    simplex.iterate(evalFunc, comparator);

                ++iteration;
  }
}
```



```
PointValuePair[] previous = null;
int iteration = 0;
final ConvergenceChecker<PointValuePair> checker = getConvergenceChecker();
while (true) {
    if (getIterations() > 0) {
        boolean converged = true;
        for (int i = 0; i < simplex.getSize(); i++) {
            PointValuePair prev = previous[i];
            converged = converged &&
                checker.converged(iteration, prev, simplex.getPoint(i));
        }
        if (converged) {
            // We have found an optimum.
            return simplex.getPoint(0);
        }
    }

    // We still need to search.
    previous = simplex.getPoints();
    simplex.iterate(evalFunc, comparator);

    incrementIterationCount();
  }
}
```

Comment:

The same error happened in this class also. The variable used for iteration in while loop was removed, and it is done using method call. This may or may not be done intentionally. This can probably lead to error.

Category: False positive

Actual code of buggy and fixed version of org.apache.commons.math3.optim.nonlinear.vector.jacobian. GaussNewtonOptimizer Class:

```
104         // iterate until convergence is reached
105         PointVectorValuePair current = null;
106         int iter = 0;
107         for (boolean converged = false; !converged;) {
108             ++iter;
109
110             // evaluate the objective function and its jacobian
111             PointVectorValuePair previous = current;
112             // Value of the objective function at "currentPoint".
113             final double[] currentObjective = computeObjectiveValue(currentPoint);
114             final double[] currentResiduals = computeResiduals(currentObjective);
115             final RealMatrix weightedJacobian = computeWeightedJacobian(currentPoint);
116             current = new PointVectorValuePair(currentPoint, currentObjective);
117
118             // build the linear problem
119             final double[]   b = new double[nC];
120             final double[][] a = new double[nC][nC];
121             for (int i = 0; i < nR; ++i) {
122
123                 final double[] grad   = weightedJacobian.getRow(i);
124                 final double weight   = residualsWeights[i];
125                 final double residual = currentResiduals[i];
126
127                 // compute the normal equation
128                 final double wr = weight * residual;
129                 for (int j = 0; j < nC; ++j) {
130                     b[j] += wr * grad[j];
131                 }
132
133                 // build the contribution matrix for measurement i
134                 for (int k = 0; k < nC; ++k) {
135                     double[] ak = a[k];
136                     double wgk = weight * grad[k];
137                     for (int l = 0; l < nC; ++l) {
138                         ak[l] += wgk * grad[l];
139                     }
140                 }
141             }
142
```

```
104         // iterate until convergence is reached
105         PointVectorValuePair current = null;
106
107         for (boolean converged = false; !converged;) {
108             incrementIterationCount();
109
110             // evaluate the objective function and its jacobian
111             PointVectorValuePair previous = current;
112             // Value of the objective function at "currentPoint".
113             final double[] currentObjective = computeObjectiveValue(currentPoint);
114             final double[] currentResiduals = computeResiduals(currentObjective);
115             final RealMatrix weightedJacobian = computeWeightedJacobian(currentPoint);
116             current = new PointVectorValuePair(currentPoint, currentObjective);
117
118             // build the linear problem
119             final double[]   b = new double[nC];
120             final double[][] a = new double[nC][nC];
121             for (int i = 0; i < nR; ++i) {
122
123                 final double[] grad   = weightedJacobian.getRow(i);
124                 final double weight   = residualsWeights[i];
125                 final double residual = currentResiduals[i];
126
127                 // compute the normal equation
128                 final double wr = weight * residual;
129                 for (int j = 0; j < nC; ++j) {
130                     b[j] += wr * grad[j];
131                 }
132
133                 // build the contribution matrix for measurement i
134                 for (int k = 0; k < nC; ++k) {
135                     double[] ak = a[k];
136                     double wgk = weight * grad[k];
137                     for (int l = 0; l < nC; ++l) {
138                         ak[l] += wgk * grad[l];
139                     }
140                 }
141             }
```

Comment:

The same error happened in this class also. The variable used for iteration in while loop was removed, and it is done using method call. This may or may not be done intentionally. This can probably lead to error.

Category: False positive

Actual code of buggy and fixed version of org.apache.commons.math3.optim.nonlinear.scalar.noderiv. CMAESOptimizer Class:

```
384         // -------------------- Generation Loop --------------------------------
385
386         generationLoop:
387         for (iterations = 1; iterations <= maxIterations; iterations++) {
388
389             // Generate and evaluate lambda offspring
390             final RealMatrix arz = randn1(dimension, lambda);
391             final RealMatrix arx = zeros(dimension, lambda);
392             final double[] fitness = new double[lambda];
393             // generate random offspring
394             for (int k = 0; k < lambda; k++) {
395                 RealMatrix arxk = null;
396                 for (int i = 0; i < checkFeasableCount + 1; i++) {
397                     if (diagonalOnly <= 0) {
398                         arxk = xmean.add(BD.multiply(arz.getColumnMatrix(k))
399                                          .scalarMultiply(sigma)); // m + sig * Normal(0,C)
400                     } else {
401                         arxk = xmean.add(times(diagD,arz.getColumnMatrix(k))
402                                          .scalarMultiply(sigma));
403                     }
404                     if (i >= checkFeasableCount ||
405                         fitfun.isFeasible(arxk.getColumn(0))) {
406                         break;
407                     }
408                     // regenerate random arguments for row
409                     arz.setColumn(k, randn(dimension));
410                 }
411                 copyColumn(arxk, 0, arx, k);
412                 try {
413                     fitness[k] = fitfun.value(arx.getColumn(k)); // compute fitness
414                 } catch (TooManyEvaluationsException e) {
415                     break generationLoop;
416                 }
417             }
```

```
384         // -------------------- Generation Loop --------------------------------
385
386         generationLoop:
387         for (iterations = 1; iterations <= maxIterations; iterations++) {
388             incrementIterationCount();
389
390             // Generate and evaluate lambda offspring
391             final RealMatrix arz = randn1(dimension, lambda);
392             final RealMatrix arx = zeros(dimension, lambda);
393             final double[] fitness = new double[lambda];
394             // generate random offspring
395             for (int k = 0; k < lambda; k++) {
396                 RealMatrix arxk = null;
397                 for (int i = 0; i < checkFeasableCount + 1; i++) {
398                     if (diagonalOnly <= 0) {
399                         arxk = xmean.add(BD.multiply(arz.getColumnMatrix(k))
400                                          .scalarMultiply(sigma)); // m + sig * Normal(0,C)
401                     } else {
402                         arxk = xmean.add(times(diagD,arz.getColumnMatrix(k))
403                                          .scalarMultiply(sigma));
404                     }
405                     if (i >= checkFeasableCount ||
406                         fitfun.isFeasible(arxk.getColumn(0))) {
407                         break;
408                     }
409                     // regenerate random arguments for row
410                     arz.setColumn(k, randn(dimension));
411                 }
412                 copyColumn(arxk, 0, arx, k);
413                 try {
414                     fitness[k] = fitfun.value(arx.getColumn(k)); // compute fitness
415                 } catch (TooManyEvaluationsException e) {
416                     break generationLoop;
417                 }
418             }
```

Comment:

The same error happened in this class also. The variable used for iteration in while loop was removed, and it is done using method call. This may or may not be done intentionally. This can probably lead to error.

Category: False positive

Actual code of buggy and fixed version of org.apache.commons.math3.optim.BaseOptimizer Class:

```
44      /**
45       * @param checker Convergence checker.
46       */
47      protected BaseOptimizer(ConvergenceChecker<PAIR> checker) {
48          this.checker = checker;
49
50          evaluations = new Incrementor(0, new MaxEvalCallback());
51          iterations = new Incrementor(0, new MaxIterCallback());
52      }
53
54      /**
55       * Gets the maximal number of function evaluations.
56       *
57       * @return the maximal number of function evaluations.
58       */
59      public int getMaxEvaluations() {
60          return evaluations.getMaximalCount();
61      }
62
```

```
44      /**
45       * @param checker Convergence checker.
46       */
47      protected BaseOptimizer(ConvergenceChecker<PAIR> checker) {
48          this.checker = checker;
49
50          evaluations = new Incrementor(0, new MaxEvalCallback());
51          iterations = new Incrementor(Integer.MAX_VALUE, new MaxIterCallback());
52      }
53
54      /**
55       * Gets the maximal number of function evaluations.
56       *
57       * @return the maximal number of function evaluations.
58       */
59      public int getMaxEvaluations() {
60          return evaluations.getMaximalCount();
61      }
62
```

Comment:

The changes that are made in the fixed code might be done intentionally or unintentionally. The developer has changed the parameter value to max. This may or may not end up in infinite loop. This can probably lead to error.

Category: False positive

Actual code of buggy and fixed version of org.apache.commons.math3.optim.nonlinear.vector.jacobian. LevenbergMarquardtOptimizer Class:

```
319     // Outer loop.
320     lmPar = 0;
321     boolean firstIteration = true;
322     int iter = 0;
323     final ConvergenceChecker<PointVectorValuePair> checker = getConvergenceChecker();
324     while (true) {
325         ++iter;
326         final PointVectorValuePair previous = current;
327
328         // QR decomposition of the jacobian matrix
329         qrDecomposition(computeWeightedJacobian(currentPoint));
330
331         weightedResidual = weightMatrixSqrt.operate(currentResiduals);
332         for (int i = 0; i < nR; i++) {
333             qtf[i] = weightedResidual[i];
334         }
335
336         // compute Qt.res
337         qTy(qtf);
338
```

```
319     // Outer loop.
320     lmPar = 0;
321     boolean firstIteration = true;
322     final ConvergenceChecker<PointVectorValuePair> checker = getConvergenceChecker();
323     while (true) {
324         incrementIterationCount();
325         final PointVectorValuePair previous = current;
326
327         // QR decomposition of the jacobian matrix
328         qrDecomposition(computeWeightedJacobian(currentPoint));
329
330         weightedResidual = weightMatrixSqrt.operate(currentResiduals);
331         for (int i = 0; i < nR; i++) {
332             qtf[i] = weightedResidual[i];
333         }
334
335         // compute Qt.res
336         qTy(qtf);
337
338
```

Comment:

The same error happened in this class also. The variable used for iteration in while loop was removed, and it is done using method call. This may or may not be done intentionally. This can probably lead to error.

Project: Math

Bug /Warning no: 51

Warning reported by Spot Bugs:

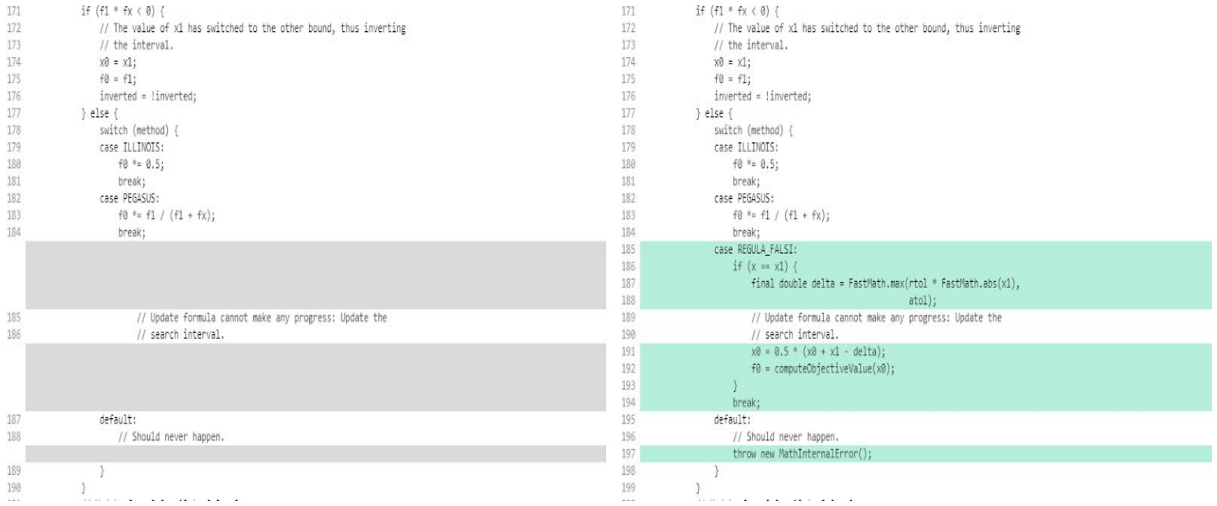

```
{
    "    Proj": "Math-51",
    "   Class": "",
    "     Cat": "",
    "  Abbrev": "",
    "    Type": "NO_WARNING",
    "Priority": "",
    "    Rank": "",
    "     Msg": "",
    "  Method": "",
    "   Field": "",
    "   Lines": []
},
```


Methodology: Removed or Fixed warning

Warning reported by diff-parsed. Json report:

```
{
    "Project: ": "Math-51",
    "  Class: ": "org.apache.commons.math.analysis.solvers.BaseSecantSolver",
    "  Lines: ": [
        192,
        193,
        194,
        195,
        196,
        197,
        198,
        184,
        185,
        186,
        187,
        188,
        189,
        190,
        191
    ]
},
```


Actual changes on buggy and fixed version of the code:



```
171         if (f1 * fx < 0) {
172             // The value of x1 has switched to the other bound, thus inverting
173             // the interval.
174             x0 = x1;
175             f0 = f1;
176             inverted = !inverted;
177         } else {
178             switch (method) {
179             case ILLINOIS:
180                 f0 *= 0.5;
181                 break;
182             case PEGASUS:
183                 f0 *= f1 / (f1 + fx);
184                 break;
185                 // Update formula cannot make any progress: Update the
186                 // search interval.
187             default:
188                 // Should never happen.
189             }
190         }
```

```
171         if (f1 * fx < 0) {
172             // The value of x1 has switched to the other bound, thus inverting
173             // the interval.
174             x0 = x1;
175             f0 = f1;
176             inverted = !inverted;
177         } else {
178             switch (method) {
179             case ILLINOIS:
180                 f0 *= 0.5;
181                 break;
182             case PEGASUS:
183                 f0 *= f1 / (f1 + fx);
184                 break;
185             case REGULA_FALSI:
186                 if (x == x1) {
187                     final double delta = FastMath.max(rtol * FastMath.abs(x1),
188                                                       atol);
189                     // Update formula cannot make any progress: Update the
190                     // search interval.
191                     x0 = 0.5 * (x0 + x1 - delta);
192                     f0 = computeObjectiveValue(x0);
193                 }
194                 break;
195             default:
196                 // Should never happen.
197                 throw new MathInternalError();
198             }
199         }
```

Category: Domain Specific

Comment:

In the fixed version of the code they have added a new case inside the switch condition and also they have added statements under default which was empty before. None of the tool identified this error, but the developer has somehow made changes which could be unintentionally fixed.

Project: Math

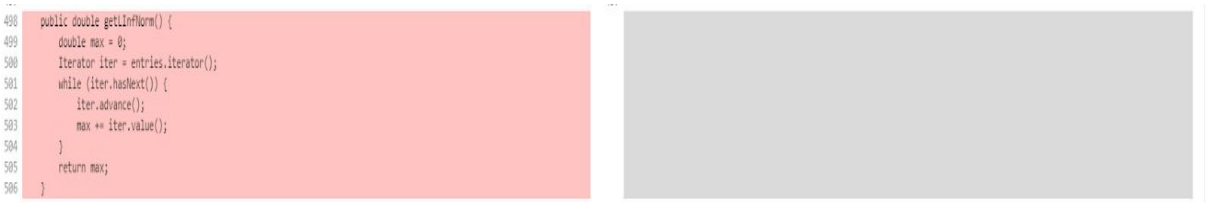
Bug /Warning no: 77

Warning reported by error-prone:

```
{
    " Proj": "Math-77",
    "Class": "org.apache.commons.math.linear.OpenMapRealVector",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "getLInfNorm overrides method in AbstractRealVector; expected @Override",
    " Code": "    public double getLInfNorm() {",
    " Mark": "                  ^",
    " Line": 498
},
```


Methodology: Differential parsed warning

Actual changes on Buggy and Fixed version of the Code:



Category: Mismatch

Comment:

The warning was missing override, but the developer has removed that entire method instead of adding override annotation.

Warning reported by Error-Prone:


```
{
    " Proj": "Math-77",
    "Class": "org.apache.commons.math.linear.OpenMapRealVector",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "getLInfNorm overrides method in AbstractRealVector; expected @Override",
    " Code": "    public double getLInfNorm() {",
    " Mark": "                  ^",
    " Line": 498
},
```


Methodology: Fixed or Removed warning

Actual changes on the buggy and fixed version of the code:

Comment:

The warning was missing override, but the developer has removed that entire method instead of adding override annotation.

Project: Math

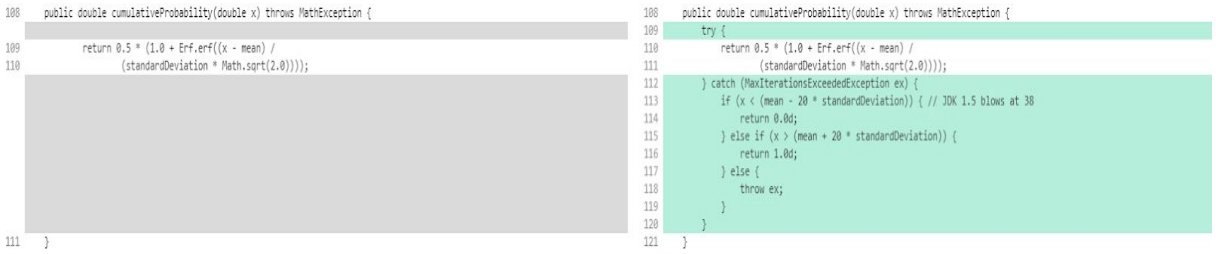
Bug /Warning no: 103

Warning reported by error-prone:

```
{
        " Proj": "Math-103",
        "Class":
"org.apache.commons.math.distribution.NormalDistributionImpl",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "cumulativeProbability implements method in
Distribution; expected @Override",
        " Code": "    public double cumulativeProbability(double
x) throws MathException {",
        " Mark": "                    ^",
        " Line": 108
    },
```


Methodology: Differential Based warning

Actual Code changes in Buggy and Fixed version of the code:



```
108   public double cumulativeProbability(double x) throws MathException {
109
110       return 0.5 * (1.0 + Erf.erf((x - mean) /
111               (standardDeviation * Math.sqrt(2.0))));
112
113
114
115
116
117
118
119
120
121   }
```

```
108   public double cumulativeProbability(double x) throws MathException {
109       try {
110           return 0.5 * (1.0 + Erf.erf((x - mean) /
111                   (standardDeviation * Math.sqrt(2.0))));
112       } catch (MaxIterationsExceededException ex) {
113           if (x < (mean - 20 * standardDeviation)) { // JDK 1.5 blows at 38
114               return 0.0d;
115           } else if (x > (mean + 20 * standardDeviation)) {
116               return 1.0d;
117           } else {
118               throw ex;
119           }
120       }
121   }
```

Category: Mismatch

Comment:

The warning was missing override, but the code changes were for addressing exception.

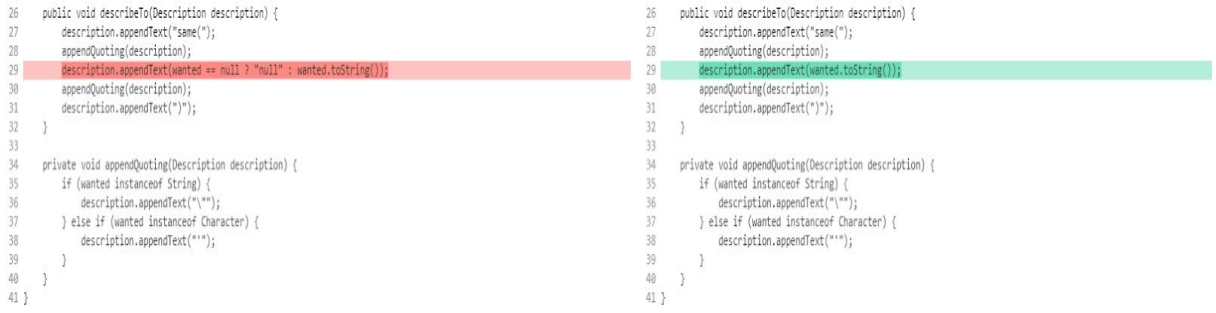
Project: Mockito

Bug /Warning no: 29

Warning reported by Diff-parsed. Json report:


```
{
        "Project: ": "Mockito-29",
        "  Class: ": "org.mockito.internal.matchers.Same",
        "  Lines: ": [
             29
        ]
},
```


Actual Code changes on Buggy and Fixed version of the Code:



```
26  public void describeTo(Description description) {
27      description.appendText("same(");
28      appendQuoting(description);
29      description.appendText(wanted == null ? "null" : wanted.toString());
30      appendQuoting(description);
31      description.appendText(")");
32  }
33
34  private void appendQuoting(Description description) {
35      if (wanted instanceof String) {
36          description.appendText("\"");
37      } else if (wanted instanceof Character) {
38          description.appendText("'");
39      }
40  }
41 }
```

```
26  public void describeTo(Description description) {
27      description.appendText("same(");
28      appendQuoting(description);
29      description.appendText(wanted.toString());
30      appendQuoting(description);
31      description.appendText(")");
32  }
33
34  private void appendQuoting(Description description) {
35      if (wanted instanceof String) {
36          description.appendText("\"");
37      } else if (wanted instanceof Character) {
38          description.appendText("'");
39      }
40  }
41 }
```

Category: Domain Specific

Comment:

In the buggy version a conditional check is used inside a method call and that is been changed in fixed version of the code. This may or may not be an error. This could be probably done unintentionally. None of the Tool identified this due to some specific cases.

Project: Mockito

Bug /Warning no: 36

Warning reported by error-prone:


```
{
        " Proj": "Mockito-36",
        "Class": "org.mockito.internal.invocation.Invocation",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "toString implements method in
PrintingFriendlyInvocation; expected @Override",
        " Code": "    public String toString(PrintSettings
printSettings) {",
        " Mark": "                    ^",
        " Line": 205
},
```


Methodology: Differential based warning

Actual Code change on Buggy and Fixed version of the Code:

```
201   public Object callRealMethod() throws Throwable {
202       return realMethod.invoke(mock, rawArguments);
203   }
204
205   public String toString(PrintSettings printSettings) {
206       return toString(argumentsToMatchers(), printSettings);
207   }
208
209   void markVerified() {
210       this.verified = true;
211   }
212
213   void markVerifiedInOrder() {
214       markVerified();
215       this.verifiedInOrder = true;
216   }
```

```
202   public Object callRealMethod() throws Throwable {
203       if (this.getMethod().getDeclaringClass().isInterface()) {
204           new Reporter().cannotCallRealMethodOnInterface();
205       }
206       return realMethod.invoke(mock, rawArguments);
207   }
208
209   public String toString(PrintSettings printSettings) {
210       return toString(argumentsToMatchers(), printSettings);
211   }
212
213   void markVerified() {
214       this.verified = true;
215   }
216
217   void markVerifiedInOrder() {
218       markVerified();
219       this.verifiedInOrder = true;
220   }
```

Category: Mismatch

Comment:

The warning was missing override on toString method, but the changes were made on another part of the code.

Project: Time

Bug /Warning no: 3

Warning reported by error-prone:

```
{
    " Proj": "Time-3",
    "Class": "org.joda.time.MutableDateTime",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "setDayOfYear implements method in ReadWritableDateTime; expected @Override",
    " Code": "    public void setDayOfYear(final int dayOfYear) {",
    " Mark": "                ^",
    " Line": 733
},
{
    " Proj": "Time-3",
    "Class": "org.joda.time.MutableDateTime",
    " Type": "warning",
    "  Cat": "MissingOverride",
    "  Msg": "setHourOfDay implements method in ReadWritableDateTime; expected @Override",
    " Code": "    public void setHourOfDay(final int hourOfDay) {",
    " Mark": "                ^",
    " Line": 774
},
```


Methodology: Differential Based Warnings

Actual Code changes on Buggy and Fixed version of the Code:

```
722   public void addWeeks(final int weeks) {
723           setMillis(getChronology().weeks().add(getMillis(), weeks));
724   }
725
726   //-----------------------------------------------------------------------
727   /**
728    * Set the day of year to the specified value.
729    *
730    * @param dayOfYear the day of the year
731    * @throws IllegalArgumentException if the value is invalid
732    */
733   public void setDayOfYear(final int dayOfYear) {
734       setMillis(getChronology().dayOfYear().set(getMillis(), dayOfYear));
735   }
736
```

```
730   public void addWeeks(final int weeks) {
731       if (weeks != 0) {
732           setMillis(getChronology().weeks().add(getMillis(), weeks));
733       }
734   }
735
736   //-----------------------------------------------------------------------
737   /**
738    * Set the day of year to the specified value.
739    *
740    * @param dayOfYear the day of the year
741    * @throws IllegalArgumentException if the value is invalid
742    */
743   public void setDayOfYear(final int dayOfYear) {
744       setMillis(getChronology().dayOfYear().set(getMillis(), dayOfYear));
745   }
746
```

```
763   public void addDays(final int days) {
764           setMillis(getChronology().days().add(getMillis(), days));
765   }
766
767   //-----------------------------------------------------------------------
768   /**
769    * Set the hour of the day to the specified value.
770    *
771    * @param hourOfDay  the hour of day
772    * @throws IllegalArgumentException if the value is invalid
773    */
774   public void setHourOfDay(final int hourOfDay) {
775       setMillis(getChronology().hourOfDay().set(getMillis(), hourOfDay));
776   }
777
778   /**
779    * Add a number of hours to the date.
780    *
781    * @param hours  the hours to add
782    * @throws IllegalArgumentException if the value is invalid
783    */
784   public void addHours(final int hours) {
785           setMillis(getChronology().hours().add(getMillis(), hours));
786   }
```

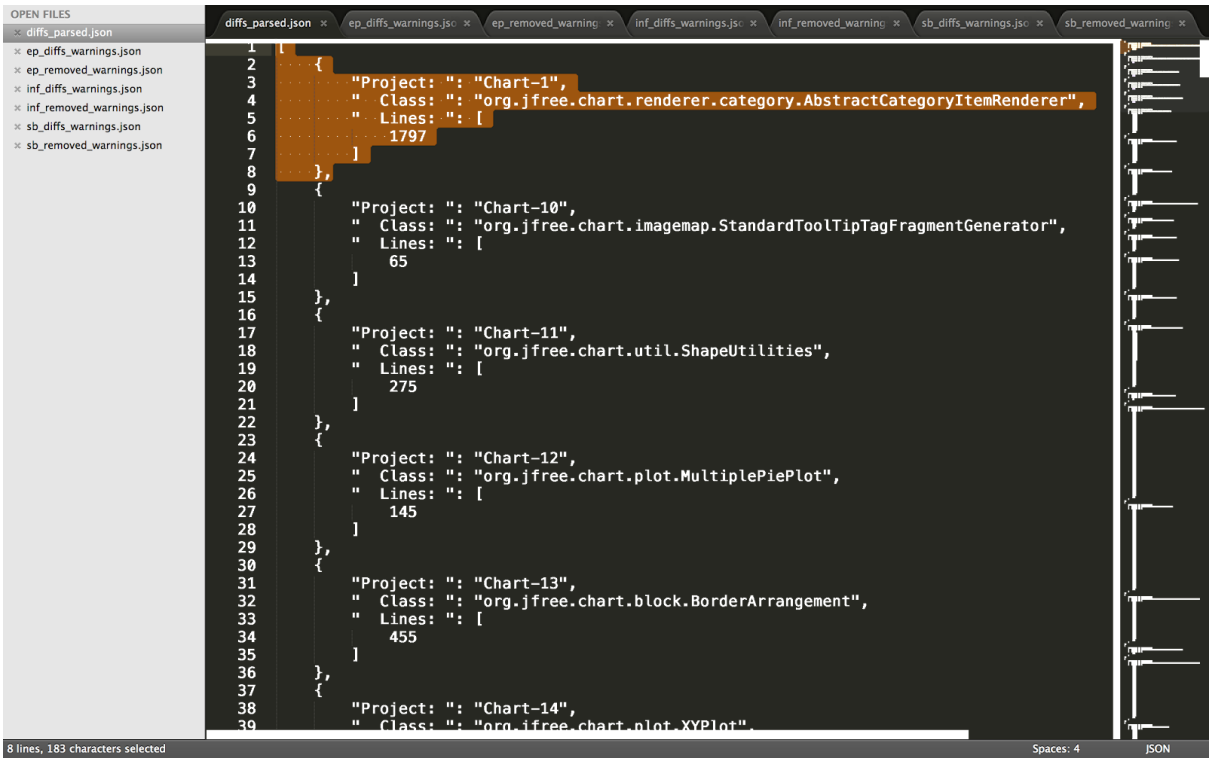
```
773   public void addDays(final int days) {
774       if (days != 0) {
775           setMillis(getChronology().days().add(getMillis(), days));
776       }
777   }
778
779   //-----------------------------------------------------------------------
780   /**
781    * Set the hour of the day to the specified value.
782    *
783    * @param hourOfDay  the hour of day
784    * @throws IllegalArgumentException if the value is invalid
785    */
786   public void setHourOfDay(final int hourOfDay) {
787       setMillis(getChronology().hourOfDay().set(getMillis(), hourOfDay));
788   }
789
790   /**
791    * Add a number of hours to the date.
792    *
793    * @param hours  the hours to add
794    * @throws IllegalArgumentException if the value is invalid
795    */
796   public void addHours(final int hours) {
797       if (hours != 0) {
798           setMillis(getChronology().hours().add(getMillis(), hours));
799       }
800   }
```

Category: Mismatch

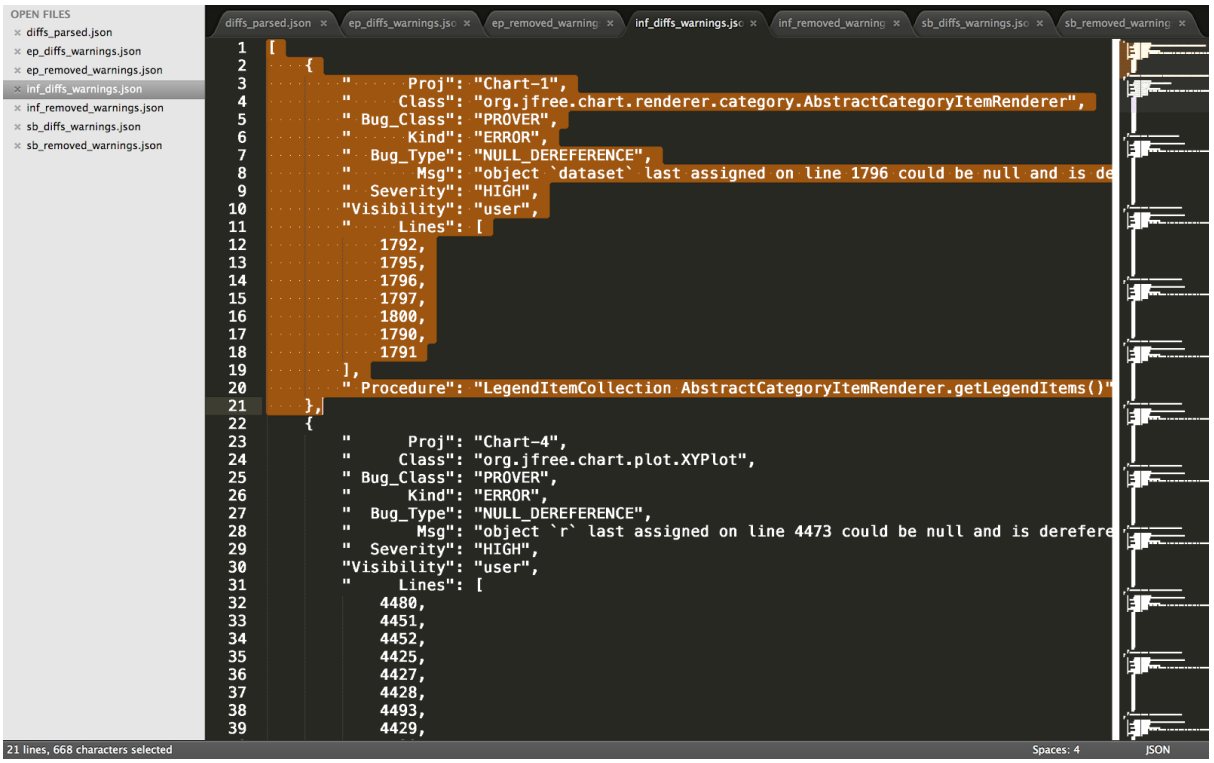
Comment:

Both the warnings were missing override, but none of the code changes were addressing that warning.

Chart 1:



Infer – 1, 2 methodology (Full match)



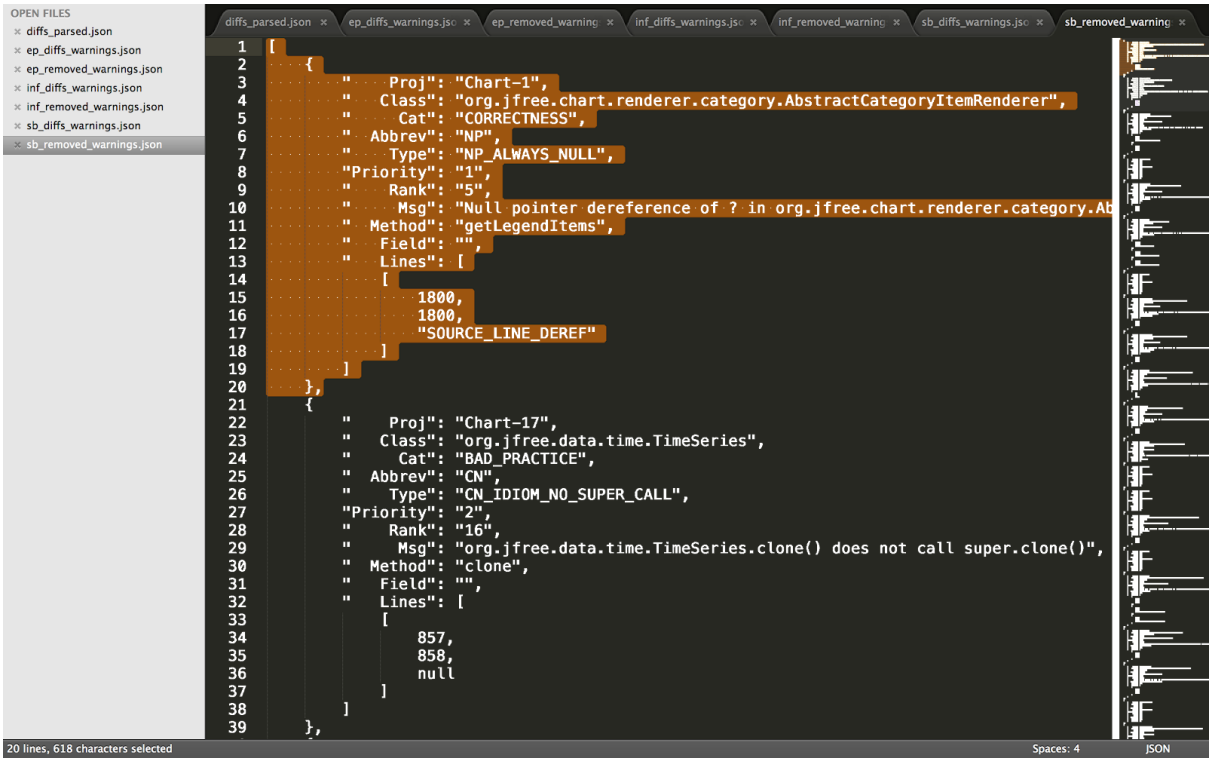
Buggy warnings:

```
Found 2 issues

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-1/source/
org/jfree/chart/renderer/category/AbstractCategoryItemRenderer.java:1129: error:
NULL_DEREFERENCE
  object `line` last assigned on line 1117 could be null and is dereferenced at line
1129.
  1127.               g2.setStroke(marker.getStroke());
  1128.               g2.draw(line);
  1129. >             bounds = line.getBounds2D();
  1130.           }
  1131.           else {

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-1/source/
org/jfree/chart/renderer/category/AbstractCategoryItemRenderer.java:1800: error:
NULL_DEREFERENCE
  object `dataset` last assigned on line 1796 could be null and is dereferenced at line
1800.
  1798.               return result;
  1799.           }
  1800. >         int seriesCount = dataset.getRowCount();
  1801.           if (plot.getRowRenderingOrder().equals(SortOrder.ASCENDING)) {
  1802.               for (int i = 0; i < seriesCount; i++) {

Summary of the reports

  NULL_DEREFERENCE: 2
```

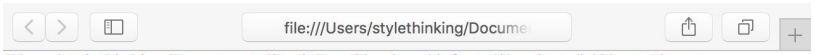
fixed warnings:

```
Found 1 issue

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Chart-1/source/org/
jfree/chart/renderer/category/AbstractCategoryItemRenderer.java:1129: error:
NULL_DEREFERENCE
  object `line` last assigned on line 1117 could be null and is dereferenced at line 1129.
  1127.               g2.setStroke(marker.getStroke());
  1128.               g2.draw(line);
  1129. >             bounds = line.getBounds2D();
  1130.           }
  1131.           else {

Summary of the reports

  NULL_DEREFERENCE: 1
```

Spotbugs – 2nd methodology (Full match)

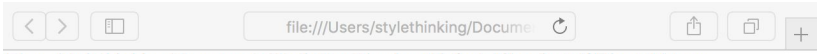


buggy warnings:



/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-1/source
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-1/build
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-1/lib/servlet.jar
Null pointer dereference Null pointer dereference of ? in org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.getLegendItems() At AbstractCategoryItemRenderer.java:[lines 249-1991] In class org.jfree.chart.renderer.category.AbstractCategoryItemRenderer In method org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.getLegendItems() Value loaded from ? Dereferenced at AbstractCategoryItemRenderer.java:[line 1800] Possible null pointer dereference Possible null pointer dereference of ? in org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.drawDomainMarker(Graphics2D, CategoryPlot, CategoryAxis, CategoryMarker, Rectangle2D) At AbstractCategoryItemRenderer.java:[lines 249-1991] In class org.jfree.chart.renderer.category.AbstractCategoryItemRenderer In method org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.drawDomainMarker(Graphics2D, CategoryPlot, CategoryAxis, CategoryMarker, Rectangle2D) Value loaded from ? Dereferenced at AbstractCategoryItemRenderer.java:[line 1129] Null value at AbstractCategoryItemRenderer.java:[line 1117] Known null at AbstractCategoryItemRenderer.java:[line 1122] Correctness Null pointer dereference A null pointer is dereferenced here. This will lead to a <code>NullPointerException</code> when the code is executed. Possible null pointer dereference There is a branch of statement that, if executed, guarantees that a null value will be dereferenced, which would generate a <code>NullPointerException</code> when the code is executed. Of course, the problem might be that the branch or statement is infeasible and that the null pointer exception can't ever be executed; deciding that is beyond the ability of SpotBugs. Null pointer dereference

fixed warnings:



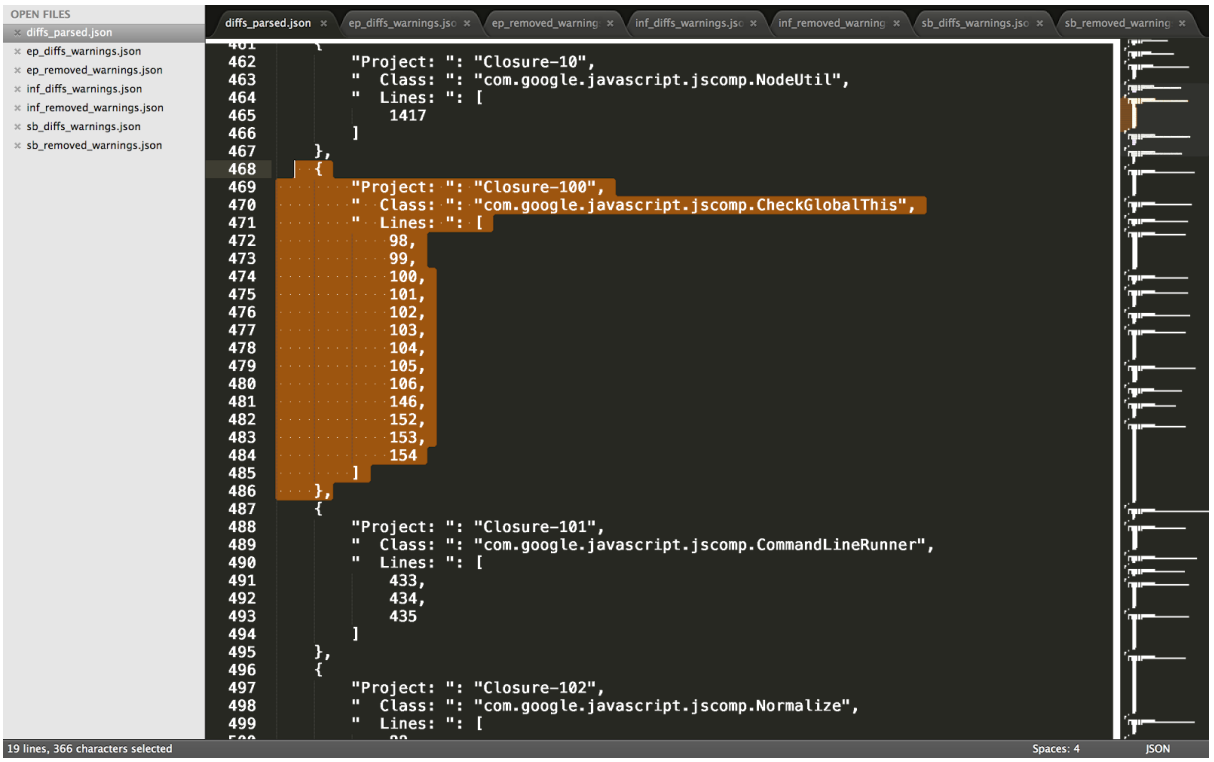
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Chart-1/source
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Chart-1/build
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Chart-1/lib/servlet.jar
Possible null pointer dereference Possible null pointer dereference of ? in org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.drawDomainMarker(Graphics2D, CategoryPlot, CategoryAxis, CategoryMarker, Rectangle2D) At AbstractCategoryItemRenderer.java:[lines 249-1991] In class org.jfree.chart.renderer.category.AbstractCategoryItemRenderer In method org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.drawDomainMarker(Graphics2D, CategoryPlot, CategoryAxis, CategoryMarker, Rectangle2D) Value loaded from ? Dereferenced at AbstractCategoryItemRenderer.java:[line 1129] Null value at AbstractCategoryItemRenderer.java:[line 1117] Known null at AbstractCategoryItemRenderer.java:[line 1122] Correctness Possible null pointer dereference There is a branch of statement that, if executed, guarantees that a null value will be dereferenced, which would generate a <code>NullPointerException</code> when the code is executed. Of course, the problem might be that the branch or statement is infeasible and that the null pointer exception can't ever be executed; deciding that is beyond the ability of SpotBugs. Null pointer dereference

Code diff:

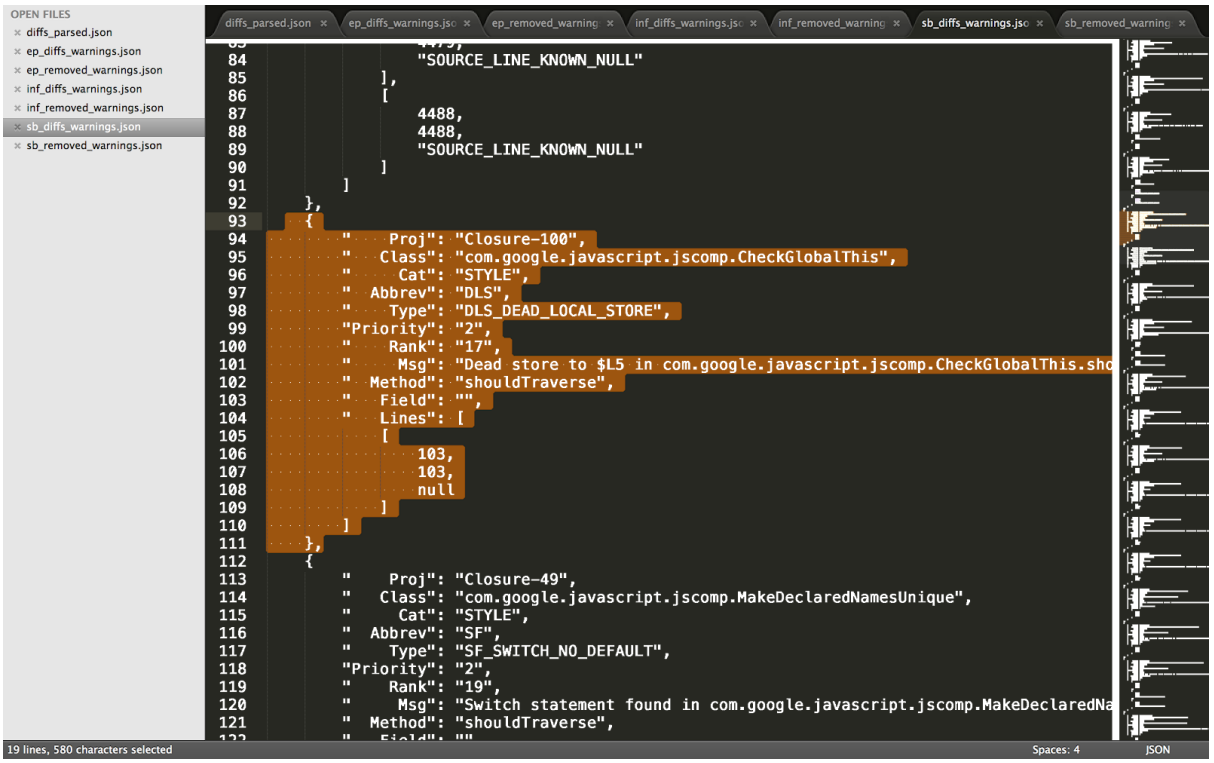
```
1788      * @see #getLegendItem(int, int)
1789      */
1790     public LegendItemCollection getLegendItems() {
1791         LegendItemCollection result = new LegendItemCollection();
1792         if (this.plot == null) {
1793             return result;
1794         }
1795         int index = this.plot.getIndexOf(this);
1796         CategoryDataset dataset = this.plot.getDataset(index);
1797         if (dataset != null) {
1798             return result;
1799         }
1800         int seriesCount = dataset.getRowCount();
1801         if (plot.getRowRenderingOrder().equals(SortOrder.ASCENDING)) {
1802             for (int i = 0; i < seriesCount; i++) {
1803                 if (isSeriesVisibleInLegend(i)) {
1804                     LegendItem item = getLegendItem(index, i);
1805                     if (item != null) {
1806                         result.add(item);
1807                     }
1808                 }
1809             }
1810         }
1811         else {
```

```
1788      * @see #getLegendItem(int, int)
1789      */
1790     public LegendItemCollection getLegendItems() {
1791         LegendItemCollection result = new LegendItemCollection();
1792         if (this.plot == null) {
1793             return result;
1794         }
1795         int index = this.plot.getIndexOf(this);
1796         CategoryDataset dataset = this.plot.getDataset(index);
1797         if (dataset == null) {
1798             return result;
1799         }
1800         int seriesCount = dataset.getRowCount();
1801         if (plot.getRowRenderingOrder().equals(SortOrder.ASCENDING)) {
1802             for (int i = 0; i < seriesCount; i++) {
1803                 if (isSeriesVisibleInLegend(i)) {
1804                     LegendItem item = getLegendItem(index, i);
1805                     if (item != null) {
1806                         result.add(item);
1807                     }
1808                 }
1809             }
1810         }
1811         else {
```

Closure-100:



spotbugs – meth. 1 (false positive[no warnings resolved])



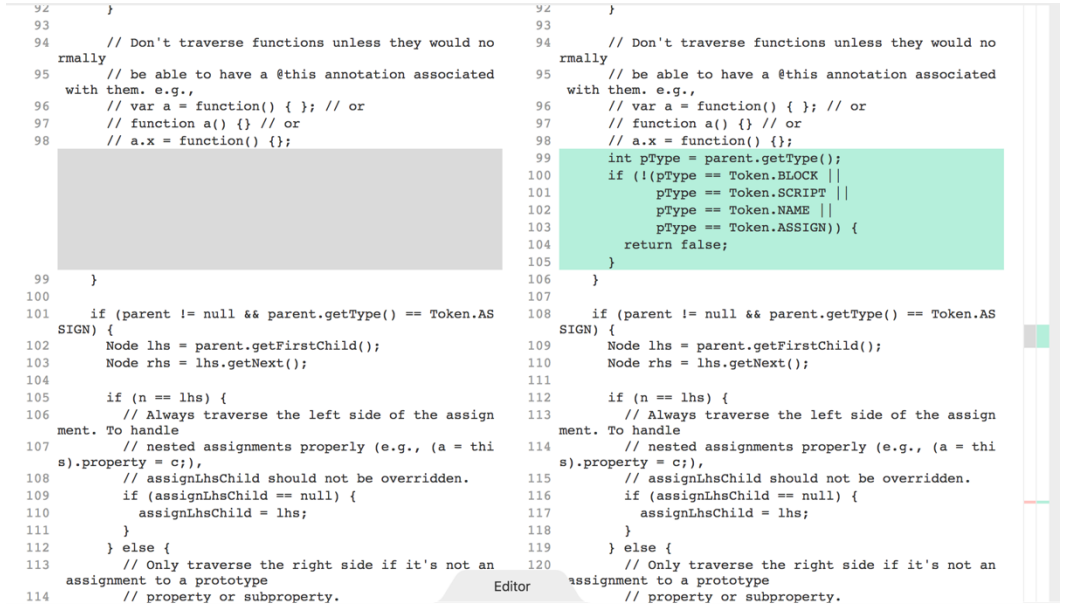
buggy warnings:

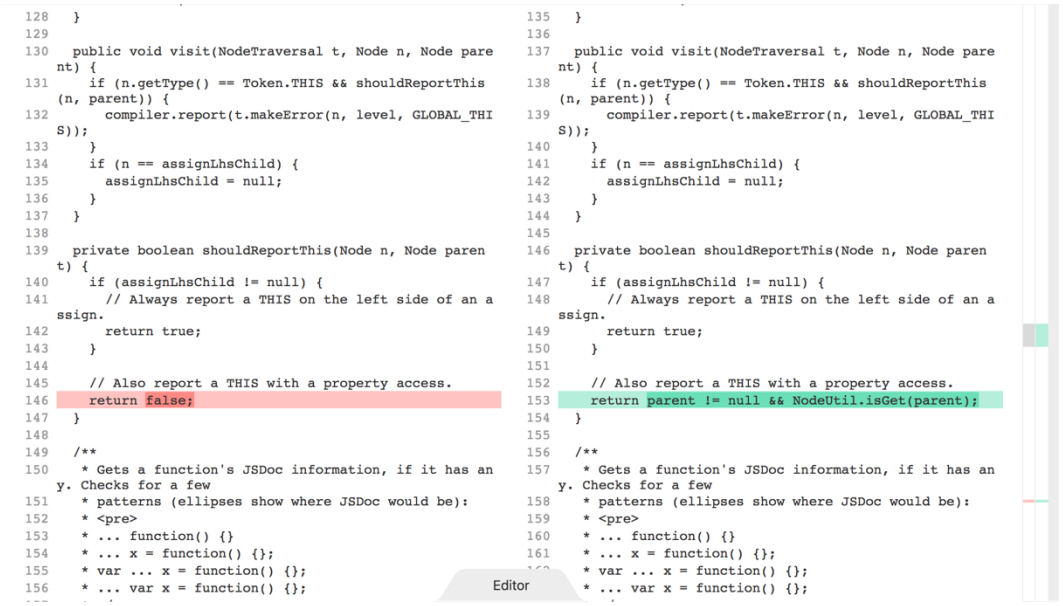
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/src /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/build/classes /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/ant_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/args4j_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/google_common_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/hamcrest-core-1.1.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/junit.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/libtrunk_rhino_parser_jarjared.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/protobuf_deploy.jar Dead store to local variable Dead store to $L5 in com.google.javascript.jscomp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node, Node) At CheckGlobalThis.java:[lines 61-174] In class com.google.javascript.jscomp.CheckGlobalThis In method com.google.javascript.jscomp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node, Node) Local variable stored in JVM register 5 At CheckGlobalThis.java:[line 103] Dodgy code Dead store to local variable This instruction assigns a value to a local variable, but the value is not read or used in any subsequent instruction. Often, this indicates an error, because the value computed is never used.   Note that Sun's javac compiler often generates dead stores for final local variables. Because SpotBugs is a bytecode-based tool, there is no easy way to eliminate these false positives. Dead local store

fixed warnings:

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/src /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/build/classes /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/ant_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/args4j_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/google_common_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/hamcrest-core-1.1.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/junit.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/libtrunk_rhino_parser_jarjared.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/protobuf_deploy.jar Dead store to local variable Dead store to $L5 in com.google.javascript.jscomp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node, Node) At CheckGlobalThis.java:[lines 61-181] In class com.google.javascript.jscomp.CheckGlobalThis In method com.google.javascript.jscomp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node, Node) Local variable stored in JVM register 5 At CheckGlobalThis.java:[line 110] Dodgy code Dead store to local variable This instruction assigns a value to a local variable, but the value is not read or used in any subsequent instruction. Often, this indicates an error, because the value computed is never used.   Note that Sun's javac compiler often generates dead stores for final local variables. Because SpotBugs is a bytecode-based tool, there is no easy way to eliminate these false positives. Dead local store

code diff:



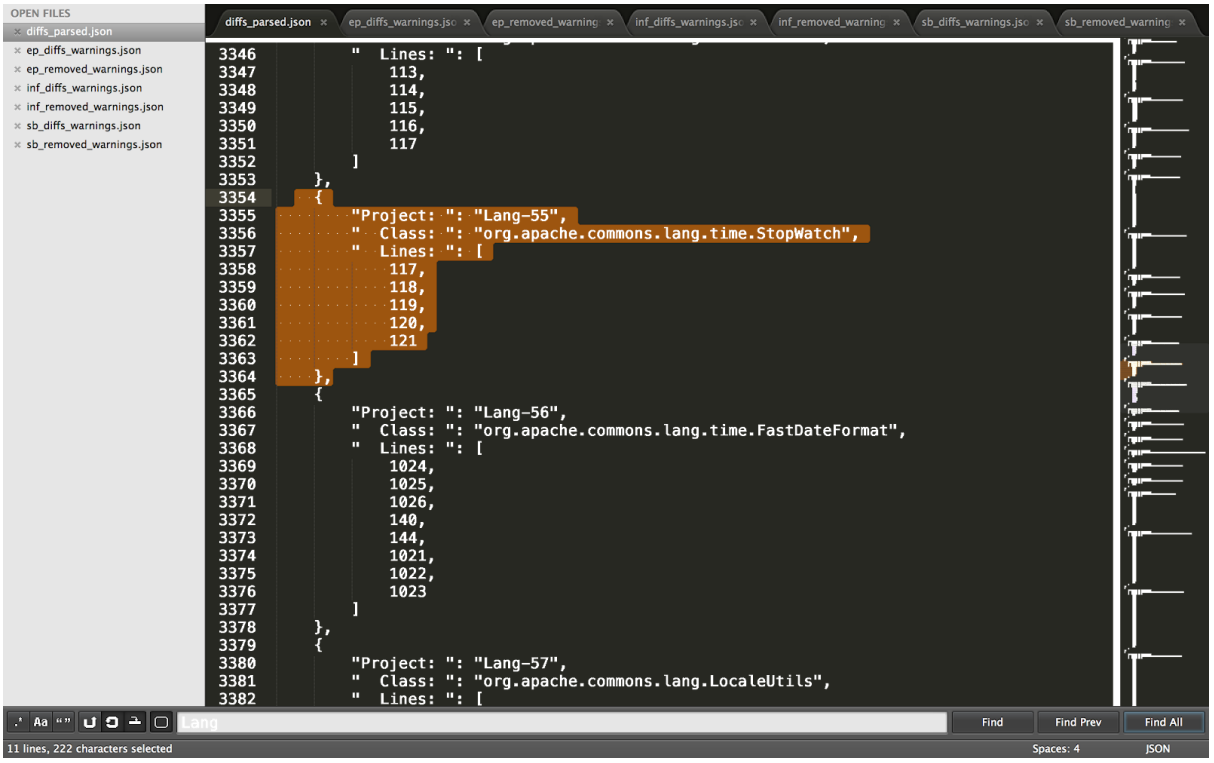


```
128   }
129
130   public void visit(NodeTraversal t, Node n, Node pare
    nt) {
131     if (n.getType() == Token.THIS && shouldReportThis
    (n, parent)) {
132       compiler.report(t.makeError(n, level, GLOBAL_THI
    S));
133     }
134     if (n == assignLhsChild) {
135       assignLhsChild = null;
136     }
137   }
138
139   private boolean shouldReportThis(Node n, Node paren
    t) {
140     if (assignLhsChild != null) {
141       // Always report a THIS on the left side of an a
    ssign.
142       return true;
143     }
144
145     // Also report a THIS with a property access.
146     return false;
147   }
148
149   /**
150    * Gets a function's JSDoc information, if it has an
    y. Checks for a few
151    * patterns (ellipses show where JSDoc would be):
152    * <pre>
153    * ... function() {}
154    * ... x = function() {};
155    * var ... x = function() {};
156    * ... var x = function() {};
```

```
135   }
136
137   public void visit(NodeTraversal t, Node n, Node pare
    nt) {
138     if (n.getType() == Token.THIS && shouldReportThis
    (n, parent)) {
139       compiler.report(t.makeError(n, level, GLOBAL_THI
    S));
140     }
141     if (n == assignLhsChild) {
142       assignLhsChild = null;
143     }
144   }
145
146   private boolean shouldReportThis(Node n, Node paren
    t) {
147     if (assignLhsChild != null) {
148       // Always report a THIS on the left side of an a
    ssign.
149       return true;
150     }
151
152     // Also report a THIS with a property access.
153     return parent != null && NodeUtil.isGet(parent);
154   }
155
156   /**
157    * Gets a function's JSDoc information, if it has an
    y. Checks for a few
158    * patterns (ellipses show where JSDoc would be):
159    * <pre>
160    * ... function() {}
161    * ... x = function() {};
    * var ... x = function() {};
    * ... var x = function() {};
```

Editor

Lang 55:



```
OPEN FILES
  diffs_parsed.json
  ep_diffs_warnings.json
  ep_removed_warnings.json
  inf_diffs_warnings.json
  inf_removed_warnings.json
  sb_diffs_warnings.json
  sb_removed_warnings.json

diffs_parsed.json | ep_diffs_warnings.jso | ep_removed_warning | inf_diffs_warnings.jso | inf_removed_warning | sb_diffs_warnings.jso | sb_removed_warning

3346            "  Lines: ": [
3347                113,
3348                114,
3349                115,
3350                116,
3351                117
3352            ]
3353        },
3354        {
3355            "Project: ": "Lang-55",
3356            "  Class: ": "org.apache.commons.lang.time.StopWatch",
3357            "  Lines: ": [
3358                117,
3359                118,
3360                119,
3361                120,
3362                121
3363            ]
3364        },
3365        {
3366            "Project: ": "Lang-56",
3367            "  Class: ": "org.apache.commons.lang.time.FastDateFormat",
3368            "  Lines: ": [
3369                1024,
3370                1025,
3371                1026,
3372                140,
3373                144,
3374                1021,
3375                1022,
3376                1023
3377            ]
3378        },
3379        {
3380            "Project: ": "Lang-57",
3381            "  Class: ": "org.apache.commons.lang.LocaleUtils",
3382            "  Lines: ": [

Aa  Lang   Find  Find Prev  Find All

11 lines, 222 characters selected        Spaces: 4   JSON
```

buggy warnings: fixed warnings: not detected by any tool
(same warnings)

```
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Lang-55/src/java/
org/apache/commons/lang/time/StopWatch.java:246: warning: [MissingOverride] toString
overrides method in Object; expected @Override
    public String toString() {
                  ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public String toString() {'?
1 warning
```

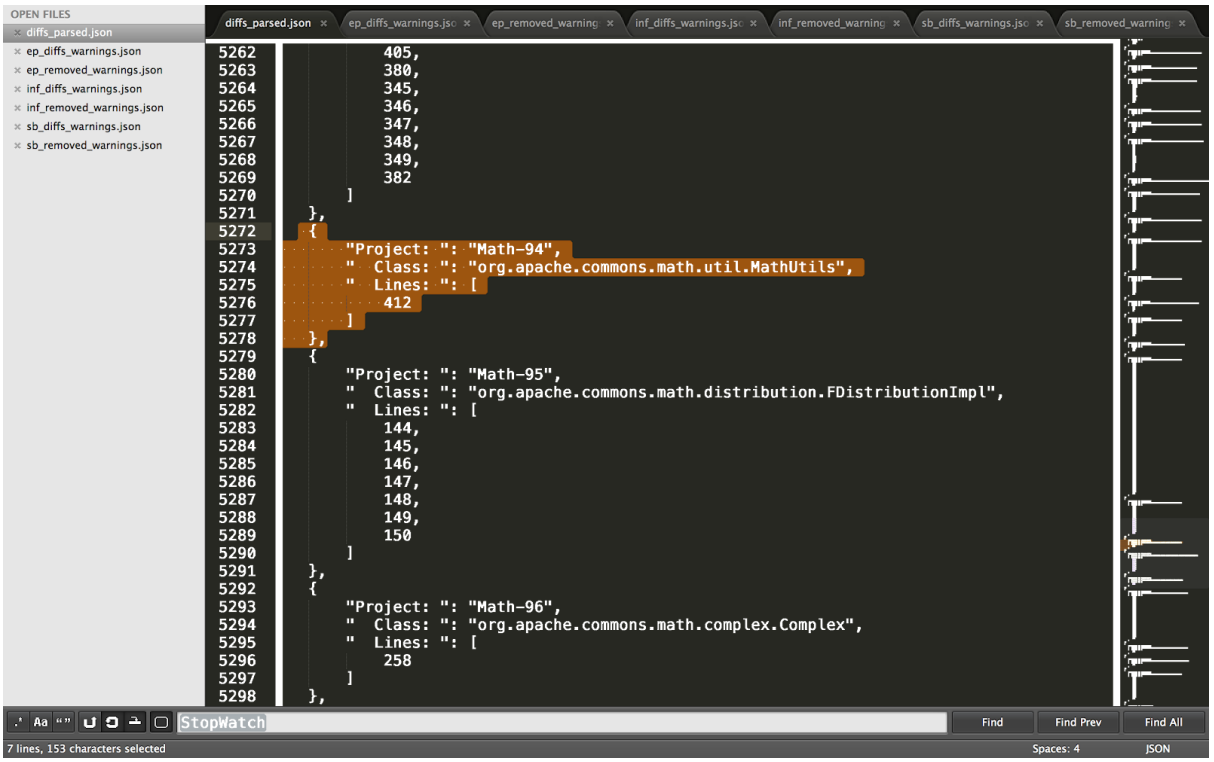
code diff: developer forgot to handle a specific case: Domain-specific

```
109      *
110      * <p>This method ends a new timing session, allow
    ing the time to be retrieved.</p>
111      *
112      * @throws IllegalStateException if the StopWatch
     is not running.
113      */
114     public void stop() {
115         if(this.runningState != STATE_RUNNING && this.
    runningState != STATE_SUSPENDED) {
116             throw new IllegalStateException("Stopwatch
     is not running. ");
117         }
118             stopTime = System.currentTimeMillis();
119         this.runningState = STATE_STOPPED;
120     }
121
122     /**
123      * <p>Resets the stopwatch. Stops it if need be.
    </p>
124      *
125      * <p>This method clears the internal values to al
    low the object to be reused.</p>
126      */
127     public void reset() {
128         this.runningState = STATE_UNSTARTED;
129         this.splitState   = STATE_UNSPLIT;
130         startTime = -1;
131         stopTime = -1;
132     }
133
134     /**
135      * <p>Split the time.</p>
136      *
```

```
109      *
110      * <p>This method ends a new timing session, allow
    ing the time to be retrieved.</p>
111      *
112      * @throws IllegalStateException if the StopWatch
     is not running.
113      */
114     public void stop() {
115         if(this.runningState != STATE_RUNNING && this.
    runningState != STATE_SUSPENDED) {
116             throw new IllegalStateException("Stopwatch
     is not running. ");
117         }
118         if(this.runningState == STATE_RUNNING) {
119             stopTime = System.currentTimeMillis();
120         }
121         this.runningState = STATE_STOPPED;
122     }
123
124     /**
125      * <p>Resets the stopwatch. Stops it if need be.
    </p>
126      *
127      * <p>This method clears the internal values to al
    low the object to be reused.</p>
128      */
129     public void reset() {
130         this.runningState = STATE_UNSTARTED;
131         this.splitState   = STATE_UNSPLIT;
132         startTime = -1;
133         stopTime = -1;
134     }
135
136     /**
      * <p>Split the time.</p>
      *
```

Editor

Math 94:

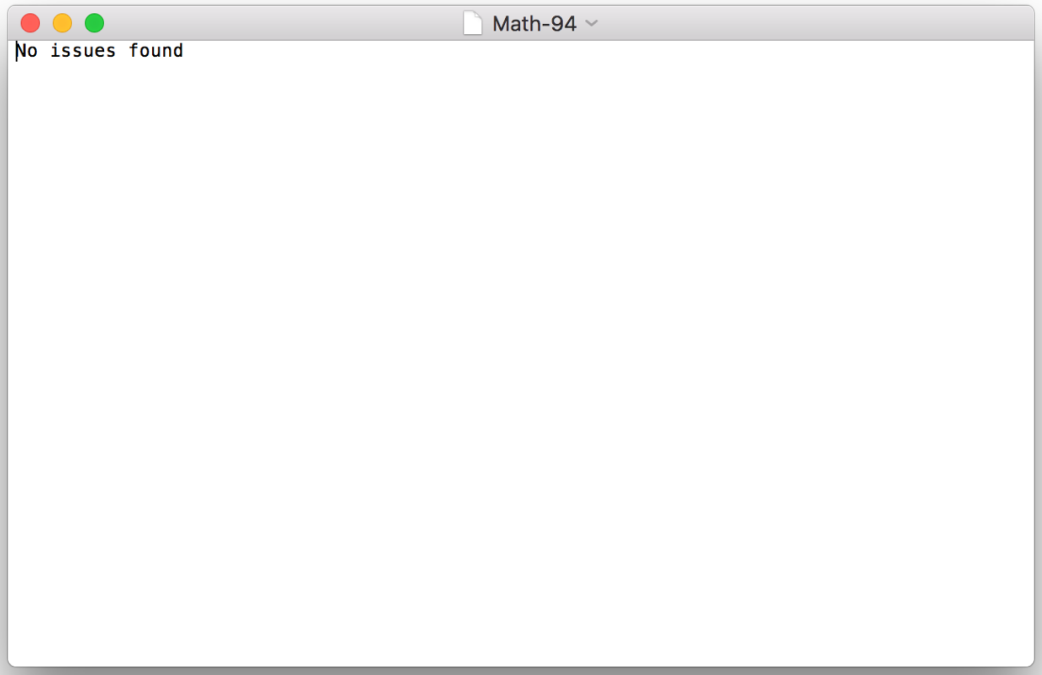


buggy warnings: fixed warnings:
(same)

Error prone: false-positive

```
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-94/src/java/
org/apache/commons/math/util/MathUtils.java:469: warning: [BoxedPrimitiveConstructor]
valueOf or autoboxing provides better time and space performance
        return new Double(value).hashCode();
               ^
    (see http://errorprone.info/bugpattern/BoxedPrimitiveConstructor)
  Did you mean 'return Double.hashCode(value);'?
1 warning
```

Infer: No warnings

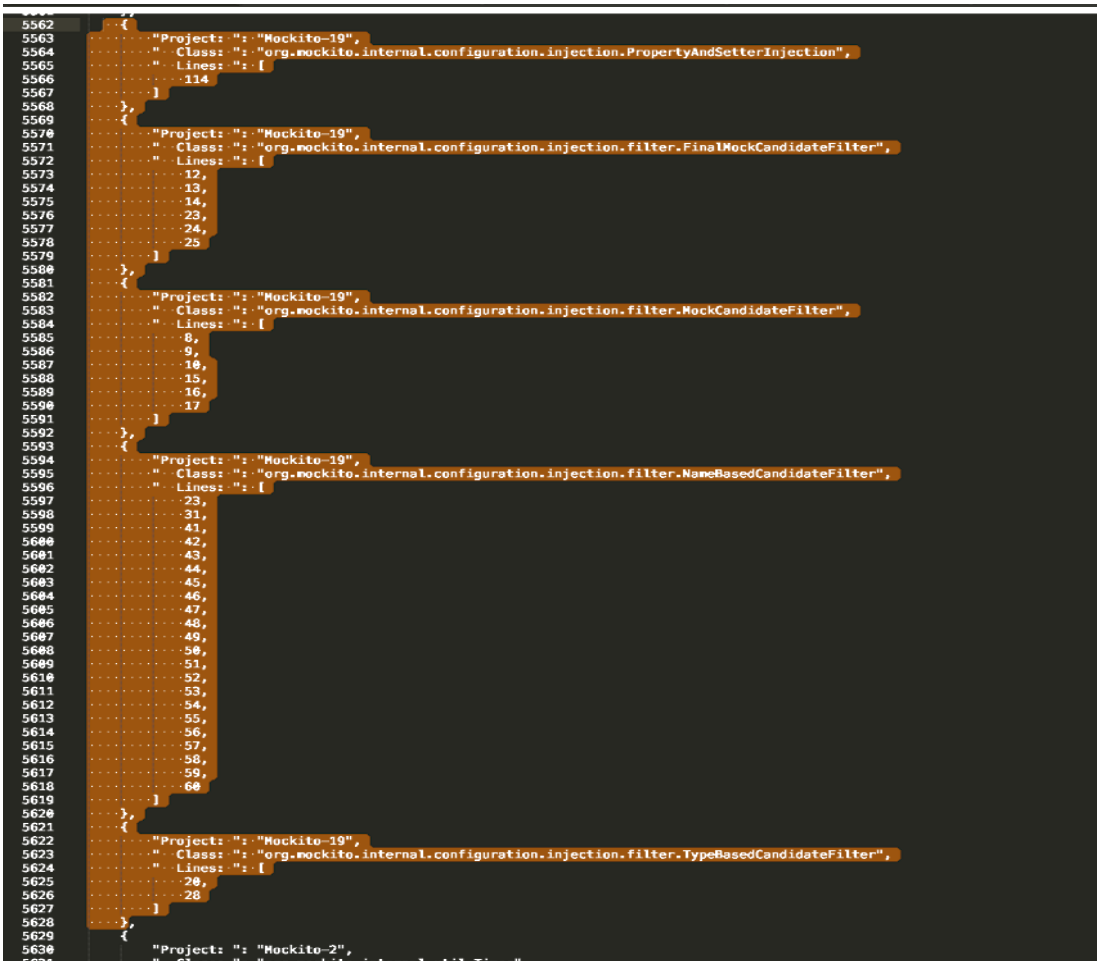


code diff:

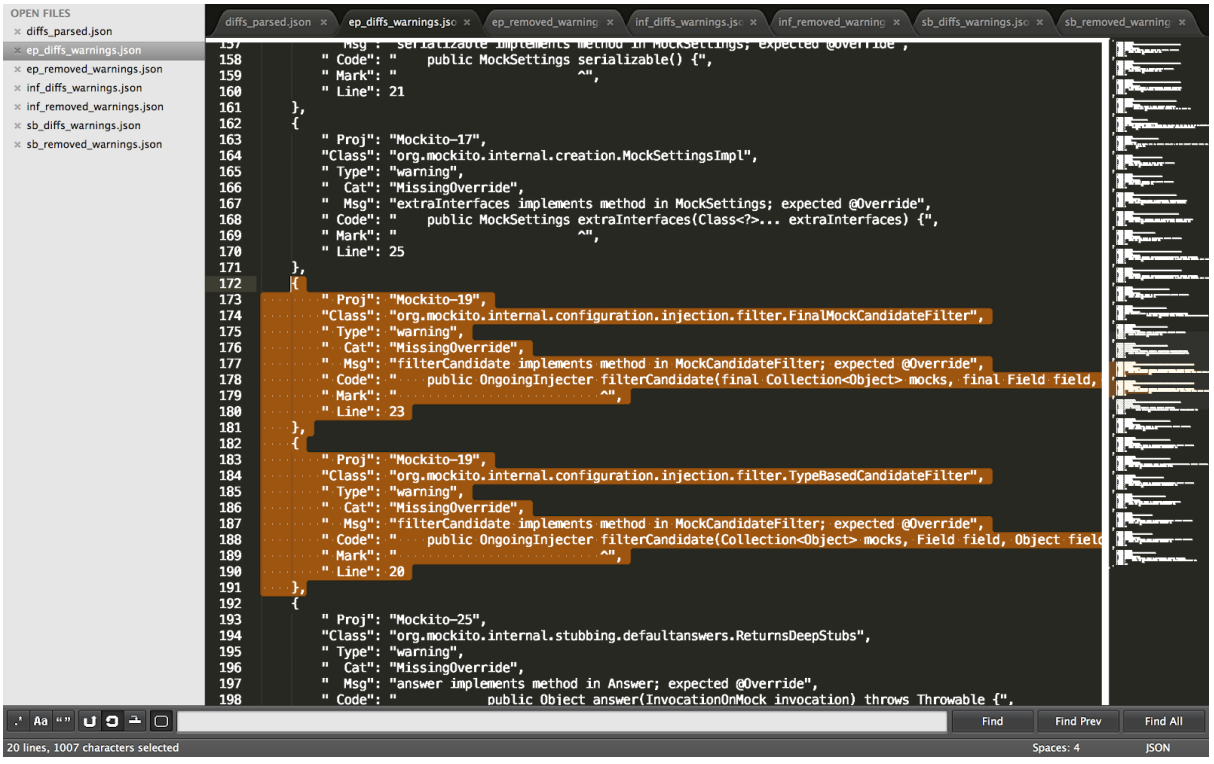
```
    e value of two numbers,
401      * using the "binary gcd" method which avoids divi
    sion and modulo
402      * operations. See Knuth 4.5.2 algorithm B. This a
    lgorithm is due to Josef
403      * Stein (1961).
404      * </p>
405      *
406      * @param u a non-zero number
407      * @param v a non-zero number
408      * @return the greatest common divisor, never zero
409      * @since 1.1
410      */
411     public static int gcd(int u, int v) {
412         if (u * v == 0) {
413             return (Math.abs(u) + Math.abs(v));
414         }
415         // keep u and v negative, as negative integers
     range down to
416         // -2^31, while positive numbers can only be a
    s large as 2^31-1
417         // (i.e. we can't necessarily negate a negativ
    e number without
418         // overflow)
419         /* assert u!=0 && v!=0; */
420         if (u > 0) {
421             u = -u;
422         } // make u negative
423         if (v > 0) {
424             v = -v;
425         } // make v negative
426         // B1. [Find power of 2]
427         int k = 0;
```

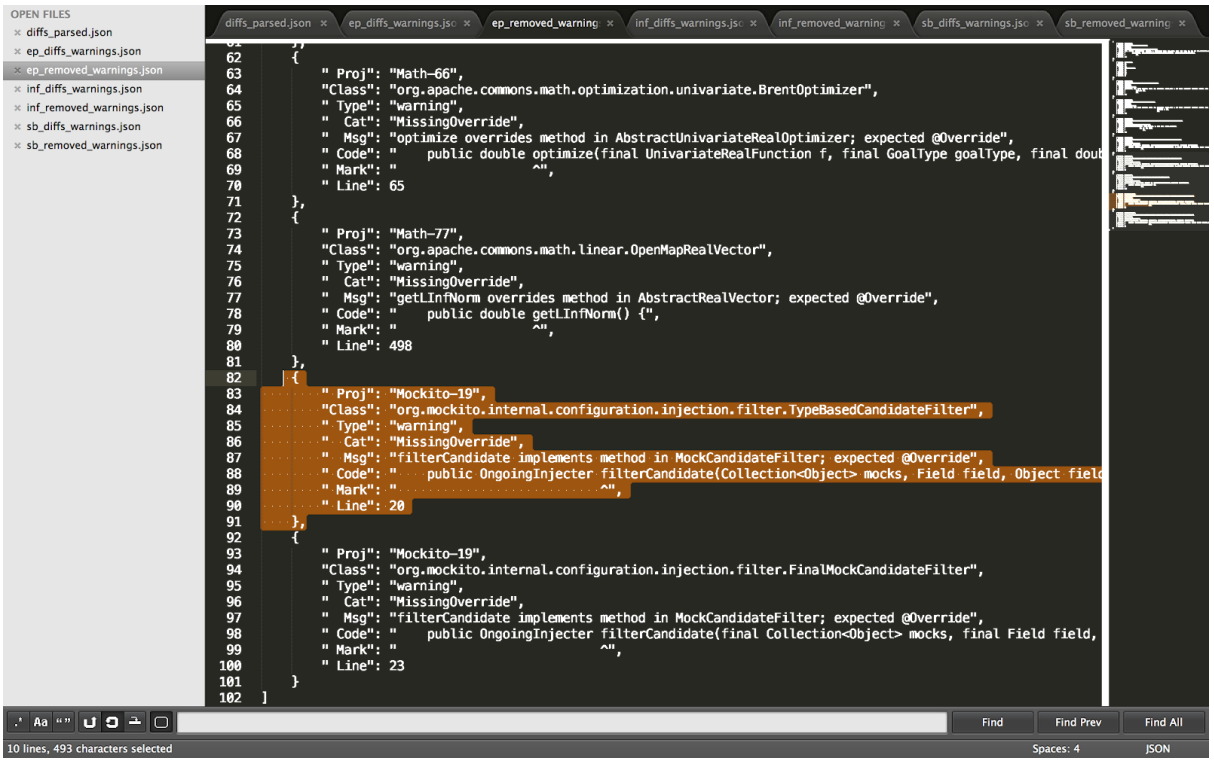
```
    e value of two numbers,
401      * using the "binary gcd" method which avoids divi
    sion and modulo
402      * operations. See Knuth 4.5.2 algorithm B. This a
    lgorithm is due to Josef
403      * Stein (1961).
404      * </p>
405      *
406      * @param u a non-zero number
407      * @param v a non-zero number
408      * @return the greatest common divisor, never zero
409      * @since 1.1
410      */
411     public static int gcd(int u, int v) {
412         if ((u == 0) || (v == 0)) {
413             return (Math.abs(u) + Math.abs(v));
414         }
415         // keep u and v negative, as negative integers
     range down to
416         // -2^31, while positive numbers can only be a
    s large as 2^31-1
417         // (i.e. we can't necessarily negate a negativ
    e number without
418         // overflow)
419         /* assert u!=0 && v!=0; */
420         if (u > 0) {
421             u = -u;
422         } // make u negative
423         if (v > 0) {
424             v = -v;
425         } // make v negative
426         // B1. [Find power of 2]
427         int k = 0;
```

Mockito-19:



Error-prone: Meth. 1, 2 (Full match)






```
    },
    {
        " Proj": "Math-66",
        "Class": "org.apache.commons.math.optimization.univariate.BrentOptimizer",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "optimize overrides method in AbstractUnivariateRealOptimizer; expected @Override",
        " Code": "    public double optimize(final UnivariateRealFunction f, final GoalType goalType, final dou
        " Mark": "                    ^",
        " Line": 65
    },
    {
        " Proj": "Math-77",
        "Class": "org.apache.commons.math.linear.OpenMapRealVector",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "getLInfNorm overrides method in AbstractRealVector; expected @Override",
        " Code": "    public double getLInfNorm() {",
        " Mark": "                  ^",
        " Line": 498
    },
    {
        " Proj": "Mockito-19",
        "Class": "org.mockito.internal.configuration.injection.filter.TypeBasedCandidateFilter",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "filterCandidate implements method in MockCandidateFilter; expected @Override",
        " Code": "    public OngoingInjecter filterCandidate(Collection<Object> mocks, Field field, Object field
        " Mark": "                              ^",
        " Line": 20
    },
    {
        " Proj": "Mockito-19",
        "Class": "org.mockito.internal.configuration.injection.filter.FinalMockCandidateFilter",
        " Type": "warning",
        "  Cat": "MissingOverride",
        "  Msg": "filterCandidate implements method in MockCandidateFilter; expected @Override",
        " Code": "    public OngoingInjecter filterCandidate(final Collection<Object> mocks, final Field field,
        " Mark": "                              ^",
        " Line": 23
    }
]
```

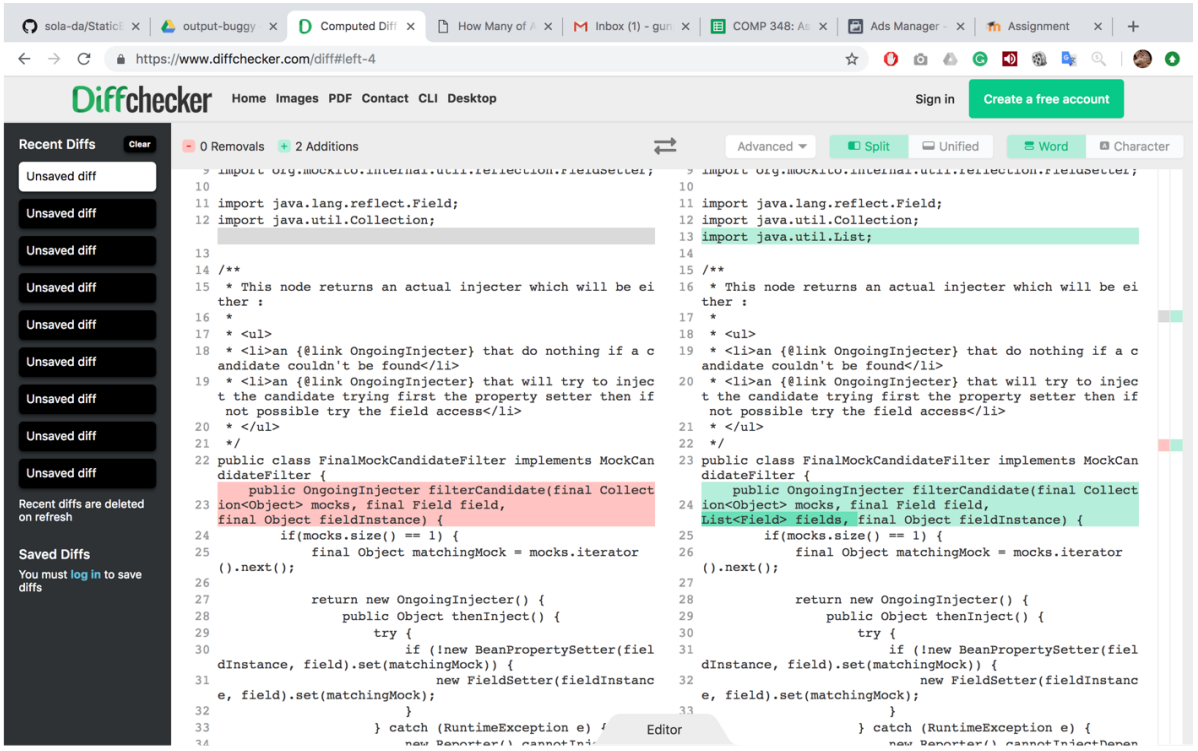
buggy warnings:

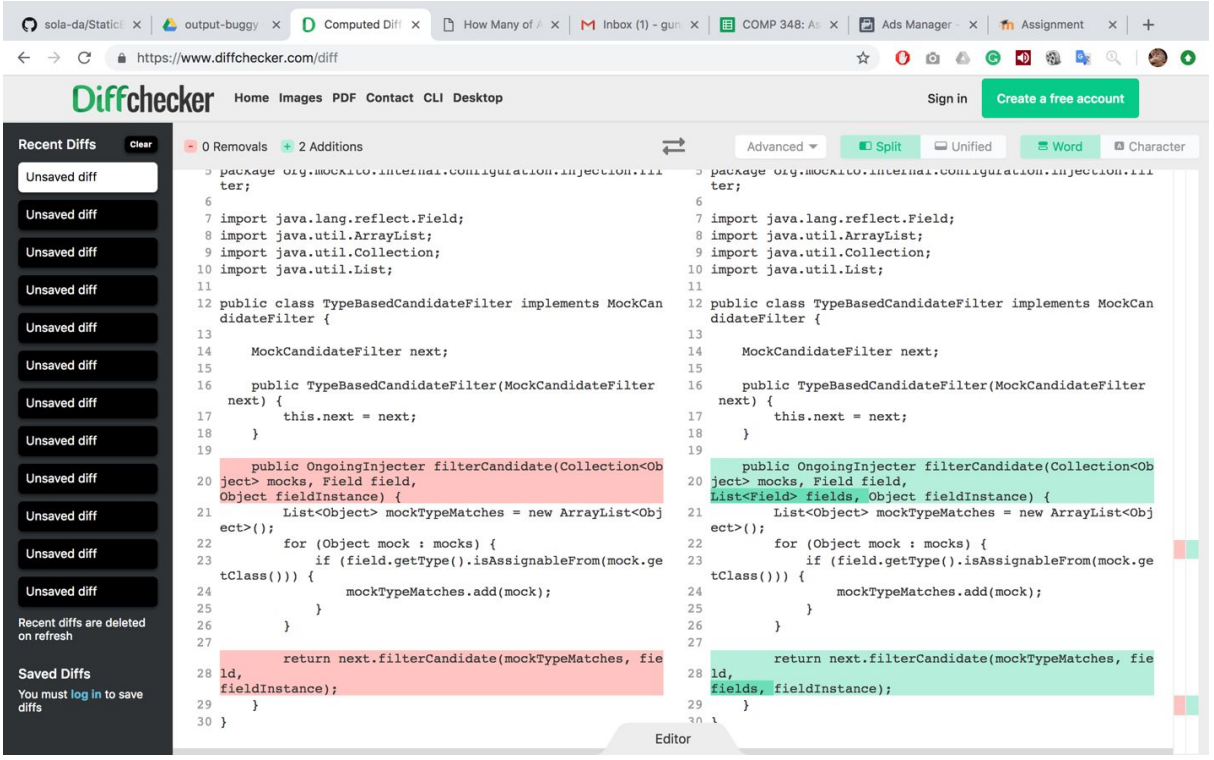
```
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Mockito-19/src/org/mockito/internal/configuration/injection/filter/NameBasedCandidateFilter.java:22: warning:
[MissingOverride] filterCandidate implements method in MockCandidateFilter; expected @Override
        public OngoingInjecter filterCandidate(Collection<Object> mocks,
                               ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public OngoingInjecter filterCandidate(Collection<Object> mocks,'?
1 warning
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Mockito-19/src/org/mockito/internal/configuration/injection/PropertyAndSetterInjection.java:65: warning:
[MissingOverride] isOut implements method in Filter; expected @Override
        public boolean isOut(Field object) {
                       ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public boolean isOut(Field object) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Mockito-19/src/org/mockito/internal/configuration/injection/PropertyAndSetterInjection.java:71: warning:
[MissingOverride] processInjection implements method in MockInjectionStrategy; expected @Override
    public boolean processInjection(Field injectMocksField, Object injectMocksFieldOwner, Set<Object> mockCandidates) {
                   ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public boolean processInjection(Field injectMocksField, Object injectMocksFieldOwner, Set<Object> mockCandidates) {'?
2 warnings
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Mockito-19/src/org/mockito/internal/configuration/injection/filter/TypeBasedCandidateFilter.java:20: warning:
[MissingOverride] filterCandidate implements method in MockCandidateFilter; expected @Override
    public OngoingInjecter filterCandidate(Collection<Object> mocks, Field field, Object fieldInstance) {
                           ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public OngoingInjecter filterCandidate(Collection<Object> mocks, Field field, Object fieldInstance) {'?
```

fixed warnings:

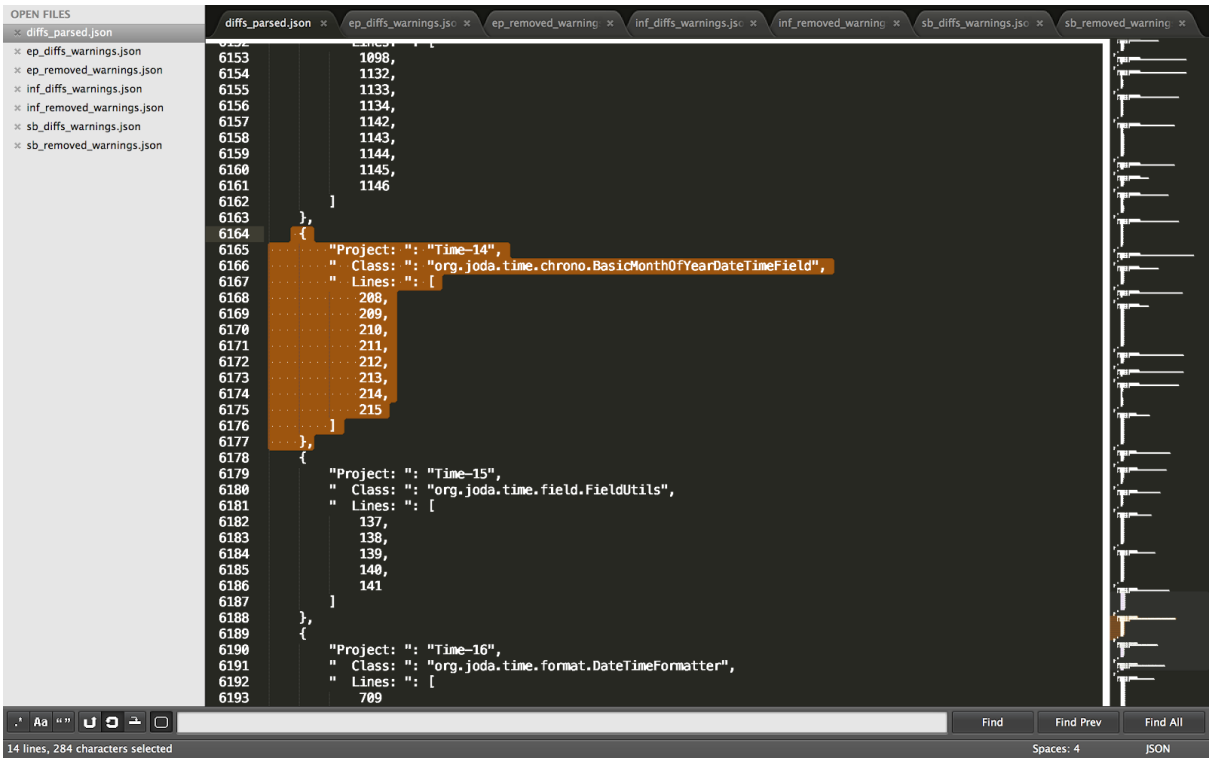
```
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Mockito-19/src/org/mockito/internal/configuration/injection/filter/TypeBasedCandidateFilter.java:20: warning:
[MissingOverride] filterCandidate implements method in MockCandidateFilter; expected @Override
    public OngoingInjecter filterCandidate(Collection<Object> mocks, Field field, List<Field> fields, Object fieldInstance) {
                           ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public OngoingInjecter filterCandidate(Collection<Object> mocks, Field field, List<Field> fields, Object fieldInstance) {'?
1 warning
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Mockito-19/src/org/mockito/internal/configuration/injection/filter/FinalMockCandidateFilter.java:24: warning:
[MissingOverride] filterCandidate implements method in MockCandidateFilter; expected @Override
    public OngoingInjecter filterCandidate(final Collection<Object> mocks, final Field field, List<Field> fields, final Object fieldInstance) {
                           ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public OngoingInjecter filterCandidate(final Collection<Object> mocks, final Field field, List<Field> fields, final Object fieldInstance) {'?
```

code diff:


Diffchecker
Home Images PDF Contact CLI Desktop
Sign in
Create a free account
Recent Diffs
Clear
Unsaved diff
Recent diffs are deleted on refresh
Saved Diffs
You must log in to save diffs
0 Removals
2 Additions
Advanced
Split
Unified
Word
Character
Editor


Diffchecker
Home Images PDF Contact CLI Desktop
Sign in
Create a free account
Recent Diffs
Clear
Unsaved diff
Recent diffs are deleted on refresh
Saved Diffs
You must log in to save diffs
0 Removals
2 Additions
Advanced
Split
Unified
Word
Character
Editor

Time-14:



No tool detects the bug

buggy warnings: fixed warnings:

```
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:58: warning: [MissingOverride]
isLenient implements method in DateTimeField; expected @Override
    public boolean isLenient() {
                   ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public boolean isLenient() {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:71: warning: [MissingOverride]
get implements method in ImpreciseDateTimeField; expected @Override
    public int get(long instant) {
               ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public int get(long instant) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:91: warning: [MissingOverride]
add implements method in ImpreciseDateTimeField; expected @Override
    public long add(long instant, int months) {
                ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public long add(long instant, int months) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:150: warning: [MissingOverride]
add implements method in ImpreciseDateTimeField; expected @Override
    public long add(long instant, long months) {
                ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public long add(long instant, long months) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:175: warning: [IntLongMath]
Expression of type int may overflow before being assigned to a long
            monthToUse = iMax - remMonthToUse + 1;
                         ^
    (see http://errorprone.info/bugpattern/IntLongMath)
  Did you mean 'monthToUse = (long) iMax - remMonthToUse + 1;'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:203: warning: [MissingOverride]
add overrides method in BaseDateTimeField; expected @Override
    public int[] add(ReadablePartial partial, int fieldIndex, int[] values, int valueToAdd) {
                 ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public int[] add(ReadablePartial partial, int fieldIndex, int[] values, int valueToAdd) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:237: warning: [MissingOverride]
addWrapField overrides method in BaseDateTimeField; expected @Override
    public long addWrapField(long instant, int months) {
                ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public long addWrapField(long instant, int months) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:242: warning: [MissingOverride]
getDifferenceAsLong overrides method in ImpreciseDateTimeField; expected @Override
    public long getDifferenceAsLong(long minuendInstant, long subtrahendInstant) {
                ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public long getDifferenceAsLong(long minuendInstant, long subtrahendInstant) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:297: warning: [MissingOverride]
set implements method in ImpreciseDateTimeField; expected @Override
    public long set(long instant, int month) {
                ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public long set(long instant, int month) {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:314: warning: [MissingOverride]
getRangeDurationField implements method in ImpreciseDateTimeField; expected @Override
    public DurationField getRangeDurationField() {
                         ^
    (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public DurationField getRangeDurationField() {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Time-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:319: warning: [MissingOverride]
isLean overrides method in BaseDateTimeField; expected @Override
```

code diff: developer forgot to handle a specific case: Domain-specific (Time-14 is due to code that handles dates but forgot to consider leap years and the consequences of February 29.)

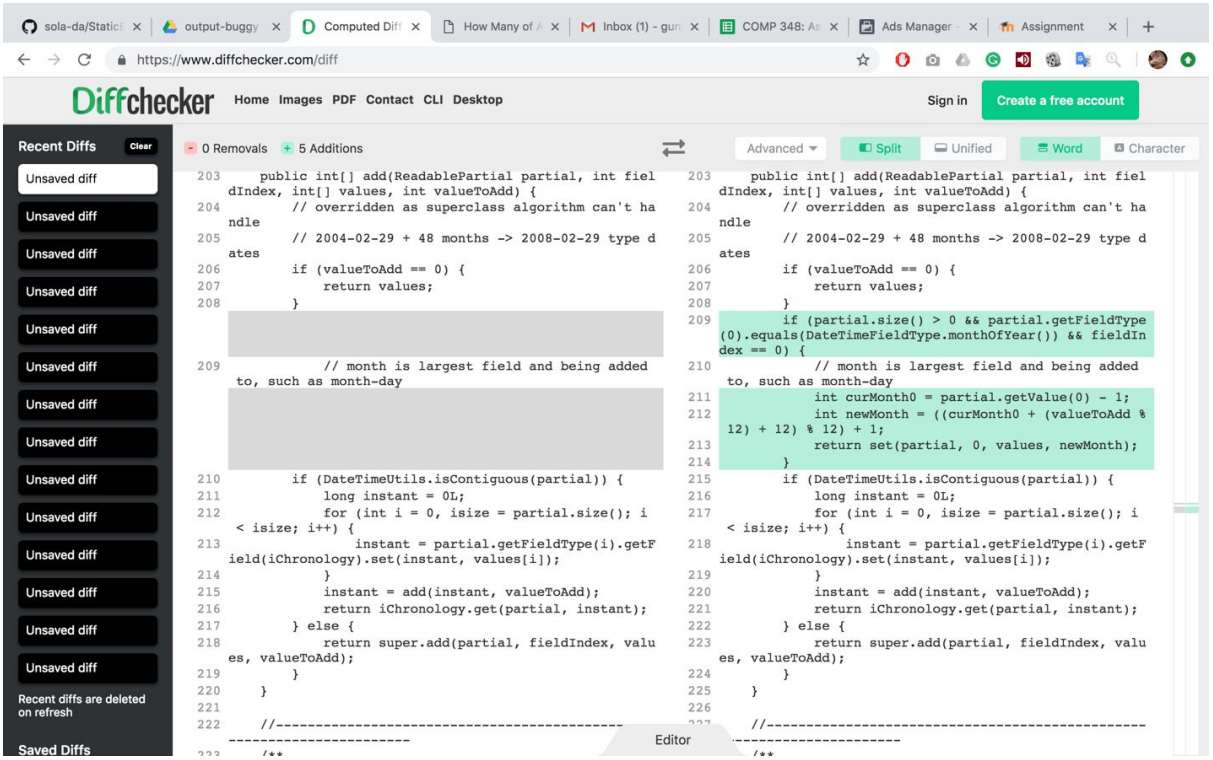
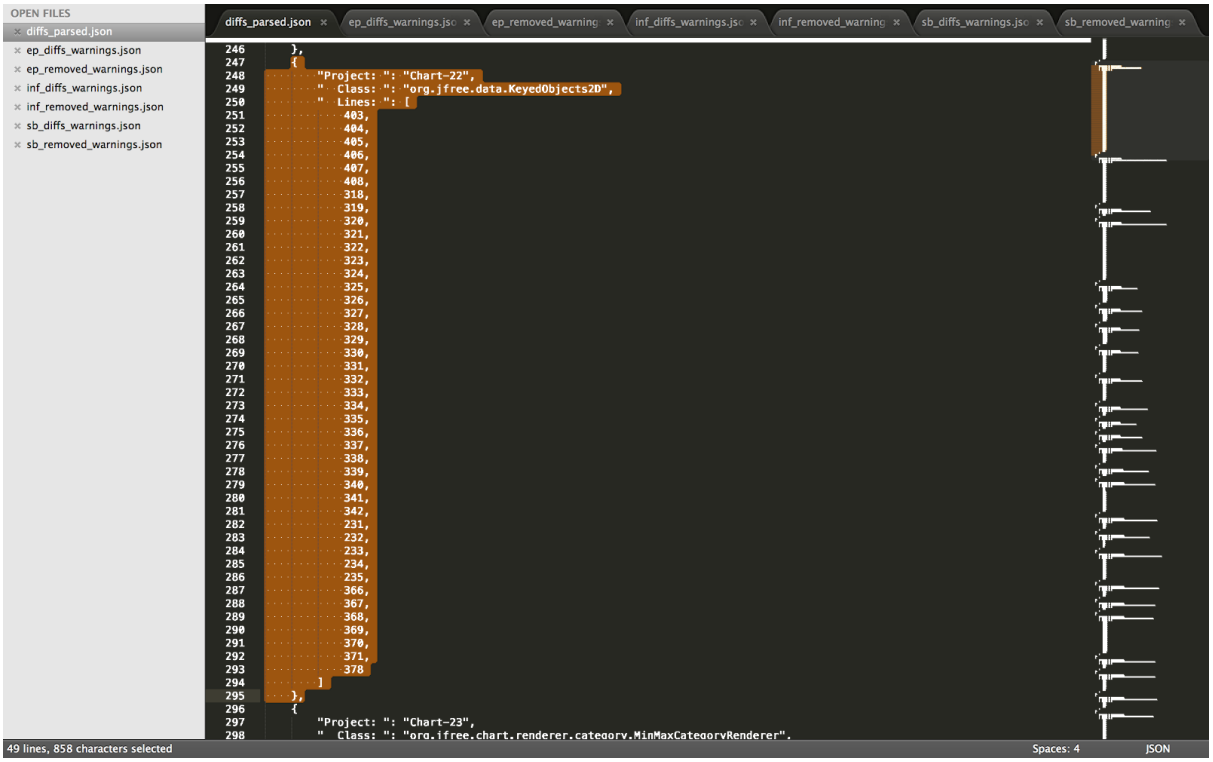
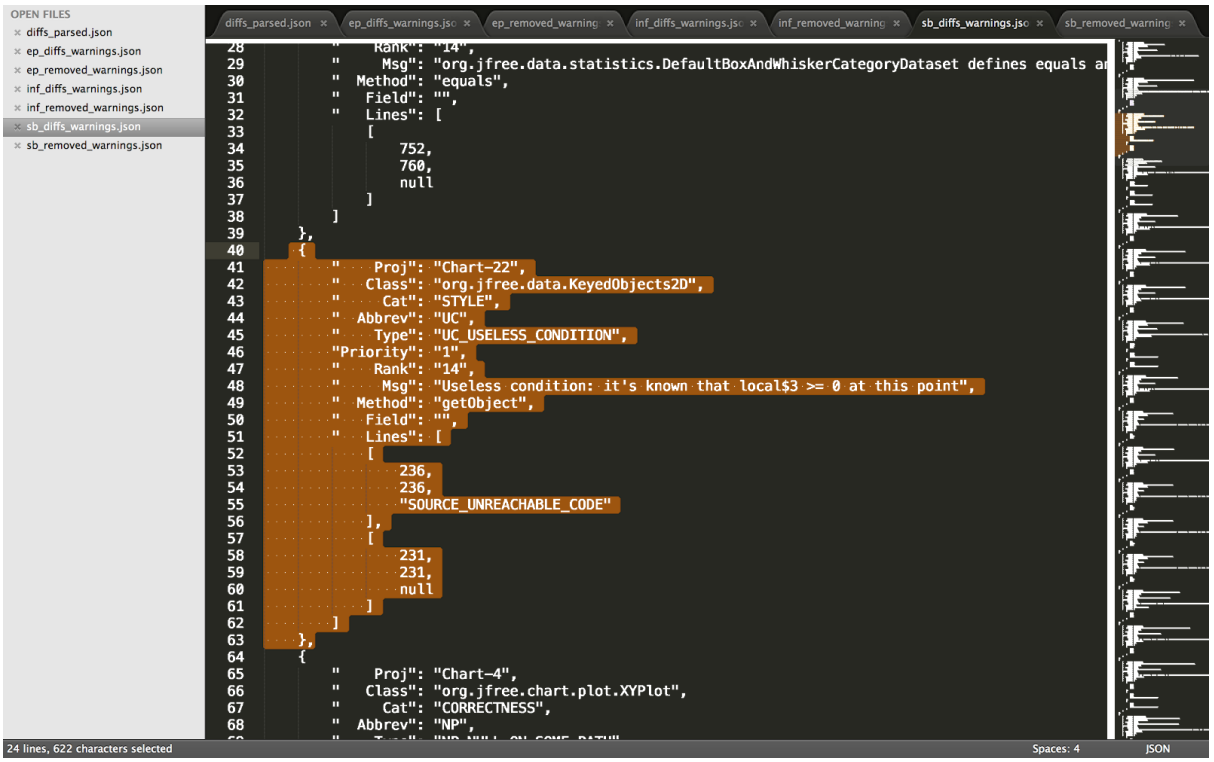


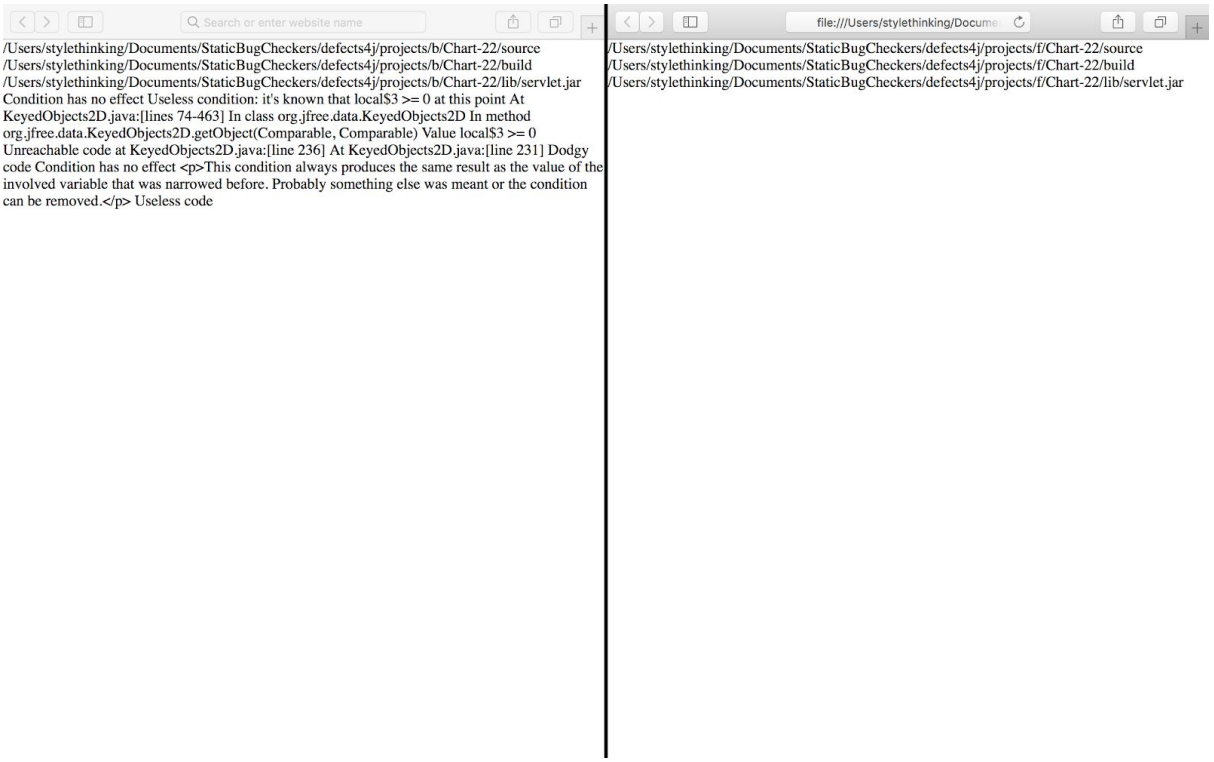
Chart-22:



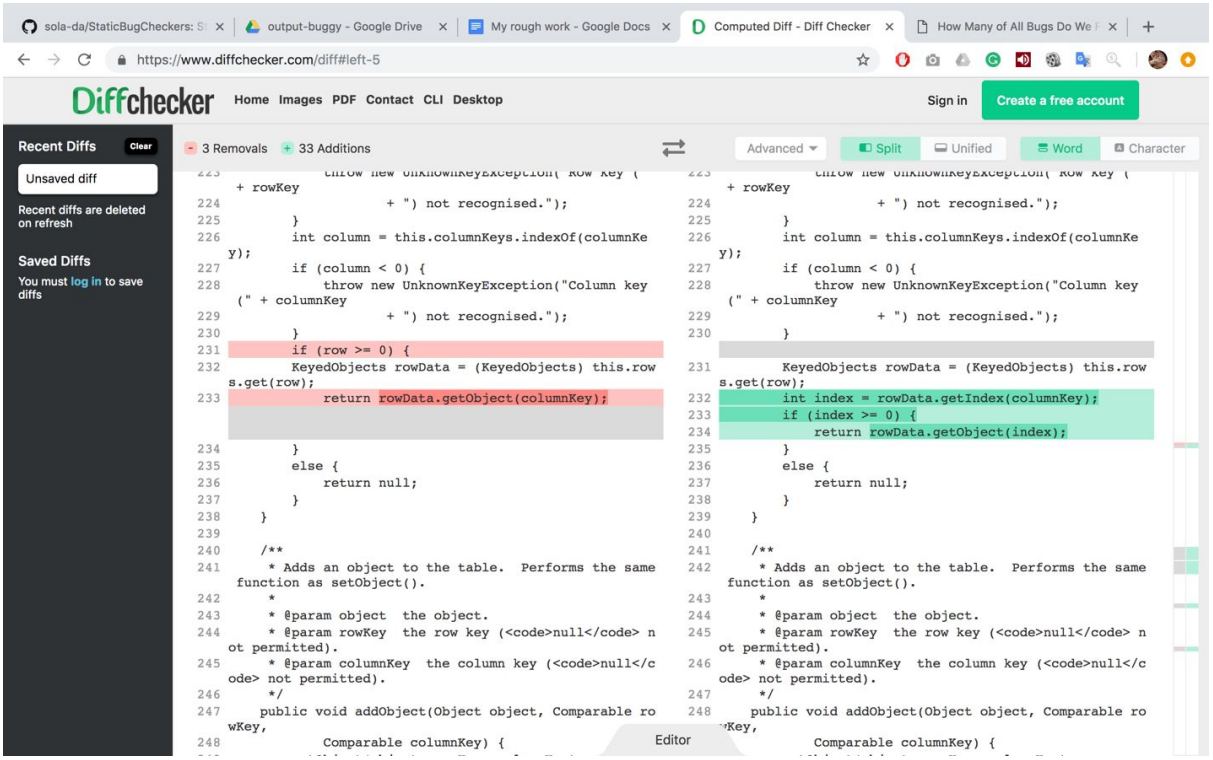
spotbugs: meth. 1, 2 (full match)



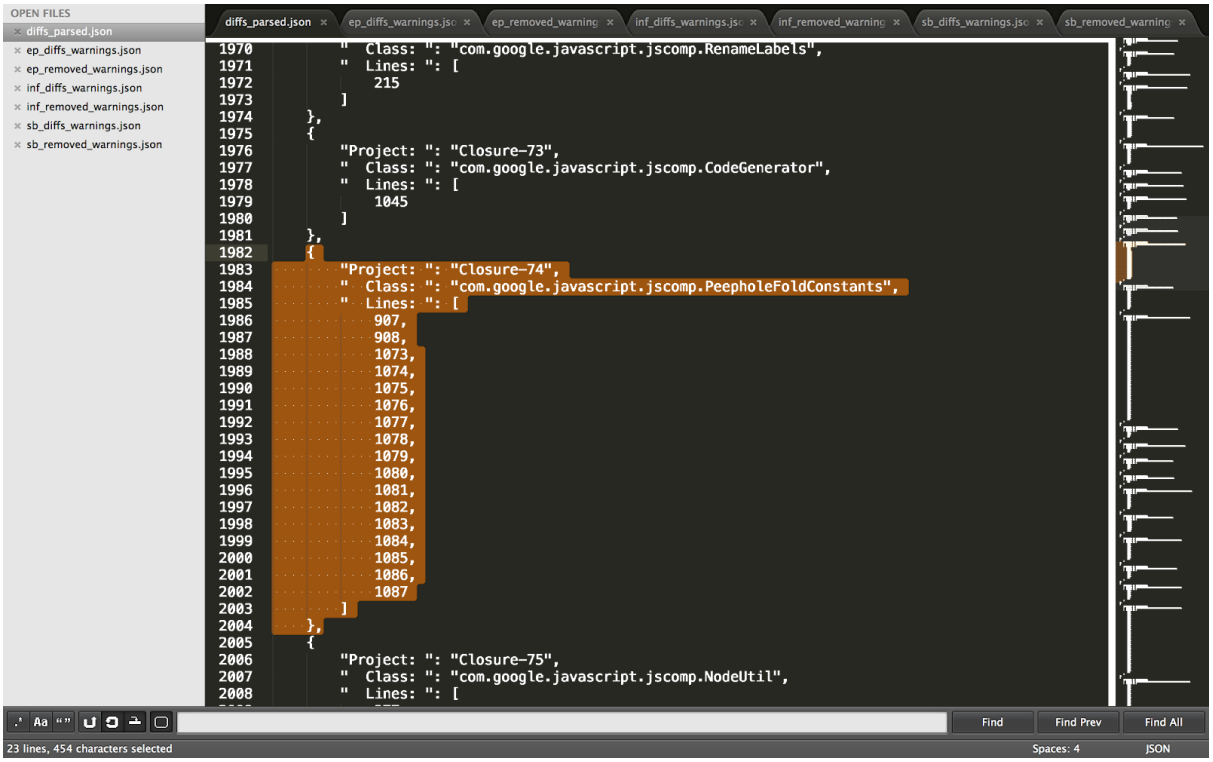
Buggy warnings | Fixed warnings:



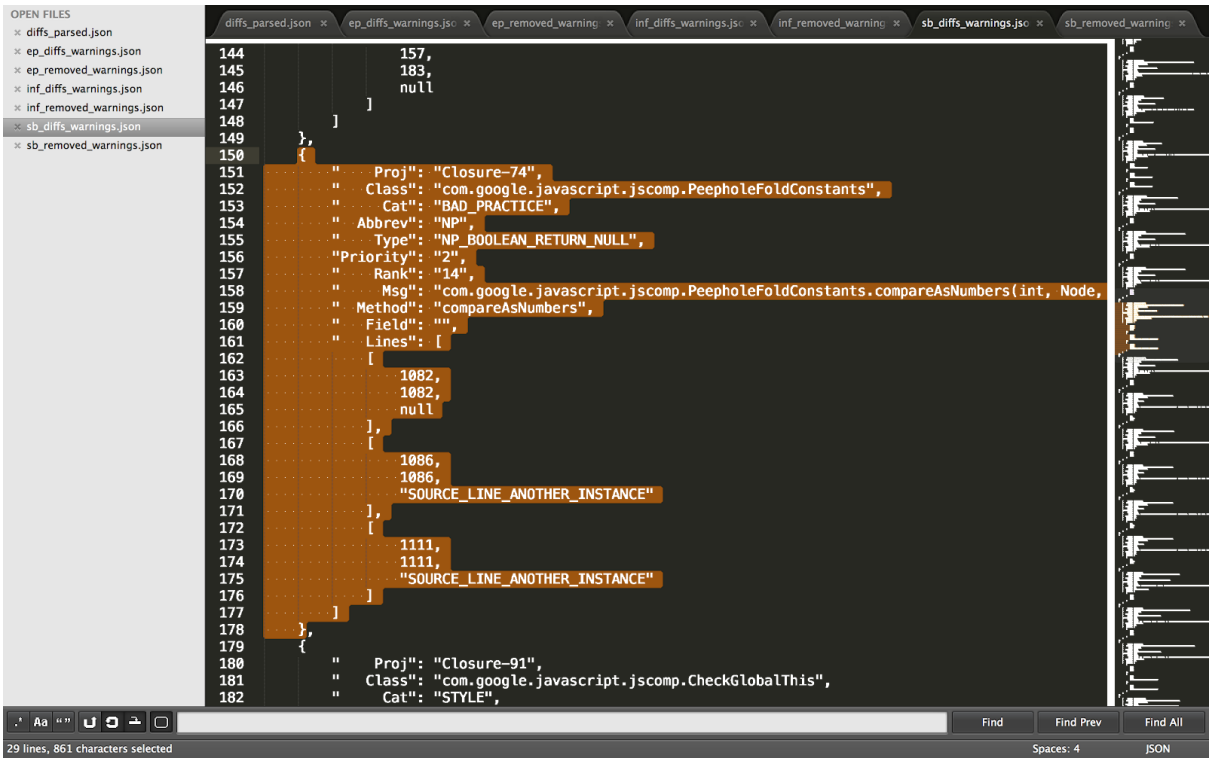
Code diff:



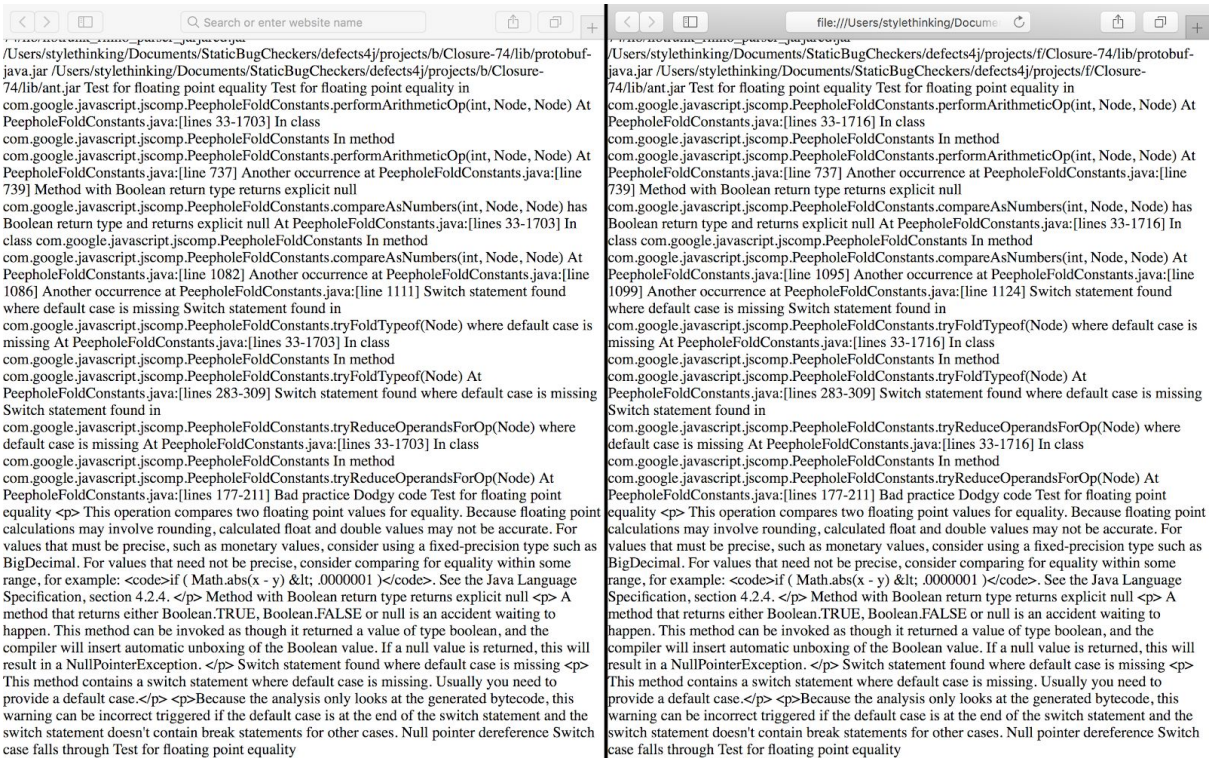
Closure-74:



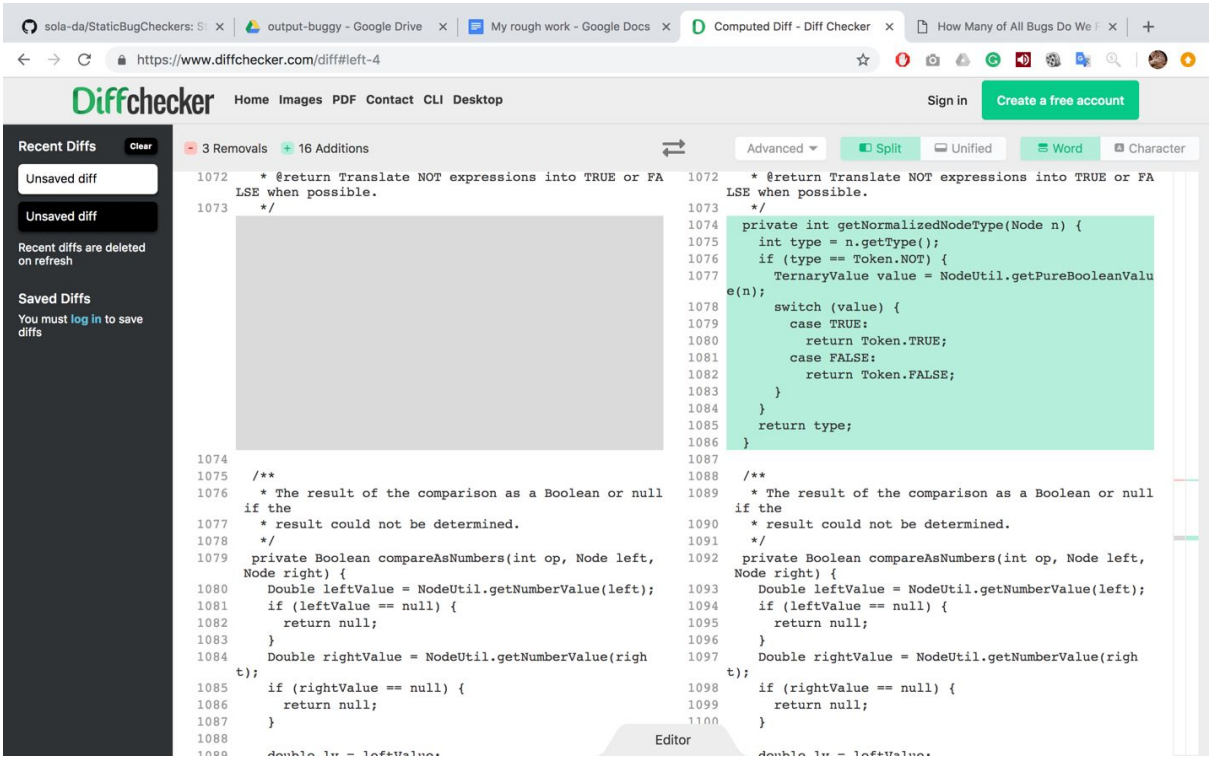
Spot bugs: meth 1 (mismatch)



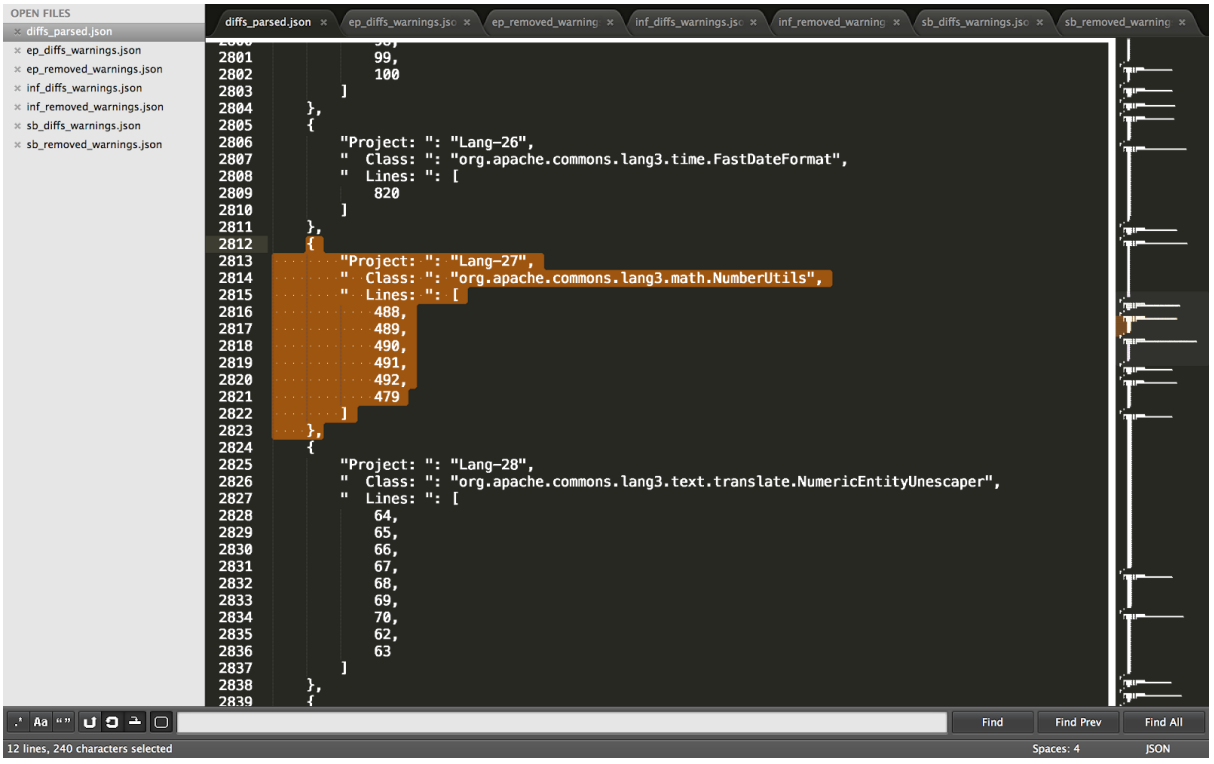
Buggy Warnings: Fixed Warning: same



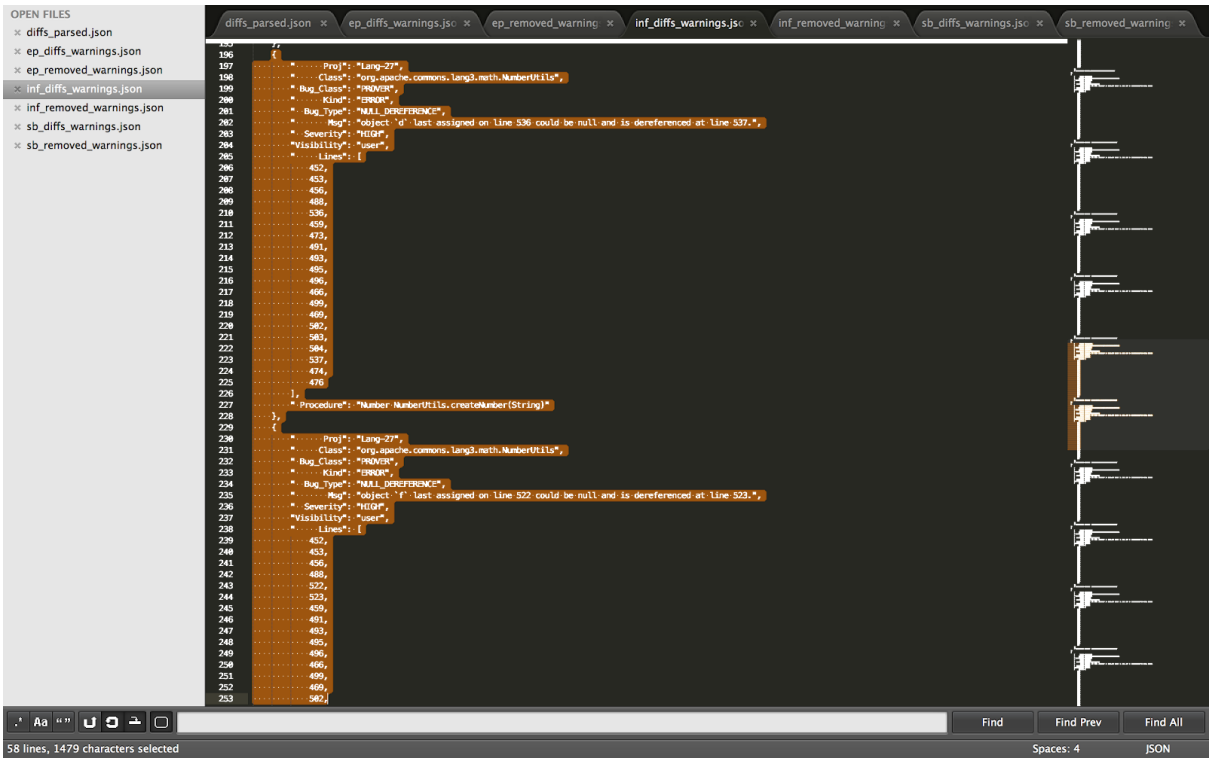
Code Diff:



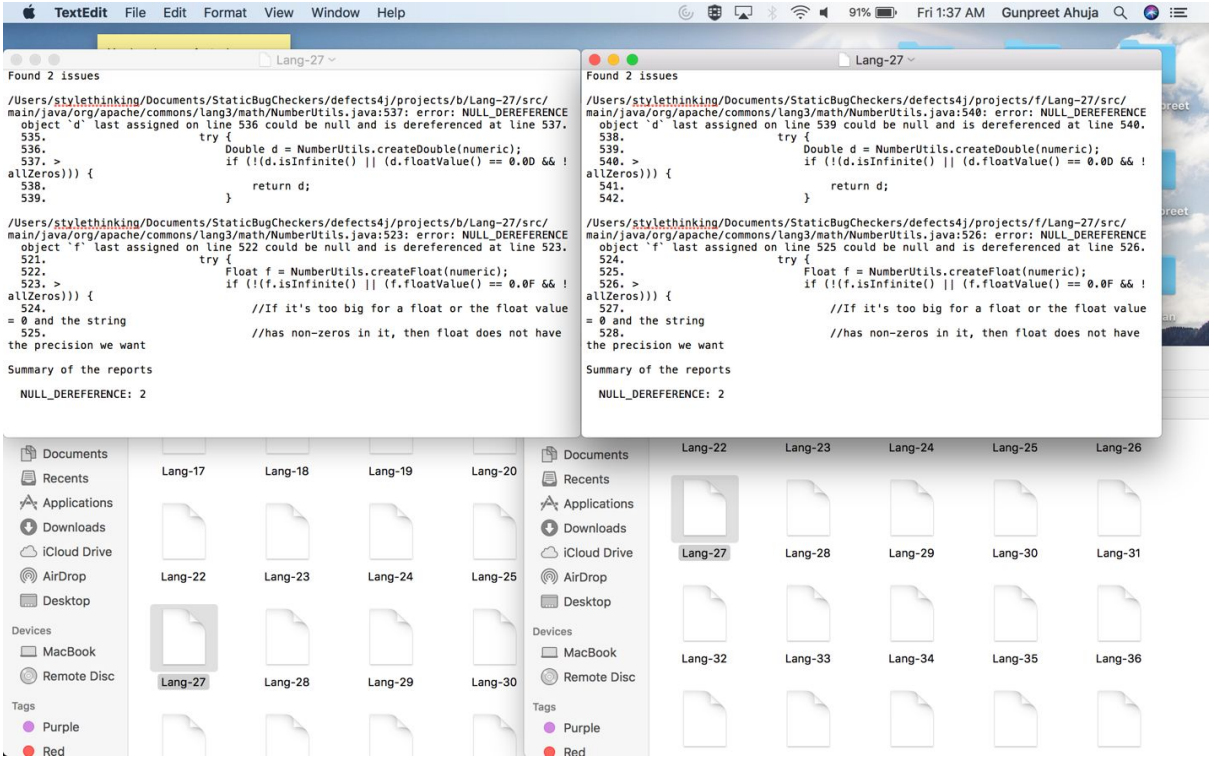
Lang-27:



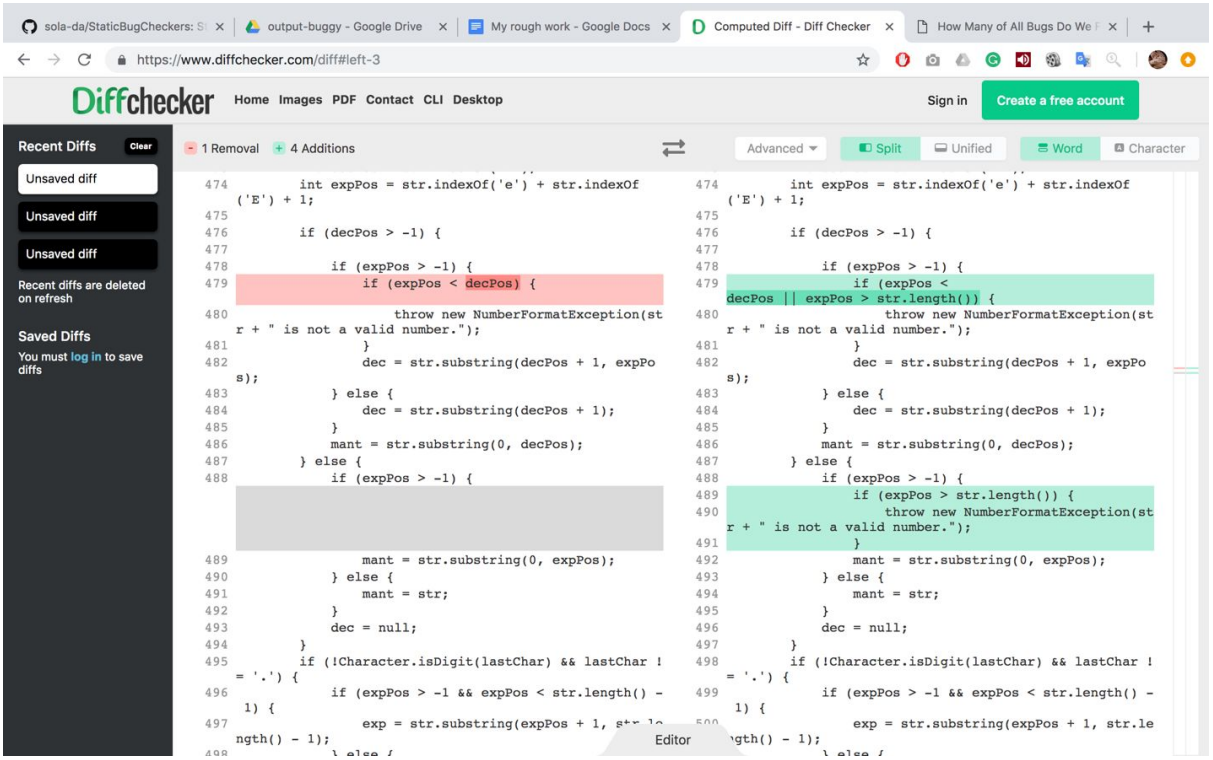
Infer: meth 1 (false positive)



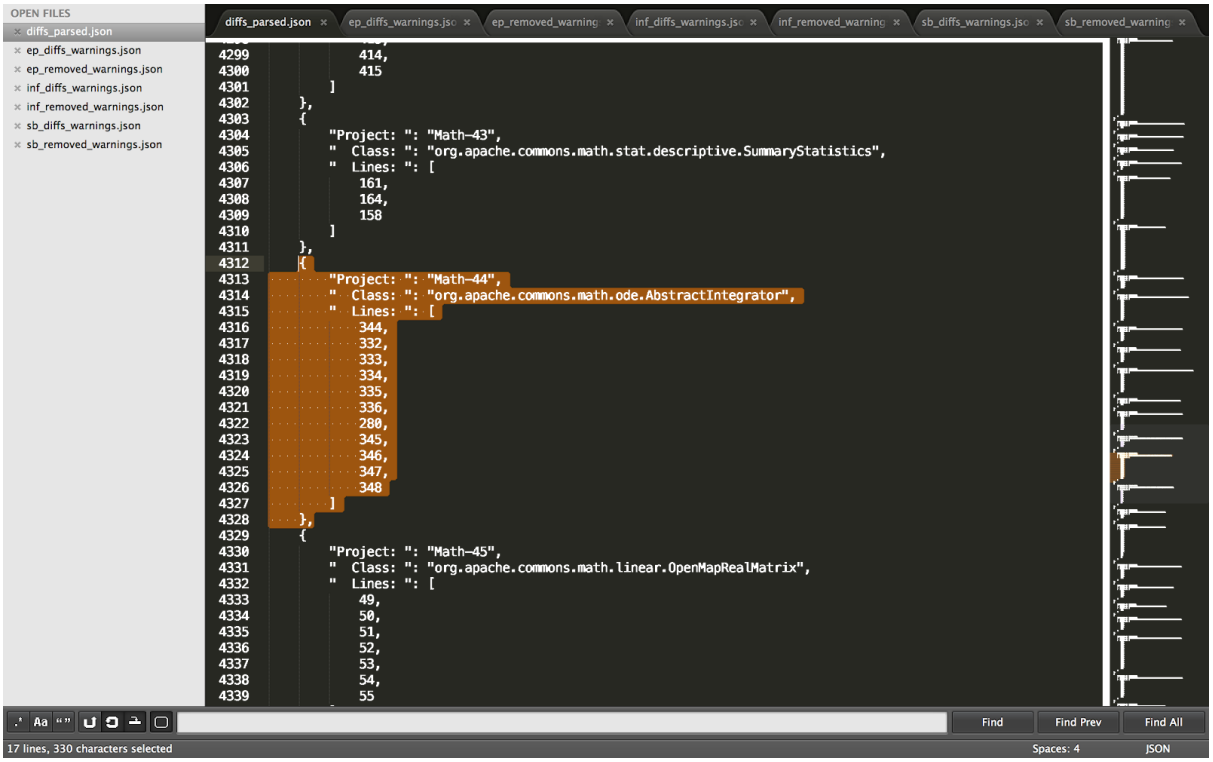
Buggy warnings: fixed warnings:



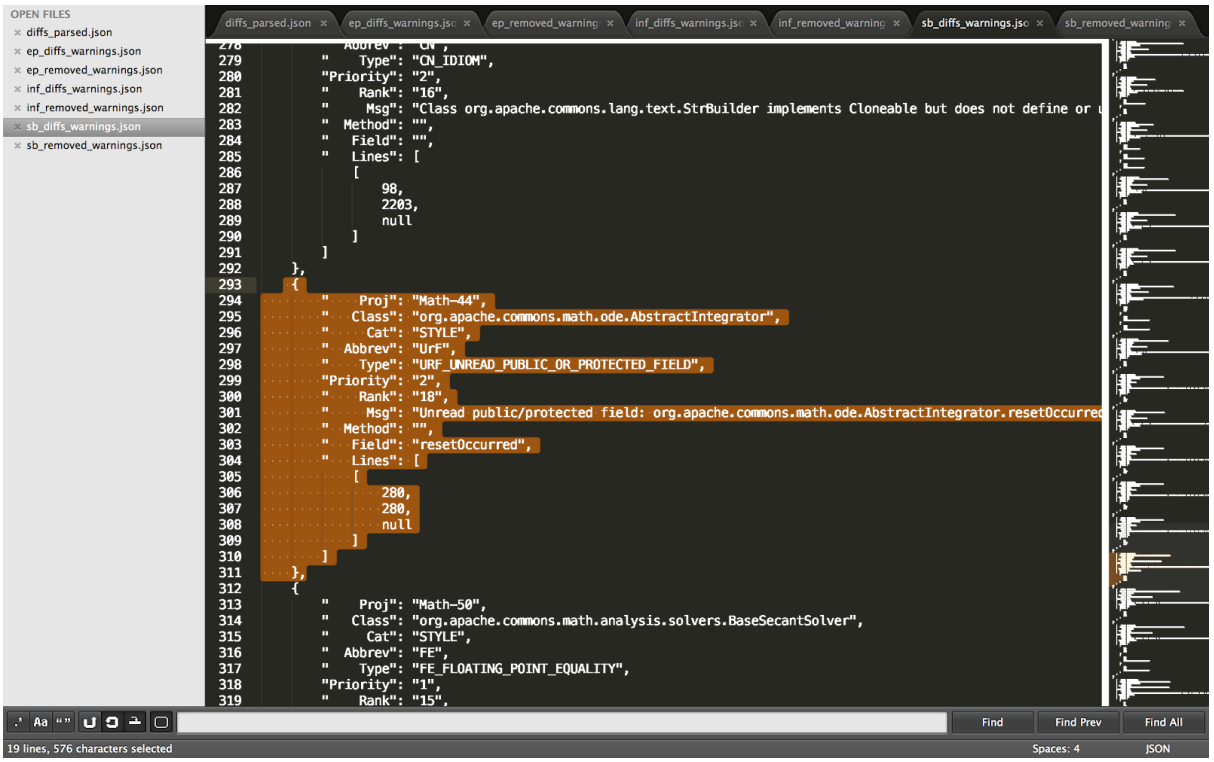
code diff:



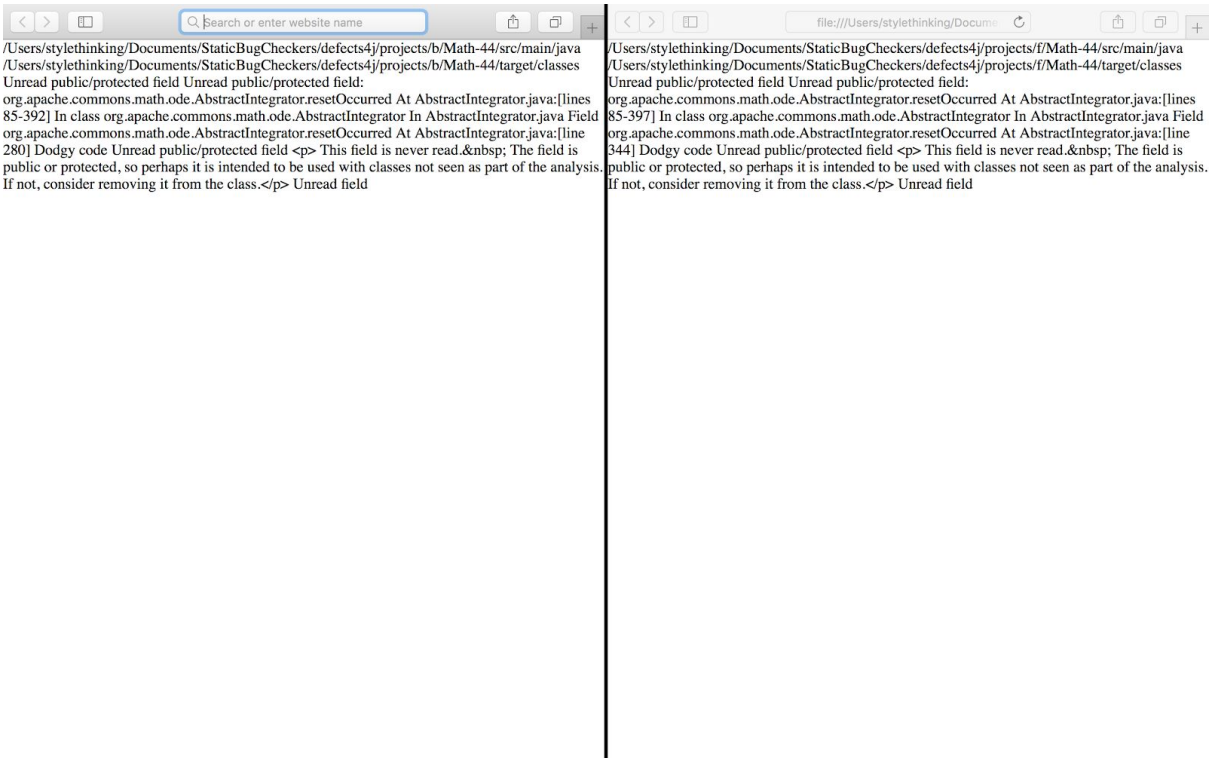
Math-44:



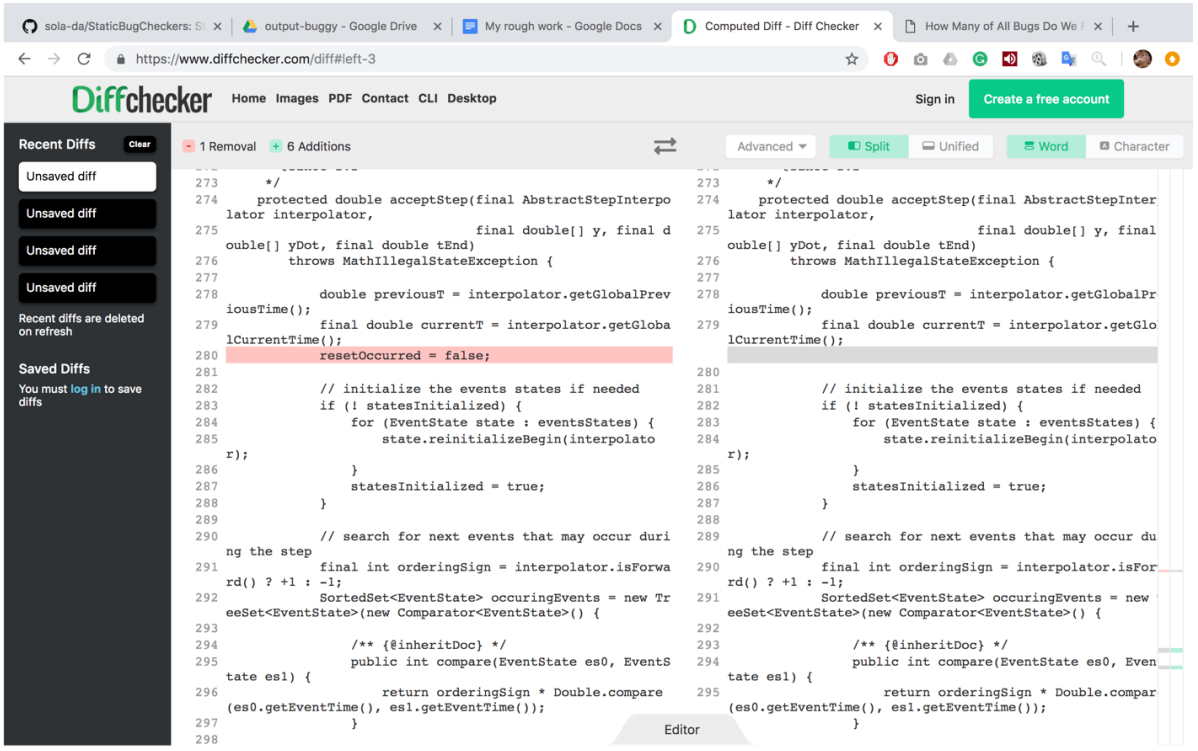
Spot bugs: meth 1 (full match)



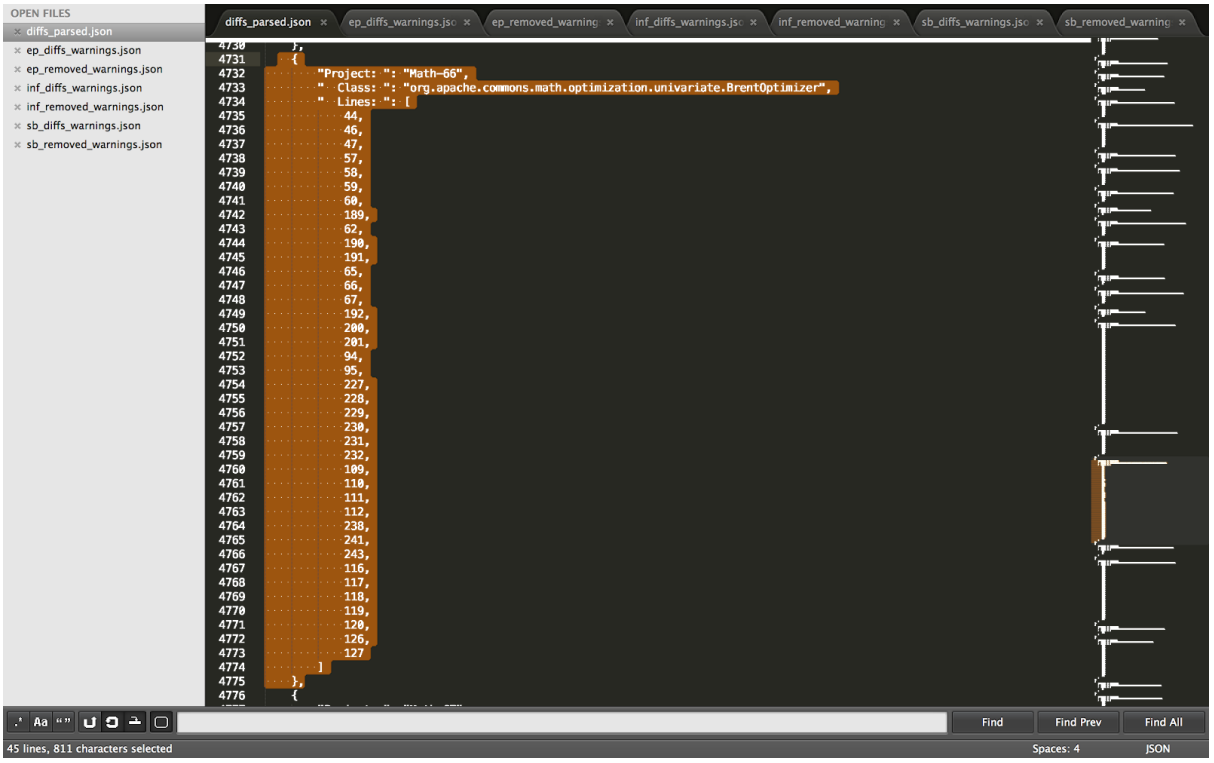
Warnings: shifted to next occurrence



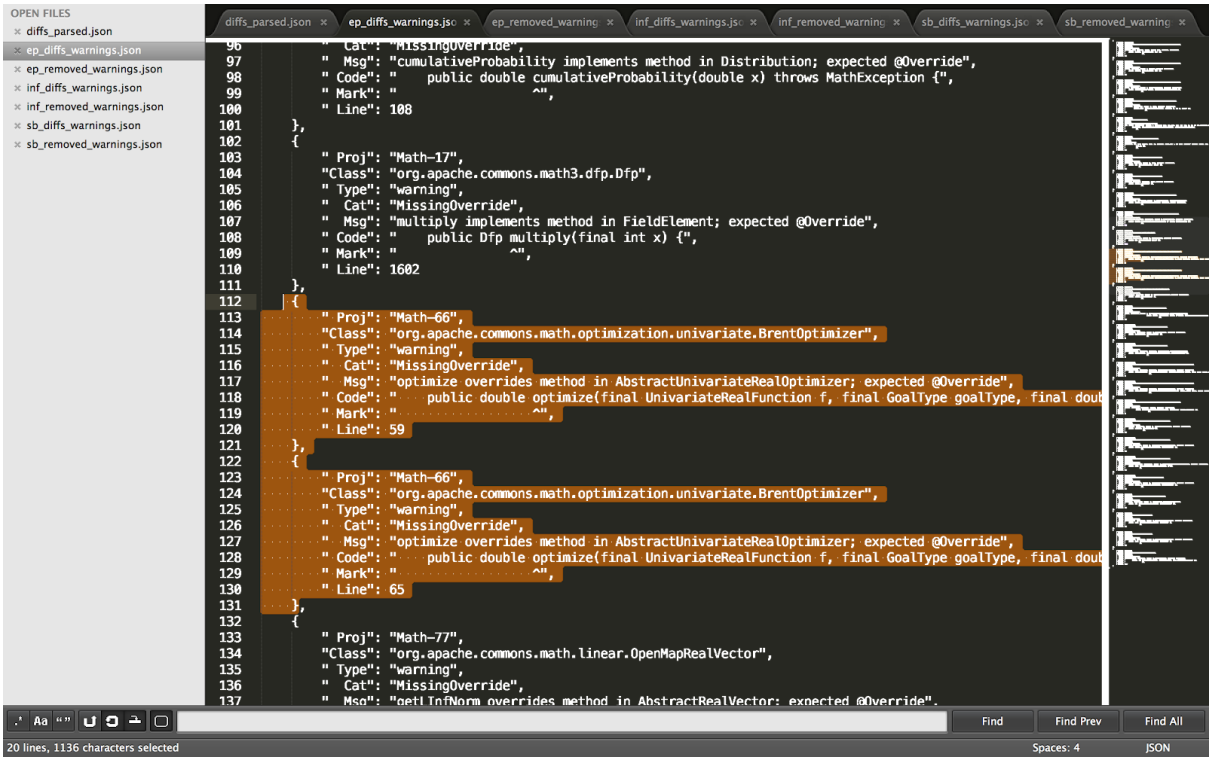
Code diff:



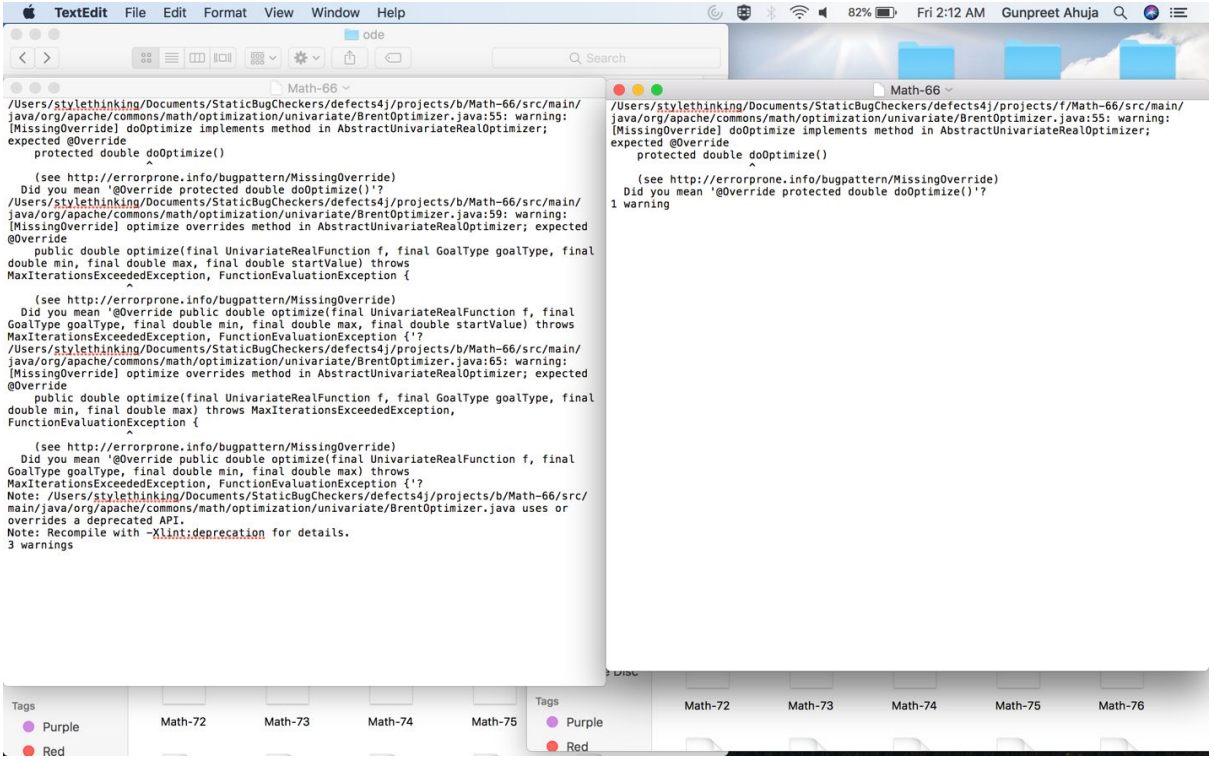
Math-66:



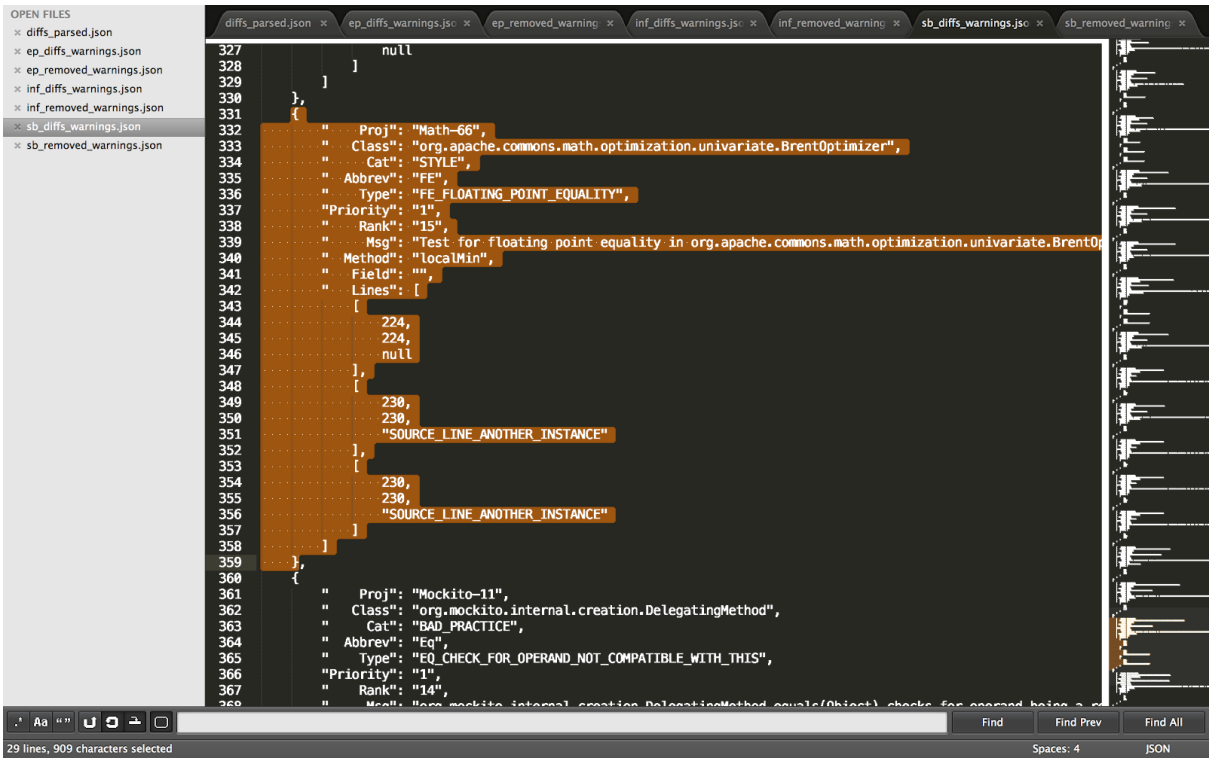
Error-prone: meth 1,2 (full match)



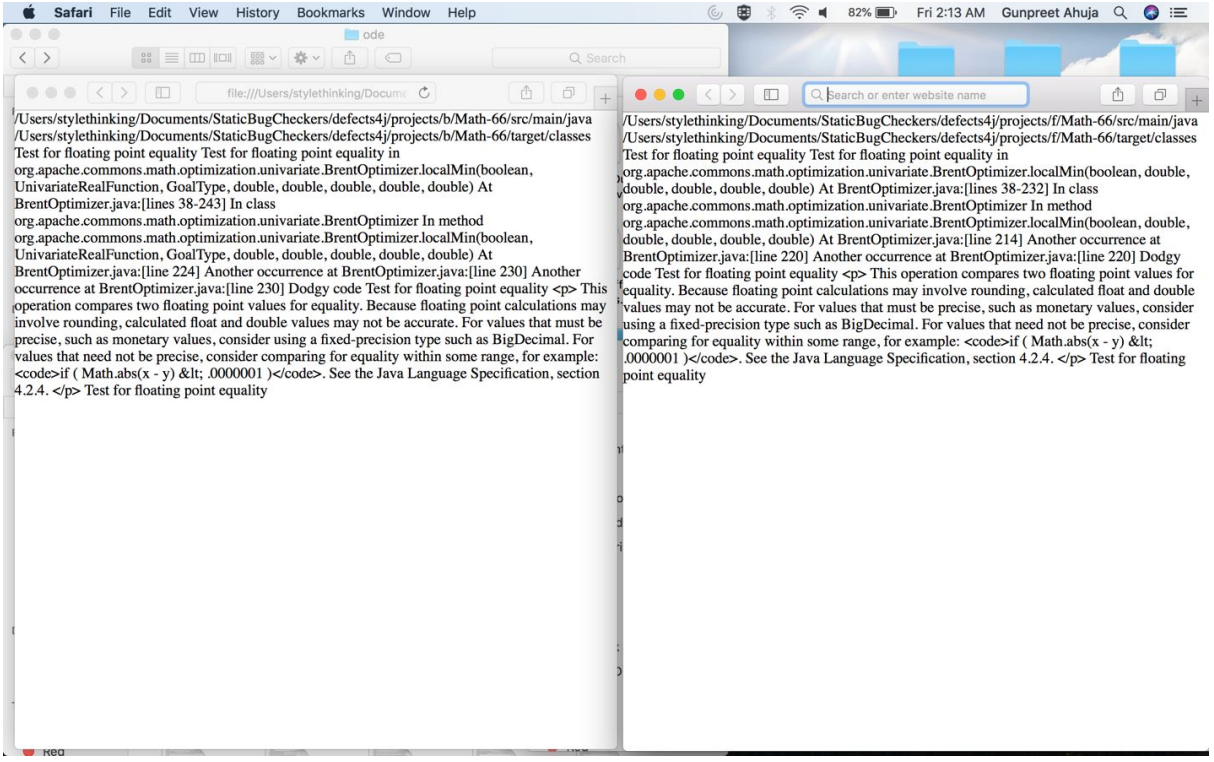
Warnings:



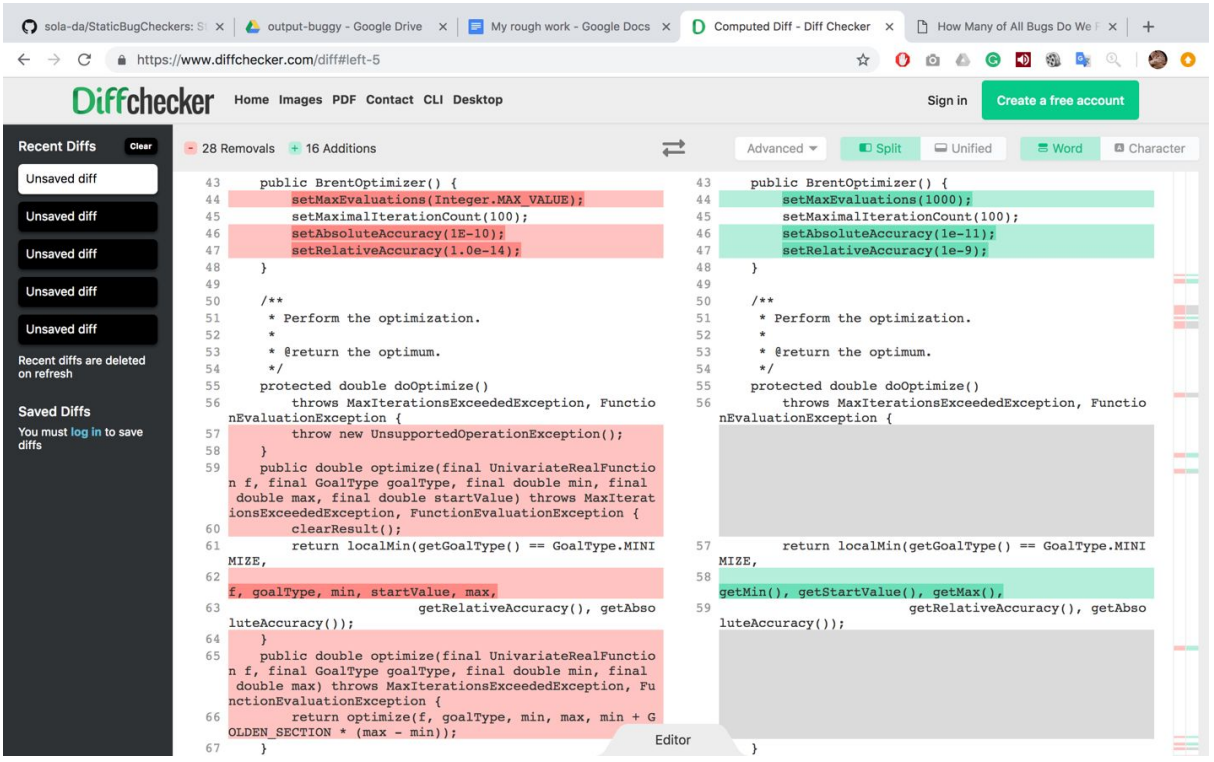
Spotbugs: meth 1,2 (false positive)

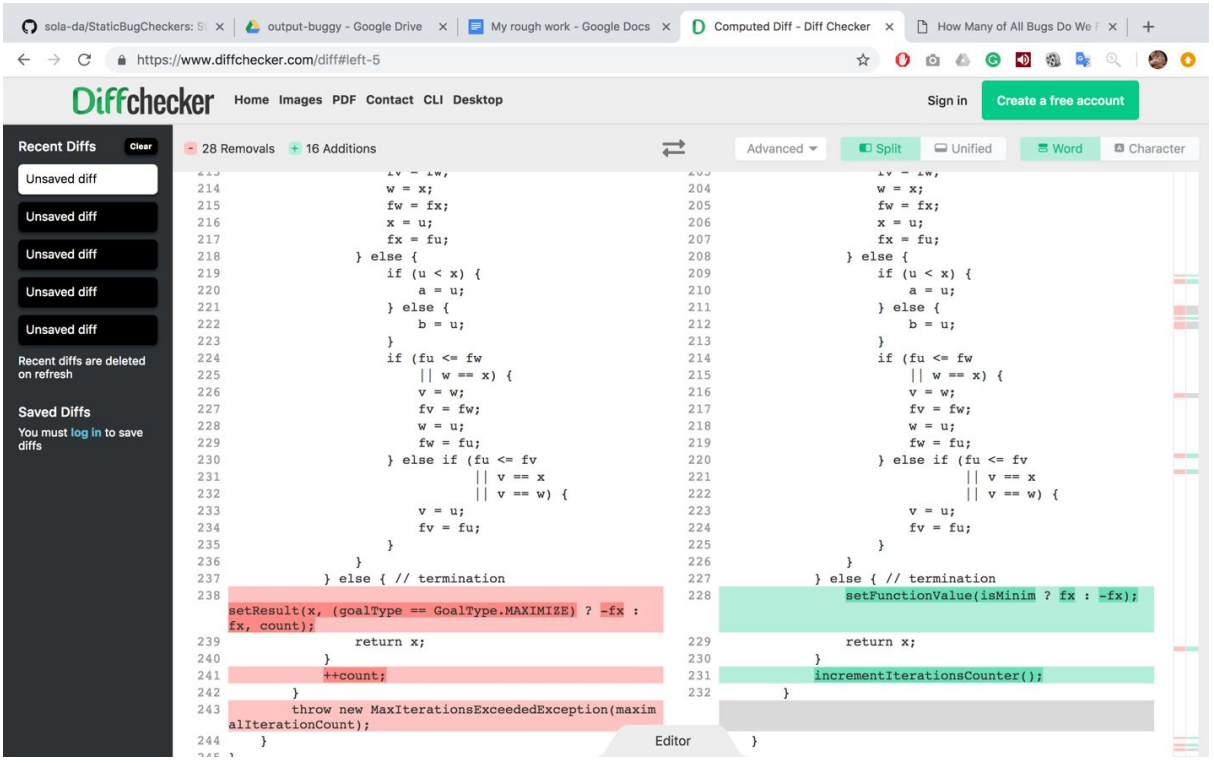


Warnings:

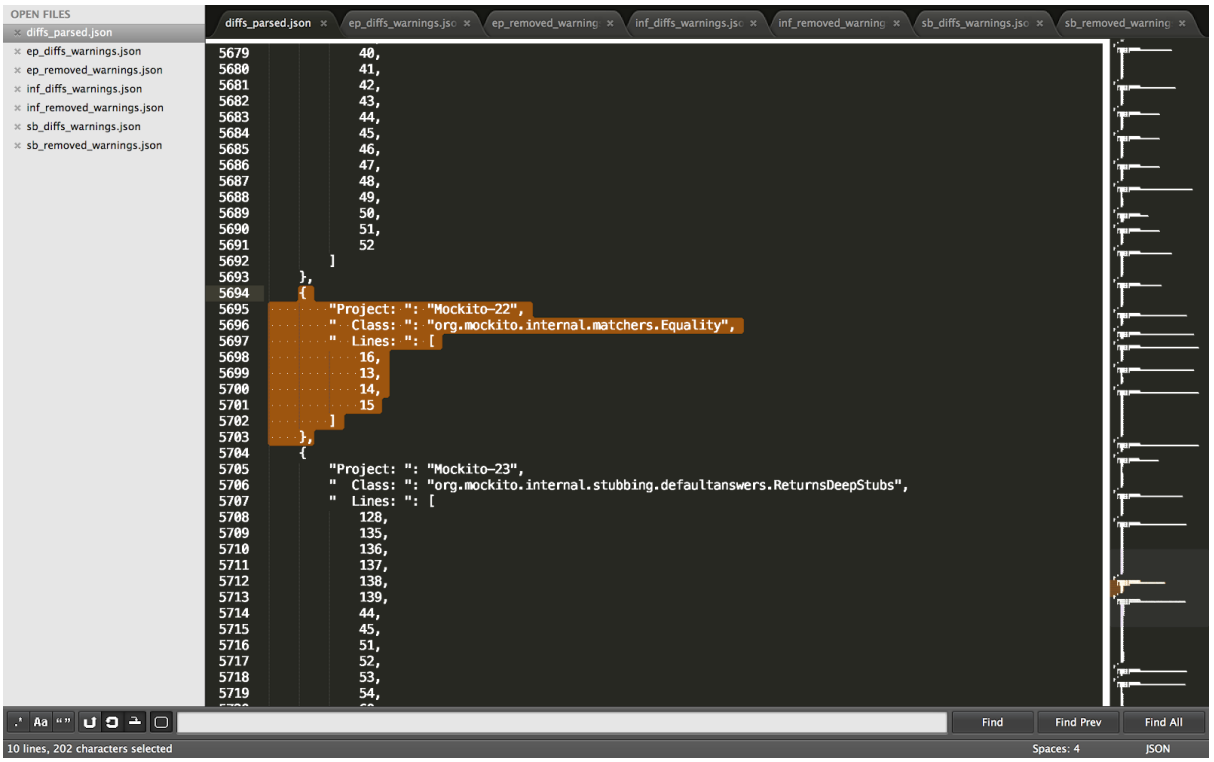


Code Diff:

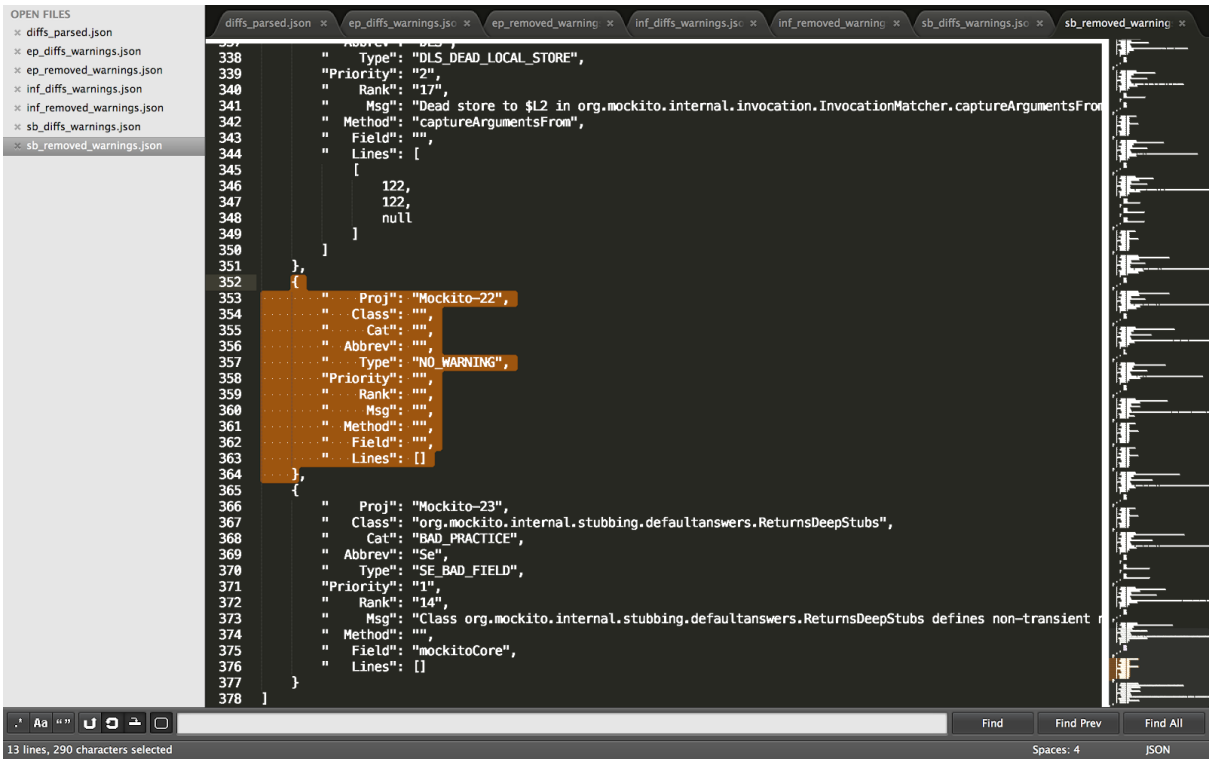

sola-da/StaticBugCheckers: S
output-buggy - Google Drive
My rough work - Google Docs
Computed Diff - Diff Checker
How Many of All Bugs Do We F
https://www.diffchecker.com/diff#left-5
Diffchecker
Home Images PDF Contact CLI Desktop
Sign in
Create a free account
Recent Diffs
Clear
Unsaved diff
Unsaved diff
Unsaved diff
Unsaved diff
Unsaved diff
Recent diffs are deleted on refresh
Saved Diffs
You must log in to save diffs
28 Removals
16 Additions
Advanced
Split
Unified
Word
Character
Editor


sola-da/StaticBugCheckers: S
output-buggy - Google Drive
My rough work - Google Docs
Computed Diff - Diff Checker
How Many of All Bugs Do We F
https://www.diffchecker.com/diff#left-5
Diffchecker
Home Images PDF Contact CLI Desktop
Sign in
Create a free account
Recent Diffs
Clear
Unsaved diff
Unsaved diff
Unsaved diff
Unsaved diff
Unsaved diff
Recent diffs are deleted on refresh
Saved Diffs
You must log in to save diffs
28 Removals
16 Additions
Advanced
Split
Unified
Word
Character
Editor

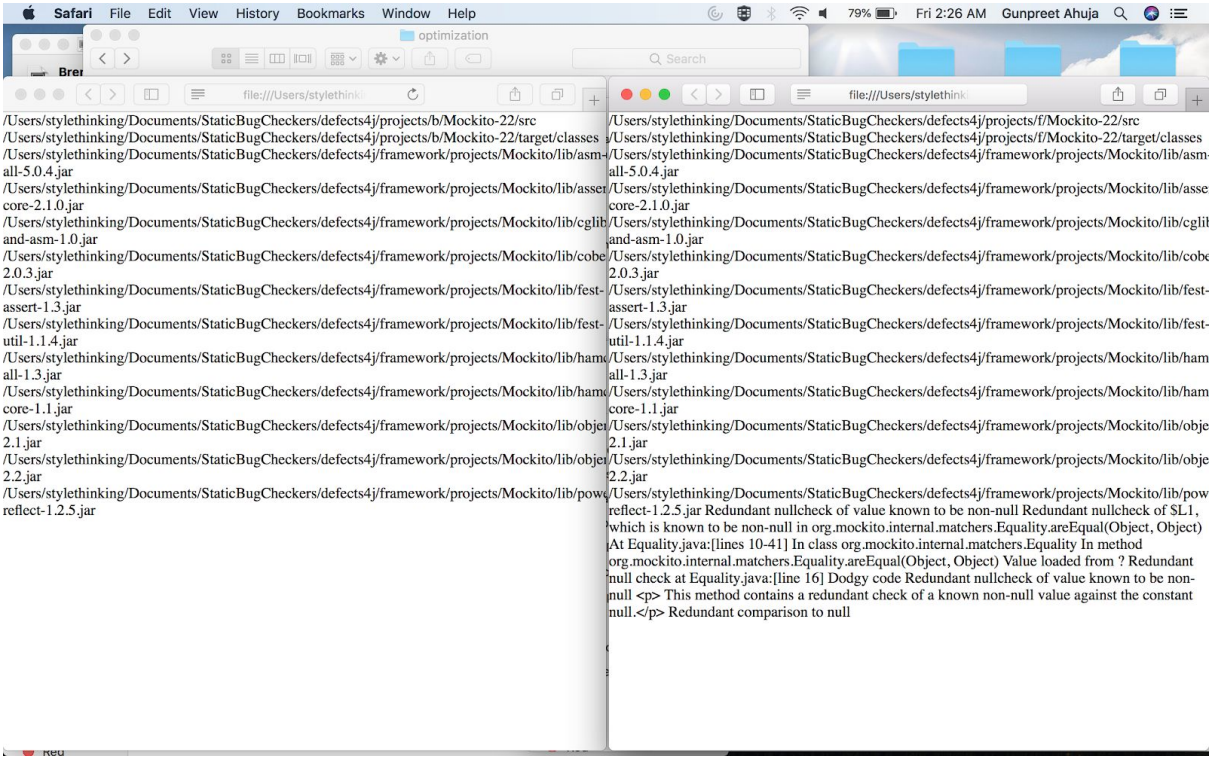
Mockito-22:



Spot bugs: no warning (meth 2)



Warnings:



Code Diff: Domain-specific