

#### 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) {
742
743             this.maybeAliased =
744                 !assignNode.getParent().isExprResult();
745             this.mayHaveSecondarySideEffects =
746                 maybeAliased ||
747
748             !exprResult.isSecondarySideEffect();
749
750             if (exprResult.isSecondarySideEffect()) {
751                 maybeAliased = false;
752             }
753         }
754     }
755 }
```

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)
746         && hasPropertyAssign) {
747
748         this.maybeAliased =
749             NodeUtil.isExpressionResultUsed(assignNode);
750         this.mayHaveSecondarySideEffects =
751             NodeUtil.isAssignmentWithSideEffects(assign);
752     }
753 }
```

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, FunctionEvaluationException {
60         clearResult();
61         return localMin(goalType() == 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     }
68
69     /**
70      * Local minimization routine.
71      *
72      * @param isMinim true if we are minimizing, false if maximizing.
73      * @param f the function to minimize/maximize.
74      * @param goalType the goal type (minimize or maximize).
75      * @param lo the lower bound of the search interval.
76      * @param mid the midpoint of the search interval.
77      * @param hi the upper bound of the search interval.
78      * @param eps the tolerance for convergence.
79      * @param t the tolerance for the width of the search interval.
80      */
81     private double localMin(boolean isMinim,
82                            UnivariateRealFunction f,
83                            GoalType goalType,
84                            double lo, double mid, double hi,
85                            double eps, double t)
86         throws MaxIterationsExceededException, FunctionEvaluationException {
87         if (eps <= 0) {
88             throw new NotStrictlyPositiveException(eps);
89         }
90         if (t <= 0) {
91             throw new NotStrictlyPositiveException(t);
92         }
93         double a, b;
94         if (lo < hi) {
95             a = lo;
96             b = hi;
97         } else {
98             a = hi;
99             b = lo;
100        }
101
102        double x = mid;
103        double v = x;
104        double w = x;
105        double d = 0;
106        double e = 0;
107        double fx = computeObjectiveValue(f, x);
108        if (goalType == GoalType.MAXIMIZE) {
109            fx = -fx;
110        }
111        double fv = fx;
112        double fw = fx;
113
114        int count = 0;
115        while (count < maximalIterationCount) {
116            double m = 0.5 * (a + b);
117
118            double xm = m;
119            if (isMinim) {
120                xm = (xm - w) / 2.0;
121            } else {
122                xm = (xm + w) / 2.0;
123            }
124
125            double xm_fx = computeObjectiveValue(f, xm);
126            if (xm_fx < fx) {
127                fx = xm_fx;
128                a = xm;
129            } else {
130                fx = xm_fx;
131                b = xm;
132            }
133
134            count++;
135        }
136    }
137
138    /**
139     * Compute the objective value of the function at the specified point.
140     *
141     * @param f the function to evaluate.
142     * @param x the point at which to evaluate the function.
143     * @return the objective value.
144     */
145    protected double computeObjectiveValue(UnivariateRealFunction f, double x) {
146        return f.value(x);
147    }
148
149    /**
150     * Set the maximum number of evaluations.
151     *
152     * @param maxEvaluations the maximum number of evaluations.
153     */
154    public void setMaxEvaluations(int maxEvaluations) {
155        this.maxEvaluations = maxEvaluations;
156    }
157
158    /**
159     * Get the maximum number of evaluations.
160     *
161     * @return the maximum number of evaluations.
162     */
163    public int getMaxEvaluations() {
164        return maxEvaluations;
165    }
166
167    /**
168     * Set the maximal iteration count.
169     *
170     * @param maximalIterationCount the maximal iteration count.
171     */
172    public void setMaximalIterationCount(int maximalIterationCount) {
173        this.maximalIterationCount = maximalIterationCount;
174    }
175
176    /**
177     * Get the maximal iteration count.
178     *
179     * @return the maximal iteration count.
180     */
181    public int getMaximalIterationCount() {
182        return maximalIterationCount;
183    }
184
185    /**
186     * Set the absolute accuracy.
187     *
188     * @param absoluteAccuracy the absolute accuracy.
189     */
190    public void setAbsoluteAccuracy(double absoluteAccuracy) {
191        this.absoluteAccuracy = absoluteAccuracy;
192    }
193
194    /**
195     * Get the absolute accuracy.
196     *
197     * @return the absolute accuracy.
198     */
199    public double getAbsoluteAccuracy() {
200        return absoluteAccuracy;
201    }
202
203    /**
204     * Set the relative accuracy.
205     *
206     * @param relativeAccuracy the relative accuracy.
207     */
208    public void setRelativeAccuracy(double relativeAccuracy) {
209        this.relativeAccuracy = relativeAccuracy;
210    }
211
212    /**
213     * Get the relative accuracy.
214     *
215     * @return the relative accuracy.
216     */
217    public double getRelativeAccuracy() {
218        return relativeAccuracy;
219    }
220
221    /**
222     * Clear the result.
223     */
224    protected void clearResult() {
225        result = null;
226    }
227
228    /**
229     * Get the result.
230     *
231     * @return the result.
232     */
233    public Object getResult() {
234        return result;
235    }
236
237    /**
238     * Set the result.
239     *
240     * @param result the result.
241     */
242    public void setResult(Object result) {
243        this.result = result;
244    }
245
246    /**
247     * Get the goal type.
248     *
249     * @return the goal type.
250     */
251    public GoalType getGoalType() {
252        return goalType;
253    }
254
255    /**
256     * Set the goal type.
257     *
258     * @param goalType the goal type.
259     */
260    public void setGoalType(GoalType goalType) {
261        this.goalType = goalType;
262    }
263
264    /**
265     * Get the minimum value.
266     *
267     * @return the minimum value.
268     */
269    public double getMin() {
270        return min;
271    }
272
273    /**
274     * Set the minimum value.
275     *
276     * @param min the minimum value.
277     */
278    public void setMin(double min) {
279        this.min = min;
280    }
281
282    /**
283     * Get the maximum value.
284     *
285     * @return the maximum value.
286     */
287    public double getMax() {
288        return max;
289    }
290
291    /**
292     * Set the maximum value.
293     *
294     * @param max the maximum value.
295     */
296    public void setMax(double max) {
297        this.max = max;
298    }
299
300    /**
301     * Get the start value.
302     *
303     * @return the start value.
304     */
305    public double getStartValue() {
306        return startValue;
307    }
308
309    /**
310     * Set the start value.
311     *
312     * @param startValue the start value.
313     */
314    public void setStartValue(double startValue) {
315        this.startValue = startValue;
316    }
317
318    /**
319     * Get the local minimum.
320     *
321     * @return the local minimum.
322     */
323    public double getLocalMin() {
324        return localMin;
325    }
326
327    /**
328     * Set the local minimum.
329     *
330     * @param localMin the local minimum.
331     */
332    public void setLocalMin(double localMin) {
333        this.localMin = localMin;
334    }
335
336    /**
337     * Get the local maximum.
338     *
339     * @return the local maximum.
340     */
341    public double getLocalMax() {
342        return localMax;
343    }
344
345    /**
346     * Set the local maximum.
347     *
348     * @param localMax the local maximum.
349     */
350    public void setLocalMax(double localMax) {
351        this.localMax = localMax;
352    }
353
354    /**
355     * Get the local width.
356     *
357     * @return the local width.
358     */
359    public double getLocalWidth() {
360        return localWidth;
361    }
362
363    /**
364     * Set the local width.
365     *
366     * @param localWidth the local width.
367     */
368    public void setLocalWidth(double localWidth) {
369        this.localWidth = localWidth;
370    }
371
372    /**
373     * Get the local height.
374     *
375     * @return the local height.
376     */
377    public double getLocalHeight() {
378        return localHeight;
379    }
380
381    /**
382     * Set the local height.
383     *
384     * @param localHeight the local height.
385     */
386    public void setLocalHeight(double localHeight) {
387        this.localHeight = localHeight;
388    }
389
390    /**
391     * Get the local slope.
392     *
393     * @return the local slope.
394     */
395    public double getLocalSlope() {
396        return localSlope;
397    }
398
399    /**
400     * Set the local slope.
401     *
402     * @param localSlope the local slope.
403     */
404    public void setLocalSlope(double localSlope) {
405        this.localSlope = localSlope;
406    }
407
408    /**
409     * Get the local curvature.
410     *
411     * @return the local curvature.
412     */
413    public double getLocalCurvature() {
414        return localCurvature;
415    }
416
417    /**
418     * Set the local curvature.
419     *
420     * @param localCurvature the local curvature.
421     */
422    public void setLocalCurvature(double localCurvature) {
423        this.localCurvature = localCurvature;
424    }
425
426    /**
427     * Get the local inflection point.
428     *
429     * @return the local inflection point.
430     */
431    public double getLocalInflectionPoint() {
432        return localInflectionPoint;
433    }
434
435    /**
436     * Set the local inflection point.
437     *
438     * @param localInflectionPoint the local inflection point.
439     */
440    public void setLocalInflectionPoint(double localInflectionPoint) {
441        this.localInflectionPoint = localInflectionPoint;
442    }
443
444    /**
445     * Get the local concavity.
446     *
447     * @return the local concavity.
448     */
449    public double getLocalConcavity() {
450        return localConcavity;
451    }
452
453    /**
454     * Set the local concavity.
455     *
456     * @param localConcavity the local concavity.
457     */
458    public void setLocalConcavity(double localConcavity) {
459        this.localConcavity = localConcavity;
460    }
461
462    /**
463     * Get the local convexity.
464     *
465     * @return the local convexity.
466     */
467    public double getLocalConvexity() {
468        return localConvexity;
469    }
470
471    /**
472     * Set the local convexity.
473     *
474     * @param localConvexity the local convexity.
475     */
476    public void setLocalConvexity(double localConvexity) {
477        this.localConvexity = localConvexity;
478    }
479
480    /**
481     * Get the local curvature sign.
482     *
483     * @return the local curvature sign.
484     */
485    public double getLocalCurvatureSign() {
486        return localCurvatureSign;
487    }
488
489    /**
490     * Set the local curvature sign.
491     *
492     * @param localCurvatureSign the local curvature sign.
493     */
494    public void setLocalCurvatureSign(double localCurvatureSign) {
495        this.localCurvatureSign = localCurvatureSign;
496    }
497
498    /**
499     * Get the local inflection point sign.
500     *
501     * @return the local inflection point sign.
502     */
503    public double getLocalInflectionPointSign() {
504        return localInflectionPointSign;
505    }
506
507    /**
508     * Set the local inflection point sign.
509     *
510     * @param localInflectionPointSign the local inflection point sign.
511     */
512    public void setLocalInflectionPointSign(double localInflectionPointSign) {
513        this.localInflectionPointSign = localInflectionPointSign;
514    }
515
516    /**
517     * Get the local concavity sign.
518     *
519     * @return the local concavity sign.
520     */
521    public double getLocalConcavitySign() {
522        return localConcavitySign;
523    }
524
525    /**
526     * Set the local concavity sign.
527     *
528     * @param localConcavitySign the local concavity sign.
529     */
530    public void setLocalConcavitySign(double localConcavitySign) {
531        this.localConcavitySign = localConcavitySign;
532    }
533
534    /**
535     * Get the local convexity sign.
536     *
537     * @return the local convexity sign.
538     */
539    public double getLocalConvexitySign() {
540        return localConvexitySign;
541    }
542
543    /**
544     * Set the local convexity sign.
545     *
546     * @param localConvexitySign the local convexity sign.
547     */
548    public void setLocalConvexitySign(double localConvexitySign) {
549        this.localConvexitySign = localConvexitySign;
550    }
551
552    /**
553     * Get the local curvature sign.
554     *
555     * @return the local curvature sign.
556     */
557    public double getLocalCurvatureSign() {
558        return localCurvatureSign;
559    }
560
561    /**
562     * Set the local curvature sign.
563     *
564     * @param localCurvatureSign the local curvature sign.
565     */
566    public void setLocalCurvatureSign(double localCurvatureSign) {
567        this.localCurvatureSign = localCurvatureSign;
568    }
569
570    /**
571     * Get the local inflection point sign.
572     *
573     * @return the local inflection point sign.
574     */
575    public double getLocalInflectionPointSign() {
576        return localInflectionPointSign;
577    }
578
579    /**
580     * Set the local inflection point sign.
581     *
582     * @param localInflectionPointSign the local inflection point sign.
583     */
584    public void setLocalInflectionPointSign(double localInflectionPointSign) {
585        this.localInflectionPointSign = localInflectionPointSign;
586    }
587
588    /**
589     * Get the local concavity sign.
590     *
591     * @return the local concavity sign.
592     */
593    public double getLocalConcavitySign() {
594        return localConcavitySign;
595    }
596
597    /**
598     * Set the local concavity sign.
599     *
600     * @param localConcavitySign the local concavity sign.
601     */
602    public void setLocalConcavitySign(double localConcavitySign) {
603        this.localConcavitySign = localConcavitySign;
604    }
605
606    /**
607     * Get the local convexity sign.
608     *
609     * @return the local convexity sign.
610     */
611    public double getLocalConvexitySign() {
612        return localConvexitySign;
613    }
614
615    /**
616     * Set the local convexity sign.
617     *
618     * @param localConvexitySign the local convexity sign.
619     */
620    public void setLocalConvexitySign(double localConvexitySign) {
621        this.localConvexitySign = localConvexitySign;
622    }
623
624    /**
625     * Get the local curvature sign.
626     *
627     * @return the local curvature sign.
628     */
629    public double getLocalCurvatureSign() {
630        return localCurvatureSign;
631    }
632
633    /**
634     * Set the local curvature sign.
635     *
636     * @param localCurvatureSign the local curvature sign.
637     */
638    public void setLocalCurvatureSign(double localCurvatureSign) {
639        this.localCurvatureSign = localCurvatureSign;
640    }
641
642    /**
643     * Get the local inflection point sign.
644     *
645     * @return the local inflection point sign.
646     */
647    public double getLocalInflectionPointSign() {
648        return localInflectionPointSign;
649    }
650
651    /**
652     * Set the local inflection point sign.
653     *
654     * @param localInflectionPointSign the local inflection point sign.
655     */
656    public void setLocalInflectionPointSign(double localInflectionPointSign) {
657        this.localInflectionPointSign = localInflectionPointSign;
658    }
659
660    /**
661     * Get the local concavity sign.
662     *
663     * @return the local concavity sign.
664     */
665    public double getLocalConcavitySign() {
666        return localConcavitySign;
667    }
668
669    /**
670     * Set the local concavity sign.
671     *
672     * @param localConcavitySign the local concavity sign.
673     */
674    public void setLocalConcavitySign(double localConcavitySign) {
675        this.localConcavitySign = localConcavitySign;
676    }
677
678    /**
679     * Get the local convexity sign.
680     *
681     * @return the local convexity sign.
682     */
683    public double getLocalConvexitySign() {
684        return localConvexitySign;
685    }
686
687    /**
688     * Set the local convexity sign.
689     *
690     * @param localConvexitySign the local convexity sign.
691     */
692    public void setLocalConvexitySign(double localConvexitySign) {
693        this.localConvexitySign = localConvexitySign;
694    }
695
696    /**
697     * Get the local curvature sign.
698     *
699     * @return the local curvature sign.
700     */
701    public double getLocalCurvatureSign() {
702        return localCurvatureSign;
703    }
704
705    /**
706     * Set the local curvature sign.
707     *
708     * @param localCurvatureSign the local curvature sign.
709     */
710    public void setLocalCurvatureSign(double localCurvatureSign) {
711        this.localCurvatureSign = localCurvatureSign;
712    }
713
714    /**
715     * Get the local inflection point sign.
716     *
717     * @return the local inflection point sign.
718     */
719    public double getLocalInflectionPointSign() {
720        return localInflectionPointSign;
721    }
722
723    /**
724     * Set the local inflection point sign.
725     *
726     * @param localInflectionPointSign the local inflection point sign.
727     */
728    public void setLocalInflectionPointSign(double localInflectionPointSign) {
729        this.localInflectionPointSign = localInflectionPointSign;
730    }
731
732    /**
733     * Get the local concavity sign.
734     *
735     * @return the local concavity sign.
736     */
737    public double getLocalConcavitySign() {
738        return localConcavitySign;
739    }
740
741    /**
742     * Set the local concavity sign.
743     *
744     * @param localConcavitySign the local concavity sign.
745     */
746    public void setLocalConcavitySign(double localConcavitySign) {
747        this.localConcavitySign = localConcavitySign;
748    }
749
750    /**
751     * Get the local convexity sign.
752     *
753     * @return the local convexity sign.
754     */
755    public double getLocalConvexitySign() {
756        return localConvexitySign;
757    }
758
759    /**
760     * Set the local convexity sign.
761     *
762     * @param localConvexitySign the local convexity sign.
763     */
764    public void setLocalConvexitySign(double localConvexitySign) {
765        this.localConvexitySign = localConvexitySign;
766    }
767
768    /**
769     * Get the local curvature sign.
770     *
771     * @return the local curvature sign.
772     */
773    public double getLocalCurvatureSign() {
774        return localCurvatureSign;
775    }
776
777    /**
778     * Set the local curvature sign.
779     *
780     * @param localCurvatureSign the local curvature sign.
781     */
782    public void setLocalCurvatureSign(double localCurvatureSign) {
783        this.localCurvatureSign = localCurvatureSign;
784    }
785
786    /**
787     * Get the local inflection point sign.
788     *
789     * @return the local inflection point sign.
790     */
791    public double getLocalInflectionPointSign() {
792        return localInflectionPointSign;
793    }
794
795    /**
796     * Set the local inflection point sign.
797     *
798     * @param localInflectionPointSign the local inflection point sign.
799     */
800    public void setLocalInflectionPointSign(double localInflectionPointSign) {
801        this.localInflectionPointSign = localInflectionPointSign;
802    }
803
804    /**
805     * Get the local concavity sign.
806     *
807     * @return the local concavity sign.
808     */
809    public double getLocalConcavitySign() {
810        return localConcavitySign;
811    }
812
813    /**
814     * Set the local concavity sign.
815     *
816     * @param localConcavitySign the local concavity sign.
817     */
818    public void setLocalConcavitySign(double localConcavitySign) {
819        this.localConcavitySign = localConcavitySign;
820    }
821
822    /**
823     * Get the local convexity sign.
824     *
825     * @return the local convexity sign.
826     */
827    public double getLocalConvexitySign() {
828        return localConvexitySign;
829    }
830
831    /**
832     * Set the local convexity sign.
833     *
834     * @param localConvexitySign the local convexity sign.
835     */
836    public void setLocalConvexitySign(double localConvexitySign) {
837        this.localConvexitySign = localConvexitySign;
838    }
839
840    /**
841     * Get the local curvature sign.
842     *
843     * @return the local curvature sign.
844     */
845    public double getLocalCurvatureSign() {
846        return localCurvatureSign;
847    }
848
849    /**
850     * Set the local curvature sign.
851     *
852     * @param localCurvatureSign the local curvature sign.
853     */
854    public void setLocalCurvatureSign(double localCurvatureSign) {
855        this.localCurvatureSign = localCurvatureSign;
856    }
857
858    /**
859     * Get the local inflection point sign.
860     *
861     * @return the local inflection point sign.
862     */
863    public double getLocalInflectionPointSign() {
864        return localInflectionPointSign;
865    }
866
867    /**
868     * Set the local inflection point sign.
869     *
870     * @param localInflectionPointSign the local inflection point sign.
871     */
872    public void setLocalInflectionPointSign(double localInflectionPointSign) {
873        this.localInflectionPointSign = localInflectionPointSign;
874    }
875
876    /**
877     * Get the local concavity sign.
878     *
879     * @return the local concavity sign.
880     */
881    public double getLocalConcavitySign() {
882        return localConcavitySign;
883    }
884
885    /**
886     * Set the local concavity sign.
887     *
888     * @param localConcavitySign the local concavity sign.
889     */
890    public void setLocalConcavitySign(double localConcavitySign) {
891        this.localConcavitySign = localConcavitySign;
892    }
893
894    /**
895     * Get the local convexity sign.
896     *
897     * @return the local convexity sign.
898     */
899    public double getLocalConvexitySign() {
900        return localConvexitySign;
901    }
902
903    /**
904     * Set the local convexity sign.
905     *
906     * @param localConvexitySign the local convexity sign.
907     */
908    public void setLocalConvexitySign(double localConvexitySign) {
909        this.localConvexitySign = localConvexitySign;
910    }
911
912    /**
913     * Get the local curvature sign.
914     *
915     * @return the local curvature sign.
916     */
917    public double getLocalCurvatureSign() {
918        return localCurvatureSign;
919    }
920
921    /**
922     * Set the local curvature sign.
923     *
924     * @param localCurvatureSign the local curvature sign.
925     */
926    public void setLocalCurvatureSign(double localCurvatureSign) {
927        this.localCurvatureSign = localCurvatureSign;
928    }
929
930    /**
931     * Get the local inflection point sign.
932     *
933     * @return the local inflection point sign.
934     */
935    public double getLocalInflectionPointSign() {
936        return localInflectionPointSign;
937    }
938
939    /**
940     * Set the local inflection point sign.
941     *
942     * @param localInflectionPointSign the local inflection point sign.
943     */
944    public void setLocalInflectionPointSign(double localInflectionPointSign) {
945        this.localInflectionPointSign = localInflectionPointSign;
946    }
947
948    /**
949     * Get the local concavity sign.
950     *
951     * @return the local concavity sign.
952     */
953    public double getLocalConcavitySign() {
954        return localConcavitySign;
955    }
956
957    /**
958     * Set the local concavity sign.
959     *
960     * @param localConcavitySign the local concavity sign.
961     */
962    public void setLocalConcavitySign(double localConcavitySign) {
963        this.localConcavitySign = localConcavitySign;
964    }
965
966    /**
967     * Get the local convexity sign.
968     *
969     * @return the local convexity sign.
970     */
971    public double getLocalConvexitySign() {
972        return localConvexitySign;
973    }
974
975    /**
976     * Set the local convexity sign.
977     *
978     * @param localConvexitySign the local convexity sign.
979     */
980    public void setLocalConvexitySign(double localConvexitySign) {
981        this.localConvexitySign = localConvexitySign;
982    }
983
984    /**
985     * Get the local curvature sign.
986     *
987     * @return the local curvature sign.
988     */
989    public double getLocalCurvatureSign() {
990        return localCurvatureSign;
991    }
992
993    /**
994     * Set the local curvature sign.
995     *
996     * @param localCurvatureSign the local curvature sign.
997     */
998    public void setLocalCurvatureSign(double localCurvatureSign) {
999        this.localCurvatureSign = localCurvatureSign;
1000    }
1001
1002    /**
1003     * Get the local inflection point sign.
1004     *
1005     * @return the local inflection point sign.
1006     */
1007    public double getLocalInflectionPointSign() {
1008        return localInflectionPointSign;
1009    }
1010
1011    /**
1012     * Set the local inflection point sign.
1013     *
1014     * @param localInflectionPointSign the local inflection point sign.
1015     */
1016    public void setLocalInflectionPointSign(double localInflectionPointSign) {
1017        this.localInflectionPointSign = localInflectionPointSign;
1018    }
1019
1020    /**
1021     * Get the local concavity sign.
1022     *
1023     * @return the local concavity sign.
1024     */
1025    public double getLocalConcavitySign() {
1026        return localConcavitySign;
1027    }
1028
1029    /**
1030     * Set the local concavity sign.
1031     *
1032     * @param localConcavitySign the local concavity sign.
1033     */
1034    public void setLocalConcavitySign(double localConcavitySign) {
1035        this.localConcavitySign = localConcavitySign;
1036    }
1037
1038    /**
1039     * Get the local convexity sign.
1040     *
1041     * @return the local convexity sign.
1042     */
1043    public double getLocalConvexitySign() {
1044        return localConvexitySign;
1045    }
1046
1047    /**
1048     * Set the local convexity sign.
1049     *
1050     * @param localConvexitySign the local convexity sign.
1051     */
1052    public void setLocalConvexitySign(double localConvexitySign) {
1053        this.localConvexitySign = localConvexitySign;
1054    }
1055
1056    /**
1057     * Get the local curvature sign.
1058     *
1059     * @return the local curvature sign.
1060     */
1061    public double getLocalCurvatureSign() {
1062        return localCurvatureSign;
1063    }
1064
1065    /**
1066     * Set the local curvature sign.
1067     *
1068     * @param localCurvatureSign the local curvature sign.
1069     */
1070    public void setLocalCurvatureSign(double localCurvatureSign) {
1071        this.localCurvatureSign = localCurvatureSign;
1072    }
1073
1074    /**
1075     * Get the local inflection point sign.
1076     *
1077     * @return the local inflection point sign.
1078     */
1079    public double getLocalInflectionPointSign() {
1080        return localInflectionPointSign;
1081    }
1082
1083    /**
1084     * Set the local inflection point sign.
1085     *
1086     * @param localInflectionPointSign the local inflection point sign.
1087     */
1088    public void setLocalInflectionPointSign(double localInflectionPointSign) {
1089        this.localInflectionPointSign = localInflectionPointSign;
1090    }
1091
1092    /**
1093     * Get the local concavity sign.
1094     *
1095     * @return the local concavity sign.
1096     */
1097    public double getLocalConcavitySign() {
1098        return localConcavitySign;
1099    }
1100
1101    /**
1102     * Set the local concavity sign.
1103     *
1104     * @param localConcavitySign the local concavity sign.
1105     */
1106    public void setLocalConcavitySign(double localConcavitySign) {
1107        this.localConcavitySign = localConcavitySign;
1108    }
1109
1110    /**
1111     * Get the local convexity sign.
1112     *
1113     * @return the local convexity sign.
1114     */
1115    public double getLocalConvexitySign() {
1116        return localConvexitySign;
1117    }
1118
1119    /**
1120     * Set the local convexity sign.
1121     *
1122     * @param localConvexitySign the local convexity sign.
1123     */
1124    public void setLocalConvexitySign(double localConvexitySign) {
1125        this.localConvexitySign = localConvexitySign;
1126    }
1127
1128    /**
1129     * Get the local curvature sign.
1130     *
1131     * @return the local curvature sign.
1132     */
1133    public double getLocalCurvatureSign() {
1134        return localCurvatureSign;
1135    }
1136
1137    /**
1138     * Set the local curvature sign.
1139     *
1140     * @param localCurvatureSign the local curvature sign.
1141     */
1142    public void setLocalCurvatureSign(double localCurvatureSign) {
1143        this.localCurvatureSign = localCurvatureSign;
1144    }
1145
1146    /**
1147     * Get the local inflection point sign.
1148     *
1149     * @return the local inflection point sign.
1150     */
1151    public double getLocalInflectionPointSign() {
1152        return localInflectionPointSign;
1153    }
1154
1155    /**
1156     * Set the local inflection point sign.
1157     *
1158     * @param localInflectionPointSign the local inflection point sign.
1159     */
1160    public void setLocalInflectionPointSign(double localInflectionPointSign) {
1161        this.localInflectionPointSign = localInflectionPointSign;
1162    }
1163
1164    /**
1165     * Get the local concavity sign.
1166     *
1167     * @return the local concavity sign.
1168     */
1169    public double getLocalConcavitySign() {
1170        return localConcavitySign;
1171    }
1172
1173    /**
1174     * Set the local concavity sign.
1175     *
1176     * @param localConcavitySign the local concavity sign.
1177     */
1178    public void setLocalConcavitySign(double localConcavitySign) {
1179        this.localConcavitySign = localConcavitySign;
1180    }
1181
1182    /**
1183     * Get the local convexity sign.
1184     *
1185     * @return the local convexity sign.
1186     */
1187    public double getLocalConvexitySign() {
1188        return localConvexitySign;
1189    }
1190
1191    /**
1192     * Set the local convexity sign.
1193     *
1194     * @param localConvexitySign the local convexity sign.
1195     */
1196    public void setLocalConvexitySign(double localConvexitySign) {
1197        this.localConvexitySign = localConvexitySign;
1198    }
1199
1200    /**
1201     * Get the local curvature sign.
1202     *
1203     * @return the local curvature sign.
1204     */
1205    public double getLocalCurvatureSign() {
1206        return localCurvatureSign;
1207    }
1208
1209    /**
1210     * Set the local curvature sign.
1211     *
1212     * @param localCurvatureSign the local curvature sign.
1213     */
1214    public void setLocalCurvatureSign(double localCurvatureSign) {
1215        this.localCurvatureSign = localCurvatureSign;
1216    }
1217
1218    /**
1219     * Get the local inflection point sign.
1220     *
1221     * @return the local inflection point sign.
1222     */
1223    public double getLocalInflectionPointSign() {
1224        return localInflectionPointSign;
1225    }
1226
1227    /**
1228     * Set the local inflection point sign.
1229     *
1230     * @param localInflectionPointSign the local inflection point sign.
1231     */
1232    public void setLocalInflectionPointSign(double localInflectionPointSign) {
1233        this.localInflectionPointSign = localInflectionPointSign;
1234    }
1235
1236    /**
1237     * Get the local concavity sign.
1238     *
1239     * @return the local concavity sign.
1240     */
1241    public double getLocalConcavitySign() {
1242        return localConcavitySign;
1243    }
1244
1245    /**
1246     * Set the local concavity sign.
1247     *
1248     * @param localConcavitySign the local concavity sign.
1249     */
1250    public void setLocalConcavitySign(double localConcavitySign) {
1251        this.localConcavitySign = localConcavitySign;
1252    }
1253
1254    /**
1255     * Get the local convexity sign.
1256     *
1257     * @return the local convexity sign.
1258     */
1259    public double getLocalConvexitySign() {
1260        return localConvexitySign;
1261    }
1262
1263    /**
1264     * Set the local convexity sign.
1265     *
1266     * @param localConvexitySign the local convexity sign.
1267     */
1268    public void setLocalConvexitySign(double localConvexitySign) {
1269        this.localConvexitySign = localConvexitySign;
1270    }
1271
1272    /**
1273     * Get the local curvature sign.
1274     *
1275     * @return the local curvature sign.
1276     */
1277    public double getLocalCur
```

```

190
191
192
193
194
195
196
197
198
199
200     double fu = computeObjectiveValue(f, u);
201     if (goalType == GoalType.MAXIMIZE) {
202         fu = -fu;
203     }
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237     } else { // termination
238         setResult(x, (goalType == GoalType.MAXIMIZE) ? -fx : fx, count);
239         return x;
240     }
241     ++count;
242 }
243 throw new MaxIterationsExceededException(maximalIterationCount);
244 }
245 }
```

Fixed:

```

43     public BrentOptimizer() {
44         setMaxEvaluations(1000);
45         setMaximalIterationCount(100);
46         setAbsoluteAccuracy(1e-11);
47         setRelativeAccuracy(1e-9);
48     }
49
50     /**
51      * Perform the optimization.
52      *
53      * @return the optimum.
54      */
55     protected double doOptimize()
56         throws MaxIterationsExceededException, FunctionEvaluationException {
57
58         return localMin(goalType() == GoalType.MINIMIZE,
59                         getMin(), getStartValue(), getMax(),
60                         getRelativeAccuracy(), getAbsoluteAccuracy());
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90     double fu = computeObjectiveValue(u);
91     if (!isMinim) {
92         fu = -fu;
93     }
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
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
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
22
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245 }
```

```

86     private double localMin(boolean isMinim,
87
88             double lo, double mid, double hi,
89             double eps, double t)
90         throws MaxIterationsExceededException, FunctionEvaluationException {
91     if (eps <= 0) {
92         throw new NotStrictlyPositiveException(eps);
93     }
94     if (t <= 0) {
95         throw new NotStrictlyPositiveException(t);
96     }
97     double a, b;
98     if (lo < hi) {
99         a = lo;
100        b = hi;
101    } else {
102        a = hi;
103        b = lo;
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) {
118
119        double l = (a + b) / 2.0;

```

```

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

## Chart 24

```

129
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     }
137
138     /**
139      */
140     public Paint getPaint(double value) {
141         double v = Math.max(value, this.lowerBound);
142         v = Math.min(v, this.upperBound);
143         int g = (int) ((v - this.lowerBound) / (this.upperBound
144             - this.lowerBound) * 255.0);
145         return new Color(g, g, g);
146     }
147
148     /**

```

## 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

```
42     private static final long serialVersionUID = -7105341425736035847L;
43
44     private MockitoCore mockitoCore = new MockitoCore();
45     private ReturnsEmptyValues delegate = new ReturnsEmptyValues();
46
47     public Object answer(InvocationOnMock invocation) throws Throwable {
48         GenericMetadataSupport returnTypeGenericMetadata =
49             actualParameterizedType(invocation.getMock()).resolveGenericReturnType(invocation.get
50             tMethod());
51
52         Class<?> rawType = returnTypeGenericMetadata.rawType();
53
54         if (!mockitoCore.isTypeMockable(rawType)) {
55             return delegate.returnValueFor(rawType);
56         }
57
58         return getMock(invocation, returnTypeGenericMetadata);
59     }
59
60
61     private synchronized void instantiateMockitoCoreIfNeeded() {
62         if (mockitoCore == null) {
63             mockitoCore = new MockitoCore();
64         }
65     }
66
67     private synchronized void instantiateDelegateIfNeeded() {
68         if (delegate == null) {
69             delegate = new ReturnsEmptyValues();
70         }
71     }
72
73     private static final long serialVersionUID = -7105341425736035847L;
74
75     private transient MockitoCore mockitoCore;
76     private transient ReturnsEmptyValues delegate;
77
78     public Object answer(InvocationOnMock invocation) throws Throwable {
79         GenericMetadataSupport returnTypeGenericMetadata =
80             actualParameterizedType(invocation.getMock()).resolveGenericReturnType(invocation.get
81             tMethod());
82
83         Class<?> rawType = returnTypeGenericMetadata.rawType();
84
85         instantiateMockitoCoreIfNeeded();
86         instantiateDelegateIfNeeded();
87
88         if (!mockitoCore.isTypeMockable(rawType)) {
89             return delegate.returnValueFor(rawType);
90         }
91
92         return getMock(invocation, returnTypeGenericMetadata);
93     }
93
```

## 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
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300

```

```

263
264     /**
265      * Return the hashCode.
266      *
267      * @return the hashCode
268     */
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
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:

	<pre> 1186     if (str == null) { 1187         str = ""; 1188     } </pre>
	<pre> 70 public class StrBuilder implements Cloneable { </pre>
	<pre> 70 public class StrBuilder implements Cloneable { </pre>

## 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:

	<pre> 920             } 921             default : { 922                 entityValue = Integer.parseInt(entityContent.s 923                     ubstring(1, 10)); 924             } </pre>		<pre> 923             break; 924         } 925         default : { 926             entityValue = Integer.parseInt(entityContent.s 927                 ubstring(1, 10)); 928         } 929         if (entityValue &gt; 0xFFFF) { 930             entityValue = -1; 931         } </pre>
	<pre> 925 } catch (NumberFormatException e) { 926 } </pre>		<pre> 932         } catch (NumberFormatException e) { 933             entityValue = -1; 934         } </pre>

## **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": "^\n",
    "Line": 180
},
```

**Methodology:** Removed/Fixed Warning

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

```
175 int sum = 0; 175 double sum = 0;
```

## **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        }  
  
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        }
```

#### Before 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];  
79     }  
80     return sumSq;
```

#### After 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 * expected[i]);  
91             sumSq += dev * dev / (ratio * expected[i]);  
92         } else {  
93             dev = ((double) observed[i] - expected[i]);  
94             sumSq += dev * dev / expected[i];  
95         }  
96     }  
97     return sumSq;
```

#### 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      return mock;  
91  }  
  
108     container.addAnswer(new Answer<Object>() {  
109         public Object answer(InvocationOnMock invocation)  
throws Throwable {  
110             return mock;  
111         }  
112     }, false);  
113     return mock;  
114 }  
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:

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

```
113     char ch3 = str.charAt(3);
114     char ch4 = str.charAt(4);
115
116     if (ch3 == '_') {
117         return new Locale(str.substring(0, 2), "", str.substring(4));
118     }
119
120     char ch4 = str.charAt(4);
```

### 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:

```
142     */
143     protected double getInitialDomain(double p) {
144         double ret;
145         double d = getDenominatorDegreesOfFreedom();
146
147         // use mean
148         ret = d / (d - 2.0);
149
150         return ret;
151     }
152     /**
153
154     */
155
156     protected double getInitialDomain(double p) {
157         double ret = 1.0;
158         double d = getDenominatorDegreesOfFreedom();
159
160         if (d > 2.0) {
161             // use mean
162             ret = d / (d - 2.0);
163         }
164
165         return ret;
166     }
167 }
```

### 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:

```
47     private boolean toStringEquals(Matcher m, Object arg) {  
48         return StringDescription.toString(m).equals(arg.toString());  
49     }  
50 }  
  
47     private boolean toStringEquals(Matcher m, Object arg) {  
48         return  
49             StringDescription.toString(m).equals(arg == null? "null" : arg.toString());  
50     }
```

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

    /**
     * @param obj
     * @return
     */
    public boolean equals(Object obj) {
        if (!(obj instanceof ShapeList)) {
            return false;
        }
        ShapeList that = (ShapeList) obj;
        int listSize = size();
        for (int i = 0; i < listSize; i++) {
            if (!ShapeUtilities.equal((Shape) get(i), (Shape) tha
t.get(i))) {
                return false;
            }
        }
        return true;
    }
}

```

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;
}

    public Object clone() throws CloneNotSupportedException {
        TimeSeries clone = (TimeSeries) super.clone();
        clone.data = (List) ObjectUtilities.deepClone(this.dat
a);
        return clone;
    }
}

```

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
},

```

```

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

```

Buggy

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     private boolean serializable;
21
22     public MockSettings serializable() {
23         return this.extraInterfaces(java.io.Serializable.class);
24     }
25 }
```

Buggy

```
20     private boolean serializable;
21
22     public MockSettings serializable() {
23         this.serializable = true;
24         return this;
25     }
26 }
```

Fixed

## Bug b

```
73     public boolean isSerializable() {
74         return
75             extraInterfaces != null && java.util.Arrays.asList(extraInterfaces).contains(java.io.Serializable.class);
76     }
77 }
```

Buggy

```
75     public boolean isSerializable() {
76         return serializable;
77     }
78 }
```

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:

```
98     if (m1.getName() != null && m1.getName().equals(m2.getName())) {
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        }
108    }
109 }
110 return false;
111 }
```

Buggy

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 }	681 } 682 } return false; 683 }
Buggy	Fixed

## Lang-48

No warning reported

Code Snippet:

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

<pre>         }         if (str.startsWith("--")) {             return null;         }         if (str.startsWith("0x")    str.startsWith("-0x")    st </pre>	<pre> 451           } 452           if (str.startsWith("0x")    str.startsWith("-0x")    st </pre>
---	--

Buggy

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

<pre> 1729     char[] thisBuf = buffer; 1730     for (int i = startIndex; i &lt; thisBuf.length; i++) { 1731         if (thisBuf[i] == ch) { </pre>	<pre> 1729     char[] thisBuf = buffer; 1730     for (int i = startIndex; i &lt; size; i++) { 1731         if (thisBuf[i] == ch) { </pre>
---	---

Buggy Code

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

Buggy Code

Fixed Code

## Math-14

### No Warning- Spotbugs Removed Warning

```
266     private RealMatrix squareRoot(RealMatrix m) {  
267         final EigenDecomposition dec = new EigenDecompositio  
n(m);  
268         return dec.getSquareRoot();  
269     }  
270     private RealMatrix squareRoot(RealMatrix m) {  
271         if (m instanceof DiagonalMatrix) {  
272             final int dim = m.getRowDimension();  
273             final RealMatrix sqrtM = new DiagonalMatrix(dim);  
274             for (int i = 0; i < dim; i++) {  
275                 sqrtM.setEntry(i, i, FastMath.sqrt(m.getEntry(i,  
i)));  
276             }  
277             return sqrtM;  
278         } else {  
279             final EigenDecomposition dec = new EigenDecompositio  
n(m);  
280             return dec.getSquareRoot();  
281         }  
282     }  
283 }
```

Buggy Code

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": "HEW",
        "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 ||
759                             Double.isNaN(
760                                 this.minimumRangeValue)) {
761                             this.minimumRangeValue = minv;
762                             this.minimumRangeValueRow = r;
763                             this.minimumRangeValueColumn =
764                             c;
765                         }
766                     }
767                     Number max = item.getMaxOutlier();
768                     if (max != null) {
769                         double maxv = max.doubleValue();
770                         if (!Double.isNaN(maxv)) {
771                             if (maxv > this.maximumRangeValue ||
772                                 Double.isNaN(
773                                     this.maximumRangeValue)) {
774                                     this.maximumRangeValue = maxv;
775                                     this.maximumRangeValueRow = r;
776                                     this.maximumRangeValueColumn = c;
777                                 }
778                             }
779                         }
780                     }
781                 }
782             }
783         }
784     }
785 }
```

**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.google.javascript.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.google.javascript.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 constraints on anonymous types.  
557     if (hasReferenceName()) {  
558         return;  
559     }  
560     // Handle the case where the constraint object is a record type.  
561     // Handle the case where the constraint object is a record type.  
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": [
        449,
        450,
        451,
        517,
        518,
        486,
        488,
        488,
        490,
        491,
        491,
        494,
        465,
        466,
        497,
        488,
        488,
        492,
        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 :

```

+     s it to    // to be an specification of class. as a java parse
5         // a wrong value.
5         return null;
7     }
3     if (str.startsWith("0x") || str.startsWith("-0x")) {
4
5         return createInteger(str);
3     }
1     char lastChar = str.charAt(str.length() - 1);
2     String mant;

```

```

+     s it to    // to be an specification of class. as a java parse
455         // a wrong value.
456         return null;
457     }
458     if (str.startsWith("0x") || str.startsWith("-0X") || str.startsWith(
459         "-0X"))
460     {
461         return createInteger(str);
462     }
461     char lastChar = str.charAt(str.length() - 1);
462     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 /**
148  * ...
149 */
150
151 // Serializing
152 /**
153 */
154
155 // Rules
156 /**
157 */
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1286
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1296
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1508
1509
1510
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2197
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2267
2268
2269
2269
2270
2271
2272
2273
2274
```

```

879     if (width > 0) {
880         ensureCapacity(size + width);
881         String str = (obj == null ? getNullText() : obj.toS
882         tring());
883         int strLen = str.length();
884         if (strLen >= width) {
885             str.getChars(0, strLen, buffer, size);
886         } else {
887             int padLen = width - strLen;
888             str.getChars(0, strLen, buffer, size);
889             for (int i = 0; i < padLen; i++) {
890                 buffer[size + strLen + i] = padChar;
891             }
892         }
893     }
894     if (width > 0) {
895         ensureCapacity(size + width);
896         String str = (obj == null ? getNullText() : obj.toS
897         tring());
898         int strLen = str.length();
899         if (strLen >= width) {
900             str.getChars(0, width, buffer, size);
901         } else {
902             int padLen = width - strLen;
903             str.getChars(0, strLen, buffer, size);
904             for (int i = 0; i < padLen; i++) {
905                 buffer[size + strLen + i] = padChar;
906             }
907         }
908     }

```

**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:**

```

181     */
182     public int compareTo(Object other) {
183         return iValue - ((ValuedEnum) other).iValue;
184     }
185
186     /**
187
188
189     * @param other the object to determine the value for
190     * @return the value
191     */
192
193     // ignore - should never happen
194     // ignore - should never happen
195     // ignore - should never happen
196
197
198
199     */
200     public int compareTo(Object other) {
201         if (other == this) {
202             return 0;
203         }
204         if (other.getClass() != this.getClass()) {
205             if (other.getClass().getName().equals(this.getClass()
206                 .getName())) {
207                 return iValue - getValueInOtherClassLoader(othe
208                     r);
209             }
210             throw new ClassCastException(
211                 "Different enum class '" + ClassUtils.getSho
212                 rtClassName(other.getClass()) + "'");
213         }
214         return iValue - ((ValuedEnum) other).iValue;
215     }
216
217     /**
218
219     * @param other the object to determine the value for
220     * @return the value
221     */
222     private int getValueInOtherClassLoader(Object other) {
223         try {
224             Method mth = other.getClass().getMethod("getValue",
225                 null);
226             Integer value = (Integer) mth.invoke(other, null);
227             return value.intValue();
228         } catch (NoSuchMethodException e) {
229             // ignore - should never happen
230         } catch (IllegalAccessException e) {
231             // ignore - should never happen
232         } catch (InvocationTargetException e) {
233             // ignore - should never happen
234         }
235         throw new IllegalStateException("This should not happe
236             n");
237     }
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317

```

## 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
189 * FastMath.abs(x1), atol));
190             f0 = computeObjectiveValue(x0);
191         }
192         break;
193     default:
194         // Should never happen.

```

```

184         break;
185     case REGULA_FALSI:
186         // Nothing.
187
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<=x<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<=x<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:**

<pre>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 &lt; getHeight(); i++) { 294             if (MathUtils.equals(getEntry(i, col), 1.0, epsilon) &amp;&amp; (row == null)) { 295                 row = i; 296             } else if (!MathUtils.equals(getEntry(i, col), 0.0,</pre>	<pre>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 &lt; getHeight(); i++) { 294             if (MathUtils.equals(getEntry(i, col), 1.0, epsilon) &amp;&amp; (row == null)) { 295                 row = i; 296             } else if (!MathUtils.equals(getEntry(i, col), 0.0,</pre>
---	---

```

340     double[] coefficients = new double[getOriginalNumDecisionVariables()];
341     Integer negativeVarBasicRow =
342         getBasicRow(getNegativeDecisionVariableOffset());
342     double mostNegative = negativeVarBasicRow == null ? 0 : getEntry(negativeVarBasicRow, getRhsOffset());
343     Set<Integer> basicRows = new HashSet<Integer>();
344     for (int i = 0; i < coefficients.length; i++) {
345         Integer basicRow =
346             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
348             to 0
340     double[] coefficients = new double[getOriginalNumDecisionVariables()];
341     Integer negativeVarBasicRow =
342         getBasicRowForSolution(getNegativeDecisionVariableOffset());
342     double mostNegative = negativeVarBasicRow == null ? 0 : getEntry(negativeVarBasicRow, getRhsOffset());
343     Set<Integer> basicRows = new HashSet<Integer>();
344     for (int i = 0; i < coefficients.length; i++) {
345         Integer basicRow =
346             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
348             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:

```
252     * Compares this object to another based on size.
253     * @param object the object to compare to
254     * @return -1 if this is less than <tt>object</tt>, +1 if th
255     * is greater
256     *      than <tt>object</tt>, 0 if they are equal.
257     */
258     public int compareTo(Fraction object) {
259         double n0d = doubleValue();
260         double d0n = object.doubleValue();
261         return (n0d < d0n) ? -1 : ((n0d > d0n) ? +1 : 0);
262     }
263     ...
264 
```

```
252     * Compares this object to another based on size.
253     * @param object the object to compare to
254     * @return -1 if this is less than <tt>object</tt>, +1 if th
255     * is greater
256     *      than <tt>object</tt>, 0 if they are equal.
257     */
258     public int compareTo(Fraction object) {
259         long n0d = ((long) numerator) * object.denominator;
260         long d0n = ((long) denominator) * object.numerator;
261         return (n0d < d0n) ? -1 : ((n0d > d0n) ? +1 : 0);
262     }
263     ...
264 
```

**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",
    "Abbreve": "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     public void captureArgumentsFrom(Invocation invocation) {
120         if (invocation.getMethod().isVarArgs()) {
121             int indexOfVararg = invocation.getRawArguments().len
122             gth - 1;
123             throw new UnsupportedOperationException();
124         } else {
125             for (int position = 0; position < matchers.size(); p
126             osition++) {
127                 for (int position = 0; position < indexOfVararg; pos
128                     ition++) {
129                     Matcher m = matchers.get(position);
130                     if (m instanceof CapturesArguments) {
131                         ((CapturesArguments) m).captureFrom(invocati
132 on.getArgumentAt(position, Object.class));
133                     }
134                 }
135             } else {
136                 for (int position = 0; position < matchers.size(); p
137                     osition++) {
138                     Matcher m = matchers.get(position);
139                     if (m instanceof CapturesArguments) {
140                         ((CapturesArguments) m).captureFrom(invocati
141 on.getRawArguments()[position - indexOfVararg]);
142                     }
143                 }
144             }
145         }
146     }

```

**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;

```

```

74      //we need to check if verification was started on the c
75      // - see VerifyingWithAnExtraCallToADifferentMockTest
76      VerificationDataImpl data = new VerificationDataImpl
77      invocationContainerImpl.getInvocations(), invocationMatcher);
78      verificationMode.verify(data);
79      return null;
80
81      //we need to check if verification was started on the c
82      // - see VerifyingWithAnExtraCallToADifferentMockTest
83      if (verificationMode instanceof MockAwareVerification
84          de && ((MockAwareVerificationMode) verificationMode).getMock() ==
85          invocation.getMock()) {
86          VerificationDataImpl data = new VerificationDataImpl
87          invocationContainerImpl.getInvocations(), invocationMatcher);
88          verificationMode.verify(data);
89          return null;
90      }
91

```

**Comment:** In this a library was imported and an if block was added but 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:**

```

225      * @return zero always
226      */
227      public int compareTo(DurationField durationField) {
228          return 0;
229      }
230
231      /**
232      * @return zero always
233      */
234      public int compareTo(DurationField durationField) {
235          if (durationField.isSupported()) {
236              return 1;
237          }
238          return 0;
239      }
240

```

**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  */  
  
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):

```
113  @Override  
114  public boolean shouldTraverse(NodeTraversal t, Node n, Node parent) {  
115      switch (n.getType()) {  
116          case Token.FUNCTION:  
117              {  
118                  // Add recursive function name, if needed.  
119                  // NOTE: "enterScope" is called after we need to pick up this name.  
120                  Renamer renamer = nameStack.peek().forChildScope();  
121  
122                  // If needed, add the function recursive name.  
123                  String name = n.getFirstChild().getString();  
124                  if (name != null && !name.isEmpty() && parent != null  
125                      && !name.equals(nameDeclaration(n))) {  
126                      renamer.addDeclaredName(name);  
127                  }  
128              }  
129  
130  
131          // Add the Function parameters  
132  
133          // Add the Function body declarations  
134  
135          nameStack.push(renamer);  
136      }  
137      break;  
138  }  
139  
140  case Token.CATCH:  
141  {  
142      Renamer renamer = nameStack.peek().forChildScope();  
143  
144      String name = n.getFirstChild().getString();  
145      renamer.addDeclaredName(name);  
146  
147      nameStack.push(renamer);  
148  }  
149  break;  
150 }  
151 return true;  
152 }  
153 }  
154  
155  @Override  
156  public boolean shouldTraverse(NodeTraversal t, Node n, Node parent) {  
157      switch (n.getType()) {  
158          case Token.NAME:  
159              String newName = getReplacementName(n.getString());  
160              if (newName != null) {  
161                  Renamer renamer = nameStack.peek();  
162                  if (renamer.stringConstReplaced()) {  
163                      // TODO(johnlens): Do we need to do anything about the javadoc?  
164                      n.removeProp(Node.IS_CONSTANT_NAME);  
165                  }  
166                  n.setString(newName);  
167                  t.getCompiler().reportCodeChange();  
168              }  
169              break;  
170          case Token.FUNCTION:  
171          // Remove the function body scope  
172  
173          // Remove function recursive name (if any).  
174          nameStack.pop();  
175          break;  
176  
177          // Note: The parameters and function body variables live in the  
178          // same scope, we introduce the scope when in the "shouldTraverse"  
179          // visit of LP, but remove it when we exit the function above.  
180  
181          case Token.CATCH:  
182          // Remove catch except name from the stack of names.  
183          nameStack.pop();  
184          break;  
185      }  
186  }  
187  
188  @Override  
189  public void visit(NodeTraversal t, Node n, Node parent) {  
190      switch (n.getType()) {  
191          case Token.FUNCTION:  
192              {  
193                  // Add recursive function name, if needed.  
194                  // NOTE: "enterScope" is called after we need to pick up this name.  
195                  Renamer renamer = nameStack.peek().forChildScope();  
196  
197                  // If needed, add the function recursive name.  
198                  String name = n.getFirstChild().getString();  
199                  if (name != null && !name.isEmpty() && parent != null  
200                      && !name.equals(nameDeclaration(n))) {  
201                      renamer.addDeclaredName(name);  
202                  }  
203  
204                  nameStack.push(renamer);  
205              }  
206              break;  
207  
208          case Token.LP:  
209          {  
210              Renamer renamer = nameStack.peek();  
211  
212              // Add the Function parameters  
213              for (Node c = n.getFirstChild(); c != null; c = c.getNext()) {  
214                  String name = c.getString();  
215                  renamer.addDeclaredName(name);  
216              }  
217  
218              // Add the Function body declarations  
219              Node FunctionBody = n.getNext();  
220              findDeclaredNames(functionBody, null, renamer);  
221  
222              nameStack.push(renamer);  
223          }  
224          break;  
225  
226          case Token.CATCH:  
227          {  
228              Renamer renamer = nameStack.peek().forChildScope();  
229  
230              String name = n.getFirstChild().getString();  
231              renamer.addDeclaredName(name);  
232  
233          }  
234  
235          nameStack.push(renamer);  
236      }  
237      break;  
238  }  
239  
240  case Token.CATCH:  
241  {  
242      Renamer renamer = nameStack.peek();  
243  
244      String name = n.getFirstChild().getString();  
245      renamer.addDeclaredName(name);  
246  
247      nameStack.push(renamer);  
248  }  
249  break;  
250  
251  }  
252  
253  return true;  
254 }
```

## 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):

```
155  @Override  
156  public void visit(NodeTraversal t, Node n, Node parent) {  
157      switch (n.getType()) {  
158          case Token.NAME:  
159              String newName = getReplacementName(n.getString());  
160              if (newName != null) {  
161                  Renamer renamer = nameStack.peek();  
162                  if (renamer.stringConstReplaced()) {  
163                      // TODO(johnlens): Do we need to do anything about the javadoc?  
164                      n.removeProp(Node.IS_CONSTANT_NAME);  
165                  }  
166                  n.setString(newName);  
167                  t.getCompiler().reportCodeChange();  
168              }  
169              break;  
170          case Token.FUNCTION:  
171          // Remove the function body scope  
172  
173          // Remove function recursive name (if any).  
174          nameStack.pop();  
175          break;  
176  
177          // Note: The parameters and function body variables live in the  
178          // same scope, we introduce the scope when in the "shouldTraverse"  
179          // visit of LP, but remove it when we exit the function above.  
180  
181          case Token.CATCH:  
182          // Remove catch except name from the stack of names.  
183          nameStack.pop();  
184          break;  
185      }  
186  }  
187  
188  @Override  
189  public void visit(NodeTraversal t, Node n, Node parent) {  
190      switch (n.getType()) {  
191          case Token.NAME:  
192              String newName = getReplacementName(n.getString());  
193              if (newName != null) {  
194                  Renamer renamer = nameStack.peek();  
195                  if (renamer.stringConstReplaced()) {  
196                      // TODO(johnlens): Do we need to do anything about the javadoc?  
197                      n.removeProp(Node.IS_CONSTANT_NAME);  
198                  }  
199                  n.setString(newName);  
200                  t.getCompiler().reportCodeChange();  
201              }  
202              break;  
203  
204          case Token.FUNCTION:  
205          // Remove the function body scope  
206          nameStack.pop();  
207          // Remove function recursive name (if any).  
208          nameStack.pop();  
209          break;  
210  
211          case Token.LP:  
212          {  
213              Renamer renamer = nameStack.peek();  
214  
215              // Note: The parameters and Function body variables live in the  
216              // same scope, we introduce the scope when in the "shouldTraverse"  
217              // visit of LP, but remove it when we exit the function above.  
218          }  
219          break;  
220  
221          case Token.CATCH:  
222          {  
223              Renamer renamer = nameStack.peek();  
224  
225              String name = n.getFirstChild().getString();  
226              renamer.addDeclaredName(name);  
227  
228          }  
229          break;  
230  
231          }  
232  
233  return true;  
234 }
```

## 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):**

```

533     case 'f' :
534     case 'F' :
535         try {
536             final Float f = NumberUtils.createFloat(numeric);
537             if (!(f.isInfinite()) || (f.floatValue() == 0.0F && !allZeros)) {
538                 //If it's too big for a float or the float value = 0 and the string
539                 //has non-zeros in it, then float does not have the precision we want
540                 return f;
541             }
542         } catch (final NumberFormatException nfe) { // NOPMD
543             // ignore the bad number
544         }
545         //$/FALL-THROUGH$:
546     case 'd' :
547     case 'D' :
548         try {
549             final Double d = NumberUtils.createDouble(numeric);
550             if (!(d.isInfinite()) || (d.floatValue() == 0.0D && !allZeros)) {
551                 return d;
552             }
553         } catch (final NumberFormatException nfe) { // NOPMD
554             // ignore the bad number
555         }
556         try {
557             return createBigDecimal(numeric);
558         } catch (final NumberFormatException e) { // NOPMD
559             // ignore the bad number
560         }
561         //$/FALL-THROUGH$:
562     default :
563         throw new NumberFormatException(str + " is not a valid number.");
564     }
565 }
566 }
567 }
```

```

542     case 'f' :
543     case 'F' :
544         try {
545             final Float f = NumberUtils.createFloat(numeric);
546             if (!(f.isInfinite()) || (f.floatValue() == 0.0F && !allZeros)) {
547                 //If it's too big for a float or the float value = 0 and the string
548                 //has non-zeros in it, then float does not have the precision we want
549                 return f;
550             }
551         } catch (final NumberFormatException nfe) { // NOPMD
552             // ignore the bad number
553         }
554         //$/FALL-THROUGH$:
555     case 'd' :
556     case 'D' :
557         try {
558             final Double d = NumberUtils.createDouble(numeric);
559             if (!(d.isInfinite()) || (d.floatValue() == 0.0D && !allZeros)) {
560                 return d;
561             }
562         } catch (final NumberFormatException nfe) { // NOPMD
563             // ignore the bad number
564         }
565         try {
566             return createBigDecimal(numeric);
567         } catch (final NumberFormatException e) { // NOPMD
568             // ignore the bad number
569         }
570         //$/FALL-THROUGH$:
571     default :
572         throw new NumberFormatException(str + " is not a valid number.");
573     }
574 }
575 }
576 }
```

## 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):

```

533     case 'f' :
534     case 'F' :
535         try {
536             final Float f = NumberUtils.createFloat(numeric);
537             if (!(f.isInfinite()) || (f.floatValue() == 0.0F && !allZeros)) {
538                 //If it's too big for a float or the float value = 0 and the string
539                 //has non-zeros in it, then float does not have the precision we want
540                 return f;
541             }
542         } catch (final NumberFormatException nfe) { // NOPMD
543             // ignore the bad number
544         }
545         //$/FALL-THROUGH$:
546     case 'd' :
547     case 'D' :
548         try {
549             final Double d = NumberUtils.createDouble(numeric);
550             if (!(d.isInfinite()) || (d.floatValue() == 0.0D && !allZeros)) {
551                 return d;
552             }
553         } catch (final NumberFormatException nfe) { // NOPMD
554             // ignore the bad number
555         }
556         try {
557             return createBigDecimal(numeric);
558         } catch (final NumberFormatException e) { // NOPMD
559             // ignore the bad number
560         }
561         //$/FALL-THROUGH$:
562     default :
563         throw new NumberFormatException(str + " is not a valid number.");
564     }
565 }
566 }
567 }
```

```

542     case 'f' :
543     case 'F' :
544         try {
545             final Float f = NumberUtils.createFloat(numeric);
546             if (!(f.isInfinite()) || (f.floatValue() == 0.0F && !allZeros)) {
547                 //If it's too big for a float or the float value = 0 and the string
548                 //has non-zeros in it, then float does not have the precision we want
549                 return f;
550             }
551         } catch (final NumberFormatException nfe) { // NOPMD
552             // ignore the bad number
553         }
554         //$/FALL-THROUGH$:
555     case 'd' :
556     case 'D' :
557         try {
558             final Double d = NumberUtils.createDouble(numeric);
559             if (!(d.isInfinite()) || (d.floatValue() == 0.0D && !allZeros)) {
560                 return d;
561             }
562         } catch (final NumberFormatException nfe) { // NOPMD
563             // ignore the bad number
564         }
565         try {
566             return createBigDecimal(numeric);
567         } catch (final NumberFormatException e) { // NOPMD
568             // ignore the bad number
569         }
570         //$/FALL-THROUGH$:
571     default :
572         throw new NumberFormatException(str + " is not a valid number.");
573     }
574 }
575 }
576 }
```

## 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:

```

458     public static Number createNumber(final String str) throws NumberFormatException {
459         if (str == null) {
460             return null;
461         }
462         if (StringUtils.isBlank(str)) {
463             throw new NumberFormatException("A blank string is not a valid number");
464         }
465         // Need to deal with all possible hex prefixes here
466         final String[] hex_prefixes = {"0x", "0X", "-0x", "-0X", "#", "-#"};
467         int pxlen = 0;
468         for(final String pfx : hex_prefixes) {
469             if (str.startsWith(pfx)) {
470                 pxlen += pfx.length();
471                 break;
472             }
473         }
474     }
475     if (pxlen > 0) { // we have a hex number
476         final int hexDigits = str.length() - pxlen;
477         if (hexDigits > 16) { // too many for Long
478             return createBigInteger(str);
479         }
480         if (hexDigits > 8) { // too many for an int
481             return createLong(str);
482         }
483         return createInteger(str);
484     }
485     final char lastChar = str.charAt(str.length() - 1);
486     String mant;
487     String dec;
488     String exp;
489     final int decPos = str.indexOf('.');
490     final int expPos = str.indexOf('e') + str.indexOf('E') + 1; // assumes both not present
491     // if both e and E are present, this is caught by the checks on expPos (which prevent IODBE)
492     // and the parsing which will detect if e or E appear in a number due to using the wrong offset
493     int numDecimals = 0; // Check required precision (LANG-693)
494     if (decPos < -1) { // there is a decimal point
495
496         if (expPos > -1) { // there is an exponent
497             if (expPos < decPos || expPos > str.length()) { // prevents double exponent causing IODBE
498                 throw new NumberFormatException(str + " is not a valid number.");
499             }
500             dec = str.substring(decPos + 1, expPos);
501             dec = str.substring(decPos + 1);
502         }
503         else {
504             dec = str.substring(decPos + 1);
505         }
506         mant = str.substring(0, decPos);
507         numDecimals = dec.length(); // gets number of digits past the decimal to ensure no loss of precision for floating
508         // point numbers.
509     } else {
510         if (expPos > -1) {
511
512             try {
513
514                 final Float f = createFloat(str);
515                 if (!f.isInfinite() || (f.floatValue() == 0.0F && !allZeros)) {
516                     return f;
517                 }
518             } catch (final NumberFormatException nfe) { // NOPMD
519                 // ignore the bad number
520             }
521             try {
522
523                 final Double d = createDouble(str);
524                 if (!d.isInfinite() || (d.doubleValue() == 0.00 && !allZeros)) {
525                     return d;
526                 }
527             } catch (final NumberFormatException nfe) { // NOPMD
528                 // ignore the bad number
529             }
530             return createBigDecimal(str);
531         }
532     }
533     try {
534
535         if (numDecimals <= 7) { // If number has 7 or fewer digits past the decimal point then make it a float
536             final Float f = createFloat(str);
537             if (!f.isInfinite() || (f.floatValue() == 0.0F && !allZeros)) {
538                 return f;
539             }
540         }
541     } catch (final NumberFormatException nfe) { // NOPMD
542         // ignore the bad number
543     }
544     try {
545
546         if (numDecimals <= 16) { // If number has between 8 and 16 digits past the decimal point then make it a double
547             final Double d = createDouble(str);
548             if (!d.isInfinite() || (d.doubleValue() == 0.00 && !allZeros)) {
549                 return d;
550             }
551         }
552     } catch (final NumberFormatException nfe) { // NOPMD
553         // ignore the bad number
554     }
555     return createBigDecimal(str);
556 }
```

### 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:**

```
383 private static StringBuilder escapeRegex(StringBuilder regex, String value, boolean unquote) {
384     boolean waswhite = false;
385     for(int i = 0; i<value.length(); ++i) {
386         char c = value.charAt(i);
387         if(Character.isWhitespace(c)) {
388             if(!waswhite) {
389                 waswhite = true;
390                 regex.append("\\s+");
391             }
392             continue;
393         }
394         waswhite = false;
395         switch(c) {
396             case "'":
397                 if(unquote) {
398                     if(i+1==value.length()) {
399                         return regex;
400                     }
401                     c = value.charAt(i+1);
402                 }
403                 break;
404             case '?':
405             case '[':
406             case ']':
407             case '(':
408             case ')':
409             case '{':
410             case '}':
411             case '\\':
412             case '^':
413             case '*':
414             case '+':
415             case '^':
416             case '$':
417                 if(unquote) {
418                     if(i+1==value.length()) {
419                         return regex;
420                     }
421                     c = value.charAt(i+1);
422                 }
423                 break;
424             case '%':
425             case '#':
426             case '<':
427             case '>':
428             case '&':
429             case '&lt;':
430             case '&gt;':
431             case '&amp;':
432             case '&lt;#':
433             case '&gt;#':
434             case '&lt;#x':
435             case '&gt;#x':
436             case '&lt;#x202f':
437             case '&gt;#x202f':
438                 if(unquote) {
439                     if(i+1==value.length()) {
440                         return regex;
441                     }
442                     c = value.charAt(i+1);
443                 }
444                 break;
445             default:
446                 regex.append(c);
447         }
448     }
449     return regex;
450 }
451
452 private static String escapeRegex(StringBuilder regex, String value, boolean unquote) {
453     for(int i = 0; i<value.length(); ++i) {
454         char c = value.charAt(i);
455
456         switch(c) {
457             case "'":
458                 if(unquote) {
459                     if(i+1==value.length()) {
460                         return regex;
461                     }
462                     c = value.charAt(i+1);
463                 }
464                 break;
465             case '?':
466             case '[':
467             case ']':
468             case '(':
469             case ')':
470             case '{':
471             case '}':
472             case '\\':
473             case '^':
474             case '*':
475             case '+':
476             case '^':
477             case '$':
478                 if(unquote) {
479                     if(i+1==value.length()) {
480                         return regex;
481                     }
482                     c = value.charAt(i+1);
483                 }
484                 break;
485             case '%':
486             case '#':
487             case '<':
488             case '>':
489             case '&':
490             case '&lt;':
491             case '&gt;':
492             case '&amp;':
493             case '&lt;#':
494             case '&gt;#':
495             case '&lt;#x':
496             case '&gt;#x':
497             case '&lt;#x202f':
498             case '&gt;#x202f':
499                 if(unquote) {
500                     if(i+1==value.length()) {
501                         return regex;
502                     }
503                     c = value.charAt(i+1);
504                 }
505                 break;
506             default:
507                 regex.append(c);
508         }
509     }
510     return regex.toString();
511 }
```

**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             throw new NumberFormatException(str + " is not a valid number.");
464         case 'F':
465         case 'f':
466             try {
467                 float f = NumberUtils.createFloat(numeric);
468                 if (((f.isInfinite()) || (f.floatValue() == 0.0F && (!allZeros))) ||
469                     //If it's too big for a float or the float value = 0 and the string
470                     //has no zeros in it, then float does not have the precision we want
471                     return f;
472             }
473             catch (NumberFormatException nfe) {
474                 // ignore the bad number
475             }
476             //Fall through
477         case 'D':
478         case 'd':
479             try {
480                 Double d = NumberUtils.createDouble(numeric);
481                 if (((d.isInfinite()) || (d.floatValue() == 0.0D && (!allZeros))) ||
482                     return d;
483             }
484             catch (NumberFormatException nfe) {
485                 // ignore the bad number
486             }
487             try {
488                 return createBigDecimal(numeric);
489             } catch (NumberFormatException e) {
490                 // ignore the bad number
491             }
492             //Fall through
493         default:
494             throw new NumberFormatException(str + " is not a valid number.");
495     }
496 }
497 }
```

```

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                 try {
456                     return createLong(numeric);
457                 } catch (NumberFormatException nfe) {
458                     //Too big for a long
459                 }
460                 return createBigInteger(numeric);
461             }
462             throw new NumberFormatException(str + " is not a valid number.");
463         case 'F':
464         case 'f':
465             try {
466                 float f = NumberUtils.createFloat(numeric);
467                 if (((f.isInfinite()) || (f.floatValue() == 0.0F && (!allZeros))) ||
468                     //If it's too big for a float or the float value = 0 and the string
469                     //has no zeros in it, then float does not have the precision we want
470                     return f;
471             }
472             catch (NumberFormatException nfe) {
473                 // ignore the bad number
474             }
475             //Fall through
476         case 'D':
477         case 'd':
478             try {
479                 Double d = NumberUtils.createDouble(numeric);
480                 if (((d.isInfinite()) || (d.floatValue() == 0.0D && (!allZeros))) ||
481                     return d;
482             }
483             catch (NumberFormatException nfe) {
484                 // ignore the bad number
485             }
486             try {
487                 return createBigDecimal(numeric);
488             } catch (NumberFormatException e) {
489                 // ignore the bad number
490             }
491             //Fall through
492         default:
493             throw new NumberFormatException(str + " is not a valid number.");
494     }
495 }
```

## 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 / caught 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:

```

187     double[] x = guess;
188     double fVal = computeObjectiveValue(x);
189     double[] xi = x.clone();
190
191     int iter = 0;
192     while (true) {
193         //iter++;
194
195         double fx = fVal;
196         double fx2 = 0;
197         double delta = 0;
198         int bigInd = 0;
199         double alphaMin = 0;
200
201         for (int i = 0; i < n; i++) {
202             final double[] d = MathArrays.copyOf(direc[i]);
203
204             fx2 += fVal;
205
206             final UnivariatePointValuePair optimum = line.search(x, d);
207             fVal = optimum.getValue();
208             alphaMin = optimum.getPoint();
209             final double[][] result = newPointAndDirection(x, d, alphaMin);
210             x = result[0];
211
212             if ((fx2 - fVal) > delta) {
213                 delta = fx2 - fVal;
214                 bigInd = i;
215             }
216         }
217
218         // Default convergence check.
219         boolean stop = 2 * (fx - fVal) <
220             (relativeThreshold * (FastMath.abs(fX) + FastMath.abs(fVal)) +
221              absoluteThreshold);
222
223         final PointValuePair previous = new PointValuePair(x, fX);
224         final PointValuePair current = new PointValuePair(x, fVal);
225         if (!stop) { // User-defined stopping criteria.
226             if (checker != null) {
227                 stop = checker.converged(iter, previous, current);
228             }
229         }
230
231         if (stop) break;
232
233         double[] x = guess;
234         double fVal = computeObjectiveValue(x);
235         double[] xi = x.clone();
236
237         while (true) {
238             incrementIterationCount();
239
240             double fx = fVal;
241             double fx2 = 0;
242             double delta = 0;
243             int bigInd = 0;
244             double alphaMin = 0;
245
246             for (int i = 0; i < n; i++) {
247                 final double[] d = MathArrays.copyOf(direc[i]);
248
249                 fx2 += fVal;
250
251                 final UnivariatePointValuePair optimum = line.search(x, d);
252                 fVal = optimum.getValue();
253                 alphaMin = optimum.getPoint();
254                 final double[][] result = newPointAndDirection(x, d, alphaMin);
255                 x = result[0];
256
257                 if ((fx2 - fVal) > delta) {
258                     delta = fx2 - fVal;
259                     bigInd = i;
260                 }
261             }
262
263             // Default convergence check.
264             boolean stop = 2 * (fx - fVal) <
265                 (relativeThreshold * (FastMath.abs(fX) + FastMath.abs(fVal)) +
266                  absoluteThreshold);
267
268             final PointValuePair previous = new PointValuePair(x, fX);
269             final PointValuePair current = new PointValuePair(x, fVal);
270             if (!stop) { // User-defined stopping criteria.
271                 if (checker != null) {
272                     stop = checker.converged(getIterations(), previous, current);
273                 }
274             }
275         }
276     }
277 }
```

## 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:

```

213     PointValuePair current = null;
214     int iter = 0;
215     int maxEval = getMaxEvaluations();
216     while (true) {
217         if (maxEval <= 0) {
218             final double objective = computeObjectiveValue(point);
219             PointValuePair previous = current;
220             current = new PointValuePair(point, objective);
221             if (previous != null) {
222                 if (checker.converged(iter, previous, current)) {
223                     // We have found an optimum.
224                     return current;
225                 }
226             }
227         }
228         // Find the optimal step in the search direction.
229         final UnivariateFunction lsf = new LineSearchFunction(point, searchDirection);
230         final double ub = findUpperBound(lsf, 0, initialStep);
231         // XXX last parameters is set to a value close to zero in order to
232         // work around the divergence problem in the "testCircleFitting"
233         // unit test (see MATH-439).
234         final double step = solver.solve(maxEval, lsf, 0, ub, 1e-15);
235         maxEval -= solver.getEvaluations(); // Subtract used up evaluations.
236     }
237 }

213     PointValuePair current = null;
214     int maxEval = getMaxEvaluations();
215     while (true) {
216         if (maxEval <= 0) {
217             incrementIterationCount();
218             final double objective = computeObjectiveValue(point);
219             PointValuePair previous = current;
220             current = new PointValuePair(point, objective);
221             if (previous != null) {
222                 if (checker.converged(getIterations(), previous, current)) {
223                     // We have found an optimum.
224                     return current;
225                 }
226             }
227         }
228         // Find the optimal step in the search direction.
229         final UnivariateFunction lsf = new LineSearchFunction(point, searchDirection);
230         final double ub = findUpperBound(lsf, 0, initialStep);
231         // XXX last parameters is set to a value close to zero in order to
232         // work around the divergence problem in the "testCircleFitting"
233         // unit test (see MATH-439).
234         final double step = solver.solve(maxEval, lsf, 0, ub, 1e-15);
235         maxEval -= solver.getEvaluations(); // Subtract used up evaluations.
236     }
237 }
```

### 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:

```

154     PointValuePair[] previous = null;
155     int iteration = 0;
156     final ConvergenceChecker<PointValuePair> checker = getConvergenceChecker();
157     while (true) {
158         if (iteration > 0) {
159             boolean converged = true;
160             for (int i = 0; i < simplex.getSize(); i++) {
161                 PointValuePair prev = previous[i];
162                 converged = converged &&
163                     checker.converged(iteration, prev, simplex.getPoint(i));
164             }
165             if (converged) {
166                 // We have found an optimum.
167                 return simplex.getPoint(0);
168             }
169         }
170         // We still need to search.
171         previous = simplex.getPoints();
172         simplex.iterate(evalFunc, comparator);
173         iteration++;
174     }
175 }
```

```

154     PointValuePair[] previous = null;
155     int iteration = 0;
156     final ConvergenceChecker<PointValuePair> checker = getConvergenceChecker();
157     while (true) {
158         if (getIterations() > 0) {
159             boolean converged = true;
160             for (int i = 0; i < simplex.getSize(); i++) {
161                 PointValuePair prev = previous[i];
162                 converged = converged &&
163                     checker.converged(iteration, prev, simplex.getPoint(i));
164             }
165             if (converged) {
166                 // We have found an optimum.
167                 return simplex.getPoint(0);
168             }
169         }
170         // We still need to search.
171         previous = simplex.getPoints();
172         simplex.iterate(evalFunc, comparator);
173         incrementIterationCount();
174     }
175 }
```

### 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     PointVectorValuePair previous = null;
107     for (boolean converged = false; !converged;) {
108         //iterate
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(currentPoint);
115         final ResidualsJacobian currentJacobian = computeJacobian(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 < nB; ++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[] weightGrad = weight * grad;
129             for (int j = 0; j < nC; ++j) {
130                 b[j] += weight * 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[] wkg = weightGrad;
137                 for (int l = 0; l < nC; ++l++) {
138                     ak[l] += wkg * grad[l];
139                 }
140             }
141         }
142     }

```

### 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 arx = random(dimension, lambda);
391         final RealMatrix arx0 = zeros(dimension, lambda);
392         final double[] fitness = new double[lambda];
393
394         // generate random offspring
395         for (int k = 0; k < lambda; k++) {
396             RealMatrix arxk = null;
397             for (int i = 0; i < checkFeasibleCount + 1; i++) {
398                 if (diagonalOnly <= 0) {
399                     arxk = xmean.add(BD.multiply(arx.getColumnMatrix(k))
400                         .scalarMultiply(sigmas)); // m + sig * Normal(0,C)
401                 } else {
402                     arxk = xmean.add(times(diagD, arx.getColumnMatrix(k))
403                         .scalarMultiply(sigmas));
404                 }
405                 if (i >= checkFeasibleCount || fitfun.isFeasible(arxk.getColumn(0))) {
406                     break;
407                 }
408             }
409             // regenerate random arguments for row
410             arx.setColumn(k, random(dimension));
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         }
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     }

```

```

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     }

```

### 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         // QR decomposition of the jacobian matrix  
328         qrDecomposition(computeWeightedJacobian(currentPoint));  
329         weightedResidual = weightMatrixSqrt.operate(currentResiduals);  
330         for (int i = 0; i < nr; i++) {  
331             qtF[i] = weightedResidual[i];  
332         }  
333         // compute Qt.res  
334         qfY(qtF);  
335         // Outer loop.  
336         lmpar = 0;  
337         boolean firstIteration = true;  
338         final ConvergenceChecker<PointVectorValuePair> checker = getConvergenceChecker();  
339         while (true) {  
340             ++incrementIterationCount();  
341             final PointVectorValuePair previous = current;  
342             // QR decomposition of the jacobian matrix  
343             qrDecomposition(computeWeightedJacobian(currentPoint));  
344             weightedResidual = weightMatrixSqrt.operate(currentResiduals);  
345             for (int i = 0; i < nr; i++) {  
346                 qtF[i] = weightedResidual[i];  
347             }  
348             // compute Qt.res  
349             qfY(qtF);  
350         }
```

#### 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:

<pre> 171     if (f1 * fx &lt; 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                 r0 *= 0.5; 181                 break; 182             case PEGASUS: 183                 r0 *= 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     } </pre>	<pre> 171     if (f1 * fx &lt; 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                 r0 *= 0.5; 181                 break; 182             case PEGASUS: 183                 r0 *= f1 / (f1 + fx); 184                 break; 185             case REGUL_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     } </pre>
---	--

### 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:**

```
498 public double getLInfNorm() {  
499     double max = 0;  
500     Iterator iter = entries.iterator();  
501     while (iter.hasNext()) {  
502         iter.advance();  
503         max += iter.value();  
504     }  
505     return max;  
506 }
```

**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:**

```
498 public double getLInfNorm() {  
499     double max = 0;  
500     Iterator iter = entries.iterator();  
501     while (iter.hasNext()) {  
502         iter.advance();  
503         max += iter.value();  
504     }  
505     return max;  
506 }
```

**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:

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

### 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:

<pre> 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 } </pre>	<pre> 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 } </pre>
---	--

### 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:**

**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 }
217
218 public Object callRealMethod() throws Throwable {
219     if (this.getMethod().getDeclaringClass().isInterface()) {
220         new Reporter().cannotCallRealMethodOnInterface();
221     }
222     return realMethod.invoke(mock, rawArguments);
223 }
224
225 public String toString(PrintSettings printSettings) {
226     return toString(argumentsToMatchers(), printSettings);
227 }
228
229 void markVerified() {
230     this.verified = true;
231 }
232
233 void markVerifiedInOrder() {
234     markVerified();
235     this.verifiedInOrder = true;
236 }
```

## 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:

<pre> 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     } 736 737 738     public void addDays(final int days) { 739         setMillis(getChronology().days().add(getMillis(), days)); 740     } 741 742     //----- 743     /** 744      * Set the hour of the day to the specified value. 745      * 746      * @param hourOfDay the hour of day 747      * @throws IllegalArgumentException if the value is invalid 748      */ 749     public void setHourOfDay(final int hourOfDay) { 750         setMillis(getChronology().hourOfDay().set(getMillis(), hourOfDay)); 751     } 752 753     /** 754      * Add a number of hours to the date. 755      * 756      * @param hours the hours to add 757      * @throws IllegalArgumentException if the value is invalid 758      */ 759     public void addHours(final int hours) { 760         setMillis(getChronology().hours().add(getMillis(), hours)); 761     } 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 </pre>	<pre> 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     } 746 747 748     public void addDays(final int days) { 749         if (days != 0) { 750             setMillis(getChronology().days().add(getMillis(), days)); 751         } 752     } 753 754     //----- 755     /** 756      * Set the hour of the day to the specified value. 757      * 758      * @param hourOfDay the hour of day 759      * @throws IllegalArgumentException if the value is invalid 760      */ 761     public void setHourOfDay(final int hourOfDay) { 762         setMillis(getChronology().hourOfDay().set(getMillis(), hourOfDay)); 763     } 764 765     /** 766      * Add a number of hours to the date. 767      * 768      * @param hours the hours to add 769      * @throws IllegalArgumentException if the value is invalid 770      */ 771     public void addHours(final int hours) { 772         if (hours != 0) { 773             setMillis(getChronology().hours().add(getMillis(), hours)); 774         } 775     } 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 </pre>
--	---

### Category: Mismatch

#### Comment:

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

### Chart 1:

OPEN FILES

```

diffs_parsed.json  ep_diffs_warnings.json  ep_removed_warning  inf_diffs_warnings.json  inf_removed_warning  sb_diffs_warnings.json  sb_removed_warning
1 [ {
2   "Project": "Chart-1",
3   "Class": "org.jfree.chart.renderer.category.AbstractCategoryItemRenderer",
4   "Lines": [
5     1797
6   ],
7 },
8 {
9   "Project": "Chart-10",
10  "Class": "org.jfree.chart.imagemap.StandardToolTipTagFragmentGenerator",
11  "Lines": [
12    65
13 ],
14 },
15 {
16   "Project": "Chart-11",
17   "Class": "org.jfree.chart.util.ShapeUtilities",
18   "Lines": [
19     275
20 ],
21 },
22 {
23   "Project": "Chart-12",
24   "Class": "org.jfree.chart.plot.MultiplePiePlot",
25   "Lines": [
26     145
27 ],
28 },
29 {
30   "Project": "Chart-13",
31   "Class": "org.jfree.chart.block.BorderArrangement",
32   "Lines": [
33     455
34 ],
35 },
36 {
37   "Project": "Chart-14",
38   "Class": "org.jfree.chart.plot.XYPlot".
39 ]

```

8 lines, 183 characters selected

Spaces: 4

JSON

## Infer – 1, 2 methodology (Full match)

OPEN FILES

```

diffs_parsed.json  ep_diffs_warnings.json  ep_removed_warning  inf_diffs_warnings.json  inf_removed_warning  sb_diffs_warnings.json  sb_removed_warning
1 [ {
2   "Proj": "Chart-1",
3   "Class": "org.jfree.chart.renderer.category.AbstractCategoryItemRenderer",
4   "Bug_Class": "PROVER",
5   "Kind": "ERROR",
6   "Bug_Type": "NULL_DEREFERENCE",
7   "Msg": "object `dataset` last assigned on line 1796 could be null and is de",
8   "Severity": "HIGH",
9   "Visibility": "user",
10  "Lines": [
11    1792,
12    1795,
13    1796,
14    1797,
15    1800,
16    1790,
17    1791
18  ],
19  "Procedure": "LegendItemCollection AbstractCategoryItemRenderer.getLegendItems()"
20 },
21 {
22   "Proj": "Chart-4",
23   "Class": "org.jfree.chart.plot.XYPlot",
24   "Bug_Class": "PROVER",
25   "Kind": "ERROR",
26   "Bug_Type": "NULL_DEREFERENCE",
27   "Msg": "object `r` last assigned on line 4473 could be null and is derefere",
28   "Severity": "HIGH",
29   "Visibility": "user",
30   "Lines": [
31     4480,
32     4451,
33     4452,
34     4425,
35     4427,
36     4428,
37     4493,
38     4429,
39   ]

```

21 lines, 668 characters selected

Spaces: 4

JSON

```

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 {
1132.         return result;
1133.     }
1134. }
1135. >     int seriesCount = dataset.getRowCount();
1136. if (plot.getRowRenderingOrder().equals(sortOrder.ASCENDING)) {
1137.     for (int i = 0; i < seriesCount; i++) {
1138.         ...
1139.     }
1140. }

Summary of the reports
NULL_DEREference: 2

```

### Buggy 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 {
1132.         return result;
1133.     }
1134. }

Summary of the reports
NULL_DEREference: 1

```

### fixed warnings:

### Spotbugs – 2<sup>nd</sup> methodology (Full match)

```

OPEN FILES
diffs_parsed.json × ep_diffs_warnings.json × ep_removed_warning × inf_diffs_warnings.json × inf_removed_warning × sb_diffs_warnings.json × sb_removed_warning ×

1 [
2   {
3     "Proj": "Chart-1",
4     "Class": "org.jfree.chart.renderer.category.AbstractCategoryItemRenderer",
5     "Cat": "CORRECTNESS",
6     "Abbrev": "NP",
7     "Type": "NP_ALWAYS_NULL",
8     "Priority": "1",
9     "Rank": "5",
10    "Msg": "Null pointer dereference of ? in org.jfree.chart.renderer.category.AbstractCategoryItemRenderer.getLegendItems()",
11    "Method": "getLegendItems",
12    "Field": "",
13    "Lines": [
14      [
15        1800,
16        1800,
17        "SOURCE_LINE_DEREF"
18      ]
19    ],
20  },
21  {
22    "Proj": "Chart-17",
23    "Class": "org.jfree.data.time.TimeSeries",
24    "Cat": "BAD_PRACTICE",
25    "Abbrev": "CN",
26    "Type": "CN_IDIOM_NO_SUPER_CALL",
27    "Priority": "2",
28    "Rank": "16",
29    "Msg": "org.jfree.data.time.TimeSeries.clone() does not call super.clone()",
30    "Method": "clone",
31    "Field": "",
32    "Lines": [
33      [
34        857,
35        858,
36        null
37      ]
38    ],
39  }
]

20 lines, 618 characters selected
Spaces: 4
JSON

```

### buggy warnings:

file:///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 class  
 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 <p> A null pointer is dereferenced here.&nbsp; This will lead to a  
<code>NullPointerException</code> when the code is executed.</p> Possible null pointer  
 dereference <p> There is a branch of statement that, <em>if executed,</em> 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. </p> Null pointer dereference

## fixed warnings:

file:///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 <p> There is a branch of statement that, <em>if executed,</em>  
 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. </p> Null pointer dereference

## Code diff:

1788        * @see #getLegendItem(int, int) 1789        */ 1790        public LegendItemCollection getLegendItems() { 1791           LegendItemCollection result = new LegendItemCo llection(); 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(SortOrd er.ASCENDING)) { 1802                for (int i = 0; i < seriesCount; i++) { 1803                   if (isSeriesVisibleInLegend(i)) { 1804                       LegendItem item = getLegendItem(in dex, i); 1805                       if (item != null) { 1806                           result.add(item); 1807                   } 1808                } 1809           } 1810        } 1811        else { 1812           for (int i = seriesCount - 1; i >= 0; i--)	1788        * @see #getLegendItem(int, int) 1789        */ 1790        public LegendItemCollection getLegendItems() { 1791           LegendItemCollection result = new LegendItemCo llection(); 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(SortOrd er.ASCENDING)) { 1802                for (int i = 0; i < seriesCount; i++) { 1803                   if (isSeriesVisibleInLegend(i)) { 1804                       LegendItem item = getLegendItem(in dex, i); 1805                       if (item != null) { 1806                           result.add(item); 1807                   } 1808                } 1809           } 1810        } 1811        else { 1812           for (int i = seriesCount - 1; i >= 0; i--)
---	---

## Closure-100:

OPEN FILES

```

diffs_parsed.json  ep_diffs_warnings.json  ep_removed_warning  inf_diffs_warnings.json  inf_removed_warning  sb_diffs_warnings.json  sb_removed_warning
 401     "Project": "Closure-10",
 402     "Class": "com.google.javascript.jscomp.NodeUtil",
 403     "Lines": "[ 1417 ]",
 404   },
 405   {
 406     "Project": "Closure-100",
 407     "Class": "com.google.javascript.jscomp.CheckGlobalThis",
 408     "Lines": "[ 98, 99, 100, 101, 102, 103, 104, 105, 106, 146, 152, 153, 154 ]",
 409   },
 410   {
 411     "Project": "Closure-101",
 412     "Class": "com.google.javascript.jscomp.CommandLineRunner",
 413     "Lines": "[ 433, 434, 435 ]",
 414   },
 415   {
 416     "Project": "Closure-102",
 417     "Class": "com.google.javascript.jscomp.Normalize",
 418     "Lines": "[ 88 ]"
 419   }
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536
 537
 538
 539
 540
 541
 542
 543
 544
 545
 546
 547
 548
 549
 550
 551
 552
 553
 554
 555
 556
 557
 558
 559
 560
 561
 562
 563
 564
 565
 566
 567
 568
 569
 570
 571
 572
 573
 574
 575
 576
 577
 578
 579
 580
 581
 582
 583
 584
 585
 586
 587
 588
 589
 590
 591
 592
 593
 594
 595
 596
 597
 598
 599
 600
 601
 602
 603
 604
 605
 606
 607
 608
 609
 610
 611
 612
 613
 614
 615
 616
 617
 618
 619
 620
 621
 622
 623
 624
 625
 626
 627
 628
 629
 630
 631
 632
 633
 634
 635
 636
 637
 638
 639
 640
 641
 642
 643
 644
 645
 646
 647
 648
 649
 650
 651
 652
 653
 654
 655
 656
 657
 658
 659
 660
 661
 662
 663
 664
 665
 666
 667
 668
 669
 670
 671
 672
 673
 674
 675
 676
 677
 678
 679
 680
 681
 682
 683
 684
 685
 686
 687
 688
 689
 690
 691
 692
 693
 694
 695
 696
 697
 698
 699
 700
 701
 702
 703
 704
 705
 706
 707
 708
 709
 710
 711
 712
 713
 714
 715
 716
 717
 718
 719
 720
 721
 722
 723
 724
 725
 726
 727
 728
 729
 730
 731
 732
 733
 734
 735
 736
 737
 738
 739
 740
 741
 742
 743
 744
 745
 746
 747
 748
 749
 750
 751
 752
 753
 754
 755
 756
 757
 758
 759
 760
 761
 762
 763
 764
 765
 766
 767
 768
 769
 770
 771
 772
 773
 774
 775
 776
 777
 778
 779
 780
 781
 782
 783
 784
 785
 786
 787
 788
 789
 790
 791
 792
 793
 794
 795
 796
 797
 798
 799
 800
 801
 802
 803
 804
 805
 806
 807
 808
 809
 8010
 8011
 8012
 8013
 8014
 8015
 8016
 8017
 8018
 8019
 8020
 8021
 8022
 8023
 8024
 8025
 8026
 8027
 8028
 8029
 8030
 8031
 8032
 8033
 8034
 8035
 8036
 8037
 8038
 8039
 8040
 8041
 8042
 8043
 8044
 8045
 8046
 8047
 8048
 8049
 8050
 8051
 8052
 8053
 8054
 8055
 8056
 8057
 8058
 8059
 8060
 8061
 8062
 8063
 8064
 8065
 8066
 8067
 8068
 8069
 8070
 8071
 8072
 8073
 8074
 8075
 8076
 8077
 8078
 8079
 8080
 8081
 8082
 8083
 8084
 8085
 8086
 8087
 8088
 8089
 8090
 8091
 8092
 8093
 8094
 8095
 8096
 8097
 8098
 8099
 80100
 80101
 80102
 80103
 80104
 80105
 80106
 80107
 80108
 80109
 80110
 80111
 80112
 80113
 80114
 80115
 80116
 80117
 80118
 80119
 80120
 80121
 80122
 80123
 80124
 80125
 80126
 80127
 80128
 80129
 80130
 80131
 80132
 80133
 80134
 80135
 80136
 80137
 80138
 80139
 80140
 80141
 80142
 80143
 80144
 80145
 80146
 80147
 80148
 80149
 80150
 80151
 80152
 80153
 80154
 80155
 80156
 80157
 80158
 80159
 80160
 80161
 80162
 80163
 80164
 80165
 80166
 80167
 80168
 80169
 80170
 80171
 80172
 80173
 80174
 80175
 80176
 80177
 80178
 80179
 80180
 80181
 80182
 80183
 80184
 80185
 80186
 80187
 80188
 80189
 80190
 80191
 80192
 80193
 80194
 80195
 80196
 80197
 80198
 80199
 80200
 80201
 80202
 80203
 80204
 80205
 80206
 80207
 80208
 80209
 80210
 80211
 80212
 80213
 80214
 80215
 80216
 80217
 80218
 80219
 80220
 80221
 80222
 80223
 80224
 80225
 80226
 80227
 80228
 80229
 80230
 80231
 80232
 80233
 80234
 80235
 80236
 80237
 80238
 80239
 80240
 80241
 80242
 80243
 80244
 80245
 80246
 80247
 80248
 80249
 80250
 80251
 80252
 80253
 80254
 80255
 80256
 80257
 80258
 80259
 80260
 80261
 80262
 80263
 80264
 80265
 80266
 80267
 80268
 80269
 80270
 80271
 80272
 80273
 80274
 80275
 80276
 80277
 80278
 80279
 80280
 80281
 80282
 80283
 80284
 80285
 80286
 80287
 80288
 80289
 80290
 80291
 80292
 80293
 80294
 80295
 80296
 80297
 80298
 80299
 80300
 80301
 80302
 80303
 80304
 80305
 80306
 80307
 80308
 80309
 80310
 80311
 80312
 80313
 80314
 80315
 80316
 80317
 80318
 80319
 80320
 80321
 80322
 80323
 80324
 80325
 80326
 80327
 80328
 80329
 80330
 80331
 80332
 80333
 80334
 80335
 80336
 80337
 80338
 80339
 80340
 80341
 80342
 80343
 80344
 80345
 80346
 80347
 80348
 80349
 80350
 80351
 80352
 80353
 80354
 80355
 80356
 80357
 80358
 80359
 80360
 80361
 80362
 80363
 80364
 80365
 80366
 80367
 80368
 80369
 80370
 80371
 80372
 80373
 80374
 80375
 80376
 80377
 80378
 80379
 80380
 80381
 80382
 80383
 80384
 80385
 80386
 80387
 80388
 80389
 80390
 80391
 80392
 80393
 80394
 80395
 80396
 80397
 80398
 80399
 80400
 80401
 80402
 80403
 80404
 80405
 80406
 80407
 80408
 80409
 80410
 80411
 80412
 80413
 80414
 80415
 80416
 80417
 80418
 80419
 80420
 80421
 80422
 80423
 80424
 80425
 80426
 80427
 80428
 80429
 80430
 80431
 80432
 80433
 80434
 80435
 80436
 80437
 80438
 80439
 80440
 80441
 80442
 80443
 80444
 80445
 80446
 80447
 80448
 80449
 80450
 80451
 80452
 80453
 80454
 80455
 80456
 80457
 80458
 80459
 80460
 80461
 80462
 80463
 80464
 80465
 80466
 80467
 80468
 80469
 80470
 80471
 80472
 80473
 80474
 80475
 80476
 80477
 80478
 80479
 80480
 80481
 80482
 80483
 80484
 80485
 80486
 80487
 80488
 80489
 80490
 80491
 80492
 80493
 80494
 80495
 80496
 80497
 80498
 80499
 80500
 80501
 80502
 80503
 80504
 80505
 80506
 80507
 80508
 80509
 80510
 80511
 80512
 80513
 80514
 80515
 80516
 80517
 80518
 80519
 80520
 80521
 80522
 80523
 80524
 80525
 80526
 80527
 80528
 80529
 80530
 80531
 80532
 80533
 80534
 80535
 80536
 80537
 80538
 80539
 80540
 80541
 80542
 80543
 80544
 80545
 80546
 80547
 80548
 80549
 80550
 80551
 80552
 80553
 80554
 80555
 80556
 80557
 80558
 80559
 80560
 80561
 80562
 80563
 80564
 80565
 80566
 80567
 80568
 80569
 80570
 80571
 80572
 80573
 80574
 80575
 80576
 80577
 80578
 80579
 80580
 80581
 80582
 80583
 80584
 80585
 80586
 80587
 80588
 80589
 80590
 80591
 80592
 80593
 80594
 80595
 80596
 80597
 80598
 80599
 80600
 80601
 80602
 80603
 80604
 80605
 80606
 80607
 80608
 80609
 80610
 80611
 80612
 80613
 80614
 80615
 80616
 80617
 80618
 80619
 80620
 80621
 80622
 80623
 80624
 80625
 80626
 80627
 80628
 80629
 80630
 80631
 80632
 80633
 80634
 80635
 80636
 80637
 80638
 80639
 80640
 80641
 80642
 80643
 80644
 80645
 80646
 80647
 80648
 80649
 80650
 80651
 80652
 80653
 80654
 80655
 80656
 80657
 80658
 80659
 80660
 80661
 80662
 80663
 80664
 80665
 80666
 80667
 80668
 80669
 80670
 80671
 80672
 80673
 80674
 80675
 80676
 80677
 80678
 80679
 80680
 80681
 80682
 80683
 80684
 80685
 80686
 80687
 80688
 80689
 80690
 80691
 80692
 80693
 80694
 80695
 80696
 80697
 80698
 80699
 80700
 80701
 80702
 80703
 80704
 80705
 80706
 80707
 80708
 80709
 80710
 80711
 80712
 80713
 80714
 80715
 80716
 80717
 80718
 80719
 80720
 80721
 80722
 80723
 80724
 80725
 80726
 80727
 80728
 80729
 80730
 80731
 80732
 80733
 80734
 80735
 80736
 80737
 80738
 80739
 80740
 80741
 80742
 80743
 80744
 80745
 80746
 80747
 80748
 80749
 80750
 80751
 80752
 80753
 80754
 80755
 80756
 80757
 80758
 80759
 80760
 80761
 80762
 80763
 80764
 80765
 80766
 80767
 80768
 80769
 80770
 80771
 80772
 80773
 80774
 80775
 80776
 80777
 80778
 80779
 80780
 80781
 80782
 80783
 80784
 80785
 80786
 80787
 80788
 80789
 80790
 80791
 80792
 80793
 80794
 80795
 80796
 80797
 80798
 80799
 80800
 80801
 80802
 80803
 80804
 80805
 80806
 80807
 80808
 80809
 80810
 80811
 80812
 80813
 80814
 80815
 80816
 80817
 80818
 80819
 80820
 80821
 80822
 80823
 80824
 80825
 80826
 80827
 80828
 80829
 80830
 80831
 80832
 80833
 80834
 80835
 80836
 80837
 80838
 80839
 80840
 80841
 80842
 80843
 80844
 80845
 80846
 80847
 80848
 80849
 80850
 80851
 80852
 80853
 80854
 80855
 80856
 80857
 80858
 80859
 80860
 80861
 80862
 80863
 80864
 80865
 80866
 80867
 80868
 80869
 80870
 80871
 80872
 80873
 80874
 80875
 80876
 80877
 80878
 80879
 80880
 80881
 80882
 80883
 80884
 80885
 80886
 80887
 80888
 80889
 80890
 80891
 80892
 80893
 80894
 80895
 80896
 80897
 80898
 80899
 80900
 80901
 80902
 80903
 80904
 80905
 80906
 80907
 80908
 80909
 80910
 80911
 80912
 80913
 80914
 80915
 80916
 80917
 80918
 80919
 80920
 80921
 80922
 80923
 80924
 80925
 80926
 80927
 80928
 80929
 80930
 80931
 80932
 80933
 80934
 80935
 80936
 80937
 80938
 80939
 80940
 80941
 80942
 80943
 80944
 80945
 80946
 80947
 80948
 80949
 80950
 80951
 80952
 80953
 80954
 80955
 80956
 80957
 80958
 80959
 80960
 80961
 80962
 80963
 80964
 80965
 80966
 80967
 80968
 80969
 80970
 80971
 80972
 80973
 80974
 80975
 80976
 80977
 80978
 80979
 80980
 80981
 80982
 80983
 80984
 80985
 80986
 80987
 80988
 80989
 80990
 80991
 80992
 80993
 80994
 80995
 80996
 80997
 80998
 80999
 80100
 80101
 80102
 80103
 80104
 80105
 80106
 80107
 80108
 80109
 80110
 80111
 80112
 80113
 80114
 80115
 80116
 80117
 80118
 80119
 80120
 80121
 80122
 80123
 80124
 80125
 80126
 80127
 80128
 80129
 80130
 80131
 80132
 80133
 80134
 80135
 80136
 80137
 80138
 80139
 80140
 80141
 80142
 80143
 80144
 80145
 80146
 80147
 80148
 80149
 80150
 80151
 80152
 80153
 80154
 80155
 80156
 80157
 80158
 80159
 80160
 80161
 80162
 80163
 80164
 80165
 80166
 80167
 80168
 80169
 80170
 80171
 80172
 80173
 80174
 80175
 80176
 80177
 80178
 80179
 80180
 80181
 80182
 80183
 80184
 80185
 80186
 80187
 80188
 80189
 80190
 80191
 80192
 80193
 80194
 80195
 80196
 80197
 80198
 80199
 80200
 80201
 80202
 80203
 80204
 80205
 80206
 80207
 80208
 80209
 80210
 80211
 80212
 80213
 80214
 80215
 80216
 80217
 80218
 80219
 80220
 80221
 80222
 80223
 80224
 80225
 80226
 80227
 80228
 80229
 80230
 80231
 80232
 80233
 80234
 80235
 80236
 80237
 80238
 80239
 80240
 80241
 80242
 80243
 80244
 80245
 80246
 80247
 80248
 80249
 80250
 80251
 80252
 80253
 80254
 80255
 80256
 80257
 80258
 80259
 80260
 80261
 80262
 80263
 80264
 80265
 80266
 80267
 80268
 80269
 80270
 80271
 80272
 80273
 80274
 80275
 80276
 80277
 80278
 80279
 80280
 80281
 80282
 80283
 80284
 80285
 80286
 80287
 80288
 80289
 80290
 80291
 80292
 80293
 80294
 80295
 80296
 80297
 80298
 80299
 802100
 802101
 802102
 802103
 802104
 802105
 802106
 802107
 802108
 802109
 802110
 802111
 802112
 802113
 802114
 802115
 802116
 802117
 802118
 802119
 802120
 802121
 802122
 802123
 802124
 802125
 802126
 802127
 802128
 802129
 802130
 802131
 802132
 802133
 802134
 802135
 802136
 802137
 802138
 802139
 802140
 802141
 802142
 802143
 802144
 802145
 802146
 802147
 802148
 802149
 802150
 802151
 802152
 802153
 802154
 802155
 802156
 802157
 802158
 802159
 802160
 802161
 802162
 802163
 802164
 802165
 802166
 802167
 802168
 802169
 802170
 802171
 802172
 802173
 802174
 802175
 802176
 80
```

```
/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/antlr_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-  
100/lib/antlr4_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/librunkh_rhino_parser.jar  
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-100/lib/probotool_deploy.jar Dead store to local variable Dead store to $L5 in  
com.google.javacscript.jscmp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node, Node) At CheckGlobalThis.java:[lines 61-174] In class com.google.javacscript.jscmp.CheckGlobalThis  
In method com.google.javacscript.jscmp.CheckGlobalThis.shouldTraverse(NodeTraversal, Node, Node) Local variable stored in JVM register 5 At CheckGlobalThis.java:[line 103] Dodgy  
code Dead store to local variable <p> 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. </p> <p> 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. </p> 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/antlr\_deploy.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/antlr4\_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/tibtrunk\_parser.jar  
Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-100/lib/protobuf\_deploy.jar Dead store to local variable Dead store to SL5 in  
com.google.javacscript.jscmp.CheckGlobalThis.shouldTraverseNodeTraversal(Node, Node) At CheckGlobalThis.java:[line 61-181] In class com.google.javacscript.jscmp.CheckGlobalThis  
In method com.google.javacscript.jscmp.CheckGlobalThis.shouldTraverseNodeTraversal(Node, Node) Local variable stored in JVM register 5 At CheckGlobalThis.java:[line 110] Dodgy  
code Dead store to local variable <p> 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. </p> </p> 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. </p> Dead local store

code diff:

```
92     }
93
94     // Don't traverse functions unless they would no
95     // rmally
96     // be able to have a @this annotation associated
97     // with them. e.g.,
98     // var a = function() { }; // or
99
100    // function a() {} // or
101    // a.x = function() {};
102
103    }
104
105    if (parent != null && parent.getType() == Token.AS
106      SIGN) {
107      Node lhs = parent.getFirstChild();
108      Node rhs = lhs.getNext();
109
110      if (n == lhs) {
111          // Always traverse the left side of the assign
112          // ment. To handle
113          // nested assignments properly (e.g., (a = thi
114          s).property = c);
115          // assignLhsChild should not be overridden.
116          if (assignLhsChild == null) {
117              assignLhsChild = lhs;
118          }
119          } else {
120              // Only traverse the right side if it's not an
121              // assignment to a prototype
122              // property or subproperty.
123
124      }
125
126      Editor
127
128      }
129
130      if (! (pType == Token.BLOCK ||
131            pType == Token.SCRIPT ||
132            pType == Token.NAME ||
133            pType == Token.ASSIGN)) {
134          return false;
135      }
136
137      if (parent != null && parent.getType() == Token.AS
138        SIGN) {
139          Node lhs = parent.getFirstChild();
140          Node rhs = lhs.getNext();
141
142          if (n == lhs) {
143              // Always traverse the left side of the assign
144              // ment. To handle
145              // nested assignments properly (e.g., (a = thi
146              s).property = c);
147              // assignLhsChild should not be overridden.
148              if (assignLhsChild == null) {
149                  assignLhsChild = lhs;
150              }
151              } else {
152                  // Only traverse the right side if it's not an
153                  // assignment to a prototype
154                  // property or subproperty.
```

```
128 }
129
130 public void visit(NodeTraversal t, Node n, Node parent) {
131     if (n.getType() == Token.THIS && shouldReportThis(n, parent)) {
132         compiler.report(t.makeError(n, level, GLOBAL_THIS));
133     }
134     if (n == assignLhsChild) {
135         assignLhsChild = null;
136     }
137 }
138
139 private boolean shouldReportThis(Node n, Node parent) {
140     if (assignLhsChild != null) {
141         // Always report a THIS on the left side of an assignment.
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 any. 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() {};
157 */
158
159 public void visit(NodeTraversal t, Node n, Node parent) {
160     if (n.getType() == Token.THIS && shouldReportThis(n, parent)) {
161         compiler.report(t.makeError(n, level, GLOBAL_THIS));
162     }
163     if (n == assignLhsChild) {
164         assignLhsChild = null;
165     }
166 }
167
168 private boolean shouldReportThis(Node n, Node parent) {
169     if (assignLhsChild != null) {
170         // Always report a THIS on the left side of an assignment.
171         return true;
172     }
173
174     // Also report a THIS with a property access.
175     return parent != null && NodeUtil.isGet(parent);
176 }
177
178 /**
179 * Gets a function's JSDoc information, if it has any. Checks for a few
180 * patterns (ellipses show where JSDoc would be):
181 * <pre>
182 * ... function() {}
183 * ... x = function() {};
184 * var ... x = function() {};
185 * ... var x = function() {};
186 */
187
188 public void visit(NodeTraversal t, Node n, Node parent) {
189     if (n.getType() == Token.THIS && shouldReportThis(n, parent)) {
190         compiler.report(t.makeError(n, level, GLOBAL_THIS));
191     }
192     if (n == assignLhsChild) {
193         assignLhsChild = null;
194     }
195 }
196
197 private boolean shouldReportThis(Node n, Node parent) {
198     if (assignLhsChild != null) {
199         // Always report a THIS on the left side of an assignment.
200         return true;
201     }
202
203     // Also report a THIS with a property access.
204     return parent != null && NodeUtil.isGet(parent);
205 }
```

## Lang 55:

OPEN FILES

diffs\_parsed.json x ep\_diffs\_warnings.json x ep\_removed\_warning x inf\_diffs\_warnings.json x inf\_removed\_warning x sb\_diffs\_warnings.json x sb\_removed\_warnings.json

```
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": [

```

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, allowing 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.
124      *
125      * <p>This method clears the internal values to allow 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, allowing 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.
126      *
127      * <p>This method clears the internal values to allow 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      /**
137      * <p>Split the time.</p>
138      *
```

### Math 94:

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

```

5262      405,
5263      380,
5264      345,
5265      346,
5266      347,
5267      348,
5268      349,
5269      382
5270  ],
5271  {
5272    "Project": ": \"Math-94",
5273    "Class": ": \"org.apache.commons.math.util.MathUtils",
5274    "Lines": "[
5275      412
5276    ]
5277  },
5278  {
5279    "Project": ": \"Math-95",
5280    "Class": ": \"org.apache.commons.math.distribution.FDistributionImpl",
5281    "Lines": [
5282      144,
5283      145,
5284      146,
5285      147,
5286      148,
5287      149,
5288      150
5289    ]
5290  },
5291  {
5292    "Project": ": \"Math-96",
5293    "Class": ": \"org.apache.commons.math.complex.Complex",
5294    "Lines": [
5295      258
5296    ]
5297  },
5298

```

StopWatch

7 lines, 153 characters selected

Find Find Prev Find All

Spaces: 4 JSON

buggy warnings: fixed warnings:  
(same)

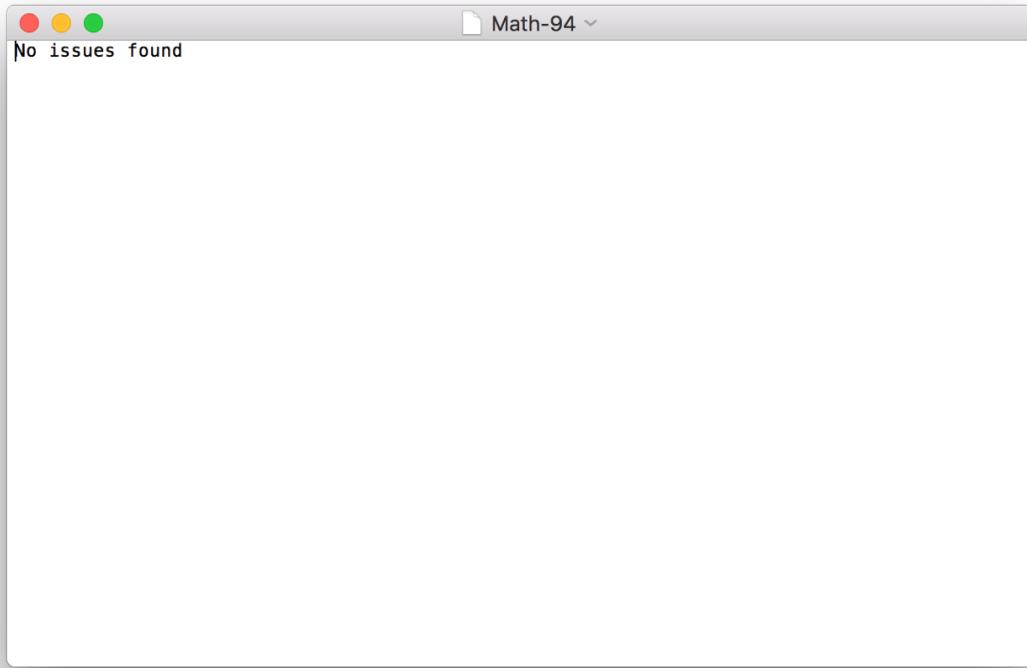
Error prone: false-positive

```

/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 division and modulo
402   * operations. See Knuth 4.5.2 algorithm B. This algorithm 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
416     // range down to
417     // -231, while positive numbers can only be a
418     // s large as 231-1
419     // (i.e. we can't necessarily negate a negative
420     // e number without
421     // overflow)
422     /* assert ui=0 && vi=0; */
423     if (u > 0) {
424       u = -u;
425     }
426     } // make u negative
427     if (v > 0) {
428       v = -v;
429     }
430     } // make v negative
431     // B1. [Find power of 2]
432     int k = 0;
433     while ((u > 0) && (v > 0)) {
434       if (u > v) {
435         u = u - v;
436       } else {
437         v = v - u;
438       }
439       k++;
440     }
441     if (u == 0) {
442       return v;
443     } else {
444       return u;
445     }
446   }
447 }
```

---

```

e value of two numbers,
401   * using the "binary gcd" method which avoids division and modulo
402   * operations. See Knuth 4.5.2 algorithm B. This algorithm 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
416     // range down to
417     // -231, while positive numbers can only be a
418     // s large as 231-1
419     // (i.e. we can't necessarily negate a negative
420     // e number without
421     // overflow)
422     /* assert ui=0 && vi=0; */
423     if (u > 0) {
424       u = -u;
425     }
426     } // make u negative
427     if (v > 0) {
428       v = -v;
429     }
430     } // make v negative
431     // B1. [Find power of 2]
432     int k = 0;
433     while ((u > 0) && (v > 0)) {
434       if (u > v) {
435         u = u - v;
436       } else {
437         v = v - u;
438       }
439       k++;
440     }
441     if (u == 0) {
442       return v;
443     } else {
444       return u;
445     }
446   }
447 }
```

### Mockito-19:

```

5562     "Project": ":"Mockito-19",
5563     "Class": ":"org.mockito.internal.configuration.injection.PropertyAndSetterInjection",
5564     "Lines": ":"[114]
5565   },
5566   {
5567     "Project": ":"Mockito-19",
5568     "Class": ":"org.mockito.internal.configuration.injection.filter.FinalMockCandidateFilter",
5569     "Lines": ":"[12,
5570             13,
5571             14,
5572             23,
5573             24,
5574             25]
5575   },
5576   {
5577     "Project": ":"Mockito-19",
5578     "Class": ":"org.mockito.internal.configuration.injection.filter.MockCandidateFilter",
5579     "Lines": ":"[8,
5580             9,
5581             16,
5582             15,
5583             16,
5584             17]
5585   },
5586   {
5587     "Project": ":"Mockito-19",
5588     "Class": ":"org.mockito.internal.configuration.injection.filter.NameBasedCandidateFilter",
5589     "Lines": ":"[23,
5590             31,
5591             41,
5592             42,
5593             43,
5594             44,
5595             45,
5596             46,
5597             47,
5598             48,
5599             49,
5600             50,
5601             51,
5602             52,
5603             53,
5604             54,
5605             55,
5606             56,
5607             57,
5608             58,
5609             59,
5610             60]
5611   },
5612   {
5613     "Project": ":"Mockito-19",
5614     "Class": ":"org.mockito.internal.configuration.injection.filter.TypeBasedCandidateFilter",
5615     "Lines": ":"[28,
5616             28]
5617   },
5618   {
5619     "Project": ":"Mockito-2",
5620   }
5621 },
5622 {
5623   "Project": ":"Mockito-19",
5624   "Class": ":"org.mockito.internal.creation.MockSettingsImpl",
5625   "Lines": ":"[28,
5626             28]
5627 },
5628 },
5629 {
5630   "Project": ":"Mockito-2",
5631

```

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

OPEN FILES

- diff\_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

```

157     "Msg": "Serializable implements method in MockSettings; expected @override",
158     "Code": "    public MockSettings serializable() {",
159     "Mark": "        ^",
160     "Line": 21
161   },
162   {
163     "Proj": "Mockito-17",
164     "Class": "org.mockito.internal.creation.MockSettingsImpl",
165     "Type": "warning",
166     "Cat": "MissingOverride",
167     "Msg": "extraInterfaces implements method in MockSettings; expected @override",
168     "Code": "    public MockSettings extraInterfaces(Class<?>... extraInterfaces) {",
169     "Mark": "        ^",
170     "Line": 25
171   },
172   [
173     "Proj": "Mockito-19",
174     "Class": "org.mockito.internal.configuration.injection.filter.FinalMockCandidateFilter",
175     "Type": "warning",
176     "Cat": "MissingOverride",
177     "Msg": "filterCandidate implements method in MockCandidateFilter; expected @override",
178     "Code": "    public OngoingInjector filterCandidate(final Collection<Object> mocks, final Field field,
179     "Mark": "        ^",
180     "Line": 23
181   },
182   {
183     "Proj": "Mockito-19",
184     "Class": "org.mockito.internal.configuration.injection.filter.TypeBasedCandidateFilter",
185     "Type": "warning",
186     "Cat": "MissingOverride",
187     "Msg": "filterCandidate implements method in MockCandidateFilter; expected @override",
188     "Code": "    public OngoingInjector filterCandidate(Collection<Object> mocks, Field field, Object field,
189     "Mark": "        ^",
190     "Line": 20
191   },
192   {
193     "Proj": "Mockito-25",
194     "Class": "org.mockito.internal.stubbing.defaultanswers.ReturnsDeepStubs",
195     "Type": "warning",
196     "Cat": "MissingOverride",
197     "Msg": "Answer implements method in Answer; expected @override",
198     "Code": "        public Object answer(InvocationOnMock invocation) throws Throwable {",

```

20 lines, 1007 characters selected

Find Find Prev Find All Spaces: 4 JSON

OPEN FILES

```

diffs_parsed.json × ep_diffs_warnings.json × ep_removed_warning × inf_diffs_warnings.json × inf_removed_warning × sb_diffs_warnings.json × sb_removed_warning ×
 62      },
 63      {
 64          "Proj": "Math-66",
 65          "Class": "org.apache.commons.math.optimization.univariate.BrentOptimizer",
 66          "Type": "warning",
 67          "Cat": "MissingOverride",
 68          "Msg": "optimize overrides method in AbstractUnivariateRealOptimizer; expected @Override",
 69          "Code": "    public double optimize(final UnivariateRealFunction f, final GoalType goalType, final doubl",
 70          "Mark": "        ^",
 71          "Line": 65
 72      },
 73      {
 74          "Proj": "Math-77",
 75          "Class": "org.apache.commons.math.linear.OpenMapRealVector",
 76          "Type": "warning",
 77          "Cat": "MissingOverride",
 78          "Msg": "getLInfNorm overrides method in AbstractRealVector; expected @Override",
 79          "Code": "    public double getLInfNorm() {",
 80          "Mark": "        ^",
 81          "Line": 498
 82      },
 83      {
 84          "Proj": "Mockito-19",
 85          "Class": "org.mockito.internal.configuration.injection.filter.TypeBasedCandidateFilter",
 86          "Type": "warning",
 87          "Cat": "MissingOverride",
 88          "Msg": "filterCandidate implements method in MockCandidateFilter; expected @Override",
 89          "Code": "... public OngoingInjector filterCandidate(Collection<Object> mocks, Field field, Object field",
 90          "Mark": "... ^",
 91          "Line": 20
 92      },
 93      {
 94          "Proj": "Mockito-19",
 95          "Class": "org.mockito.internal.configuration.injection.filter.FinalMockCandidateFilter",
 96          "Type": "warning",
 97          "Cat": "MissingOverride",
 98          "Msg": "filterCandidate implements method in MockCandidateFilter; expected @Override",
 99          "Code": "... public OngoingInjector filterCandidate(final Collection<Object> mocks, final Field field,
100          "Mark": "... ^",
101      }
102  ]

```

Find Find Prev Find All

10 lines, 493 characters selected Spaces: 4 JSON

## 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 OngoingInjector filterCandidate(Collection<Object> mocks,
                                         ^
(see http://errorprone.info/bugpattern/MissingOverride)
Did you mean '@Override public OngoingInjector 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 OngoingInjector filterCandidate(Collection<Object> mocks, Field field, Object fieldInstance) {
                                         ^
(see http://errorprone.info/bugpattern/MissingOverride)
Did you mean '@Override public OngoingInjector 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 OngoingInjector filterCandidate(Collection<Object> mocks, Field field, List<Field> fields, Object fieldInstance) {
                                         ^
(see http://errorprone.info/bugpattern/MissingOverride)
Did you mean '@Override public OngoingInjector 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 OngoingInjector 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 OngoingInjector 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

0 Removals + 2 Additions

```

1 import org.mockito.internal.util.reflection.FieldSetter;
2 import java.lang.reflect.Field;
3 import java.util.Collection;
4
5 /**
6  * This node returns an actual injector which will be ei
7  * ther :
8  * <ul>
9  * <li>an {@link OngoingInjector} that do nothing if a c
10 candidate couldn't be found</li>
11 * <li>an {@link OngoingInjector} that will try to injec
12 t the candidate trying first the property setter then if
13 not possible try the field access</li>
14 */
15 /**
16  * This node returns an actual injector which will be ei
17 * ther :
18 * <ul>
19 * <li>an {@link OngoingInjector} that do nothing if a c
20 candidate couldn't be found</li>
21 * <li>an {@link OngoingInjector} that will try to injec
22 t the candidate trying first the property setter then if
23 not possible try the field access</li>
24 */
25 public class FinalMockCandidateFilter implements MockCan
26 dicateFilter {
27     public OngoingInjector filterCandidate(Collection<Obj
28 ect> mocks, final Field field,
29     final Object fieldInstance) {
30         if(mocks.size() == 1) {
31             final Object matchingMock = mocks.iterator
32             (.next());
33             return new OngoingInjector() {
34                 public Object thenInject() {
35                     try {
36                         if (!new BeanPropertySetter(fiel
37 dInstance, field).set(matchingMock)) {
38                             new FieldSetter(fieldInstanc
39 e, field).set(matchingMock);
40                         }
41                     } catch (RuntimeException e) {
42                         new Reporter().cannotInj
43                         ict();
44                     }
45                 }
46             }
47         }
48     }
49 }

```

Recent diffs are deleted on refresh

Saved Diffs You must log in to save diffs

Diffchecker Home Images PDF Contact CLI Desktop Sign in Create a free account

Recent Diffs Clear

0 Removals + 2 Additions

```

1 package org.mockito.internal.configuration.injection.fil
2 ter;
3 import java.lang.reflect.Field;
4 import java.util.ArrayList;
5 import java.util.Collection;
6 import java.util.List;
7
8 public class TypeBasedCandidateFilter implements MockCan
9 dicateFilter {
10     MockCandidateFilter next;
11
12     public TypeBasedCandidateFilter(MockCandidateFilter
13         next) {
14         this.next = next;
15     }
16
17     public OngoingInjector filterCandidate(Collection<Ob
18 ject> mocks, Field field,
19     Object fieldInstance) {
20         List<Object> mockTypeMatches = new ArrayList<Obj
21 ect>();
22         for (Object mock : mocks) {
23             if (field.getType().isAssignableFrom(mock.ge
24 tClass())) {
25                 mockTypeMatches.add(mock);
26             }
27         }
28         return next.filterCandidate(mockTypeMatches, fie
29 ld,
30     fieldInstance);
31     }
32 }

```

Recent diffs are deleted on refresh

Saved Diffs You must log in to save diffs

**Time-14:**

OPEN FILES

diffs\_parsed.json x ep\_diffs\_warnings.json x ep\_removed\_warning x inf\_diffs\_warnings.json x inf\_removed\_warning x sb\_diffs\_warnings.json x sb\_removed\_warning

```
2229      ],
6153      1098,
6154      1132,
6155      1133,
6156      1134,
6157      1142,
6158      1143,
6159      1144,
6160      1145,
6161      1146
6162    ],
6163  },
6164  {
6165    "Project": ": \"Time-14\"",
6166    "Class": ": \"org.joda.time.chrono.BasicMonthOfYearDateTimeField\"",
6167    "Lines": [
6168      208,
6169      209,
6170      210,
6171      211,
6172      212,
6173      213,
6174      214,
6175      215
6176    ],
6177  },
6178  {
6179    "Project": ": \"Time-15\"",
6180    "Class": ": \"org.joda.time.field.FieldUtils\"",
6181    "Lines": [
6182      137,
6183      138,
6184      139,
6185      140,
6186      141
6187    ],
6188  },
6189  {
6190    "Project": ": \"Time-16\"",
6191    "Class": ": \"org.joda.time.format.DateTimeFormatter\"",
6192    "Lines": [
6193      709
6194    ]
6195  }
6196]
```

. Aa " Find Find Prev Find All

14 lines, 284 characters selected Spaces: 4 JSON

\*\*\*No tool detects the bug\*\*\*

buggy warnings: fixed warnings:

```
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/fTime-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:58: warning: [MissingOverride]
isLenient implements method in DateTextField; expected @Override
    public boolean isLenient() {
        ^
        (see http://errorprone.info/bugpattern/MissingOverride)
        Did you mean @Override public boolean isLenient() {'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/fTime-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/fTime-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/fTime-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/fTime-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 @Override public long add(int monthToUse + 1;?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/fTime-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/fTime-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/fTime-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/fTime-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/fTime-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/fTime-14/src/main/java/org/joda/time/chrono/BasicMonthOfYearDateTimeField.java:319: warning: [MissingOverride]
isClosed implements method in RangeDurationField; expected @Override
    isClosed(RangeDurationField rangeDurationField) {
```

**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.)**

Screenshot of Diffchecker.com showing a code diff between two Java files. The left file has lines 203-222, and the right file has lines 203-298. A green box highlights a section of code in the right file starting at line 209.

```

203     public int[] add(ReadablePartial partial, int fieldIndex, int[] values, int valueToAdd) {
204         // overridden as superclass algorithm can't handle
205         // 2004-02-29 + 48 months -> 2008-02-29 type dates
206         if (valueToAdd == 0) {
207             return values;
208         }
209         // month is largest field and being added to, such as month-day
210         if (DateTimeUtils.isContiguous(partial)) {
211             long instant = 0L;
212             for (int i = 0, isize = partial.size(); i < isize; i++) {
213                 instant = partial.getFieldType(i).getField(iChronology).set(instant, values[i]);
214             }
215             instant = add(instant, valueToAdd);
216             return iChronology.get(partial, instant);
217         } else {
218             return super.add(partial, fieldIndex, values, valueToAdd);
219         }
220     }
221     //-----
222     /**
223      */
224 
```

**Chart-22:**

Screenshot of a code editor showing a JSON file named 'diffs\_parsed.json'. The file contains a large array of objects, each representing a difference between two versions of a chart. One object is highlighted with a yellow background, showing its properties: Project: "Chart-22", Class: "org.jfree.data.KeyedObject2D", Lines: "[...]".

```

OPEN FILES
diffs_parsed.json x ep_diffs_warnings.json x ep_removed_warning x inf_diffs_warnings.json x inf_removed_warning x sb_diffs_warnings.json x sb_removed_warning x

246     },
247     {
248         "Project": "Chart-22",
249         "Class": "org.jfree.data.KeyedObject2D",
250         "Lines": "[...]"
251         483,
252         484,
253         485,
254         486,
255         487,
256         488,
257         318,
258         319,
259         320,
260         321,
261         322,
262         323,
263         324,
264         325,
265         326,
266         327,
267         328,
268         329,
269         330,
270         331,
271         332,
272         333,
273         334,
274         335,
275         336,
276         337,
277         338,
278         339,
279         340,
280         341,
281         342,
282         231,
283         232,
284         233,
285         234,
286         235,
287         366,
288         367,
289         368,
290         369,
291         370,
292         371,
293         372,
294         373,
295         374,
296         375,
297         376,
298         377
299     ],
300     {
301         "Project": "Chart-23",
302         "Class": "org.jfree.chart.renderer.category.MinMaxCategoryRenderer"
303     }
304 
```

49 lines, 858 characters selected      Spaces: 4      JSON

spotbugs: meth. 1, 2 (full match)

OPEN FILES

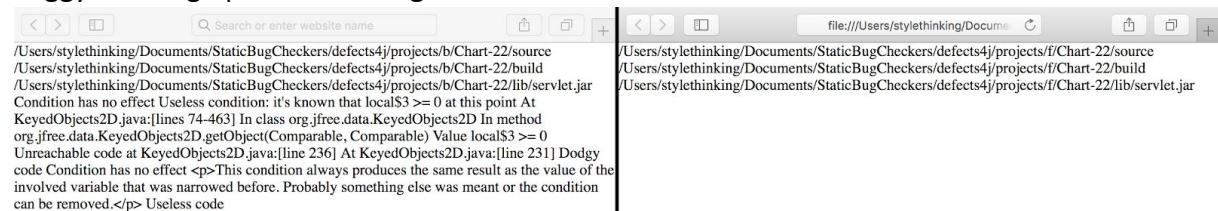
```

diffs_parsed.json x ep_diffs_warnings.json x ep_removed_warning x inf_diffs_warnings.json x inf_removed_warning x sb_diffs_warnings.json x sb_removed_warning x
28     "Rank": "14",
29     "Msg": "org.jfree.data.statistics.DefaultBoxAndWhiskerCategoryDataset defines equals an
30     "Method": "equals",
31     "Field": "",
32     "Lines": [
33         [
34             752,
35             760,
36             null
37         ]
38     ],
39 },
40 {
41     "Proj": "Chart-22",
42     "Class": "org.jfree.data.KeyedObjects2D",
43     "Cat": "STYLE",
44     "Abbrev": "UC",
45     "Type": "UC_USELESS_CONDITION",
46     "Priority": "1",
47     "Rank": "14",
48     "Msg": "Useless condition: it's known that local$3 >= 0 at this point",
49     "Method": "getObject",
50     "Field": "",
51     "Lines": [
52         [
53             236,
54             236,
55             "SOURCE_UNREACHABLE_CODE"
56         ],
57         [
58             231,
59             231,
60             null
61         ]
62     ],
63 },
64 {
65     "Proj": "Chart-4",
66     "Class": "org.jfree.chart.plot.XYPlot",
67     "Cat": "CORRECTNESS",
68     "Abbrev": "NP",
69     "Msg": "No effect on some paths"

```

24 lines, 622 characters selected      Spaces: 4      JSON

## Buggy warnings | Fixed warnings:


 /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-22/source  
 /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-22/build  
 /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Chart-22/lib/servlet.jar  
 Condition has no effect Useless condition: it's known that local\$3 >= 0 at this point At  
 KeyedObjects2D.java:[lines 74-463] In class org.jfree.data.KeyedObjects2D In method  
 org.jfree.data.KeyedObjects2D.getObject(Comparable, Comparable) Value local\$3 >= 0  
 Unreachable code at KeyedObjects2D.java:[line 236] At KeyedObjects2D.java:[line 231] Dodgy  
 code Condition has no effect <p>This condition always produces the same result as the value of the  
 involved variable that was narrowed before. Probably something else was meant or the condition  
 can be removed.</p> Useless code

## Code diff:

### Closure-74:

OPEN FILES

diffs\_parsed.json x ep\_diffs\_warnings.json x ep\_removed\_warning x inf\_diffs\_warnings.json x inf\_removed\_warning x sb\_diffs\_warnings.json x sb\_removed\_warning

```
1970     "Class: ": "com.google.javascript.jscomp.RenameLabels",
1971     "Lines: ": [
1972       215
1973     ]
1974   },
1975   {
1976     "Project: ": "Closure-73",
1977     "Class: ": "com.google.javascript.jscomp.CodeGenerator",
1978     "Lines: ": [
1979       1045
1980     ]
1981   },
1982   {
1983     "Project: ": "Closure-74",
1984     "Class: ": "com.google.javascript.jscomp.PeepholeFoldConstants",
1985     "Lines: ": [
1986       907,
1987       998
1988       1073,
1989       1074,
1990       1075,
1991       1076,
1992       1077,
1993       1078,
1994       1079,
1995       1080,
1996       1081,
1997       1082,
1998       1083,
1999       1084,
2000       1085,
2001       1086,
2002       1087
2003     ],
2004   },
2005   {
2006     "Project: ": "Closure-75",
2007     "Class: ": "com.google.javascript.jscomp.NodeUtil",
2008     "Lines: ": [
2009     ]
2010   }
2011 }
```

Find Find Prev Find All

23 lines, 454 characters selected Spaces: 4 JSON

## Spot bugs: meth 1 (mismatch)

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

```

144      157,
145      183,
146      null
147    ],
148  },
149  [
150    {
151      "Proj": "Closure-74",
152      "Class": "com.google.javascript.jscomp.PeepholeFoldConstants",
153      "Cat": "BAD_PRACTICE",
154      "Abbrev": "NP",
155      "Type": "NP_BOOLEAN_RETURN_NULL",
156      "Priority": "2",
157      "Rank": "14",
158      "Msg": "com.google.javascript.jscomp.PeepholeFoldConstants.compareAsNumbers(int, Node, Method) compareAsNumbers",
159      "Field": "",
160      "Lines": [
161        [
162          1082,
163          1082,
164          null
165        ],
166        [
167          [
168            1086,
169            1086,
170            "SOURCE_LINE_ANOTHER_INSTANCE"
171          ],
172          [
173            1111,
174            1111,
175            "SOURCE_LINE_ANOTHER_INSTANCE"
176          ]
177        ],
178      ],
179    },
180    {
181      "Proj": "Closure-91",
182      "Class": "com.google.javascript.jscomp.CheckGlobalThis",
183      "Cat": "STYLE"
184    }
185  ],
186  [
187    {
188      "Proj": "Closure-91",
189      "Class": "com.google.javascript.jscomp.CheckGlobalThis",
190      "Cat": "STYLE"
191    }
192  ]
193 ]
194

```

Find Find Prev Find All  
Spaces: 4 JSON  
29 lines, 861 characters selected

## Buggy Warnings: Fixed Warning: same

file:///Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-74/lib/protobuf-java.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Closure-74/lib/ant.jar Test for floating point equality Test for floating point equality in com.google.javascript.jscomp.PeepholeFoldConstants.performArithmeticOp(int, Node, Node) At PeepholeFoldConstants.java:[lines 33-1703] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.performArithmeticOp(int, Node, Node) At PeepholeFoldConstants.java:[line 737] Another occurrence at PeepholeFoldConstants.java:[line 739] Method with Boolean return type returns explicit null com.google.javascript.jscomp.PeepholeFoldConstants.compareAsNumbers(int, Node, Node) has Boolean return type and returns explicit null At PeepholeFoldConstants.java:[lines 33-1703] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.compareAsNumbers(int, Node, Node) At PeepholeFoldConstants.java:[line 1082] Another occurrence at PeepholeFoldConstants.java:[line 1086] Another occurrence at PeepholeFoldConstants.java:[line 1111] Switch statement found where default case is missing Switch statement found in com.google.javascript.jscomp.PeepholeFoldConstants.tryFoldTypeof(Node) where default case is missing At PeepholeFoldConstants.java:[lines 33-1703] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.tryFoldTypeof(Node) At PeepholeFoldConstants.java:[lines 283-309] Switch statement found where default case is missing Switch statement found in com.google.javascript.jscomp.PeepholeFoldConstants.tryReduceOperandsForOp(Node) where default case is missing At PeepholeFoldConstants.java:[lines 33-1703] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.tryReduceOperandsForOp(Node) At PeepholeFoldConstants.java:[lines 177-211] Bad practice Dodgy code Test for floating point equality <p> This operation compares two floating point values for equality. Because floating point calculations may involve rounding, calculated float and double values may not be accurate. For values that must be precise, such as monetary values, consider using a fixed-precision type such as BigDecimal. For values that need not be precise, consider comparing for equality within some range, for example: <code>if (Math.abs(x - y) < .000001 )</code>. See the Java Language Specification, section 4.2.4. <p> Method with Boolean return type returns explicit null <p> A method that returns either Boolean.TRUE, Boolean.FALSE or null is an accident waiting to happen. This method can be invoked as though it returned a value of type boolean, and the compiler will insert automatic unboxing of the Boolean value. If a null value is returned, this will result in a NullPointerException. <p> Switch statement found where default case is missing <p> This method contains a switch statement where default case is missing. Usually you need to provide a default case.</p> <p> Because the analysis only looks at the generated bytecode, this warning can be incorrect triggered if the default case is at the end of the switch statement and the switch statement doesn't contain break statements for other cases. Null pointer dereference Switch case falls through Test for floating point equality

file:///Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-74/lib/protobuf-java.jar /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Closure-74/lib/ant.jar Test for floating point equality Test for floating point equality in com.google.javascript.jscomp.PeepholeFoldConstants.performArithmeticOp(int, Node, Node) At PeepholeFoldConstants.java:[lines 33-1716] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.compareAsNumbers(int, Node, Node) has Boolean return type and returns explicit null At PeepholeFoldConstants.java:[lines 33-1716] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.compareAsNumbers(int, Node, Node) At PeepholeFoldConstants.java:[line 1095] Another occurrence at PeepholeFoldConstants.java:[line 1099] Another occurrence at PeepholeFoldConstants.java:[line 1124] Switch statement found where default case is missing Switch statement found in com.google.javascript.jscomp.PeepholeFoldConstants.tryFoldTypeof(Node) where default case is missing At PeepholeFoldConstants.java:[lines 33-1716] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.tryFoldTypeof(Node) At PeepholeFoldConstants.java:[lines 283-309] Switch statement found where default case is missing Switch statement found in com.google.javascript.jscomp.PeepholeFoldConstants.tryReduceOperandsForOp(Node) where default case is missing At PeepholeFoldConstants.java:[lines 33-1716] In class com.google.javascript.jscomp.PeepholeFoldConstants In method com.google.javascript.jscomp.PeepholeFoldConstants.tryReduceOperandsForOp(Node) At PeepholeFoldConstants.java:[lines 177-211] Bad practice Dodgy code Test for floating point equality <p> This operation compares two floating point values for equality. Because floating point calculations may involve rounding, calculated float and double values may not be accurate. For values that must be precise, such as monetary values, consider using a fixed-precision type such as BigDecimal. For values that need not be precise, consider comparing for equality within some range, for example: <code>if ( Math.abs(x - y) < .000001 )</code>. See the Java Language Specification, section 4.2.4. <p> Method with Boolean return type returns explicit null <p> A method that returns either Boolean.TRUE, Boolean.FALSE or null is an accident waiting to happen. This method can be invoked as though it returned a value of type boolean, and the compiler will insert automatic unboxing of the Boolean value. If a null value is returned, this will result in a NullPointerException. <p> Switch statement found where default case is missing <p> This method contains a switch statement where default case is missing. Usually you need to provide a default case.</p> <p> Because the analysis only looks at the generated bytecode, this warning can be incorrect triggered if the default case is at the end of the switch statement and the switch statement doesn't contain break statements for other cases. Null pointer dereference Switch case falls through Test for floating point equality

## 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-4>

**Diffchecker** Home Images PDF Contact CLI Desktop Sign in Create a free account

Recent Diffs Clear

3 Removals + 16 Additions

Advanced Split Unified Word Character

```

1072     * @return Translate NOT expressions into TRUE or FA 1072     * @return Translate NOT expressions into TRUE or FA
1073     LSE when possible. 1073     LSE when possible.
1074     */
1075     private int getNormalizedNodeType(Node n) {
1076         int type = n.getType();
1077         if (type == Token.NOT) {
1078             TernaryValue value = NodeUtil.getPureBooleanValue(n);
1079             switch (value) {
1080                 case TRUE:
1081                     return Token.TRUE;
1082                 case FALSE:
1083                     return Token.FALSE;
1084             }
1085         }
1086         return type;
1087     }
1088     /**
1089      * The result of the comparison as a Boolean or null
1090      * if the
1091      * result could not be determined.
1092      */
1093     private Boolean compareAsNumbers(int op, Node left,
1094         Node right) {
1095         Double leftValue = NodeUtil.getNumberValue(left);
1096         if (leftValue == null) {
1097             return null;
1098         }
1099         Double rightValue = NodeUtil.getNumberValue(right);
1100         if (rightValue == null) {
1101             return null;
1102         }
1103     }
1104 
```

Editor

### Lang-27:

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.json ep\_removed\_warning inf\_diffs\_warnings.json inf\_removed\_warning sb\_diffs\_warnings.json sb\_removed\_warning

```

2801     99,
2802     100
2803   ],
2804 },
2805 [
2806   "Project": "Lang-26",
2807   "Class": "org.apache.commons.lang3.time.FastDateFormat",
2808   "Lines": [
2809     820
2810   ],
2811 },
2812 [
2813   "Project": "Lang-27",
2814   "Class": "org.apache.commons.lang3.math.NumberUtils",
2815   "Lines": [
2816     488,
2817     489,
2818     490,
2819     491,
2820     492,
2821     479
2822   ],
2823 ],
2824 [
2825   "Project": "Lang-28",
2826   "Class": "org.apache.commons.lang3.text.translate.NumericEntityUnescaper",
2827   "Lines": [
2828     64,
2829     65,
2830     66,
2831     67,
2832     68,
2833     69,
2834     70,
2835     62,
2836     63
2837   ],
2838 ],
2839 ]

```

Find Find Prev Find All

12 lines, 240 characters selected Spaces: 4 JSON

Infer: meth 1 (false positive)

## Buggy warnings: fixed warnings:

```
TextEdit File Edit Format View Window Help
Lang-27 ~
Found 2 issues
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Lang-27/src/main/java/org/apache/commons/lang3/math/NumberUtils.java:537: error: NULL_DEREFERENCE
    object `d` last assigned on line 536 could be null and is dereferenced at line 537.
      535.             try {
      536.                 Double d = NumberUtils.createDouble(numeric);
      537.                 if (!d.isInfinite() || (d.floatValue() == 0.0F && !
allZeros)) {
      538.                     return d;
      539.                 }
    }

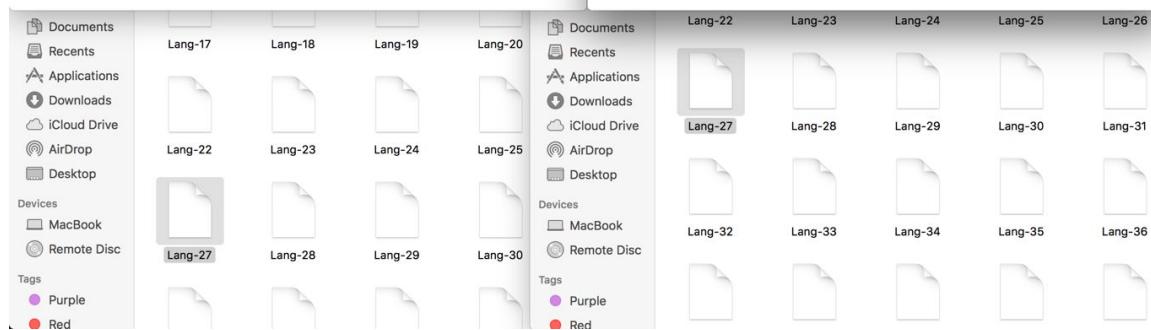
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Lang-27/src/main/java/org/apache/commons/lang3/math/NumberUtils.java:523: error: NULL_DEREFERENCE
    object `f` last assigned on line 522 could be null and is dereferenced at line 523.
      521.             try {
      522.                 Float f = NumberUtils.createFloat(numeric);
      523.                 if (!f.isInfinite() || (f.floatValue() == 0.0F && !
allZeros)) {
      524.                     // If it's too big for a float or the float value
= 0 and the string
      525.                     // has non-zeros in it, then float does not have
the precision we want
    }

Summary of the reports
NULL_DEREFERENCE: 2

Lang-27 ~
Found 2 issues
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Lang-27/src/main/java/org/apache/commons/lang3/math/NumberUtils.java:540: error: NULL_DEREFERENCE
    object `d` last assigned on line 539 could be null and is dereferenced at line 540.
      538.             try {
      539.                 Double d = NumberUtils.createDouble(numeric);
      540.                 if (!d.isInfinite() || (d.floatValue() == 0.0F && !
allZeros)) {
      541.                     return d;
      542.                 }
    }

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Lang-27/src/main/java/org/apache/commons/lang3/math/NumberUtils.java:526: error: NULL_DEREFERENCE
    object `f` last assigned on line 525 could be null and is dereferenced at line 526.
      524.             try {
      525.                 Float f = NumberUtils.createFloat(numeric);
      526.                 if (!f.isInfinite() || (f.floatValue() == 0.0F && !
allZeros)) {
      527.                     // If it's too big for a float or the float value
= 0 and the string
      528.                     // has non-zeros in it, then float does not have
the precision we want
    }

Summary of the reports
NULL_DEREFERENCE: 2
```



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 Find | +

https://www.diffchecker.com/diff#left-3

# Diffchecker

Home Images PDF Contact CLI Desktop Sign in Create a free account

Recent Diffs Clear

1 Removal + 4 Additions Advanced Split Unified Word Character

Recent diffs are deleted on refresh

Saved Diffs You must log in to save your diffs

```
474     int expPos = str.indexOf('e') + str.indexOf('E') + 1;
475
476     if (decPos > -1) {
477
478         if (expPos > -1) {
479             if (expPos < decPos) {
480                 throw new NumberFormatException(str + " is not a valid number.");
481             }
482             dec = str.substring(decPos + 1, expPos);
483         } else {
484             dec = str.substring(decPos + 1);
485         }
486         mant = str.substring(0, decPos);
487     } else {
488         if (expPos > -1) {
489             mant = str.substring(0, expPos);
490         } else {
491             mant = str;
492         }
493         dec = null;
494     }
495     if (!Character.isDigit(lastChar) && lastChar != '.') {
496         if (expPos > -1 && expPos < str.length() - 1) {
497             exp = str.substring(expPos + 1, str.length() - 1);
498         } else {
499             if (expPos > -1 && expPos < str.length() - 1) {
500                 exp = str.substring(expPos + 1, str.length() - 1);
501             } else {
502                 exp = str.substring(expPos + 1, str.length());
503             }
504         }
505     }
506     if (mant == null) {
507         mant = str;
508     }
509     if (dec == null) {
510         dec = str.substring(str.length() - 1);
511     }
512     if (exp == null) {
513         exp = str.substring(str.length() - 1);
514     }
515 }
```

Math-44:

OPEN FILES

diffs\_parsed.json x ep\_diffs\_warnings.json x ep\_removed\_warnings.json x inf\_diffs\_warnings.json x inf\_removed\_warnings.json x sb\_diffs\_warnings.json x sb\_removed\_warnings.json

```
4299      414,
4300      415
4301    ],
4302  },
4303  {
4304    "Project": ": \"Math-43",
4305    "Class": ": \"org.apache.commons.math.stat.descriptive.SummaryStatistics",
4306    "Lines": ": [
4307      161,
4308      164,
4309      158
4310    ]
4311  },
4312  [
4313    "Project": ": \"Math-44",
4314    "Class": ": \"org.apache.commons.math.ode.AbstractIntegrator",
4315    "Lines": ": [
4316      344,
4317      332,
4318      333,
4319      334,
4320      335,
4321      336,
4322      280,
4323      345,
4324      346,
4325      347,
4326      348
4327    ],
4328  },
4329  {
4330    "Project": ": \"Math-45",
4331    "Class": ": \"org.apache.commons.math.linear.OpenMapRealMatrix",
4332    "Lines": ": [
4333      49,
4334      50,
4335      51,
4336      52,
4337      53,
4338      54,
4339      55
```

## Spot bugs: meth 1 (full match)

OPEN FILES

```

diffs_parsed.json * ep_diffs_warnings.json * ep_removed_warning * inf_diffs_warnings.json * inf_removed_warning * sb_diffs_warnings.json * sb_removed_warning *
diffs_parsed.json
276     "Abbrev": "CN",
279     "Type": "CN_IDIOM",
280     "Priority": "2",
281     "Rank": "16",
282     "Msg": "Class org.apache.commons.lang.text.StrBuilder implements Cloneable but does not define or implement its own serialVersionUID field. This is a potential security risk as it can be exploited by an attacker to create a clone of the object that is not fully initialized, leading to undefined behavior or security vulnerabilities.", "Method": "", "Field": "", "Lines": [
283         98,
284         2203,
285         null
286     ],
287 },
288 },
289 ],
290 ],
291 },
292 },
293 {
294     "Proj": "Math-44",
295     "Class": "org.apache.commons.math.ode.AbstractIntegrator",
296     "Cat": "STYLE",
297     "Abbrev": "Urf",
298     "Type": "URF_UNREAD_PUBLIC_OR_PROTECTED_FIELD",
299     "Priority": "2",
300     "Rank": "18",
301     "Msg": "Unread public/protected field: org.apache.commons.math.ode.AbstractIntegrator.resetOccurred At AbstractIntegrator.java:[lines 85-397] In class org.apache.commons.math.ode.AbstractIntegrator In AbstractIntegrator.java Field org.apache.commons.math.ode.AbstractIntegrator.resetOccurred At AbstractIntegrator.java:[line 280] Dodgy code Unread public/protected field <p> This field is never read.&nbsp; The field is public or protected, so perhaps it is intended to be used with classes not seen as part of the analysis. If not, consider removing it from the class.</p> Unread field",
302     "Method": "",
303     "Field": "resetOccurred",
304     "Lines": [
305         280,
306         280,
307         null
308     ],
309 ],
310 },
311 },
312 },
313     "Proj": "Math-50",
314     "Class": "org.apache.commons.math.analysis.solvers.BaseSecantSolver",
315     "Cat": "STYLE",
316     "Abbrev": "FE",
317     "Type": "FE_FLOATING_POINT_EQUALITY",
318     "Priority": "1",
319     "Rank": "15",
320

```

Find Find Prev Find All  
19 lines, 576 characters selected Spaces: 4 JSON

## Warnings: shifted to next occurrence

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-44/src/main/java  
 /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-44/target/classes  
 Unread public/protected field Unread public/protected field:  
 org.apache.commons.math.ode.AbstractIntegrator.resetOccurred At AbstractIntegrator.java:[lines 85-392] In class org.apache.commons.math.ode.AbstractIntegrator In AbstractIntegrator.java Field org.apache.commons.math.ode.AbstractIntegrator.resetOccurred At AbstractIntegrator.java:[line 280] Dodgy code Unread public/protected field <p> This field is never read.&nbsp; The field is public or protected, so perhaps it is intended to be used with classes not seen as part of the analysis. If not, consider removing it from the class.</p> Unread field

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Math-44/src/main/java  
 /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Math-44/target/classes  
 Unread public/protected field Unread public/protected field:  
 org.apache.commons.math.ode.AbstractIntegrator.resetOccurred At AbstractIntegrator.java:[lines 85-397] In class org.apache.commons.math.ode.AbstractIntegrator In AbstractIntegrator.java Field org.apache.commons.math.ode.AbstractIntegrator.resetOccurred At AbstractIntegrator.java:[line 344] Dodgy code Unread public/protected field <p> This field is never read.&nbsp; The field is public or protected, so perhaps it is intended to be used with classes not seen as part of the analysis. If not, consider removing it from the class.</p> Unread field

Code diff:

## Math-66:

OPEN FILES

\* diff\_parsed.json x diff\_parsed.json x ep\_diffs\_warnings.json x ep\_removed\_warning x inf\_diffs\_warnings.json x inf\_removed\_warning x sb\_diffs\_warnings.json x sb\_removed\_warning

```
4730 },
4731 {
4732     "Project": "Math-66",
4733     "Class": "org.apache.commons.math.optimization.univariate.BrentOptimizer",
4734     "Lines": [
4735         44,
4736         46,
4737         47,
4738         57,
4739         58,
4740         59,
4741         60,
4742         189,
4743         62,
4744         190,
4745         191,
4746         65,
4747         66,
4748         67,
4749         192,
4750         200,
4751         201,
4752         94,
4753         95,
4754         227,
4755         228,
4756         229,
4757         230,
4758         231,
4759         232,
4760         109,
4761         110,
4762         111,
4763         112,
4764         238,
4765         241,
4766         243,
4767         116,
4768         117,
4769         118,
4770         119,
4771         120,
4772         126,
4773         127
4774     ],
4775 },
4776 {
```

.. Aa .. U D C O Find Find Prev Find All

45 lines, 811 characters selected Spaces: 4 JSON

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

OPEN FILES

```

diffs_parsed.json * ep_diffs_warnings.json * ep_removed_warning * inf_diffs_warnings.json * inf_removed_warning * sb_diffs_warnings.json * sb_removed_warning *
diffs_parsed.json
ep_diffs_warnings.json
inf_diffs_warnings.json
inf_removed_warnings.json
sb_diffs_warnings.json
sb_removed_warnings.json

```

```

96     " Cat": "MissingOverride",
97     " Msg": "cumulativeProbability implements method in Distribution; expected @Override",
98     " Code": "    public double cumulativeProbability(double x) throws MathException {",
99     " Mark": "        ^",
100    " Line": 108
101  },
102  [
103    " Proj": "Math-17",
104    " Class": "org.apache.commons.math3.dfp.Dfp",
105    " Type": "Warning",
106    " Cat": "MissingOverride",
107    " Msg": "multiply implements method in FieldElement; expected @Override",
108    " Code": "    public Dfp multiply(final int x) {",
109    " Mark": "        ^",
110    " Line": 1602
111  },
112  [
113    " Proj": "Math-66",
114    " Class": "org.apache.commons.math.optimization.univariate.BrentOptimizer",
115    " Type": "Warning",
116    " Cat": "MissingOverride",
117    " Msg": "optimize overrides method in AbstractUnivariateRealOptimizer; expected @Override",
118    " Code": "    public double optimize(final UnivariateRealFunction f, final GoalType goalType, final dou",
119    " Mark": "        ^",
120    " Line": 59
121  },
122  [
123    " Proj": "Math-66",
124    " Class": "org.apache.commons.math.optimization.univariate.BrentOptimizer",
125    " Type": "Warning",
126    " Cat": "MissingOverride",
127    " Msg": "optimize overrides method in AbstractUnivariateRealOptimizer; expected @Override",
128    " Code": "    ... public double optimize(final UnivariateRealFunction f, final GoalType goalType, final dou",
129    " Mark": "        ^",
130    " Line": 65
131  ],
132  [
133    " Proj": "Math-77",
134    " Class": "org.apache.commons.math.linear.OpenMapRealVector",
135    " Type": "Warning",
136    " Cat": "MissingOverride",
137    " Msg": "@etlInfNorm overrides method in AbstractRealVector: expected @Override".

```

Find Find Prev Find All  
Spaces: 4 JSON  
20 lines, 1136 characters selected

## Warnings:

TextEdit File Edit Format View Window Help

ode

Math-66

```

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-66/src/main/
java/org/apache/commons/math/optimization/univariate/BrentOptimizer.java:55: warning:
[MissingOverride] doOptimize implements method in AbstractUnivariateRealOptimizer;
expected @Override
protected double doOptimize()
  (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override protected double doOptimize()'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-66/src/main/
java/org/apache/commons/math/optimization/univariate/BrentOptimizer.java:59: warning:
[MissingOverride] optimize overrides method in AbstractUnivariateRealOptimizer; expected
@Override
public double optimize(final UnivariateRealFunction f, final GoalType goalType, final
double min, final double max, final double startValue) throws
MaxIterationsExceededException, FunctionEvaluationException {
  (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public double optimize(final UnivariateRealFunction f, final
GoalType goalType, final double min, final double max, final double startValue) throws
MaxIterationsExceededException, FunctionEvaluationException'?
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-66/src/main/
java/org/apache/commons/math/optimization/univariate/BrentOptimizer.java:65: warning:
[MissingOverride] optimize overrides method in AbstractUnivariateRealOptimizer; expected
@Override
public double optimize(final UnivariateRealFunction f, final GoalType goalType, final
double min, final double max) throws MaxIterationsExceededException,
FunctionEvaluationException {
  (see http://errorprone.info/bugpattern/MissingOverride)
  Did you mean '@Override public double optimize(final UnivariateRealFunction f, final
GoalType goalType, final double min, final double max)' throws MaxIterationsExceededException,
FunctionEvaluationException'?
Note: /Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-66/src/
main/java/org/apache/commons/math/optimization/univariate/BrentOptimizer.java uses or
overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
3 warnings

```

Tags Purple Red

Math-72 Math-73 Math-74 Math-75 Tags Purple Red

Math-72 Math-73 Math-74 Math-75 Math-76

Spotbugs: meth 1,2 (false positive)

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

```

327         null
328     ]
329   ],
330 },
331 [
332   {
333     "Proj": "Math-66",
334     "Class": "org.apache.commons.math.optimization.univariate.BrentOptimizer",
335     "Cat": "STYLE",
336     "Abrev": "FEF",
337     "Type": "FE_FLOATING_POINT_EQUALITY",
338     "Priority": "1",
339     "Rank": "15",
340     "Msg": "Test for floating point equality in org.apache.commons.math.optimization.univariate.BrentO",
341     "Method": "localMin",
342     "Field": "",
343     "Lines": [
344       [
345         224,
346         224,
347         null
348       ],
349       [
350         230,
351         238,
352         "SOURCE_LINE_ANOTHER_INSTANCE"
353       ],
354       [
355         238,
356         238,
357         "SOURCE_LINE_ANOTHER_INSTANCE"
358       ]
359     ],
360   },
361   {
362     "Proj": "Mockito-11",
363     "Class": "org.mockito.internal.creation.DelegatingMethod",
364     "Cat": "BAD_PRACTICE",
365     "Abrev": "Eq",
366     "Type": "EQ_CHECK_FOR_OPERAND_NOT_COMPATIBLE_WITH_THIS",
367     "Priority": "1",
368     "Rank": "14",
369     "Msg": "org.mockito.internal.creation.DelegatingMethod.equals(Object) checks for operand being a rea

```

Find Find Prev Find All

29 lines, 909 characters selected Spaces: 4 JSON

## Warnings:

Safari File Edit View History Bookmarks Window Help

file:///Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-66/src/main/java

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Math-66/target/classes

Test for floating point equality Test for floating point equality in org.apache.commons.math.optimization.univariate.BrentOptimizer.localMin(boolean, UnivariateRealFunction, GoalType, double, double, double, double, double) At BrentOptimizer.java:[lines 38-243] In class org.apache.commons.math.optimization.univariate.BrentOptimizer In method org.apache.commons.math.optimization.univariate.BrentOptimizer.localMin(boolean, UnivariateRealFunction, GoalType, double, double, double, double) At BrentOptimizer.java:[line 224] Another occurrence at BrentOptimizer.java:[line 230] Another occurrence at BrentOptimizer.java:[line 230] Dodgy code Test for floating point equality <p> This operation compares two floating point values for equality. Because floating point calculations may involve rounding, calculated float and double values may not be accurate. For values that must be precise, such as monetary values, consider using a fixed-precision type such as BigDecimal. For values that need not be precise, consider comparing for equality within some range, for example: <code>if ( Math.abs(x - y) < 0.000001 )</code>. See the Java Language Specification, section 4.2.4. </p> Test for floating point equality

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Math-66/src/main/java

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Math-66/target/classes

Test for floating point equality Test for floating point equality in org.apache.commons.math.optimization.univariate.BrentOptimizer.localMin(boolean, double, double, double, double, double) At BrentOptimizer.java:[lines 38-232] In class org.apache.commons.math.optimization.univariate.BrentOptimizer In method org.apache.commons.math.optimization.univariate.BrentOptimizer.localMin(boolean, double, double, double, double, double) At BrentOptimizer.java:[line 214] Another occurrence at BrentOptimizer.java:[line 220] Another occurrence at BrentOptimizer.java:[line 220] Dodgy code Test for floating point equality <p> This operation compares two floating point values for equality. Because floating point calculations may involve rounding, calculated float and double values may not be accurate. For values that must be precise, such as monetary values, consider using a fixed-precision type such as BigDecimal. For values that need not be precise, consider comparing for equality within some range, for example: <code>if ( Math.abs(x - y) &lt; .000001 )</code>. See the Java Language Specification, section 4.2.4. </p> Test for floating point equality

## Code Diff:

Diffchecker Home Images PDF Contact CLI Desktop Sign in Create a free account

Recent Diffs Clear

28 Removals + 16 Additions

```

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 min, final double max, final double startValue) throws MaxIterationsExceededException, FunctionEvaluationException {
60         clearResult();
61         return localMin(goalType == GoalType.MINIMIZE,
62                         f, goalType, min, startValue, max,
63                         getRelativeAccuracy(), getAbsoluteAccuracy());
64     }
65     public double optimize(final UnivariateRealFunction f, final GoalType goalType, final double min, final double max) throws MaxIterationsExceededException, FunctionEvaluationException {
66         return optimize(f, goalType, min, max, min + GOLDEN_SECTION * (max - min));
67     }

```

Editor

Diffchecker Home Images PDF Contact CLI Desktop Sign in Create a free account

Recent Diffs Clear

28 Removals + 16 Additions

```

143     lv = lw;
144     w = x;
145     fw = fx;
146     x = u;
147     fu = fu;
148     } else {
149         if (u < x) {
150             a = u;
151         } else {
152             b = u;
153         }
154         if (fu <= fw
155             || w == x) {
156             v = w;
157             fv = fw;
158             w = u;
159             fw = fu;
160         } else if (fu <= fv
161             || v == x
162             || v == w) {
163             v = u;
164             fv = fu;
165         }
166     } else { // termination
167         setResult(x, (goalType == GoalType.MAXIMIZE) ? -fx : fx, count);
168         return x;
169     }
170     ++count;
171 }
172 }
173 throw new MaxIterationsExceededException(maximalIterationCount);
174 }

```

Editor

## Mockito-22:

OPEN FILES

```

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

```

```

5679      40,
5680      41,
5681      42,
5682      43,
5683      44,
5684      45,
5685      46,
5686      47,
5687      48,
5688      49,
5689      50,
5690      51,
5691      52
5692    ],
5693  },
5694  {
5695    "Project": "Mockito-22",
5696    "Class": "org.mockito.internal.matchers.Equality",
5697    "Lines": [
5698      16,
5699      13,
5700      14,
5701      15
5702    ],
5703  },
5704  {
5705    "Project": "Mockito-23",
5706    "Class": "org.mockito.internal.stubbing.defaultanswers.ReturnsDeepStubs",
5707    "Lines": [
5708      128,
5709      135,
5710      136,
5711      137,
5712      138,
5713      139,
5714      44,
5715      45,
5716      51,
5717      52,
5718      53,
5719      54,
5720    ]

```

Find Find Prev Find All  
10 lines, 202 characters selected Spaces: 4 JSON

## Spot bugs: no warning (meth 2)

OPEN FILES

```

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

```

```

337      "Project": "Mockito-22",
338      "Type": "DLS_DEAD_LOCAL_STORE",
339      "Priority": "2",
340      "Rank": "17",
341      "Msg": "Dead store to $L2 in org.mockito.internal.invocation.InvocationMatcher.captureArgumentsFrom",
342      "Method": "captureArgumentsFrom",
343      "Field": "",
344      "Lines": [
345        [
346          122,
347          122,
348          null
349        ]
350      ],
351    },
352    {
353      "Proj": "Mockito-22",
354      "Class": "",
355      "Cat": "",
356      "Abbrev": "",
357      "Type": "NO_WARNING",
358      "Priority": "",
359      "Rank": "",
360      "Msg": "",
361      "Method": "",
362      "Field": "",
363      "Lines": []
364    },
365    {
366      "Proj": "Mockito-23",
367      "Class": "org.mockito.internal.stubbing.defaultanswers.ReturnsDeepStubs",
368      "Cat": "BAD_PRACTICE",
369      "Abbrev": "Se",
370      "Type": "SE_BAD_FIELD",
371      "Priority": "1",
372      "Rank": "14",
373      "Msg": "Class org.mockito.internal.stubbing.defaultanswers.ReturnsDeepStubs defines non-transient field $L1 which is annotated with @Transient. This is a bad practice and may lead to unexpected behavior in some environments.",
374      "Method": "",
375      "Field": "mockitoCore",
376      "Lines": []
377    }

```

Find Find Prev Find All  
13 lines, 290 characters selected Spaces: 4 JSON

## Warnings:

```

file:///Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Mockito-22/src
file:///Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/f/Mockito-22/src

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/projects/b/Mockito-22/target/classes
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/asm-all-5.0.4.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/asm-core-2.1.0.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/cglib-and-asn-1.0.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/cobertura-2.0.3.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/test-assert-1.3.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/test-util-1.1.4.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/hamcrest-1.1.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/object2.1.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/object2.2.jar
/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/powerreflect-1.2.5.jar

/Users/stylethinking/Documents/StaticBugCheckers/defects4j/framework/projects/Mockito/lib/powerelect-1.2.5.jar Redundant nullcheck of value known to be non-null Redundant nullcheck of SL1, which is known non-null in org.mockito.internal.matchers.Equality.areEqual(Object, Object) A Equality.java:[lines 10-41] In class org.mockito.internal.matchers.Equality In method org.mockito.internal.matchers.Equality.areEqual(Object, Object) Value loaded from ? Redundant null check at Equality.java:[line 16] Dodgy code Redundant nullcheck of value known to be non-null <p> This method contains a redundant check of a known non-null value against the constant null.</p> Redundant comparison to null

```

## Code Diff: Domain-specific

Diffchecker Home Images PDF Contact CLI Desktop Sign in Create a free account

Recent Diffs Clear

0 Removals + 3 Additions

```

4 */
5 package org.mockito.internal.matchers;
6
7 import java.lang.reflect.Array;
8
9 //stolen from hamcrest because I didn't want to have more dependency than Matcher class
10 public class Equality {
11
12     public static boolean areEqual(Object o1, Object o2) {
13         if (o1 == null || o2 == null) {
14             return o1 == null && o2 == null;
15         } else if (isArray(o1)) {
16             return isArray(o2) && areArraysEqual(o1, o2);
17         } else {
18             return o1.equals(o2);
19         }
20     }
21
22     static boolean areArraysEqual(Object o1, Object o2) {
23         return areArrayLengthsEqual(o1, o2)
24             && areArrayElementsEqual(o1, o2);
25     }
26
27     static boolean areArrayLengthsEqual(Object o1, Object o2) {
28         return Array.getLength(o1) == Array.getLength(o2);
29     }
30

```

```

4 */
5 package org.mockito.internal.matchers;
6
7 import java.lang.reflect.Array;
8
9 //stolen from hamcrest because I didn't want to have more dependency than Matcher class
10 public class Equality {
11
12     public static boolean areEqual(Object o1, Object o2) {
13         if (o1 == o2) {
14             return true;
15         } else if (o1 == null || o2 == null) {
16             return o1 == null && o2 == null;
17         } else if (isArray(o1)) {
18             return isArray(o2) && areArraysEqual(o1, o2);
19         } else {
20             return o1.equals(o2);
21         }
22     }
23
24     static boolean areArraysEqual(Object o1, Object o2) {
25         return areArrayLengthsEqual(o1, o2)
26             && areArrayElementsEqual(o1, o2);
27     }
28
29     static boolean areArrayLengthsEqual(Object o1, Object o2) {
30         return Array.getLength(o1) == Array.getLength(o2);
31     }

```

Editor