



DEVOPS TRUENORTH PACKAGE FROM



PolSource + `{/code.scan}`TM +



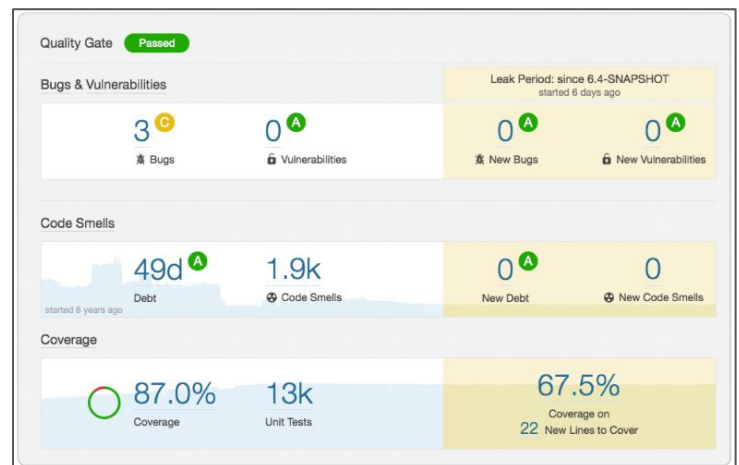
COPADO

Get the power of three leading Salesforce partners' expertise to benchmark and improve your DevOps solution

The DevOps TrueNorth package is a complete Salesforce DevOps health check to evaluate the current situation of a company's DevOps architecture and compare it to industry best practices, guided by PolSource, and powered by Copado's DevOps platform and CodeScan's Static Code Analysis Technology.

The solution helps your company:

- Reduce technical debt
- Sanction DevOps processes that enforce code gates to manage working with multi vendor org structures
- Empower developers to write high quality code
- Increase code visibility and improve code review process



REDUCE CURRENT & AVOID NEW TECHNICAL DEBT!

Why is now the time to find your TrueNorth?

As your business adjusts to the new normal, it is vital to:

- Understand your solution's current level of Technical Debt
- Avoid new technical debt - applying best practices at source that provide developers rapid feedback
- Prepare for arising new technology such as SFDC releases
- Avoid legacy code base that slows down development
- Help developers write high quality code, *fast*
- Effect a prioritised remediation plan where appropriate
- Enable secure control gates to reach and maintain expected quality levels





PolSource + {/code.scan}™



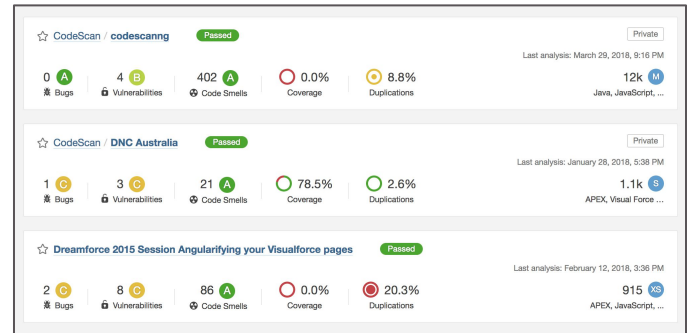
COPADO +



The DevOps TrueNorth Package provides you with three leading Salesforce partners' expertise to benchmark and improve your DevOps solution.

The offering includes:

- 30 min code assessment review with DevOps expert using CodeScan technology
- Call with a DevOps expert to build a health plan
- Plan execution using Polsource and CodeScan to:
 - Reduce technical debt
 - Gain visibility into code base
 - Enable quality code gates
 - Archive repeatable deployment
 - Improve code review process
 - Manage multi vendor org structure
- OPEX reduction to move cheaper and faster with Salesforce development
- **Peace of mind!** Enjoy Salesforce releases without the worry of system breaks



```

1 public class BadClass{
2
3     integer unusedVariable1 = 0;
4     integer unusedVariable2 = 0;
5     integer unusedVariable34 = 0;
6     integer unusedVariable66 = 0;
7     integer unusedVariable6666 = 0;
8
9     public void BadClass () {
10        integer x = 1;
11
12        string y = x > 0 ? 'good' : 'bad';
13        system.debug(x);
14    }
15 }
  
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

- BadClass.cls src/classes 19
- Classes should not have non-constructor methods with the same name as the class (sf.MethodWithSameNameAsEnclosingClass) (9, 14)
- Avoid unused local variables such as 'y'. (sf.UnusedLocalVariable) (12, 1)
- Must have a single whitespace before opening braces (sf.LeftBracesSpacingPositions) (1, 1)
- Class does not have a corresponding test class (sf.ClassWithoutTestClass) (1, 1)
- Header comments are Required (sf.CommentRequired) (1, 1)
- Field comments are Required (sf.CommentRequired) (3, 1)
- Field comments are Required (sf.CommentRequired) (4, 1)
- Field comments are Required (sf.CommentRequired) (5, 1)
- Field comments are Required (sf.CommentRequired) (6, 1)
- Field comments are Required (sf.CommentRequired) (7, 1)
- Avoid excessively long variable names like unusedVariable6666 (sf.LongVariable) (7, 13)
- Public method and constructor comments are Required (sf.CommentRequired) (9, 1)

CREATE TRANSPARENCY & TRUST

Ready to find your TrueNorth?

Contact us today for more information:

hello@polsource.com