



Automating Code Quality: A Guide to Using Custom Code Analysis for AL

Who am I?

Tine Starič

Software Developer @ Companial



Agenda

What is Code Analysis and why is it important?

Existing solutions

Creating custom linters for AL

Integration with development workflows

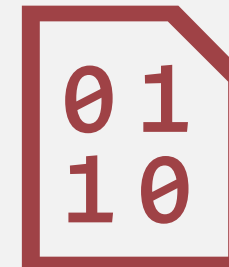
Linting use cases



What is Code Analysis and why is it important?

- A way to examine source code before the program is run
- .Net library containing rules to validate source code against
- Increase quality, find mistakes faster, standardize code, accelerate development cycle
- Subscribes to AL Compiler Events

```
codeunit 50100 MyCodeunit
{
    trigger OnRun()
    begin
        if not IsInterestingSession() then
            exit;
    end
}
```



Existing solutions

 AppSourceCop, CodeCop, UI Cop, PTECop

 BusinessCentral.LinterCop

 Custom Code Analysis



BC Linter Cop

BusinessCentral.LinterCop

Preview

Stefan Maron |  26,023 installs | ★★★★★ (2) | Free

Provides Linting for the AL Language (Business Central)

Install

[Trouble Installing?](#) 





Creating custom linters for AL

A short overview of Companial Code Cop
source and rule development

- Companial Cop Source Code
- Creating a new Rule
- Debugging a rule
- Support and maintainability disclaimer

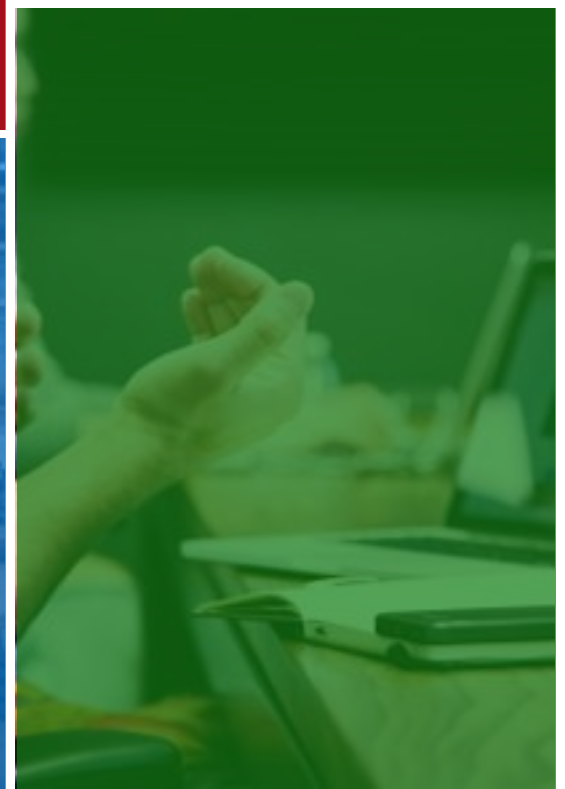
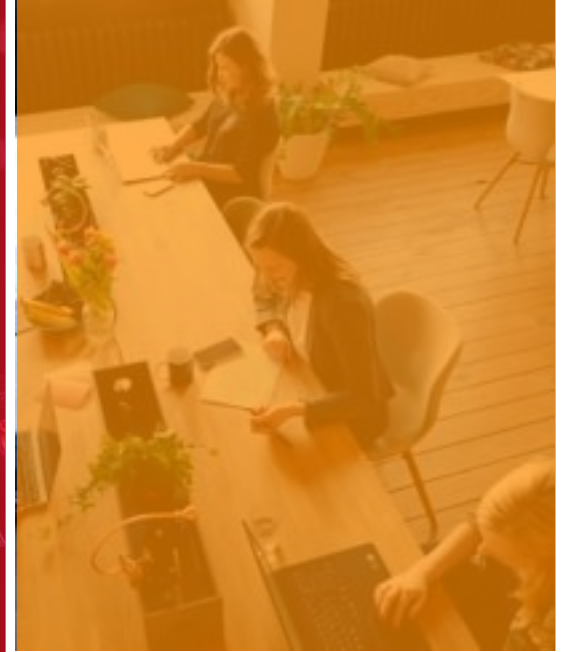


Integration with development workflows

-  VS Code
-  BcContainerHelper
-  AL-Go! for GitHub
-  ALOps

```
"settings": {  
  "al.enableCodeAnalysis": true,  
  "al.codeAnalyzers": [  
    "${CodeCop}",  
    "${AppSourceCop}",  
    "${UICop}",  
    "../BusinessCentral.CompanialCodeCop.dll"  
  ],  
  "al.ruleSetPath": "../RuleSets/Companial.ruleset.json",  
}
```

```
Compile-AppInBCContainer -  
  -containerName $containerName -  
  -credential $credential -  
  -appProjectFolder $buildProjectFolder -  
  -appOutputFolder $buildOutputFolder -  
  -AzureDevOps -  
  -CustomCodeCops $BCCodeAnalyzerDllPath
```



Linting use cases

- ▣ Examples of rules:
- ▣ Run-time errors
- ▣ Maintainability improvements
- ▣ Extensibility improvements
- ▣ Readability improvements
- ▣ Coding Conventions
- ▣ Nice-to-haves



Linting use cases – Run-time errors

```
local procedure DoSomethingWithEntries  
var  
    CustomerLedgerEntry: Record "Cust. Ledger Entry";  
begin  
    CustomerLedgerEntry.SetRange("Customer No.", CustomerNo);  
    CustomerLedgerEntry.SetAutoCalcFields(Amount, "Amount (LCY)", "Sales (LCY)");  
    if CustomerLedgerEntry.FindSet() then  
        repeat  
            //Do Something with amounts  
        until CustomerLedgerEntry.Next() = 0;  
    end;  
end;
```

1 reference | 0% Coverage

```
local procedure DoSomethingWithEntries(CustomerNo: Code[20])  
var  
    CustomerLedgerEntry: Record "Cust. Ledger Entry";  
begin  
    CustomerLedgerEntry.SetRange("Customer No.", CustomerNo);  
    CustomerLedgerEntry.SetAutoCalcFields(Amount, "Amount (LCY)", "Sales (LCY)");  
    if CustomerLedgerEntry.FindSet() then  
        repeat  
            //Do Something with amounts  
        until CustomerLedgerEntry.Next() = 0;  
    end;  
end;
```



The Sales (LCY) field is not a FlowField.

How to report this issue >

(field) "Sales (LCY)": Decimal

The CalcFields method should only be used with FlowFields or fields of type Blob. The field Sales (LCY) is not a FlowField or of type Blob. AL(TS0018)

View Problem (Alt+F8) Quick Fix... (Ctrl+.)

Amount "Amount (LCY)" "Sales (LCY)".



Linting use cases – R

```
tableextension 50100 CustomerExt extends Customer
{
  fields
  {
    0 references
    field(50100; IsBroker; Boolean)
    {
      Caption = 'Is Broker';
      DataClassification = CustomerContent;
    }
    0 references
    field(50101; Occupation; Text[250])
    {
      Caption = 'Occupation';
      DataClassification = CustomerContent;
    }
    0 references
    field(50102; LegislationOffice; Text[50])
    {
      Caption = 'Legislation Office';
      DataClassification = CustomerContent;
    }
    0 references
    field(50103; StrictConfidentiality; Boolean)
    {
      Caption = 'Strict Confidentiality';
      DataClassification = CustomerContent;
    }
  }
}
```

```
tableextension 50101 ContactExt extends Contact
{
  fields
  {
    0 references
    field(50100; HasValidCertification; Boolean)
    {
      Caption = 'Has Valid Certification';
      DataClassification = CustomerContent;
    }
    0 references
    field(50101; IsBroker; Boolean)
    {
      Caption = 'Is Broker';
      DataClassification = CustomerContent;
    }
    0 references
    field(50102; Occupation; Text[250])
    {
      Caption = 'Occupation';
      DataClassification = CustomerContent;
    }
    0 references
    field(50103; LegislationOffice; Text[50])
    {
      Caption = 'Legislation Office';
      DataClassification = CustomerContent;
    }
    0 references
    field(50104; StrictConfidentiality; Boolean)
    {
      Caption = 'Strict Confidentiality';
      DataClassification = CustomerContent;
    }
  }
}
```

Linting use cases – Run-time errors

```
table 50101 Contract
```

```
{
    Caption = 'Contract';
    Access = Internal;
    Extensible = false;

    fields
    {
        0 references
        field(1; Number; Integer)
        {
            Caption = 'Number';
        }
        0 references
        field(20; Description; Text[50])
        {
            Caption = 'Description';
        }
        0 references
        field(30; ContractDate; Date)
        {
            Caption = 'Contract Date';
        }
        0 references
        field(40; ExpirationDate; Date)
        {
            Caption = 'Expiration Date';
        }
    }
}
```

transferTo

ord Contr
ve: Recor

ve.Transf

```
table 50102 ContractArchive
```

```
{
    Caption = 'Contracts Archive';
    Access = Internal;
    Extensible = false;

    fields
    {
        0 references
        field(1; Number; Integer)
        {
            Caption = 'Number';
        }
        0 references
        field(20; Description; Text[50])
        {
            Caption = 'Description';
        }
        0 references
        field(30; ExpirationDate; Date)
        {
            Caption = 'Expiration Date';
        }
        0 references
        field(40; ContractDate; Date)
        {
            Caption = 'Contract Date';
        }
    }
}
```



Linting use cases – Maintainability improvements

1 reference | 0% Coverage

```
local procedure ImportVendorBankAccounts()  
begin  
    ImportBankAccountsFromExcel();  
  
    Commit();  
  
    AssignPrimaryBankAccount();  
end;
```

1 reference | 0% Coverage

```
local procedure CreateNewCustomer(Number: Code[20]; Name: Text[100])  
var  
    Customer: Record Customer;  
begin  
    Customer.Init();  
    Customer."No." := Number;  
    Customer.Name := Name;  
    Customer.Insert();  
end;
```

```
procedure Commit()
```

Ends the current write transaction.

Commit() needs a comment to justify its existence. Either a leading or a trailing comment. AL(TS0012)

[View Problem \(Alt+F8\)](#) [Quick Fix... \(Ctrl+.\)](#)

```
procedure Insert(): Boolean
```

Inserts a record into a table without executing the code in the OnInsert trigger.

Internal Methods must be invoked with explicit parameters AL(TS0013)

[View Problem \(Alt+F8\)](#) [Quick Fix... \(Ctrl+.\)](#)



Linting use cases – Extensibility improvements

```
1 reference | 0% Coverage  
procedure SetHideMessage(NewHideMessage: Boolean)  
begin  
    HideMessage := NewHideMessage;  
end;
```

```
#pragma warning disable CM0012 // PTE requirement  
1 reference | 0% Coverage  
    procedure SetHideMessage(NewHideMessage: Boolean)  
#pragma warning restore CM0012  
begin  
    HideMessage := NewHideMessage;  
end;
```

Procedure must be either local or internal. AL(CM0012)

View Problem (Alt+F8) Quick Fix... (Ctrl+.)

```
codeunit 50100 MyCodeunit  
{  
    ...  
}
```

Codeunit MyCodeunit

Objects need to have the Access property set to Internal AL(TS0015)

View Problem (Alt+F8) Quick Fix... (Ctrl+.)



Linting use cases – Readability improvements

```
0 references | 0% Coverage
internal procedure OpenRelatedEntries();
begin
    // ...
end;

...

var
    myInt: Integer;

trigger OnInsert()
begin
    // ...
end;

1 reference | 0% Coverage
local procedure ValidateCustomerNo()
begin
    // ...
end;
```

```
internal procedure CalcAmounts()
begin
    // ...
end;

internal procedure OpenRelatedEntries()
begin
    // ...
end;

var
    myInt: Integer;

trigger OnInsert()
begin
    // ...
end;

local procedure ValidateCustomerNo()
begin
    // ...
end;

trigger OnModify()
begin
    // ...
end;
```

The ordering should be like this: Global Variables -> Triggers -> Methods AL(TS0009)

View Problem (Alt+F8) Quick Fix... (Ctrl+.)



Linting use cases – Coding Conventions

You, 54 seconds ago | 1 author (You)

var

2 references

HideMessage: Boolean;

(global) HideMessage: Boolean

Global variables should be prefixed with "g" AL(TS0002)

[View Problem \(Alt+F8\)](#) [Quick Fix... \(Ctrl+.\)](#)

```
if SkipCheck then  
    exit;
```

The exit keyword ends the procedure with the specified exit value.

EXIT keyword must be Capitalized AL(TS0005)

[View Problem \(Alt+F8\)](#) [Quick Fix... \(Ctrl+.\)](#)



Linting use cases – Nice-to-haves

0 references | 0% Coverage

```
internal procedure CalcAmounts()  
begin  
    // ...  
end;
```

0 references | 0% Coverage

```
internal procedure OpenRelatedEntries();  
begin  
    // ...  
end;
```

```
value(0; " ")  
{  
    Caption = ' ';  
}
```

Empty captions should be locked. AL(TS0017)

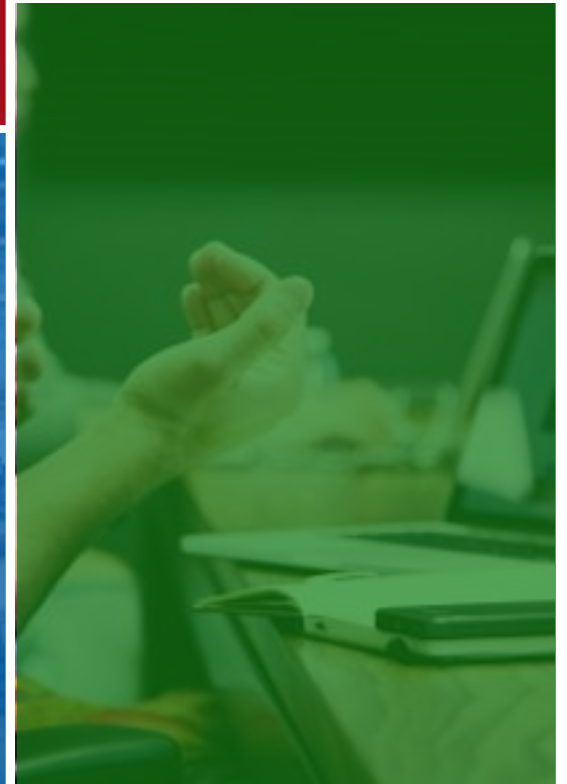
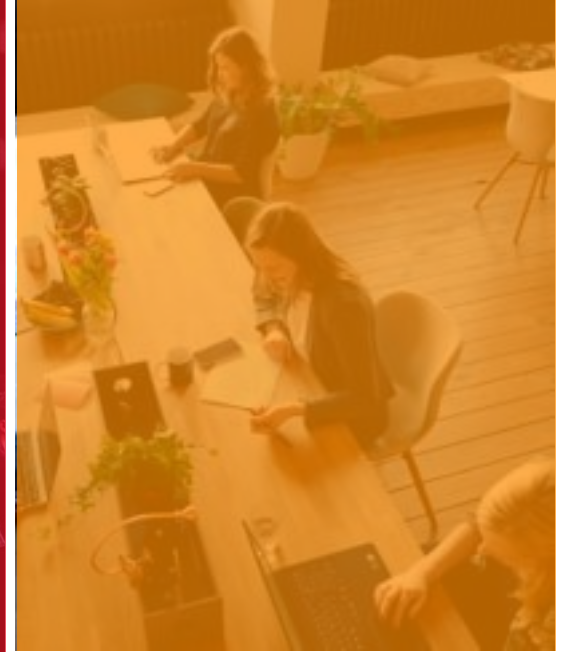
Procedure must not end with ";" AL(TS0006)



Conclusion

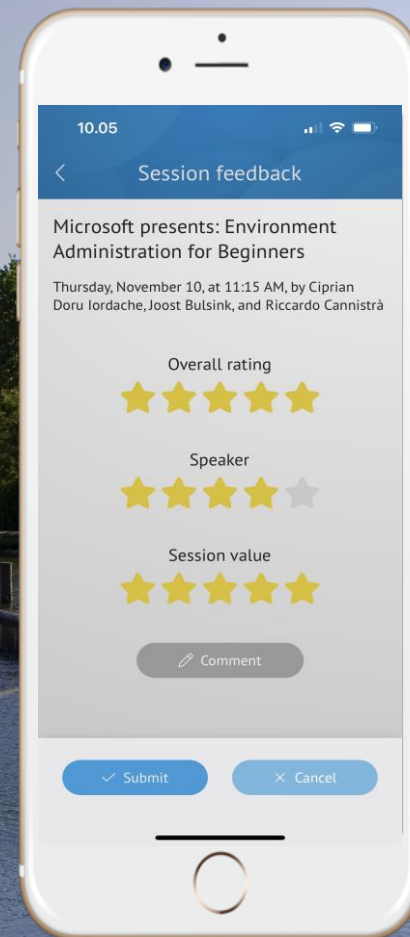
Most important insights from Automating Code Quality:

- Turn on MS Analyzers, explore the community driven initiatives
- Find and Define your Coding Conventions. And then Automate them
- Contribute



Thank you !

Please rate the session in the Conference App !



10.05

< Session feedback

Microsoft presents: Environment Administration for Beginners

Thursday, November 10, at 11:15 AM, by Ciprian Doru Iordache, Joost Bulsink, and Riccardo Cannistrà

Overall rating


★★★★★

Speaker

★★★★☆

Session value

★★★★★

 Comment

✓ Submit ✕ Cancel