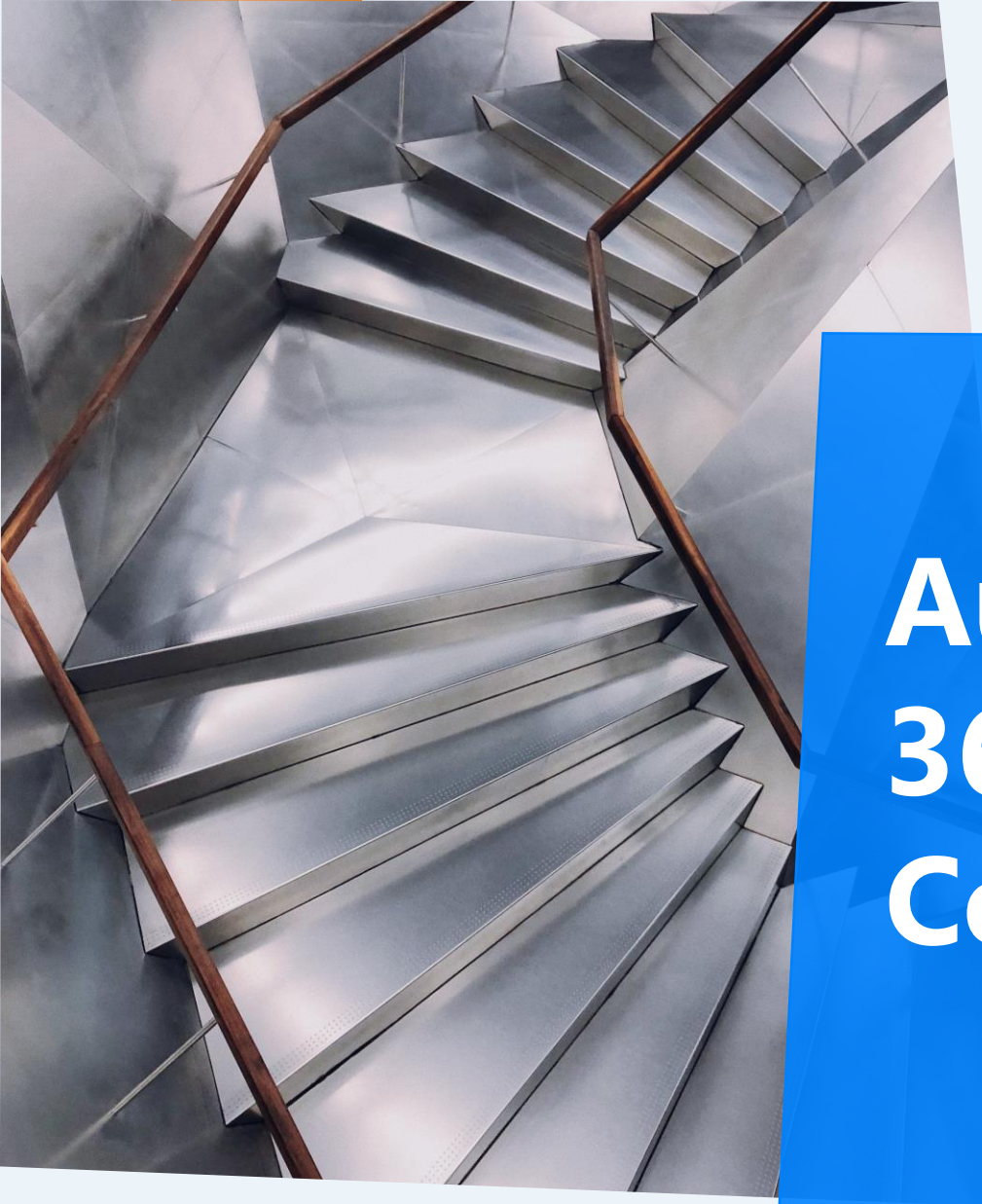


companial

Automating Dynamics 365 Business Central Code Quality

companial

Use live chat to engage with panelists and other participants!

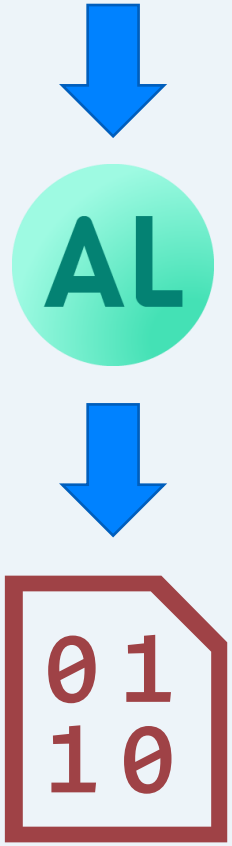
Tine Starič
D365 Business Central Developer | Companial

Agenda

1. What is Code Analysis and why is it important?
2. Existing solutions
3. Creating custom linters for AL
4. Integration with development workflows
5. Linting use cases

1. What is Code Analysis and why is it important?

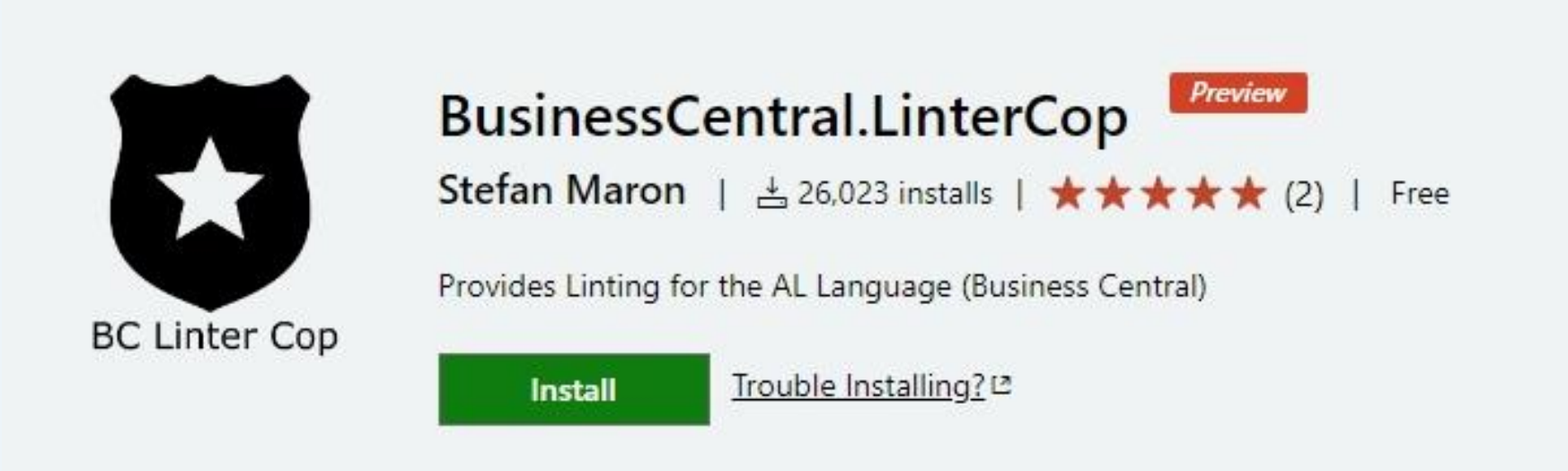
```
codeunit 50100 MyCodeunit
{
    trigger OnRun()
    begin
        if not IsInterestingSession() then
            exit;
```



- 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

2. Existing solutions

- AppSourceCop, CodeCop, UI Cop, PTECop
- BusinessCentral.LinterCop
- Custom Code Analysis



3. 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

4. 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
```

5. Linting use cases

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

5. Linting use cases – Run-time errors

```
local procedure D
var
    CustomerLedge
begin
    CustomerLedge
    CustomerLedge
```

The Sales (LCY) fi
FlowField.

How to report this issue >

```
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;
```

```
(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+.)
```

5. 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 existance. 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+.)

companial

5. 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;
```

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

```
Procedure must be either local or internal. AL(CM0012)
View Problem (Alt+F8)    Quick Fix... (Ctrl+.)
```

```
Codeunit MyCodeunit
Objects need to have the Access property set to Internal AL(TS0015)
View Problem (Alt+F8)    Quick Fix... (Ctrl+.)
```

5. Linting use cases – Readability improvements

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

...
var
    0 references
    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+.)
```

5. 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+.)
```

companial

5. 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

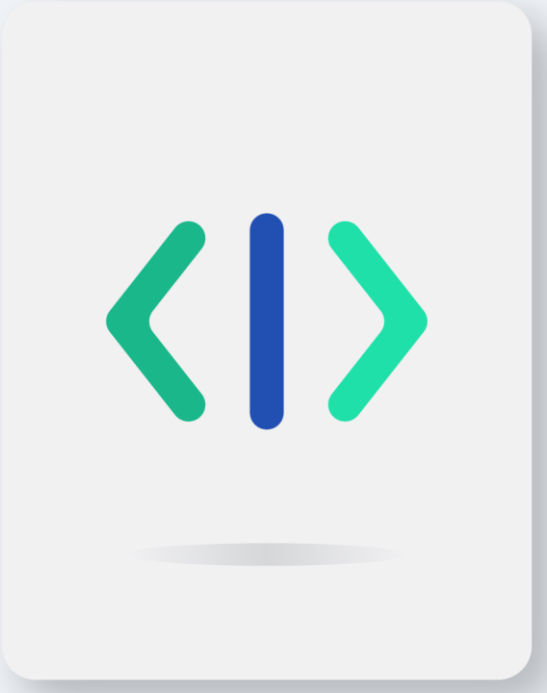
Most important insights from Automating Code Quality:

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

Companial development services for Business Central

We cover or extend partner's capacity.

- We develop customizations, Extensions, Add-ons, integrations, automated tests, data migration tools, etc.
- Companial is Microsoft appointed ISV Development Centre and has **13 years of experience** in delivering development services (over **25000+ development hours a year**).
- We provide early feedback, ensure reduced complexity, transparancy and risk management.



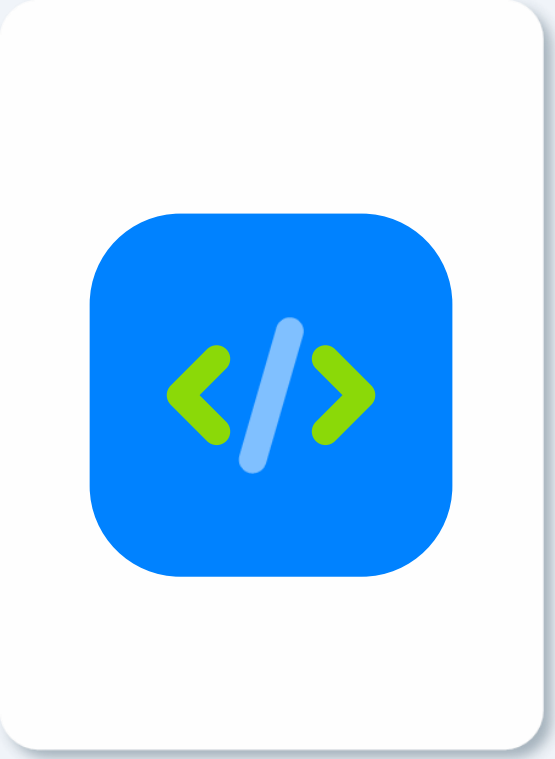
Get in touch to find out more!

www.companial.com

Companial development services for Business Central

We cover or extend partner's capacity.

- We develop customizations, Extensions, Add-ons, integrations, automated tests, data migration tools, etc.
- Companial is Microsoft appointed ISV Development Centre and has **13 years of experience** in delivering development services (over **25000+ development hours a year**).
- We provide early feedback, ensure reduced complexity, transparancy and risk management.



Get in touch to find out more!

www.companial.com