|  |
| --- |
| Gosu CodeNarc++ Static Analysis Tool  Custom Rule Implementation Readme Specification |

**AUTHORS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Department** | **Date** |
| Joseph Tino | Developer | GW Practice | 26/11/2018 |
| Arpita K C | Developer | GW Practice | 26/11/2018 |
| Lavanya D Honnappa | Developer | GW Practice | 26/11/2018 |
| Vipul Kumar | Developer | GW Practice | 26/11/2018 |
| Aravind Ramachandran | Developer | GW Practice | 26/11/2018 |

**REVIEWER**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Department** | **Date** |
| Parikshith Keshavamurthy | Lead | GW Practice | 30/11/2018 |

**Table of Contents**

[1. Introduction 4](#_Toc532404063)

[2. Purpose of the document 4](#_Toc532404064)

[3. Custom Rule Specifications and Usage Description 4](#_Toc532404065)

[3.1 Rule 001 - Validate Class Name Suffix 4](#_Toc532404066)

[3.2 Rule 002 - Add a Suffix to New Display Keys to Avoid Name Conflicts 5](#_Toc532404067)

[3.3 Rule 003 - Validate if all destination ID in messaging-config.xml has at least one event registered. 6](#_Toc532404068)

[3.4 Rule 004 – Validate if Destination ID is used as a pre-condition for executing Event Fired rule. 6](#_Toc532404069)

[3.5 Rule 005 - Validate plural variable names 6](#_Toc532404070)

[4. References 8](#_Toc532404071)

[5. Open Issues 8](#_Toc532404072)

[6. Appendix 8](#_Toc532404073)

# Introduction

The Gosu static analysis tool provides a number of out of the box rules which help developers to be able to perform static analysis checks on Gosu code. However, sometimes there is a need to add custom rules as per project needs, and to ensure best programming standards are being followed, and to ensure there are no code flaws that may potentially impact the performance of the application. The following custom rules were developed by extending the open source project CodeNarc to be able to interpret and analyze Gosu codes (.gs, .gsx...etc.).

# Purpose of the document

The purpose of this document is to provide brief explanation on how to setup the newly written gosu custom rule which require additional configuration.

# Custom Rule Specifications and Usage Description

## Rule 001 - Validate Class Name Suffix

This rule will look for suffixes added to gosu classes (gs and gsx files), in the configured packages and sub packages. This rule by default is not enabled, unless user configures the properties in codenarc- gosu-classname-rules.properties. Once the properties are configured and if the filename is missing the suffix then the rule will flag the violation.

**Priority of the Violation:** 2

**Steps to configure property file:**

Step 1: Refer the ‘codenarc- gosu-classname-rules.properties’ file inside the ‘./tools/gosu-codenarc/lib/ directory.

Step 2: Configure ‘codenarc- gosu-classname-rules.properties’ file based on the below instructions:

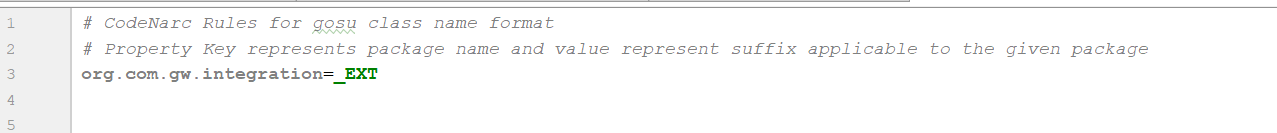
1. Configure the ‘package name’ as key on which the rule needs to be executed.

Note: By default program will take care of all sub packages configured in the property file.

1. Configure the ‘Suffix’ as Value against the above package name key.

For example, if project XYZ enforces all newly created classes to have a specific suffix (\_EXT) then the rule will look for \_EXT in the gosu classes (gs and gsx files), coming under the configured package key.

Note: Suffix can be a single entry or it can be comma separated string without any whitespaces. Also there are no restriction on suffix length, rule can support any String as in below format.



## Rule 002 - Add a Suffix to New Display Keys to Avoid Name Conflicts

This rule will look for suffixes added to the new display keys which are project specific. If the newly added display keys does not contain the configured project suffix then this rule will flag a violation on the newly created display keys.

**Priority of the Violation:** 3

**Steps to configure property file:**

Step 1: Navigate to the folder named ‘displaykey’ under lib path (‘./tools/gosu-codenarc/lib/).

There are sub folders as below for each Guidewire product under displaykey folder:

* policycenter (./tools/gosu-codenarc/lib/displaykey/policycenter)
* billingcenter (./tools/gosu-codenarc/lib/displaykey/billingcenter)
* claimcenter (./tools/gosu-codenarc/lib/displaykey/claimcenter)

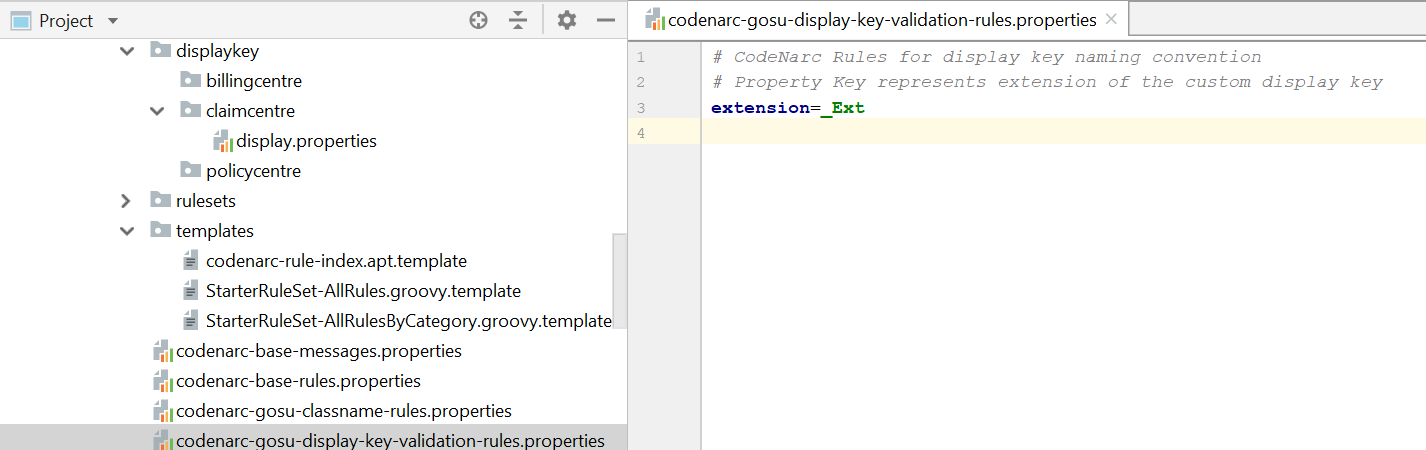
Step 2: Place the OOTB display.properties file inside the appropriate folder referenced in the step 1.

For example, if the codenarc is running against PolicyCenter then place the OOTB display.properties corresponding to Policy Center under ‘policycenter’ folder.

Step 3: Refer the ‘codenarc-gosu-display-key-validation-rules.properties’ under lib folder.

Add a key as ‘extension’ and ‘suffix’ as value specific to your project.

For example, please refer the screenshot:



## Rule 003 - Validate if all destination ID in messaging-config.xml has at least one event registered.

This rule will look for all the destination IDs in the messaging-confg.xml file and check if at least one event is registered for the destination. This rule by default is not active or enforced, unless user configures it in ‘codenarc-project-basics.properties’. If the rule is enforced and any destination ID in messaging-config.xml does not have any event(s) registered, the rule will flag a violation.

**Priority of the Violation:** 1

## Rule 004 – Validate if Destination ID is used as a pre-condition for executing Event Fired rule.

This rule will look for the Event Fired Rules and checks if the parent event fired rule is filtered using Destination ID. If the destination ID is not used as a pre-condition to execute the event fired rule, it will flag a violation.

**Priority of the Violation:** 1

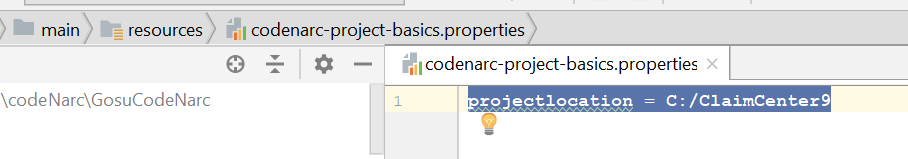
**Steps to configure property file for Rule 003 and 004:**

Step 1: Refer the ‘codenarc-project-basics.properties’ file inside the ‘./tools/gosu-codenarc/lib/ directory.

Step 2: Configure the codenarc-project-basics.properties file by providing ‘projectlocation’ as property key and mention ‘project folder’ path as value.

Note: The project folder path must be till modules folder as highlighted in the below example.

Eg: **C:/ClaimCenter9**/modules/configuration/config/messaging



## Rule 005 - Validate plural variable names

This rule will look for variable name declarations and validates if the variable name is in plural and it matches with the data type.

If the variable name is plural and the data type of the variable does not belong to type array, List and other Collection types, it will flag a violation.

Eg: var claim: Claim - > Will not flag a violation.

var claims: Claim - > Will flag a violation

Note: The rule looks for variable names with ‘s’ or ‘es’ appended to the variable names. Additional plural names can be configured in the property file as given below.

**Priority of the Violation:** 3

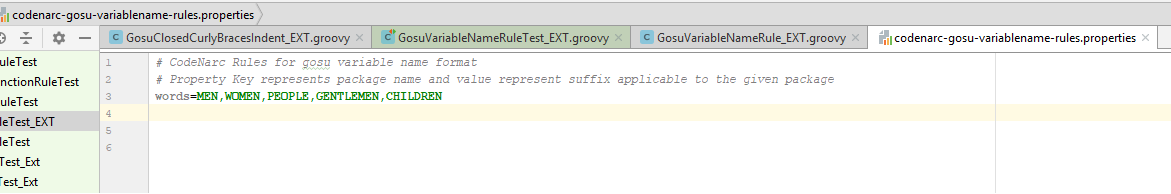
**Steps to configure property file:**

Step 1: Refer the ‘codenarc-gosu-variablename-rules.properties’ file inside the ‘./tools/gosu-codenarc/lib/ directory.

Step 2: Configure the ‘codenarc-gosu-variablename-rules.properties’ file by providing ‘words’ as property key and mention the plural words as value which needs to be explicitly included while validating the variable names.

**Note: This property file configuration is optional as the default configuration already handles most of the plural keywords. DO NOT DELETE the existing keywords. Additional project specific keywords can only be added if required.**

Eg: Men, Women, Children.



# References

The sample property files which are referenced in this document are attached along with the tool package.

# Open Issues

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Issue ID** | **Issue** | **Raised By** | **Raised On** | **Solution/ Decision** | **Resolved By** | **Resolved On** | **Status** |
|  |  |  |  |  |  |  |  |

# Appendix