# **[Code review good practices](https://wiki.eisgroup.com/display/GlicAbsence/Code+review+good+practices)**

Code review helps us maintain the quality of the code and is essential to prevent defects and mistakes that we may have missed.

It leads to improvement of code readability, more expertise, and knowledge transfer.

This list may suggest what to keep in mind and what to pay attention to when making a review code.

General

1. The pipeline of build and ETCS pass.
2. All requirements from the bug/user story were implemented.
3. Any new parameters and migration jobs that need to be triggered manually are present in the release note.
4. Merge requests need 1-2 approvals to be considered as reviewed. When 1 of the approvals is provided by experienced devs

Kraken

1. New rules are covered by unit test (exception is a rule that includes context.external attribute)
2. New rules are attached to appropriate entry point
3. New rules have proper descriptions and error messages correspond to the logic of the rule.
4. Rule naming convention maintained.
5. If a rule has an error message such message should be also present in the properties file
6. If lookups are used, the values should appear as codes, not translations:  
   Graphical user interface, application

   Description automatically generated
7. If the rule was edited it should be also reflected in unit tests

DSL

1. The new attribute should be annotated with @Description with the business name
2. The new attribute should be covered by a special entity unit test (
3. If an attribute will be used in Openl as a condition should be annotated with @ExternalValidation and @Dimension
4. If the attribute was removed/excluded it was removed from all kraken rules
5. The attribute should follow the naming convention [DSL Files Naming](https://wiki.eisgroup.com/display/GRC/DSL+Files+Naming)
6. When it's requested to rename an attribute, we have 2 options other than renaming key to avoid serious issues during base upgrade
   1. Renaming description and name in OpenL but attribute key is kept same
   2. Marking attribute as @Exclude (maybe also @Depreicated) and creating a new one
7. In occasions as in dental, using error messages in rule and in messages.properties that have curly braces e.g. (situs state has to be in {"GA", "TX"}) can result in error (e.g. "Failed to convert parameters, message: can't parse argument number: 'GA'" )
8. The error "Cycle detected between rules" could happen even if no cycle is present if the rule targets are inherited from same parent

Unit Tests

1. Make sure that test cases have been written and cover a sufficient percentage of new code.
2. For BC patches and Migration job important is to test if the applied patch "does the job" so for example test the "apply" method.
3. Assertions are set up correctly

Java

1. Follow [RxJava Convention](https://wiki.eisgroup.com/display/GRC/RxJava+Convention)
2. Follow [Saga Usage Conventions](https://wiki.eisgroup.com/display/GRC/Saga+Usage+Conventions)
3. Follow [Spring Convention](https://wiki.eisgroup.com/display/GRC/Spring+Convention)
4. Follow [Modularity Convention](https://wiki.eisgroup.com/display/GRC/Modularity+Convention)
5. On top of each class, a file should be presented copyright:

|  |
| --- |
| /\* Copyright © 2021 EIS Group and/or one of its affiliates. All rights reserved. Unpublished work under U.S. copyright laws.   CONFIDENTIAL AND TRADE SECRET INFORMATION. No portion of this work may be copied, distributed, modified, or incorporated into any other media without EIS Group prior written consent.\*/ |

To setup it automatically we can:  
Intellij IDEA should be configured in the following way *(Settings / Editor / File and Code Templates / Class)*:

|  |
| --- |
| /\* Copyright © ${YEAR} EIS Software Ltd. and/or one of its affiliates. All rights reserved. Unpublished work under U.S. copyright laws.  CONFIDENTIAL AND TRADE SECRET INFORMATION. No portion of this work may be copied, distributed, modified, or incorporated into any other media without prior written consent from an authorized EIS representative. \*/  #**if** (${PACKAGE\_NAME} && ${PACKAGE\_NAME} != "")**package** ${PACKAGE\_NAME};#end  #parse("File Header.java")  /\*\*   \* TODO description   \*   \* @author ${USER}   \* @since ${SPRINT}   \*/  **public** **class** ${NAME} {  } |

1. Java Docs: On top of each class file should be present @author and @since tags so that it was clear who initially created this file and since which version.
   1. Version is a sprint number.
2. In BC patches class should be annotated with @DataMigration and it is good to check if all possible nulls have been handled.
3. In Migration Job, class should be annotated with @MigrationJob
4. The class is free of any Sonar Issues and was analysed by IntelliJ tools.
5. Any overridden base beans should be annotated with @Customization