A *GitHub Action* to check your code with *diKTat*

Table of Contents

License MIT

[GitHub release] | *https://badgen.net/github/release/saveourtool/benedikt/latest?color=green*

[Ubuntu Linux] | <em>https://badgen.net/badge/icon/Ubuntu?icon=terminal&label&color=green</em>

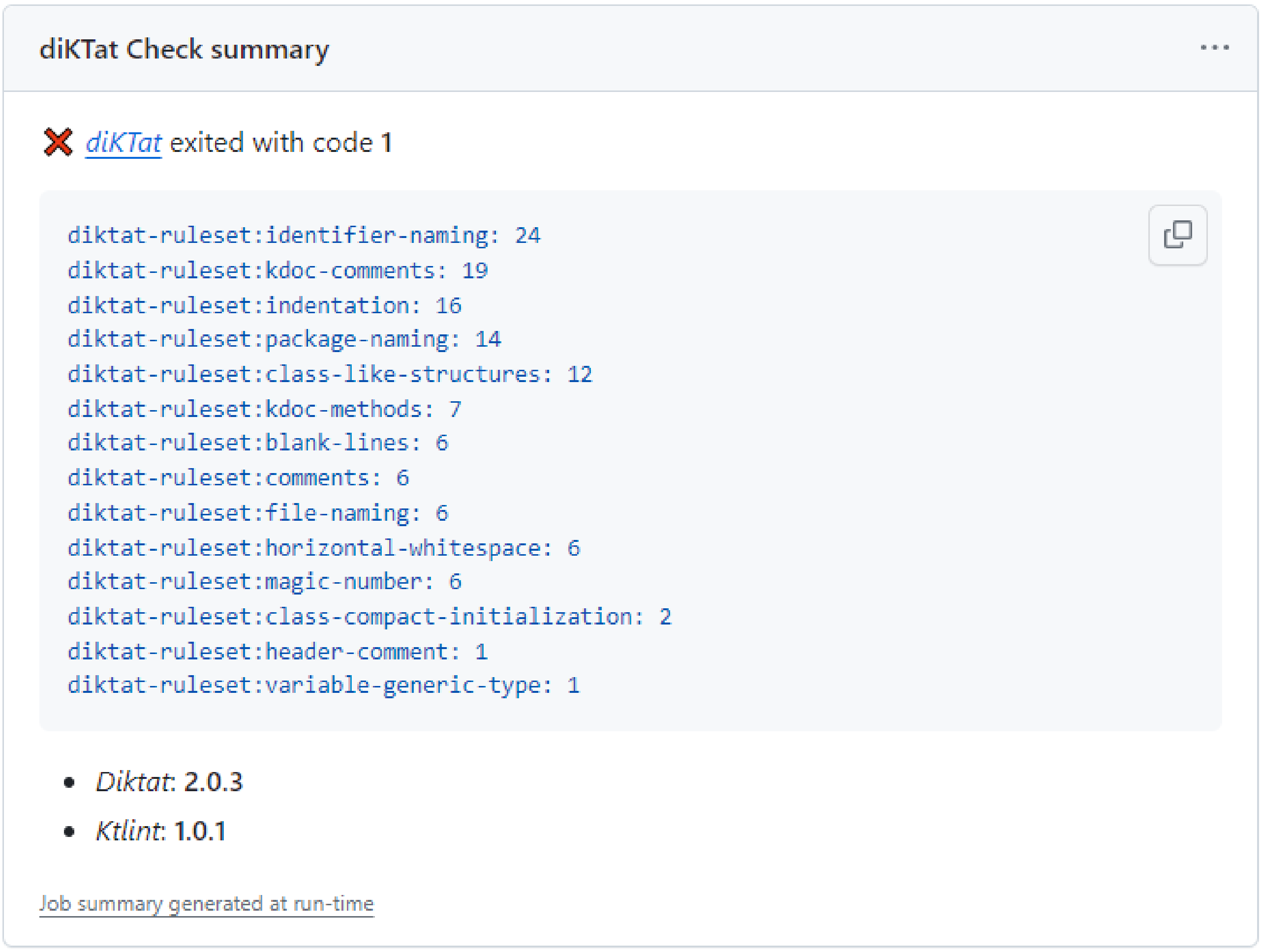
[macOS] | <em>https://badgen.net/badge/icon/macOS?icon=apple&label&color=green</em>

[Windows] | <em>https://badgen.net/badge/icon/Windows?icon=windows&label&color=green</em>

> An always updated version of this document is available here as a PDF e-book.

Features

- Customizable `diktat-analysis.yml` location. You can use the rule set configuration with an alternate name or at a non-default location.
- Customizable JVM vendor and version. You can run *diKTat* using a default JVM, or you can set up your own one.
- Customizable reporter (*SARIF* or *Checkstyle* XML).
- Allows multiple input paths. If you have a multi-module project and only wish to check certain directories or modules, you can configure the action accordingly.
- The summary page contains statistic about detected errors:


diKTat Check summary

❌ *diKTat* exited with code 1

```
diktat-ruleset:identifier-naming: 24
diktat-ruleset:kdoc-comments: 19
diktat-ruleset:indentation: 16
diktat-ruleset:package-naming: 14
diktat-ruleset:class-like-structures: 12
diktat-ruleset:kdoc-methods: 7
diktat-ruleset:blank-lines: 6
diktat-ruleset:comments: 6
diktat-ruleset:file-naming: 6
diktat-ruleset:horizontal-whitespace: 6
diktat-ruleset:magic-number: 6
diktat-ruleset:class-compact-initialization: 2
diktat-ruleset:header-comment: 1
diktat-ruleset:variable-generic-type: 1
```

- *Diktat*: 2.0.3
- *Ktlint*: 1.0.1

Job summary generated at run-time

Usage

In the simplest scenario, the action can be used without input parameters; you just need to check out your code first, using `actions/checkout`:

```
jobs:
  diktat_check:
    name: 'diKTat Check'
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4

      - uses: saveourtool/benedikt@v2
```

Configuration

`config`: custom configuration file

- Default: `diktat-analysis.yml`
- Required: **no**

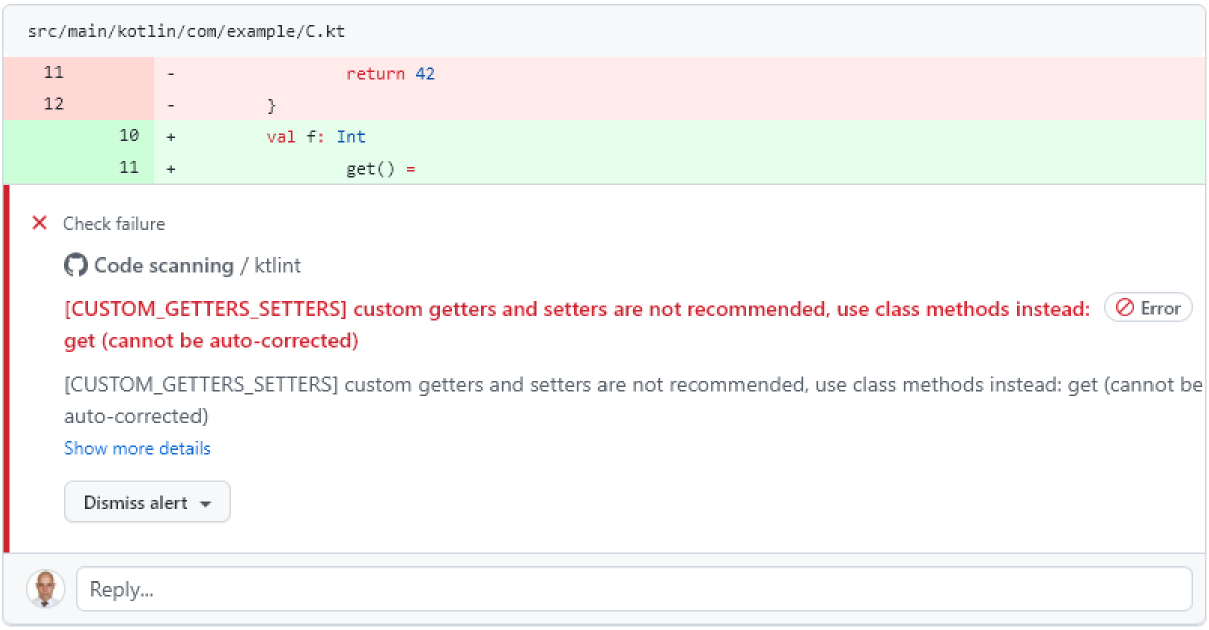
You can override the name or the path of your YAML configuration file using the `config` input parameter, e. g.:

```
    - uses: saveourtool/benedikt@v2
      with:
        config: path/to/diktat-analysis-custom.yml
```

`reporter`: requesting a type of reporter

If you wish, you can report errors in a different format.

- Default: `sarif`
- Required: **no**
- Possible values: any of the following.
    - `sarif`: report errors in the *SARIF* format. The output file will be named `report.sarif` and automatically uploaded to *GitHub* using the `upload-sarif` action. This will enable the check results to be shown as annotations in the pull request:

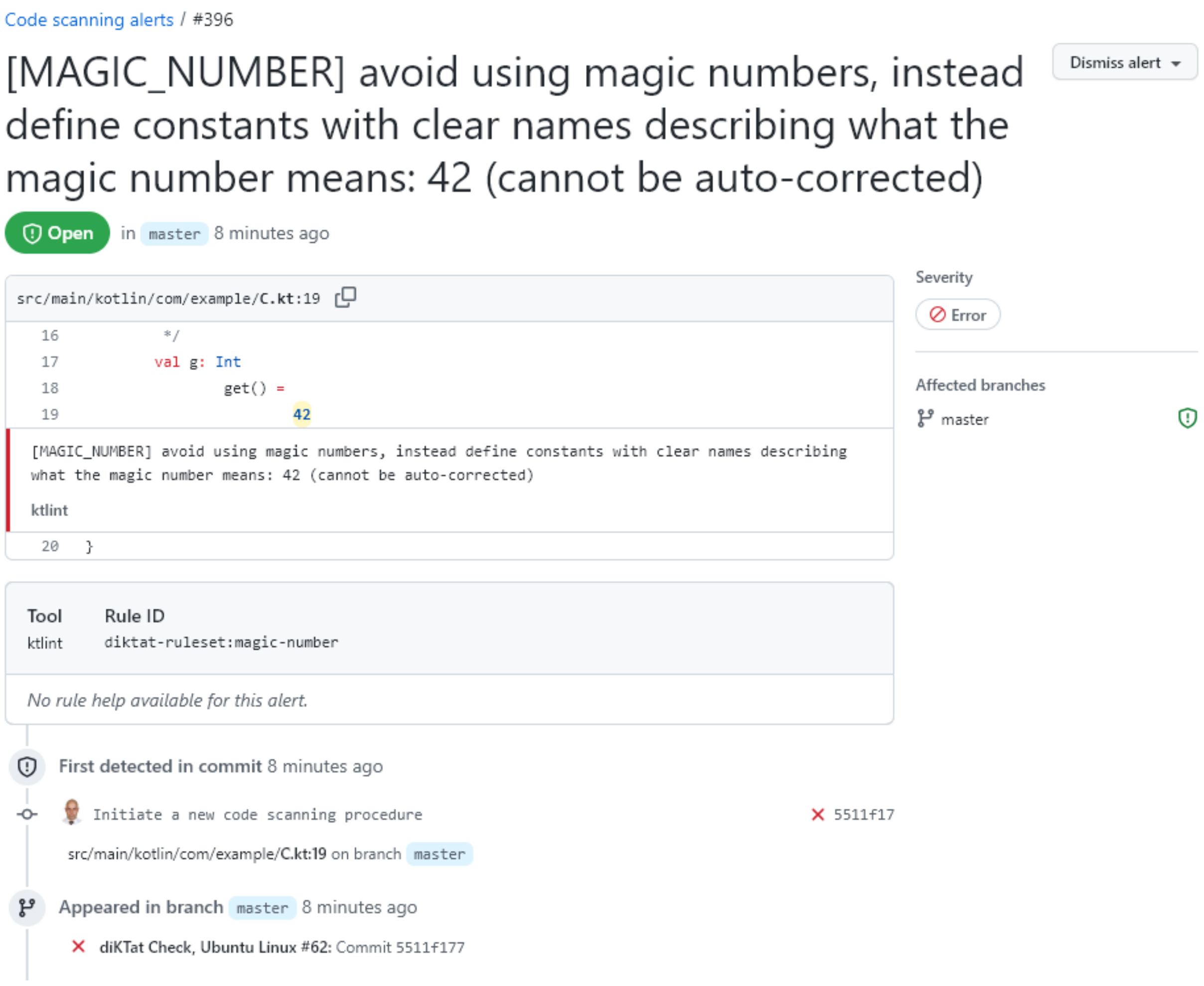


```
src/main/kotlin/com/example/C.kt
11        -                return 42
12        -           }
     10   +           val f: Int
     11   +                get() =
```

Check failure
Code scanning / ktlint
[CUSTOM_GETTERS_SETTERS] custom getters and setters are not recommended, use class methods instead: get (cannot be auto-corrected)

[CUSTOM_GETTERS_SETTERS] custom getters and setters are not recommended, use class methods instead: get (cannot be auto-corrected)

as well as in the *Code scanning alerts* section of your repository:


Code scanning alerts / #396

[MAGIC_NUMBER] avoid using magic numbers, instead define constants with clear names describing what the magic number means: 42 (cannot be auto-corrected)

Open in master 8 minutes ago

Dismiss alert

src/main/kotlin/com/example/C.kt:19

```
        */
        val g: Int
            get() =
                42
```

[MAGIC_NUMBER] avoid using magic numbers, instead define constants with clear names describing what the magic number means: 42 (cannot be auto-corrected)

ktlint

```
}
```

Severity: Error

Affected branches: master

Tool	Rule ID
ktlint	diktat-ruleset:magic-number

No rule help available for this alert.

First detected in commit 8 minutes ago

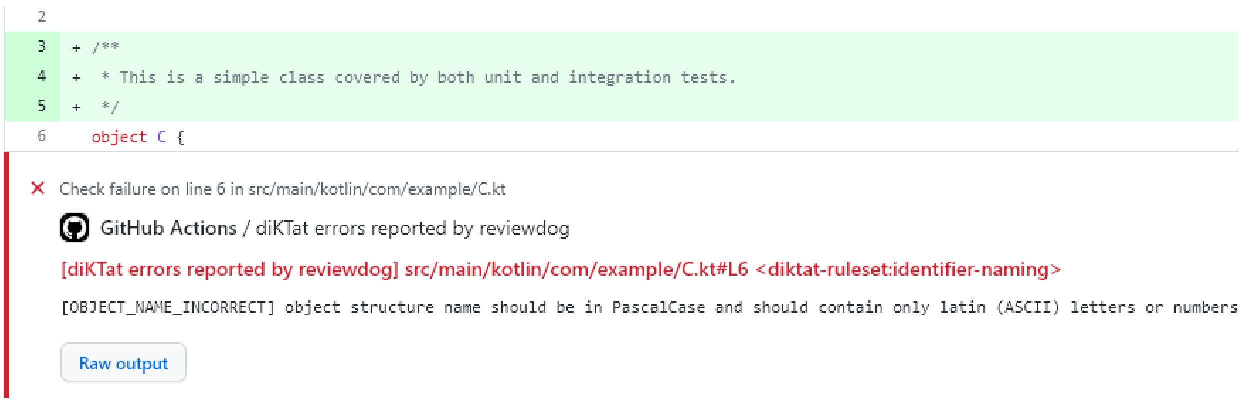
Initiate a new code scanning procedure — 5511f17

src/main/kotlin/com/example/C.kt:19 on branch master

Appeared in branch master 8 minutes ago

diKTat Check, Ubuntu Linux #62: Commit 5511f177

- `checkstyle`: this is the reporter of choice if you ever encounter issues with the `sarif` reporter. Errors are reported in the *Checkstyle-XML* format to the file named `checkstyle-report.xml`. The report is then consumed by the `reviewdog` tool and uploaded to *GitHub*, resulting in code annotations similar to those produced by the `sarif` reporter:



```
+ /**
+  * This is a simple class covered by both unit and integration tests.
+  */
  object C {
```

Check failure on line 6 in src/main/kotlin/com/example/C.kt

GitHub Actions / diKTat errors reported by reviewdog

[diKTat errors reported by reviewdog] src/main/kotlin/com/example/C.kt#L6 <diktat-ruleset:identifier-naming>

```
[OBJECT_NAME_INCORRECT] object structure name should be in PascalCase and should contain only latin (ASCII) letters or numbers
```

Raw output

`input-paths`: custom source sets

- Default: none
- Required: **no**

One or more patterns which indicate the files or directories to check. Use a multiline string to specify multiple inputs.

- If an input is a path to a file, it is passed to *diKTat* as-is:

  ```
      - uses: saveourtool/benedikt@v2
        with:
          input-paths: |
            path/to/file.kt
  ```

- If an input is a path to a directory, the directory is recursively traversed, and all `*.kt` and `*.kts` files are passed to *diKTat*.

  ```
      - uses: saveourtool/benedikt@v2
        with:
          input-paths: |
            src/main/kotlin
            src/test/kotlin
  ```

- If an input is an *Ant*-style path pattern (such as `**/*.kt`), *diKTat* expands it into the list of files that match the path pattern. Path patterns may be negated, e. g.: `!src/**/*Test.kt` or `!src/**/generated/**`.

  ```
      - uses: saveourtool/benedikt@v2
        with:
          input-paths: |
            **/*.kt
            **/*.kts
            !**/generated/**
  ```

If this input parameter is not specified, this is equivalent to setting it to `.`, meaning *diKTat* will check all `*.kt` and `*.kts` files in the project directory unless configured otherwise.

`java-distribution` and `java-version`: running *diKTat* using a custom JVM

It's possible to run *diKTat* with a custom JVM using the `actions/setup-java` action. The following input parameters may be specified:

- `java-distribution`: the Java distribution, see the list of supported distributions.
  - Default: `temurin`
  - Required: **no**
- `java-version`: the Java version to set up. Takes a whole or semver Java version. See examples of supported syntax.
  - Default: none
  - Required: **no**

NOTE Setting just the `java-distribution` property in order to use a custom JDK is not sufficient: you'll need to set **both** `java-distribution` **and** `java-version`:

```
    - uses: saveourtool/benedikt@v2
      with:
        java-distribution: 'temurin'
        java-version: 17
```

`fail-on-error`: suppressing lint errors

- Default: `true`
- Required: **no**

If `false`, the errors are still reported, but the step completes successfully. If `true` (the default), then lint errors reported by *diKTat* are considered fatal (i.e. the current step terminates with a failure):

```
    - uses: saveourtool/benedikt@v2
      with:
        fail-on-error: true
```

NOTE This flag only affects the case when *diKTat* exits with code **1**. Higher exit codes are *always* fatal.

`debug`: enabling debug logging

- Default: `false`
- Required: **no**

Debug logging can be enabled by setting the `debug` input parameter to `true`:

```
    - uses: saveourtool/benedikt@v2
      with:
        debug: true
```

Outputs

The action returns the exit code of the command-line client using the `exit-code` output parameter, e. g.:

```
jobs:
  diktat_check:
    name: 'diKTat Check'
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4

      - id: diktat
        uses: saveourtool/benedikt@v2

      - name: 'Read the exit code of diKTat'
        if: ${{ always() }}
        run: echo "diKTat exited with code ${{ steps.diktat.outputs.exit-code }}"
        shell: bash
```

The exit codes are documented here.