rules.sonarsource.com

Kotlin static code analysis: Regular expressions should not be too complicated

3-4 minutes

Overly complicated regular expressions are hard to read and to maintain and can easily cause hard-to-find bugs. If a regex is too complicated, you should consider replacing it or parts of it with regular code or splitting it apart into multiple patterns at least.

The complexity of a regular expression is determined as follows:

Each of the following operators increases the complexity by an amount equal to the current nesting level and also increases the current nesting level by one for its arguments:

- `|` - when multiple `|` operators are used together, the subsequent ones only increase the complexity by 1
- && (inside character classes) - when multiple && operators are used together, the subsequent ones only increase the complexity by 1
- Quantifiers (`*`, `+`, `?`, `{n,m}`, `{n,}` or `{n}`)
- Non-capturing groups that set flags (such as `(?i:some_pattern)` or `(?i)some_pattern`)
- Lookahead and lookbehind assertions

Additionally, each use of the following features increase the complexity by 1 regardless of nesting:

- character classes
- back references

If a regular expression is split among multiple variables, the complexity is calculated for each variable individually, not for the whole regular expression. If a regular expression is split over multiple lines, each line is treated individually if it is accompanied by a comment (either a Java comment or a comment within the regular expression), otherwise the regular expression is analyzed as a whole.

Noncompliant Code Example

```
if (dateString.matches(Regex("^(?:(?:31(\\/|-|\\.)(?:0?[13578]|1[02]))
\\1|(?:(?:29|30)(\\/|-|\\.)(?:0?[13-9]|1[0-2])\\2))(?:(?:1[6-9]|[2-9]\\d)?
\\d{2})$|^(?:29(\\/|-|\\.)0?2\\3(?:(?:(?:1[6-9]|[2-9]\\d)?(?:0[48]|[2468]
[048]|[13579][26])|(?:(?:16|[2468][048]|[3579][26])00))))$|^(?:0?
[1-9]|1\\d|2[0-8])(\\/|-|\\.)(?:(?:0?[1-9])|(?:1[0-2]))\\4(?:(?:1[6-9]|
[2-9]\\d)?\\d{2})$"))) {
    handleDate(dateString)
}
```

Compliant Solution

```
if (dateString.matches(Regex("^\\d{1,2}([-/.])\\d{1,2}\\1\\d{1,4}$"))) {
    val dateParts = dateString.split("[-/.]").toTypedArray()
    val day = dateParts[0].toInt()
    val month = dateParts[1].toInt()
    val year = dateParts[2].toInt()
    // Put logic to validate and process the date based on its integer
parts here
}
```

Exceptions

Regular expressions are only analyzed if all parts of the regular expression are either string literals, effectively final local variables or `static final` fields, all of which can be combined using the '+' operator.

When a regular expression is split among multiple variables or commented lines, each part is only analyzed if it is syntactically valid by itself.