rules.sonarsource.com

Swift static code analysis: Trailing closure syntax should be used for all closure parameters at the end of a parameter list

2 minutes

The use of trailing closure syntax can make the code clearer, but it should only be used when all the closure arguments can be passed as trailing closures. The use of a trailing-closure syntax only for some of them can be messy and confusing.

If the function takes a closure parameter in the middle of the parameter list, followed by a non-closure parameter, these parameters cannot use trailing closure syntax. In such a case, passing all the closures consistently as regular parameters inside the parameter list improves readability.

This rule raises an issue when trailing closure syntax is not used or is used partially for a call with all the closure arguments at the end of the parameter list.

Noncompliant Code Example

```
UIView.animateWithDuration(1.0, animations: {  // Noncompliant
  self.myView.alpha = 0
})
```

```
UIView.animateWithDuration(1.0, animations: {  // Noncompliant, only one closure uses the syntax
  self.myView.alpha = 0
}) {
  self.blurBg.hidden = true
}

complexAction( // Noncompliant, only one closure uses the syntax
  first: {$0 + $1},
  repeat: 15
) {$0 * 2}
```

Compliant Solution

```
UIView.animateWithDuration(1.0) {
  self.myView.alpha = 0
}

UIView.animateWithDuration(1.0) {
  self.myView.alpha = 0
} completion: {
  self.blurBg.hidden = true
}

complexAction(first: { $0 + $1 }, repeat: 15, last: { $0 * 2 } ) // The
first closure cannot be placed in the trailing position
```