Swift static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your SWIFT code

1. Hard-coded credentials are security-sensitive
   Security Hotspot
2. Methods and field names should not be the same or differ only by capitalization
   Code Smell
3. Cipher algorithms should be robust
   Vulnerability
4. Using weak hashing algorithms is security-sensitive
   Security Hotspot
5. Cognitive Complexity of functions should not be too high
   Code Smell
6. "try!" should not be used
   Code Smell
7. String literals should not be duplicated
   Code Smell
8. Functions and closures should not be empty
   Code Smell
9. Collection elements should not be replaced unconditionally
   Bug
10. Collection sizes comparisons should make sense
    Bug
11. All branches in a conditional structure should not have exactly the same implementation
    Bug
12. Infix operators that end with "=" should update their left operands
    Bug
13. Precedence and associativity of standard operators should not be changed
    Bug
14. Return values from functions without side effects should not be ignored
    Bug
15. Related "if/else if" statements and "cases" in a "switch" should not have the same condition
    Bug

16. Identical expressions should not be used on both sides of a binary operator
Bug
17. All code should be reachable
Bug
18. Loops with at most one iteration should be refactored
Bug
19. "IBInspectable" should be used correctly
Code Smell
20. Functions should not have identical implementations
Code Smell
21. Ternary operators should not be nested
Code Smell
22. Closure expressions should not be nested too deeply
Code Smell
23. Backticks should not be used around symbol names
Code Smell
24. Operators should be surrounded by whitespace in function definitions
Code Smell
25. Two branches in a conditional structure should not have exactly the same implementation
Code Smell
26. Unused assignments should be removed
Code Smell
27. A field should not duplicate the name of its containing class
Code Smell
28. "switch" statements should not have too many "case" clauses
Code Smell
29. Sections of code should not be commented out
Code Smell
30. Statements should be on separate lines
Code Smell
31. Unused function parameters should be removed
Code Smell
32. Unused "private" functions should be removed

	Code Smell
33.	
	Track uses of "FIXME" tags Code Smell
34.	
	Local variables should not have the same name as fields or "enum" cases Code Smell
35.	
	Redundant pairs of parentheses should be removed Code Smell
36.	
	Nested blocks of code should not be left empty Code Smell
37.	
	Functions should not have too many parameters Code Smell
38.	
	Collapsible "if" statements should be merged Code Smell
39.	
	Unused labels should be removed Code Smell
40.	
	Using hardcoded IP addresses is security-sensitive Security Hotspot
41.	
	Jump statements should not be redundant Code Smell
42.	
	"break" should be the only statement in a "case" Code Smell
43.	
	Sequential tests should not check the same condition Code Smell
44.	
	"catch" clauses should do more than rethrow Code Smell
45.	
	Trailing closures should not begin on new lines Code Smell
46.	
	Fields and variables that are never updated should be constant Code Smell
47.	
	Classes should not be empty Code Smell
48.	
	Boolean checks should not be inverted Code Smell
49.	
	Multiple variables should not be declared on the same line

	Code Smell
50.	
	Unused local variables should be removed Code Smell
51.	
	"switch" statements should have at least 3 "case" clauses Code Smell
52.	
	Type parameter names should comply with a naming convention Code Smell
53.	
	Local variable and function parameter names should comply with a naming convention Code Smell
54.	
	Field names should comply with a naming convention Code Smell
55.	
	Constant names should comply with a naming convention Code Smell
56.	
	Protocol names should comply with a naming convention Code Smell
57.	
	Boolean literals should not be redundant Code Smell
58.	
	URIs should not be hardcoded Code Smell
59.	
	Class names should comply with a naming convention Code Smell
60.	
	Function names should comply with a naming convention Code Smell
61.	
	Track uses of "TODO" tags Code Smell
62.	
	Deprecated code should be removed Code Smell
63.	
	Neither DES (Data Encryption Standard) nor DESede (3DES) should be used Vulnerability
64.	
	Functions and variables should not be defined outside of classes Code Smell
65.	
	Track lack of copyright and license headers Code Smell
66.	
	SHA-1 and Message-Digest hash algorithms should not be used in secure contexts

Vulnerability

67.

Implicitly unwrapped optionals should not be used
Bug

68.

Conditional compilation should not be used
Code Smell

69.

Modulus results should not be checked for direct equality
Code Smell

70.

"switch" statements should not be nested
Code Smell

71.

Cyclomatic Complexity of functions should not be too high
Code Smell

72.

Control flow statements "if", "for", "for in", "while", "do while" and "switch" should not be nested too deeply
Code Smell

73.

Cyclomatic Complexity of classes should not be too high
Code Smell

74.

"if ... else if" constructs should end with "else" clauses
Code Smell

75.

Expressions should not be too complex
Code Smell

76.

The ternary operator should not return the same value regardless of the condition
Bug

77.

Floating point numbers should not be tested for equality
Bug

78.

Useless "if true {...}" and "if false {...}" blocks should be removed
Bug

79.

Increment (++) and decrement (--) operators should not be used in a method call or mixed with other operators in an expression
Code Smell

80.

"IBOutlet" variables should be private
Code Smell

81.

Filter conditions should be used as predicates to "first"
Code Smell

82.

Duplicate values should not be passed as arguments
Code Smell

83.	Force casts should not be used Code Smell
84.	Trailing closure syntax should be used for all closure parameters at the end of a parameter list Code Smell
85.	Function type parameters should come at the end of the parameter list Code Smell
86.	Trailing closure syntax should not be used when multiple parameters are of function type Code Smell
87.	Comments should not be nested Code Smell
88.	Track parsing failures Code Smell
89.	Classes should not have too many lines of code Code Smell
90.	Types should be defined in separate source files Code Smell
91.	Files should not be too complex Code Smell
92.	Functions should not have too many lines of code Code Smell
93.	Closures should not have too many lines Code Smell
94.	"switch case" clauses should not have too many lines of code Code Smell
95.	Functions should not contain too many return statements Code Smell
96.	Files should not have too many lines of code Code Smell
97.	Lines should not be too long Code Smell
98.	Parentheses should be omitted when trailing closure is the only argument Code Smell
99.	Tuples should not be too large

Code Smell

100. Functions should not return constants
Code Smell

101. Operator functions should call existing functions
Code Smell

102. Optionals should not be force-unwrapped
Code Smell

103. "self" should only be used when required
Code Smell

104. "get" should be omitted in read-only computed properties and subscripts
Code Smell

105. Statements should not end with semicolons
Code Smell

106. "return" should be omitted from single-expression closures
Code Smell

107. Access control should be specified for top-level definitions
Code Smell

108. Enumeration members should comply with a naming convention
Code Smell

109. Enumeration types should comply with a naming convention
Code Smell

110. Files should not be empty
Code Smell

111. Underscores should be used to make large numbers readable
Code Smell

112. "struct" names should comply with a naming convention
Code Smell

113. Statements should end with semicolons
Code Smell

114. Comments should not be located at the end of lines of code
Code Smell

115. Lines should not end with trailing whitespaces
Code Smell

116. Files should contain an empty newline at the end

	Code Smell
117.	
	A close curly brace should be located at the beginning of a line Code Smell
118.	
	An open curly brace should be located at the end of a line Code Smell
119.	
	Tabulation characters should not be used Code Smell