C and Cpp Analyzer 2.2 - Release Notes

2.2.2-funcrel

Performance Improvements

Summary
Correction of CPP-346 C++ extension execution at application level takes too long

2.2.1-funcrel

Resolved Issues

Customer Ticket Id	Details
24685	Incorrect links to Cast-Sytem objects leading to false violations

2.2.0-funcrel

Other Updates

Details
Missing violations for "Avoid using 'enum' types as operands for arithmetic contexts" while using & or &= with enums
False violation for "Avoid performing conversion from a function pointer to any other type (C/C++)" for GetProcAddress
False violation for "Avoid performing conversion from a function pointer to any other type (C/C++)" for MFC standard macros
Missing violation for "Avoid calling virtual method from constructor\destructor"
False violation for "Avoid using 'enum' types as operands for arithmetic contexts"

2.2.0-beta1

Rules

Rule Id	New Rule	Details
1065084	TRUE	Avoid performing conversion from a function pointer to any other type (C/C++)
1065086	TRUE	Avoid casting and converting a pointer type to an integral type (C/C++)

1065090	TRUE	Avoid using 'enum' types as operands for arithmetic contexts
1065092	TRUE	Avoid having boolean operators with non-boolean operands (C/C++)
1065094	TRUE	Avoid having expressions with bool type to be used as operands on operators other than =, &&, ||, !, ==, !=, the unary & operator, and the conditional operator
1065096	TRUE	Avoid dynamic_cast to convert current object to its derived class from its constructor or destructor
1065098	TRUE	Avoid delete operators to exit with an exception
1065100	TRUE	Avoid move constructor and move assignment operator to exit with an exception
1065080	TRUE	Avoid using C-style and functional notation casts (C++)
1065082	TRUE	Handlers of a function-try-block implementation of a class constructor ordestructor shall not reference non-static members from this class or its bases

Other Updates

Details
Constructors and destructors are detected as "C++ Function" with name ending with "@ctor" and "@dtor" respectively. In case, if/when class definition is not found, constructors and destructors would be named as @ctor and @dtor respectively. Analyzer log will contain following message in this scenario: Definition for '<class name>' not found. '@ctor\@dtor' would be detected as 'C++ Function'.
Incorrect object detection when unresolved macro is used inside struct\class
False violation for lambda for "Avoid having a method call or additional expressions in a statement using "" or "--" operators (C/C)"
Missing violation for _tmain for "Ensure that there is at least one exception handler to catch-all otherwise unhandled exceptions in the main function (C++)"
False violation for _tmain for "Avoid using catch all"
False violations for QR "Avoid testing floating point numbers for equality"

2.2.0-alpha1

Rules

Rule Id	New Rule	Details
1065066	TRUE	Avoid having a method call or additional expressions in a statement using "+" or "--" operators (C/C+)
1065068	TRUE	Avoid using Digraphs (C++)
1065070	TRUE	Avoid using Trigraphs
1065072	TRUE	Avoid using Unions
1065074	TRUE	All constructors that are callable with a single argument of fundamental type shall be declared explicit.
1065076	TRUE	Avoid declaring data members in non-POD classes as public or protected
1065078	TRUE	A base class shall be declared virtual only if it is used in a diamond hierarchy

Other Updates

Details
CASTONCAST: access violation while parsing template. NOTE: As a result of this fix, objects and links may increase. Also, as the files are not be skipped, analysis time may increase.
CASTONCAST: access violation while parsing function declaration. NOTE: As a result of this fix, objects and links may increase. Also, as the files are not be skipped, analysis time may increase.
CASTONCAST: False violation for "All if … else if constructs shall be terminated with an else clause (C/C++)" if else is not delimited by {}

CASTONCAST: False violations for "Avoid non-void return type function without an explicit return of an expression (C/C++)". NOTE: In certain cases, constructors and destructors were detected as C++ Function instead of constructors and destructors. In such cases, false violations was displayed. This has been fixed.
CASTONCAST: False Violation for "Ensure Switch statements have at least 2 case clauses (C/C++)" with nested switch
CASTONCAST: False violation for "Avoid missing default in switch statements" with nested switch
Run-time exception while parsing struct when macro used in struct is not resolved. NOTE: As a result of this fix, objects and links may increase. Also, as the files are not be skipped, analysis time may increase.
CASTONCAST: False violation for "Avoid using "sizeof" on expressions that contain side effects" when using C++ version of _countof