

Table of Contents (exploded view)

- [Abstract for XL C/C++ Language Reference](#)
- [Scope and linkage](#)
 - [Scope](#)
 - [Block/local scope](#)
 - [Function scope](#)
 - [Function prototype scope](#)
 - [File/global scope](#)
 - [Examples of scope in C](#)
 - [Class scope \(C++ only\)](#)
 - [Namespaces of identifiers](#)
 - [Name hiding \(C++ only\)](#)
 - [Program linkage](#)
 - [Internal linkage](#)
 - [External linkage](#)
 - [No linkage](#)
 - [Language linkage \(C++ only\)](#)
- [Lexical elements](#)
 - [Tokens](#)
 - [Keywords](#)
 - [Identifiers](#)
 - [Literals](#)
 - [Integer literals](#)
 - [Boolean literals](#)
 - [Floating-point literals](#)
 - [Fixed-point decimal literals](#)
 - [Character literals](#)
 - [String literals](#)
 - [Pointer literal \(C++11\)](#)
 - [Punctuators and operators](#)
 - [Source program character set](#)
 - [Multibyte characters](#)
 - [Escape sequences](#)
 - [The Unicode standard](#)
 - [Digraph characters](#)
 - [Trigraph sequences](#)
 - [Comments](#)
- [Data objects and declarations](#)
 - [Overview of data objects and declarations](#)
 - [Overview of data objects](#)
 - [Overview of data declarations and definitions](#)
 - [Static_assert declaration \(C11\)](#)
 - [static_assert declaration \(C++11\)](#)
 - [Storage class specifiers](#)
 - [The auto storage class specifier](#)
 - [The static storage class specifier](#)
 - [The extern storage class specifier](#)
 - [The mutable storage class specifier \(C++ only\)](#)
 - [The register storage class specifier](#)
 - [Type specifiers](#)
 - [Integral types](#)
 - [Boolean types](#)
 - [Floating-point types](#)
 - [Fixed point decimal types \(C only\)](#)
 - [Character types](#)
 - [The void type](#)
 - [User-defined types](#)
 - [Structures and unions](#)
 - [Enumerations](#)
 - [Compatibility of structures, unions, and enumerations \(C only\)](#)
 - [typedef definitions](#)
 - [The auto type specifier \(C++11\)](#)
 - [The decltype\(expression\) type specifier \(C++11\)](#)
 - [The constexpr specifier \(C++11\)](#)
 - [Compatibility of arithmetic types \(C only\)](#)
 - [Type qualifiers](#)
 - [The __callback type qualifier](#)
 - [The const type qualifier](#)
 - [The __far type qualifier \(C only\)](#)
 - [The __ptr32 type qualifier](#)
 - [The __ptr64 type qualifier \(C only\)](#)
 - [The restrict type qualifier](#)
 - [The volatile type qualifier](#)
 - [Type attributes \(IBM extension\)](#)
 - [The amode31 | amode64 type attribute \(C only\) \(IBM Extension\)](#)
 - [The amode | noarmode type attribute \(C only\) \(IBM extension\)](#)

- [The `may_alias` type attribute \(IBM extension\)](#)
- [Declarators](#)
 - [Overview of declarators](#)
 - [Examples of declarators](#)
 - [Type names](#)
 - [Pointers](#)
 - [Pointer arithmetic](#)
 - [Type-based aliasing](#)
 - [Compatibility of pointers \(C only\)](#)
 - [Null pointers](#)
 - [Arrays](#)
 - [Variable length arrays](#)
 - [Compatibility of arrays \(C only\)](#)
 - [References \(C++ only\)](#)
 - [Initializers](#)
 - [Initialization and storage classes](#)
 - [Designated initializers for aggregate types \(C only\)](#)
 - [Initialization of structures and unions](#)
 - [Initialization of enumerations](#)
 - [Initialization of pointers](#)
 - [Initialization of arrays](#)
 - [Initialization of references \(C++ only\)](#)
 - [Initialization of complex types \(C11\)](#)
 - [Declarator qualifiers](#)
 - [The `_Packed` qualifier \(C only\)](#)
 - [The `_Export` qualifier \(C++ only\)](#)
 - [Variable attributes \(IBM extension\)](#)
 - [The aligned variable attribute \(IBM extension\)](#)
- [Type conversions](#)
 - [Arithmetic conversions and promotions](#)
 - [Integral conversions](#)
 - [Boolean conversions](#)
 - [Floating-point conversions](#)
 - [Packed decimal conversions \(C only\)](#)
 - [Usual arithmetic conversions](#)
 - [Integral and floating-point promotions](#)
 - [Lvalue-to-rvalue conversions](#)
 - [Pointer conversions](#)
 - [Conversion to `void*`](#)
 - [Reference conversions \(C++ only\)](#)
 - [Function argument conversions](#)
- [Expressions and Operators](#)
 - [Lvalues and rvalues](#)
 - [Primary expressions](#)
 - [Names](#)
 - [Literals](#)
 - [Integer constant expressions](#)
 - [Identifier expressions \(C++ only\)](#)
 - [Parenthesized expressions \(\)](#)
 - [Generic selection \(C11\)](#)
 - [Scope resolution operator `::` \(C++ only\)](#)
 - [Generalized constant expressions \(C++11\)](#)
 - [Function call expressions](#)
 - [Member expressions](#)
 - [Dot operator `.`](#)
 - [Arrow operator `->`](#)
 - [Unary expressions](#)
 - [Increment operator `++`](#)
 - [Decrement operator `--`](#)
 - [Unary plus operator `+`](#)
 - [Unary minus operator `-`](#)
 - [Logical negation operator `!`](#)
 - [Bitwise negation operator `~`](#)
 - [Address operator `&`](#)
 - [Indirection operator `*`](#)
 - [The `typeid` operator \(C++ only\)](#)
 - [The `__alignof__` operator \(IBM extension\)](#)
 - [The `sizeof` operator](#)
 - [The `typeof` operator \(IBM extension\)](#)
 - [The `digitsof` and `precisionof` operators \(C only\)](#)
 - [The `__real__` and `__imag__` operators \(IBM extension\)](#)
 - [Binary expressions](#)
 - [Assignment operators](#)
 - [Multiplication operator `*`](#)
 - [Division operator `/`](#)
 - [Remainder operator `%`](#)
 - [Addition operator `+`](#)
 - [Subtraction operator `-`](#)

- [Bitwise left and right shift operators << >>](#)
 - [Relational operators < > <= >=](#)
 - [Equality and inequality operators == !=](#)
 - [Bitwise AND operator &](#)
 - [Bitwise exclusive OR operator ^](#)
 - [Bitwise inclusive OR operator |](#)
 - [Logical AND operator &&](#)
 - [Logical OR operator ||](#)
 - [Array subscripting operator \[\]](#)
 - [Comma operator ,](#)
 - [Pointer to member operators .* ->* \(C++ only\)](#)
- [Conditional expressions](#)
 - [Types in conditional C expressions \(C only\)](#)
 - [Types in conditional C++ expressions \(C++ only\)](#)
 - [Examples of conditional expressions](#)
- [Cast expressions](#)
 - [Cast operator \(\)](#)
 - [The static_cast operator \(C++ only\)](#)
 - [The reinterpret_cast operator \(C++ only\)](#)
 - [The const_cast operator \(C++ only\)](#)
 - [The dynamic_cast operator \(C++ only\)](#)
- [Compound literal expressions](#)
- [new expressions \(C++ only\)](#)
 - [Placement syntax](#)
 - [Initialization of objects created with the new operator](#)
 - [Handling new allocation failure](#)
- [delete expressions \(C++ only\)](#)
- [throw expressions \(C++ only\)](#)
- [Operator precedence and associativity](#)
- [Reference collapsing \(C++11\)](#)
- [Statements](#)
 - [Labeled statements](#)
 - [Labels as values \(IBM extension\)](#)
 - [Expression statements](#)
 - [Resolution of ambiguous statements \(C++ only\)](#)
 - [Block statements](#)
 - [Example of blocks](#)
 - [Selection statements](#)
 - [The if statement](#)
 - [The switch statement](#)
 - [Iteration statements](#)
 - [The while statement](#)
 - [The do statement](#)
 - [The for statement](#)
 - [Jump statements](#)
 - [The break statement](#)
 - [The continue statement](#)
 - [The return statement](#)
 - [The goto statement](#)
 - [Null statement](#)
 - [Inline assembly statements \(IBM extension\)](#)
 - [Restrictions on inline assembly statements \(IBM extension\)](#)
 - [Examples of inline assembly statements \(IBM extension\)](#)
- [Functions](#)
 - [Function declarations and definitions](#)
 - [Function declarations](#)
 - [Function definitions](#)
 - [Explicitly defaulted functions \(C++11\)](#)
 - [Deleted functions \(C++11\)](#)
 - [Examples of function declarations](#)
 - [Examples of function definitions](#)
 - [Compatible functions \(C only\)](#)
 - [Multiple function declarations \(C++ only\)](#)
 - [Function storage class specifiers](#)
 - [The static storage class specifier](#)
 - [The extern storage class specifier](#)
 - [Function specifiers](#)
 - [The inline function specifier](#)
 - [The _Noreturn function specifier](#)
 - [The __cdecl function specifier \(C++ only\)](#)
 - [The __Export function specifier \(C++ only\)](#)
 - [Function return type specifiers](#)
 - [Function return values](#)
 - [Function declarators](#)
 - [Parameter declarations](#)
 - [Trailing return type \(C++11\)](#)
 - [Function attributes \(IBM extension\)](#)
 - [always_inline \(IBM extension\)](#)

- [amode31 | amode64 \(C only\) \(IBM extension\)](#)
 - [armode | noarmode \(C only\) \(IBM extension\)](#)
 - [gnu_inline \(IBM extension\)](#)
 - [malloc \(IBM extension\)](#)
 - [used \(IBM extension\)](#)
- [The main\(\) function](#)
 - [Command-line arguments](#)
- [Function calls](#)
 - [Pass by value](#)
 - [Pass by pointer](#)
 - [Pass by reference \(C++ only\)](#)
- [Allocation and deallocation functions \(C++ only\)](#)
- [Default arguments in C++ functions \(C++ only\)](#)
 - [Restrictions on default arguments \(C++ only\)](#)
 - [Evaluation of default arguments \(C++ only\)](#)
- [Pointers to functions](#)
- [Constexpr functions \(C++11\)](#)
- [Namespaces \(C++ only\)](#)
 - [Defining namespaces \(C++ only\)](#)
 - [Declaring namespaces \(C++ only\)](#)
 - [Creating a namespace alias \(C++ only\)](#)
 - [Creating an alias for a nested namespace \(C++ only\)](#)
 - [Extending namespaces \(C++ only\)](#)
 - [Namespaces and overloading \(C++ only\)](#)
 - [Unnamed namespaces \(C++ only\)](#)
 - [Namespace member definitions \(C++ only\)](#)
 - [Namespaces and friends \(C++ only\)](#)
 - [The using directive \(C++ only\)](#)
 - [The using declaration and namespaces \(C++ only\)](#)
 - [Explicit access \(C++ only\)](#)
 - [Inline namespace definitions \(C++11\)](#)
- [Overloading \(C++ only\)](#)
 - [Overloading functions \(C++ only\)](#)
 - [Restrictions on overloaded functions \(C++ only\)](#)
 - [Overloading operators \(C++ only\)](#)
 - [Overloading unary operators \(C++ only\)](#)
 - [Overloading increment and decrement operators \(C++ only\)](#)
 - [Overloading binary operators \(C++ only\)](#)
 - [Overloading assignments \(C++ only\)](#)
 - [Overloading function calls \(C++ only\)](#)
 - [Overloading subscripting \(C++ only\)](#)
 - [Overloading class member access \(C++ only\)](#)
 - [Overload resolution \(C++ only\)](#)
 - [Implicit conversion sequences \(C++ only\)](#)
 - [Resolving addresses of overloaded functions \(C++ only\)](#)
- [Classes \(C++ only\)](#)
 - [Declaring class types \(C++ only\)](#)
 - [Using class objects \(C++ only\)](#)
 - [Classes and structures \(C++ only\)](#)
 - [Scope of class names \(C++ only\)](#)
 - [Incomplete class declarations \(C++ only\)](#)
 - [Nested classes \(C++ only\)](#)
 - [Local classes \(C++ only\)](#)
 - [Local type names \(C++ only\)](#)
- [Class members and friends \(C++ only\)](#)
 - [Class member lists \(C++ only\)](#)
 - [Data members \(C++ only\)](#)
 - [Member functions \(C++ only\)](#)
 - [Inline member functions \(C++ only\)](#)
 - [Constant and volatile member functions \(C++ only\)](#)
 - [Virtual member functions \(C++ only\)](#)
 - [Special member functions \(C++ only\)](#)
 - [Member scope \(C++ only\)](#)
 - [Pointers to members \(C++ only\)](#)
 - [The this pointer \(C++ only\)](#)
 - [Static members \(C++ only\)](#)
 - [Using the class access operators with static members \(C++ only\)](#)
 - [Static data members \(C++ only\)](#)
 - [Static member functions \(C++ only\)](#)
 - [Member access \(C++ only\)](#)
 - [Friends \(C++ only\)](#)
 - [Friend scope \(C++ only\)](#)
 - [Friend access \(C++ only\)](#)
- [Inheritance \(C++ only\)](#)
 - [Derivation \(C++ only\)](#)
 - [Inherited member access \(C++ only\)](#)
 - [Protected members \(C++ only\)](#)
 - [Access control of base class members \(C++ only\)](#)

- [The using declaration and class members \(C++ only\)](#)
 - [Overloading member functions from base and derived classes \(C++ only\)](#)
 - [Changing the access of a class member \(C++ only\)](#)
- [Multiple inheritance \(C++ only\)](#)
 - [Virtual base classes \(C++ only\)](#)
 - [Multiple access \(C++ only\)](#)
 - [Ambiguous base classes \(C++ only\)](#)
- [Virtual functions \(C++ only\)](#)
 - [Ambiguous virtual function calls \(C++ only\)](#)
 - [Virtual function access \(C++ only\)](#)
- [Abstract classes \(C++ only\)](#)
- [Special member functions \(C++ only\)](#)
 - [Overview of constructors and destructors \(C++ only\)](#)
 - [Constructors \(C++ only\)](#)
 - [Default constructors \(C++ only\)](#)
 - [Delegating constructors \(C++11\)](#)
 - [Constexpr constructors \(C++11\)](#)
 - [Explicit initialization with constructors \(C++ only\)](#)
 - [Initialization of base classes and members \(C++ only\)](#)
 - [Constructor execution order for class objects \(C++ only\)](#)
 - [Destructors \(C++ only\)](#)
 - [Pseudo-destructors \(C++ only\)](#)
 - [User-defined conversions \(C++ only\)](#)
 - [Conversion constructors \(C++ only\)](#)
 - [Explicit conversion constructors \(C++ only\)](#)
 - [Conversion functions \(C++ only\)](#)
 - [Explicit conversion operators \(C++11\)](#)
 - [Copy constructors \(C++ only\)](#)
 - [Copy assignment operators \(C++ only\)](#)
- [Templates \(C++ only\)](#)
 - [Template parameters \(C++ only\)](#)
 - [Type template parameters \(C++ only\)](#)
 - [Non-type template parameters \(C++ only\)](#)
 - [Template template parameters \(C++ only\)](#)
 - [Default arguments for template parameters \(C++ only\)](#)
 - [Naming template parameters as friends \(C++11\)](#)
 - [Template arguments \(C++ only\)](#)
 - [Template type arguments \(C++ only\)](#)
 - [Template non-type arguments \(C++ only\)](#)
 - [Template template arguments \(C++ only\)](#)
 - [Class templates \(C++ only\)](#)
 - [Class template declarations and definitions \(C++ only\)](#)
 - [Static data members and templates \(C++ only\)](#)
 - [Member functions of class templates \(C++ only\)](#)
 - [Friends and templates \(C++ only\)](#)
 - [Function templates \(C++ only\)](#)
 - [Template argument deduction \(C++ only\)](#)
 - [Overloading function templates \(C++ only\)](#)
 - [Partial ordering of function templates \(C++ only\)](#)
 - [Template instantiation \(C++ only\)](#)
 - [Explicit instantiation \(C++ only\)](#)
 - [Implicit instantiation \(C++ only\)](#)
 - [Template specialization \(C++ only\)](#)
 - [Explicit specialization \(C++ only\)](#)
 - [Partial specialization \(C++ only\)](#)
 - [Variadic templates \(C++11\)](#)
 - [Name binding and dependent names \(C++ only\)](#)
 - [The typename keyword \(C++ only\)](#)
 - [The template keyword as qualifier \(C++ only\)](#)
- [Exception handling \(C++ only\)](#)
 - [try blocks \(C++ only\)](#)
 - [Nested try blocks \(C++ only\)](#)
 - [catch blocks \(C++ only\)](#)
 - [Function try block handlers \(C++ only\)](#)
 - [Arguments of catch blocks \(C++ only\)](#)
 - [Matching between exceptions thrown and caught \(C++ only\)](#)
 - [Order of catching \(C++ only\)](#)
 - [throw expressions \(C++ only\)](#)
 - [Rethrowing an exception \(C++ only\)](#)
 - [Stack unwinding \(C++ only\)](#)
 - [Exception specifications \(C++ only\)](#)
 - [Special exception handling functions \(C++ only\)](#)
 - [The unexpected\(\) function \(C++ only\)](#)
 - [The terminate\(\) function \(C++ only\)](#)
 - [The set_unexpected\(\) and set_terminate\(\) functions \(C++ only\)](#)
 - [Example using the exception handling functions \(C++ only\)](#)
- [Preprocessor directives](#)
 - [Macro definition directives](#)

- [The #define directive](#)
 - [The #undef directive](#)
 - [The # operator](#)
 - [The ## operator](#)
 - [Standard predefined macro names](#)
- [File inclusion directives](#)
 - [The #include directive](#)
 - [The #include_next directive \(IBM extension\)](#)
- [Conditional compilation directives](#)
 - [The #if and #elif directives](#)
 - [The #ifdef directive](#)
 - [The #ifndef directive](#)
 - [The #else directive](#)
 - [The #endif directive](#)
 - [Extension of #endif and #else \(IBM extension\)](#)
- [Message generation directives](#)
 - [The #error directive](#)
 - [The #line directive](#)
- [The null directive \(#\)](#)
- [Pragma directives](#)
 - [The _Pragma preprocessing operator](#)
 - [Standard pragmas](#)
- [C99 preprocessor features adopted in C++11 \(C++11\)](#)
- [z/OS XL C/C++ pragmas](#)
 - [Pragma directive syntax](#)
 - [Scope of pragma directives](#)
 - [IPA effects](#)
 - [Summary of compiler pragmas by functional category](#)
 - [Language element control](#)
 - [C++ template pragmas](#)
 - [Floating point and integer control](#)
 - [Error checking and debugging](#)
 - [Listings, messages and compiler information](#)
 - [Optimization and tuning](#)
 - [Object code control](#)
 - [Portability and migration](#)
 - [Individual pragma descriptions](#)
 - [#pragma arch_section \(IBM extension\)](#)
 - [#pragma chars](#)
 - [#pragma checkout](#)
 - [#pragma comment](#)
 - [#pragma convert](#)
 - [#pragma convlit](#)
 - [#pragma csect](#)
 - [#pragma define \(C++ only\)](#)
 - [#pragma disjoint](#)
 - [#pragma do_not_instantiate \(C++ only\)](#)
 - [#pragma enum](#)
 - [#pragma environment \(C only\)](#)
 - [#pragma execution_frequency](#)
 - [#pragma export](#)
 - [#pragma extension](#)
 - [#pragma filetag](#)
 - [#pragma hashome \(C++ only\)](#)
 - [#pragma implementation \(C++ only\)](#)
 - [#pragma info \(C++ only\)](#)
 - [#pragma inline \(C only\) / noline](#)
 - [#pragma insert_asm \(C only\)](#)
 - [#pragma ishome \(C++ only\)](#)
 - [#pragma isolated_call](#)
 - [#pragma langlvl \(C only\)](#)
 - [#pragma leaves](#)
 - [#pragma linkage \(C only\)](#)
 - [#pragma longname/nolongname](#)
 - [#pragma map](#)
 - [#pragma margins/nomargins](#)
 - [#pragma namemangling \(C++ only\)](#)
 - [#pragma namemanglingrule \(C++ only\)](#)
 - [#pragma object_model \(C++ only\)](#)
 - [#pragma operator_new \(C++ only\)](#)
 - [#pragma option_override](#)
 - [#pragma options \(C only\)](#)
 - [#pragma pack](#)
 - [#pragma page \(C only\)](#)
 - [#pragma pagesize \(C only\)](#)
 - [#pragma priority \(C++ only\)](#)
 - [#pragma prolog \(C only\), #pragma epilog \(C only\)](#)
 - [#pragma reachable](#)

- [#pragma report \(C++ only\)](#)
 - [#pragma runopts](#)
 - [#pragma sequence](#)
 - [#pragma skip \(C only\)](#)
 - [#pragma strings](#)
 - [#pragma subtitle \(C only\)](#)
 - [#pragma target \(C only\)](#)
 - [#pragma title \(C only\)](#)
 - [#pragma unroll](#)
 - [#pragma variable](#)
 - [#pragma wsizeof](#)
 - [#pragma XOPTS](#)
 - [Pragma directives for parallel processing](#)
- [Compiler predefined macros](#)
 - [General macros](#)
 - [Macros indicating the z/OS XL C/C++ compiler](#)
 - [Macros related to the platform](#)
 - [Macros related to compiler features](#)
 - [Macros related to compiler option settings](#)
 - [Macros related to language levels](#)
- [The IBM XL C/C++ language extensions](#)
 - [General IBM extensions](#)
 - [Extensions for C11 compatibility](#)
 - [C++11 compatibility](#)
 - [Extensions for GNU C/C++ compatibility](#)
 - [Extensions for Unicode support](#)
 - [Extensions for vector processing support](#)
- [C and C++ compatibility on the z/OS platform](#)
- [Common Usage C language level for the z/OS platform](#)
- [Conforming to POSIX 1003.1](#)
- [Implementation-defined behavior](#)