Sela.

CPP20Update

C++20 Update for C++ Developers

college@sela.co.il 03-6176666

C++20 Update for C++ Developers

CPP20Update - Version: 1

3 days Course

Description:

A three-day instructor-led course covers the new features in C++20 language and standard library. C++20 is a major update to the language and includes new additions like immediate functions, concepts, coroutines and modules and also enhancements to existing language features like lambdas and constexpr virtual functions. With C++20 the language becomes more powerful and capable and enables writing more correct and efficient code.

Intended audience:

C++ Developers with experience in C++11/C++17 standards

Prerequisites:

Experience in programming with modern C++ is a must.
Experience with C++17 is preferable.

Objectives:

Become versed in C++20 language features and core standard library facilities new in C++20
The participants will understand the new additions to the language.
The participants will understand the purpose and optimal usage of new library additions.
The participants will be able to understand the internal workings of coroutines.
The participants will be able to use the new module system.
The participants will be able to use the new synchronization facilities.

Topics:

Core Language

- Basics
  - º Deprecation of inconsistent arithmetic conversion on enumerations
  - º Consistency improvements for comparisons
  - º char8_t and UTF-8
  - º Signed integers and 2’s complement
  - º Consistent bitwise shift operators
  - º Array size deduction in new-expressions
  - º Using enum
  - º Nested inline namespaces
- Compile Time
  - º constinit
  - º Enhancements to constexpr: virtual, try, allocations,
  - º consteval and immediate functions
- Range-based for loop with initializer
- [[no_unique_address]] attribute
- [[likely]] and [[unlikely]] attributes
- Class Related
  - º Aggregates
    - - Aggregate designated initializers
    - - Default member initializers for bit-fields
    - - Initializing aggregates from a parenthesized list of values
  - º Uniform Initialization semantics and preventing accidental aggregates
  - º Conditionally Trivial Special Member Functions
  - º Class-specific destroying deallocation functions
  - º Default comparison operators
  - º 3-way comparison operator (spaceship operator)

    - Conditionally explicit specifier
- Template Related
    - Abbreviated function template
    - Relaxed requirements for typename with dependent names
    - Non-type template parameter using float and literal class types
- Lambda Expression Enhancements
    - Familiar template syntax for generic lambdas
    - Lambda expressions init-capture and template pack expansion
    - Explicit by value capture of this and deprecating implicit capture of this by value
    - Default construction and assigning
- Constraints
    - Requirements
    - Require clause
- Concepts
    - Defining Concepts
    - Standard library Concepts
    - Constrained auto declaration
- Coroutines
    - Execution model and compiler transformations
    - Promise
    - co_yield, co_redturn and co_await
    - Generator
- Modules

Standard Library

- Formatting Library
- Threading
    - Floating point atomics
    - shared_ptr atomics
    - std::barrier

    - º std::latch
    - º std::counting_semaphore
    - º std::jthread and cancelation classes
- Ranges Library