Полезные нововведения C++17

№ урока: 10 **Курс:** C++ Advanced

Средства обучения: Qt Creator

Обзор, цель и назначение урока

Научить студентов понимать и применять на практике механизмы нового стандарта C++17.

Изучив материал данного занятия, учащийся сможет

- Понимать, что такое std::variant, std::optional, std::any и т.д.
- Уметь объяснить разницу между union & std::variant.
- Понимать, что такое новые атрибуты nodiscard, fallthrough, maybe_unused.
- Знать особенности работы constexpr if, constexpr lambda.
- Знать о таком понятии, как structural binding.

Содержание урока

1.«Синтаксический сахар» C++ 17
2.Нововведения шаблонов
3.Новые возможности лямбда выражений
4.Новые атрибуты
5.Новые полезные классы стандартной библиотеки
6.Декомпозиция при объявлении
7.Атрибуты nodiscard, fallthrough, maybe_unused
8.string_view
9.optional и variant
10.std::filesystem

Резюме

Стандарт добавил к себе +200 страниц, в то время как C++14 добавил около 20 страниц.

Является major release (C++14 – minor release).

Около 90 предложений для внесения в стандарт были получены.

Планируется выход C++20.

C++17 поддерживается в clang4, gcc7, MSVC2017.

В C++17 есть ограничения декомпозиции при объявлении:

нельзя явно указывать типы декомпозируемых элементов
нельзя использовать вложенную декомпозицию вида auto [title, [header, content]] = ...

CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Декомпозиция при объявлении в принципе может раскладывать любой класс — достаточно один раз написать подсказку путём специализации tuple_element, tuple_size и get.

Ключевые правила:

функции вида std::make_pair больше не нужны: смело пишите выражения std::pair{10, "hello"s}, компилятор сам выведет тип
шаблонные RAII вида std::lock_guard<std::mutex> guard(mutex); станут короче: std::lock_guard guard(mutex);
функции std::make_unique и std::make_shared по-прежнему нужны

завершайте все блоки case, кроме последнего, либо атрибутом [[fallthrough]], либо инструкцией break;
используйте [[nodiscard]] для функций, возвращающих код ошибки или владеющий указатель (неважно, умный или нет)
используйте [[maybe_unused]] для переменных, которые нужны только для проверки в assert

Правила:

в параметрах всех функций и методов вместо const string& старайтесь принимать невладеющий string_view по значению
возвращайте из функций и методов владеющий string, как и раньше

будьте осторожны с возвратом string_view из функции: это может привести к проблеме висячих ссылок (англ. dangling pointers)

предпочитайте optional<T> вместо unique_ptr<T> для композиции объекта T, время жизни которого короче времени жизни владельца

для PIMPL используйте unique_ptr<Impl>, потому что определение Impl скрыто в файле реализации класса

используйте тип variant вместо enum или полиморфных классов в ситуации, когда состояния, такие как состояние лицензии, не могут быть описаны константами enum из-за наличия дополнительных данных в каждом из состояний

используйте тип variant вместо enum в ситуации, когда данные, такие как код ошибки в исключении, должны быть обработаны во всех вариантах, и неполная обработка вариантов должна приводить к ошибке компиляции

используйте тип variant вместо any везде, где это возможно

optional можно использовать для композиции объекта, время жизни которого короче времени жизни владельца

Закрепление материала

- Чем отличается std::string_view от std::string?
- Чем отличается std::optional от std::variant?

CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

- Для чего необходим nodiscard?

Дополнительное задание

Задание
Изучите особенности std::filesystem , parallel STL, templates.

Самостоятельная деятельность учащегося

Задание 1
Выучите основные понятия, рассмотренные на уроке.

Задание 2
Проведите миграцию одного из своих проектов на C++17 с использованием новых возможностей.

Задание 3
Зайдите на сайт MSDN.
Используя поисковые механизмы MSDN, найдите самостоятельно описание темы по каждому примеру, который был рассмотрен на уроке, так, как это представлено ниже, в разделе «Рекомендуемые ресурсы», описания данного урока. Сохраните ссылки и дайте им короткое описание.

Рекомендуемые ресурсы

https://habr.com/ru/post/343622/

https://github.com/cCppProsto/cpp/tree/master/cpp_17_features

https://ru.wikipedia.org/wiki/C%2B%2B17

http://www.modernescpp.com/index.php/c-17-avoid-copying-with-std-string-view

https://en.cppreference.com/w/cpp/filesystem/path

CyberBionic systematics
CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com