Общее введение и обзор технологий

Программная инженерия с использованием C++

Определение C++

C++ – это компилируемый язык программирования общего назначения со строгой статической типизацией, который поддерживает несколько парадигм программирования и предоставляет как низкоуровневые, так и высокоуровневые средства.

- Исходный код, сборка, инструкции машинного кода
- Процессор, среда выполнения операционной системы
- Узкая специализация, несколько сильных конкурентов
- Строгая или слабая статическая система типизации
- Сочетание нескольких парадигм программирования
- Низкоуровневые оптимизации и удобные абстракции

Формирование C++

- C++ изначально разрабатывался как набор расширений C
- C++ не является подмножеством или надмножеством C
- C++ унаследовал компоненты Ada, Algol, Fortran и Simula
- C++ повлиял на C#, Java, Python и много других языков

Первый коммерческий релиз C++ состоялся 14 октября 1985 г.

Стандартизация C++

- C++98 – фундаментальный стандарт
- C++03 – патч
- Technical Report 2005, Boost
- C++11 – ядро языка +30%, стандартная библиотека +100%
- C++14 – патч
- C++17 – патч
- C++20 – последний реализованный стандарт
- C++23 – ожидаемый стандарт
- C++26 – ожидаемый стандарт

Использование C++

- Операционные системы и системы управления
- Производительные вычислительные системы
- Математическое и физическое моделирование
- Игры и различные прикладные библиотеки
- Финансовые системы автоматической торговли
- Высоконадежные системы военного назначения

Сочетание низкоуровневых и высокоуровневых возможностей.

Парадигмы программирования

- Декларативное программирование – ожидания
- Функциональное программирование – функции
- Императивное программирование – инструкции
- Процедурное программирование – подпрограммы
- Структурное программирование – ветвления и циклы
- Объектно-ориентированное программирование – классы
- Обобщенное программирование – шаблоны и концепты
- Параллельное программирование – процессы и потоки
- Событийно-ориентированное программирование – события

C++ предоставляет полную, но зачастую опасную свободу действий.

Дополнительные библиотеки

Стандартная библиотека шаблонов и дополнительные библиотеки Boost предоставляют множество полезных инструментов.

- Google – средства с открытым исходным кодом
- OpenCV – системы компьютерного зрения
- OpenMP – параллельные вычисления
- OpenGL – высокопроизводительная графика
- Qt – графические пользовательские интерфейсы
- QuantLib – количественные финансы

Инструменты разработчика

- Компиляторы – Clang, GCC, ICC, MinGW, MSVC
- Среды разработки – CLion, Visual Studio, XCode
- Редакторы кода – Geany, Vim, Visual Studio Code
- Отладчики и профилировщики – GDB, Valgrind
- Системы автоматизации сборки – CMake, MSBuild
- Системы контроля версий – Git, SVN, Mercurial
- Сервисы для хостинга проектов – GitHub, Bitbucket
- Графические клиенты Git – GitHub Desktop, SmartGit
- Системы управления проектами – Asana, Jira, Trello

Этапы жизненного цикла проекта

- Формирование набора требований
- Оценка необходимых ресурсов
- Разработка архитектуры решения
- Детальное проектирование компонент
- Механическое написание кода
- Отладка и оптимизация системы
- Внедрение и техническая поддержка
- Вывод продукта из эксплуатации