

Современный C++

Максим Федоренко

ИАТЭ НИЯУ МИФИ
Кафедра ПМ

26 ноября 2016 г.

О себе



- ▶ Закончил ИАТЭ в 2015г, специалист АСУ
- ▶ Ведущий программист кафедры ПМ



- ▶ Пишу на C++ с 2010 года
- ▶ Руководитель команды разработки в **Latista**
- ▶ Разработчик в **Mail.Ru Group**. Департамент игр

Цели курса



Проблема: выпускник ИАТЭ, попадая в IT компанию, оказывается не знаком с современными принципами, средствами и технологиями разработки ПО

- ▶ Передача опыта
- ▶ Популяризация C++
- ▶ Повышение общей культуры программирования



► 4 занятия

1. Обзор языка C++
2. Средства и технологии разработки ПО
3. Совместная работа над проектом
4. Современный эффективный C++

C++



- ▶ Страуструп, 1983 год
- ▶ *Си с классами*
- ▶ Стандартизация языка в 1998 и 2003 годах
- ▶ Современные стандарты C++11 и C++14

Характеристики



- ▶ Универсальный язык общего назначения
- ▶ Компилируемый
- ▶ Статическая типизация



- ▶ Переносимость на уровне языка Си
- ▶ Поддержка множества стилей программирования
 - ▶ процедурное
 - ▶ ООП
 - ▶ обобщённое
 - ▶ функциональное



- ▶ Свобода выбора у программиста
- ▶ *Ты не платишь за то, что не используешь*



- ▶ Богатая семантика
- ▶ Высокий порог вхождения
- ▶ *Пример кода*



- ▶ Системное программирование
- ▶ Высокопроизводительное серверное ПО
- ▶ Кроссплатформенное ПО
- ▶ Мобильные устройства
- ▶ **Игры**

Практикум



- ▶ Проект лежит на **GitHub**: *URL*
- ▶ Сборка осуществляется с помощью **CMake**
- ▶ Среда разработки: **QtCreator**

Объектно-ориентированное программирование



- ▶ Модель предметной области представляется в виде множества типизированных объектов
- ▶ Объект абстрагирует данные, скрывая их внутри собственной реализации
- ▶ Объекты обмениваются **сообщениями**
- ▶ Типы объектов образуют иерархию наследования



- ▶ Работа с объектами различных типов единым образом
- ▶ Статический (параметрический)
- ▶ Динамический

Реализация ООП в C++



- ▶ Пользовательские типы
- ▶ Уровни доступа
- ▶ Наследование
- ▶ Виртуальные функции



- ▶ Злоупотребление наследованием
- ▶ Иерархия не обладает гибкостью
- ▶ Динамическая диспетчеризация типов снижает производительность

Комбинирование стилей при разработке на C++



- ▶ Статический полиморфизм
- ▶ Отделение алгоритмов от деталей реализации



- ▶ Замыкания (λ – функции)
- ▶ ...

Мифы о C++



Миф: Чтобы понять C++, сначала нужно выучить Си

- ▶ Си – лишь одно из подмножеств C++, далеко не лучшее
- ▶ В Си нет типобезопасности
- ▶ На Си сложно решать простые задачи



Миф: В C++ необходимо вручную управлять памятью и указателями. Языки со сборщиком мусора намного удобней и эффективней

- ▶ Сборка мусора – не понадея
- ▶ Концепция **RAII** справляется с управлением ресурсами
- ▶ Ручное управление памятью с помощью `new/delete` в C++ – моветон

Заключение



- ▶ Страуструп *Язык программирования C++*

Контакты



email: varlllog@gmail.com

skype: [varlogg](#)