Современный C++

Максим Федоренко

ИАТЭ НИЯУ МИФИ
Кафедра ПМ

26 ноября 2016 г.

О себе

О себе

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

Опыт

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

Цели курса

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

План занятий

- 4 занятия

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

C++

История

- Бьёрн 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

Пример

```
int i;
i = 5;
i = 3.14 // Warning!
i = "hello" // Error!
```

Философия C++

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

Философия C++

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

Следствия

- Богатая семантика
- Высокий порог вхождения

Пример

```
auto timer = [](auto&& func, auto&&... params) {
        // start timer
        std::forward<decltype(func)>(func) (
                std::forward<decltype(params)>
                        (params)...
        );
        // stop timer
};
```

Область применения

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

Практикум

Тестовый проект

- Проект лежит на **GitHub**:
  `https://github.com/VarLog/cpp_lessons`
- Сборка осуществляется с помощью **CMake**
- Среда разработки: **QtCreator**

Объектно-ориентированное программирование

Концепция

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

Полиморфизм

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

Реализация ООП в C++

- Пользовательские типы
- Уровни доступа
- Наследование
- Виртуальные функции

Недостатки ООП

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

Комбинирование стилей при разработке на C++

Обобщённое программирование

- Статический полиморфизм
- Отделение алгоритмов от деталей реализации

Функциональное программирование

- Замыкания (λ – функции)
- Неизменяемые типы данных

Мифы о C++

C++ vs Си

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

Ручное управление памятью

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

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

Заключение

Заключение

Литература

- Стивен Прата ***Язык программирования C++. Лекции и упражнения***
- Роберт Лафоре ***Объектно-ориентированное программирование в C++***
- Бьёрн Страуструп ***Программирование. Принципы и практика с использованием C++***

Контакты

email: varlllog@gmail.com

skype: varlogg