

ББК 32.973.26-018.1
С80

Страуструп Б.

С80 Дизайн и эволюция C++: Пер. с англ. — М.: ДМК Пресс, 2016. — 446 с.: ил. (Серия «Для программистов»).

ISBN 978-5-97060-419-9

В книге, написанной создателем языка C++ Бьерном Страуструпом, представлено описание процесса проектирования и разработки языка программирования C++.

Здесь изложены цели, принципы и практические ограничения, наложившие отпечаток на структуру и облик C++, обсужден дизайн недавно добавленных в язык средств: шаблонов, исключений, идентификации типа во время исполнения и пространств имен. Автор анализирует решения, принятые в ходе работы над языком, и демонстрирует, как правильно применять «реальный объектно-ориентированный язык программирования».

Книга удобно организована, поучительна, написана с юмором. Описание ключевых идей даст начинающему пользователю ту основу, на которой позже он выстроит свое понимание всех деталей языка. Опытный программист найдет здесь обсуждение принципиальных вопросов проектирования, что позволит ему лучше понять язык, с которым он работает.

Права на издание книги были получены по соглашению с Addison Wesley Longman, Inc. и Литературным агентством Мэтлок (Санкт-Петербург).

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но, поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-0-201-54330-8 (англ.)
ISBN 978-5-97060-419-9 (рус.)

Copyright © by AT&T Bell Labs.
© Перевод на русский язык, оформление.
ДМК Пресс, 2016



Содержание

Предисловие	13
Благодарности	15
Обращение к читателю	16

Часть I

Глава 1. Предыстория C++	30
1.1. Язык Simula и распределенные системы	30
1.2. Язык С и системное программирование	33
1.3. Немного об авторе книги	33
Глава 2. Язык C with Classes	36
2.1. Рождение C with Classes	36
2.2. Обзор языковых возможностей	38
2.3. Классы	39
2.4. Эффективность исполнения	41
2.4.1. Встраивание	42
2.5. Модель компоновки	43
2.5.1. Простые реализации	46
2.5.2. Модель размещения объекта в памяти	48
2.6. Статический контроль типов	49
2.6.1. Сужающие преобразования	50
2.6.2. О пользе предупреждений	51
2.7. Почему C?	52
2.8. Проблемы синтаксиса	54
2.8.1. Синтаксис объявлений в языке C	54
2.8.2. Тэги структур и имена типов	56
2.8.3. Важность синтаксиса	58
2.9. Производные классы	58
2.9.1. Полиморфизм без виртуальных функций	59
2.9.2. Контейнерные классы без шаблонов	60
2.9.3. Модель размещения объекта в памяти	61
2.9.4. Ретроспектива	62
2.10. Модель защиты	62

2.11. Гарантии времени исполнения	65
2.11.1. Конструкторы и деструкторы	65
2.11.2. Распределение памяти и конструкторы	66
2.11.3. Функции call и return	67
2.12. Менее существенные средства	67
2.12.1. Перегрузка оператора присваивания	67
2.12.2. Аргументы по умолчанию	68
2.13. Что не реализовано в C with Classes	69
2.14. Рабочая обстановка	70
Глава 3. Рождение C++	73
3.1. От C with Classes к C++	73
3.2. Цели C++	74
3.3. Компилятор Cfront	76
3.3.1. Генерирование С-кода	77
3.3.2. Синтаксический анализ C++	79
3.3.3. Проблемы компоновки	80
3.3.4. Версии Cfront	80
3.4. Возможности языка	82
3.5. Виртуальные функции	82
3.5.1. Модель размещения объекта в памяти	85
3.5.2. Замещение и поиск подходящей виртуальной функции	87
3.5.3. Скрытие членов базового класса	87
3.6. Перегрузка	88
3.6.1. Основы перегрузки	89
3.6.2. Функции-члены и дружественные функции	91
3.6.3. Операторные функции	93
3.6.4. Перегрузка и эффективность	94
3.6.5. Изменение языка и новые операторы	96
3.7. Ссылки	96
3.7.1. Lvalue и Rvalue	98
3.8. Константы	99
3.9. Управление памятью	101
3.10. Контроль типов	103
3.11. Второстепенные возможности	104
3.11.1. Комментарии	104
3.11.2. Нотация для конструкторов	104
3.11.3. Квалификация	105
3.11.4. Инициализация глобальных объектов	106
3.11.5. Предложения объявления	109
3.12. Языки С и C++	111
3.13. Инструменты для проектирования языка	114

3.14. Книга «Язык программирования C++»	116
3.15. Статья «WhatIs?»	117
Глава 4. Правила проектирования языка C++	120
4.1. Правила и принципы	120
4.2. Общие правила	121
4.3. Правила поддержки проектирования	125
4.4. Технические правила	128
4.5. Правила поддержки низкоуровневого программирования	132
4.6. Заключительное слово	134
Глава 5. Хронология 1985–1993 гг.	135
5.1. Введение	135
5.2. Версия 2.0	136
5.2.1. Обзор возможностей	137
5.3. Аннотированное справочное руководство	138
5.3.1. Обзор ARM	139
5.4. Стандартизация ANSI и ISO	140
5.4.1. Обзор возможностей	143
Глава 6. Стандартизация	144
6.1. Что такое стандарт?	144
6.1.1. Детали реализации	145
6.1.2. Тест на реалистичность	146
6.2. Работа комитета	146
6.2.1. Кто работает в комитете	148
6.3. Как велась работа	148
6.3.1. Разрешение имен	149
6.3.2. Время жизни объектов	153
6.4. Расширения	157
6.4.1. Критерии рассмотрения предложений	159
6.4.2. Текущее состояние дел	161
6.4.3. Проблемы, связанные с полезными расширениями	162
6.4.4. Логическая непротиворечивость	163
6.5. Примеры предлагавшихся расширений	164
6.5.1. Именованные аргументы	164
6.5.2. Ограниченные указатели	168
6.5.3. Наборы символов	169
Глава 7. Заинтересованность и использование	174
7.1. Рост интереса к C++	174
7.1.1. Отсутствие маркетинга C++	175
7.1.2. Конференции	175

7.1.3. Журналы и книги	176
7.1.4. Компиляторы	177
7.1.5. Инструментальные средства и среды программирования	177
7.2. Преподавание и изучение C++	178
7.3. Пользователи и приложения	183
7.3.1. Первые пользователи	183
7.3.2. Сферы применения C++	184
7.4. Коммерческая конкуренция	184
7.4.1. Традиционные языки	185
7.4.2. Современные языки	186
7.4.3. Как выдержать конкуренцию	187
Глава 8. Библиотеки	189
8.1. Введение	189
8.2. Проектирование библиотеки C++	189
8.2.1. Альтернативы при проектировании библиотеки	190
8.2.2. Языковые средства и построение библиотеки	190
8.2.3. Как работать с разнообразными библиотеками	191
8.3. Ранние библиотеки	192
8.3.1. Библиотека потокового ввода/вывода	193
8.3.2. Поддержка параллельности	196
8.4. Другие библиотеки	198
8.4.1. Базовые библиотеки	199
8.4.2. Устойчивость и базы данных	200
8.4.3. Библиотеки для численных расчетов	200
8.4.4. Специализированные библиотеки	201
8.5. Стандартная библиотека	201
Глава 9. Перспективы развития языка C++	203
9.1. Введение	203
9.2. Оценка пройденного пути	203
9.2.1. Достигнуты ли основные цели C++?	204
9.2.2. Является ли C++ логически последовательным языком?	204
9.2.3. Основная недоработка языка	207
9.3. Всего лишь мост?	208
9.3.1. Мост нужен надолго	208
9.3.2. Если C++ – это ответ, то на какой вопрос?	209
9.4. Что может сделать C++ более эффективным	213
9.4.1. Стабильность и стандарты	213
9.4.2. Обучение и приемы	213
9.4.3. Системные вопросы	213
9.4.4. За пределами файлов и синтаксиса	214
9.4.5. Подведение итогов и перспективы	215

Часть II	217
Глава 10. Управление памятью	218
10.1. Введение	218
10.2. Отделение распределения памяти и инициализации	219
10.3. Выделение памяти для массива	220
10.4. Размещение объекта в памяти	221
10.5. Проблемы освобождения памяти	222
10.5.1. Освобождение памяти для массивов	224
10.6. Нехватка памяти	225
10.7. Автоматическая сборка мусора	226
10.7.1. Необязательный сборщик мусора	226
10.7.2. Как должен выглядеть необязательный сборщик мусора?	228
Глава 11. Перегрузка	230
11.1. Введение	230
11.2. Разрешение перегрузки	230
11.2.1. Детальное разрешение	231
11.2.2. Управление неоднозначностью	233
11.2.3. Нулевой указатель	236
11.2.4. Ключевое слово overload	238
11.3. Типобезопасная компоновка	239
11.3.1. Перегрузка и компоновка	239
11.3.2. Реализация компоновки в C++	240
11.3.3. Анализ пройденного пути	241
11.4. Создание и копирование объектов	244
11.4.1. Контроль допустимости копирования	244
11.4.2. Управление распределением памяти	244
11.4.3. Управление наследованием	245
11.4.4. Почленное копирование	246
11.5. Удобство нотации	248
11.5.1. «Умные» указатели	248
11.5.2. «Умные» ссылки	249
11.5.3. Перегрузка операторов инкремента и декремента	252
11.5.4. Перегрузка ->*	254
11.5.5. Перегрузка оператора «запятая»	254
11.6. Добавление в C++ операторов	254
11.6.1. Оператор возведения в степень	254
11.6.2. Операторы, определяемые пользователем	257
11.6.3. Составные операторы	258
11.7. Перечисления	259
11.7.1. Перегрузка на базе перечислений	261
11.7.2. Тип Boolean	261

Глава 12. Множественное наследование	263
12.1. Введение	263
12.2. Базовые классы	264
12.3. Виртуальные базовые классы	265
12.3.1. Виртуальные базовые классы и виртуальные функции	267
12.4. Модель размещения объекта в памяти	270
12.4.1. Размещение в памяти объекта виртуального базового класса	272
12.4.2. Виртуальные базовые классы и приведение типов	273
12.5. Комбинирование методов	274
12.6. Полемика о множественном наследовании	276
12.7. Делегирование	279
12.8. Переименование	280
12.9. Инициализаторы членов и базовых классов	282
Глава 13. Уточнения понятия класса	284
13.1 Введение	284
13.2. Абстрактные классы	284
13.2.1. Абстрактные классы и обработка ошибок	284
13.2.2. Абстрактные типы	286
13.2.3. Синтаксис	288
13.2.4. Виртуальные функции и конструкторы	288
13.3. Константные функции-члены	291
13.3.1. Игнорирование const при приведении типов	291
13.3.2. Уточнение определения const	292
13.3.3. Ключевое слово mutable и приведение типов	293
13.4. Статические функции-члены	294
13.5. Вложенные классы	295
13.6. Ключевое слово inherited	297
13.7. Ослабление правил замещения	299
13.7.1. Ослабление правил аргументов	301
13.8. Мультиметоды	303
13.8.1. Когда нет мультиметодов	305
13.9. Защищенные члены	307
13.10. Улучшенная генерация кода	308
13.11. Указатели на функции-члены	309
Глава 14. Приведение типов	311
14.1. Крупные расширения	311
14.2. Идентификация типа во время исполнения	312
14.2.1. Зачем нужен механизм RTTI	313
14.2.2. Оператор dynamic_cast	313
14.2.3. Правильное и неправильное использование RTTI	319

14.2.4. Зачем давать «опасные средства»	321
14.2.5. Оператор typeid()	322
14.2.6. Модель размещения объекта в памяти	326
14.2.7. Простой ввод/вывод объектов	327
14.2.8. Другие варианты	329
14.3. Новая нотация для приведения типов	333
14.3.1. Недостатки старых приведений типов	334
14.3.2. Оператор static_cast	335
14.3.3. Оператор reinterpret_cast	337
14.3.4. Оператор const_cast	339
14.3.5. Преимущества новых приведений типов	340
Глава 15. Шаблоны	343
15.1. Введение	343
15.2. Зачем нужны шаблоны	344
15.3. Шаблоны классов	346
15.3.1. Аргументы шаблонов, не являющиеся типами	347
15.4. Ограничения на аргументы шаблонов	348
15.4.1. Ограничения за счет наследования	349
15.4.2. Ограничения за счет использования	350
15.5. Устранение дублирования кода	351
15.6. Шаблоны функций	353
15.6.1. Выведение аргументов шаблона функции	354
15.6.2. Задание аргументов шаблона функции	355
15.6.3. Перегрузка шаблона функции	357
15.7. Синтаксис	360
15.8. Методы композиции	361
15.8.1. Представление стратегии реализации	362
15.8.2. Представление отношений порядка	363
15.9. Соотношения между шаблонами классов	365
15.9.1. Отношения наследования	365
15.9.2. Преобразования	367
15.9.3. Шаблоны-члены	368
15.10. Инстанцирование шаблонов	369
15.10.1. Явное инстанцирование	371
15.10.2. Точка инстанцирования	372
15.10.3. Специализация	378
15.10.4. Нахождение определений шаблонов	381
15.11. Последствия введения шаблонов	383
15.11.1. Отделение реализации от интерфейса	384
15.11.2. Гибкость и эффективность	384
15.11.3. Влияние на другие компоненты C++	385

Глава 16. Обработка исключений	387
16.1. Введение	387
16.2. Цели и предположения	388
16.3. Синтаксис	389
16.4. Группировка	390
16.5. Управление ресурсами	391
16.5.1. Ошибки в конструкторах	393
16.6. Возобновление или завершение?	394
16.6.1. Обходные пути для реализации возобновления	397
16.7. Асинхронные события	398
16.8. Распространение на несколько уровней	399
16.9. Статическая проверка	399
16.9.1. Вопросы реализации	401
16.10. Инварианты	402
Глава 17. Пространства имен	403
17.1. Введение	403
17.2. Для чего нужны пространства имен	404
17.2.1. Обходные пути	404
17.3. Какое решение было бы лучшим?	406
17.4. Решение: пространства имен	408
17.4.1. Мнения по поводу пространств имен	410
17.4.2. Внедрение пространств имен	411
17.4.3. Псевдонимы пространства имен	412
17.4.4. Использование пространств имен для управления версиями	413
17.4.5. Технические детали	415
17.5. Классы и пространства имен	421
17.5.1. Производные классы	421
17.5.2. Использование базовых классов	423
17.5.3. Исключение глобальных статических объявлений	424
17.6. Совместимость с C	425
Глава 18. Препроцессор C	427
Алфавитный указатель	431