Миллионам нынешних программистов на C++

Предисловие

Идите проторенной дорогой — делайте одинаковые вещи одинаковыми способами. Накапливайте идиомы. ***Стандартизируйте****. Единственное отличие между вами и Шекспиром — в количестве используемых идиом, а не в размере словаря.*

— Алан Перлис (Alan Perlis) [выделено нами]

Лучшее в стандарте то, что он предоставляет богатый выбор.

— Приписывается разным людям

Мы бы хотели, чтобы эта книга стала основой для стандартов кодирования, используемых вашей командой, по двум основным причинам.

- *Стандарты кодирования должны отражать лучший опыт проб и ошибок всего сообщества программистов*. В них должны содержаться проверенные идиомы, основанные на опыте и твердом понимании языка. В частности, стандарт кодирования должен основываться на исчерпывающем анализе литературы по разработке программного обеспечения, и объединять воедино правила, рекомендации и наилучшие практические решения, которые в противном случае оказываются разбросанными по многочисленным источникам.
- *Природа не терпит пустоты*. Если вы не разработаете набор правил, то это сделает кто-то другой. Такие "самопальные" стандарты, как правило, грешат тем, что включают нежелательные для стандарта требования; например, многие из них, по сути, заставляют программистов использовать C++ просто как улучшенный C.

Множество таких плохих стандартов кодирования разработаны людьми, которые недостаточно хорошо понимают язык программирования C++ или пытаются чрезмерно детализировать его применение. Плохой стандарт кодирования быстро теряет кредит доверия, и в результате несогласие или неприятие программистами части его положений распространяется на весь стандарт целиком, перечеркивая содержащиеся в нем различные положительные советы и рекомендации. И это — в лучшем случае, потому что в худшем случае такой стандарт и его выполнение могут быть навязаны руководством.

Как пользоваться этой книгой

Думать. Надо добросовестно следовать умным советам, но делать это не вслепую. Во многих разделах этой книги есть подраздел "Исключения", в котором приводятся нестандартные, редко встречающиеся ситуации, когда совет из основного раздела может оказаться неприменим. Никакое количество даже самых хороших (мы на это надеемся) советов не могут заменить голову.

Каждая команда разработчиков отвечает за принятие собственных стандартов и несет ответственность за них и за неукоснительное следование им. Ваша команда — не исключение. Если вы — руководитель, предложите членам своей группы поучаствовать в разработке стандартов, которым группа будет следовать в своей работе. Люди всегда охотнее следуют правилам, которые они сами для себя вырабатывают, чем тем, которые им навязаны.

Эта книга предназначена для того, чтобы послужить основой для вашего стандарта кодирования и быть в той или иной мере включенной в него. Это — не ultima ratio в стандартах кодирования, и ваша группа может разработать (или прибавить) свои правила, в наибольшей степени подходящие для вашей конкретной группы или для конкретной решаемой задачи, так что вы не должны быть скованы этой книгой по рукам и ногам. Тем не менее, мы надеемся, что эта книга поможет вам сберечь время и усилия при разработке собственного стандарта кодирования, а также будет способствовать повышению его качества и последовательности.

Ознакомьте членов своей команды с этой книгой, и после того, как они полностью прочтут ее и познакомятся со всеми рекомендациями и их обоснованиями, решите, какие из них подходят вам, а какие в силу особенностей вашего проекта для вас неприменимы. После этого строго придерживайтесь выбранной стратегии. После того как командный стандарт принят, он не должен нарушаться иначе как по согласованному решению всей команды в целом.

И наконец, периодически творчески пересматривайте собственные стандарты с учетом практического опыта, приобретенного всей командой при работе над проектом.

Стандарты кодирования и вы

Хорошие стандарты кодирования могут принести немалую выгоду с различных точек зрения.

- *Повышение качества кода*. Работа в соответствии со стандартом приводит к однотипному решению одинаковых задач, что повышает ясность кода и упрощает его сопровождение.
- *Повышение скорости разработки*. Разработчику не приходится решать все задачи и принимать решения "с нуля".
- *Повышение уровня взаимодействия в команде*. Наличие стандарта позволяет уменьшить разногласия в команде и устранить ненужные дебаты по мелким вопросам, облегчает понимание и поддержку чужого кода членами команды.
- *Согласованность в работе*. При использовании стандарта разработчики направляют свои усилия в верном направлении, на решение действительно важных задач.

В напряженной обстановке, при жестких временны*х* рамках люди обычно делают то, чему их учили, к чему они привыкли. Вот почему в больницах в пунктах первой помощи предпочитают опытных, тренированных сотрудников — даже хорошо обученные и знающие новички склонны к панике.

У разработчиков программного обеспечения регулярно возникают ситуации, когда что-то надо было сделать еще вчера — на позавчера. Когда на нас давит график работ (который к тому же имеет тенденцию сдвигаться в одном направлении, и то, что по плану должно было заработать завтра, от нас начинают требовать еще вчера…), мы работаем так, как приучены. Неряшливые программисты, которые даже при обычной неспешной работе не помнят о правильных принципах разработки программного обеспечения (а то и вовсе не знакомы с ними), при нехватке времени окажутся еще небрежнее, а их код будет изобиловать ошибками. Соответственно, программист, который выработал в себе хорошие привычки и регулярно ими пользуется, при "повышенном давлении" будет продолжать выдавать качественный код.

Стандарты кодирования, приведенные в этой книге, представляют собой набор рекомендаций по написанию высококачественного кода на C++. В них сконцентрирован богатый коллективный опыт всего сообщества программистов на C++. Многие из этих знаний разбросаны по частям по самым разным книгам, но не меньшее количество знаний передается изустно. Мы постарались собрать разрозненные сведения в одной книге в виде коллекции ясных, компактных правил с пояснениями, простых для понимания и следования им.

Конечно, даже самые лучшие стандарты не могут помешать написанию плохого кода. То же можно сказать о любом языке программирования, процессе или методологии. Хороший набор стандартов воспитывает хорошие привычки и дисциплину, превышающую обычные

нормы. Это служит хорошим фундаментом для дальнейшего усовершенствования и повышения квалификации. Это не преувеличение и не красивые слова — перед тем, как начать писать стихи, надо владеть словарным запасом и знать грамматику. Мы надеемся, что наша книга упростит для вас путь к поэзии программирования.

Эта книга предназначена для программистов всех уровней.

Если вы начинающий программист — мы надеемся, что рекомендации и их пояснения достаточно поучительны и помогут вам в понимании того, какие стили и идиомы C++ поддерживает наиболее естественным образом. В описании каждого правила и рекомендации приводится краткое обоснование и обсуждение, чтобы вы не просто механически запомнили правило, а поняли его суть.

Для программистов среднего и высокого уровня при описании каждого правила приводится список ссылок, который позволит вам углубленно изучить заинтересовавший вас вопрос, проведя поиск корней правила в системе типов, грамматике и объектной модели C++.

Каким бы ни был ваш уровень как программиста — вероятно, вы работаете над сложным проектом не в одиночку, а в команде. Именно в этом случае разработка стандартов кодирования окупается сполна. Вы можете использовать их для того, чтобы подтянуть всю свою команду к одному, более высокому уровню, и обеспечить повышение качества разрабатываемого кода.

Об этой книге

Основными принципами дизайна данной книги являются следующие.

- *Краткость — сестра таланта*. Чем больше стандарт кодирования по размеру, тем больше шансов, что он будет благополучно проигнорирован. Читают и используют обычно короткие стандарты. Длинные разделы, как правило, просто просматривают "по диагонали", короткие статьи обычно удостаиваются внимательного прочтения.
- *Никакой материал не должен вызывать дискуссий*. Эта книга документирует широко используемые стандарты, а не изобретает их. Если некоторая рекомендация не применима во всех ситуациях, то мы так и пишем — "подумайте о применении X" вместо "делайте X". Кроме того, к каждому правилу указаны все общепринятые исключения.
- *Весь материал должен быть обоснован*. Все рекомендации в этой книге взяты из существующих печатных работ. В конце книги приведен список использованной литературы по C++.
- *Материал не должен быть банален*. Мы не даем рекомендации, которым вы и так следуете, которые обеспечиваются компилятором или которые уже изложены в других разделах.
    - Например, "не возвращайте указатель/ссылку на локальную переменную" — хороший совет, но он не включен в данную книгу, поскольку практически все компиляторы выдают соответствующее предупреждение, к тому же этот вопрос раскрывается в первом разделе книги.
    - Рекомендация "используйте редактор (компилятор, отладчик)" — тоже хороший совет, но, конечно, вы используете эти инструменты и без нашего напоминания. Вместо этого мы рекомендуем использовать автоматизированные системы сборки программ и системы управления версиями.
    - Еще один совет — "не злоупотребляйте `goto`" — исходя из нашего опыта, и так широко известен всем программистам, так что нет смысла повторяться.

Каждый раздел состоит из следующих частей.

- *Заглавие*. Краткое название раздела, поясняющее, о чем будет идти речь.
- *Резюме*. Краткое изложение сути вопроса.

- *Обсуждение*. Расширенное пояснение рекомендации. Зачастую включает краткое обоснование, но учтите, что полную информацию по данному вопросу следует искать в приведенных ссылках.
- *Примеры* (если таковые имеются). Примеры, демонстрирующие правило или позволяющие лучше его понять и запомнить.
- *Исключения* (если таковые имеются). Описание ситуаций (обычно редких), когда приведенное правило неприменимо. Однако остерегайтесь попасть в ловушку, думая, что ваш случай особый и что в вашей ситуации это правило неприменимо, — обычно при здравом размышлении оказывается, что ничего особого в вашей ситуации нет и описанное правило может быть с успехом вами применено.
- *Ссылки*. В приведенной в этом подразделе литературе по C++ вы найдете более полный анализ рассматриваемого в разделе вопроса.

В каждой части книги имеется "наиболее важный раздел" — обычно это первый раздел части. Однако иногда это правило нарушалось в связи с необходимостью последовательного и связного изложения материала.

Благодарности

Мы от всей души благодарны редактору серии Бьярну Страуструпу (Bjarne Stroustrup), редакторам Питеру Гордону (Peter Gordon) и Дебби Лафферти (Debbie Lafferty), а также Тирреллу Албаху (Tyrrell Albaugh), Ким Бодихаймер (Kim Boedigheimer), Джону Фуллеру (John Fuller), Бернарду Гаффни (Bernard Gaffney), Курту Джонсону (Curt Johnson), Чанде Лири-Коту (Chanda Leary-Coutu), Чарли Ледди (Charles Leddy), Хитеру Муллэйн (Heather Mullane), Чути Прасертсиху (Chuti Prasertsith), Ларе Вайсонг (Lara Wysong) и всем остальным работникам издательства Addison-Wesley, помогавшим нам в нашей работе над этим проектом. Нам было очень приятно работать с ними.

На идею и оформление книги нас натолкнули несколько источников, включая такие, как [Cline99], [Peters99], а также работы легендарного и широко цитируемого Алана Перлиса (Alan Perlis).

Особая благодарность тем людям, чьи отзывы помогли нам сделать многие части книги намного лучшими, чем они были бы без этих замечаний. Особенно большое влияние на книгу оказали острые комментарии Бьярна Страуструпа. Мы очень хотим поблагодарить за активное участие в обсуждении материала и высказанные замечания таких людей, как Дэйв Абрамс (Dave Abrahams), Маршалл Клайн (Marshall Cline), Кевлин Хенни (Kevlin Henney), Говард Хиннант (Howard Hinnant), Джим Хайслоп (Jim Hyslop), Николаи Джосаттис (Nicolai Josuttis), Йон Калб (Jon Kalb), Макс Хесин (Max Khesin), Стен Липпман (Stan Lippman), Скотт Мейерс (Scott Meyers) и Дэвид Вандевурд (Daveed Vandevoorde). Кроме того, отдельное спасибо хотелось бы сказать Чаку Аллисону (Chuck Allison), Самиру Байяю (Samir Bajaj), Марку Барбуру (Marc Barbour), Тревису Брауну (Travis Brown), Нилу Кумбесу (Nil Coombes), Дамиану Дечеву (Damian Dechev), Стиву Дьюхарсту (Steve Dewhurst), Питеру Димову (Peter Dimov), Аттиле Фехеру (Attila Feher), Алану Гриффитсу (Alan Griffiths), Мичи Хеннингу (Michi Henning), Джеймсу Канзе (James Kanze), Бартошу Милевски (Bartosz Milewski), Мэтту Маркусу (Matt Marcus), Балогу Палу (Balog Pal), Владимиру Прусу (Vladimir Prus), Дэну Саксу (Dan Saks), Люку Вагнеру (Luke Wagner), Мэтью Вильсону (Matthew Wilson) и Леору Золману (Leor Zolman).

Как обычно, все оставшиеся в книге ошибки и недосмотры — на нашей совести, а не на их.

Герб Саттер (Herb Sutter)
Андрей Александреску (Andrei Alexandrescu)

Сиэтл, сентябрь 2004

Вопросы организации и стратегии

Если бы строители строили здания так же, как программисты пишут программы, — то первый же залетевший дятел разрушил бы всю цивилизацию.

— Джеральд Вайнберг (Gerald Weinberg)

Следуя великой традиции C и C++, мы начинаем отсчет с нуля. Главный совет — под номером 0 — говорит о том, что основной советчик по поводу стандартов кодирования — наши чувства и ощущения.

Остальная часть этой главы состоит из небольшого количества тщательно отобранных вопросов, которые не имеют прямого отношения к коду и посвящены важным инструментам и методам написания надежного кода.

В этом разделе книги наиболее важной мы считаем нулевую рекомендацию — “Не мелочитесь, или Что не следует стандартизировать”.