Введение

Термин *адаптивный код* как нельзя лучше обозначает последствия применения принципов разработки программного обеспечения, рассматриваемых в этой книге, т.е. способность кода адаптироваться к любым новым требованиям или непредвиденным обстоятельствам без существенной его переделки. Назначение этой книги — соединить в одном издании многие передовые методики программирования на C# для разработки приложений на платформе .NET Framework корпорации Microsoft. И хотя этой же теме посвящено немало другой литературы, основное внимание в ней уделяется теории, а если и практике, то не разработке приложений на платформе .NET Framework.

Процесс программирования может быть долгим. И если код адаптивный, то вносить изменения в него можно быстрее, легче и с меньшими ошибками, чем в работе с кодовой базой, затрудняющей внесение изменений. Как известно всем разработчикам, технические требования претерпевают изменения. Контроль над изменениями служит главным фактором, отличающим удачные проекты разработки программного обеспечения от неудачных, исчерпавшихся из-за расползания их рамок до неконтролируемых размеров. Разработчики могут по-разному реагировать на изменения технических требований, придерживаясь двух противоположных точек зрения на истину, которая обычно лежит посредине.

С одной стороны, разработчики могут придерживаться жесткой точки зрения. При таком подходе к процессу разработки (вплоть до конструирования класса) проект становится негибким, как в те далекие времена, когда программисты набивали код программ на перфокартах. Водопадные методики разработки служат явными препятствиями к свободному внесению изменений в программное обеспечение. Раз и навсегда установленные ими стадии анализа, разработки, реализации и тестирования затрудняют или, по крайней мере, делают слишком дорогостоящими изменения технических требований после начала стадии реализации проекта. В таком случае код не нужно создавать с учетом возможных изменений, поскольку они запрещаются в самом процессе разработки.

С другой стороны, разработчики могут выбрать гибкую методику, которая служит не альтернативой, а, скорее, реакцией на жесткие методики разработки. Гибкие процессы призваны охватить изменения как необходимую часть контракта между заказчиком и исполнителем. Если заказчикам потребуется изменить что-нибудь в программном продукте, за который они платят, затраты времени или финансов должны быть увязаны с объемом вносимых изменений, а не с текущей стадией процесса разработки.

В отличие от проектирования осязаемых объектов, разработка программного обеспечения ведется весьма гибким инструментом: исходным кодом. Кирпичи и раствор буквально соединяются в единую кладку по ходу строительства дома. Расходы на изменение конструкции дома должны быть связаны с завершением стадии его

строительства. Если строительный проект еще не начат, а находится на стадии вычерчивания чертежей, то внесение изменений обойдется относительно недорого. А если окна уже вставлены, электропроводка проведена и канализация подключена, то перенести спальню из верхнего этажа поближе к кухне будет недопустимо дорого. На первый взгляд, функции прикладного кода намного проще перенести из одного места в другое или переделать пользовательский интерфейс, но, к сожалению, это возможно далеко не всегда. Одни только затраты времени препятствуют подобным изменениям. А происходит это, главным образом, из-за того, что прикладному коду явно недостает адаптируемости.

В этой книге демонстрируется второй подход к разработке программного обеспечения. На реальных примерах в ней поясняется, каким образом адаптивный код реализуется на практике.

Кому адресована книга

Эта книга преследует цель восполнить пробел между теорией и практикой. Она рассчитана на опытных программистов, которым требуются практические примеры применения проектных шаблонов, принципов SOLID, модульного тестирования, реорганизации кода и прочих передовых методик разработки программного обеспечения.

Способные программисты среднего уровня, которым требуется восполнить пробел в знаниях и у которых имеются сомнения и вопросы по поводу совместного применения передовых методик разработки программного обеспечения, смогут извлечь наибольшую пользу из этой книги, особенно потому, что простые примеры и теория редко совпадают с повседневной практикой программирования. Бо́льшая часть принципов SOLID теперь понятна многим, но особые трудности вызывает соблюдение принципов открытости-закрытости и подстановки Лисков, рассматриваемых в главах 6 и 7 соответственно. Даже опытные программисты порой не полностью осознают выгоды, которые сулит принцип внесения зависимостей, обсуждаемый в главе 9. Аналогично из виду нередко упускается гибкость и приспособляемость интерфейсов благодаря адаптивному коду, как поясняется в главе 3.

Менее опытным разработчикам эта книга поможет сразу разобраться в благоприятных и неблагоприятных в долгосрочной перспективе особенностях общих шаблонов и методик проектирования. У кода из примеров, рассматриваемых в этой книге с точки зрения их возможного практического применения, много общего. У этого кода есть почти все, что требуется для его нормальной эксплуатации, — остается лишь направить его в нужное русло, чтобы он стал много лучше. В частности, антишаблоны “Антураж” (Entourage; см. главу 2) и “Определитель служб” (Service Locator; см. главу 9) преобладают в примерах кода из этой книги, где также представлены практические альтернативы этим антишаблонам и доводы в пользу их применения.

Предположения автора

В идеальном случае у читателя этой книги должен быть некоторый практический опыт программирования на языке, синтаксически сходном с языком C#, например

Java или C++. У него должно быть также твердое знание основных понятий процедурного программирования, в том числе условных переходов, циклов и выражений, а также принципов объектно-ориентированного программирования, включая умение пользоваться классами и хотя бы беглое знакомство с интерфейсами.

На кого не рассчитана эта книга

Эта книга не подойдет тем, кто только начинает изучать программирование. В ней рассматриваются расширенные возможности и специальные понятия, требующие твердых знаний основополагающих принципов программирования.

Структура книги

Книга состоит из трех частей, причем материал каждой последующей части основывается на материале предыдущей. Впрочем, эту книгу можно читать в любом порядке. Каждая ее глава посвящена отдельной подробно рассматриваемой теме со ссылками на другие главы там, где это уместно.

Часть I. Основы гибкой разработки

В этой части закладываются основы разработки программного обеспечения адаптивным способом. В ней рассматривается высокоуровневый процесс, называемый *Scrum* (Толкучка) и требующий от прикладного кода приспособляемости к изменениям. В отдельных главах этой части основное внимание уделяется интерфейсам, проектным шаблонам, реорганизации и модульному тестированию прикладного кода.

- **Глава 1. Введение в Scrum.** В этой главе подготавливается почва для материала всей книги введением в Scrum — основную методику управления гибкой разработкой проекта. В ней дан углубленный обзор артефактов, ролей, количественных показателей и стадий выполнения проекта по методике Scrum. И наконец, здесь будет показано, каким образом разработчики должны организовать себя и свой код, работая в среде гибкой разработки.
- **Глава 2. Зависимости и разделение на уровни.** В этой главе рассматриваются зависимости и разделение на архитектурные уровни. Код может быть адаптивным только в том случае, если это допускает структура проекта. В этой главе описываются разные виды зависимостей: основных, сторонних, каркасных, а также порядок управления и организации зависимостей от антишаблонов, которых следует избегать, до шаблонов, которые следует только приветствовать. В ней также вводятся такие специальные понятия, как аспектно-ориентированное программирование и асимметричное разделение на уровни, обеспечивающее дополнительную глубину структуры проекта.
- **Глава 3. Интерфейсы и проектные шаблоны.** Интерфейсы теперь широко применяются в разработке приложений на платформе .NET. Но ими нередко злоупотребляют и пользуются неуместно, не зная их истинного назначения.

В этой главе демонстрируются некоторые из наиболее общих и практически полезных шаблонов, а также исследуются универсальные возможности интерфейсов. Выходя за рамки обычного извлечения интерфейса, в этой главе показаны разные способы тщательной разработки интерфейсов для решения конкретных задач. Примеси, утипизация и текучесть интерфейсов дополнительно подчеркивают универсальность этого главного оружия из арсенала средств разработчиков.

- **Глава 4. Модульное тестирование и реорганизация кода.** К числу обязательных практических навыков разработчиков все чаще относится умение выполнять модульное тестирование и реорганизацию кода. Обе эти методики тесно связаны и применяются совместно для получения адаптивного кода. Без страховочной сети из модульных тестов реорганизация кода чревата ошибками, а без реорганизации код становится громоздким, жестким и трудно понимаемым. В этой главе приведен пример модульного тестирования, начиная с самых азов и продолжая внедрением более развитых, но практичных шаблонов и методик вроде текучих утверждений, разработки посредством тестирования и имитации. А реорганизация кода демонстрируется в этой главе на реальных примерах повышения удобочитаемости и сопровождаемости исходного кода.

Часть II. Написание кода по принципам SOLID

Материал данной части опирается на основы, заложенные в части I. Каждая глава этой части посвящена одному из принципов SOLID. Основное внимание в главах уделяется практическим примерам реализации принципов SOLID, а не теоретическим основаниям для их применения. Каждый пример в главах из этой части перенесен в реальный контекст, что позволяет ясно продемонстрировать полезность принципов SOLID.

- **Глава 5. Принцип единственной ответственности.** В этой главе показано, каким образом принцип единственной ответственности реализуется на практике с помощью шаблонов “Декоратор” (Decorator) и “Адаптер” (Adapter). В результате применения данного принципа увеличивается количество классов и уменьшаются их размеры. В главе также показано, что эти мелкие классы, в отличие от монолитных классов с обширным рядом предоставляемых средств, в большей степени сосредоточены и направлены на решение только небольшой части крупной задачи. А в общей совокупности эти классы образуют нечто большее, чем сумму своих частей.
- **Глава 6. Принцип открытости-закрытости.** Этот принцип лишь формулируется, но может оказывать значительное влияние на прикладной код. Он отвечает за то, что прикладной код, следующий принципам SOLID, только присоединяется, но вообще не редактируется. В главе обсуждается также понятие предсказуемого изменения по отношению к принципу открытости-закрытости и показано, каким образом оно может помочь разработчикам выявить точки расширения прикладного кода, чтобы сделать его еще более адаптивным.

- **Глава 7. Принцип подстановки Лисков.** В этой главе демонстрируются те положительные последствия, которые дает применение принципа подстановки Лисков в прикладном коде, в особенности тот факт, что рекомендации помогают соблюсти принцип открытости-закрытости и предотвратить последствия неумышленных изменений. Контракты на основе исходных и выходных условий и инвариантов данных описываются с применением инструментальных средств, доступных в библиотеке Code Contracts. В главе рассматриваются также правила выделения подтипов, включая ковариантность, контравариантность и инвариантность, наряду с отрицательным влиянием нарушения этих правил.
- **Глава 8. Принцип разделения интерфейса.** В этой главе показано, что мелкими должны быть не только классы, но и интерфейсы, которые зачастую оказываются слишком крупными. Принцип разделения интерфейса очень просто реализуется на практике, но нередко упускается из виду, поэтому здесь будут описаны преимущества, которые дает ограничение интерфейсов как можно меньшими размерами наряду с выгодами от обращения с мелкими интерфейсами. Кроме того, здесь также исследуются различные побудительные причины для разделения интерфейсов, например, потребности клиентов или архитектурные потребности.
- **Глава 9. Внедрение зависимостей.** Материал этой главы служит связующим звеном для остальных глав. Без внедрения зависимостей мало что оказывается возможным, и именно поэтому данный принцип так важен. Данная глава служит введением в принцип внедрения зависимостей и сравнение разных способов его реализации. В ней также обсуждаются вопросы управления сроком действия объектов, обращения с контейнерами инверсии управления, исключения антишаблонов, связанных с определением служб, а также выявления корней композиции и разрешения.

Часть III. Пример адаптивной разработки приложения

В этой части на примере разработки приложения связываются все остальные части данной книги. И хотя главы этой части содержат немало исходного кода, он снабжается подробными пояснениями. Данная книга посвящена среде гибкой разработки, и поэтому в главах этой части рассматриваются спринты по методике Scrum и все виды работ, вытекающие из элементов задела и изменений в запросах клиентов.

- **Глава 10. Пример адаптивной разработки приложения: введение.** В этой главе описывается пример приложения для интерактивной переписки в оперативном режиме, разрабатываемого на платформе ASP.NET MVC 5. В качестве руководства к действию предоставляется краткое описание планируемой архитектуры и дополнительное пояснение ее особенностей в заделе.
- **Глава 11. Пример адаптивной разработки приложения: спринт 1.** В этой главе рассматривается стадия проектирования первых функциональных возможностей приложения по методике разработки посредством тестирования, включая просмотр, создание дискуссионных команд и составление сообщений.

- **Глава 12. Пример адаптивной разработки приложения: спринт 2.** В этой главе обсуждаются неизбежные изменения, вносимые в требования клиента к разрабатываемому приложению, а также приспособляемость команды разработчиков к этим изменениям через адаптивный код.

Приложения

В приложениях приведен справочный материал, полезный для работы с системой Git для контроля версий исходного кода и поясняющий порядок организации исходного кода к данной книге в информационном хранилище GitHub.

- **Приложение А. Инструментальные средства адаптивной разработки.** В этом приложение дается очень краткое введение в систему Git для контроля версий исходного кода, чтобы загружать его из информационного хранилища GitHub и компилировать в ИСР Visual Studio 2013 корпорации Microsoft. Оно не претендует на роль исчерпывающего руководства по системе Git, поэтому рекомендуем другие отличные источники, в том числе официальную документацию, доступную по адресу `http://git-scm.com/docs/gittutorial`. Эти источники нетрудно найти в Интернете. В этом приложении рассматриваются также другие инструментальные средства разработки, в том числе среда непрерывной интеграции и разработки.
- **Приложение Б. Примеры исходного кода из хранилища GitHub.** Размещение исходного кода примеров к данной книге в информационном хранилище GitHub позволяет вносить в него изменения в одном центральном месте. Это хранилище доступно только для чтения, но в приложениях А и Б поясняется, как найти исходный код для просмотра, загрузки, компиляции, выполнения и внесения локальных изменений. Если вы обнаружите в нем изъяны или желаете предложить свои изменения в нем, разместите свой запрос в главном информационном хранилище AdaptiveCode, чтобы автор с удовольствием рассмотрел его.

Условные обозначения, принятые в книге

В этой книге принят ряд условных обозначений, которые в основном носят стандартный для изданий характер, но все же требуют некоторых пояснений.

Листинги исходного кода

Включаются в текст книги там, где это уместно. Как правило, они сопровождаются заголовками, как показано ниже.

**Листинг 1.1. Это пример листинга исходного кода.
Таких листингов немало в данной книге.**

```
public void MyService : Iservice
{
}
```

Там, где требуется привлечь внимание к отдельным частям исходного кода, например, к изменениям, внесенным в предыдущий пример кода, соответствующие строки кода выделяются полужирным шрифтом.

Примечания, предупреждения и врезки

Примечания и предупреждения обозначены соответствующими пиктограммами на полях книги, а во врезках выделены их текст и более пространные описания, как показано в следующих примерах:

Иллюстрации

Иногда текстового пояснения, каким бы пространным оно ни было, оказывается недостаточно. В подобных случаях на помощь приходят иллюстрации. Все иллюстрации к данной книге были созданы в ИСР Microsoft Visio 2013 без помощи тем оформления, чтобы сделать их высококонтрастными и сосредоточить основное внимание только на экспозиции. А моментальные снимки экрана были сделаны с применением темы оформления с высокой контрастностью.

На заметку

Это примечание. Оно содержит краткое описание, связанное с основным содержимым, но имеет дополнительное значение.

Это врезка

Несмотря на необходимую краткость врезок, они обычно служат для дополнительного обсуждения вопросов, касающихся главной темы.

Системные требования

Чтобы воспользоваться примерами кода из данной книги, вам потребуются следующие аппаратные и программные средства.

- Версия ОС Windows XP Service Pack 3 (кроме функционально ограниченного выпуска Starter Edition), Windows Vista Service Pack 2 (кроме функционально ограниченного выпуска Starter Edition), Windows 7, Windows Server 2003 Service Pack 2, Windows Server 2003 R2, Windows Server 2008 Service Pack 2 или Windows Server 2008 R2.
- Интегрированная среда разработки (ИСР) Visual Studio 2013 любого выпуска (если используются программные продукты типа Express Edition, то потребуются неоднократные загрузки).

- Сервер базы данных Microsoft SQL Server 2008 Express Edition, начиная с выпуска 2008 или R2, а также SQL Server Management Studio 2008 Express, включая ИСР Visual Studio (если используются программные продукты типа Express Edition, то потребуются неоднократные загрузки).
- ПК с ЦП на 1,6 ГГц или более быстродействующим (рекомендуется на 2 ГГц).
- Оперативная память объемом 1 Гбайт (для 32-разрядных систем) или 2 Гбайт (для 64-разрядных систем) плюс 512 Мбайт, если применяется виртуальная машина или версии SQL Server Express Editions, а для расширенных версий SQL Server потребуется еще больше оперативной памяти.
- Свободное место на жестком диске до 3,5 Гбайт.
- Накопитель на жестких дисках со скоростью вращения 5400 об/мин.
- Видеоадаптер, поддерживающий технологию DirectX 9 и разрешение изображения 1024×768.
- Накопитель на DVD, если ИСР устанавливается с DVD.
- Подключение к Интернету для загрузки программного обеспечения и примеров исходного кода.

Для установки или настройки программных продуктов Visual Studio 2013 и SQL Server 2008 могут потребоваться полномочия администратора, хотя это зависит от конкретной конфигурации ОС Windows.

Загрузка примеров исходного кода

Автор книги постарался, чтобы исходный код из листингов, составляющих более крупный пример, можно было выполнять как автономное приложение или модульный тест. Средствами MSTest было написано немало простых модульных тестов, чтобы не потребовался внешний модуль выполнения тестов. А более сложные тесты были написаны средствами NUnit. Весь исходный код был написан в интегрированной среде разработки (ИСР) Visual Studio 2013 Ultimate. И хотя отдельные фрагменты исходного кода были написаны в версии для предварительного просмотра, весь код был скомпилирован и проверен в полной версии. При написании исходного кода средства, недоступные в выпусках Express Editions ИСР Visual Studio 2013, не применялись там, где это было возможно, хотя сделать это удалось не везде. Поэтому читателям, желающим выполнить этот исходный код, придется установить платные версии ИСР Visual Studio 2013.

Весь исходный код, прилагаемый к данной книге, доступен по адресу https://github.com/garymcleanhall/AdaptiveCode. В приложении А вкратце поясняется, как пользоваться системой Git, а в приложении Б — каким образом организован код в информационном хранилище AdaptiveCode. Если читатели желают сделать комментарии к исходному коду, они могут направить их в блог WordPress автора книги по адресу `http://garymcleanhall.wordpress.com`.

Благодарности

Я не смог бы написать ни строчки без перечисленных ниже людей, оказавших мне так или иначе помощь в работе над книгой, за что я им искренне благодарен.

Моей жене Виктории — за возможность написать эту книгу. И это не пустые слова, а истинная правда.

Моей дочери Амелии — за идеальность во всем.

Моей матери Памеле — за вычитку текста книги и замечательные слова поддержки.

Моему отцу Лесли — за все его тяжкие труды.

Моему брату Даррину — за ценные указания и постоянную опеку.

Кэти Краузе (Kathy Krause) из компании Online Training Solutions, Inc. — за отличную работу по повышению удобочитаемости книги.

Девону Масгрэйву (Devon Musgrave) — за бесконечное терпение.

Ждем ваших отзывов!

Вы, читатель этой книги, и есть главный ее критик. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать лучше и что еще вы хотели бы увидеть изданным нами. Нам интересны любые ваши замечания в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо либо просто посетить наш веб-сайт и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится ли вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Отправляя письмо или сообщение, не забудьте указать название книги и ее авторов, а также свой обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию новых книг.

Наши электронные адреса:

E-mail: `info@williamspublishing.com`
WWW: `http://www.williamspublishing.com`

Наши почтовые адреса:

в России: 127055, г. Москва, ул. Лесная, д. 43, стр. 1
в Украине: 03150, Киев, а/я 152