BookStore

Search string

***Currency BUN Basket***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Author | Book title | The year of publishing | Price | Format |  |  |
| *Джеффри Рихтер* | CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. — 4-е изд. — СПб.: [Питер](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_%28%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE%29), 2013. — 896 с. — [ISBN 978-5-496-00433-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785496004336). | 2013 | 74,61 | paper | ***Preview*** | ***Buy*** |
| *Steve McConnell (Стив МакКоннел)* | Совершенный код. Мастер-класс / Пер. с англ. — М. : Издательско-торговый дом «Русская Редакция» ; СПб.: Питер, 2005. — 896 стр.: ил. [ISBN 5-7502-0064-7](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5750200647) [ISBN 5-469-00822-3](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5469008223) | 2005 | 38,66 | Pdf | ***Preview*** | ***Buy*** |
| *Jeffrey Richter* | *CLR via C#,* Fourth Edition  <https://en.wikipedia.org/wiki/Jeffrey_Richter> | 2012 | 174,61 | paper | ***Preview*** | ***Buy*** |

Preview

*Author:* *Джеффри Рихтер*

*Book title:* CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. — 4-е изд. — СПб.: [Питер](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_%28%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE%29), 2013. — 896 с. — [ISBN 978-5-496-00433-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785496004336).

*Content:*

**Запуск уборки мусора**

Как вы уже знаете, CLR запускает уборку мусора, когда обнаруживает, что объем поколения 0 достиг своего порогового значения. Это самая распространенная причина запуска уборки мусора, однако есть и другие:

* Вызов статического метода Collect объекта System.GC. Код явно указывает, в какой момент должна быть выполнена уборка мусора. Хотя Microsoft решительно не рекомендует использовать этот метод, иногда принудительная уборка мусора в приложении может быть оправдана. Этот способ рассматривается позднее в этой главе.
* Windows сообщает о нехватке памяти. CLR использует функции Win32 CreateMemoryResourceNotification и QueryMemoryResourceNotification для контроля состояния памяти системы. Если Windows сообщает о недостаточном объеме свободной памяти, CLR запускает уборку мусора, чтобы избавиться от неиспользуемых объектов и сократить размер рабочего набора процесса.
* Выгрузка домена приложения. При выгрузке домена приложения CLR выполняет полную уборку мусора для всех поколений. Домены приложений рассматриваются в главе 22.
* Завершение работы CLR. CLR завершает работу при нормальном завершении процесса (по сравнению, например, с внешним завершением работы из Диспетчера задач). Во время заверения CLR считает, что в процессе нет корневых ссылок; объектам предоставляется возможность выполнить очистку, но CLR не пытается дефрагментировать или освобождать память, потому что после завершения всего процесса Windows автоматически освобождает всю его память.

Basket

***Currency BUN***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Author | Book title | The year of publishing | Price | Count |  |
| *Джеффри Рихтер* | CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. — 4-е изд. — СПб.: [Питер](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_%28%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE%29), 2013. — 896 с. — [ISBN 978-5-496-00433-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785496004336). | 2013 | 74,61 | ***2*** | ***Удалить*** |
| *Jeffrey Richter* | *CLR via C#,* Fourth Edition  <https://en.wikipedia.org/wiki/Jeffrey_Richter> | 2012 | 174,61 | ***1*** | ***Удалить*** |

Apply promo code

Promo code

Amount: 323.83

Иванов Иван Иванович

Name of contact person

8 029 333-33-33

Phone

Брест

Address

Confirm order

Backend

## Модель данных для работы с хранилищем данных

**class BookData** – тип данных книга

***{***

***int Id***

***IList<NameBooksTranslate> NameBooksTranslates*** – названия книги на разных языках

***dateTime Year*** – год издания

***IList<FormatBookData> FormatBook –*** формат книги

***IList<BooksStorage> BooksStorages*** – список сопоставлений

***IList<AuthorData> Authors*** – авторы книги

***IList<PricePolicyData> PricePolicy –*** данные по цене и количеству книг

***IList<PreviewData> Preview*** – данные по фрагменту книги для показа

***}***

**class BooksStorage** – сопоставление складов и книг

***{***

***int Id***

***int* StorageDataId *-*** FK на таблицу склад

***StorageData Storage*** – название склада

***int BookId*** ***-*** FK на таблицу книг

***BookData Book*** – название книги

***int Count*** – количество экземпляров книги в наличии

***int CountInBlocked*** – количество экземпляров книги заблокированных для доставки

***}***

**class StorageData** – склад

***{***

***int Id***

***string NameStorage –*** название хранилища

***IList<BooksStorage> BooksStorages*** – список сопоставлений

***}***

***class FormatBookData***

***{***

***int Id***

***string FormatName –*** формат данных книги(paper, pdf, fb2, doc, rtf, txt)

***int BookDataId -*** FK на таблицу книг

***BookData Book*** – книга

***}***

**class NameBooksTranslate** – язык наименования книги

***{***

***int Id***

***int BookDataId -*** FK на таблицу книг

***BookData Book*** – книга

***string NameBook –*** название книги

***int LanguageDataId -*** FK на таблицу языков

***LanguageData Language –*** язык наименования

***}***

**class *AuthorData*** – тип данных автор

***{***

***int Id***

***string Name*** – наименование автора

***string Year*** – годы жизни

***string Info –*** краткая информация об авторе

***IList<BookData> Books*** – книги автора

***}***

**class *PricePolicyData*** – тип ценовая политика для разных стран

***{***

***int Id***

***int*** ***BookDataId -*** FK на таблицу книг

***BookData Book*** – данные по книге для которой применена ценовая политика

***decimal Price*** – цена книги

***int CurrencyDataId -*** FK на таблицу валют

***CurrencyData Currency –*** валюта для цены

***int CountryDataId -*** FK на таблицу стран

***CountryData Country*** – страна для которой применена ценовая политика

***}***

***class PreviewData***

***{***

***int Id***

***FormatPreviewData Format –*** формат данных для превью

***string Path –*** путь к файлу в котором храниться информация по превью

***string Data*** – если поле пути пустое – отображаются данные этого поля

***int*** ***BookDataId -*** FK на таблицу книг

***BookData Book*** – данные по книге для которой применена ценовая политика

***}***

***class FormatPreviewData***

***{***

***int Id***

***string FormatName –*** формат данных для превью(pdf, jpg, xml)

***}***

**class *PurchaseData*** – тип данных покупка(подразумевается покупка только одной книги)

***{***

***int Id***

***dateTime Date*** дата покупки

***int BuyerId*** – FK на таблицу Buyer

***BuyerData*** ***Buyer –*** данные покупателя

***int BookId*** – FK на таблицу Book

***BookData Book*** – данные по книге

***int OrderDataId*** – FK на таблицу OrderData

***OrderData*** ***Order*** – данные по заказу

***int Count*** – количество купленных/покупаемых экземпляров книги

***decimal Amount*** – сумма покупки

***int CurrencyDataId*** – FK на таблицу CurrencyData

***CurrencyData Currency –*** валюта для суммы

***bool IsGetMoney*** – флаг получения оплаты

***bool IsTransferComplete*** – флаг передачи книги покупателю

***}***

**class *BuyerData*** – тип данных покупатель

***{***

***int Id***

***string FullName*** – полное имя покупателя

***string Phone*** – телефон покупателя

***IList<PurchaseData> ListPurchases*** – список покупок пользователя завершенных и нет

***IList<OrderData> ListOrders*** – список заказов пользователя

***IList<BuyerAddressData>*** ***BuyerAddress*** – адрес покупателя (может содержать самые разные данные типа почтового индекса, адреса) на основании поля возможно указать место для доставки

***}***

**class *BuyerAddressData*** – тип адрес покупателя

***{***

***int Id***

***int*** ***BuyerDataId -*** FK на таблицу покупателей

***BuyerData*** ***Buyer –*** данные покупателя

***FormatAdressBuyerData FormatAdressBuyer*** – формат адреса покупателя

***string Adress –*** значение адреса

***}***

**class *FormatAdressBuyerData*** – тип формат адреса покупателя

***{***

***int Id***

***string FormatAdressName –*** формат адреса(физический, электронный)

***}***

**class *OrderData*** – тип заказ

***{***

***int Id***

***dateTime Date*** дата заказа

***int*** ***BuyerDataId -*** FK на таблицу покупателей

***BuyerData*** ***Buyer –*** данные покупателя

***IList<PurchaseData> ListPurchases*** – список покупок в заказе

***}***

**class *ExchangeRatesData*** – тип обменный курс

***{***

***int Id***

***dateTime Date*** – дата, время задания курса конверсии

***decimal Rate*** – курс конверсии

***int CurrencyDataFromId*** – FK на таблицу валют

[ForeignKey("CurrencyDataFromId")]

***CurrencyData CurrencyFrom –*** валюта «Из»

***int CurrencyDataToId*** – FK на таблицу CurrencyData

[ForeignKey("CurrencyDataToId")]

***CurrencyData CurrencyTo –*** валюта «В»

***}***

***class CurrencyData***

***{***

***int Id***

***string CurrencyCode –*** строковыйкод валюты по ISO

***}***

***class CountryData***

***{***

***int Id***

***string CountryCode*** – код страны по спецификации ISO

***}***

***class LanguageData***

***{***

***int Id***

***string LanguageCode*** – код языка по спецификации ISO

***}***

***class PromocodeData***

***{***

***int Id***

***string Code*** – значение для сравнения

***dateTime Date*** – дата, время задания значения(учитывается последнее значение)

***}***

## Модель данных для работы с UI

***Описание***

На форме «Книжный магазин» отображается список сущностей Book и возможность выбора валюты для показа цен

На форме «Просмотр» часть сущности Book и отрывок подгруженный из файла по указанному пути

На форме «Корзина» отображается список сущностей **PurchaseUi** для темпового покупателя **BuyerUi**

**public class BookUi** – тип данных книга

***{***

***int BookId*** – уникальный идентификатор книги

***string Author*** – автор книги

***string Name*** – название книги

***dateTime Year*** – год издания

***decimal Price*** – цена

***CurrencyUi Currency –*** валюта

***int Count*** – количество книг в наличии (за вычетом блокированных)

***}***

**public class PurchaseUi** – тип данных покупка

***{***

***int PurchaseId*** – уникальный идентификатор покупки

***dateTime Date*** дата покупки

***BuyerUi*** ***Buyer –*** покупатель книги

***BookUi Book*** – ид книги

***int Count*** – количество купленных/покупаемых экземпляров книги

***decimal Amount*** – сумма покупки

***CurrencyUi Currency –*** валюта для суммы

***bool IsGetMoney*** – флаг получения оплаты

***bool IsTransferComplete*** – флаг передачи книги покупателю

***}***

**public class BuyerUi** – тип данных покупатель

***{***

***int BuyerId*** – уникальный идентификатор покупателя

***string FullName*** – полное имя покупателя

***string Address*** – адрес покупателя(может содержать самые разные данные типа почтового индекса, адреса) на основании поля возможно указать место для доставки

***string Phone*** – телефон покупателя

***IList<PurchaseUi> ListPurchases*** – список покупок пользователя завершенных и нет

***}***

**public class *CurrencyUi*** – тип данных валюта

***{***

***int CurrencyId*** – уникальный идентификатор валюты

***string CurrencyCode –*** строковыйкод валюты по ISO

***}***

## Логика работы

***списокКниг ПоказатьВсеКниги()***- показ всего списка книг с их ценами(учитывается ценовая политика для конкретной страны) и выбранная пользователем валюта. В количестве книг показывается значение равное “количество книг в наличии” – “количество заблокированных книг”.

***списокПокупок ПоказатьВсеПокупки()***- показ всего списка покупок

***списокПокупателей ПоказатьВсехПокупателей()***- показ всего списка покупателей

***списокКниг Найти(строка запроса(Автор или название книги))***- показ списка найденных совпадений

***списокКниг ПоказатьСписокТекущихПокупок()***- возвращается список книг из tempBuyer

***списокКниг ПоказатьАрхивПокупок(BuyerId)***- возвращается список книг покупок для конкретного покупателя

***строкаОтображения ПоказатьПревью(идКниги)*** – подгрузка из файла отформатированного фрагмента книги

***флаг Купить(идКниги)***- формируется tempBuyer и ему в список покупок добавляется книга, проверяется что количество выбранных книг не превышает общего(в противном случае false)

***флаг Вернуть(идКниги)***- из списка книг tempBuyer удаляется книга с указанным ид

***флаг ПодтверждениеЗаказа()***- возвращает флаг успешности/ошибки принятия заказа, tempBuyer добавляется в хранилище данных, выбранные книги помечаются флагом блокировки, и не отображаются пользователю. Заново проверяется, что наличие и количество выбранных книг не превышает доступного значения.

Для каждой книги формируется покупка(**Purchase**), и записывается в базу.

Необходимо учесть что доставка книги на бумаге невозможна на электронный адрес.

***флаг ОтменаПокупки(идПокупки)*** – отмена конкретной книги(ее покупки) при отказе пользователя от покупки после подтверждения заказа.

***флаг ПокупкаЗавершена(идПокупки)*** – если IsGetMoney и IsTransferComplete равны true – то покупка завершена

***флаг ИзменитьВалютуКниг(идВалюты)*** – пересчет цен книг в зависимости от выбранной валюты(***идВалюты***) и значения конверсии + вызов метода показа всех книг, false – не удалось изменить валюту

***списокКниг ПрименитьПромокод(строкаСкодом)*** – возвращается список книг из tempBuyer с учетом промокода, если он был подтвержден. Изменяется цена каждой книги из заказа.

***public IBooksShopLogic***

***{***

***IList<BookUi> ShowAllBooks();***

***IList<BookUi> FindBooks(string);***

***bool Buy(BookId);***

***bool Reversal(BookId);***

***IList<BookUi> ChangeCurrencyBook(idCurrency);***

***}***

***public IWorkWithOrder***

***{***

***IList<BookUi> ShowListCurrentPurchases();***

***bool ConfirmationOrder();***

***bool CancelPurchase(PurchaseId);***

***bool IsCompleted(PurchaseId);***

***IList<BookUi> ApplyPromocode(string);***

***}***

***public IPreview***

***{***

***string ShowPreview(BookId);***

***}***

***public IPromoCode***

***{***

***bool ConsiderPromoCode(ref decimal);***

***}***

## Модуль загрузки данных из хранилища

***– может быть отдельной библиотекой***

- реализация загрузки данных для модели данных из хранилища(реализация интерфейса ***IDataRepository*** из ядра)в виде конкретных классов ***BookSqlRepository, PurchaseSqlRepository, BuyerSqlRepository***

***internal Interface IDataRepository<T>***

***{***

***T ReadElement(index)*** – чтение элемента из хранилища по индексу

***IEnumerable<T> ReadListElements()*** – чтение всех элементов из хранилища

***Create(T)***- создание элемента

***Update(T)***- обновление элемента

***Delete(index)***- удаление элемента по индексу

***}***

***BookSqlRepository:IDataRepository<BookData>***

***PurchaseSqlRepository:IDataRepository<PurchaseData>***

***BuyerSqlRepository:IDataRepository<BuyerData>***

***..//..***

***internal class BookSqlRepository:IDataRepository<BookData>***

***{***

***private BookStoreContext db;***

***public BookSqlRepository(BookStoreContext context)***

***{***

***this.db=context;***

***}***

***..//..***

***}***

***internal BookStoreContext:DbContext***

***{***

***public DbSet<BookData> Books;***

***public DbSet<PurchaseData> Purchases;***

***..//..***

***}***

***internal WorkWithDataStorage***

***{***

***private BookStoreContext db***

***private IDataRepository<BookData> Books***

***private IDataRepository<*PurchaseData*> Purchases***

***private IDataRepository<BuyerData> Buyers***

***public WorkWithDataStorage()***

***{***

***db = new BookStoreContext();***

***Books = new BookSqlRepository(db);***

***Purchases = new PurchaseSqlRepository(db);***

***..//..***

***}***

***}***

## Описание основного модуля(Ядра) системы - либа

- Модель данных(классы BookData, PurchaseData, BuyerData)

- Модуль логики(не содержит данных, только методы выполняющие логическую работу с сущностями)

- Модуль работы с данными в хранилище

- Интерфейс подключения к данным(***IDataRepository***)

- механизм начального получения данных

- Основной модуль координации всех частей в виде класса BooksShop(возможно стоит объединить с модулем логики)

***public class BooksShop: IBooksShopLogic, IWorkWithOrder***

***{***

***private WorkWithDataStorage DataStorage;***

***private BuyerUi tempBuyer;***

***Public BooksShop(IPreview,IPromoCode)***

***{***

***DataStorage = new WorkWithDataStorage();***

***..//..  
}***

***Реализация логики***

***}***

На внешку из либы видна модель данных(BookUi,PurchaseUi, BuyerUi,CurrencyUi), интерфейс логики и класс BookShop

## Контроллер взаимодействия с представлением

***class BooksShopController***

***{***

***IBooksShopLogic booksShop = new BooksShop();***

***public ActionResult Index()***

***{return View;}***

***..//..***

***}***

## Схема работы

***BooksShopController*** 

BookSqlRepository

IDataRepository

BuyerSqlRepository

**PurchaseUi**

CurrencyUi

PromocodeData

OrderData

ExchangeRatesData

PurchaseDataData

BuyerData

CurrencyData

PricePolicyDate

AuthorData

PreviewData

BookData

IPromoCode

IPreview

IBooksShopLogic

BooksShop

BooksShopCore.dll

***IWorkWithOrder***

BuyerUi

BookUi

PurchaseSqlRepository

## Вопросы:

***где хранить фрагменты книг в текстовом виде(в формате удобном для чтения):***

– в файле и добавлять путь в сущность book. Каталог Books\Имя автора\Название книги(мб. ID???).xml

- в виде потока или строки в сущности Book.

Это вопрос загрузки информации – стоит сделать настраиваемым

***Как добавлять новые книги:***

- загружать при каждом запуске из определенного каталога в зависимости от папок(добавить процедуру обновления по запросу?)

- создавать админку?

- написать настроечный файл типа xml для сериализации сущности Book.

И UI приложение с помощью которого можно будет изменять содержимое настроечного файла вне сайта.

А на сайте повесить службу которая, к примеру, раз в час проверяет файл, если его размер изменился обновляет базу данных сайта и заставляет сайт обновить контент.

***Как определить что за пользователь?***

- предполагается что пользователь может выбирать книги в рамках одной сессии, если он по каким-то причинам выходит из сессии списки выбранных им книг сбрасываются. Как только пользователь выбрал нужные книги и нажал «подтвердить заказ» данные о нем и его заказе добавляются в список покупателей(сущность Buyer)

***Добавление цен для разных стран***

- фактически означает, кроме перевода валюты и изменения цены по конверсии – еще и изменение языка интерфейса?

Язык интерфейса всегда англик, цены книг могут отличаться в зависимости от страны покупателя

***Учет промо кодов***

- получается отдельная сущность промокод, добавить при покупке возможность ввода промокода, если значение совпадает – то скидка 10%. Значение для сравнения подгружается из базы, последнее заданное.

**Название книги на разных языках**

- предполагается что язык в названии книги совпадает с языком содержимого книги. Не вижу смысла переводить название содержимое книги от этого не изменится. Книги в переводе будут иметь новую позицию в таблице книг и соответственно название в переводе.

**Несколько авторов одной книги**

- создавать отдельную таблицу авторы и оттуда брать их id для сущности Book?

Для каждого автора будет свой список книг